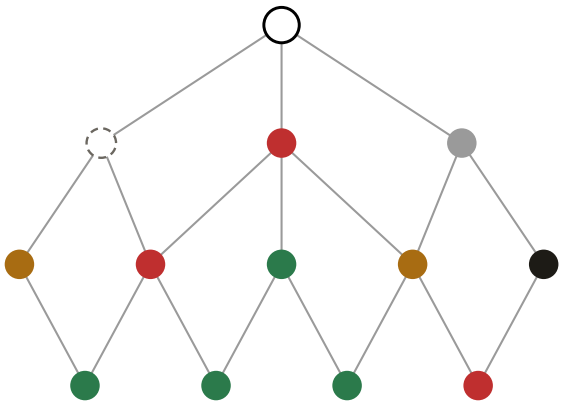
Mobility Matters

A Living Survey of Open Problems on Autonomous Agents, Robots and Particles

What is known, what is tight, and what is open, under which assumptions.



Debasish Pattanayak

Last updated 11 November 2026

AI USE Claude (Anthropic) was used in cataloguing the problems.

Abstract

Mobility Matters is a living survey of open problems in distributed computing by mobile entities: agents moving on graphs, robots in the plane, robots searching geometric domains, and programmable matter. The settings of each problem form a directed acyclic graph: a setting is a set of assumptions (timing, faults, communication, symmetry breaking, knowledge, and so on), the setting that assumes nothing comes first, and each edge leads to a setting that assumes more, so that algorithms carry over to stronger settings and lower bounds to weaker ones. For every setting the survey records the proven lower bounds and what forces them, the paper that introduced the setting, the current best result, the gap between them, and the open questions, each checked against the cited papers. Every assumption comes with its origin and formal definition, every problem with a plain-language first look, and essays connect the problems across models. The survey is maintained as a website and regenerated as a book from the same data.

Keywords: mobile agents; mobile robots; programmable matter; distributed computing; open problems; survey.

How to cite

Debasish Pattanayak. *Mobility Matters: A Living Survey of Open Problems on Autonomous Agents, Robots and Particles.* Version 2026-11-11, 2026. https://debasishpattanayak.com/mobility-matters/

BibTeX

```
@misc{Pattanayak2026,
  author = {Debasish Pattanayak},
  title  = {Mobility Matters: A Living Survey of Open Problems
            on Autonomous Agents, Robots and Particles},
  year   = {2026},
  note   = {Version 2026-11-11},
  url    = {https://debasishpattanayak.com/mobility-matters/},
}
```

This is a living survey: the website and this book are generated from the same data, and each version is dated. The latest version is at https://debasishpattanayak.com/mobility-matters/.

Contents

Acknowledgements v

How to read this atlas 1

I Models and their assumptions 2

1 Notation 4

2 Mobile agents on graphs 5

2.1 How the assumptions fit together . . . 5

2.2 Notation . . . 7

2.3 Assumption axes . . . 10

3 Mobile robots in the plane 17

3.1 How the assumptions fit together . . . 17

3.2 Notation . . . 19

3.3 Assumption axes . . . 19

Which model is stronger . . . 26

4 Search in geometric domains 38

4.1 How the assumptions fit together . . . 38

4.2 Notation . . . 39

4.3 Assumption axes . . . 40

5 Programmable matter 42

5.1 How the assumptions fit together . . . 42

5.2 Notation . . . 43

5.3 Assumption axes . . . 44

II Perspectives 48

6 Why compute by moving 50

7 Symmetry is the adversary 53

8 Finding your way 56

9 From exploration to dispersion, and beyond 59

10 Meeting at one place 62

11 Weak robots in the plane 65

12 From robots to particles 69

13 From the cow path to evacuation 72

14 When agents fail 74

15 What forces the bounds 77

III Mobile agents on graphs 80

16 Exploration 82

16.1 Graph exploration by a single agent . . . 82
16.2 Treasure hunt . . . 107
16.3 Rendezvous . . . 116
16.4 Gathering in graphs . . . 131
16.5 Gathering by oblivious robots on graphs . . . 139
16.6 Map construction . . . 155
16.7 Exploration by oblivious robots . . . 163
16.8 Exploration by luminous robots . . . 174
16.9 Exploration of recolourable graphs . . . 205
16.10 Online exploration of weighted graphs . . . 222
16.11 Collision-free exploration . . . 232
16.12 Collaborative exploration . . . 244
16.13 Dispersion . . . 258
16.14 Black hole search . . . 288
16.15 Exploration of dynamic graphs by teams . . . 305
16.16 Uniform deployment on graphs . . . 310
16.17 Network decontamination . . . 316
16.18 Broadcast by agents in dynamic graphs . . . 323
16.19 Delivery by energy-constrained agents . . . 331

Claude (Anthropic) was used in cataloguing the problems.

17 Computing with agents **340**

17.1 Leader election by agents . . . 340
17.2 Maximal independent set by agents . . . 345
17.3 Minimal dominating set by agents . . . 351
17.4 Triangle and butterfly counting by agents . . . 354
17.5 Minimum spanning tree by agents . . . 358
17.6 BFS tree by agents . . . 363
17.7 Deciding graph properties by mobile agents . . . 366

IV Mobile robots in the plane **377**

18 Formation problems **379**

18.1 Pattern formation . . . 379
18.2 Circle formation . . . 393
18.3 Uniform circle formation . . . 397
18.4 Universal pattern formation . . . 404
18.5 Pattern sequences . . . 407
18.6 Mutual visibility . . . 412
18.7 k-grouping . . . 424
18.8 Gathering . . . 429
18.9 Near-gathering . . . 460
18.10 Partitioning . . . 464
18.11 Convergence . . . 467

19 Symmetry breaking and spreading **473**

19.1 Scattering . . . 473
19.2 Leader election in the plane . . . 480
19.3 Separation . . . 483
19.4 Flocking . . . 487
19.5 Uniform deployment on a ring . . . 489

V Search in geometric domains **494**

20 Search, evacuation and patrolling **496**

20.1 Linear search (cow path) . . . 496
20.2 Evacuation . . . 505
20.3 Chauffeuring a crashed robot . . . 540
20.4 Search for a moving target . . . 544

Claude (Anthropic) was used in cataloguing the problems.

20.5 Patrolling . . . 566
20.6 Search with segment visibility . . . 574

VI Programmable matter **581**

21 Self-organising particle systems **583**
21.1 Leader election in programmable matter . . . 583
21.2 Shape formation in programmable matter . . . 598
21.3 Coating . . . 604
21.4 Compression . . . 608
21.5 Separation of coloured particles . . . 611
21.6 Alignment . . . 614
21.7 Bridging . . . 617
21.8 Convex hull formation . . . 619
21.9 Scattering of particles . . . 622
21.10 Energy distribution . . . 627
21.11 Shape formation by hybrid programmable matter . . . 630
21.12 Centralised transformations of modular robots . . . 634

Appendices **640**

A All open questions **641**

B Index of assumptions **668**

C Timeline **691**

Bibliography **713**

Claude (Anthropic) was used in cataloguing the problems.

Acknowledgements

The atlas is maintained by Debasish Pattanayak. It grows with help from readers: everyone whose correction, result or suggested problem is accepted through a GitHub issue is listed here, with the issue that brought it in.

Tools and data

- **Claude (Anthropic)**: AI assistance in searching the literature, reading papers, cataloguing results, drafting text and building the site and the book. Every entry marked verified was checked against the full text of the cited paper; the maintainer reviews the content.
- **Crossref, DataCite, arXiv and OpenAlex**: bibliographic metadata and open access to papers
- **KaTeX and Marked**: mathematics and text rendering on the website
- **LuaLaTeX and the Libertinus fonts**: typesetting of the book

The atlas rests on the work of the authors of the papers it cites; the bibliography lists them all.

How to contribute

Corrections, new results and missing problems are welcome as GitHub issues at `https://github.com/drdebmath/mobility-matters/issues`; every submission is checked against the full text of the paper, and accepted contributions are acknowledged here.

How to read this atlas

The problem space is a **directed acyclic graph**. Independent assumptions, one per axis (the graph or domain, the scheduler, what the entities see, remember and share, the faults), come together to define a problem, and each particular setting appears as a leaf of the graph: a problem solved under one set of assumptions. Results move along the edges: an algorithm for weaker assumptions works under stronger ones, and a lower bound for stronger assumptions holds under weaker ones. Assumption sets also change over time: later work minimises the set a result needs, a stronger assumption subsumes weaker ones, and some assumptions become irrelevant in the presence of better ones. Every setting records the proven lower bounds and what forces them, the paper that introduced it, the current best result, and the gap between them.

419	**319**	**26**	**74**	**272**	**296**
settings	tight bounds	near-optimal	polynomial gaps	unknown	open questions

Status of a bound. Each measure of each variant carries a status, judged against the best proven lower bound for that variant: ● *tight*; ● *near-optimal*; ● *polynomial gap*; ● *unknown*; ● *impossible*. Within a variant, results are listed in order of publication and marked [introductory], [current best], [also], [disputed proof].

Verification. A variant marked VERIFIED has its bounds checked against the papers; DRAFT marks entries with something not yet checked, which its notes name. A result or lower bound whose paper exists only as a preprint is marked [preprint]: it has not been peer reviewed.

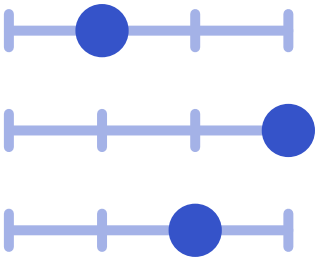
Citations. A paper is cited by its journal version, else by its full conference version, else by a brief announcement or short paper, else by its preprint. The bibliography lists the other versions with their links, the year of first appearance (which dates the paper in the timeline) and, where it differs, the version that was read.

Order of the book. Part I opens with the notation (Chapter 1), then introduces the four models and their assumption axes: where each assumption comes from, why the community studies it, and its formal definition. Part II reads across problems and models. The remaining parts give the problems, one part per model and one chapter per family, each problem with its statement, assumption tree and every catalogued setting. The appendices collect the open questions, starting with the most consequential, an alphabetical index of every assumption with its definition, the timeline and the bibliography.

Two orders. The parts arrange problems logically, by model and family; Appendix C lists the same material chronologically.

Part I

Models and their assumptions



What an agent, a robot or a particle can see, remember and do, and what the adversary controls. Each chapter tells how the assumptions of one model fit together, then gives every axis with its origin and formal definitions.

Claude (Anthropic) was used in cataloguing the problems.

CHAPTER 1

Notation

Each model uses the symbols of its own literature, so that a bound reads the same here as in the papers it comes from. Inside one model a symbol has exactly one meaning. The same letter may mean different things in different models (for example n counts nodes on graphs and robots in the plane); essays that cross models say which model a symbol belongs to. Where a paper writes a different symbol for the same quantity, the table says so under *papers write*. Symbols that a single paper defines for its own result are defined where they are used and listed under that problem.

General conventions

Symbol	Meaning	Papers write
$O, \Omega, \Theta, o, \omega$	asymptotic notation, as n (or the stated parameter) grows; $\tilde{O}$ hides polylogarithmic factors	
log	logarithm to base 2 unless stated	
ε	an arbitrarily small positive real (a constant unless stated)	
κ	a positive constant whose value does not matter for the statement	
t	time: a round, or a point of time in an asynchronous execution	
round	in synchronous settings (FSYNC, SSYNC, synchronous agents), one step in which every activated entity performs one full cycle	
epoch	in asynchronous settings, an interval of time in which every entity completes at least one full cycle; it need not be the shortest such interval. The execution is divided into consecutive epochs, and a bound of X epochs holds for every such division. Programmable matter calls the same notion an *asynchronous round*	

The symbols of each model are listed at the start of its chapter; Appendix B indexes every assumption with its formal definition.

Chapter 2

Mobile agents on graphs

Memory-bounded agents move along the edges of a port-labelled graph, by default with anonymous nodes; what they can see, remember, say and rely on is fixed by its assumption axes.

What is optimised

Once a task is known to be solvable, the community has mostly asked for it to be **fast**: first the number of rounds (or epochs, or edge traversals), then the **memory per agent** in bits. Problems with hostile nodes or agents, such as black hole search, ask instead for the **smallest number of agents** that still succeeds.

The graph $G = (V, E)$ has n nodes, m edges, maximum degree Δ and diameter D. Nodes are **anonymous** (no identifiers) and, unless assumed otherwise, have **no memory** (no whiteboards). Edges are **port-labelled**: the edges incident to a node v carry distinct labels $1, \dots, \deg(v)$, and the two endpoints of an edge label it independently. An agent at v sees $\deg(v)$ and, after arriving through an edge, the port through which it entered.

There are k agents. Each has a bounded amount of **memory** (persistent across rounds, measured in bits) and runs the same algorithm. In every step an agent (i) exchanges information with whatever agents its communication model allows, (ii) computes, and (iii) either stays or moves through one port. Agents moving in opposite directions along the same edge do not see each other.

Agents know **nothing** about n, m, k, Δ or D unless a result says so.

How the axes work. Every axis is a scale of assumptions. Level 0 is *no assumption*: the environment may be as hostile as the axis allows (a changing, directed graph; asynchronous, anonymous, Byzantine agents that cannot talk). Each higher level adds one assumption. A problem variant is a point on all axes at once, and the catalogue arranges variants as a tree whose root assumes nothing and whose edges each add assumptions. The *standard setting* of the literature (static undirected graph, $k \leq n$, synchronous rounds, local communication, unique identifiers, no faults) sits several steps below the root.

The two standard cost measures are the number of rounds until the problem is solved (time) and the number of bits of memory per agent (memory).

2.1 How the assumptions fit together

Agents that move through a network

The model asks what a team of computing entities can achieve when the only way to act on a network is to walk along its edges. Pelc names the two motivating systems: autonomous robots that start in different places of a terrain or a labyrinth, and software agents, "mobile pieces of software that navigate in a computer network in order to perform maintenance of its components or to collect data distributed in nodes of the network" (Pelc 2019). Since an algorithm depends only on what the entities perceive, remember and how they move, the same model serves both, and there is no central monitor, so the questions belong to distributed computing (Pelc 2019). Later problems kept this double reading:

dispersion was motivated by self-driving electric cars looking for free charging stations with "car-to-car interactions" only (Augustine and Moses Jr. 2018), and black hole search by host attacks on mobile code, a site that destroys incoming agents without a trace (Di Luna et al. 2025b). Each axis of the model fixes one thing that the agents may or may not rely on: the network (dynamics, direction, topology, node identifiers, node memory), the team (count, identifiers, randomness, faults, communication), time, and the start.

Every axis is a freedom of the adversary

Level 0 on each axis is what the adversary controls when nothing is assumed. In the deterministic scenario "the initial positions of the agents are chosen by an adversary which models the worst-case situation" (Pelc 2019); under asynchrony the adversary also chooses when and how fast each move happens (Pelc 2019); in a dynamic network it chooses the edges of every round after seeing the states of the nodes (Kuhn et al. 2010); and with Byzantine agents it controls part of the team (Dieudonné et al. 2014). The remaining axes govern **symmetry**. Two identical agents that run the same deterministic algorithm in an oriented ring and start at the same time keep their distance forever (Pelc 2019): anonymous agents, anonymous nodes, no coins and an adversarial start leave the adversary free to keep the configuration symmetric. Each higher level removes one of these freedoms (labels, node names, coins, a common start, a stable or undirected graph, synchronous rounds, honest agents) and makes the problem easier. This is why results move along the scale: an algorithm proved against a stronger adversary holds against a weaker one, and an impossibility for a weaker adversary holds for a stronger one, as the inclusion hierarchy of time-varying graph classes already makes explicit for dynamics (Casteigts et al. 2012).

How the axes interact

The axes are independent as assumptions but not in their effect. **Identifiers and the start.** Anonymity matters only when the rest of the instance is symmetric: on a line with an odd number of nodes two identical agents meet at the centre, and from asymmetric positions they can still break symmetry (Pelc 2019); from a rooted start, where agents share their position, only labels or coins can decide which agent settles first, which is how Molla and Moses Jr. use randomness in dispersion (Molla and Moses Jr. 2019). **Faults and identifiers.** The two Byzantine levels differ only in whether a faulty agent may forge its label; with known n this is the difference between one good agent sufficing and needing $f + 1$, but with unknown n both need $f + 2$ (Dieudonné et al. 2014; Bouchard et al. 2016). **Faults and timing.** Under full asynchrony a waiting agent cannot tell a slow agent from a crashed one, so gathering with crash faults is impossible; Pelc restores it by fixing each agent's speed (Pelc 2018a). **Dynamics and timing.** In temporal graphs "the distributed computation is by definition synchronous" (Di Luna et al. 2025b), so dynamic settings in the atlas are synchronous. **Communication and the graph.** Global communication does not tell two agents on neighbouring nodes which edge joins them, so edges must still be traversed (Kshemkalyani et al. 2020b). **Node memory and node identifiers.** A whiteboard or a pebble lets an agent recognise a node it has seen, the service that node labels would give (Bender et al. 2002; Fraigniaud and Ilcinkas 2004); without either, settled agents play that role in dispersion (Kshemkalyani and Ali 2019). **Direction and exploration.** An undirected graph can be explored by depth-first search in fewer than $2|V|$ steps, while some directed graphs need $\Theta(|V|^2)$ (Erlebach et al. 2021), because backtracking along an edge is no longer available (Italiano et al. 2025).

Why the standard setting is where it is

The setting most papers start from is a static, undirected, connected graph with anonymous nodes and locally labelled ports, synchronous rounds, deterministic fault-free agents with distinct labels,

communication between co-located agents only, and no node memory. Each choice has a stated reason. Undirected edges capture movement in both directions along a link, and connectivity is necessary for agents in different components to meet at all (Pelc 2019). Anonymous nodes are assumed because labels may be withheld for privacy or security, may be unreadable by weak sensors, and because readable labels would reduce rendezvous to exploration; port numbers stay, because without them an agent may never reach all neighbours of a node of degree at least 3 (Pelc 2019). Distinct agent labels are the cheapest deterministic way to break symmetry (Pelc 2019; Dessmark et al. 2006); determinism is kept because probabilistic approaches "typically fail to provide absolute guarantees" (Devismes et al. 2026b). Local communication matches the physical picture of agents that meet (Augustine and Moses Jr. 2018). Synchronous rounds let agents use waiting as a tool (Pelc 2019). The research programme then removes these assumptions one at a time: asynchrony (De Marco et al. 2006; Kshemkalyani and Ali 2019), dynamic graphs (Di Luna et al. 2020b; Agarwalla et al. 2018), directed graphs (Italiano et al. 2025), Byzantine agents (Dieudonné et al. 2014; Molla et al. 2021b), silent agents (Gorain et al. 2024), and repeated labels (Devismes et al. 2026b). The level-0 corner of the model, where none of the assumptions hold, is where most open questions sit.

2.2 Notation

Symbols of this model. Inside the model each symbol has one meaning; *papers write* gives other symbols found in the literature.

Symbol	Meaning	Papers write	Used in
$G = (V, E)$	the graph; for a directed graph E is a set of arcs, written $G = (V, A)$ where both appear		whole model
n	number of nodes, $n = \|V\|$		whole model
m	number of edges (arcs)		whole model
Δ	maximum degree		whole model
Δ_{out}	maximum out-degree of a directed graph	d in some exploration papers	whole model
$\deg(v)$, $\deg_{out}(v)$	degree and out-degree of node v	$\delta(v)$, $\delta_{out}(v)$	whole model
D	diameter of the graph (for a tree explored from a root: its height)		whole model
tw	treewidth of the graph (of the underlying graph of a temporal graph)	k (Adamson and Spirakis 2026), w (Carnevale 2026)	Graph exploration by a single agent, Exploration of dynamic graphs by teams
u, v, w	nodes (local names)		whole model
s	the starting node of a single agent, the root of a rooted team, or the home base of black hole search and decontamination	h (home base), r (root)	whole model
k	number of agents	n for robots on graphs (Maiti et al. 2026e,d); k for the ignorant agents only, the source agent not counted, so their k is $k - 1$ here (Das et al. 2026b; Moses Jr. et al. 2025)	whole model

Claude (Anthropic) was used in cataloguing the problems.

Symbol	Meaning	Papers write	Used in
ℓ	number of initially occupied nodes; $\ell = 1$ is a rooted start, $\ell = k$ a dispersed start		Dispersion, Gathering by oblivious robots on graphs
f	maximum number of faulty agents (crashed or Byzantine)		whole model
N	a known upper bound on n	$\hat{n}$	whole model
$\mathscr{L}$	the set of agent labels in use; $\lvert\mathscr{L}\rvert$ distinct labels, $\lvert\mathscr{L}\rvert = 1$ anonymous, $\lvert\mathscr{L}\rvert = k$ distinct identifiers	ℓ (homonyms)	whole model
L	the largest possible label: labels are integers in $\{1, \dots, L\}$	λ (largest identifier), μ (largest label)	whole model
λ	the smallest label among the agents (for two agents: the smaller label)	l (Pelc 2019)	whole model
τ	delay between the starting times of two agents		whole model
d	distance from the starting node to the target (the treasure); otherwise the largest distance between two starting nodes (of the two agents in rendezvous, of two initially occupied nodes in gathering)	D for the distance between starting positions (Pelc 2019); $\Delta(C)$ (Cicerone et al. 2025d,b)	whole model
v^*	the node of the treasure	t	whole model
$e(d)$	number of edges with an endpoint at distance less than d from the start		Treasure hunt
$w(\cdot)$, W	weight of the cheapest path type to the treasure (treasure hunt); W its maximum over the two directions (rendezvous)		Treasure hunt, Rendezvous
h	communication range in hops (h-hop communication)	ℓ (Saxena et al. 2026a)	whole model
T	window length in T-interval and T-path connectivity of dynamic graphs		whole model
μ	maximum number of edges missing in a round (μ-bounded 1-interval connected graphs)	ℓ-bounded (Saxena et al. 2026a); k (k-edge-deficient temporal graphs) (Lahtin and Zamaraev 2027); f (Kaur et al. 2026b)	whole model
η	number of transient edges of a temporally connected graph		Exploration of dynamic graphs by teams
δ	longest time an edge of a dynamic graph may stay missing		Graph exploration by a single agent, Network decontamination
p	number of pebbles		whole model
b	number of bits of an agent's memory, when it is a parameter		whole model
β	number of bits of a label written at a node in advance		Graph exploration by a single agent
ϕ	deficiency of a directed graph: the number of arcs whose addition makes it Eulerian	d (Italiano et al. 2025)	Dispersion, Graph exploration by a single agent
ν	number of leaves of a tree	λ	Rendezvous
CR(k)	competitive ratio of a collaborative exploration algorithm with k agents		Collaborative exploration

Claude (Anthropic) was used in cataloguing the problems.

Symbol	Meaning	Papers write	Used in
$X(n)$	time to explore any n-node graph	$R(n)$ (Bhagat and Pelc 2025), $R(N)$ (Papin 2026)	Dispersion, Collision-free exploration, Gathering in graphs
port_v	port numbering at node v	λ_v	Graph exploration by a single agent
R	running time, in rounds, of a message-passing algorithm simulated by agents		Leader election by agents, Maximal independent set by agents, Minimal dominating set by agents, Minimum spanning tree by agents, BFS tree by agents, Triangle and butterfly counting by agents
K	number of states of a finite-automaton agent		Graph exploration by a single agent, Exploration of recolourable graphs
γ	exponent of the team size in large-team collaborative exploration, $k = Dn^{\gamma}$		Collaborative exploration
θ	the exponent of Reingold's universal exploration sequences		Treasure hunt
(q, ρ)	type of a path: its largest port q and its length ρ		Treasure hunt
z	number of nodes of non-zero capacity		Dispersion
r	radius of a ball of the grid		Dispersion
c	number of colours (of a robot's light, or of the colours a node can hold)	κ, ℓ	Exploration by luminous robots, Exploration of recolourable graphs
φ	visibility range of a myopic robot, in hops		Exploration by luminous robots
$a \times b$	size of a grid or torus with a rows and b columns		Exploration by luminous robots, Broadcast by agents in dynamic graphs, Gathering by oblivious robots on graphs
$\psi(v)$	current colour of node v	$c(v)$	Exploration of recolourable graphs
c_0	initial colour of the nodes		Exploration of recolourable graphs
π_v	rotor pointer at node v: the port the next departing agent takes		Exploration of recolourable graphs
$N(v)$	the neighbours of node v		Exploration of recolourable graphs
$\text{circ}(G)$	circumference of G: the length of its longest cycle	k	Exploration of recolourable graphs
g	a known upper bound on the circumference		Exploration of recolourable graphs

Claude (Anthropic) was used in cataloguing the problems.

Symbol	Meaning	Papers write	Used in
$\mathrm{wt}(e)$	weight (length) of edge e of a weighted graph	$w(e)$	Online exploration of weighted graphs, Delivery by energy-constrained agents
OPT	cost of a shortest closed walk through all nodes (an optimal travelling-salesperson tour)	$\mathrm{Opt}(G)$	Online exploration of weighted graphs
ϑ	largest ratio $\mathrm{wt}^+(e)/\mathrm{wt}^-(e)$ between the ends of the announced interval of an edge weight	α (Gehnen et al. 2025)	Online exploration of weighted graphs
Π	a decision problem: a set of configurations (graph, start nodes and inputs of the agents)		Deciding graph properties by mobile agents
B, B_i	energy budget: the largest total distance an agent may travel (B_i for agent i, B when all budgets are equal)	R_i (range) (Chalopin et al. 2013, 2014); P (power) (Anaya et al. 2016)	Delivery by energy-constrained agents
v_s, v_t	source and target node of the message in delivery	s, t	Delivery by energy-constrained agents

2.3 Assumption axes

Each axis says where the assumption comes from and why the community studies it, then defines its values. Every axis is a scale. Level 0 assumes nothing; each higher level is a stronger assumption, which makes a problem easier. Values on the same level are alternatives that do not imply each other. An algorithm designed for weaker assumptions works under stronger ones; a lower bound proved under stronger assumptions holds under weaker ones.

Graph dynamics

Graphs that change all the time entered distributed computing as models of mobile, wireless and peer-to-peer networks. Kuhn, Lynch and Oshman introduced T-interval connectivity in a worst-case model where an adversary picks the edges of each round; "for $T = 1$ this means that the graph is connected in every round, but changes arbitrarily between rounds" (Kuhn et al. 2010). Casteigts, Flocchini, Quattrociocchi and Santoro unified the many models of time-varying graphs into a hierarchy of classes: the always-connected class, used earlier by O'Dell and Wattenhofer (as reported there), and recurrent connectivity, in which journeys exist from every time onward, which is level 0 here (Casteigts et al. 2012). They put the reason for the scale in one sentence: in these systems "changes are not anomalies but rather integral part of the nature of the system" (Casteigts et al. 2012). Kuhn et al. show the adversary's strength with a token that must keep moving: in a 1-interval connected graph the adversary can keep it between two nodes forever (Kuhn et al. 2010), which is in effect a single agent that cannot explore. Mobile agents entered the setting with live exploration of 1-interval connected rings (Di Luna et al. 2020b) and dispersion in dynamic rings (Agarwalla et al. 2018); offline exploration of temporal graphs is the centralised counterpart (Erlebach et al. 2021).

Level	Value	Formal definition
0	**any dynamics**	A **temporal graph** is a sequence $\mathcal{G} = (G_t)_{t\geq 0}$ with $G_t = (V, E_t)$ on a fixed node set. It is *temporally connected* if for all $u, v \in V$ and every time t there is a journey (a path whose edges appear at non-decreasing times $\geq t$) from u to v. No G_t need be connected.
1	**connected every round**	$\mathcal{G} = (G_t)_{t\geq 0}$ is **1-interval connected** if every snapshot $G_t = (V, E_t)$ is connected. More generally it is T-interval connected if for every t the graph $\left(V, \bigcap_{i=t}^{t+T-1} E_i\right)$ is connected.
2	**static graph**	$G_t = G$ for all t: the edge set never changes.

Edge direction

Most of the mobile-agent literature uses undirected graphs because this "captures the ability of the agents to move in both directions along each link" (Pelc 2019). Directed graphs come from the exploration literature. The earliest paper in the atlas on exploring an unknown directed graph is Deng and Papadimitriou's (Deng and Papadimitriou 1999), continued by Albers and Henzinger (Albers and Henzinger 2000). Bender, Fernández, Ron, Sahai and Vadhan asked what a robot needs to map an unlabelled strongly connected digraph, and showed that one pebble suffices when a bound on n is known (Bender et al. 2002); Fraigniaud and Ilcinkas studied how little memory digraph exploration needs (Fraigniaud and Ilcinkas 2004). Direction removes backtracking: a static undirected graph is explored by depth-first search in fewer than $2|V|$ steps, while some static directed graphs need $\Theta(|V|^2)$ steps (Erlebach et al. 2021). Italiano, Pattanayak and Sharma posed dispersion on directed anonymous graphs because "the existing approaches for undirected graphs rely substantially on visiting a graph edge in both directions", and showed that without strong connectivity dispersion may be unsolvable (Italiano et al. 2025).

Level	Value	Formal definition
0	**directed**	$G = (V, A)$ with $A \subseteq V \times V$; an agent at u may move along $(u, v) \in A$ only, never back along it unless $(v, u) \in A$. G is strongly connected.
1	**undirected**	$G = (V, E)$ with $E \subseteq \binom{V}{2}$; every edge can be traversed in both directions.

Number of agents

Most tasks in this model are posed for a team of any size: gathering an unknown team of anonymous agents (Dieudonné and Pelc 2016), black hole search, where the size of the team is the main measure (Dobrev et al. 2006), and collective exploration by k agents (Fraigniaud et al. 2006a). The cap $k \leq n$ comes from dispersion, where every agent needs a node of its own. Augustine and Moses Jr. posed dispersion with $k = n$, motivated by electric cars that should not queue at one charging station (Augustine and Moses Jr. 2018), and Kshemkalyani and Ali relaxed it to $k \leq n$ (Kshemkalyani and Ali 2019). Molla and Moses Jr. extended dispersion to any k, asking for at most $\lceil k/n \rceil$ agents per node (Molla and Moses Jr. 2019). The extension is not free: with $k > n$ Byzantine dispersion is impossible deterministically whenever $\lceil k/n \rceil > \lceil (k - f)/n \rceil$ (Molla et al. 2021b). The axis thus separates tasks where the team must fit into the graph from tasks where the team size is unconstrained.

Level	Value	Formal definition
0	**any** k	The number of agents $k \geq 1$ is arbitrary; $k > n$ is allowed.
1	$k \leq n$	The number of agents satisfies $1 \leq k \leq n$.

Identifiers

Pelc's survey names anonymous versus labelled agents as one of the two main pairs of assumptions in rendezvous, and gives the reason: two identical agents running the same deterministic algorithm in an oriented ring, starting together, keep the same distance forever (Pelc 2019). Distinct labels break this symmetry: in the work of Dessmark, Fraigniaud, Kowalski and Pelc each agent knows only its own label and uses it as a parameter of the common algorithm, and rendezvous becomes feasible in every graph at cost polynomial in n, the delay and $\log \lambda$, λ the smaller label; without labels it fails already on two nodes joined by an edge, since both agents always stay or always move together (Dessmark et al. 2006). Without labels the only asymmetry comes from the graph and the starting positions; Dieudonné and Pelc characterised the configurations of anonymous agents that can be gathered (Dieudonné and Pelc 2016). Homonyms, several agents sharing a label, come from Byzantine agreement with homonyms (Delporte-Gallet et al. 2013) and were taken to robots in the plane (Bouzid and Lamani 2011). In graphs, Devismes, Dieudonné and Labourel motivate them by the cost of "generating and maintaining globally unique labels" and by privacy, and ask how many distinct labels gathering really needs (Devismes et al. 2026b).

Level	Value	Formal definition
0	**anonymous**	All agents run the same deterministic algorithm on the same input; no agent has a label. Formally the label function is constant: $\mathrm{id}(a) = \mathrm{id}(b)$ for all agents a, b.
1	**homonymous**	Each agent a has a label $\mathrm{id}(a) \in \mathscr{L}$ known to itself (and revealed on meeting), but the labelling need not be injective: $1 \le \lvert\{\mathrm{id}(a)\}\rvert \le k$. With $\lvert\mathscr{L}\rvert$ distinct labels, $\lvert\mathscr{L}\rvert = 1$ is the anonymous case and $\lvert\mathscr{L}\rvert = k$ the eponymous one (Delporte-Gallet et al. 2013).
2	**distinct IDs (eponymous)**	**Distinct identifiers** (eponymous agents): the label function $\mathrm{id} : \{1, \dots, k\} \to \{1, \dots, L\}$ is injective; usually $L = k^{O(1)}$, so a label takes $O(\log k)$ bits.

Randomness

Pelc counts determinism versus randomisation among the two assumptions that most shape the methods for rendezvous: a deterministic algorithm faces adversarial starting positions and is judged by its worst-case cost, while randomised algorithms often use random walks and are judged by expected cost (Pelc 2019). The random-walk view of graph traversal goes back to Aleliunas et al., who connected random walks with universal traversal sequences (Aleliunas et al. 1979). In this model randomness serves mainly to break symmetry: Molla and Moses Jr. let co-located anonymous agents elect a local leader by coin flips, so that dispersion needs $O(\log \Delta)$ bits per agent instead of the $\Omega(\log n)$ bits of deterministic algorithms (Molla and Moses Jr. 2019). Deterministic algorithms are level 0 because they promise a bound for every execution; Devismes, Dieudonné and Labourel note that probabilistic approaches "typically fail to provide absolute guarantees in terms of complexity and/or accuracy" (Devismes et al. 2026b). Hence a deterministic algorithm also counts in the randomized setting, and a lower bound against randomized algorithms holds for deterministic ones.

Level	Value	Formal definition
0	**deterministic**	The next state and move of an agent are a function of its current state and what it observes.
1	**private coins**	Each agent additionally has access to independent fair coin flips; guarantees hold with probability 1, in expectation, or with high probability, as stated per result.

Faults

The first faults studied for mobile agents sat in the network: Dobrev, Flocchini, Prencipe and Santoro introduced the black hole, a node that destroys every agent that enters it, in rings and then in arbitrary networks (Dobrev et al. 2007, 2006); it models a host that trashes incoming code or crashes without notice (Di Luna et al. 2025b). Faults in the agents, which this axis records, came later. Dieudonné, Pelc and Peleg started the study of gathering with Byzantine agents and defined the two levels used here: a strongly Byzantine agent "can choose an arbitrary port when it moves and it can convey arbitrary information to other agents, while a weakly Byzantine agent can do the same, except changing its label" (Dieudonné et al. 2014). The distinction is about identity and it changes the answer: with known n a single good agent gathers despite f weakly Byzantine agents, but $f+1$ good agents are needed against strongly Byzantine ones, even in a ring of known size, while with unknown n both cases need $f+2$ (Dieudonné et al. 2014); Dieudonné, Pelc and Peleg left the strongly Byzantine upper bounds at $2f+1$ (known n) and $4f+2$ (unknown n), and Bouchard, Dieudonné and Ducourthial closed them (Bouchard et al. 2016; Pelc 2019). Crash faults are the mildest level; Pelc showed that with fully asynchronous agents gathering with crashes is impossible, since a waiting agent cannot tell a slow agent from a crashed one, and solved it when each agent has a fixed speed (Pelc 2018a). Dispersion adopted both Byzantine levels (Molla et al. 2021b) and crash faults (Chand et al. 2025a).

Level	Value	Formal definition
0	**any faults**	Up to f agents behave arbitrarily (adversarially) and may **forge** identifiers, i.e. report any label in $\mathscr{L}$.
1	**at most weak Byzantine faults**	Up to f agents behave arbitrarily but always report their own true identifier.
2	**at most crash faults**	Up to f agents may stop forever at an adversarially chosen time; a crashed agent stays at its node (or vanishes, as stated per result) and never acts again.
3	**no faults**	All agents follow the algorithm forever.

Communication

Communication between co-located agents is the classical assumption of rendezvous and gathering, where agents that meet "can exchange all information they currently have" (Pelc 2019). Augustine and Moses Jr. kept it for dispersion, assuming robots "can only communicate with other robots present on their own node", in line with their picture of cars that interact only car to car (Augustine and Moses Jr. 2018). Di Luna, Dobrev, Flocchini and Santoro list the forms used in exploration: face-to-face, wireless (agents on different nodes), and whiteboards (Di Luna et al. 2020b). Kshemkalyani, Molla and Sharma introduced global communication to dispersion to measure "how much global communication helps", and observed that it does not tell two agents on neighbouring nodes which edge joins them, so the graph must still be walked (Kshemkalyani et al. 2020b). For collective tree exploration Fraigniaud, Gąsieniec, Kowalski and Pelc showed that without communication k agents gain nothing over one (Fraigniaud et al. 2006a). Between the two, Italiano, Pattanayak and Sharma studied a **1-hop model**, in which "a robot may also communicate with robots positioned on its 1-hop neighbors", and used it to disperse on directed graphs in $O(kD)$ rounds by broadcasting and gathering information along the graph (Italiano et al. 2025). Saxena, Kaur and Mondal generalised it to **ℓ-hop communication** (h-hop here), "a spectrum between f-2-f ($l=0$) and global communication ($l=D$)", and showed that for balanced dispersion on 1-bounded 1-interval connected graphs agents need 1-hop visibility and 1-hop communication (Saxena et al. 2026a); the axis writes h for their ℓ, which the atlas uses for occupied nodes. At level 0, Gorain, Mandal, Mondal and Pandit asked whether talking is needed at all, and dispersed agents that only learn whether they are alone and whether the number of agents at their node changed (Gorain et al. 2024).

Level	Value	Formal definition
0	**no communication**	Agents exchange no messages; an agent at v observes only the number of agents at v (and, where stated, whether that number changed).
1	**local communication**	Agents at the same node at the same time can read each other's entire memory (face-to-face communication). Agents crossing on an edge do not meet.
2	**h-hop communication**	For a fixed $h \geq 1$, stated with each result: at every round an agent at v can read the memory of every agent at a node u with $\mathrm{dist}(v, u) \leq h$ (in the current snapshot G_t for a dynamic graph), in addition to the co-located agents. $h = 1$ is **1-hop communication** (agents at neighbouring nodes communicate); local communication is the case $h = 0$, and global communication the case $h \geq D$. Reading another agent's memory does not tell an agent which port leads to it.
3	**global communication**	At every round every agent can read the memory of every other agent, wherever it is.

Timing

Pelc names synchronous versus asynchronous movement as the second main pair of assumptions for rendezvous: synchronous agents can alternate measured periods of activity and waiting, "a powerful tool", while asynchronous agents move at speeds the adversary varies, so algorithms must force trajectories to overlap (Pelc 2019). The asynchronous model for agents in graphs was introduced by De Marco, Gargano, Kranakis, Krizanc, Pelc and Vaccaro for rendezvous; since a meeting at a node cannot be guaranteed even in the two-node graph, agents may also meet inside an edge (De Marco et al. 2006) (as described in Pelc 2019). Black hole search was posed for asynchronous agents with whiteboards (Dobrev et al. 2006), while dispersion started synchronous and Kshemkalyani and Ali gave its first asynchronous algorithms (Kshemkalyani and Ali 2019). Intermediate schedulers exist: Di Luna et al. brought the semi-synchronous model, "common in the context of mobile agents in continuous spaces", to agents in graphs (Di Luna et al. 2020b). Asynchrony interacts with other axes: it makes gathering with crash faults impossible (Pelc 2018a), and dynamic graphs are synchronous by definition (Di Luna et al. 2025b).

Level	Value	Formal definition
0	**asynchronous**	Each agent's Communicate–Compute–Move steps take finite but unbounded, adversarially chosen time. Time is measured in epochs. An **epoch** is an interval of time in which every agent completes at least one full Communicate–Compute–Move step; it need not be the shortest such interval. Time is counted by dividing the execution into consecutive epochs, and a bound of X epochs holds for every such division.
1	**synchronous**	Time proceeds in rounds $t = 1, 2, \ldots$; in every round every agent communicates, computes and moves (at most one edge) simultaneously.

Node memory

Whiteboards are the shared memory of the mobile-agent model: each node keeps a little storage that visiting agents read and write in mutual exclusion, "where the agents communicate by writing on and reading from whiteboards present in each node" (Di Luna et al. 2020b). They are the standard tool in black hole search, posed for asynchronous agents with whiteboards (Dobrev et al. 2006), and in map

construction, where identical agents with whiteboards build a map exactly when symmetry can be broken (Das et al. 2007). Their purpose is marking: an agent that can leave information at a node can recognise it later, which is what node labels would otherwise provide (Bender et al. 2002). The cheaper markers are pebbles or tokens, and black hole search showed that whiteboards can be replaced by them (Dobrev et al. 2013; Flocchini et al. 2012b); for digraphs, whiteboards of $O(\log \Delta_{out})$ bits let an agent with constant memory explore (Fraigniaud and Ilcinkas 2004). Pelc counts the ability to see marks among the assumptions that most change methods, since it "significantly facilitates" rendezvous (Pelc 2019). Level 0 has no node memory, as in dispersion, where settled agents serve as markers instead (Kshemkalyani and Ali 2019).

Level	Value	Formal definition
0	**no whiteboards**	Nodes have no storage: no information can be left at a node except by an agent staying there.
1	**whiteboards**	Each node v has a whiteboard of bounded size that agents at v can read and write, with mutual exclusion.

Node identifiers

Anonymous nodes with locally labelled ports are the standard network of this model, and Pelc gives the reasons. Practically, nodes may withhold their labels "for privacy or security reasons", or the agents' sensors may be too weak to read them; methodologically, readable labels would let agents meet at the smallest label, reducing rendezvous to exploration (Pelc 2019). Port numbers are kept because without them an agent may be unable to visit all neighbours of a node of degree at least 3, and because reading a port needs far less than reading a name (Pelc 2019). The same convention underlies memory-bounded exploration, where a finite automaton explores a graph "whose nodes are unlabeled and whose edge ports are locally labeled" (Fraigniaud et al. 2005a). Labelled nodes belong to the online-exploration tradition, in which the agent recognises nodes and builds a map (Deng and Papadimitriou 1999; Albers and Henzinger 2000), and to collective exploration, where nodes are typically labelled (Augustine and Moses Jr. 2018; Fraigniaud et al. 2006a). With labelled nodes many symmetry questions disappear, which is why the axis separates navigation from symmetry breaking.

Level	Value	Formal definition
0	**anonymous nodes**	Nodes have no identifiers; an agent at v sees only $\deg(v)$ and the port numbers $1, \dots, \deg(v)$, which are assigned locally and independently at each endpoint of an edge.
1	**labelled nodes**	Each node v carries a distinct identifier that agents at v can read; hence an agent can recognise a node it has visited.

Topology

Special topologies enter for two reasons: they are the simplest graphs with a given feature, and they are networks that are used. Trees can be explored with termination by a single agent and have a central node or edge that acts as a focal point, so rendezvous was first solved there; rings are "the simplest class" of graphs with cycles (Pelc 2019; Dessmark et al. 2006). The oriented ring is also the most symmetric instance, where identical agents cannot meet (Pelc 2019). The ring, oriented or not, was the first topology for black hole search (Dobrev et al. 2007), and the oriented ring was the first for live exploration of dynamic graphs (Di Luna et al. 2020b) and, as reported by (Molla et al. 2021b), for Byzantine dispersion. Grids model discretised space: the infinite grid is a tool for rendezvous in the plane (Pelc 2019), and dispersion on grids is studied for its own sake (Kshemkalyani et al. 2020c). Hypercubes are interconnection networks on which decontamination was solved before unknown graphs (Flocchini et al. 2008c); the complete graph K_3 is the witness in Pelc's crash impossibility (Pelc 2018a); bipartite

graphs appear with butterfly counting by agents (Chand et al. 2025b). Level 0 is an arbitrary connected graph, unknown to the agents.

Level	Value	Formal definition
0	**any topology**	G is any connected graph in the class allowed by the other axes.
1	**ring**	G is a cycle C_n (every node of degree 2), or a path when stated.
1	**tree**	G is connected and acyclic: $\|E\| = n - 1$.
1	**grid**	G is a $\sqrt{n} \times \sqrt{n}$ grid graph; whether ports are globally consistent (oriented) is stated per result.
1	**complete graph**	$G = K_n$: every pair of nodes is adjacent.
1	**hypercube**	$G = Q_j$: nodes are the $n = 2^j$ binary strings of length j, adjacent when they differ in one bit.
1	**bipartite**	$V = V_1 \cup V_2$ with $V_1 \cap V_2 = \emptyset$ and every edge joining V_1 to V_2; trees are a special case.

Initial configuration

The starting configuration is the adversary's last choice. Rooted starts come from exploration, where the team leaves a common home: collective tree exploration places all k agents at the root (Fraigniaud et al. 2006a), black hole search starts from a safe home base (Dobrev et al. 2006), and Molla and Moses Jr. call this a rooted graph (Molla and Moses Jr. 2019). Dispersed starts come from rendezvous and gathering, where agents start at distinct nodes (Pelc 2019; Dieudonné and Pelc 2016), and from black hole search with scattered agents (Di Luna et al. 2025b). A dispersed configuration is also the output of dispersion, so tasks computed by agents after dispersing, such as leader election, start from it (Kshemkalyani et al. 2026; Sudo et al. 2026a). Dispersion itself was posed from arbitrary configurations, robots "placed in an n node graph arbitrarily" (Augustine and Moses Jr. 2018), and rooted dispersion became a separate variant (Molla and Moses Jr. 2019). The two special starts expose different symmetries: co-located agents are indistinguishable by position, so labels or coins must decide who moves (Molla and Moses Jr. 2019); scattered agents have distinct positions that may still look alike in a symmetric graph (Pelc 2019).

Level	Value	Formal definition
0	**any start**	Agents start at arbitrary nodes; $\ell \leq k$ nodes are initially occupied, possibly with several agents each.
1	**rooted start**	All k agents start at the same node ($\ell = 1$). Also called *co-located* (black hole search: all agents at the *home base*) or a *rooted graph* (Molla and Moses Jr. 2019).
1	**dispersed start**	Every initially occupied node holds exactly one agent ($\ell = k$). Also called a *scattered* start, notably in black hole search.

The problems of this model are in Part III: Exploration, Computing with agents.

Chapter 3

Mobile robots in the plane

Anonymous point robots in the Euclidean plane repeat Look–Compute–Move cycles; what they agree on, remember, see and rely on is fixed by its assumption axes.

What is optimised

The primary question has been **solvability under the fewest assumptions**: which combinations of synchrony, agreement on coordinates, rigidity, multiplicity detection, visibility and memory make a task possible at all. Time (in rounds or epochs) and the number of light colours became the measures later, once the feasibility frontier was mapped.

There are n robots, modelled as points in $\mathbb{R}^2$. They are **anonymous** (no identifiers), **identical** (same algorithm) and **silent** (no messages). Each robot repeats a cycle: **Look** (take a snapshot of the positions of the robots it can see, in its own local coordinate system), **Compute** (run the algorithm on the snapshot) and **Move** (move towards the computed point). Unless assumed otherwise robots are **oblivious**: nothing survives from one cycle to the next.

A **scheduler** decides when robots are active: in FSYNC all robots act in every round; in SSYNC an arbitrary non-empty subset acts in each round, atomically; in ASYNC cycles are arbitrary and a robot may look at others while they move. Time is counted in rounds (FSYNC, SSYNC) or epochs (ASYNC): intervals of time in which every robot completes at least one full cycle, not necessarily the shortest ones.

This is the OBLOT model of Suzuki and Yamashita and of Flocchini, Prencipe and Santoro; **lights are the robots' memory**: a light with a constant number of colours, set by its robot, persists between cycles. A robot may see only its own light (FSTA: finite state), only the others' lights (FCOM: finite communication), or all of them (LUMI); oblivious robots (OBLOT) have none. The number of colours is a measure.

How the axes work. As for graphs, level 0 on every axis assumes nothing, and higher levels are stronger assumptions. A problem variant is a point on all axes; the catalogue arranges variants as a tree rooted at *no assumptions* and as a graph of strengthenings.

3.1 How the assumptions fit together

Weak robots and the least that suffices

Suzuki and Yamashita (Suzuki and Yamashita 1999) modelled a swarm of identical robots that observe each other's positions at unpredictable times and "communicate" only by their moves, with no common coordinate system. Flocchini, Prencipe and Santoro describe the field that grew from this as asking for "the minimal capabilities that the robots should have in order to solve a problem" (Flocchini et al. 2012a), and later work states the goal as identifying "the minimal assumptions on internal capabilities of the robots (e.g., persistent memory, communication) and external conditions of the system (e.g., synchrony, activation scheduler)" (Flocchini et al. 2023). The robots are deliberately weak: anonymous, uniform, disoriented, oblivious and silent, the "weakest and de facto standard model" OBLOT (Flocchini et al. 2023). Each axis of this model is one capability that can be granted on top of that baseline (memory,

agreement, synchrony, detection, sight, identity, randomness), and a result is read as a point in the product of these scales. Impossibilities such as gathering without multiplicity detection (Prencipe 2007) tell which capability is needed; algorithms tell which suffices.

The adversary acts in time, in space and in the frames

Correctness is proved against an adversary, and most axes describe what that adversary controls. The scheduler decides when robots act and, in ASYNC, how long each phase lasts, so that a robot may compute on an outdated snapshot (Prencipe 2007; Flocchini et al. 2023). Non-rigid moves let it decide where a move ends, subject only to a minimum distance δ (Prencipe 2007; Flocchini et al. 2016a). Disorientation lets it choose each robot's coordinate system, and in some models change it at every activation (Di Luna et al. 2014; Izumi et al. 2012). Faults let it control some robots outright (Agmon and Peleg 2006), and with randomization the question is what it may know about the coins (Efrima and Peleg 2009). Weakening one of these powers is the usual way to study an axis: SSYNC removes stale snapshots, rigid moves remove the choice of stopping point, a compass removes the choice of frame.

Symmetry is the obstacle

Anonymous, disoriented, deterministic robots in a symmetric configuration receive equivalent snapshots and move symmetrically, so symmetry can persist forever; the formability characterisations are stated in terms of the symmetry of the configuration (Suzuki and Yamashita 1999; Dieudonné et al. 2010). Several axes are ways to break it. Agreement on chirality or on axes removes reflections and rotations (Dieudonné et al. 2010; Flocchini et al. 2008a); labels distinguish robots (Bouzid and Lamani 2011; Asahiro and Yamashita 2025); random bits break ties with probability 1 (Dieudonné and Petit 2009; Bramas and Tixeuil 2016); a sequential scheduler activates one robot at a time and so elects a leader for free (Flocchini et al. 2026). For gathering, a multiplicity point is itself an asymmetry, which is why multiplicity detection lets $n \geq 3$ robots gather (Cieliebak et al. 2012) while two robots, which cannot create one, need memory, lights or compasses (Suzuki and Yamashita 1999; Izumi et al. 2012; Viglietta 2013).

Lights: memory and communication in one register

Persistent memory and explicit communication were both absent from the original model. The light, suggested by Peleg (Peleg 2005) and formalised by Das et al. (Das et al. 2016), gives each robot a constant number of colours that persist and are seen by all, and it removes the difference between SSYNC and ASYNC (Das et al. 2016). Showing only one's own light (FSTA) or only the others' (FCOM) separates remembering from communicating (Flocchini et al. 2016a), which turned the memory axis and the scheduler axis into a single landscape of model–scheduler pairs (Flocchini et al. 2019b, 2023). Lights also serve the geometric axes: with obstructed visibility they let robots coordinate without collisions (Di Luna et al. 2014), and with two robots they replace persistent memory (Viglietta 2013; Heriban et al. 2018).

Where the standard setting sits

The usual setting is weak on the axes that concern computability (anonymous, oblivious, disoriented, deterministic, SSYNC or ASYNC, often non-rigid) and idealised on the geometric ones: point robots, transparent, with unlimited visibility and exact sensing (Flocchini et al. 2023; Bandettini et al. 2011). The idealisations were kept because they make the questions about synchrony, memory and agreement clean, and they were later relaxed one at a time: limited visibility (Ando et al. 1999; Flocchini et al. 2005), fat bodies (Czyzowicz et al. 2009b; Agathangelou et al. 2013), opacity (Di Luna et al. 2014), faults (Agmon

Claude (Anthropic) was used in cataloguing the problems.

and Peleg 2006). In the catalogue every axis starts at its most hostile value, so that the root of each tree is the setting in which nothing is assumed; the standard variant of a problem is then a point well above the root on the geometric axes and close to it on the others.

3.2 Notation

Symbols of this model. Inside the model each symbol has one meaning; *papers write* gives other symbols found in the literature.

Symbol	Meaning	Papers write	Used in
n	number of robots	N in some mutual visibility papers	whole model
f	maximum number of faulty robots (crashed or Byzantine)		whole model
ℓ	number of distinct labels of homonymous robots; $\ell = 1$ anonymous, $\ell = n$ distinct identifiers	h	whole model
c	number of colours of a light		whole model
V	visibility radius (limited visibility)		whole model
δ	minimum distance a robot travels in a non-rigid move before the adversary may stop it		whole model
Δ_0	diameter of the initial configuration	D_E	whole model
k	number of groups (k-grouping; in partitioning the number of fixed points, each receiving n/k robots)	(Bhagat et al. 2021) write k for the group size of partitioning, which is n/k here	k-grouping, Partitioning, Separation
L	length (circumference) of the ring or circle	ℓ	Uniform deployment on a ring
d	the target spacing L/n in uniform deployment		Uniform deployment on a ring
m	number of patterns in a sequence		Pattern sequences
λ	contraction factor of a λ-contracting gathering protocol		Gathering, Near-gathering
r, s	robots (local names)		whole model
g	hop diameter of the initial visibility graph	D_G, D	Gathering, Scattering
ϕ	expected running time function of scattering algorithm SA_ϕ	f	Scattering

3.3 Assumption axes

Each axis says where the assumption comes from and why the community studies it, then defines its values. Every axis is a scale. Level 0 assumes nothing; each higher level is a stronger assumption, which makes a problem easier. Values on the same level are alternatives that do not imply each other. An algorithm designed for weaker assumptions works under stronger ones; a lower bound proved under stronger assumptions holds under weaker ones.

Body

Suzuki and Yamashita's robots are points: each one observes "the positions of all the robots" (Suzuki and Yamashita 1999), several robots may share a point, and nobody blocks anybody's view. Czyzowicz, Gąsieniec and Pelc (Czyzowicz et al. 2009b) introduced *fat* robots, closed unit discs that cannot overlap

and hide what lies behind them: a robot sees another if some segment between their boundary circles meets no third robot, and a robot that touches another stops and ends its cycle. Gathering then asks for "a configuration for which the union of all discs representing them is connected", in which moreover every robot sees all others, so that all know the task is done (Czyzowicz et al. 2009b). Their motivation is realism: "In reality, even very small robots occupy some space", so robots may block each other's view and motion (Czyzowicz et al. 2009b); later accounts say the same (Agathangelou et al. 2013). (Czyzowicz et al. 2009b) solved three and four oblivious, disoriented robots in ASYNC (non-rigid moves, robots know n) by case analysis and left $n \geq 5$ open ("We were unable to do it"); Agathangelou, Georgiou and Mavronicolas (Agathangelou et al. 2013) answered it for any n in ASYNC, assuming chirality and knowledge of n. Fatness is not weaker in every respect: the common radius gives all robots "a common measure of unit" (Czyzowicz et al. 2009b), a unit of length "for free" (Agathangelou et al. 2013; Bandettini et al. 2011). The catalogue puts fat robots at level 0 because they lose multiplicity points and gain obstruction and collisions; point robots are the setting of almost all other work (Bandettini et al. 2011).

Level	Value	Formal definition
0	**fat**	Each robot is a closed disc of radius 1 (unit disc). Discs may not overlap, so robots cannot share a point and must avoid collisions; a disc hides what lies behind it from other robots. Gathering then asks for a configuration whose union of discs is connected and in which every robot sees all others (Czyzowicz et al. 2009b).
1	**point**	Each robot is a point of the plane: several robots may share a point (multiplicity), and a robot hides only what lies exactly behind it on a line of sight.

Scheduler

Suzuki and Yamashita (Suzuki and Yamashita 1999) introduced the semi-synchronous model: robots act "at infinitely many unpredictable time instants", and in each round of a discrete time an arbitrary non-empty set of robots observes and moves instantaneously, every robot being active infinitely often; Prencipe calls it *Atom* (Prencipe 2007) and FSYNC, where every robot acts in every round, is its special case (Das et al. 2016). The asynchronous model (later named CORDA) is due to Flocchini, Prencipe, Santoro and Widmayer (Flocchini et al. 1999), as (Flocchini et al. 2023) and (Prencipe 2007) also state: they assume "asynchronicity within one step", so that "the actual move of a robot may be based on a situation that lies arbitrarily far in the past", and a robot "may be seen an arbitrary number of times by other robots" while moving; they call this "the most radical deviation" from Suzuki and Yamashita's atomic steps (Flocchini et al. 1999). In Prencipe's words, "computations can be made based on obsolete observations" (Prencipe 2007). Prencipe motivates delays inside a cycle as a way "to model also different motorial speeds of the robots" (Prencipe 2007). The schedulers are interesting because they form a hierarchy: rendezvous of two oblivious robots is solvable in FSYNC and not in SSYNC (Suzuki and Yamashita 1999) (Theorem 3.1 and footnote 5), while a problem separating SSYNC from ASYNC for oblivious robots was sought "for more than two decades" and is known only under limited visibility (Flocchini et al. 2023; Kirkpatrick et al. 2024). Intermediate schedulers (RSYNC, LC-, CM- and Move-atomic ASYNC) were introduced to locate exactly where the power is lost (Flocchini et al. 2023; Buchin et al. 2025). Sequential schedulers, which activate one robot per round, break symmetry by themselves; Flocchini, Navarra, Pattanayak, Piselli and Santoro note that their relationship with FSYNC "has so far been ignored" and show that the two are orthogonal (Flocchini et al. 2026).

Level	Value	Formal definition
0	**ASYNC**	**ASYNC**: each robot's Look, Compute and Move phases take finite but unbounded, adversarially chosen time; a robot may look while others move, and compute on an outdated snapshot. Time is counted in epochs: an **epoch** is an interval of time in which every robot completes at least one full Look–Compute–Move cycle; it need not be the shortest such interval. The execution is divided into consecutive epochs, and a bound of X epochs holds for every such division. In FSYNC and SSYNC time is counted in rounds. Called *Corda* by Prencipe (Prencipe 2007). Intermediate variants restrict interleavings: *LC-atomic* (no Look by another robot between a robot's Look and its Compute), *Move-atomic* (no Look during another robot's Move) and *CM-atomic* (Flocchini et al. 2023).
1	**SSYNC**	**SSYNC**: time is divided into rounds; in each round a non-empty subset of robots, chosen by a fair adversary, executes one atomic Look–Compute–Move cycle. Called *Atom* by Prencipe (Prencipe 2007).
2	**sequential**	**Sequential**: an SSYNC scheduler that activates exactly one robot per round (fairly). Also called a *centralized* scheduler (Terai et al. 2023); a *round-robin* scheduler is a fair special case.
2	**FSYNC**	**FSYNC**: every robot executes one atomic cycle in every round.

Lights (persistent memory)

Suzuki and Yamashita model a robot as "a mobile processor with infinite memory" and call an algorithm *oblivious* when the new position depends only on the current observation, noting that "oblivious algorithms are self-stabilizing by definition" (Suzuki and Yamashita 1999). Obliviousness then became the default, partly because "any algorithm proved correct in this model is also self-stabilizing" (Clement et al. 2010) (see also Défago et al. 2020); oblivious, silent robots communicate only by moving, "reminiscent of bees in a bee dance" (Flocchini et al. 1999; Prencipe 2007). Unbounded memory was compared early: two non-oblivious robots can rendezvous in SSYNC and oblivious ones cannot (Suzuki and Yamashita 1999) (Theorems 3.1 and 3.3) (Izumi et al. 2012). Lights were "initially suggested by Peleg" (Das et al. 2016; Di Luna et al. 2014): his survey proposes giving robots "a small amount (say, O(1) bits) of stable memory" and "marking (at least some of) the robots with colors, flags or visible indicator lights" (Peleg 2005) (Efrima and Peleg likewise give each robot a two-state light bulb, "in effect amounting to one bit of memory and communication", to partition robots by a randomized algorithm Efrima and Peleg 2007); Das, Flocchini, Prencipe, Santoro and Yamashita (Das et al. 2016) turned them into the LUMI model, a constant-size register that is persistent and visible to all, so that a robot can remember and communicate a constant number of bits per cycle. Their main results are that LUMI under ASYNC is equivalent to LUMI under SSYNC and strictly stronger than OBLOT under SSYNC (Das et al. 2016). To separate the two roles of a light, Flocchini, Santoro, Viglietta and Yamashita (Flocchini et al. 2016a) defined FSTA (a robot sees only its own light: constant memory, no communication) and FCOM (it sees only the others' lights: constant communication, no memory), and showed that FSTA robots with rigid moves rendezvous in SSYNC and FCOM robots even in ASYNC. The question "is it better to remember or to communicate" then became a programme comparing the four models under every scheduler (Flocchini et al. 2019b, 2023; Das et al. 2024).

Claude (Anthropic) was used in cataloguing the problems.

Level	Value	Formal definition
0	**oblivious**	**OBLOT**: a robot's computation depends only on the current snapshot; nothing persists across cycles.
1	**internal light (FSTA)**	**FSTA**: each robot has a light with c colours that only it can see; it persists between cycles (constant persistent memory, no communication). Also called an *internal* light.
1	**external light (FCOM)**	**FCOM**: each robot has a light with c colours that only the others can see (constant communication, no persistent memory of its own). Also called an *external* light.
2	**lights (LUMI)**	**LUMI**: each robot has a light with c colours, set at the end of Compute, visible to all robots including itself; it is both memory and communication. Also called *luminous* robots, or a *full* light.

Agreement on coordinates

Suzuki and Yamashita's robots "initially do not have a common x-y coordinate system", although they share handedness: in every local frame "the positive y-direction is 90 degrees counterclockwise from the positive x-direction" (Suzuki and Yamashita 1999); together with anonymity, this disorientation is what keeps symmetric configurations symmetric. Flocchini, Prencipe, Santoro and Widmayer (Flocchini et al. 1999, 2008a) dropped that common handedness and made agreement a parameter, studying "the role of common knowledge" in pattern formation by asynchronous robots: with the directions and orientations of both axes any pattern can be formed; with one axis direction (with or without its orientation) the problem is solvable when n is odd and in general not when n is even; with no axis it is unsolvable in general (ISAAC version, proofs omitted) (Flocchini et al. 1999). The journal version proves these results for agreement on the direction and orientation of one axis, without common handedness (Theorem 4.1, Theorem 5.1, Corollary 2.1, Theorem 7.1 of Flocchini et al. 2008a). Efrima and Peleg (Efrima and Peleg 2009), "elaborating on" that work, list full compass, half compass, direction-only, axes-only and no compass, and note that without a common orientation robots "do not share the notion of 'clockwise'". Common handedness, *chirality*, became the usual intermediate assumption: Dieudonné, Petit and Villain list sense of direction and chirality among the global properties that decide what a swarm can do, and show that leader election and arbitrary pattern formation are equivalent in ASYNC for $n \geq 4$ with chirality and $n \geq 5$ without (Dieudonné et al. 2010); removing chirality from pattern formation and gathering is a research line of its own (Bramas and Tixeuil 2015; Cicerone et al. 2019). Partial agreement has also been quantified: Izumi et al. give each robot an "unreliable compass" and find the largest deviation that still lets two oblivious robots gather (Izumi et al. 2012), and agreement on one axis is used for gathering under limited visibility (Poudel and Sharma 2021; Castenow et al. 2022).

Claude (Anthropic) was used in cataloguing the problems.

Level	Value	Formal definition
0	**disoriented**	Each robot r observes in its own coordinate system, obtained from the global one by an unknown similarity (translation, rotation, uniform scaling, and possibly reflection); systems may differ between robots and, in some models, between cycles. Papers that call robots *disoriented* do not always mean this: Défago and Souissi use the word while assuming chirality (Défago and Souissi 2008). Some papers also let the adversary change a robot's rotation, reflection and unit at every Look (*variable disorientation*) (Flocchini et al. 2023; Wada et al. 2026).
1	**chirality**	All local coordinate systems have the same handedness (clockwise orientation): no reflection. Also called *handedness*. Prencipe uses *chirality* for agreement on North and East, i.e. a compass (Prencipe 2007).
1	**one axis**	All robots agree on the direction and orientation of one axis (e.g. North), but not necessarily on handedness: two robots may disagree on which side is East. This is how the papers use *one axis agreement* (e.g. robots "agree on only 'left' and 'right'" Bose et al. 2021a; robots "do not necessarily have the same handedness" Pagli et al. 2015). One axis and chirality are alternatives on this axis; together they amount to a common compass.
2	**compass**	All robots agree on the directions and orientations of both axes (a common compass); they may still disagree on origin and unit. Called *full compass* by Efrima and Peleg, who also use *half compass* (both axis directions, the positive sense of one axis, no chirality), *direction-only* and *axes-only* (Efrima and Peleg 2009). Agreement on one axis together with common chirality is equivalent: the orientation of the second axis then follows from handedness.

Movement

In CORDA a move may stop before its destination: Prencipe's Assumption A1 gives each robot a constant $\delta_r > 0$ such that it either reaches its destination or travels at least δ_r towards it, so that "it cannot know how far it will go in the current cycle" (Prencipe 2007). The physical motivation is "limits to the robot's motorial capabilities" (Prencipe 2007); the lower bound δ is needed because without it no algorithm could be guaranteed to terminate (Bandettini et al. 2011). Non-rigid moves let the adversary act in space as the scheduler acts in time, and *rigid* moves (every move reaches its destination) are the stronger assumption that many results state explicitly. Flocchini, Santoro, Viglietta and Yamashita (Flocchini et al. 2016a) first proved their rendezvous algorithms for rigid moves and then showed that "rigidity of the movements can be traded with knowledge of δ". Heriban, Défago and Tixeuil (Heriban et al. 2018) work explicitly in "the non-rigid ASYNC model", and later landscape papers removed rigidity (and chirality) from the earlier synchronous comparisons of OBLOT, FSTA, FCOM and LUMI, as reported by (Das et al. 2024).

Level	Value	Formal definition
0	**non-rigid**	In a Move phase the adversary may stop a robot before its destination, but only after it has travelled at least $\delta > 0$ (a constant unknown to the robots) or reached the destination.
1	**rigid**	Every Move phase ends at the computed destination.

Claude (Anthropic) was used in cataloguing the problems.

Multiplicity detection

With point robots several robots can share a point, and the snapshot decides whether this is visible. In Prencipe's statement of CORDA robots "can not distinguish whether there is more than one fellow on a given position" (Prencipe 2007). Prencipe observed that every known gathering algorithm used some "extra" assumption, mainly that robots "detect whether a given point on the plane is occupied by one or more robots", and proved gathering impossible without such help (Prencipe 2007). Cieliebak, Flocchini, Prencipe and Santoro (Cieliebak et al. 2012) then showed that with multiplicity detection any $n > 2$ oblivious, disoriented, asynchronous robots gather; their robots tell "none, one, or more than one" robot at a point, and their Theorem 5.36 covers $n \geq 5$, the cases $n = 3, 4$ coming from earlier work of Cieliebak and Prencipe (Cieliebak et al. 2012); the idea is to create one multiplicity point and keep it unique (Bandettini et al. 2011). Later work distinguishes a *weak* detector (one robot or more) from a *strong* one (the exact number) (Bouzid et al. 2013; Pattanayak et al. 2019b); the earliest source of this split was not traced. The level matters: Terai, Wada and Katayama note that strong detection alone lets robots without lights gather, so a model with lights must keep colours from revealing multiplicities (Terai et al. 2023); under sequential schedulers weak detection is necessary and sufficient for gathering (Flocchini et al. 2026).

Level	Value	Formal definition
0	**no multiplicity detection**	A snapshot is a set of points: a point occupied by several robots looks like a single robot.
1	**weak multiplicity**	A snapshot marks each occupied point as holding one robot or more than one.
2	**strong multiplicity**	A snapshot is a multiset: each occupied point comes with the exact number of robots on it.

Visibility

Ando, Oasa, Suzuki and Yamashita (Ando et al. 1999) introduced robots that see only within a fixed radius V; Kirkpatrick et al. call it "the pioneering work on Point Convergence under limited visibility", (Kirkpatrick et al. 2024). Each robot moves towards the centre of the smallest enclosing circle of the robots it sees, but only so far that every robot it sees stays within distance V; the proof assumes points, no obstruction, and phases of negligible duration (so the scheduler is SSYNC although the paper calls the robots asynchronous), with each move reaching its computed destination, at most a small constant away (Ando et al. 1999). Flocchini, Prencipe, Santoro and Widmayer (Flocchini et al. 2005) gathered oblivious robots with limited visibility in ASYNC, with non-rigid moves and without multiplicity detection, when they share "orientation", i.e. agree on the directions and orientations of both axes but not on unit or origin; they conclude that "orientation is at least as powerful as instantaneous movements". Limited visibility makes the model local: the initial visibility graph must be connected (Flocchini et al. 2005) (Lemma 4.7), and known algorithms keep every initially visible pair visible (Ando et al. 1999; Bandettini et al. 2011; Kirkpatrick et al. 2024). It also became a tool for separation: Katreniak solved convergence with limited visibility under 1-bounded asynchrony, adapting a k-bounded scheduler from earlier work on flocking (Katreniak 2011), and convergence with limited visibility is the known problem that separates SSYNC from ASYNC for oblivious robots (Kirkpatrick et al. 2024; Flocchini et al. 2023). Das et al. argue that full visibility "is not practically feasible" and compare OBLOT, FSTA, FCOM and LUMI under both visibilities (Das et al. 2024).

Level	Value	Formal definition
0	**limited visibility**	A robot sees exactly the robots within Euclidean distance V of its position, for a fixed radius $V > 0$.
1	**unlimited visibility**	A robot sees all robots (subject to opacity).

Opacity

In Suzuki and Yamashita's model and in CORDA robots are transparent: "three or more collinear agents are assumed to be mutually visible" (Di Luna et al. 2014). Di Luna, Flocchini, Gan Chaudhuri, Santoro and Viglietta (Di Luna et al. 2014, 2017) started "the study of computing when visibility is obstructable" for point robots, since "given three collinear agents, the one in the middle blocks the visibility between the other two". Before them obstruction had been studied for fat robots (Czyzowicz et al. 2009b; Agathangelou et al. 2013) and for robots on a line, according to (Di Luna et al. 2014). The basic problem they posed, Complete (Mutual) Visibility, asks the robots to reach a configuration in which all see each other; they solved it in ASYNC without collisions with a constant number of colours (Di Luna et al. 2014). Opaque robots with lights then became a line of work on time and colours (Vaidyanathan et al. 2015; Sharma et al. 2021a) and on pattern formation (Bose et al. 2021a), and Feletti, Mambretti, Mereghetti and Palano compare the computational power of transparent and opaque robots directly (Feletti et al. 2025).

Level	Value	Formal definition
0	**opaque**	Robots are points that obstruct sight: r sees s iff no third robot lies on the open segment $\overline{rs}$.
1	**transparent**	Robots do not obstruct each other's view.

Faults

Agmon and Peleg (Agmon and Peleg 2006) (SODA 2004) brought crash and Byzantine robots into the Look–Compute–Move model, requiring the algorithm "to successfully gather the nonfaulty robots, independently of the behavior of the faulty ones". They observed that most existing algorithms fail with a single crash, gathered $n \geq 3$ robots with one crash in SSYNC (Theorem 3.8), proved that three robots cannot gather, or even converge, in SSYNC if one may be Byzantine (Theorem 4.4; hence also in ASYNC, which their abstract states), and gathered $n \geq 3f + 1$ robots with f Byzantine faults in FSYNC (Theorem 5.10) (Agmon and Peleg 2006). Their robots share handedness, see multiplicities (a Look returns a multiset) and make non-rigid moves of at least a fixed distance (Agmon and Peleg 2006). The motivation is deployment in adverse environments, where a swarm must "ensure coordination in spite of faulty robots" (Clement et al. 2010), and oblivious algorithms are already self-stabilizing, so transient faults come for free and permanent ones are the question (Défago et al. 2020). Défago et al. (Défago et al. 2020) extended the feasibility study to bounded schedulers and to probabilistic gathering under crash and Byzantine faults. A crashed robot "simply stops moving" and, since correct robots may also stay still, cannot be identified (Efrima and Peleg 2009); crash-tolerant gathering of many faults was obtained with strong multiplicity detection (Bouzid et al. 2013) and with weak multiplicity detection (Pattanayak et al. 2019b).

Level	Value	Formal definition
0	**Byzantine**	Up to f robots move arbitrarily (adversarially); with lights, they may also set arbitrary colours.
1	**crash faults**	Up to f robots stop forever at adversarial times and remain visible at their last position.
2	**no faults**	All robots follow the algorithm forever.

Randomness

The classical model is deterministic: in CORDA the Compute phase runs "a deterministic algorithm" (Prencipe 2007), so a configuration that looks the same to two anonymous, disoriented robots makes them act alike forever. Randomization was introduced to escape such impossibilities: Dieudonné and Petit (Dieudonné and Petit 2009) proved that scattering cannot be solved deterministically in SSYNC

and gave a randomized algorithm, and Défago et al. (Défago et al. 2020) studied deterministic gathering under bounded schedulers and probabilistic gathering, with and without crash and Byzantine faults. Clément et al. note that work around the impossibilities had focused on additional assumptions, "less attention being shown to the use of randomization", and analyse probabilistic gathering and scattering with Markov chains (Clement et al. 2010). Yamauchi and Yamashita (Yamauchi and Yamashita 2014) formed any pattern in ASYNC without a common coordinate system by randomization, for $n \geq 5$ robots with common handedness, and patterns without multiplicities; Bramas and Tixeuil then treated randomness as a resource, using one random bit per cycle for pattern formation (Bramas and Tixeuil 2016) and proving that $n \log n$ random bits are necessary to scatter n robots (Bramas and Tixeuil 2017). What the adversary may learn about the coin flips (adaptive or not) is part of the model and is fixed per paper (Efrima and Peleg 2009).

Level	Value	Formal definition
0	**deterministic**	The destination computed in a cycle is a function of the snapshot (and the light, if any).
1	**randomized**	Robots may use random bits in Compute; guarantees hold with probability 1 or in expectation, as stated per result.

Identifiers

Suzuki and Yamashita's robots are anonymous: "they all execute the same algorithm and they cannot be distinguished by their appearances" (Suzuki and Yamashita 1999). With disorientation and determinism, anonymity makes symmetry the central obstacle, and the characterisations of solvable instances are stated in terms of the symmetry of the configuration (Suzuki and Yamashita 1999; Dieudonné et al. 2010). Das, Flocchini, Santoro and Yamashita (Das et al. 2015b) compared anonymous and identified (*eponymous*) robots on forming sequences of patterns and found that anonymous robots form only sequences of equal symmetricity while identified robots form nearly all: in SSYNC with rigid moves, chirality and strong multiplicity detection, anonymous robots form a cyclic sequence exactly when all its patterns have the same symmetricity, a multiple of that of the initial configuration, and robots with distinct visible identifiers form every sequence of patterns of at most n points (Theorems 1 and 3 of Das et al. 2015b). To measure the gap, Bouzid and Lamani (Bouzid and Lamani 2011) imported the homonymous model of Delporte-Gallet et al. (Delporte-Gallet et al. 2013), in which n robots share ℓ labels, into the Suzuki–Yamashita model. Asahiro and Yamashita (Asahiro and Yamashita 2025) recast labels as algorithms: robots running one of ℓ algorithms behave like robots with ℓ shared labels, so the *minimum algorithm size* of a task measures how much identification it needs.

Level	Value	Formal definition
0	**anonymous**	Robots are indistinguishable and run the same algorithm.
1	**homonymous**	Robots carry labels from a set of size ℓ, $1 \leq \ell \leq n$, which may repeat; equivalently, robots may run one of ℓ algorithms. The minimum ℓ that solves a problem is its *minimum algorithm size* (Asahiro and Yamashita 2025).
2	**distinct IDs (eponymous)**	**Distinct identifiers** (eponymous robots): labels are pairwise distinct ($\ell = n$); whether others can see a robot's label is stated per result.

Which model is stronger

Robot models combine a memory model (OBLOT, FSTA, FCOM, LUMI) with a scheduler. Model X is **stronger** than Y ($X > Y$) if every problem solvable in Y is solvable in X and some problem, the

separator, is solvable in X but not in Y; $X \equiv Y$ if they solve the same problems; $X \perp Y$ if each solves a problem the other cannot. The schedulers between FSYNC and ASYNC are RSYNC (after an optional fully synchronous prefix, consecutive sets of active robots are disjoint), SSYNC, and three restrictions of ASYNC: LC-atomic (no robot looks between another's Look and Compute), CM-atomic and Move-atomic (no robot looks while another moves) (Flocchini et al. 2023; Buchin et al. 2025).

A scheduler that allows fewer executions gives at least as much power: FSYNC $\geq$ SSYNC, RSYNC $\geq$ SSYNC, and SSYNC $\geq$ each atomic ASYNC $\geq$ ASYNC. More visible light gives at least as much power under the same scheduler: LUMI $\geq$ FCOM, FSTA $\geq$ OBLOT.

The website shows the same relations as an interactive grid: choosing a model colours every other model as stronger, weaker, equivalent, incomparable or unknown, including relations that follow by transitivity and incomparabilities that carry over to equivalent models.

Any number of robots

Most results assume chirality and rigid moves; separators use small teams (often $n \leq 4$). Sources: (Das et al. 2016; Flocchini et al. 2019b; Buchin et al. 2025; Flocchini et al. 2023; Das et al. 2021b; Naito et al. 2025; Flocchini et al. 2026; Kirkpatrick et al. 2024; Feletti et al. 2026b). The sequential-scheduler relations of (Feletti et al. 2026b) assume variable disorientation (no chirality, no common unit) and strong multiplicity detection.

Model		Model	Separator	Source
OBLOT$^{\text{FSYNC}}$	$>$	OBLOT$^{\text{SSYNC}}$	RDV	Suzuki and Yamashita (1999) Theorem 3.1 (no oblivious algorithm for two robots) and footnote 5 (midpoint algorithm for synchronized robots); see also (Flocchini et al. 2019b) Thm 20.4 and (Das et al. 2016) Lemma 3.3; also Flocchini et al. (2019b) Suzuki and Yamashita's robots share handedness and each move reaches its computed destination (bounded step length); the impossibility holds a fortiori without these.
LUMI$^{\text{ASYNC}}$	$>$	OBLOT$^{\text{SSYNC}}$	RDV	Das et al. (2016) Thm 3.4 (via Lemma 3.3, Thm 3.3)
LUMI$^{\text{ASYNC}}$	$\equiv$	LUMI$^{\text{SSYNC}}$	*simulation*	Das et al. (2016) Thm 3.6 (via Thm 3.5); also Flocchini et al. (2023)
FCOM$^{\text{FSYNC}}$	$\equiv$	LUMI$^{\text{FSYNC}}$	*simulation*	Flocchini et al. (2019b) Thm 2 (Lemma 1, simulation LUbyFCinFSY)
FCOM$^{\text{FSYNC}}$	$>$	FSTA$^{\text{FSYNC}}$	¬IL	Flocchini et al. (2019b) Thm 6 (Lemmas 4, 5); also Buchin et al. (2025)
FSTA$^{\text{FSYNC}}$	$>$	OBLOT$^{\text{FSYNC}}$	OSP	Flocchini et al. (2019b) Thm 10 (Lemmas 8, 9)
FCOM$^{\text{SSYNC}}$	$\perp$	FSTA$^{\text{SSYNC}}$	¬IL, TAR(d)	Flocchini et al. (2019b) Thm 14
FSTA$^{\text{SSYNC}}$	$>$	OBLOT$^{\text{SSYNC}}$	RDV	Flocchini et al. (2019b) Thm 15 Uses rendezvous by FSTA robots in SSYNC (Flocchini et al. 2016a), which needs each robot's unit of distance to stay fixed.
FCOM$^{\text{SSYNC}}$	$>$	OBLOT$^{\text{SSYNC}}$	RDV	Flocchini et al. (2019b) Thm 15
LUMI$^{\text{SSYNC}}$	$>$	FSTA$^{\text{SSYNC}}$	OSP	Flocchini et al. (2019b) Thm 17 (Lemma 16)
LUMI$^{\text{SSYNC}}$	$>$	FCOM$^{\text{SSYNC}}$	OSP	Flocchini et al. (2019b) Thm 17 (Lemma 16)
LUMI$^{\text{FSYNC}}$	$>$	LUMI$^{\text{SSYNC}}$	CGE	Flocchini et al. (2019b) Thm 20.1 (Lemma 19)

Model		Model	Separator	Source
FSTA$^{\text{FSYNC}}$	$>$	FSTA$^{\text{SSYNC}}$	CGE	Flocchini et al. (2019b) Thm 20.2
FCOM$^{\text{FSYNC}}$	$>$	LUMI$^{\text{SSYNC}}$	CGE	Flocchini et al. (2019b) Thm 20.3
OBLOT$^{\text{FSYNC}}$	$\perp$	FCOM$^{\text{SSYNC}}$	SRO, ¬IL	Flocchini et al. (2019b) Thm 25 (Lemmas 22, 23)
OBLOT$^{\text{FSYNC}}$	$\perp$	FSTA$^{\text{SSYNC}}$	SRO, TAR(d)	Flocchini et al. (2019b) Thm 25
LUMI$^{\text{SSYNC}}$	$\perp$	FSTA$^{\text{FSYNC}}$	¬IL, CGE	Flocchini et al. (2019b) Thm 26
FCOM$^{\text{SSYNC}}$	$\perp$	FSTA$^{\text{FSYNC}}$	¬IL, CGE	Flocchini et al. (2019b) Thm 26
LUMI$^{\text{SSYNC}}$	$\perp$	OBLOT$^{\text{FSYNC}}$	OSP, CGE*	Flocchini et al. (2019b) Thm 28 (answers the Das et al. 2016 question for LUMI^A since LUMI^A ≡ LUMI^S)
FSTA$^{\text{FSYNC}}$	$>$	OBLOT$^{\text{SSYNC}}$	CGE, RDV	Flocchini et al. (2019b) Table 3 (Th. 15, 20)
OBLOT$^{\text{RSYNC}}$	$\perp$	FCOM$^{\text{SSYNC}}$	SRO, RDV	Buchin et al. (2025) Thm 3.1
OBLOT$^{\text{RSYNC}}$	$\perp$	FSTA$^{\text{SSYNC}}$	SRO, RDV	Buchin et al. (2025) Thm 3.2
FCOM$^{\text{RSYNC}}$	$>$	FCOM$^{\text{SSYNC}}$	SRO	Buchin et al. (2025) Thm 3.3
FSTA$^{\text{RSYNC}}$	$>$	FSTA$^{\text{SSYNC}}$	SRO	Buchin et al. (2025) Thm 3.4
OBLOT$^{\text{RSYNC}}$	$>$	OBLOT$^{\text{SSYNC}}$	SRO	Buchin et al. (2025) Thm 3.5
LUMI$^{\text{SSYNC}}$	$>$	FSTA$^{\text{RSYNC}}$	CYC	Buchin et al. (2025) Thm 4.1
FSTA$^{\text{RSYNC}}$	$\perp$	FCOM$^{\text{SSYNC}}$	SRO, CYC	Buchin et al. (2025) Thm 4.2
LUMI$^{\text{SSYNC}}$	$\equiv$	LUMI$^{\text{RSYNC}}$	*simulation*	Buchin et al. (2025) Thm 6 (Thm 5, simulation sim-RS-by-S)
FSTA$^{\text{FSYNC}}$	$\perp$	LUMI$^{\text{RSYNC}}$	CYC, CGE	Buchin et al. (2025) Thm 8.1
FSTA$^{\text{FSYNC}}$	$\perp$	FCOM$^{\text{RSYNC}}$	CYC, CGE	Buchin et al. (2025) Thm 8.2
OBLOT$^{\text{FSYNC}}$	$\perp$	LUMI$^{\text{RSYNC}}$	OSP, CGE*	Buchin et al. (2025) Thm 9.1
OBLOT$^{\text{FSYNC}}$	$\perp$	FCOM$^{\text{RSYNC}}$	OSP, CGE*	Buchin et al. (2025) Thm 9.2
OBLOT$^{\text{FSYNC}}$	$\perp$	FSTA$^{\text{RSYNC}}$	OSP, CGE*	Buchin et al. (2025) Thm 9.3; also Naito et al. (2025)
FCOM$^{\text{RSYNC}}$	$\equiv$	LUMI$^{\text{RSYNC}}$	*simulation*	Buchin et al. (2025) Thm 11 (Thm 10: LUMI^S ⊠ FCOM^RS)
FCOM$^{\text{RSYNC}}$	$>$	FSTA$^{\text{RSYNC}}$	CYC	Buchin et al. (2025) Thm 11
FSTA$^{\text{RSYNC}}$	$>$	OBLOT$^{\text{RSYNC}}$	RDV	Buchin et al. (2025) Thm 11 (Lemma 1: RDV ∉ OBLOT^RS)
OBLOT$^{\text{SSYNC}}$	$>$	OBLOT$^{\text{ASYNC}}$	MLCv	Flocchini et al. (2023) LIPIcs Thm 6 (arXiv Thm 1); also Kirkpatrick et al. (2024)
OBLOT$^{\text{Move-atomic}}$	$>$	OBLOT$^{\text{ASYNC}}$	TF	Flocchini et al. (2023) LIPIcs Thm 10 (arXiv Thm 2)
OBLOT$^{\text{SSYNC}}$	$>$	OBLOT$^{\text{Move-atomic}}$	MLCv	Flocchini et al. (2023) LIPIcs Thm 11 (arXiv Thm 3)
OBLOT$^{\text{LC-atomic}}$	$\equiv$	OBLOT$^{\text{ASYNC}}$	*simulation*	Flocchini et al. (2023) LIPIcs Thm 11 (arXiv Thm 3)
FCOM$^{\text{SSYNC}}$	$>$	FCOM$^{\text{ASYNC}}$	MLCv	Flocchini et al. (2023) LIPIcs Thm 14 (arXiv Thm 4); also Naito et al. (2025)
FCOM$^{\text{CM-atomic}}$	$\equiv$	FCOM$^{\text{ASYNC}}$	*simulation*	Flocchini et al. (2023) LIPIcs Thm 15 (arXiv Thm 5), simulation SIM
FCOM$^{\text{SSYNC}}$	$\equiv$	FCOM$^{\text{LC-atomic}}$	*simulation*	Flocchini et al. (2023) LIPIcs Thm 17 (arXiv Thm 6)
FCOM$^{\text{LC-atomic}}$	$>$	FCOM$^{\text{CM-atomic}}$	MLCv	Flocchini et al. (2023) LIPIcs Thm 18 (arXiv Thm 7)

Claude (Anthropic) was used in cataloguing the problems.

Model		Model	Separator	Source
FSTA$^{\text{SSYNC}}$	$>$	FSTA$^{\text{ASYNC}}$	MLCv	Flocchini et al. (2023) LIPIcs Thm 21 (arXiv Thm 8)
FSTA$^{\text{Move-atomic}}$	$>$	FSTA$^{\text{ASYNC}}$	TF	Flocchini et al. (2023) LIPIcs Thm 23 (arXiv Thm 9)
FSTA$^{\text{SSYNC}}$	$>$	FSTA$^{\text{Move-atomic}}$	MLCv	Flocchini et al. (2023) LIPIcs Thm 24.2 (arXiv Thm 10.2); also Naito et al. (2025)
FSTA$^{\text{LC-atomic}}$	$\equiv$	FSTA$^{\text{ASYNC}}$	*simulation*	Flocchini et al. (2023) LIPIcs Thm 24.1
FCOM$^{\text{LC-atomic}}$	$\perp$	FSTA$^{\text{Move-atomic}}$	CYC, GCNCL	Flocchini et al. (2023) LIPIcs Thm 28.1 (arXiv Thm 11.1)
FCOM$^{\text{LC-atomic}}$	$\perp$	FSTA$^{\text{ASYNC}}$	CYC, GCNCL	Flocchini et al. (2023) LIPIcs Thm 28.2
FCOM$^{\text{LC-atomic}}$	$>$	OBLOT$^{\text{SSYNC}}$	RDV	Flocchini et al. (2023) LIPIcs Thm 28.3
FCOM$^{\text{ASYNC}}$	$\perp$	FSTA$^{\text{Move-atomic}}$	CYC, GCNCL	Flocchini et al. (2023) LIPIcs Thm 29.1 (arXiv Thm 12.1)
FCOM$^{\text{ASYNC}}$	$\perp$	FSTA$^{\text{ASYNC}}$	CYC, GCNCL	Flocchini et al. (2023) LIPIcs Thm 29.2; also Das et al. (2021b)
FCOM$^{\text{ASYNC}}$	$\perp$	OBLOT$^{\text{SSYNC}}$	MLCv, CYC	Flocchini et al. (2023) LIPIcs Thm 29.3
FCOM$^{\text{ASYNC}}$	$>$	OBLOT$^{\text{Move-atomic}}$	RDV	Flocchini et al. (2023) LIPIcs Thm 29.4 Uses rendezvous by FCOM robots in ASYNC with 12 colours (Flocchini et al. 2016a), which needs a fixed unit of distance; under variable disorientation it is impossible (Wada et al. 2026).
FSTA$^{\text{Move-atomic}}$	$\perp$	OBLOT$^{\text{SSYNC}}$	RDV, MLCv	Flocchini et al. (2023) LIPIcs Thm 30.1 (arXiv Thm 13.1) Uses rendezvous by FSTA robots in ASYNC, stated without proof; under variable disorientation it is impossible (Wada et al. 2026).
FSTA$^{\text{Move-atomic}}$	$>$	OBLOT$^{\text{Move-atomic}}$	RDV	Flocchini et al. (2023) LIPIcs Thm 30.2; also Naito et al. (2025) Uses rendezvous by FSTA robots in ASYNC, stated without proof; under variable disorientation it is impossible (Wada et al. 2026).
FSTA$^{\text{ASYNC}}$	$\perp$	OBLOT$^{\text{Move-atomic}}$	RDV, TF	Flocchini et al. (2023) LIPIcs Thm 30.3 Uses rendezvous by FSTA robots in ASYNC, stated without proof; under variable disorientation it is impossible (Wada et al. 2026).
FSTA$^{\text{ASYNC}}$	$\perp$	OBLOT$^{\text{SSYNC}}$	RDV, MLCv	Flocchini et al. (2023) LIPIcs Thm 30.4 Uses rendezvous by FSTA robots in ASYNC, stated without proof; under variable disorientation it is impossible (Wada et al. 2026).
FSTA$^{\text{ASYNC}}$	$>$	OBLOT$^{\text{ASYNC}}$	RDV	Flocchini et al. (2023) LIPIcs Thm 30.5; also Das et al. (2021b), Naito et al. (2025) Uses rendezvous by FSTA robots in ASYNC, stated without proof; under variable disorientation it is impossible (Wada et al. 2026).
OBLOT$^{\text{FSYNC}}$	$\perp$	FSTA$^{\text{ASYNC}}$	SRO, OC	Das et al. (2021b) Thm 1 (Lemmas 1, 2)
FCOM$^{\text{ASYNC}}$	$\perp$	FSTA$^{\text{FSYNC}}$	¬IL, CGE	Das et al. (2021b) Thm 3 (Lemma 3: ¬IL ∈ FCOM^A)
FCOM$^{\text{ASYNC}}$	$\perp$	FSTA$^{\text{SSYNC}}$	¬IL, TAR(d)	Das et al. (2021b) Thm 4
FCOM$^{\text{ASYNC}}$	$\perp$	OBLOT$^{\text{FSYNC}}$	¬IL, SRO	Das et al. (2021b) Thm 5
FCOM$^{\text{SSYNC}}$	$\perp$	FSTA$^{\text{ASYNC}}$	¬IL, IOP	Das et al. (2021b) Thm 6 (Lemmas 4, 5)
LUMI$^{\text{FSYNC}}$	$>$	FSTA$^{\text{FSYNC}}$	ETE	Naito et al. (2025) Thm 3 (Lemmas 3–5)

Model		Model	Separator	Source
LUMI$^{\text{FSYNC}}$	$>$	LUMI$^{\text{RSYNC}}$	ETE	Naito et al. (2025) Thm 3
FSTA$^{\text{FSYNC}}$	$>$	FSTA$^{\text{RSYNC}}$	HET	Naito et al. (2025) Thm 4 (Lemmas 6–10)
LUMI$^{\text{RSYNC}}$	$>$	FSTA$^{\text{RSYNC}}$	HET, TAR(d)*	Naito et al. (2025) Thm 4, Thm 5
FSTA$^{\text{Move-atomic}}$	$\perp$	FCOM$^{\text{SSYNC}}$	VEC, CYC	Naito et al. (2025) Thm 7 (+ CYC from Buchin et al. 2025)
FCOM$^{\text{Move-atomic}}$	$\perp$	FSTA$^{\text{SSYNC}}$	LP–Cv, ZCC	Naito et al. (2025) Thm 8, Thm 9
OBLOT$^{\text{FSYNC}}$	$>$	OBLOT$^{\text{RSYNC}}$	LCM–M	Naito et al. (2025) Thm 11
FSTA$^{\text{ASYNC}}$	$\perp$	OBLOT$^{\text{RSYNC}}$	LCM–M, SRO	Naito et al. (2025) Thm 11 (+ SRO from Flocchini et al. 2019b
FCOM$^{\text{SSYNC}}$	$>$	FCOM$^{\text{Move-atomic}}$	RDV	Naito et al. (2025) Table 1 (compiled 'known' relations
OBLOT$^{\text{SEQ}}$	$\perp$	OBLOT$^{\text{FSYNC}}$	UPF*, Gathering (no multiplicity detection)	Flocchini et al. (2026) Thms 1, 2, 3, 4
OBLOT$^{\text{SEQ}}$	$>$	OBLOT$^{\text{SSYNC}}$	UPF*	Flocchini et al. (2026) Sect. 8.1 (Thm 1 + Thms 2, 3)
FCOM$^{\text{FSYNC}}$	$>$	FCOM$^{\text{RSYNC}}$	CGE	Buchin et al. (2025) Thm 7
OBLOT$^{\text{RROBIN}}$	$>$	OBLOT$^{\text{RPERM}}$	Fan	Feletti et al. (2026b) Thm 6 (Lemmas 4, 5) Variable disorientation (no chirality, no common unit) and strong multiplicity detection.
FSTA$^{\text{RROBIN}}$	$>$	FSTA$^{\text{RPERM}}$	Fan	Feletti et al. (2026b) Thm 6 (Lemmas 4, 5) The only relation the paper proves for FSTA under sequential schedulers.
FCOM$^{\text{RROBIN}}$	$>$	FCOM$^{\text{RPERM}}$	Fan	Feletti et al. (2026b) Thm 6 (Lemmas 4, 5)
LUMI$^{\text{RROBIN}}$	$>$	LUMI$^{\text{RPERM}}$	Fan	Feletti et al. (2026b) Thm 6 (Lemmas 4, 5)
LUMI$^{\text{SEQ}}$	$\equiv$	LUMI$^{\text{RPERM}}$	*simulation*	Feletti et al. (2026b) Thm 7 (simulator in LUMI^SEQ of any LUMI^R-PERM algorithm)
LUMI$^{\text{SEQ}}$	$\equiv$	LUMI$^{\text{PERM}}$	*simulation*	Feletti et al. (2026b) Thm 7 with Thm 1 First shown in the authors' SSS 2025 paper on sequential schedulers (their reference [14]).
LUMI$^{\text{SEQ}}$	$\equiv$	LUMI$^{\text{RSEQ}}$	*simulation*	Feletti et al. (2026b) Thm 7 with Thm 1
OBLOT$^{\text{RPERM}}$	$>$	OBLOT$^{\text{PERM}}$	SquareCenter	Feletti et al. (2026b) Thm 11 (Lemmas 9, 10) Lemma 10 uses the symmetry of the square together with disorientation: the robot that has just moved takes the same snapshot as the three others.
OBLOT$^{\text{RSEQ}}$	$>$	OBLOT$^{\text{SEQ}}$	SquareCenter	Feletti et al. (2026b) Thm 11 (Lemmas 9, 10)
OBLOT$^{\text{PERM}}$	$>$	OBLOT$^{\text{SEQ}}$	Pinwheel	Feletti et al. (2026b) Thm 15 (Lemmas 13, 14) Lemma 13 (Pinwheel in OBLOT^PERM) is taken from the authors' SSS 2025 paper.
OBLOT$^{\text{RPERM}}$	$>$	OBLOT$^{\text{RSEQ}}$	Pinwheel	Feletti et al. (2026b) Thm 15 (Lemmas 13, 14)
OBLOT$^{\text{PERM}}$	$\perp$	OBLOT$^{\text{RSEQ}}$	Pinwheel, SquareCenter	Feletti et al. (2026b) Thm 16
FCOM$^{\text{RPERM}}$	$\equiv$	FCOM$^{\text{RSEQ}}$	*simulation*	Feletti et al. (2026b) Thm 17 (simulator with a shared parity light)
FCOM$^{\text{PERM}}$	$>$	FCOM$^{\text{SEQ}}$	HalveTwice	Feletti et al. (2026b) Thm 21 (Lemmas 19, 20) Lemma 20 needs the unit of distance of a robot to change between activations (variable disorientation).

Claude (Anthropic) was used in cataloguing the problems.

Model		Model	Separator	Source
$\text{FCOM}^{\text{RPERM}}$	>	$\text{FCOM}^{\text{PERM}}$	HalveOnce	Feletti et al. (2026b) Thm 25 (Lemmas 23, 24) Lemma 24 needs the unit of distance of a robot to change between activations (variable disorientation).
$\text{FCOM}^{\text{RSEQ}}$	>	FCOM^{SEQ}	HalveTwice	Feletti et al. (2026b) Table 1 (from Thms 1, 17, 21)

Two robots

Two robots, variable disorientation, chirality and rigid moves; the settings collapse to 13 classes (Kitamura et al. 2025; Ninomiya et al. 2026), both preprints. (Ninomiya et al. 2026) let a robot see when the two robots coincide (no general multiplicity detection); (Kitamura et al. 2025) assume no multiplicity detection. Relations cited to (Ninomiya et al. 2026) were checked against its statements and the proofs of its Appendices C and D; the auxiliary execution lemmas of its Appendix E, on which those proofs rely, were not checked.

Model		Model	Separator	Source
$\text{LUMI}^{\text{FSYNC}}$	≡	$\text{FCOM}^{\text{FSYNC}}$	*simulation*	Kitamura et al. (2025) Thm 2; also Ninomiya et al. (2026)
$\text{LUMI}^{\text{FSYNC}}$	≡	$\text{FSTA}^{\text{FSYNC}}$	*simulation*	Kitamura et al. (2025) Thm 2; also Ninomiya et al. (2026)
$\text{FSTA}^{\text{FSYNC}}$	>	$\text{OBLOT}^{\text{FSYNC}}$	CFSM, VSS	Ninomiya et al. (2026) Lemma 3.4(4), (5); Thm 3.5 CFSM and VSS are solvable in FSTA^{A} and FCOM^{A}, both below $\text{LUMI}^{F} \equiv \text{FSTA}^{F}$, and not in OBLOT^{F}.
$\text{LUMI}^{\text{FSYNC}}$	>	$\text{LUMI}^{\text{RSYNC}}$	EPG*	Ninomiya et al. (2026) Lemma 3.4(1) (Lemma D.11); also Kitamura et al. (2025) (Kitamura et al. 2025) Thm 6 separates these with CGE*; (Ninomiya et al. 2026) (Remark D.9) argue that CGE*'s coordinatewise floor is not a predicate on trajectories under variable disorientation and replace it by EPG*.
$\text{FCOM}^{\text{FSYNC}}$	>	$\text{FCOM}^{\text{RSYNC}}$	EPG*	Ninomiya et al. (2026) Lemma 3.4(1) (Lemma D.11); also Kitamura et al. (2025) (Kitamura et al. 2025) Thm 6 separates these with CGE*; (Ninomiya et al. 2026) (Remark D.9) argue that CGE*'s coordinatewise floor is not a predicate on trajectories under variable disorientation and replace it by EPG*.
$\text{FSTA}^{\text{FSYNC}}$	>	$\text{FSTA}^{\text{RSYNC}}$	EPG*	Ninomiya et al. (2026) Lemma 3.4(1) (Lemma D.11); also Kitamura et al. (2025) (Kitamura et al. 2025) Thm 6 separates these with CGE*; (Ninomiya et al. 2026) (Remark D.9) argue that CGE*'s coordinatewise floor is not a predicate on trajectories under variable disorientation and replace it by EPG*.
$\text{LUMI}^{\text{RSYNC}}$	≡	$\text{LUMI}^{\text{SSYNC}}$	*simulation*	Kitamura et al. (2025) Thm 1
$\text{LUMI}^{\text{SSYNC}}$	≡	$\text{LUMI}^{\text{ASYNC}}$	*simulation*	Kitamura et al. (2025) Thm 1
$\text{LUMI}^{\text{RSYNC}}$	≡	$\text{FCOM}^{\text{RSYNC}}$	*simulation*	Kitamura et al. (2025) Thm 3
$\text{FCOM}^{\text{RSYNC}}$	>	$\text{FCOM}^{\text{SSYNC}}$	SRO	Kitamura et al. (2025) Thm 7
$\text{FSTA}^{\text{RSYNC}}$	>	$\text{FSTA}^{\text{SSYNC}}$	SRO	Kitamura et al. (2025) Thm 7
$\text{OBLOT}^{\text{RSYNC}}$	>	$\text{OBLOT}^{\text{SSYNC}}$	SRO	Kitamura et al. (2025) Thm 7
$\text{FCOM}^{\text{SSYNC}}$	≡	$\text{FCOM}^{\text{LC-atomic}}$	*simulation*	Kitamura et al. (2025) Thm 5
$\text{FCOM}^{\text{SSYNC}}$	>	$\text{FCOM}^{\text{CM-atomic}}$	MCv	Kitamura et al. (2025) Thm 8

Model		Model	Separator	Source
FSTA$^{\text{SSYNC}}$	$>$	FSTA$^{\text{CM-atomic}}$	MCv	Kitamura et al. (2025) Thm 8
OBLOT$^{\text{SSYNC}}$	$>$	OBLOT$^{\text{CM-atomic}}$	MCv	Kitamura et al. (2025) Thm 8
FCOM$^{\text{CM-atomic}}$	$\equiv$	FCOM$^{\text{ASYNC}}$	*simulation*	Kitamura et al. (2025) Thm 4
FSTA$^{\text{CM-atomic}}$	$>$	FSTA$^{\text{ASYNC}}$	DMSD	Kitamura et al. (2025) Thm 9
OBLOT$^{\text{CM-atomic}}$	$>$	OBLOT$^{\text{ASYNC}}$	DMSD	Kitamura et al. (2025) Thm 9
FCOM$^{\text{SSYNC}}$	$\perp$	FSTA$^{\text{RSYNC}}$	SM, RDAM	Kitamura et al. (2025) Thms 10, 11; also Ninomiya et al. (2026) Also separated by CFSM (FSTA^{A}, not FCOM^{S}) and VSS (FCOM^{A}, not FSTA^{R}) (Ninomiya et al. 2026) Lemma 3.4(4), (5).
FCOM$^{\text{SSYNC}}$	$\perp$	FSTA$^{\text{SSYNC}}$	SM, RDAM	Kitamura et al. (2025) Thms 10, 11; also Ninomiya et al. (2026) Also separated by CFSM (FSTA^{A}, not FCOM^{S}) and VSS (FCOM^{A}, not FSTA^{R}) (Ninomiya et al. 2026) Lemma 3.4(4), (5).
FCOM$^{\text{SSYNC}}$	$\perp$	FSTA$^{\text{CM-atomic}}$	SM, RDAM	Kitamura et al. (2025) Thms 10, 11; also Ninomiya et al. (2026) Also separated by CFSM (FSTA^{A}, not FCOM^{S}) and VSS (FCOM^{A}, not FSTA^{R}) (Ninomiya et al. 2026) Lemma 3.4(4), (5).
FCOM$^{\text{SSYNC}}$	$\perp$	FSTA$^{\text{ASYNC}}$	SM, RDAM	Kitamura et al. (2025) Thms 10, 11; also Ninomiya et al. (2026) Also separated by CFSM (FSTA^{A}, not FCOM^{S}) and VSS (FCOM^{A}, not FSTA^{R}) (Ninomiya et al. 2026) Lemma 3.4(4), (5).
FCOM$^{\text{CM-atomic}}$	$\perp$	FSTA$^{\text{RSYNC}}$	SM, RDAM	Kitamura et al. (2025) Thms 10, 11; also Ninomiya et al. (2026) Also separated by CFSM (FSTA^{A}, not FCOM^{S}) and VSS (FCOM^{A}, not FSTA^{R}) (Ninomiya et al. 2026) Lemma 3.4(4), (5).
FCOM$^{\text{CM-atomic}}$	$\perp$	FSTA$^{\text{SSYNC}}$	SM, RDAM	Kitamura et al. (2025) Thms 10, 11; also Ninomiya et al. (2026) Also separated by CFSM (FSTA^{A}, not FCOM^{S}) and VSS (FCOM^{A}, not FSTA^{R}) (Ninomiya et al. 2026) Lemma 3.4(4), (5).
FCOM$^{\text{CM-atomic}}$	$\perp$	FSTA$^{\text{CM-atomic}}$	SM, RDAM	Kitamura et al. (2025) Thms 10, 11; also Ninomiya et al. (2026) Also separated by CFSM (FSTA^{A}, not FCOM^{S}) and VSS (FCOM^{A}, not FSTA^{R}) (Ninomiya et al. 2026) Lemma 3.4(4), (5).
FCOM$^{\text{CM-atomic}}$	$\perp$	FSTA$^{\text{ASYNC}}$	SM, RDAM	Kitamura et al. (2025) Thms 10, 11; also Ninomiya et al. (2026) Also separated by CFSM (FSTA^{A}, not FCOM^{S}) and VSS (FCOM^{A}, not FSTA^{R}) (Ninomiya et al. 2026) Lemma 3.4(4), (5).
FCOM$^{\text{ASYNC}}$	$\perp$	FSTA$^{\text{RSYNC}}$	SM, RDAM	Kitamura et al. (2025) Thms 10, 11; also Ninomiya et al. (2026) Also separated by CFSM (FSTA^{A}, not FCOM^{S}) and VSS (FCOM^{A}, not FSTA^{R}) (Ninomiya et al. 2026) Lemma 3.4(4), (5).

Claude (Anthropic) was used in cataloguing the problems.

Model		Model	Separator	Source
FCOM$^{\text{ASYNC}}$	$\perp$	FSTA$^{\text{SSYNC}}$	SM, RDAM	Kitamura et al. (2025) Thms 10, 11; also Ninomiya et al. (2026) Also separated by CFSM (FSTA^{A}, not FCOM^{S}) and VSS (FCOM^{A}, not FSTA^{R}) (Ninomiya et al. 2026) Lemma 3.4(4), (5).
FCOM$^{\text{ASYNC}}$	$\perp$	FSTA$^{\text{CM-atomic}}$	SM, RDAM	Kitamura et al. (2025) Thms 10, 11; also Ninomiya et al. (2026) Also separated by CFSM (FSTA^{A}, not FCOM^{S}) and VSS (FCOM^{A}, not FSTA^{R}) (Ninomiya et al. 2026) Lemma 3.4(4), (5).
FCOM$^{\text{ASYNC}}$	$\perp$	FSTA$^{\text{ASYNC}}$	SM, RDAM	Kitamura et al. (2025) Thms 10, 11; also Ninomiya et al. (2026) Also separated by CFSM (FSTA^{A}, not FCOM^{S}) and VSS (FCOM^{A}, not FSTA^{R}) (Ninomiya et al. 2026) Lemma 3.4(4), (5).
OBLOT$^{\text{Move-atomic}}$	$\equiv$	OBLOT$^{\text{CM-atomic}}$	*simulation*	Ninomiya et al. (2026) Prop 3.2
FSTA$^{\text{Move-atomic}}$	$\equiv$	FSTA$^{\text{CM-atomic}}$	*simulation*	Ninomiya et al. (2026) Prop 3.2
FCOM$^{\text{Move-atomic}}$	$\equiv$	FCOM$^{\text{CM-atomic}}$	*simulation*	Ninomiya et al. (2026) §2.2, citing (Flocchini et al. 2023) (as reported) The paper cites (Flocchini et al. 2023) for this equivalence; the theorem there was not checked.
OBLOT$^{\text{LC-atomic}}$	$\equiv$	OBLOT$^{\text{ASYNC}}$	*simulation*	Ninomiya et al. (2026) Prop 3.2; also Flocchini et al. (2023)
FSTA$^{\text{LC-atomic}}$	$\equiv$	FSTA$^{\text{ASYNC}}$	*simulation*	Ninomiya et al. (2026) Prop 3.2; also Flocchini et al. (2023)
OBLOT$^{\text{FSYNC}}$	$>$	OBLOT$^{\text{RSYNC}}$	EPG*	Ninomiya et al. (2026) Lemma 3.4(1); Thm 3.5
LUMI$^{\text{RSYNC}}$	$>$	FSTA$^{\text{RSYNC}}$	VSS	Ninomiya et al. (2026) Lemma 3.4(5); Thm 3.5
FSTA$^{\text{RSYNC}}$	$>$	OBLOT$^{\text{RSYNC}}$	CFSM	Ninomiya et al. (2026) Lemma 3.4(4); Thm 3.5
FCOM$^{\text{SSYNC}}$	$>$	OBLOT$^{\text{SSYNC}}$	VSS	Ninomiya et al. (2026) Lemma 3.4(5); Thm 3.5
FSTA$^{\text{SSYNC}}$	$>$	OBLOT$^{\text{SSYNC}}$	CFSM	Ninomiya et al. (2026) Lemma 3.4(4); Thm 3.5
FCOM$^{\text{ASYNC}}$	$>$	OBLOT$^{\text{Move-atomic}}$	VSS	Ninomiya et al. (2026) Lemma 3.4(5); Thm 3.5
FSTA$^{\text{Move-atomic}}$	$>$	OBLOT$^{\text{Move-atomic}}$	CFSM	Ninomiya et al. (2026) Lemma 3.4(4); Thm 3.5
FSTA$^{\text{ASYNC}}$	$>$	OBLOT$^{\text{ASYNC}}$	CFSM	Ninomiya et al. (2026) Lemma 3.4(4); Thm 3.5
OBLOT$^{\text{FSYNC}}$	$\perp$	LUMI$^{\text{RSYNC}}$	EPG*, CFSM	Ninomiya et al. (2026) Table 1; Thm 3.5
OBLOT$^{\text{FSYNC}}$	$\perp$	FSTA$^{\text{RSYNC}}$	EPG*, CFSM	Ninomiya et al. (2026) Table 1; Thm 3.5
OBLOT$^{\text{FSYNC}}$	$\perp$	FCOM$^{\text{SSYNC}}$	EPG*, VSS	Ninomiya et al. (2026) Table 1; Thm 3.5
OBLOT$^{\text{FSYNC}}$	$\perp$	FCOM$^{\text{ASYNC}}$	EPG*, VSS	Ninomiya et al. (2026) Table 1; Thm 3.5
OBLOT$^{\text{FSYNC}}$	$\perp$	FSTA$^{\text{SSYNC}}$	EPG*, CFSM	Ninomiya et al. (2026) Table 1; Thm 3.5
OBLOT$^{\text{FSYNC}}$	$\perp$	FSTA$^{\text{Move-atomic}}$	EPG*, CFSM	Ninomiya et al. (2026) Table 1; Thm 3.5
OBLOT$^{\text{FSYNC}}$	$\perp$	FSTA$^{\text{ASYNC}}$	EPG*, CFSM	Ninomiya et al. (2026) Table 1; Thm 3.5
OBLOT$^{\text{RSYNC}}$	$\perp$	FCOM$^{\text{SSYNC}}$	SRO, VSS	Ninomiya et al. (2026) Table 1; Thm 3.5
OBLOT$^{\text{RSYNC}}$	$\perp$	FCOM$^{\text{ASYNC}}$	SRO, VSS	Ninomiya et al. (2026) Table 1; Thm 3.5
OBLOT$^{\text{RSYNC}}$	$\perp$	FSTA$^{\text{SSYNC}}$	SRO, CFSM	Ninomiya et al. (2026) Table 1; Thm 3.5
OBLOT$^{\text{RSYNC}}$	$\perp$	FSTA$^{\text{Move-atomic}}$	SRO, CFSM	Ninomiya et al. (2026) Table 1; Thm 3.5

Claude (Anthropic) was used in cataloguing the problems.

Model		Model	Separator	Source
OBLOT$^{\text{RSYNC}}$	$\perp$	FSTA$^{\text{ASYNC}}$	SRO, CFSM	Ninomiya et al. (2026) Table 1; Thm 3.5
FCOM$^{\text{ASYNC}}$	$\perp$	OBLOT$^{\text{SSYNC}}$	VSS, MCv	Ninomiya et al. (2026) Table 1; Thm 3.5
FSTA$^{\text{Move-atomic}}$	$\perp$	OBLOT$^{\text{SSYNC}}$	CFSM, MCv	Ninomiya et al. (2026) Table 1; Thm 3.5
FSTA$^{\text{ASYNC}}$	$\perp$	OBLOT$^{\text{SSYNC}}$	CFSM, MCv	Ninomiya et al. (2026) Table 1; Thm 3.5
FSTA$^{\text{ASYNC}}$	$\perp$	OBLOT$^{\text{Move-atomic}}$	CFSM, DMSD	Ninomiya et al. (2026) Table 1; Thm 3.5

Separating problems

RDV: Rendezvous
Two robots must, within finite time, occupy the same point (not fixed in advance) and stay there. (Suzuki and Yamashita 1999; Flocchini et al. 2016a)

OSP: Oscillating Points
Two robots, from any distance d(t0) > 0, alternately get strictly closer (monotonically) and strictly farther (monotonically) forever. (Das et al. 2016)

¬IL: ¬IL
Three robots in a fixed initial pattern must form a second pattern (a moves 90° clockwise) and then an asymmetric third pattern (b moves 45° clockwise) whose formation requires b to tell a from c. (Flocchini et al. 2019b)

TAR(d): Triangle Rotation
Three robots forming triangle ABC (circumcircle C, d known) must form A'B'C' on the same circumcircle with |AA'| = |BB'| = |CC'| = d. (Flocchini et al. 2019b)

TAR(d)*: Perpetual Triangle Rotation
TAR(d) repeated forever from each new triangle. (Naito et al. 2025)

CGE: Center of Gravity Expansion
Every robot moves directly from (x,y) to (⌧2x − c_x⌧, ⌧2y − c_y⌧), (c_x, c_y) the initial centre of gravity, then stops. (Flocchini et al. 2019b)

CGE*: Perpetual CGE
CGE repeated forever from each new configuration. (Flocchini et al. 2019b)

SRO: Shrinking Rotation
Two robots repeatedly move so that the segment either rotates 90° clockwise about its midpoint or rotates 45° and shrinks by $1/\sqrt{2}$, always staying inside the square of the previous step. (Flocchini et al. 2019b)

CYC: Cyclic Circles
n ≥ 3 robots: n−1 on a circle, one at the centre; the centre robot must visit, in cyclic order, the 2^{n−1} positions at distances d(i) (d non-invertible) along a fixed radius, returning to the centre between visits. (Buchin et al. 2025)

MLCv: Monotone Line Convergence
CLCv with the distance between the robots never increasing. (Flocchini et al. 2023)

MCv: Monotone Convergence
Two robots converge to a common point without ever increasing their distance (no line constraint). (Kitamura et al. 2025)

TF: Trapezoid Formation
Four robots on a convex quadrilateral with one longest side CD: stay still if it is a trapezoid; otherwise exactly one of a, b (chosen by whether angle $\alpha \geq \pi/4$) moves perpendicular to CD to make ab ∥ CD. (Flocchini et al. 2023)

Claude (Anthropic) was used in cataloguing the problems.

GCNCL: Get Closer but Not too Close on Line
Two robots approach each other on their segment, never increasing distance, and stop at final distance in [d0/2, d0). (Flocchini et al. 2023)

SM: Single Move
Each of two robots makes exactly one non-null move (attributed to Flocchini et al. 2023 by Naito et al. 2025 and Kitamura et al. 2025, but not in its arXiv or LIPIcs text). (Naito et al. 2025; Kitamura et al. 2025)

DMSD: Dyadic-Multiple Stopped-Distance
Whenever both of two robots are stopped, their distance is a dyadic-rational multiple of the initial distance. (Kitamura et al. 2025)

RDAM: Rendezvous in one move or Asymmetric Move
RDV1 (meet after exactly one move each) or AM (one robot moves ≤ 1 time and stops, the other moves ≥ 2 times). (Kitamura et al. 2025)

ETE: Exponential Times Expansion
n ≥ 3 robots (circle + centre) must form a sequence of 2^{n−1} expansion patterns determined by a non-invertible function (details in paper). (Naito et al. 2025)

HET: Hexagonal Edge Traversal
Two robots on opposite vertices of a regular hexagon must perform a prescribed cyclic 13-step traversal (edge to adjacent vertex, diagonal, return). (Naito et al. 2025)

VEC: Vertical Configuration
SRO-like two-robot rotation problem with an extra 'stay' option and conditions 1–5 (see paper). (Naito et al. 2025)

ZCC: Zig-Zag Collision-free Convergence
Two robots converge collision-free while oscillating along their line (alternately approaching and receding from the start). (Naito et al. 2025)

LP–Cv: Leave the Place – Convergence
Each robot first moves away from the other along their line, then they converge. (Naito et al. 2025)

LCM–M: Least Common Multiple – Movement
Three collinear robots A, B, C with AB:BC = 2:1; B must twice move away from A by d_AB, C once away from A by d_AC. (Naito et al. 2025)

OC: Oscillating Configurations
Four robots must cycle forever through configurations I-II-III-II-I-... (Das et al. 2021b)

IOP: Independent Oscillating Problem
Three collinear robots; each end robot independently oscillates between distance x_i and 2x_i from the middle robot whenever activated. (Das et al. 2021b)

EPG*: Perpetual Expansion Preserving Gravity
Two robots must forever double the segment joining them while keeping its midpoint at the initial midpoint, read at the times when neither robot moves; replaces CGE* under variable disorientation. (Ninomiya et al. 2026)

CFSM: Collision-Free Single Move
Each of two robots makes exactly one non-null move, and the robots never occupy the same point, even while moving. (Ninomiya et al. 2026)

VSS: Visit Swap and Stop
With $A = 2a_0 - b_0$ and $B = 2b_0 - a_0$, every move ends in $\{A, a_0, b_0, B\}$; the robots first form (B, b_0), (a_0, A) or (B, A) with move counts $(1, 0)$, $(0, 1)$ or $(1, 1)$, then stop forever in one of (a_0, b_0), (B, b_0), (a_0, A), (B, A). (Ninomiya et al. 2026)

Fan: Fan
$n + 2$ robots on a scalene right triangle, $n > 3$ of them on the right-angle vertex and one on each other vertex: forever, the robots on the right-angle vertex must move to distinct points and come back, each robot returning every time to the same point; the robots on the other two vertices never move. (Feletti et al. 2026b)

Claude (Anthropic) was used in cataloguing the problems.

SquareCenter: Square Center
Four robots a, b, c, d on a kite with symmetry axis ad, a right angle at a and an acute angle at d: d moves along the axis to complete a square with a, b, c, then one robot other than d moves to the centre of the square; no other move is allowed. (Feletti et al. 2026b)

Pinwheel: Pinwheel
Two similar acute scalene triangles with the same orientation and circumcentre, the inner one with one robot per vertex, the outer one with $m \geq 3$ robots in total on its vertices, and $m + 1$ robots on the circumcentre: forever, the robots on the outer triangle move along its edges to the next vertex in the clockwise order fixed by the inner triangle, without stopping on the way; no other move is allowed. (Feletti et al. 2026b)

HalveTwice: Halve Twice
Three collinear robots, two on one point and the third at distance d0 from it: the third robot must move along the line and stop forever at distance d0/2 or d0/4, stopping at no other distance; no other move is allowed. (Feletti et al. 2026b)

HalveOnce: Halve Once
As HalveTwice, but the third robot must stop forever at distance d0/2, with no other stop. (Feletti et al. 2026b)

Other settings

- **Limited visibility.** Under FSYNC and SSYNC, every model with full visibility is stronger than the same model with limited visibility, FSYNC is stronger than SSYNC under limited visibility, and a model with limited visibility under FSYNC is incomparable with the same model with full visibility under SSYNC (Das et al. 2024).
- **Opaque robots.** For every memory model and scheduler, opaque robots are weaker than transparent ones; for opaque robots, OBLOT and FSTA are stronger under SSYNC than under ASYNC, and whether LUMI is equally strong under SSYNC and ASYNC is open (Feletti et al. 2025).
- **Robots on graphs.** On graphs embedded in the plane the FSYNC/SSYNC landscape is mapped; one separator uses a crash fault (Sharma et al. 2025a).
- **Bounded asynchrony.** With limited visibility and imprecise measurements, oblivious robots are stronger under k-bounded asynchrony (k-ASYNC) than under ASYNC, separated by cohesive convergence; whether k-bounded asynchrony is weaker than SSYNC is open (Kirkpatrick et al. 2024).
- **Sequential schedulers.** Oblivious robots activated one at a time are stronger than under SSYNC and incomparable with FSYNC (Flocchini et al. 2026). Among sequential schedulers (any order SEQ, permutations PERM, round-robin RROBIN, and the restricted R-SEQ and R-PERM, which never activate a robot twice in a row), round-robin is strictly the strongest for every memory model; with full lights all the others are equivalent; for oblivious robots each further restriction (to permutations, or to never activating a robot twice in a row) strictly adds power, and the same holds with external lights except that R-SEQ and R-PERM are equivalent (Feletti et al. 2026b).
- **Robots on graphs: computation.** For oblivious FSYNC robots with strong multiplicity detection on a finite graph, unlimited local computation gives one algorithm that, for every formation problem given by its set of final configurations, decides whether the current configuration is solvable and, if it is, reaches a final configuration in the minimum number of rounds that the adversary can force; the authors argue that the method does not carry over to SSYNC (fairness) or ASYNC (pending moves), and that algorithms on graphs should be required to run in polynomial time (Cicerone et al. 2025c).

Where the papers disagree

- Rendezvous by FSTA robots in SSYNC (Flocchini et al. 2016a) needs each robot's unit of distance to stay fixed; several separations use it, and under variable disorientation it is impossible (Wada et al.

2026).

- One paper uses rendezvous by FSTA robots in ASYNC without proof, under variable disorientation where it is impossible; five FSTA-versus-OBLOT relations rest on it (Flocchini et al. 2023; Wada et al. 2026). $\text{FSTA}^A > \text{OBLOT}^A$ still holds by another separator (Das et al. 2021b).
- The Single Move problem is attributed to (Flocchini et al. 2023) by (Kitamura et al. 2025), (Naito et al. 2025) and (Ninomiya et al. 2026), but neither version of that paper contains it.
- Center of Gravity Expansion (CGE, CGE*) prescribes landing points by a coordinatewise floor; under variable disorientation this is either not computable or not a predicate on the trajectory (Ninomiya et al. 2026) (Remark D.9). (Kitamura et al. 2025) Thm 6 uses CGE* to separate FSYNC from RSYNC for two robots; (Ninomiya et al. 2026) replace it by EPG*.
- Simulation costs disagree across papers: for LUMI under SSYNC simulated by FCOM under RSYNC, the colour counts reported for a simulated light with c colours are $O(c \cdot 2^c)$, $64c \cdot 2^c$ and $64c^2$ (Buchin et al. 2025; Nakajima et al. 2024; Kitamura et al. 2025).
- Two papers each claim the first separation of SSYNC from ASYNC for oblivious robots, under different assumptions: limited visibility with imprecision (Kirkpatrick et al. 2024), and full visibility (Flocchini et al. 2023).

Open questions

Q1. Is there a problem separating k-bounded asynchrony from SSYNC (Kirkpatrick et al. 2024)?

Q2. The landscape without chirality and rigid moves, especially under ASYNC (Flocchini et al. 2019b; Kitamura et al. 2025).

Q3. Can FSTA robots solve rendezvous in ASYNC with a fixed unit of distance (Flocchini et al. 2016a; Wada et al. 2026)?

Q4. Is LUMI equally strong under SSYNC and ASYNC for opaque robots (Feletti et al. 2025)?

Q5. Which model classes have no problem solvable there but in no strictly weaker model (the "white regions") (Naito et al. 2025)? For two robots with variable disorientation, chirality and rigid moves there are none: every upward-closed set of the 13 classes is the exact solvability set of some task (Ninomiya et al. 2026), a preprint (the exactness proofs, its Appendices F and G, were not checked here). The question is open for three or more robots.

Q6. Schedulers beyond the atomicity-defined ones, and robots with energy constraints (Flocchini et al. 2023; Buchin et al. 2025).

Q7. The relations of FSTA under the sequential schedulers other than RROBIN, and the relations across memory models among sequential settings, e.g. FSTA under R-SEQ against OBLOT under R-PERM (Feletti et al. 2026b).

The problems of this model are in Part IV: Formation problems, Symmetry breaking and spreading.

Chapter 4

Search in geometric domains

Robots with unit speed move continuously in a known domain (a line, rays, a disk, a polygon) to find a target or an exit whose position is unknown.

What is optimised

The measure is **worst-case time**: for search on unbounded domains, the competitive ratio (time divided by the distance to the target); for evacuation of a bounded domain, the time until the last robot reaches the exit. The questions are exact constants, the price of weaker communication, and the price of faults.

Robots start together (at the origin of a line or rays, or at the centre of a disk or polygon) and move with at most unit speed. A target (or exit) lies at an unknown point; a robot recognises it only when it reaches it. Robots know the domain. In **search** the task ends when the first robot reaches the target; in **evacuation** it ends when the last robot reaches the exit. Robots exchange information either only when they meet (**face-to-face**) or instantly at any distance (**wireless**); some models allow one-way communication.

How the axes work. As in the other models, level 0 assumes nothing and higher levels are stronger assumptions.

4.1 How the assumptions fit together

From the cow path to evacuation

Bellman posed the linear search problem in 1963 and Beck studied it in 1964, with the target placed according to a known probability distribution (Bellman 1963; Beck 1964) (as reported by Czyzowicz et al. 2021a). Baeza-Yates, Culberson and Rawlins recast it as searching an unbounded domain where each probe costs "proportional to the distance of the next probe position relative to our current position": doubling the turning point (their *linear spiral search*) finds a point at distance d within $9d$ steps, and no algorithm does better up to lower-order terms (Baeza-Yates et al. 1993). Kao, Reif and Tate call it the *cow-path problem* (Kao et al. 1996). Group search sends several robots, and evacuation, posed for a disk by Czyzowicz et al. at DISC 2014, ends only when the last robot reaches the exit (Czyzowicz et al. 2014). Their trade-off is between "spreading sufficiently the robots so that they can find the exit point fast in parallel" and "not to spread them too far" (Czyzowicz et al. 2014). Czyzowicz, Georgiou and Kranakis survey this line of work (Czyzowicz et al. 2019d).

Communication

Once several robots search, what they can tell each other changes the answer. The first evacuation paper already compared robots that talk only when they meet with robots that broadcast at any distance, and showed "a clear separation between the local and the wireless communication models" (Czyzowicz et al. 2014). Later work considered asymmetric senders and receivers, whose evacuation time on the disk lies

strictly between the two (Czyzowicz et al. 2021d; Georgiou et al. 2022). Communication is therefore the first axis of the model.

Unreliable and unequal robots

Two further axes describe how the robots differ from the ideal. Faults: Czyzowicz et al. introduced crash-faulty searchers that fail to detect the target (Czyzowicz et al. 2019c) and studied Byzantine searchers that may also lie (Czyzowicz et al. 2021a). Since a lower bound for crash faults is also a lower bound for Byzantine faults, the two form a scale (Kupavskii and Welzl 2021). Speeds: patrolling (Czyzowicz et al. 2011) and two-robot search on the line (Chrobak et al. 2015; Bampas et al. 2019b) gave the robots different maximum speeds, and it turned out that a slow robot can help or be useless depending on the speed ratio (Bampas et al. 2019b).

Randomness and the problem axes

Randomness is the fourth axis: Kao, Reif and Tate showed that a randomized cow, whose expected distance is compared with the distance to a goal fixed in advance, does "almost twice as good as is possible with a deterministic algorithm" (Kao et al. 1996). Everything else differs from problem to problem and is kept on the problems as their own axes: the domain (line, m rays, disk, polygons, a fence), the number of robots, what counts as success (first arrival, last arrival, the queen's arrival, fetching a treasure), and how many exits there are.

4.2 Notation

Symbols of this model. Inside the model each symbol has one meaning; *papers write* gives other symbols found in the literature.

Symbol	Meaning	Papers write	Used in
k	number of robots	n	whole model
f	maximum number of faulty robots (crash or Byzantine)		whole model
m	number of rays sharing an endpoint		whole model
e	number of exits on the boundary		Chauffeuring a crashed robot
d	distance from the origin to the target (on a line, on rays or in the plane)		whole model
x	position of the target on the line		whole model
R	radius of the disk (results are usually stated for $R = 1$)		whole model
v, v_i	speeds of robots (robot i moves at speed at most v_i)	s, s_i	whole model
α	known arc distance along the boundary (between two exits, or between the treasure and the exit)	d	whole model
r	communication range of wireless robots		whole model
ℓ	length of the fence (segment or circle)		Patrolling
L	known distance between the starting points of two robots		Evacuation
p	index of an ℓ_p norm		Evacuation
w	weight of the second robot's arrival time in weighted group search		Evacuation

Claude (Anthropic) was used in cataloguing the problems.

Symbol	Meaning	Papers write	Used in
σ	number of servants in priority evacuation (the queen and σ servants, $k = \sigma + 1$ robots)	n	Evacuation
u	speed of a moving target, which moves at constant speed toward or away from the origin	v; (Coleman et al. 2026a) write u for the evasiveness $1/(1-u)$	Linear search (cow path), Search for a moving target

4.3 Assumption axes

Each axis says where the assumption comes from and why the community studies it, then defines its values. Every axis is a scale. Level 0 assumes nothing; each higher level is a stronger assumption, which makes a problem easier. Values on the same level are alternatives that do not imply each other. An algorithm designed for weaker assumptions works under stronger ones; a lower bound proved under stronger assumptions holds under weaker ones.

Communication

Czyzowicz et al. defined evacuation from the disk in two communication models, "non-wireless (or local)", where robots exchange information only at the same point at the same moment, and wireless, where a broadcast "is instantly acquired by other robots, independently of their current positions" (Czyzowicz et al. 2014). Their bounds for two and three robots, and for large k, showed "a clear separation between the local and the wireless communication models" (Czyzowicz et al. 2014). The earliest paper in this atlas with mixed abilities is by Czyzowicz et al.: senders transmit wirelessly but receive only face-to-face, and receivers do the opposite; such robots could be cheaper to build, or could be fully wireless robots whose sender or receiver module has failed (Czyzowicz et al. 2021d). For two robots on the disk, Georgiou, Giachoudis and Kranakis showed that the sender/receiver evacuation time lies strictly between the face-to-face and wireless times (Georgiou et al. 2022). Not every setting separates the levels: for two robots of very different speeds on the line, "wireless communication offers no advantage over communication by meeting" (Bampas et al. 2019b).

Level	Value	Formal definition
0	**face-to-face**	**Face-to-face**: two robots exchange information only when they are at the same point at the same time.
1	**sender/receiver**	**Sender/receiver**: one robot can only transmit and the other only receive, instantly at any distance.
2	**wireless**	**Wireless**: every robot can send messages to every other robot instantly at any distance.

Faults

Fault-tolerant group search began on the line in 2016. Czyzowicz, Kranakis, Krizanc, Narayanan and Opatrny introduced crash-type faults: a faulty robot follows its trajectory and "is indistinguishable from a reliable robot, except that a faulty robot does not detect the target while visiting its location" (Czyzowicz et al. 2019c). At ISAAC 2016 Czyzowicz et al. studied Byzantine robots, which "can fail to report the target even after reaching it, or ... make malicious claims about having found the target", so that search ends only when the reliable robots have "sufficient verification" (Czyzowicz et al. 2021a). Because a lower bound for crash faults is also a lower bound for Byzantine faults, the two levels form a scale (Kupavskii and Welzl 2021). Unreliable robots had appeared a year earlier in patrolling: at ISAAC 2015 Czyzowicz et al. studied patrollers that "fail to comply with their patrolling duties", so that only visits by reliable robots count (Czyzowicz et al. 2017e). The faults were later carried over to

Claude (Anthropic) was used in cataloguing the problems.

evacuation from the disk (Czyzowicz et al. 2017c). Georgiou, Giachoudis and Kranakis note that "fault tolerant group search has been considered less extensively mainly due to the complexity of the resulting optimization problem" (Georgiou et al. 2022).

Level	Value	Formal definition
0	**Byzantine**	Up to f robots may lie: stay silent when they find the target, or claim to have found it where it is not.
1	**crash faults**	Up to f robots may fail to detect or to report the target when they pass over it.
2	**no faults**	Every robot detects and reports the target.

Speeds

In the classical problems all searchers move at unit speed (Baeza-Yates et al. 1993; Czyzowicz et al. 2014). Czyzowicz, Gąsieniec, Kosowski and Kranakis studied patrolling of a segment and a circle by robots with distinct maximum speeds at ESA 2011, and proved the proportional partition strategy optimal on a segment for two robots (Czyzowicz et al. 2011). Kawamura and Kobayashi report that Czyzowicz et al. conjectured it optimal for every number of robots, and disproved this with six robots (Kawamura and Kobayashi 2015); see also (Kawamura and Soejima 2020). For search, Chrobak et al. solved two-robot evacuation on the line when the slower robot has at least a third of the faster one's speed (Chrobak et al. 2015), and Bampas et al. covered every speed ratio (Bampas et al. 2019b). Czyzowicz et al. motivate such variants by noting that "it is quite natural to consider collaborative tasks in which the participant robots have different capabilities" (Czyzowicz et al. 2020b); disk evacuation with two speeds followed (Lamprou et al. 2020).

Level	Value	Formal definition
0	**distinct speeds**	Each robot i has its own maximum speed v_i, known to all.
1	**equal speeds**	All robots move at most at unit speed.

Randomness

Bellman and Beck posed linear search with a target drawn from a known distribution and the expected time as the measure (Bellman 1963; Beck 1964) (as reported by Czyzowicz et al. 2021a). In the competitive formulation the target is adversarial, and every deterministic strategy has ratio at least 9 (Theorem 2.1 of Baeza-Yates et al. 1993). Kao, Reif and Tate gave the first randomized algorithm for the cow-path problem on several paths, SmartCow, which picks a random permutation of the paths and a random initial distance; for every goal at distance d fixed in advance its expected distance is at most $c\,d + O(1)$, with $c \approx 4.5911$ on the line, and they proved this optimal on the line (Kao et al. 1996). They note that Gal had given many of these results earlier in a game-theoretic setting (Kao et al. 1996). Randomization thus improves the ratio on the line by about half, which is why the catalogue keeps it as an axis. Most of the group-search and evacuation literature in this atlas is deterministic.

Level	Value	Formal definition
0	**deterministic**	Trajectories are fixed functions of the information received.
1	**randomized**	Trajectories may depend on random choices; the measure is the expected competitive ratio against an oblivious adversary.

The problems of this model are in Part V: Search, evacuation and patrolling.

Claude (Anthropic) was used in cataloguing the problems.

Chapter 5

Programmable matter

Many tiny particles with constant memory occupy nodes of the triangular lattice, move by expanding and contracting, and must self-organise into a shape or elect a leader.

What is optimised

The first question was **what constant-memory, anonymous particles can do at all** (leader election, shapes, coating); then **rounds**, with matching lower bounds where possible, and how weak a scheduler and how little orientation the algorithms can tolerate.

In the **amoebot model** (Derakhshandeh et al. 2014) each particle occupies one node of the infinite triangular lattice (contracted) or two adjacent nodes (expanded). A particle moves by expanding into an empty neighbouring node and contracting into one of its two nodes. Particles are anonymous, have constant memory, and communicate only with neighbours; in the founding announcement each particle is a finite-state machine that reads and writes one flag per incident edge, and a pair of neighbours may combine an expansion and a contraction into a *handover* (Derakhshandeh et al. 2014). In the circuit extension, particles also share reconfigurable circuits on which a beep reaches every particle of the circuit in the next round (Feldmann et al. 2022). An adversarial scheduler activates particles; time is measured in **asynchronous rounds**: intervals of time in which every particle is activated at least once (the epochs of the other models; they need not be the shortest such intervals, and a bound holds for every division of the execution into consecutive rounds). The canonical amoebot model separates the core model from families of assumption variants, including concurrency (Daymude et al. 2023).

How the axes work. As in the other models, level 0 assumes nothing and higher levels are stronger assumptions.

5.1 How the assumptions fit together

Matter that computes

Toffoli and Margolus coined *programmable matter* for matter that changes its physical properties under program control (Toffoli and Margolus 1991) (as reported by Di Luna et al. 2020e, Emek et al. 2019 and Derakhshandeh et al. 2014). Derakhshandeh et al. proposed the amoebot model in a brief announcement at SPAA 2014: particles on the triangular lattice that "have constant memory (in particular, they have no identifiers)", "act asynchronously and make their decision based on local information", and move by expanding and contracting, "which resembles the behavior of amoeba" (Derakhshandeh et al. 2014). The founding papers fixed many details at once: common chirality, one particle active at a time, random bits, constant memory and a structure that stays connected (Derakhshandeh et al. 2014, 2015b); the announcement already named compass, initialisation and fault tolerance as dimensions along which the model could vary (Derakhshandeh et al. 2014). Most axes of this model are the history of removing those defaults one by one.

Removing the defaults

Di Luna et al. listed the defaults behind the early shape-formation results (chirality, a sequential schedule and, "more important, randomization") and asked: "What can be done deterministically? Is chirality necessary? ... What happens if the scheduler is not sequential?" (Di Luna et al. 2020e). Their answers, and the leader-election papers that followed (Emek et al. 2019; Dufoulon et al. 2021; Chalopin et al. 2026b), gave the axes *orientation*, *scheduler*, *randomness* and *initial shape*. Chalopin, Das and Kokkou observe that every deterministic two-dimensional leader-election algorithm uses at least two extra assumptions from a short menu: randomness, simply connected shape, chirality, movement, a sequential scheduler (Chalopin et al. 2026b). Daymude, Richa and Scheideler then wrote the hierarchy down as the canonical amoebot model, with variants for orientation, memory, concurrency and fairness, because without it "precise systematic comparison of results under the amoebot model is difficult" (Daymude et al. 2023).

Weaker particles, looser structures

Two axes push below the original model. SILBOT particles keep no memory beyond being contracted or expanded, send no messages and act asynchronously with non-atomic actions, so a step depends only on what a particle sees within two hops (D'Angelo et al. 2020); this is the *oblivious* memory level and the *weak* scheduler. The *connectivity* axis records that the requirement that the structure stays connected at every moment is a convention of the literature rather than of the model: Dufoulon, Kutten and Moses Jr. disconnected and reconnected the structure on purpose to elect a leader faster (Dufoulon et al. 2021).

Faster communication

Reconfigurable circuits push the other way. Feldmann, Padalkin, Scheideler and Dolev observed that problems such as leader election need time of the order of the diameter of the structure, since "information only travels particle by particle", and let particles link pins on their edges into circuits on which a beep reaches every particle of the circuit in the next round (Feldmann et al. 2022). This is the *communication* axis. Circuit algorithms are stated for stationary particles that act in synchronous rounds, the *synchronous* value of the scheduler. With circuits a leader is elected in $\Theta(\log n)$ rounds with high probability (Feldmann et al. 2022), and in $O(\sqrt{\log n}\ \log n)$ rounds even when each particle misses a beep with constant probability (Warner and Scheideler 2026b).

Other mechanisms

Not every problem in this model is about amoebots. Problem-specific axes cover modules on the square grid that rotate or slide under central control (Michail et al. 2019), the hybrid model in which one finite-state agent moves passive tiles in two or three dimensions (Gmyr et al. 2020; Hinnenthal et al. 2025), and, for leader election, whether particles may move at all (Chalopin et al. 2026b). These axes apply to a single problem each. They are kept on the problem so that the model's own axes stay shared across problems.

5.2 Notation

Symbols of this model. Inside the model each symbol has one meaning; *papers write* gives other symbols found in the literature.

Symbol	Meaning	Papers write	Used in
n	number of particles (modules, tiles)		whole model
D	diameter of the configuration (largest lattice distance between two particles)		whole model
L_{out}, L_{max}	length of the outer boundary and of the longest boundary of the configuration		Leader election in programmable matter
λ	bias towards having more neighbours (stochastic approach)		whole model
γ	bias towards neighbours of the same colour (separation)		Separation of coloured particles
α	compression factor: perimeter at most α times the minimum		Compression
q	number of orientations (alignment)		Alignment
s	number of energy sources		Energy distribution
k	symmetricity of a configuration (a k-symmetric configuration admits up to k leaders)		Leader election in programmable matter

5.3 Assumption axes

Each axis says where the assumption comes from and why the community studies it, then defines its values. Every axis is a scale. Level 0 assumes nothing; each higher level is a stronger assumption, which makes a problem easier. Values on the same level are alternatives that do not imply each other. An algorithm designed for weaker assumptions works under stronger ones; a lower bound proved under stronger assumptions holds under weaker ones.

Orientation

In the geometric amoebot model of Derakhshandeh et al., each particle labels its six ports clockwise from an arbitrary edge, so all particles share chirality but not a direction (Derakhshandeh et al. 2015b); the 2014 announcement numbers the edges counter-clockwise and calls this the "no-compass model where particles only share their sense of orientation", as opposed to a full-compass model (Derakhshandeh et al. 2014). Di Luna et al. asked whether chirality is necessary and showed that it is "computationally irrelevant for shape formation" from simply connected configurations (Di Luna et al. 2020e). Emek et al. elected a leader with no orientation at all, using a procedure by which the particles reach common chirality (Emek et al. 2019). Chalopin, Das and Kokkou replaced chirality by agreement on one lattice direction and found that the difficulties are different: a unique stationary leader can then be elected, but explicit termination is impossible (Chalopin et al. 2026b). The canonical amoebot model lists all four levels: assorted, common chirality, common direction, and common direction and chirality (Daymude et al. 2023).

Level	Value	Formal definition
0	**no orientation**	Each particle labels its six incident directions in its own clockwise or counter-clockwise order, starting anywhere.
1	**chirality**	All particles agree on clockwise order, not on a common direction.
1	**common direction**	All particles agree on one lattice direction, not on handedness. Chalopin, Das and Kokkou call this a *common direction* (Chalopin et al. 2026b), but the same words mean agreement on all six directions in other papers (Chalopin et al. 2026a; Navarra and Piselli 2023b).
2	**common compass**	All particles agree on all six directions.

Scheduler

The early amoebot papers assumed "the standard asynchronous computation model, i.e., only one particle can be active at a time" (Derakhshandeh et al. 2015b); in the 2014 announcement, from the set of enabled actions "a single action is arbitrarily chosen and executed" (Derakhshandeh et al. 2014). Di Luna et al. dropped this for a strong scheduler that activates many particles but executes each activation atomically (Di Luna et al. 2020e). Dufoulon, Kutten and Moses Jr. use *strong* for atomic activations and *weaker* for non-atomic ones, under which, they report, deterministic election of a unique leader becomes impossible in some cases (attributed by them to Bazzi and Briones and to Di Luna et al.) (Dufoulon et al. 2021; Bazzi and Briones 2019). Daymude, Richa and Scheideler observed that "much of the existing literature implicitly assumes amoebot actions are isolated and reliable, reducing analysis to the sequential setting", whereas "real programmable matter systems are concurrent"; their canonical model orders the concurrency variants from asynchronous through synchronous and k-isolated to sequential (Daymude et al. 2023). The fully asynchronous SILBOT model sits at the weak end: all operations are "non-atomic", with possible delays between sensing, computing and acting, so an action may be based on an obsolete observation (D'Angelo et al. 2020). In the circuit extension "the amoebot structure is assumed to progress in synchronized rounds" (Feldmann et al. 2022); this is the *synchronous* value.

Level	Value	Formal definition
0	**weak scheduler**	An activation's look, compute and move need not be atomic; concurrent activations may interfere. The scheduler of Bazzi and Briones is described as weak sequential, as non-atomic, and as fair asynchronous by different later papers (Briones et al. 2023; Dufoulon et al. 2021; Chalopin et al. 2026b).
1	**strong scheduler**	Each activation is executed atomically; the adversary chooses the order, fairly.
2	**sequential**	At most one particle is active at any time.
2	**synchronous**	Time proceeds in rounds; in every round every particle executes one look–compute–act cycle, and what a particle sends in a round is received in the next (Feldmann et al. 2022). It is an alternative to the sequential scheduler: one activates every particle at once, the other one at a time.

Communication

Feldmann, Padalkin, Scheideler and Dolev introduced reconfigurable circuits, modelled on the nervous system, because leader election and similar problems need time of the order of the diameter when "information only travels particle by particle"; with circuits, stationary particles elect a leader in $\Theta(\log n)$ rounds with high probability (Feldmann et al. 2022). Circuit algorithms are stated for stationary particles

in synchronous rounds. Warner and Scheideler let each particle miss a beep with constant probability and keep most of the speed-up (Warner and Scheideler 2026b).

Level	Value	Formal definition
0	**local communication**	A particle exchanges constant-size information only with the particles on adjacent nodes, by messages or by reading and writing their memory, so information travels at most one particle per activation.
1	**circuits**	Each particle has a constant number of **pins** on every incident edge, shared with the neighbour across that edge, and in every round it links its pins into groups. A **circuit** is a connected component of the resulting graph on pins. A particle may send a **beep** on any of its circuits; every particle on the circuit receives it in the next round, without learning who sent it or how many did (Feldmann et al. 2022). Local communication remains available.

Memory

Constant memory is part of the amoebot model from its start: each particle runs a fixed finite program and reads and writes memory shared with its neighbours (Derakhshandeh et al. 2015b); in the 2014 announcement the state set and the flag alphabet on each edge are finite (Derakhshandeh et al. 2014). D'Angelo et al. introduced SILBOT at AAMAS 2020, in which particles are "asynchronous, without any direct means of communication (silent) and without memory of past events (oblivious)", to find "the minimal settings for PM under which it is possible to accomplish basic global tasks" (D'Angelo et al. 2020) (as described by Navarra and Piselli 2023b). Each SILBOT particle is "an automaton with two states, CONTRACTED or EXPANDED (they do not have any other form of persistent memory)" (D'Angelo et al. 2020). In exchange, particles see two hops instead of one, because "without communication, 1-hop visibility is not enough for a particle to decide whether it is a LEADER or not" (D'Angelo et al. 2020). The canonical amoebot model lists oblivious, constant, finite and unbounded memory as its memory variants (Daymude et al. 2023). Oblivious particles have since solved leader election, line formation, coating and self-stabilising election with holes (D'Angelo et al. 2020; Navarra and Piselli 2023b,a; Chalopin et al. 2026a).

Level	Value	Formal definition
0	**oblivious**	No persistent state and no messages; a particle decides from its view only.
1	**constant memory**	Each particle has $O(1)$ bits of persistent memory and reads or writes neighbours' memory.

Randomness

The first leader-election algorithm for amoebots used random bits to break symmetry (Derakhshandeh et al. 2015b), and Di Luna et al. listed randomization as the "more important" of the assumptions behind the early shape-formation results (Di Luna et al. 2020e). Deterministic algorithms then became a goal of their own: for a simply connected start up to k leaders are elected in a k-symmetric configuration (Di Luna et al. 2020e), and Emek et al. gave the first deterministic election of a unique leader using movement (Emek et al. 2019). A second use of randomness is the stochastic approach of Cannon, Daymude, Randall and Richa, who design a Markov chain whose stationary distribution favours the desired configurations and turn it into a local asynchronous algorithm (Cannon et al. 2016). The design is motivated by statistical physics, and the approach has been applied to separation, alignment and bridging (Cannon et al. 2019; Kedia et al. 2022; Arroyo et al. 2018).

Level	Value	Formal definition
0	**deterministic**	No random choices.
1	**randomized**	Particles flip fair coins; guarantees hold with high probability or in expectation.

Initial shape

The first leader-election algorithm for amoebots worked on every boundary of the structure, outer and inner, and its running time depends on the longest boundary (Derakhshandeh et al. 2015b) (as summarised by Dufoulon et al. 2021). Several early deterministic algorithms assumed the initial configuration has no holes: Di Luna et al. start from any simply connected configuration (Di Luna et al. 2020e) and Gastineau et al. assume chirality and no holes (Gastineau et al. 2018). Dufoulon, Kutten and Moses Jr. summarise that early deterministic algorithms that did not use movement relied on "assumptions on the initial shape ... (intuitively, that it had no 'holes')" (Dufoulon et al. 2021). Emek et al. removed the assumption by letting particles move (Emek et al. 2019), and Chalopin, Das and Kokkou extended self-stabilising election from simply connected systems to systems with holes (Chalopin et al. 2026a).

Level	Value	Formal definition
0	**holes allowed**	The initial set of occupied nodes induces a connected subgraph; it may enclose empty regions.
1	**no holes**	The initial configuration is connected and encloses no empty node. Also called *contractible* (Briones et al. 2023) or *hole-free* (Gastineau et al. 2018).

Connectivity during execution

The founding papers required the particles to stay connected at all times, "to prevent the particles from drifting apart in an uncontrolled manner like in fluids and because in our case particles communicate only via bonds" (Derakhshandeh et al. 2015b). Emek et al. still followed "the common practice" that the shape is connected at all times (Emek et al. 2019). Dufoulon, Kutten and Moses Jr. observed that the model itself does not require this, and gave a leader-election algorithm that disconnects and later reconnects, gaining a large speed-up in time (Dufoulon et al. 2021). In SILBOT, line formation also passes through disconnected configurations and proves that connectivity is recovered (Navarra and Piselli 2023b). In centralised transformations the same distinction separates transformations that may break connectivity from those that may not (Michail et al. 2019).

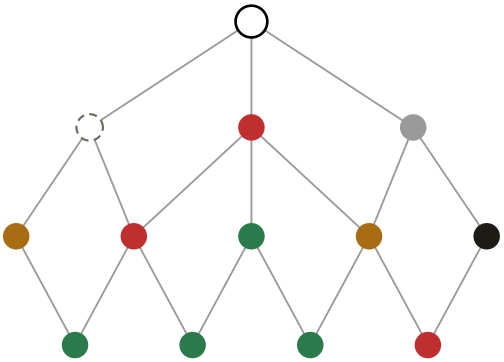
Level	Value	Formal definition
0	**stays connected**	At every time the occupied nodes induce a connected subgraph.
1	**may disconnect**	Connectivity is required only initially and at the end.

The problems of this model are in Part VI: Self-organising particle systems.

Claude (Anthropic) was used in cataloguing the problems.

Part II

Perspectives



Essays that connect the problems of the atlas across models: where they come from, why the papers say they matter, and how techniques and lower bounds move between them. They assume the models of Part I.

Claude (Anthropic) was used in cataloguing the problems.

Chapter 6

Why compute by moving

The atlas collects four models that grew out of four different wishes: software that travels through a network, swarms of robots that are as weak as possible, searchers that must find something at an unknown place, and matter that reshapes itself. They share an adversary, locality and symmetry; they differ in what they count.

Problems: Graph exploration by a single agent, Collaborative exploration, Dispersion, Rendezvous, Black hole search, Gathering, Pattern formation, Convergence, Linear search (cow path), Evacuation, Leader election in programmable matter, Shape formation in programmable matter.

The four models of the atlas (agents on graphs, robots in the plane, search in geometric domains, programmable matter) were built by different groups for different reasons, and their papers still open with different stories. What they have in common is the decision to abstract from the hardware. Pelc puts it this way for rendezvous: "Since rendezvous algorithms do not depend on the physical nature of the mobile entities executing them, but only on their perception capabilities, memory size, mobility characteristics and on the structure of the environment, we will not distinguish between natural and artificial agents" (Pelc 2019). Each model then fixes those four ingredients differently, and the choice is what makes a problem easy or hard in it.

This essay follows the introductions of the founding papers and surveys: what each community wanted when it started, and what it chose to measure. The other perspectives take single threads (meeting, faults, lower bounds) across the same four models.

In the graph model the entity is usually a program. Pelc's survey names "software agents, i.e., mobile pieces of software that navigate in a computer network in order to perform maintenance of its components or to collect data distributed in nodes of the network" (Pelc 2019), and later papers keep the same picture of "a software entity endowed with limited memory, capable of traversing a network and visiting nodes" (Kaur et al. 2026b). The oldest problem of the model is older than computer networks: Shannon built a mouse that explores a 5×5 maze using constant memory per vertex (Shannon 1951), as recalled by (Disser et al. 2019). Exploration also mattered to complexity theory. Fraigniaud et al. list "robot navigation, network maintenance, resource discovery, and WWW search" as applications and add that a finite set of finite automata exploring all graphs "would have put USTCON in L" (Fraigniaud et al. 2006c); Disser, Hackfeld and Klimm call exploration "a natural abstraction of a process of computing where nodes correspond to states" (Disser et al. 2019).

Later problems kept the network story but added a reason to move. Black hole search starts from security: a host that "disposes of visiting agents upon their arrival, leaving no observable trace", which can be a virus or simply "the undetectable crash failure of a site" (Di Luna et al. 2025b). Dispersion was introduced with self-driving electric cars looking for free charging stations, and from the start was placed next to scattering, exploration and load balancing (Augustine and Moses Jr. 2018). What the model counts follows from the picture of a small program on a large network: rounds or moves, the number of agents, and above all the bits of memory an agent carries.

The plane model starts from the opposite end: the robots are physical, but the question is how weak

they can be. “The study of swarms of mobile robots from an algorithmic perspective was initiated by Suzuki and Yamashita, and the simplicity of their model fostered many research results” (Bramas et al. 2023b), referring to (Suzuki and Yamashita 1999). The applied motivation is deployment where no infrastructure exists: “space, deep sea, or after some natural (or unnatural) disaster”, so that the group “must self-organize in the absence of any prior infrastructure (e.g., no global positioning)” (Clement et al. 2010). The research motivation is stated plainly: “The primary motivation in this field is to find the minimum capability required to achieve certain objectives for a system of robots” (Pattanayak et al. 2019b).

Hence the questions are of the form “under which conditions this task can be accomplished by the robots?” (Prencipe 2007). Robots are anonymous, silent and oblivious; what varies is synchrony, agreement on coordinates, multiplicity detection, visibility and, since lights were added, a constant number of persistent colours (Das et al. 2016). Flocchini, Santoro and Wada describe the programme as determining “the impact that internal capabilities (e.g., memory, communication) and external conditions (e.g. synchrony, activation scheduler) have on the solvability of a problem” (Flocchini et al. 2019b). The primary measure is therefore feasibility; colours come second, and time (in rounds or epochs) is often not analysed at all.

The search model comes from operations research rather than computing. Bellman posed the linear search problem in 1963 and Beck studied it in 1964; Kupavskii and Welzl quote Beck’s formulation, “A man in an automobile searches for another man who is located at some point of a certain road” (Kupavskii and Welzl 2021; Beck 1964). Computer scientists rediscovered it in the late 1980s as the cow-path problem, with the worst-case (competitive) ratio as the measure, and the doubling strategy with ratio 9 is optimal (Kupavskii and Welzl 2021; Baeza-Yates et al. 1993). Brandt et al. state the method directly: “The standard toolkit to analyze this class of problems is competitive analysis” (Brandt et al. 2020b).

The distributed version adds several robots and a communication model. Czyzowicz et al. describe what is abstracted away: in geometric environments “their performance is measured by the geometric distance travelled, most often disregarding their computing, communicating and environment-perceiving activities” (Czyzowicz et al. 2014). Evacuation, which they posed, asks for “a compromise between, on the one hand, spreading sufficiently the robots so that they can find the exit point fast in parallel, and, on the other hand, not to spread them too far” (Czyzowicz et al. 2014). The robots know the domain and each other’s trajectories; what they lack is the location of the target. The answers are exact constants, and the open problems are gaps between two constants.

Programmable matter begins with a vision rather than an application: “a material that can dynamically alter its physical properties in a programmable fashion, controlled either by user input or its own autonomous sensing of its environment” (Daymude et al. 2023). The term is attributed to Toffoli and Margolus by (Di Luna et al. 2020e) and (Dufoulon et al. 2021); Di Luna et al. list smart materials, monitoring and repair, and minimally invasive surgery as envisioned uses (Di Luna et al. 2020e). The amoebot model, introduced at SPAA 2014 (Derakhshandeh et al. 2014), is “motivated by micro- and nano-scale robotic systems with strictly limited computational and locomotive capabilities” (Daymude et al. 2023). Dufoulon, Kutten and Moses note that smart particles “need to be small. Thus, they are also weak in their memory size, computational power, communication ability, movement ability” (Dufoulon et al. 2021).

The first problems were shape formation and leader election, and their link was there from the beginning: “most shape formation algorithms depend on some seed element”, and the founding paper asks “whether shape formation can even be solved in circumstances where leader election is not possible” (Derakhshandeh et al. 2015b). Later models removed more: SILBOT has no communication and no

Claude (Anthropic) was used in cataloguing the problems.

memory, to find "the minimal settings for PM under which it is possible to accomplish basic global tasks" (Navarra and Piselli 2023b). The measures are rounds (under an adversarial scheduler) and the total number of moves; a centralised line of work asks what transformations are possible at all under given movement mechanisms (Michail et al. 2019).

Three ingredients recur. The first is an **adversary**: in a deterministic setting "the initial positions of the agents are chosen by an adversary which models the worst-case situation" (Pelc 2019), and the adversary also picks the port numbers, the activation schedule, the moving edges of a dynamic graph, or the exit of a disk. The second is **locality**: an agent sees only its node and its ports, an oblivious robot sees a snapshot but no history, a particle sees its neighbours, and a searcher recognises the target only when it stands on it. The third is **symmetry**: identical entities in symmetric positions cannot be separated by any deterministic rule. Ta-Shma and Zwick give the standard example, two identical robots in a ring whose ports are all oriented the same way: "they would never meet!" (Ta-Shma and Zwick 2014).

What each model counts follows from its origin. Agents on graphs count rounds, moves, team size and memory bits, because the entity is a program in a network. Robots in the plane first ask whether a task is solvable at all, then how many colours it needs. Search counts distance travelled, as a competitive ratio with exact constants. Programmable matter counts rounds and moves of a very large number of constant-memory particles. Bampas et al. make the contrast explicit within one model: in search, cost is "the time when the first searcher arrives at the target", while in rendezvous it is "the time when the last involved robot reaches the meeting point" (Bampas et al. 2019b). The same task, read with a different measure, becomes a different problem, and that is why the atlas keeps the four models apart while linking their problems.

Claude (Anthropic) was used in cataloguing the problems.

Chapter 7

Symmetry is the adversary

Identical agents running the same deterministic program can be kept in lockstep forever. Across graphs, the plane and programmable matter, the papers break this symmetry with identifiers, with randomness, with asymmetric starts, or with the scheduler, and leader election keeps reappearing as the common core.

Problems: Rendezvous, Gathering in graphs, Gathering, Scattering, Dispersion, Leader election in the plane, Pattern formation, Universal pattern formation, Leader election in programmable matter, Leader election by agents.

Pelc's survey of deterministic rendezvous names the obstacle directly: "The main problem that has to be solved in order to make deterministic rendezvous possible, both in networks and in the plane, is breaking symmetry" (Pelc 2019). His example is an oriented ring, port 0 clockwise at every node, with two identical agents that start at the same time: whatever one does, the other does too, so the distance between them never changes. The same holds in the empty plane for agents with the same compass, unit and speed (Pelc 2019). Ta-Shma and Zwick make the same point when they give their rendezvous agents labels: without them "the rendezvous problem cannot be solved deterministically" (Ta-Shma and Zwick 2014).

The argument travels to every model in the catalogue. For dispersion, Das, Bose and Sau spell it out: anonymous agents that start on one node are in the same state and run the same algorithm, so "they will always mirror each other's move" and never separate; the same reasoning gives the $\Omega(\log k)$ bits per agent, for k agents, that every deterministic dispersion algorithm needs (Das et al. 2023; Augustine and Moses Jr. 2018). In the plane, two oblivious robots cannot gather under a semi-synchronous scheduler (Flocchini et al. 2016a) (first shown by Suzuki and Yamashita Suzuki and Yamashita 1999, as the catalogue records it), and scattering robots that may start on a common point has no deterministic solution (Dieudonné and Petit 2009). Clement, Défago, Potop-Butucaru, Izumi and Messika put it generally: because robots are anonymous, "the impossibility results proved in classical distributed computing also hold in robot networks", and flocking, gathering or scattering are impossible without additional assumptions (Clement et al. 2010).

Each model has its own notion of symmetry, and the strongest results characterise it exactly. In port-labelled graphs the notion is the *view* of a node, the infinite tree of port sequences seen from it (introduced by Yamashita and Kameda, as recounted by Pelc 2019). Two anonymous agents can meet if and only if their starting nodes have different views (Czyzowicz et al. 2012b), as stated in (Pelc 2019). For larger anonymous teams the asymmetry may come from the agents' positions relative to each other rather than from the graph, which Pelc singles out as the main difficulty solved by Dieudonné and Pelc (Dieudonné and Pelc 2016; Pelc 2019).

In the plane the notion is geometric. Dieudonné, Levé, Petit and Villain characterise the positions from which disoriented robots can deterministically elect a leader, with and without a common handedness, for every $n > 1$; both characterisations rest on Lyndon words built from the positions (Dieudonné et al. 2013). For fat robots, Gan Chaudhuri and Mukhopadhyaya show that a leader can be elected exactly from asymmetric configurations (Chaudhuri and Mukhopadhyaya 2015).

Dieudonné, Levé, Petit and Villain open their paper with the reason leader election matters: "many problems that are hard to solve become easier to solve with a leader to coordinate the system" (Dieudonné et al. 2013). For robots in the plane the link is an equivalence. Flocchini, Prencipe, Santoro and Widmayer showed that $n \geq 3$ robots that cannot elect a leader cannot form arbitrary patterns (as reported by Dieudonné et al. 2013). Dieudonné, Petit and Villain proved the converse in asynchronous systems for $n \geq 4$ with chirality and $n \geq 5$ without (Dieudonné et al. 2010). Cicerone, Di Stefano and Navarra later gave a full characterisation without chirality, arbitrary pattern formation being solvable exactly from leader configurations, and reported that several earlier proofs, including the arXiv version of (Dieudonné et al. 2010), are not correct (Cicerone et al. 2019).

Programmable matter repeats the pattern. Derakhshandeh et al. motivate leader election by observing that consensus "can easily be solved once the leader election problem can be solved" and that most shape-formation algorithms depend on a seed (Derakhshandeh et al. 2015b). Emek, Kutten, Lavi and Moses Jr. call it "an important primitive ... that subsequently breaks symmetry and coordinates the remaining amoebots", and their deterministic algorithm without chirality also makes the particles agree on a handedness (Emek et al. 2019). In the agent-based model on graphs, Kshemkalyani, Kumar, Molla and Sharma note that leader election "can also be seen as a form of symmetry breaking" and build minimum spanning trees, independent sets and dominating sets on the elected leader (Kshemkalyani et al. 2024).

The first way out is to give agents labels. Pelc lists it first: distinct labels, each known to its owner, make deterministic rendezvous possible in every connected graph, at a cost that grows with the length of the labels (Pelc 2019; Dessmark et al. 2006). In dispersion the identifiers are what the memory pays for: the $\Omega(\log k)$ lower bound is the cost of storing a distinct identifier (Kshemkalyani et al. 2025b).

Between anonymous and fully labelled lies homonymy, which Delporte-Gallet et al. introduced for Byzantine agreement (Delporte-Gallet et al. 2013). Bouzid and Lamani brought it to robots, recalling that sequences of patterns are formable by labelled robots almost without restriction but by anonymous robots only when all patterns have the same symmetricity (Das et al. 2015b); this gap, they write, "illustrates how much anonymity can be a limiting factor" (Bouzid and Lamani 2011). Asahiro and Yamashita treat the number of distinct programs (target functions) as a resource: gathering needs two, and three once a robot may crash (Asahiro and Yamashita 2025). On graphs, Devismes, Dieudonné and Labourel argue that unique labels can be costly to maintain or undesirable for privacy, and that randomised alternatives give no absolute guarantees; they characterise the teams with repeated labels that can gather and gather all of them in time polynomial in the number n of nodes and in $\log \lambda$, where λ is the smallest label (Devismes et al. 2026b).

The second way out is randomness. Clement et al. remark that work on robot impossibilities had focused on extra assumptions rather than on randomisation (Clement et al. 2010). Scattering is the clean case: no deterministic algorithm exists (Dieudonné and Petit 2009), and Bramas and Tixeuil show that, to scatter n robots, $n \log n$ random bits are necessary in total, give a scattering algorithm that uses that many bits and runs in $O(\log \log n)$ rounds without strong multiplicity detection, and prove that no algorithm without strong multiplicity detection runs in constant expected time (Bramas and Tixeuil 2017). For dispersion, coin flips replace identifiers: anonymous randomised agents disperse with $O(\log \Delta)$ bits on graphs of maximum degree Δ (Das et al. 2023), matching the $\Omega(\log \Delta)$ bound of Molla and Moses Jr. (Molla and Moses Jr. 2019). The randomised amoebot algorithm of Derakhshandeh et al. elects a leader in expected time linear in the longest boundary (Derakhshandeh et al. 2015b).

The third way out lives in the scheduler. Under a sequential scheduler, which activates one robot per round, robots can break symmetry and elect a leader. Flocchini, Navarra, Pattanayak, Piselli and Santoro show that this power is "much stronger than the obvious capacity these schedulers offer of breaking

symmetry": every pattern except a point becomes formable from every start, which is impossible under the fully synchronous scheduler even for robots with a leader (Flocchini et al. 2026).

The cost of breaking symmetry is not settled even in the problems that are otherwise solved. Deterministic dispersion runs in optimal $O(k)$ rounds, but with $O(\log(k + \Delta))$ bits; whether $O(\log k)$ bits, the price of an identifier, suffice is open (Kshemkalyani et al. 2025b). Anonymous randomised agents disperse with optimal memory, but in $\Theta(k^2)$ rounds against an $\Omega(k)$ lower bound (Das et al. 2023). Whether anonymous randomised agents can disperse from an arbitrary configuration and then stop, rather than stay active forever, is open (Molla and Moses Jr. 2019; Das et al. 2023). And faults bring indistinguishability back: weak Byzantine agents can behave exactly like honest ones and occupy nodes, which makes balanced dispersion impossible whenever $\lceil k/n \rceil > \lceil (k - f)/n \rceil$ (Molla et al. 2021b).

Chapter 8

Finding your way

A single agent in an anonymous graph: how much memory it needs, what a pebble, an advice string or knowledge of n buys, why stopping is harder than exploring, and how treasure hunt asks for exploration only as far as the target. Each line of papers, in its own words, on why the question matters.

Problems: Graph exploration by a single agent, Treasure hunt, Rendezvous, Collaborative exploration.

The papers give three reasons, and they explain why the field measures memory rather than time. The first is practical. Fraigniaud, Ilcinkas, Peer, Pelc and Peleg: visiting all nodes "is fundamental in searching for data stored at unknown nodes of a network", traversing all edges is needed "in network maintenance and when looking for defective components", and since "robots and mobile agents are meant to be simple, often small and inexpensive devices", they ask for the least memory that suffices, "regardless of the time of exploration" (Fraigniaud et al. 2005a). They also explain the anonymous model: node labels may be absent or invisible to the agent, but ports must be distinguishable at a node, otherwise not even a star with three leaves can be explored (Fraigniaud et al. 2005a).

The second reason comes from complexity theory. Fraigniaud, Ilcinkas, Rajsbaum and Tixeuil recall that a finite set of finite automata able to explore all graphs "would have put USTCON in L", which motivated "quite a long sequence of studies" until Reingold placed undirected connectivity in log-space (Fraigniaud et al. 2006c; Reingold 2008). The third is a modelling view stated by Disser, Hackfeld and Klimm: exploration is "a natural abstraction of a process of computing where nodes correspond to states", so a single agent is a single processor and a team of agents is a parallel machine (Disser et al. 2019).

The history is told the same way in several introductions. Shannon's maze-solving machine of 1951 is the "first known finite automaton algorithm designed for graph exploration" (Fraigniaud et al. 2005a; Shannon 1951). Rabin claimed in 1967 that no finite automaton with finitely many pebbles explores all graphs; Budach gave the first formal proof for an automaton without pebbles, phrased for labyrinths (Budach 1978); and Rollik proved that no finite set of cooperating finite automata explores all cubic planar graphs, which implies Rabin's claim because an automaton is at least as strong as a pebble (Rollik 1980). In mazes the picture is kinder: Blum and Kozen showed that a finite automaton with two pebbles, or two cooperating automata, search every two-dimensional maze, because an automaton in a maze "has a compass: it can distinguish N,E,S,W"; on finite planar cubic graphs, by contrast, "no three finite automata together can search all" of them (Theorem 6.1) (Blum and Kozen 1978). The statements about Rabin, Budach and Rollik are as recounted by (Fraigniaud et al. 2005a) and (Disser et al. 2019) (Blum and Kozen also recount Budach's result); those originals were not read for this catalogue.

The modern question starts from those traps. Fraigniaud et al. build, for every K-state automaton, a planar trap with at most $K + 1$ nodes, and deduce that exploring all graphs of diameter D and maximum degree Δ needs $\Omega(D \log \Delta)$ bits; a depth-first search of depth $D + 1$ shows this is tight (Fraigniaud et al. 2005a). Two resources appear already in this early work and stay in all later work: memory inside the agent, and markers outside it.

On trees the answer depends on what the agent must do at the end. Perpetual exploration needs

and can do with $\Theta(\log \Delta)$ bits (Diks et al. 2004); a memoryless agent that always leaves by the next port after the one it entered also works, as Cohen et al. recall (Cohen et al. 2008). Exploration with return needs $\Theta(\log n)$ bits (Diks et al. 2004; Ambühl et al. 2011). Exploration with stop sits between $\Omega(\log \log \log n)$ and $O(\log n)$, and whether stopping is cheaper than returning is the open problem left by Diks, Fraigniaud, Kranakis and Pelc (Diks et al. 2004).

Knowledge of n changes what is possible. With an upper bound on n, a random walk with an $O(\log n)$-bit step counter explores any graph with high probability (Aleliunas et al. 1979), and Reingold's log-space universal exploration sequences make this deterministic with $O(\log n)$ bits (Reingold 2008). Disser et al. summarise the single-agent picture as $\Theta(\log n)$ bits, necessary and sufficient (Disser et al. 2019); the lower bound the catalogue records for agents that know n is $\omega((\log n)^{1-\varepsilon})$ bits, and whether the full $\Omega(\log n)$ holds when n is known is listed as open on the exploration page.

Visiting every node is not the same as knowing that every node has been visited. Without a bound on n and without any way to mark a node, a deterministic agent cannot explore with stop, whatever its memory (Dieudonné and Pelc 2012); already an even cycle cannot be explored with stop without a marker (Fraigniaud et al. 2006c). One pebble that marks the start restores it: depth-first phases of increasing depth find new nodes, and the agent stops after phase $D+1$, using $O(D \log \Delta)$ bits (Fraigniaud et al. 2005b). Fraigniaud, Ilcinkas, Rajsbaum and Tixeuil introduced the reduced-automata technique to prove lower bounds of this kind, and showed $\Omega(\log n)$ bits for exploration with stop and one pebble (Fraigniaud et al. 2006c).

Termination returns in the multi-agent problems. In dispersion, Sudo et al. make all agents stop in the rooted case only with global upper bounds on k and on the number of edges (Sudo et al. 2026a), and the only asynchronous dispersion algorithm known in which every agent terminates takes time exponential in the diameter (Kshemkalyani and Ali 2019); the dispersion variants list the open questions.

Cohen, Fraigniaud, Ilcinkas, Korman and Peleg view pebbles as the agent's ability "to dynamically label the nodes", and ask what happens if a designer labels the nodes beforehand (Cohen et al. 2008). Their answer is that 2-bit labels (three values) let an automaton with fewer than 25 states explore every graph and stop, and 1-bit labels suffice for graphs of bounded degree, although no finite automaton explores all graphs of maximum degree 3 unaided (Cohen et al. 2008). Ilcinkas asks the dual question for ports: since an adversarial port numbering defeats every finite automaton, how short can the period of a finite automaton be if the designer chooses the ports? Periodic exploration, he writes, is "particularly useful for network maintenance, where every node has to be regularly checked"; a three-state automaton achieves period $4n-2$ (Ilcinkas 2008), later improved to $3.5n-2$ against a lower bound of $2n-2$ (Czyzowicz et al. 2012c).

Disser, Hackfeld and Klimm settled the exchange rate between memory and markers: a constant-memory agent needs and can do with $\Theta(\log \log n)$ pebbles, even without knowing n, and the trade-off has two thresholds, no pebbles needed with $\Omega(\log n)$ bits and $\Omega(\log \log n)$ pebbles needed as soon as the memory drops to $O((\log n)^{1-\varepsilon})$ (Disser et al. 2019). At the far end, an automaton with a single state explores every connected graph with two pebbles, and two is the minimum (Pattanayak and Pelc 2024a).

Ta-Shma and Zwick introduced deterministic treasure hunt as "a version of the rendezvous problem in which one of the robots is completely passive" (Ta-Shma and Zwick 2014). They point out that the problem is easy if the treasure is present when the agent starts; the hard case is a treasure placed later, which must still be found within time polynomial in n, and it led them to strongly universal exploration sequences. Whether this is possible without backtracking (when the agent does not see its entry port) is still open in their words: "We do not know if an efficient solution is possible when backtracking is not allowed" (Ta-Shma and Zwick 2014). Xin improved the cover time of these sequences (Xin 2007), and

Claude (Anthropic) was used in cataloguing the problems.

in a 2026 preprint brought it within a polylogarithmic factor of the length of one ordinary universal exploration sequence on graphs of fixed degree (Xin 2026).

Later papers motivate the problem with rescue. Bouchard, Dieudonné, Labourel and Pelc: "When a miner is lost in a contaminated mine, it may have to be found by a robot, and the length of the robot's trajectory should be as short as possible" (Bouchard et al. 2023a). Cost is then measured against the distance d to the treasure, or $e(d)$, the number of edges near the start, not against the size of the graph; with labelled nodes they reach $O(e(d)\log d)$ against $\Omega(e(d))$ (Bouchard et al. 2023a). Gorain, Mondal, Nayak and Pandit note that depth-first search costs $O(\Delta^d)$ for a treasure at distance d and that urgent searches call for help (Gorain et al. 2022). Miller and Pelc measure that help as bits of advice from an oracle and give the trade-off between advice and cost (Miller and Pelc 2015b); Gorain et al. instead place pebbles once, so that many agents can reuse them (Gorain et al. 2022). Pattanayak and Pelc drop the assumption that degrees are finite, which they report all earlier treasure hunt and rendezvous algorithms made, motivated by graphs of very large degree such as social networks (Pattanayak and Pelc 2024b).

Further reading. Directed graphs, where the agent cannot walk back, are a separate branch: one pebble and $O(n\Delta_{out}\log n)$ bits suffice against an $\Omega(n\log\Delta_{out})$ lower bound, where Δ_{out} is the maximum out-degree (Fraigniaud and Ilcinkas 2004), and a known bound on n gives polynomial time (Bender et al. 2002). See the directed variants of exploration.

Claude (Anthropic) was used in cataloguing the problems.

Chapter 9

From exploration to dispersion, and beyond

How collaborative exploration asks how much a team speeds up a search, how dispersion arose as the case $k = n$ under weaker assumptions, what dispersion gives back to exploration, and why dispersing first has become the standard opening move for computing graph structures with agents.

Problems: Graph exploration by a single agent, Collaborative exploration, Dispersion, Leader election by agents, Maximal independent set by agents, Minimal dominating set by agents, Minimum spanning tree by agents, BFS tree by agents, Triangle and butterfly counting by agents.

Fraigniaud, Gąsieniec, Kowalski and Pelc posed collective tree exploration: k agents start at the root of an unknown tree and must visit every node as fast as possible (Fraigniaud et al. 2006a). The offline optimum is $\Theta(n/k + D)$, by spreading the agents along an Euler tour of the tree (Disser et al. 2020). Their algorithm is $O(k/\log k)$-competitive, which Disser, Mousset, Noever, Škorić and Steger call "only slightly better than the trivial upper bound of $O(k)$" of a single depth-first traversal (Disser et al. 2020; Fraigniaud et al. 2006a); without any information sharing, a factor k is unavoidable (Fraigniaud et al. 2006a). Cosson, Massoulié and Viennot frame the problem as "a fundamental problem behind the exploration of an unknown physical environment with a team of agents" and add: "Despite almost two decades of research, we still do not completely understand how much exploration time can decrease with the number of agents" (Cosson et al. 2023).

The lower bound side was opened by Dynia, Łopuszański and Schindelhauer, who showed $\Omega(\log k/\log\log k)$ for $k < \sqrt{n}$ (as reported by Disser et al. 2020); Disser et al. extended it to $k \le n \log^{\kappa} n$ for every constant κ (Disser et al. 2020). The upper bound moved from $O(k/\log k)$ to below $k/\log k$ (Cosson 2024b), to $O(\sqrt{k})$ (Cosson and Massoulié 2024), and in a 2026 preprint to $O(\log^2 k)$ (Cosson and Massoulié 2026). These results assume labelled nodes and, except in the whiteboard variants, free communication, so the agents share a map of what has been explored.

A second line asks what a team can do with very little memory, and here agents act as markers for each other. Rollik's traps show that no finite team of finite automata explores all graphs (Rollik 1980) (as recounted by Disser et al. 2019). Disser, Hackfeld and Klimm observe that a team of $p + 1$ agents can simulate one agent with p pebbles, one agent walking while the others stand still, and that before their work no algorithm explored undirected graphs with sublogarithmic memory using several agents or pebbles (Disser et al. 2019). Their result closes the question:

> *"Since an additional agent is at least as powerful as a pebble, this implies that $O(\log\log n)$ agents with constant memory can explore any n-vertex graph." (Disser et al. 2019)*

They also show that $\Omega(\log\log n)$ agents are needed even when each has $O((\log n)^{1-\varepsilon})$ bits (Disser et al. 2019). An agent that stays still is a marker; dispersion is built on the same idea.

Augustine and Moses Jr. introduced dispersion in 2018: k agents on an n-node graph must reach a configuration with at most one agent per node. Their motivation is self-driving electric cars that look for free charging stations instead of queueing, and they place the problem between scattering, exploration and load balancing from the first page (Augustine and Moses Jr. 2018):

> *"Since any solution to a problem of dispersion acts as a solution to n robot exploration under the same conditions, any possibility results achieved for dispersion apply to n robot exploration." (Augustine and Moses Jr. 2018)*

Lower bounds move the other way, so the $\Omega(D^2)$ bound for n agents on trees (Disser et al. 2020) is a dispersion lower bound. The catalogue records that this bound is existential in D (the hard trees have height about $\sqrt{\log n}$), so whether rooted dispersion on taller trees needs $\Omega(D^2)$ rounds is open.

The assumptions are much weaker than in collaborative exploration. Sudo, Shibata, Nakamura, Kim and Masuzawa single out "the unique computation model": no node identifiers, no memory at the nodes, agents with identifiers that talk only when they share a node (Sudo et al. 2026a). The algorithms recover the marker idea: in the depth-first dispersion of Kshemkalyani and Ali, "A robot that docks at a node helps other robots to determine whether they have visited this node before" (Kshemkalyani and Ali 2019). Every later improvement up to time-optimal $O(k)$ rounds with $O(\log(k + \Delta))$ bits, synchronous (Kshemkalyani et al. 2025b) and asynchronous (Pattanayak et al. 2026), is a faster way of using settled agents as markers on anonymous nodes.

Because dispersion with $k = n$ explores, the dispersion results are also the best known results for n agents exploring an anonymous graph with only local communication: $O(n)$ rounds and $O(\log n)$ bits per agent from a rooted start (Kshemkalyani et al. 2025b), an inference the catalogue makes and the paper does not state. With labelled nodes and global communication, n agents explore a tree in D^2 rounds (Disser et al. 2020); whether anonymity and local communication force $\Omega(n)$ rounds on shallow trees is open, and Augustine and Moses Jr. already asked for $o(n)$-round dispersion on trees (Augustine and Moses Jr. 2018).

Dispersion also gave exploration-type questions new resources to measure. Gorain, Mandal and Mondal took up the question of Bouchard, Dieudonné and Pelc whether gathering needs communication (Bouchard et al. 2023b) and asked it for dispersion: their silent agents convey information only by the timing of moves and by counts of co-located agents (Gorain et al. 2024). Molla and Moses Jr. replaced identifiers by coin flips (Molla and Moses Jr. 2019). Each such variant is, for $k = n$, a statement about collaborative exploration under the same assumptions.

Kshemkalyani, Kumar, Molla and Sharma describe the agent-based model by its two differences from message passing: the nodes have no identifiers, computation or storage, and devices can exchange information only when co-located, so a message to a neighbour requires a move (Kshemkalyani et al. 2024). With n agents on n nodes the model corresponds to message passing with one device per node, and their simulation theorem converts any message-passing algorithm that runs in R rounds into an $O(\Delta R \log n)$-round agent algorithm from a dispersed start (plus $O(n \log^2 n)$ rounds otherwise), provided n and Δ are known (Kshemkalyani et al. 2024). Their stated goal is to remove that knowledge, and they note that limited storage "makes it impossible for the relocatable devices to first traverse the graph to learn the topology and then run graph computation as a second step" (Kshemkalyani et al. 2024).

Dispersion is the first step in most of what followed. Chand, Molla and Sivasubramaniam built dominating sets on it: "Our algorithms rely on the ideas used for dispersion", taking "advantage of the fact that most dispersion protocols involve the exploration of G" (Chand et al. 2023). For triangle counting, Chand, Das and Molla state that a dispersed configuration "is a requirement" (Chand et al. 2026). Kaur, Gorain and Mondal reach maximal independent sets through distance-2 dispersion, which with n agents is the same problem, and elect a leader on the way (Kaur et al. 2026c). Chand and Molla build on time-optimal dispersion (Kshemkalyani et al. 2025b) to obtain, in $O(n)$ rounds, termination after dispersion, a spanning tree, a leader and a minimal dominating set (Chand and Molla 2026).

The progress is quick. Leader election without knowledge went from $O(m)$ rounds with $O(n \log n)$ bits

(Kshemkalyani et al. 2024) to $O(n \log^2 n)$ rounds with $O(\log n)$ bits (Kshemkalyani et al. 2025c) and to $O(n)$ rounds (Chand and Molla 2026). Maximal independent sets (Pattanayak et al. 2024; Kaur et al. 2026c), minimum spanning trees and BFS trees (Chand et al. 2025c), and triangle and butterfly counts (Chand et al. 2026, 2025b) followed the same route.

1. **Height versus size.** Is $\Omega(D^2)$ the right bound for n agents on a tree of every height? Exploration theory proves it for one height (Disser et al. 2020); dispersion uses it for all.
2. **Settled agents versus pebbles.** A pebble is a mute marker; a settled agent stores and exchanges $O(\log n)$ bits. How much of the $\Theta(\log \log n)$ pebble bound (Disser et al. 2019) changes when markers can talk?
3. **Anonymity versus labels.** The collaborative-exploration bounds assume labelled nodes and global communication; dispersion assumes neither. Where does anonymity start to cost time?
4. **Lower bounds for agent-based problems.** No time lower bound specific to leader election, independent sets, dominating sets or spanning trees by agents is recorded in the catalogue; the only bounds come from dispersion itself. Whether the $O(n)$ rounds of (Chand and Molla 2026) are optimal for leader election, in particular from a dispersed start where no dispersion is needed, is open.

Further reading. The open questions above are recorded with their variants: the rooted-tree and tree variants of dispersion, and the *anonymous graphs with $k = n$ agents* variant of collaborative exploration.

Claude (Anthropic) was used in cataloguing the problems.

Chapter 10

Meeting at one place

10

Rendezvous and gathering on graphs, gathering and convergence in the plane, and evacuation through an unknown exit all ask entities to end at one point. The papers give the same reasons for meeting, but each model makes a different part of it hard: symmetry and exploration on graphs, capability in the plane, information in continuous search.

Problems: Rendezvous, Gathering in graphs, Treasure hunt, Gathering, Convergence, Near-gathering, Evacuation, Linear search (cow path).

The reasons given are practical and much the same in every model. Robots exploring a terrain meet "to exchange samples of the ground previously collected by the robots, or exchange information obtained when exploring different parts of the terrain", and software agents meet periodically "to exchange collected data and plan further actions" (Pelc 2019). Bouchard, Dieudonné and Pelc add mobile robots "moving along corridors of a contaminated mine that is not accessible to humans", and give as reasons to gather samples or "to coordinate some future task, such as network maintenance or finding a map of the network" (Bouchard et al. 2023b). In the plane, gathering is called "one of the most fundamental tasks one can ask mobile robots to perform, e.g. to collect acquired data or to update the robots' algorithm" (Bramas et al. 2023b).

The second reason is that meeting is a building block. Gathering "turns out to be an essential prerequisite to achieve more complex cooperative works", so that results on gathering are results on everything built on it (Bouchard et al. 2022); Devismes, Dieudonné and Labourel use almost the same words (Devismes et al. 2026b). Clement et al. place gathering among agreement problems: like scattering, it is "similar to the consensus problem in conventional distributed systems", with simple definitions but solvability that "greatly depends on the synchrony of the systems as well as the nature of the faults" (Clement et al. 2010).

Pelc separates rendezvous settings by whether agents can see something other than the environment, other agents or marks (Pelc 2019). On graphs they usually cannot, and two conventions fix the difficulty. Nodes are anonymous because "if nodes had distinct labels, then agents might explore the graph and meet in the smallest node, hence gathering would reduce to the well-studied task of exploration"; ports are numbered because without them "the adversary could prevent an agent from taking some edge" (Bouchard et al. 2023b). Identical agents may never meet: two of them starting together on a ring with consistent ports run forever in parallel (Ta-Shma and Zwick 2014). So the labels of the agents are the symmetry breaker, and the cost is polynomial in n and the label length (Dessmark et al. 2006; Ta-Shma and Zwick 2014). In trees the cost is $\Theta(n + \log \lambda)$, where λ is the smaller label (Dessmark et al. 2006). Ta-Shma and Zwick also posed treasure hunt, rendezvous in which one agent never moves, as a tool for this question (Ta-Shma and Zwick 2014).

Without labels the question becomes which initial configurations can meet. Dieudonné and Pelc characterise the configurations of identical agents that can gather, and show that detecting that gathering is complete needs a bound on n (Dieudonné and Pelc 2016). For two identical asynchronous agents, meeting is feasible exactly when their views differ or their starting nodes are joined by a path whose port sequence is a palindrome (Guilbault and Pelc 2011), as summarised in (Pelc 2019). A middle ground is homonymous agents, motivated by cost and privacy: unique labels "can be costly", and agents "may

deliberately blur their identity by revealing it only partially" (Devismes et al. 2026b). With asynchronous agents the adversary controls speeds and agents may meet inside an edge; polynomial cost was reached only in (Dieudonné et al. 2015), after the ring case (De Marco et al. 2006).

Robots in the plane see each other, so finding the others is not the issue; agreeing on a point is. Prencipe states the research question as "under which conditions this task can be accomplished by the robots?", and shows that without extra assumptions such as multiplicity detection gathering is in general impossible (Prencipe 2007). Suzuki and Yamashita had shown that two oblivious robots cannot meet in SSYNC (Suzuki and Yamashita 1999), and Flocchini et al. explain why two robots matter: rendezvous "separates FSynch from SSynch for oblivious robots", whereas $n \geq 3$ robots gather even in ASYNC with multiplicity detection (Flocchini et al. 2016a; Cieliebak et al. 2012). Izumi et al. note that, for semi-synchronous robots, two-robot gathering is the single pattern that memory makes formable (Izumi et al. 2012).

That made rendezvous the test case for every new capability. Unreliable compasses solve it below a threshold deviation (Izumi et al. 2012). A light with two colours solves it in ASYNC (Das et al. 2016; Heriban et al. 2018), and three colours suffice when a robot sees only its own light (FSTA, in SSYNC) or only the other's (FCOM, even in ASYNC) (Flocchini et al. 2016a), and three colours are also necessary (Otaka et al. 2026). The measure throughout is feasibility and the number of colours, not time.

Relaxing the goal removes most of the difficulty. Izumi et al. remark in passing that "Two oblivious robots can converge to a point with a naive algorithm that consists of always moving toward the midpoint of their positions" (Izumi et al. 2012). Convergence of n robots was given by Suzuki and Yamashita (Suzuki and Yamashita 1999), with limited visibility by Ando et al. the same year (Ando et al. 1999), and the centre of gravity converges in ASYNC (Cohen and Peleg 2005).

With convergence and with limited visibility, time becomes measurable again, and the bounds are about locality. Under limited visibility in FSYNC, a class of local protocols needs and achieves $\Theta(\Delta_0^2)$ rounds, where Δ_0 is the initial diameter in units of the visibility range (Castenow et al. 2022); near-gathering, in which robots must come close without colliding (Pagli et al. 2015), takes $O(\Delta_0^2)$ epochs for disoriented robots in SSYNC, again tight within that class (Castenow et al. 2022). With rigid moves and a common compass, asynchronous gathering takes $\Theta(\Delta_0)$ epochs; with one common axis the same time only brings the robots onto a horizontal segment of length 1 (Poudel and Sharma 2021).

Evacuation fixes the meeting point in advance but hides it. Bampas et al. set the three measures side by side: in search the cost is the arrival of the first searcher at a target placed by an adversary; in rendezvous it is the arrival of the last robot, and "the meeting point is arbitrary, i.e., the robots may choose the most convenient one"; in their linear search problem "a pair of robots has to meet at an unknown fixed target point of the environment" and the cost is the arrival of the second (Bampas et al. 2019b). Evacuation is this third problem.

Czyzowicz et al., posing evacuation from a disk, describe the tension: robots should spread to find the exit fast but not so far that the walk to it becomes long (Czyzowicz et al. 2014). Communication decides how much the meeting costs. Two wireless robots evacuate in exactly $1 + 2\pi/3 + \sqrt{3} \approx 4.826$ (Czyzowicz et al. 2014); with face-to-face communication the best algorithm takes about 5.6234 (Disser and Schmitt 2019) against a lower bound of about 5.255 (Czyzowicz et al. 2020d). On the line, face-to-face robots that must all reach the target cannot beat the single-robot ratio 9 (Bampas et al. 2019b; Chrobak et al. 2015).

Read across the models, meeting is hard for three different reasons. On graphs agents cannot see each other, so meeting needs exploration and symmetry breaking, and the open questions are about cost: labelled rendezvous is polynomial, but the exponent is far from the lower bound (see the rendezvous page). In the plane robots see each other but cannot remember or agree, so the open questions are

Claude (Anthropic) was used in cataloguing the problems.

about which capabilities suffice and how many colours they need. In continuous search robots know the domain and each other's plans but not the target, so the open questions are gaps between constants.

Faults sharpen each version differently: f Byzantine agents on graphs require $f + 1$ good agents, or $f + 2$ when the number n of nodes is unknown (Bouchard et al. 2016); one Byzantine robot among three already prevents gathering in the plane in SSYNC (Agmon and Peleg 2006); and a faulty robot in a disk raises the evacuation time (Czyzowicz et al. 2017c). The perspective *When agents fail* follows that thread.



Claude (Anthropic) was used in cataloguing the problems.

Chapter 11

Weak robots in the plane

11

Why the theory of robots in the plane starts from oblivious, anonymous, disoriented points; how formation problems became its testbed and the scheduler its adversary; and how a light with a few colours turned into memory, communication and a measure.

Problems: Gathering, Pattern formation, Leader election in the plane, Circle formation, Uniform circle formation, Mutual visibility, Pattern sequences, Universal pattern formation.

Since the work of Suzuki and Yamashita (Suzuki and Yamashita 1999), the theory of robots in the plane has asked which *minimal* assumptions let a team solve a task, rather than how a capable robot should solve it (Flocchini et al. 2023). A robot is a point that repeats Look-Compute-Move cycles; it is anonymous, runs the same algorithm as everyone else, has its own coordinate system, and in the standard model OBLOT it is also oblivious and silent (Flocchini et al. 2019b). Dieudonné, Petit and Villain state the motive plainly: "The motivation behind such a weak and unrealistic model is the study of the minimal level of ability the robots are required to have in the accomplishment of some basic cooperative tasks in a deterministic way" (Dieudonné et al. 2008).

The introductions also give practical reasons. Many simple robots are cheaper and easier to extend than one strong robot (Bandettini et al. 2011); silent robots can be deployed where communication is impossible or jammed, under water or in a military setting (Di Luna et al. 2014); and robots that do not know n make the system "scalable and open to robot insertions or removals" (Feletti et al. 2023a). Flocchini, Prencipe, Santoro and Viglietta summarise the resulting research programme: the literature has "almost exclusively focused on very simple entities operating in strong adversarial conditions", robots that are anonymous, autonomous, oblivious, disoriented and non-rigid (Flocchini et al. 2017).

Almost all computability results in this model concern geometric pattern formation: the robots must reach, and stay in, a configuration similar to a given pattern (Flocchini et al. 2017). As Flocchini et al. recount, Suzuki and Yamashita observed that whether a pattern can be formed depends on the *symmetricity* of the pattern and of the start, so the only patterns that might be formable from every start are a point and a regular polygon (Flocchini et al. 2017). These two became the reference problems. A point is gathering: without multiplicity detection and without extra assumptions, oblivious robots cannot gather (Prencipe 2007), while with weak multiplicity detection $n \geq 3$ asynchronous robots gather from any start (Cieliebak et al. 2012). The regular polygon is uniform circle formation. Défago and Konagaya conjectured that the last step, spreading robots evenly on a circle, has no deterministic finite-time solution; Dieudonné, Petit and Villain proved the conjecture and still formed the circle in SSYNC by another route, for $n \notin \{4, 6, 8\}$ (Dieudonné et al. 2008). In ASYNC the question stayed open until Flocchini, Prencipe, Santoro and Viglietta solved it from every start of distinct points (Flocchini et al. 2017), with the square ($n = 4$) handled in (Mamino and Viglietta 2016); they conclude that for pattern formation "asynchrony is not a computational handicap" and that chirality and rigidity are "computationally irrelevant" (Flocchini et al. 2017).

Arbitrary patterns connect formation to symmetry breaking. Flocchini, Prencipe, Santoro and Widmayer posed arbitrary pattern formation for asynchronous oblivious robots (Flocchini et al. 2008a), and showed, as Dieudonné, Petit and Villain quote, that forming arbitrary patterns implies leader election for $n \geq 3$ (Dieudonné et al. 2010). The converse holds with chirality for $n \geq 4$ (Dieudonné et al. 2010) and, after a

careful re-examination of earlier proofs, without it (Cicerone et al. 2019); which configurations admit a leader is characterised by Dieudonné, Levé, Petit and Villain (Dieudonné et al. 2013). Later formation problems change one physical assumption at a time. Mutual visibility drops the assumption, shared by all earlier models, "that visibility is unobstructed" (Di Luna et al. 2014): opaque robots must reach a position in which no three are collinear (Di Luna et al. 2017). Pattern sequences ask the robots to form patterns one after another (Das et al. 2015b).

Because robots do nothing but look and move, the main adversary is time. In FSYNC all robots act in every round; in SSYNC a fair adversary picks which robots act in each round; in ASYNC each phase takes a finite but unpredictable time (Das et al. 2016). Asynchrony hurts because robots "can be seen while moving" and compute "on the base of obsolete information" (Cicerone et al. 2019); Flocchini, Santoro, Sudo and Wada name as "one of the strongest debilitating effects of unrestricted asynchrony" that "a robot, when looking, cannot detect if another robot is still or moving" (Flocchini et al. 2023). The adversary also chooses how far a robot moves, subject to a minimum distance δ (non-rigid moves) (Flocchini et al. 2017).

Rendezvous is the separating example: two oblivious robots meet trivially in FSYNC but cannot meet in SSYNC (Flocchini et al. 2016a), while $n \geq 3$ robots with multiplicity detection gather even in ASYNC (Flocchini et al. 2016a; Cieliebak et al. 2012). Between SSYNC and ASYNC lie finer schedulers, which make some of the phases atomic (LC-atomic, CM-atomic, M-atomic ASYNC), and below SSYNC lie restricted ones such as RSYNC and the sequential schedulers (Flocchini et al. 2023). Weakening the adversary can be decisive: under any sequential scheduler oblivious robots form every pattern from every start, except a point, which needs weak multiplicity detection (Flocchini et al. 2026).

The luminous model, which Das et al. attribute to a suggestion of Peleg, gives each robot a light with a constant number of colours that persists between cycles and is seen by all robots; a robot can therefore both remember and communicate a constant number of bits (Das et al. 2016). Das et al. asked what additional power "would allow the robots to overcome the inherent difficulties and weaknesses of the ASYNC setting". Their answer: with 5 colours ASYNC robots simulate any SSYNC algorithm, two robots with 4 colours meet in ASYNC, and with lights the difference between SSYNC and ASYNC disappears (Das et al. 2016).

Lights then split into two capabilities. Flocchini, Santoro and Wada ask directly, "is it better to remember or to communicate?" (Flocchini et al. 2019b). In FSTA the light is seen only by its owner (finite state, silent); in FCOM only by the others (finite communication, oblivious) (Flocchini et al. 2019b). Flocchini, Santoro, Viglietta and Yamashita had already shown that rendezvous does not need both: either kind of constant memory suffices under suitable conditions (Flocchini et al. 2016a). Under FSYNC, LUMI and FCOM are equivalent and stronger than FSTA, which is stronger than OBLOT; under SSYNC, FCOM and FSTA are incomparable (Flocchini et al. 2019b). Under asynchrony, SSYNC is strictly stronger than ASYNC for OBLOT, FCOM and FSTA, and only LUMI removes the difference (Flocchini et al. 2023). These separations mostly assume chirality and rigid moves. For two robots, LUMI, FCOM and FSTA coincide under FSYNC (Kitamura et al. 2025), a preprint.

Once lights were known to help, the number of colours became a resource. For two robots with full lights, the count fell from 4 (Das et al. 2016) to 3, optimal for algorithms that look only at colours (Viglietta 2013), and to 2, which is optimal, in ASYNC with non-rigid moves and arbitrary initial colours (Heriban et al. 2018). With external lights (FCOM), 3 colours suffice in SSYNC (Flocchini et al. 2016a) and 2 do not (Otaka et al. 2026); with internal lights (FSTA), 3 colours suffice in rigid SSYNC (Flocchini et al. 2016a; Wada et al. 2026). For $n \geq 3$ robots without multiplicity detection, full lights gather with 2 colours in SSYNC (Terai et al. 2023) and with 3 in ASYNC (Nakai et al. 2021). Opaque robots reach

Claude (Anthropic) was used in cataloguing the problems.

mutual visibility with 2 colours in rigid SSYNC, which is optimal (Di Luna et al. 2017), and 2 colours also suffice with non-rigid moves in SSYNC, with rigid moves in ASYNC, and with non-rigid moves in ASYNC when the robots agree on one axis (Sharma et al. 2015b).

Colours are then traded against time. For uniform circle formation by opaque luminous robots in ASYNC, the time fell from $O(n)$ epochs (Feletti et al. 2023b) to $O(\log n)$ (Feletti et al. 2023a), Pattanayak and Sharma traded colours for time, with $O(x)$ epochs and $O(n^{1/2^x})$ colours, hence $O(1)$ epochs with $O(\sqrt{n})$ colours or $O(\log \log n)$ epochs with $O(1)$ colours, for rigid robots (Pattanayak and Sharma 2024), and $O(1)$ epochs with $O(1)$ colours is achievable, which is optimal in both (Feletti et al. 2026a). Mutual visibility also takes $O(1)$ epochs with a constant number of colours in ASYNC (Sharma et al. 2021a). With lights and a sequential scheduler, robots can form any sequence of patterns up to a length fixed by the number of colours (Feletti et al. 2026c).

Whether two robots with internal lights (FSTA) can meet in ASYNC at all is open, and in rigid ASYNC the best count for external lights is 6 colours against a lower bound of 3 (Wada et al. 2026; Otaka et al. 2026). Whether 2 colours suffice for gathering $n \geq 3$ robots in ASYNC is open (Nakai et al. 2021). For uniform circle formation, the exact number of colours for $O(1)$ time, and whether oblivious robots can do it in $O(1)$ epochs, are open (Feletti et al. 2026a). Das et al. left open whether ASYNC with lights is stronger than, or incomparable with, FSYNC without lights (Das et al. 2016).

Further reading. **What weak robots can compute.** A few oblivious robots with limited visibility can together hold real numbers and simulate a rigid robot with memory (Di Luna et al. 2021); oblivious robots that move on a graph in synchronous rounds compute functions of their initial configuration, and several classes of such systems compute every finite function (Flocchini et al. 2016c). Asynchronous robots with lights and limited visibility can simulate SSYNC algorithms that obey visibility constraints (Flocchini et al. 2020), a preprint. A robot can store information in its own position, even with non-rigid moves if it has $O(1)$ bits and can move along circles, which suffices to map a polygon (Bose et al. 2019); a robot that only sees the distance to the nearest end of a segment can settle at its midpoint under small relaxations (Monde et al. 2017). Robots that cannot steer at all but bounce off each other on a ring or a segment can still discover every robot's initial position, except in characterised configurations (Czyzowicz et al. 2015b). Robots that form pairs and recognise their partner, a model proposed in a preprint, solve perpetual marching and the gathering of seven pairs under ASYNC (Kim et al. 2020), and proof assistants help design correct protocols for new tasks (Balabonski et al. 2021). On graphs, D'Emidio, Di Stefano, Frigioni and Navarra rank the schedulers for oblivious robots as FSYNC above SSYNC above ASYNC, and show that FSYNC robots and ASYNC robots with a light of $O(1)$ colours are incomparable there (D'Emidio et al. 2016, 2018), as SSYNC luminous and FSYNC oblivious robots are in the plane (Flocchini et al. 2019b). They also note that the simulation of FSYNC by ASYNC robots with three colours and one stored snapshot (Theorem 4.4 of Das et al. 2016) synchronises the Looks but not the Moves, so it does not cover tasks that forbid collisions (D'Emidio et al. 2018); a later claim that ASYNC robots with limited visibility, a common compass, four colours and one stored snapshot execute every FSYNC algorithm rests on an informal proof and does not address this point (Das et al. 2021c). A certified proof in Coq shows that oblivious ASYNC robots on a graph that move continuously along edges, but whose sensors report positions only at nodes, have exactly the executions of robots that jump from node to node (Balabonski et al. 2019). Other tasks test the same models: ASYNC robots with limited visibility and a common compass, spread densely enough over an obstacle-free rectangle, paint all of it (Das and Mukhopadhyaya 2018); and a swarm can detect that some robot has crashed with lights under FSYNC or under a scheduler that activates every robot once per epoch, one at a time, and with external lights under FSYNC, while telling which robots crashed is impossible in all twenty models studied unless the adversary that decides crashes is restricted (Clemente and Feletti 2026). These papers were read for the backfill of October 2026 and are not yet placed in settings.

Claude (Anthropic) was used in cataloguing the problems.



Claude (Anthropic) was used in cataloguing the problems.

Chapter 12

From robots to particles

Programmable matter re-asks the questions of robots in the plane on a lattice, with particles that touch, remember a little and stay connected; SILBOT takes memory and messages away again, and the stochastic approach replaces algorithms by a biased local rule.

Problems: Shape formation in programmable matter, Leader election in programmable matter, Coating, Scattering of particles, Compression, Separation of coloured particles, Alignment, Bridging, Pattern formation, Leader election in the plane.

Programmable matter, a term that Di Luna et al. trace to Toffoli and Margolus, is matter that can change its physical properties "in a programmable fashion, based upon user input or autonomous sensing", usually imagined as many small, locally interacting computational particles; the intended uses include smart materials, autonomous monitoring and repair, and minimally invasive surgery (Di Luna et al. 2020e). Navarra and Piselli add that ideas that were "mostly theoretical" in the early 1990s have become realistic as nano-technology advanced (Navarra and Piselli 2023b).

The amoebot model, introduced at SPAA 2014 (Derakhshandeh et al. 2014), is "motivated by micro- and nano-scale robotic systems with strictly limited computational and locomotive capabilities" (Daymude et al. 2023). Particles occupy nodes of the triangular lattice, have constant memory, and move by expanding into an empty neighbouring node and contracting again, like an amoeba (Emek et al. 2019). The system must stay connected, "to prevent the particles from drifting apart in an uncontrolled manner like in fluids and because in our case particles communicate only via bonds" (Derakhshandeh et al. 2015b). As results accumulated, the model's assumptions diverged; Daymude, Richa and Scheideler proposed the canonical amoebot model to separate core features from assumption variants and to treat concurrency explicitly, since much earlier work "implicitly assumes amoebot actions are isolated and reliable" (Daymude et al. 2023).

Di Luna, Flocchini, Santoro, Viglietta and Yamauchi say it directly: shape formation is the problem "called pattern formation in swarm robotics", and it has been "intensively studied" for autonomous mobile robots (Di Luna et al. 2020e). Derakhshandeh et al. posed shape formation together with leader election, because "most shape formation algorithms depend on some seed element", and asked whether shape formation can be solved when leader election cannot (Derakhshandeh et al. 2015b). With chirality, a sequential scheduler and randomization, any shape made of a constant number of triangles can be formed from a well-initialised start (Derakhshandeh et al. 2016).

The deterministic answer echoes Suzuki and Yamashita. Di Luna et al. characterise the shapes that can be formed from any simply connected start: as in the plane, feasibility depends on the symmetry of the start; chirality turns out to be "computationally irrelevant"; randomization lets any shape be formed once there are enough particles; and $O(n^2)$ moves suffice for n particles, which is optimal (Di Luna et al. 2020e). The comparison with the plane is the one in (Flocchini et al. 2017): there the symmetricity of the start decides which patterns are formable.

Leader election is the primitive behind most amoebot algorithms: a unique leader "breaks symmetry and coordinates the remaining amoebots", and Emek et al. set the goal of an algorithm that "requires

as few assumptions as possible" (Emek et al. 2019). Dufoulon, Kutten and Moses recount the history: early deterministic algorithms assumed that the system has no holes, or elected up to six leaders; using movement removed the restriction on the shape but cost at least quadratic time (Dufoulon et al. 2021). Emek et al. elect a unique leader deterministically without chirality, and the particles also agree on a chirality (Emek et al. 2019); with chirality and a strong scheduler, Dufoulon et al. do it in linear time, as fast as the randomized algorithms (Dufoulon et al. 2021; Daymude et al. 2017).

The contrast with robots in the plane is instructive. Robots that only see can elect a leader only from configurations that are not symmetric (Dieudonné et al. 2013). Particles have constant memory, read their neighbours and stay connected, and these powers break symmetries that positions alone cannot. Particles that do not move form a separate line of work: with chirality on hole-free systems (Gastineau et al. 2018), with holes but up to six leaders (Bazzi and Briones 2019), under an asynchronous adversary (Briones et al. 2023), and self-stabilising (Chalopin et al. 2025).

SILBOT takes the extra powers away. Navarra and Piselli describe it as "one of the weakest models" for programmable matter: it "includes concurrency and eliminates direct communication among particles as well as local and shared memory", and in exchange particles see two hops instead of one (Navarra and Piselli 2023b). Its aim, introduced by D'Angelo et al. (D'Angelo et al. 2020), is the minimal setting in which basic tasks remain solvable, which is the question Suzuki and Yamashita asked of robots.

The limits of oblivious robots return. In SILBOT up to three leaders are elected on hole-free configurations (D'Angelo et al. 2020). A line cannot be formed with chirality alone; with a full common orientation it takes $O(n^2)$ moves for n particles, which is optimal (Navarra and Piselli 2023b). With chirality, particles coat a convex object with one layer in $O(n^2)$ rounds (Navarra and Piselli 2023a), and particles can scatter so that no two are neighbours (Navarra et al. 2025), a lattice counterpart of scattering in the plane (Dieudonné and Petit 2009).

A second approach drops coordination altogether. Cannon, Daymude, Randall and Richa define an energy function whose minima are the desired configurations, design a Markov chain whose stationary distribution favours them, and make each transition probability computable from a particle's neighbourhood, so that the chain becomes a local, asynchronous algorithm; being a Markov chain, it is "inherently robust and oblivious" (Cannon et al. 2016). The motivation is natural systems such as fire ants forming rafts, where each individual "can take cues from its immediate neighbors" without seeing the group (Cannon et al. 2016). For compression, a bias $\lambda > 2 + \sqrt{2}$ towards having more neighbours makes the system compress, while $\lambda < 2.17$ makes it expand (Cannon et al. 2016).

The same recipe was extended one ingredient at a time. With two colours and a second bias, one algorithm separates or integrates the colour classes, a behaviour seen in bacteria, social insects and human groups (Cannon et al. 2019). With orientations, particles align or do not for every number $q \geq 2$ of orientations, a variant of the Potts model (Kedia et al. 2022). With terrain, particles build shortcut bridges across a gap, inspired by army ants that balance "the benefit of increased foraging trail efficiency" against the cost of the workers in the bridge (Arroyo et al. 2018). For fire-ant bridges, Oh et al. show that a single bridge forms for large biases, and argue that this "follows from a statistical inevitability, and not from the sophistication of the individuals" (Oh et al. 2024).

The stochastic results describe the stationary distribution, not when it is reached: no catalogued paper bounds the mixing time, and for compression the behaviour for $2.17 \leq \lambda \leq 2 + \sqrt{2}$ is not determined (Cannon et al. 2016). For SILBOT, Navarra and Piselli ask whether one common direction suffices for line formation, whether 1-hop visibility suffices, and which other shapes can be formed (Navarra and Piselli 2023b); coating without chirality is open (Navarra and Piselli 2023a). For leader election, linear-time deterministic election without chirality, and under schedulers weaker than the strong one, are open

Claude (Anthropic) was used in cataloguing the problems.

(Dufoulon et al. 2021).



Claude (Anthropic) was used in cataloguing the problems.

Chapter 13

From the cow path to evacuation

The line of work from Bellman's and Beck's linear search problem to group search with faulty robots, evacuation of disks and polygons, priority evacuation, and patrolling by robots with distinct speeds.

Problems: Linear search (cow path), Evacuation, Patrolling.

Kupavskii and Welzl open their paper with the original question, posed by Bellman in 1963 and studied by Beck, in Beck's words: "A man in an automobile searches for another man who is located at some point of a certain road. [...] How does he search so as to minimize the expected distance traveled?" (Kupavskii and Welzl 2021; Beck 1964). In that version the searcher knows the distribution of the target. Beck and Newman showed, as reported by (Kupavskii and Welzl 2021), that without knowledge of the distribution the distance cannot be smaller than 9 times the expected distance to the target.

Computer scientists rediscovered the problem as the cow-path problem: a cow at an infinite fence must find the gate, and optimality is the worst-case ratio between the distance travelled and the distance to the gate (Kupavskii and Welzl 2021). Doubling the turning point gives ratio 9 on the line, which is optimal; on m rays, turning distances that grow by the factor $m/(m-1)$ give the optimal ratio $1 + 2m^m/(m-1)^{m-1}$ (Baeza-Yates et al. 1993); randomization lowers the line to about 4.59 (Kao et al. 1996). Searching, Czyzowicz et al. note, "is central to many areas of computer science including data structures, computational geometry, and artificial intelligence" (Czyzowicz et al. 2021a).

Czyzowicz et al. proposed parallel versions of the cow path in which k robots search the line and up to f of them are faulty (Kupavskii and Welzl 2021). A crash-faulty robot passes the target without detecting it (Czyzowicz et al. 2019c); a Byzantine robot may also stay silent or announce a target that is not there (Czyzowicz et al. 2021a). For crash faults the optimal ratio is known exactly: proportional schedules (Czyzowicz et al. 2019c) are optimal, as Kupavskii and Welzl proved (Kupavskii and Welzl 2021). For Byzantine faults it is not; already three robots with one liar are open (Czyzowicz et al. 2021a; Kupavskii and Welzl 2021).

The measure also changed. Bampas et al. compare search, where cost is "the time when the first searcher arrives", with rendezvous, where it is the time when "the last involved robot reaches the meeting point", and study two robots with different speeds that must both reach the target (Bampas et al. 2019b). They find optimal strategies for every speed ratio with communication by meeting, and show that wireless communication helps only up to a certain speed ratio (Bampas et al. 2019b).

Czyzowicz et al. posed evacuation: k robots start at the centre of a unit disk and must all leave through an exit at an unknown point of the boundary; the cost is the worst-case time until the last robot is out (Czyzowicz et al. 2014). They compare it with the lost-at-sea and cow-path problems and with plane search, and cite Chrobak et al. for the observation that evacuation of uniform agents on a line "is as hard as the cow-path problem" (Czyzowicz et al. 2014; Chrobak et al. 2015). Two communication models were introduced: wireless, in which a robot that finds the exit tells everyone at once, and face-to-face (local), in which robots exchange information only when they meet (Czyzowicz et al. 2014).

For two wireless robots the answer is exactly $1 + 2\pi/3 + \sqrt{3} \approx 4.826$ (Czyzowicz et al. 2014). Face-to-face,

the robots must decide when to look for each other. Czyzowicz et al. obtained a lower bound of $3 + \pi/6 + \sqrt{3} \approx 5.255$ by studying an exit at a vertex of a hexagon, and introduced a "nonintuitive" strategy in which searchers "are forced to detour and meet" (Czyzowicz et al. 2020d). Brandt et al. showed that such meetings need not exchange useful information (Brandt et al. 2017), and Disser and Schmitt brought the upper bound to about 5.6234 (Disser and Schmitt 2019); the gap remains open.

Each variation changes one ingredient of the original problem. Communication: a sender and a receiver evacuate the disk strictly between the wireless and face-to-face optima (Georgiou et al. 2022), and on the line exactly in ratio $3 + 2\sqrt{2}$ (Czyzowicz et al. 2021d). Faults: with one crash-faulty or Byzantine robot among three, the disk bounds are not tight (Czyzowicz et al. 2017c); on the line, evacuation with $k = 2f + 1$ robots is a sub-problem of Byzantine search, since all robots may have to reach the target (Czyzowicz et al. 2021e). Domain: equilateral triangles and squares (Chuangpishit et al. 2020b; Georgiou and Jang 2026), m rays (Brandt et al. 2020b), and disks with two exits (Pattanayak et al. 2018). Speeds and energy: two robots with different speeds (Lamprou et al. 2020), and the energy spent searching, which Czyzowicz et al. consider "for the first time" (Czyzowicz et al. 2019a).

Czyzowicz et al. observe that searchers are usually assumed to have the same capabilities and the same role, yet in practice "only one of the robots has the capability to address an urgent need at the target", such as performing an emergency procedure or closing a breach (Czyzowicz et al. 2020b). Priority evacuation makes this the measure: σ servants help a distinguished robot, the queen, and only the queen's evacuation time counts (Czyzowicz et al. 2020b). For σ servants both bounds tend to 2, but they differ in the $1/\sigma$ term (Czyzowicz et al. 2020b); for one and two servants, the problem was posed in (Czyzowicz et al. 2020a), and the lower bound for one servant was later raised through weighted group search (Georgiou and Wang 2025), leaving a gap between about 4.568 and 4.819.

Patrolling keeps the geometry and the robots with given maximum speeds but replaces a one-time target by a continuing duty: every point must be visited often enough, and the cost is the idle time, the longest time a point is left unvisited (Kawamura and Kobayashi 2015). Kawamura and Kobayashi note that patrolling had been studied mostly through heuristics and experiments, and that theoretical studies "have revealed that there are interesting questions and intricacies even in the simplest settings" (Kawamura and Kobayashi 2015). Czyzowicz et al. studied fences patrolled by agents with distinct speeds (Czyzowicz et al. 2011) and, as Kawamura and Kobayashi report, conjectured that partitioning the fence in proportion to the speeds is always optimal; Kawamura and Kobayashi confirmed this for three agents and disproved it in general (Kawamura and Kobayashi 2015), and Kawamura and Soejima found schedules that patrol a fence $4/3$ times longer than partition and a circle $21/20$ times better than the simple strategies (Kawamura and Soejima 2020). Faults enter as in search: with up to f unreliable robots, the optimal idleness of an Eulerian graph is $(f + 1)|E|/k$ (Czyzowicz et al. 2017e). How far an optimal fence schedule can beat partition remains open (Kawamura and Soejima 2020).

Claude (Anthropic) was used in cataloguing the problems.

Chapter 14

When agents fail

A node that destroys visitors, a robot that stops, an agent that lies, a searcher that misses the target: the papers motivate each fault by the hardware or the network, and each kind of fault changes a different measure, from team size to feasibility to an exact constant.

Problems: Black hole search, Gathering in graphs, Gathering, Dispersion, Mutual visibility, Linear search (cow path), Evacuation.

The papers give two kinds of reasons. The first is hardware and environment. For robots in the plane, faults "can occur due to the unreliable components in the system, which are custom made or manually built", from manufacturing defects, or from "some external factor in the field of deployment" (Pattanayak et al. 2019b); since swarms "are envisioned for applications such as rescue, exploration, and surveillance in hazardous environments, it is natural that robots may experience faults" (Pramanick et al. 2025). The second is scale: "When the number of robots in a swarm increases, the possibility of having one robot crash unpredictably becomes important to tackle" (Bramas et al. 2023b), and on graphs "The scale-up when considering numerous agents is inevitably tied to the occurrence of faults among them" (Bouchard et al. 2022).

The fault models are borrowed from classical distributed computing. A crash fault stops an entity for ever; a Byzantine fault lets it behave arbitrarily, and "the Byzantine fault subsumes the crash fault" (Molla et al. 2021a). On graphs a further distinction matters: a weakly Byzantine agent cannot fake its identifier, a strongly Byzantine one can (Molla et al. 2021b). Bouzid, Das and Tixeuil import wait-freedom: "an arbitrary and unexpected delay observed at one robot may not prevent other robots from making progress toward the solution", even if $n-1$ of the n robots are delayed or crashed (Bouzid et al. 2013).

Black hole search puts the fault in the network instead of the agents. Di Luna et al. motivate it by host attacks on mobile code: a black hole is a site "that disposes of visiting agents upon their arrival, leaving no observable trace of such a destruction", which can be a virus that trashes incoming messages or "the undetectable crash failure of a site" (Di Luna et al. 2025b). Kaur et al. add a node in a cloud that loses essential data (Kaur et al. 2026b). The task is to locate it with at least one agent surviving, so the primary measure is the team size: how many agents the adversary may destroy.

The answers depend on what agents know and how they communicate. Asynchronous agents with whiteboards and no map need $\Delta+1$ agents and $\Theta(n^2)$ moves; with a map two agents suffice and $\Theta(n \log n)$ moves (Dobrev et al. 2006). Tokens can replace whiteboards: $\Delta + 2$ agents without a map (Balamohan et al. 2016), two with a map at the same cost (Flocchini et al. 2012b). When the adversary may also remove one edge per round of a ring, three agents are necessary and sufficient, with $\Theta(n^{1.5})$ moves using pebbles (Di Luna et al. 2025b); in arbitrary dynamic graphs with one missing edge per round, six co-located agents with whiteboards suffice (Kaur et al. 2026b).

A crash is the mildest fault, and in the plane it is often harmless for feasibility. In SSYNC, the correct robots gather despite up to $n-1$ crashes, wait-free, with only weak multiplicity detection and no agreement on coordinates (Pattanayak et al. 2019b); the first crash-tolerant algorithm, for one crash, is in (Agmon and Peleg 2006). The subtlety is what "gathering" means: in the weak version only correct

robots must meet, in the strong version all robots must meet at the crash site; Bramas, Lamani and Tixeuil describe the strong one as one "where correct robots are requested to rescue failed ones", and show that two robots with lights cannot solve it in SSYNC (Bramas et al. 2023b). For mutual visibility with lights one crash fault is tolerated (Aljohani and Sharma 2018).

On graphs a crashed agent "loses all its information immediately, as if the robot has vanished from the network" (Chand et al. 2025a). For dispersion this costs time rather than feasibility: the best crash-tolerant algorithm for k agents runs in $O(\min(m, k\Delta))$ rounds on a graph with m edges and maximum degree Δ (Pattanayak et al. 2023) against the fault-free $O(k)$ (Kshemkalyani et al. 2025b), while on oriented grids crash-tolerant dispersion takes $O(\sqrt{n})$ rounds on n nodes, optimal when $k = \Omega(n)$ (Banerjee et al. 2025). In search, a crashed robot is silent: it visits the target without reporting it (Kupavskii and Welzl 2021). On the line the optimal ratio with f crash faults among k robots is known exactly (Czyzowicz et al. 2019c; Kupavskii and Welzl 2021); with one crash among three wireless robots in a disk, evacuation takes between $1 + 4\pi/3 \approx 5.188$ and about 6.309 (Czyzowicz et al. 2017c).

Byzantine faults are "considered as the worst faults that can occur", and a Byzantine agent "may also convey arbitrary information to the other agents, impersonate the identity of another agent, and so on" (Bouchard et al. 2022). In the plane one Byzantine robot among three already prevents gathering, and even convergence, in SSYNC, while in FSYNC $n \geq 3f + 1$ robots tolerate f Byzantine ones (Agmon and Peleg 2006); for mutual visibility with lights one Byzantine robot among three is likewise fatal (Aljohani and Sharma 2018). On graphs, gathering with f strongly Byzantine agents needs at least $f + 1$ good agents when the number n of nodes is known and $f + 2$ otherwise, as Dieudonné, Pelc and Peleg proved when they posed the problem (Dieudonné et al. 2014), and these team sizes suffice (Bouchard et al. 2016); the algorithms are exponential, and polynomial time is known only for larger teams (Bouchard et al. 2022).

Dispersion was taken to Byzantine agents because "in the real world, where systems must be fault tolerant and errors are the norm instead of the exception, more work must be done" (Molla et al. 2021b). On rings it costs $\Theta(n)$ rounds under the assumptions of (Molla et al. 2021a); on general graphs the best bound is $\tilde{O}(n^9)$ against $\Omega(n)$, because the algorithms go through Byzantine gathering (Molla et al. 2021b); with strong Byzantine agents an arbitrary start becomes polynomial only with a trusted shepherd (Moses Jr. and Redlich 2024). Black hole search with weakly Byzantine companions that survive the black hole needs $2f + 2$ agents with a map (Di Luna et al. 2025a).

In search a Byzantine robot can claim a target that is not there, so "a reliable robot cannot necessarily trust an announcement that the target has been found" (Czyzowicz et al. 2021a). On the line the optimal ratio is known only for some densities f/k (Czyzowicz et al. 2021a); with three robots and one Byzantine, disk evacuation takes between about 5.948 and 6.921 (Czyzowicz et al. 2017c).

Each model in the atlas orders faults from most hostile (level 0) to none: strong Byzantine, weak Byzantine, crash, none for agents on graphs; Byzantine, crash, none for robots in the plane and for search. An algorithm proved against Byzantine entities therefore carries over to crashes, and an impossibility for crashes carries over to Byzantine entities. The black hole is not a value on this axis: it is part of the statement of black hole search, whose faults axis concerns the agents (the Byzantine-companion variants sit at weak Byzantine).

The measure that a fault changes differs by model, and that is the clearest lesson of the comparison. Losing agents in a black hole changes the team size. Crashes in the plane rarely change feasibility, but Byzantine robots often do. Crashes on graphs change time by a factor that is not yet understood, and Byzantine agents change it from linear to a high polynomial. In search every fault changes an exact constant, and the gaps between the constants are the open problems.

Claude (Anthropic) was used in cataloguing the problems.



Claude (Anthropic) was used in cataloguing the problems.

Chapter 15

What forces the bounds

Every lower bound in the atlas names the resource the adversary exploits. Read side by side, a few bottlenecks (distance, symmetry, missing information, an adversary that acts) explain most tight results, and the open entries cluster where no bottleneck beyond distance is known.

Problems: Dispersion, Collaborative exploration, Graph exploration by a single agent, Rendezvous, Treasure hunt, Black hole search, Gathering, Gathering in graphs, Linear search (cow path), Evacuation, Shape formation in programmable matter, Leader election in programmable matter.

Each lower bound in the atlas carries a `bottleneck`: two or three words for what the adversary uses. Across all problems the commonest names are *distance*, *symmetry breaking* and *symmetry*, then *memory* (colours of lights), *unknown exit*, *adversarial edge changes* and *unknown distance*; a long tail of names (*trap graphs*, *port addressing*, *indistinguishable Byzantine agents*, *timing is the only channel*) covers single arguments. This essay groups them into four kinds and ends with what is left open.

The grouping matters because results move along the assumption axes in opposite directions: an algorithm for a weak setting holds in every stronger one, a lower bound for a strong setting holds in every weaker one. A bound whose bottleneck survives every strengthening (distance, for instance) is inherited by a whole subtree of settings; a bound that depends on an assumption (asynchrony, an unknown exit) disappears as soon as that assumption is lifted.

The plainest argument is that an entity must cover a distance. Kshemkalyani et al. state it for dispersion in one sentence: "Suppose two nodes of a graph are at a distance of $\Omega(k)$, then to disperse from one node to the furthest node takes $\Omega(k)$ time (e.g., line graph)" (Kshemkalyani et al. 2025b). The bound is from (Kshemkalyani and Ali 2019), and after several rounds of algorithms it is matched by $O(k)$ rounds, synchronous and asynchronous (Kshemkalyani et al. 2025b; Pattanayak et al. 2026). On grids the distance is $\sqrt{k}$ (Kshemkalyani et al. 2020c). For collaborative exploration the offline cost $\Theta(n/k + D)$ is the distance part (Disser et al. 2020).

Programmable matter has the same argument, counted in moves: n particles reshaping a compact configuration into a line need $\Omega(n^2)$ moves in total (Derakhshandeh et al. 2015b), as do deterministic shape formation (Di Luna et al. 2020e), universal coating (Derakhshandeh et al. 2017) and centralised transformations by sliding (Michail et al. 2019); universal shape formation needs $\Omega(\sqrt{n})$ rounds (Derakhshandeh et al. 2016). Distance bounds are usually the easy half of a tight result. What took years in dispersion was the algorithm, not the bound.

The second kind of argument is about possibility rather than cost. Identical entities that the adversary keeps in symmetric positions cannot be separated by a deterministic rule: two oblivious robots in SSYNC (Suzuki and Yamashita 1999; Flocchini et al. 2016a), anonymous agents whose views coincide (Dieudonné and Pelc 2016), leader election among disoriented robots in symmetric positions (Dieudonné et al. 2013), arbitrary patterns in ASYNC without a common orientation (Flocchini et al. 2008a), and silent particles that share only chirality (Navarra and Piselli 2023b). Identical points that the robots cannot count produce the same effect: without multiplicity detection gathering is impossible (Prencipe

2007).

Even when identifiers break symmetry, carrying them costs memory. For dispersion of k agents the $\Omega(\log k)$ bits bound holds "since at least one agent needs $\Omega(\log k)$ bits to store its unique ID" (Kshemkalyani et al. 2025b; Augustine and Moses Jr. 2018). Anonymous rendezvous in graphs of n nodes needs $\Theta(\log n)$ bits (Czyzowicz et al. 2012b), and labelled rendezvous on a ring with simultaneous start takes $\Theta(d \log \lambda)$ rounds, where d is the initial distance between the agents and λ the smaller label: the price of reading a label of $\log \lambda$ bits bit by bit (Dessmark et al. 2006). When agents cannot talk at all, time is the only channel: silent dispersion needs $\Omega(k + \log L)$ rounds (Das et al. 2026a) and $\Omega(\log L + \log \Delta)$ bits (Gorain et al. 2024), with identifiers at most L and maximum degree Δ. In the plane the same economy appears as colours: two robots with lights need two (Viglietta 2013), with only internal or only external lights three (Otaka et al. 2026).

Search problems are bounded by what the searcher does not know. Beck already showed that, without knowing the distribution of the target, the distance travelled cannot be below 9 times the expected distance (as reported by Kupavskii and Welzl 2021), and the cow-path ratio 9 is the competitive form of the same fact (Baeza-Yates et al. 1993); randomization lowers it to about 4.5911 (Kao et al. 1996). Kupavskii and Welzl's bound for m rays, specialised to robots without faults, settled the question of parallel search on m rays "posed by three groups of scientists some 15 to 30 years ago" (Kupavskii and Welzl 2021). An unknown exit gives every evacuation bound on the disk (Czyzowicz et al. 2014), and one-way communication gives the line its exact ratio $3 + 2\sqrt{2}$ (Czyzowicz et al. 2021d).

On graphs the missing information is which port to take. Remembering one port among Δ costs $\lceil \log \Delta \rceil$ bits for perpetual tree exploration (Diks et al. 2004) and $\Omega(\log \Delta)$ bits for dispersion by anonymous randomized agents (Molla and Moses Jr. 2019); treasure hunt with pebbles needs $\Omega(D \log \Delta)$ rounds (Gorain et al. 2022). Stronger memory bounds come from trap graphs, built so that an agent with too little memory (or too few pebbles) misses part of them: $\Omega(D \log \Delta)$ bits for perpetual exploration (Fraigniaud et al. 2005a) and $\Theta(\log \log n)$ pebbles for a constant-memory agent (Disser et al. 2019). Fraigniaud et al. recall why these bounds were pursued: exploration by finite automata was tied to the complexity of undirected connectivity (Fraigniaud et al. 2006c).

Some bounds need an adversary that does more than choose the input. An adversary that removes edges forces three agents and $\Omega(n^{1.5})$ moves for black hole search in a dynamic ring (Di Luna et al. 2025b), makes dispersion with local communication impossible in 1-interval-connected graphs (Kshemkalyani et al. 2020a), and forces $\Omega(n^2)$ steps to explore some temporal graphs even offline, whereas static undirected graphs need fewer than $2n$ (Erlebach et al. 2021). An adversary that delays agents forces $\Theta(n^2)$ moves in black hole search without a map (Dobrev et al. 2006) and an $\Omega(\log^2 k)$ factor in asynchronous collaborative tree exploration (Cosson and Massoulié 2025); one that reveals a tree online forces a competitive ratio of $\Omega(\log k / \log \log k)$ for collaborative exploration (Disser et al. 2020).

Byzantine entities act from inside. Because they are indistinguishable from good ones, gathering on graphs with f Byzantine agents needs $f + 1$ good agents (Bouchard et al. 2016), three robots with one Byzantine cannot gather in SSYNC (Agmon and Peleg 2006), dispersion of $k > n$ agents on n nodes into a balanced configuration is impossible whenever $\lceil k/n \rceil > \lceil (k - f)/n \rceil$, because Byzantine agents can occupy nodes exactly like honest ones (Molla et al. 2021b), and the best lower bound for search on the line with Byzantine robots, about 5.23, comes from the argument for crash faults (Kupavskii and Welzl 2021). The black hole is the extreme case of an adversary inside the network: an asynchronous team without a map needs $\Delta + 1$ agents, Δ the maximum degree, and it cannot succeed at all if n is unknown or the graph is not 2-connected (Dobrev et al. 2006).

At the time of writing, the atlas judges optimality per measure for every variant, and more than two

Claude (Anthropic) was used in cataloguing the problems.

in five of these judgements are *unknown*: no non-trivial lower bound, or no analysis of that measure. Most of them come from two habits of the literature. In the plane, time is often not analysed because the question was feasibility. On graphs, rendezvous and exploration algorithms often leave memory unbounded. Both are open questions that nobody has asked yet rather than hard ones.

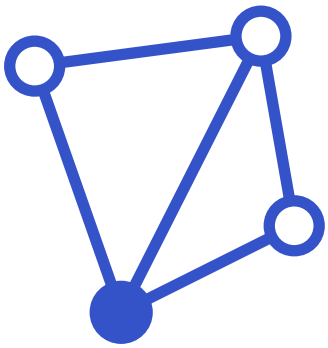
The polynomial gaps cluster where the only lower bound is distance. Dispersion on trees with n nodes has $O(n)$ for $k = n$ agents against an $\Omega(D^2)$ bound proved only for trees of height about $\sqrt{\log n}$ (Disser et al. 2020); crash-tolerant dispersion is a factor Δ (the maximum degree) from $\Omega(k)$ (Pattanayak et al. 2023); Byzantine dispersion is $\tilde{O}(n^9)$ against $\Omega(n)$ (Molla et al. 2021b). Black hole search with tokens is $O(mn)$, for m edges, against $\Omega(n^2/\Delta)$ (Balamohan et al. 2016). Stopping tree exploration without knowing n uses $O(\log n)$ bits against an $\Omega(\log\log\log n)$ bound (Ambühl et al. 2011; Diks et al. 2004), and leader election in programmable matter has no proven lower bound, only an informal $\Omega(D)$ remark, D being the diameter of the particle configuration (Dufoulon et al. 2021). Where the bottleneck is symmetry or missing information, results tend to be tight or exact; the gaps that remain in search are between two constants, as in two-robot face-to-face evacuation, about 5.255 against 5.6234 (Czyzowicz et al. 2020d; Disser and Schmitt 2019). The progress map of each problem shows the bottleneck next to every bound, so these patterns can be checked directly.



Claude (Anthropic) was used in cataloguing the problems.

Part III

Mobile agents on graphs



The model and its assumption axes are in Chapter 2.

Claude (Anthropic) was used in cataloguing the problems.

CHAPTER 16

Exploration

Visit every node of an unknown graph: one agent with as little memory or as few markers as possible, or a team as fast as possible. Treasure hunt and rendezvous are subproblems of exploration; dispersion is derived from collaborative exploration.

Problems in this family:

- Graph exploration by a single agent
- Treasure hunt — subproblem of Graph exploration by a single agent
- Rendezvous — subproblem of Graph exploration by a single agent
- Gathering in graphs — generalisation of Rendezvous
- Gathering by oblivious robots on graphs — variant of Gathering in graphs
- Map construction — strengthening of Graph exploration by a single agent
- Exploration by oblivious robots — variant of Graph exploration by a single agent
- Exploration by luminous robots — variant of Exploration by oblivious robots
- Exploration of recolourable graphs — variant of Graph exploration by a single agent
- Online exploration of weighted graphs — variant of Graph exploration by a single agent
- Collision-free exploration — variant of Graph exploration by a single agent
- Collaborative exploration
- Dispersion — derived from Collaborative exploration
- Black hole search — variant of Collaborative exploration
- Exploration of dynamic graphs by teams — variant of Collaborative exploration
- Uniform deployment on graphs
- Network decontamination
- Broadcast by agents in dynamic graphs
- Delivery by energy-constrained agents



16.1 Graph exploration by a single agent

One agent starts at a node of an unknown anonymous graph and must visit every node; how much memory does it need, which pebbles or knowledge make it possible, and how fast can it be?

First posed in 1951 (Shannon 1951). Shannon's maze-solving mouse Theseus (first prototype 1950) explored a 5×5 grid with constant memory per cell (Disser et al. 2019); the first impossibility theorem for finite automata is Budach's (Budach 1978).

Problem statement and measures

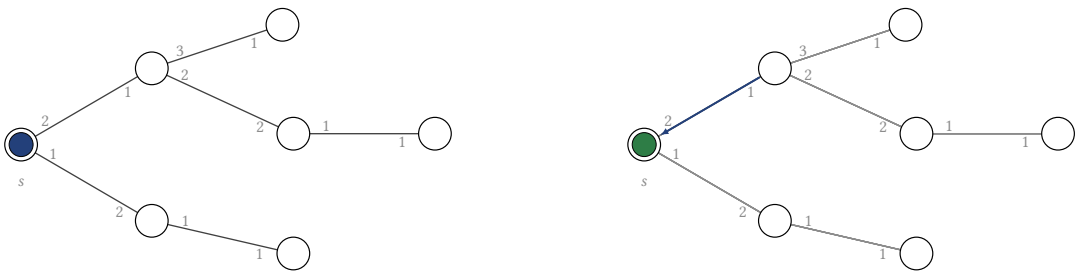
A single deterministic agent is placed at a node s of a connected graph G with n nodes, m edges and maximum degree Δ. Nodes are anonymous (they carry no names the agent can read), and the edges at each node are numbered locally; these numbers are the node's **ports**. The agent sees the degree of its node and the port by which it entered. It must **visit every node of** G. Three termination requirements are standard:

- **perpetual exploration**: every node is eventually visited; the agent never has to stop;
- **exploration with stop**: every node is visited and the agent then stops, anywhere;
- **exploration with return**: every node is visited and the agent stops at s.

Return implies stop, and stop implies perpetual exploration. **Memory** is the number of bits of the agent's state; when the agent carries **pebbles** (objects it can drop on a node and pick up again) their number is reported with the memory. **Time** is the number of moves (edge traversals) until exploration is complete; for periodic exploration it is the period. Unless a variant says otherwise the agent knows nothing about G, not even n.

Exploration is the oldest problem of the mobile-agent literature: it started as maze search by finite automata, and the question of how little memory suffices runs from Budach's impossibility (1978) to Reingold's log-space algorithm (2005). Several agents exploring together form the sibling problem, collaborative exploration.

Measures: **Time**, edge traversals until every node is visited (for periodic exploration: the period); **Memory**, bits of agent memory; pebbles, if any, are counted separately.



(a) Start: one agent at s; ports 1..d at each node

(b) Final: back at s after 2(n-1) = 12 moves; all 7 nodes visited

Figure 16.1: Exploration of a tree by the rule Choose-Next of (Diks et al. 2004): the agent leaves the start node s by port 1 and, entering a node of degree d by port i, leaves by port $(i \bmod d) + 1$. It walks around the tree, each edge once in each direction, and is back at s after $2(n-1)$ moves with every node visited (perpetual exploration repeats this tour). One frame per move.

A first look

Imagine a small cleaning machine put down in a large building with no map, where every room looks the same. We model the building as a **graph**: rooms are **nodes** and corridors are **edges** (each edge joins

Claude (Anthropic) was used in cataloguing the problems.

two nodes), and the machine is an **agent** that walks from node to node along edges; one crossing of one edge is a **move**. n is the number of nodes and m the number of edges; the **degree** of a node is the number of edges at it, and Δ is the largest degree. At each node the edges are numbered $0, 1, \ldots$; these numbers are called **ports**, and the numbering at one node has nothing to do with the numbering at the next. The agent sees only the degree of its current node and the port by which it came in; nodes are **anonymous** (they have no names), so it cannot tell a node it has seen before from a new one. In the basic version (**perpetual exploration**) it must eventually visit every node of every connected graph, without ever having to stop, and the question is how few bits of **memory** (what the agent keeps from one move to the next) make this possible.

1. Algorithm: Leave by the next port (trees). (Diks et al. 2004) (Trees, perpetual exploration, p. 93) This is the algorithm Choose-Next of (Diks et al. 2004), for graphs without cycles (trees); here the ports of a node of degree d are numbered $1, \ldots, d$.

1. Leave the **start node** (the node where you are placed) by port 1.
2. When you enter a node of degree d by port i, leave it by port $(i \bmod d) + 1$, the next port in cyclic order (at a node of degree 1 this sends you straight back).
3. Repeat forever.

Why it works and what it costs. On a tree this is the rule "keep your right hand on the wall": the agent walks around the tree, and every edge is walked once in each direction before the walk repeats, so all n nodes are visited within $2(n-1)$ moves. The agent only has to remember one port number, which (Diks et al. 2004) count as $O(\log \Delta)$ bits, and they show that $\Omega(\log \Delta)$ bits are needed on trees, so this is optimal. On graphs with cycles the same rule can go round a cycle forever and miss nodes; there the right amount of memory is $\Theta(D \log \Delta)$ bits, with D the diameter (the largest distance between two nodes) (Fraigniaud et al. 2005a).

Why it matters. Choosing the next port from the entry port alone is the basic move of almost every memory-efficient exploration algorithm, and the universal exploration sequences used for general graphs are a generalisation of it.

2. Impossibility: No stopping on a ring without a pebble. (Dieudonné and Pelc 2012) (Exploration with stop, nothing known, p. 94) Ask for more: the agent must visit every node **and then stop**, without knowing n and without a **pebble** (an object it could drop on a node to recognise that node later). Take rings (cycles) of every size, numbered so that at every node port 0 leads clockwise and port 1 counterclockwise. In such a ring every node has degree 2 and every arrival looks the same, so what the agent sees never depends on the size of the ring. A deterministic agent (one with no random choices) therefore makes the same sequence of moves in every such ring. Since it explores and stops in, say, the ring of 3 nodes, it stops after some fixed number T of moves, and it does the same in the ring of $T + 2$ nodes.

Why it holds. In T moves the agent can visit at most $T + 1$ different nodes, so in the ring of $T + 2$ nodes at least one node is never visited: the agent stopped too early. Extra memory does not help, because the problem is not remembering but recognising: the agent can never tell "I am back where I started" from "this is a new node that looks the same". This is why every stopping variant in the catalogue adds a pebble or knowledge of a bound $N \geq n$ (Dieudonné and Pelc 2012).

Why it matters. It shows that the hard part of exploration is often knowing when you are done, a theme that returns in dispersion, rendezvous and gathering.

3. Algorithm: Random walk. (Aleliunas et al. 1979) (Randomized exploration when n is known, p. 97) Allow the agent to flip coins, and tell it a number $N \geq n$.

Claude (Anthropic) was used in cataloguing the problems.

1. At each node, pick one of the ports uniformly at random and take its edge.
2. Count the moves; stop after $2N^3$ moves.

Why it works and what it costs. (Aleliunas et al. 1979) proved that a random walk visits every node of a connected graph within $2m(n-1)$ moves on average. Since $m \le N^2/2$ and $n - 1 < N$, this average is below N^3; a walk that runs twice as long as its average cover time fails to cover the graph with probability at most $1/2$ (Markov's inequality), and repeating the walk lowers the failure probability further. The agent needs only a counter, $O(\log N)$ bits. A deterministic agent with the same memory also exists, but its algorithm (Reingold's log-space construction) is much harder (Reingold 2008).

Why it matters. The random-walk bound was the starting point for universal traversal and exploration sequences, which turn the random walk into a fixed list of port choices that works on every graph.

What the community started looking at

Exploration began as **maze search by finite automata**. The first question was whether a machine with bounded memory can visit every node at all: Budach showed it cannot for mazes (Budach 1978), Blum and Kozen showed that two pebbles or two automata are enough in mazes (Blum and Kozen 1978), and Rollik showed that every finite team has a planar trap (Rollik 1980).

From there the community turned to **how much memory** an agent needs as a function of the graph: $\Theta(D \log \Delta)$ bits in general (Fraigniaud et al. 2005a), $O(\log n)$ once n is known (Reingold 2008), and the effect of the termination requirement on trees (Diks et al. 2004; Ambühl et al. 2011). Markers came next: what one pebble buys (Fraigniaud et al. 2006c), and the exact exchange rate between pebbles and memory (Disser et al. 2019). Time was a secondary measure throughout.

Further reading

Papers on this problem that are not yet placed in a setting.

- Bhagat and Pelc (2026). Bhagat and Pelc: a deterministic finite automaton in the infinite anonymous oriented grid $\mathbb{Z} \times \mathbb{Z}$ (ports N, E, S, W at every node) must visit every node of a given wedge, the grid nodes between two rays from a common origin, whose directions are built into the automaton. Each ray is a wall (the edges across it are blocked) or free, and the agent starts at a node of the wedge chosen by an adversary (at the origin for a wedge with two free rays). The minimum number of pebbles is 0, 1 and 2 for wedges of angle below π with two walls, one wall and no wall, and 1, 2 and 3 for angles of at least π (Theorems 2.1–2.5 and 3.1–3.5, Corollaries 2.1, 2.2, 3.1 and 3.2); a narrow walled wedge can be disconnected, and then the component of the start node is explored. The impossibilities for wedges with two free rays follow from bounds on teams of two and of three automata in the grid by Emek, Langner, Stolz, Uitto and Wattenhofer and by Brandt, Uitto and Wattenhofer, as cited there. This is perpetual exploration of one given infinite graph, which no setting of this problem covers. Read (arXiv v1, 2023).
- Adamson (2025a). Exploration with the schedule known when every edge of the underlying graph is present at least once in every F consecutive steps, no snapshot being required to be connected: one agent explores from any start node within $F(2n-3)$ steps by following a spanning tree of minimum total waiting time (Theorem 3.5, Lemma 3.6), and a star in which all edges but one are present only every F-th step needs $F(2n-5)+1$ steps (Theorem 3.10, Corollary 3.11); edges present periodically with period at most F are a special case (Corollary 3.8). Consequences for public-transport timetables (Theorem 4.1), for graphs in which each node opens one incoming edge per step in a fixed cyclic order ($4m$ steps, Theorem 4.2) and for broadcast networks (Corollary 4.4, Theorem 4.6). Preprint; read.
- Adamson (2025b). Exploration with the schedule known, for temporal graphs given by a word over the nodes: the word is cut into factors without a repeated symbol, one per step, and an edge of the underlying graph (a pair of symbols that alternate in the word) is present in a step when one of its end nodes occurs in that factor. If every snapshot is connected, one agent explores within $2n \cdot \min_v \deg(v)$ steps, the degree taken in the

Claude (Anthropic) was used in cataloguing the problems.

underlying graph (Theorem 4). If only the underlying graph is connected, with diameter D (their d), and the word has length at least $n(2Dn + D)$, it explores within $2Dn$ steps (Theorem 11), and for all n and D some such temporal graph needs $\Omega(Dn)$ steps (Theorem 22). Preprint; read.

- Baguley et al. (2026). Random temporal graphs in which every snapshot is a spanning tree drawn independently from a fixed distribution over a set of trees on the n nodes. With high probability an agent that knows the whole sequence explores within $O(n^{3/2})$ steps (Theorem 1.1), which is tight for the uniform distribution over $\lceil n/3 \rceil$ stars (Theorem 5.4), and within $54m$ steps when all trees are spanning trees of one graph with m edges (Theorem 1.2). Trees of maximum degree Δ can force $\Omega(\sqrt{\Delta}\, n)$ steps (Theorem 5.12, odd $\Delta \geq 5$), and $O(\sqrt{\Delta}\, n)$ is conjectured (Conjecture 1.3). (Bastide et al. 2025) prove $O(\Delta n \log^3 n)$ in expectation and with probability $1 - e^{-\log^3 n}$ (Theorem 1.6 and Corollary A.3), for any distribution over connected graphs with Δ replaced by the average temporal maximum degree in expectation. An agent that sees only the current snapshot needs $\Omega(n \log n)$ steps on some such model to succeed with probability $1 - n^{-\Omega(1)}$ (Proposition 4.4), and more than $n^2/15$ steps with high probability on $n/2$ uniform stars (Proposition 5.13). Read.
- Balev et al. (2025). Shortest temporal exploration: in an always-connected temporal graph whose whole schedule is known, the agent minimises the number of edges it traverses rather than its arrival time. With lifetime at least $12n^{3.5} + 4n$ some exploration traverses $O(n^{1.5})$ edges (Theorem 12), against the $O(n^2)$ implied by the arrival-time bound; if every snapshot has diameter at most some parameter, fewer edges than that parameter times n suffice within that parameter times n^2 steps (Theorem 13); on a temporal ring with lifetime at least $2n-2$, $\lfloor 3(n-1)/2 \rfloor$ edges suffice and may be needed, whereas with lifetime exactly $2n-3$ an exploration may need $2n-3$ edges (Lemmas 16 to 18). The authors conjecture that every always-connected temporal graph has an exploration of $\Theta(n^2)$ steps that traverses $\Theta(n)$ edges. Preprint; read.
- Chalopin et al. (2015a). An agent with binoculars (it sees the graph within a constant distance) can explore an anonymous graph and stop in many graphs that are not trees: a characterisation of the graphs explorable this way by simple connectivity of a clique complex, with a universal algorithm that explores each of them and never stops elsewhere (Theorem 1). Read (arXiv v1).
- Doi et al. (2022). A different model, a metamorphic robotic system: anonymous oblivious modules on the square grid that slide or rotate while staying connected search an unknown finite grid for a target. Three modules with a common compass are necessary and sufficient from any initial shape (Theorem 1); without a compass, five modules are necessary and sufficient from restricted initial shapes (Theorem 4). Read (arXiv v2).
- Gorain and Pelc (2018). How much advice an agent needs to explore an anonymous graph in a given time, from an oracle that knows the whole instance or only the map. Abstract read.
- Pelc (2018b). Which families of anonymous graphs can be explored deterministically without knowledge: a characterisation, and whether one universal algorithm explores every explorable family. Abstract read.
- Menc et al. (2015). An agent with M_a bits of memory and M_w bits per node that does not learn its entry port: one bit of agent memory changes what can be explored compared with the rotor-router, a $(0, \log d)$-agent. A brief announcement; introduction read.
- Disser and Klimm (2022). A survey of space complexity: $\Theta(\log n)$ bits are necessary and sufficient to explore all n-node graphs, and with sublogarithmic memory $\Theta(\log \log n)$ agents are needed. Abstract read.
- Devismes et al. (2026a). Distance-constrained exploration: the agent must always know a path of length at most $(1 + \alpha)r$ back to its start, with r the eccentricity of the start; the penalty is the number of traversals beyond the edges. Abstract read.
- Caissy and Pelc (2016). Exploration of the fault-free component when some edges are faulty: a perfectly competitive algorithm for rings, and depth-first search is within $10/9$ of the best overhead on Hamiltonian graphs. Abstract read.
- Dobrev et al. (2024). Finding a treasure in the n-dimensional infinite grid by finite-state agents that start together: three randomised synchronous agents suffice for every n, disproving a conjecture that $n + 1$ are needed. Abstract read.
- Stopkin (2025b). A mobile agent that reads and changes marks on nodes and edges explores a finite undirected graph in $O(n^3)$ time with $O(n^2)$ space. Abstract read.
- Stopkin (2026). A further marking algorithm for exploring finite undirected graphs by one agent. Abstract read.

Claude (Anthropic) was used in cataloguing the problems.

- Li et al. (2021b). Computing optimal offline exploration paths in grid environments for a robot with limited visibility (AAAI 2012). Abstract read.
- Ilcinkas and Wade (2021). A single agent that knows the dynamics explores constantly connected dynamic graphs whose underlying graph is a cactus in $O(n \log n / \log \log n)$ time units, with a near-matching lower bound. Abstract read.
- Ilcinkas and Wade (2020). Exploration of carrier-based time-varying networks (carriers on periodic routes) when the agent may wait at stations. Abstract read.
- Wade (2021). Exploration with return of constantly connected dynamic rings: $3n - 4$ time units are necessary and sufficient. Abstract read.
- Arrighi et al. (2023). Kernelization of non-strict temporal exploration of a given temporal graph. A preprint; abstract read.
- Bodlaender and van der Zanden (2018). Temporal graph exploration is NP-complete even when the underlying graph has pathwidth 2 and every snapshot is connected. A preprint; abstract read.
- Bellitto et al. (2023). Restless temporal exploration (the agent must move at every step) of periodic temporal graphs: polynomial for period 2, hard otherwise. Abstract read.
- Brandt et al. (2020c). Three semi-synchronous agents running one finite automaton (distinct initial states, sensing only agents in the same cell) cannot explore the infinite grid, answering a question of Emek et al. Abstract read.
- Baaziz and Pelc (2026b). A deterministic finite automaton in a convex open region of the plane: it sees the disc of radius 1 around it, moves less than 1 per step, and must eventually see every point, with movable pebbles to mark places. Convex regions are bounded ones, cups, belts, half-planes and the plane (Theorem 1). Bounded regions need no pebble (Theorem 2), cups and belts exactly two (Corollaries 1, 2), half-planes and the plane exactly three (Theorems 5, 6, Corollary 3; the lower bound uses the grid result of Brandt et al. 2020c); the agent knows the type of region but not the region or its start. Read in full.
- Reingold et al. (2006). Reingold, Trevisan and Vadhan carry the log-space method of (Reingold 2008) over to directed graphs in which every node has the same in-degree and out-degree D (biregular digraphs): a deterministic log-space algorithm finds a path between two given nodes. The algorithm yields a log-space constructible universal traversal sequence, a sequence of port numbers that leads a walk from any start node of any graph in the family through every node, for biregular digraphs whose labelling is consistent (at each node the incoming arcs carry distinct labels); every biregular digraph has such a labelling. They also give a log-space pseudorandom walk for these graphs, and show that extending it to every labelling of biregular digraphs would give RL = L. Read: abstract and introduction of the ECCC report TR05-022 (2005).
- Rivest and Schapire (1993). Rivest and Schapire learn an unknown finite automaton, a directed graph whose arcs are labelled by actions and whose nodes show outputs, from its input/output behaviour alone, without a reset that returns it to a start state; their motivating example is a robot in an unfamiliar environment. The key tool is a homing sequence, a sequence of actions whose observed outputs determine the state reached. With a teacher who answers wrong conjectures by counterexamples, the algorithm outputs a correct description with any prescribed probability of success, in time polynomial in the size of the automaton, the length of the longest counterexample and the logarithm of the inverse failure probability. For permutation automata they give probabilistic algorithms that need no teacher. (Bender and Slonim 1994) introduce a homing sequence for two agents and need no teacher. Read: abstract and first page (scanned journal version).

Relations to other problems

- **Derived problem: Treasure hunt** (subproblem of). An agent that explores the graph finds any treasure, so every exploration algorithm solves treasure hunt; the question is how much cheaper it can be when the treasure is close, measured against its distance d (or a weight of the path to it) rather than the size of the graph.

Claude (Anthropic) was used in cataloguing the problems.

- **Derived problem: Rendezvous** (subproblem of). If the agents can break symmetry (for example by their labels), one may wait while the other explores, so exploration solves rendezvous; the work is in doing better than that, or in meeting when neither knows who should wait (Pelc 2019).
- **Derived problem: Map construction** (strengthening of). Building an isomorphic map implies having visited every node, so map construction is exploration with more output; it is used to solve rendezvous and leader election (Das 2019).
- **Derived problem: Exploration by oblivious robots** (variant of). The robots follow the Look–Compute–Move model of the plane, moved to graphs: they see the positions of all robots but remember nothing, so the configuration itself must encode the progress of the exploration.
- **Derived problem: Exploration of recolourable graphs** (variant of). The same task for one agent (visit every node; with return, stop or perpetually), but every node holds a colour from a palette of c colours that the agent at the node reads and, unless the colours are fixed, overwrites; the agent itself may have no memory at all, and in the main model it has no port numbers and sees its neighbours' colours instead. With $c = 1$ the colours carry nothing: an algorithm of the parent problem that uses no markers works here whenever this setting grants its ports, entry port and memory, and a lower bound of this problem proved for any number of colours holds for the parent setting with the same view. The parent's pebbles are movable markers carried by the agent (Pattanayak and Pelc 2024a; Disser et al. 2019); here the marks stay in the nodes and there is one per node.
- **Derived problem: Online exploration of weighted graphs** (variant of). The same task for one agent (visit every node and return to the start node), in the tradition of online algorithms: nodes carry identifiers, edges carry weights, memory and computation are unbounded, and an algorithm is judged by its competitive ratio against a shortest tour rather than by its memory. Results do not transfer in either direction without care: the parent counts moves on anonymous unweighted graphs, this problem counts weight on labelled weighted graphs. The setting directed-labelled of the parent (Deng and Papadimitriou 1999) is the analogue for unweighted directed graphs.
- **Derived problem: Collision-free exploration** (variant of). Every agent explores the graph as in exploration, and no two agents may ever be at the same node. A single-agent exploration procedure is the building block: with vision of radius 2, two agents run any such procedure with a slowdown of at most a factor $\log n$ (Bhagat and Pelc 2025) (a constant factor is claimed by Papin 2026). Lower bounds for exploring by one agent hold here whenever every agent must visit every node.
- **Related: Deciding graph properties by mobile agents.** A lone agent with constant memory and constant-size whiteboards explores by a depth-first search in port order (Izumi et al. 2022); without memory it cannot explore every graph (Cohen et al. 2008), which is why it cannot decide whether n is odd.

Assumptions specific to this problem

Knowledge

Knowing the size of the graph is the classical way to let an exploring walk end. Aleliunas, Karp, Lipton, Lovász and Rackoff proved that a random walk on a connected graph visits every node within $2m(n-1)$ expected steps from any start (Theorem 4), so by Markov's inequality $O(mn \log n) \subseteq O(n^3 \log n)$ steps cover it with high probability, and a step counter and the value of n give a randomized explorer that stops (Aleliunas et al. 1979); Reingold's log-space universal exploration sequences do the same deterministically when n is known (Reingold 2008) (Corollary 5.5). Without a bound and without a marker an agent cannot tell a cycle from a longer one, so it can explore perpetually but never stop

Claude (Anthropic) was used in cataloguing the problems.

(Fraigniaud et al. 2006c). Bender, Fernández, Ron, Sahai and Vadhan measured what the bound is worth in directed graphs: one pebble suffices when an upper bound on n is known, while without it $\Theta(\log \log n)$ pebbles are necessary and sufficient, the lower bound holding even for randomized robots (Bender et al. 2002). The top level, a full map, makes static exploration an offline planning task (a known undirected graph is explored by depth-first search in fewer than $2n$ steps); it matters for temporal graphs, where exploration with the whole schedule known in advance was introduced by Michail and Spirakis, who are given the whole temporal graph as input and show that the fastest exploration cannot be approximated within cn for some constant $c > 0$ unless P = NP (Theorem 1) (Michail and Spirakis 2016), and studied by Erlebach, Hoffmann and Kammer (Erlebach et al. 2021).

Level	Value	Formal definition
0	**nothing known**	The agent knows nothing about G: not n, m, Δ or D.
1	**knows a bound on n**	The agent is given an integer $N \geq n$ (an upper bound on the number of nodes) as input.
2	**knows the map**	The agent is given a graph isomorphic to G, with port numbers, and (where stated) its own position on it; for a dynamic graph, the whole sequence (G_t).

Markers

Pebbles come from the tradition of automata in mazes. After Budach proved that no finite automaton explores every finite maze (Budach 1978), Blum and Kozen showed that two pebbles suffice for a finite automaton in a maze, against Shah's conjecture that five were needed, and left open whether one pebble suffices (Blum and Kozen 1978); Hoffmann showed that it does not (as cited by Disser et al. 2019). Cohen, Fraigniaud, Ilcinkas, Korman and Peleg describe this "Rabin's pebble model" as the ability of the robot to label nodes dynamically, at most one unit of label per pebble (Cohen et al. 2008). Bender et al. carried the question to directed graphs and agents with unbounded memory, titling their paper "The power of a pebble": one pebble and a bound on n already give polynomial-time mapping (Bender et al. 2002). In anonymous graphs a marker is what makes stopping possible at all, and by convention a robot asked to stop is given a pebble (Fraigniaud et al. 2006c). The number of pebbles became a resource of its own when Disser, Hackfeld and Klimm showed that $\Theta(\log \log n)$ pebbles are necessary and sufficient for a constant-memory agent on undirected graphs, and that an agent with $O((\log n)^{1-\varepsilon})$ bits already needs $\Omega(\log \log n)$ of them (Disser et al. 2019); this is why one pebble and several pebbles are separate levels.

Level	Value	Formal definition
0	**no markers**	The agent cannot mark nodes.
1	**one pebble**	The agent carries one pebble that it can drop at its current node and later pick up; it sees whether a pebble is at its node.
2	**pebbles**	The agent carries p pebbles (identical or distinguishable, as stated); p is a resource reported with the memory.

Preprocessing

This axis asks what a designer who prepares the graph before the agent arrives can do for it, a question that grew out of the impossibility results of Budach and Rollik for finite automata (Budach 1978; Rollik 1980). Cohen, Fraigniaud, Ilcinkas, Korman and Peleg let the designer write short labels on the nodes and showed that three label values let a constant-memory robot explore every graph and stop; by Rollik's result two values are the minimum one could hope for (Cohen et al. 2008). They place the model next to pebbles: it "can be considered stronger than Rabin's pebble model since labels are given in a preprocessing stage, but it can also be considered weaker since, once assigned to nodes, the labels cannot be modified", and they cite it as part of a line of work on short information supplied in advance, including the oracle-size measure of Fraigniaud, Ilcinkas and Pelc (Cohen et al. 2008; Fraigniaud et al.

Claude (Anthropic) was used in cataloguing the problems.

2006b). Choosing the port numbers is preprocessing that writes nothing on the nodes: Ilcinkas starts from Budach's result that no finite automaton explores all graphs when an adversary sets the local orientation, and asks for the shortest exploration period when the designer sets it, improving the period $10n$ of Dobrev et al. to $4n$ (Ilcinkas 2008). The two forms of preprocessing are kept on the same level because neither is known to simulate the other.

Level	Value	Formal definition
0	**adversarial ports**	Port numbers (and any node labels) are chosen by an adversary.
1	**designer ports**	Before the agent starts, the designer may choose the port numbering $\text{port}_v : \{\text{edges at } v\} \to \{1, \dots, \deg(v)\}$ at every node. Ilcinkas calls a designer's port numbering a *local orientation* (Ilcinkas 2008).
1	**designer labels**	Before the agent starts, the designer may write a label of b bits (a constant) at every node, which the agent reads on arrival.

Termination requirement

The three termination requirements were separated in the memory-bounded exploration literature of the early 2000s. Fraigniaud, Ilcinkas, Peer, Pelc and Peleg list them explicitly, exploration with return, exploration with stop and perpetual exploration, and recall that Diks, Fraigniaud, Kranakis and Pelc showed for trees that exploration with stop needs $\Omega(\log \log \log n)$ bits of memory (Fraigniaud et al. 2005a; Diks et al. 2004); the same paper shows that exploration with return needs $\Omega(\log n)$ bits (Diks et al. 2004) (as summarised in Disser et al. 2019 and Czyzowicz et al. 2012c), while perpetual exploration of bounded-degree trees is possible with $O(1)$ bits (Fraigniaud et al. 2006c). In arbitrary anonymous graphs the requirement also decides feasibility: a robot cannot recognise that it has closed a cycle, so without a marker or a bound on n it can explore perpetually but never stop (Fraigniaud et al. 2006c). The axis orders the requirements from the hardest (return) to the weakest (perpetual); periodic exploration, which asks every node to be visited infinitely often within a bounded period, is a perpetual task measured by its period (Ilcinkas 2008).

Level	Value	Formal definition
0	**with return**	The agent visits every node and then halts at its start node s.
1	**with stop**	The agent visits every node and then halts, at any node. Also called *terminating exploration*.
2	**perpetual**	Every node is visited at least once (and, for periodic exploration, infinitely often); the agent never needs to halt.

Case tree

The root assumes nothing; going down, each edge adds the assumptions written in grey (+) or leaves an axis unassumed. Bold entries are catalogued settings with their status per measure (Time, Memory) and page; italic entries are branch points, named after the axis on which their branches differ. Axis order: Graph dynamics → Edge direction → Node identifiers → Knowledge → Randomness → Markers → Preprocessing → Termination requirement → Topology.

No assumptions
+ static graph
Edge direction
+ undirected
Knowledge

Claude (Anthropic) was used in cataloguing the problems.

nothing known, no markers, adversarial ports, + perpetual
Arbitrary graphs, perpetual exploration STANDARD SETTING p. 93

+ tree
Trees, perpetual exploration p. 93

nothing known, no markers, adversarial ports, + with stop
Exploration with stop, nothing known p. 94

+ tree
Trees, exploration with stop p. 94

nothing known, + pebbles
Exploration with return and pebbles p. 95

+ perpetual
Memoryless automaton with pebbles p. 96

+ knows a bound on n
Randomness

deterministic, + with stop
Exploration with stop when n is known p. 96

+ private coins, + with stop
Randomized exploration when n is known p. 97

nothing known, + one pebble, + with stop
Exploration with stop and one pebble p. 98

nothing known, no markers, adversarial ports, with return, + tree
Trees, exploration with return p. 98

nothing known, no markers, + designer labels, + with stop
Node labels chosen by the designer p. 99

nothing known, no markers, + designer ports, + perpetual
Port numbers chosen by the designer (periodic exploration) p. 99

directed, anonymous nodes, nothing known, + one pebble, + with stop
Directed graphs, one pebble p. 100

directed, anonymous nodes, + knows a bound on n, + one pebble, + with stop
Directed graphs, one pebble, bound on n known p. 101

directed, + labelled nodes, + with stop
Directed graphs with labelled nodes p. 102

+ connected every round, + undirected
Node identifiers

anonymous nodes, nothing known, + perpetual, + ring
Dynamic rings p. 103

anonymous nodes, + knows the map, + perpetual
Temporal graphs, schedule known p. 103

+ labelled nodes, + with stop
T-interval-connected graphs, labelled nodes p. 105

Claude (Anthropic) was used in cataloguing the problems.

What is known

The best upper bound against the best lower bound in each catalogued setting, in tree order. The settings follow in detail.

Setting	Time	Memory	p.
Arbitrary graphs, perpetual exploration STANDARD SETTING	● unknown — vs —	● tight $O(D \log \Delta)$ vs $\Omega(D \log \Delta)$	93
Trees, perpetual exploration	● unknown — vs —	● tight $O(\log \Delta)$ vs $\lceil \log \Delta \rceil$	93
Exploration with stop, nothing known	● impossible	● impossible	94
Trees, exploration with stop	● unknown — vs —	● polynomial gap $O(\log n)$ vs $\Omega(\log \log \log n)$	94
Exploration with return and pebbles	● unknown $n^{O(1)}$ vs —	● tight $O(1)$ bits, $O(\log \log n)$ pebbles vs $\Omega(\log \log n)$ pebbles	95
Memoryless automaton with pebbles	● unknown — vs —	● tight 0 bits, 2 pebbles vs 2 pebbles	96
Exploration with stop when n is known	● unknown $n^{O(1)}$ vs —	● near-optimal $O(\log n)$ vs $\omega((\log n)^{1-\varepsilon})$	96
Randomized exploration when n is known	● unknown $O(mn)$ vs —	● unknown $O(\log n)$ vs —	97
Exploration with stop and one pebble	● unknown — vs —	● polynomial gap $O(D \log \Delta)$ vs $\Omega(\log n)$	98
Trees, exploration with return	● unknown — vs —	● tight $O(\log n)$ vs $\Omega(\log n)$	98
Node labels chosen by the designer	● unknown $O(m)$ vs —	● tight $O(1)$ vs —	99
Port numbers chosen by the designer (periodic exploration)	● tight $3.5n - 2$ vs $2n - 2$	● tight $O(1)$ vs —	99
Directed graphs, one pebble	● unknown *exponential* vs —	● polynomial gap $O(n\Delta_{out} \log n)$ vs $\Omega(n \log \Delta_{out})$	100
Directed graphs, one pebble, bound on n known	● unknown $O(N^2 n^6 \Delta_{out}^2)$ vs —	● unknown — vs —	101
Directed graphs with labelled nodes	● polynomial gap $O(\phi^8 m)$ vs $\Omega(\phi^2 m)$	● unknown — vs —	102
Dynamic rings	● tight $n + \frac{n}{\max\{1,T-1\}}(\delta - 1) + O(\delta)$ vs $n + \frac{n}{\max\{1,T-1\}}(\delta - 1) - O(\delta)$	● unknown — vs —	103
Temporal graphs, schedule known	● tight $O(n^2)$ vs $\Omega(n^2)$	● unknown — vs —	103
T-interval-connected graphs, labelled nodes	● near-optimal $O((m - n + 1)\, n + n \log^2 n)$ vs $\Omega((m - n + 1)\, n)$	● unknown — vs —	105

Claude (Anthropic) was used in cataloguing the problems.

16.1.1 Arbitrary graphs, perpetual exploration

Adds nothing known, no markers, adversarial ports, + perpetual STANDARD SETTING VERIFIED

The classical memory question: how many bits must a finite-state agent have to visit every node of every graph of diameter D and degree Δ?

Assumed: graph dynamics: static graph; edge direction: undirected; termination requirement: visit every node; never required to stop (perpetual). **Not assumed:** node identifiers: anonymous nodes; knowledge: nothing known about the graph; randomness: deterministic; markers: no markers; preprocessing: ports and nodes as given by an adversary; topology: arbitrary connected graph.

Measure	Lower bound	Best upper bound	Status
Time	none known The paper does not study time ("time is not an issue in our approach").	—	● unknown
Memory	$\Omega(D \log \Delta)$ (Fraigniaud et al. 2005a) $\Theta(D \log \Delta)$ bits in the worst case over graphs of diameter D and degree Δ.	$O(D \log \Delta)$ (Fraigniaud et al. 2005a)	● tight

Results.

- Fraigniaud et al. (2005a) [lower bound]: memory $\Omega(D \log \Delta)$. *Bottleneck:* trap graphs. *Holds when:* deterministic agent, no knowledge of the graph; degree $\Delta \geq 3$; the traps are planar; Theorem 4 and Corollary 5: for every K-state agent there is a planar trap of degree Δ and diameter $O(\log K / \log \Delta)$. Corollary 3 gives $\lceil \log n \rceil$ bits directly: for every K-state agent there is a planar trap with at most $K + 1$ nodes (Theorem 2; with loops and multiple edges, or $K + \Delta + 2$ nodes for simple graphs). Earlier: no finite automaton explores all mazes (Budach 1978); no finite team of automata explores all cubic planar graphs (Rollik 1980).
- Fraigniaud et al. (2005a) [current best]: memory $O(D \log \Delta)$. Introductory and current best (Increasing-DFS, Theorem 6 and Corollary 7): depth-first searches of depth $1, 2, 3, \ldots$; no knowledge of D or Δ is needed. The bound is the memory used until exploration is complete; by Theorem 8 any agent that explores all graphs uses unboundedly many states over time.

Open questions.

Q1. What is the time of memory-optimal perpetual exploration, and is there a trade-off between memory and time?

Notes. In (Fraigniaud et al. 2005a) exploring means traversing every edge, not only visiting every node. The model goes back to maze search: Shannon's maze-solving mouse, Budach's theorem that no finite automaton explores all mazes (Budach 1978), Blum and Kozen (Blum and Kozen 1978), and Rollik's result that no finite team of automata explores all cubic planar graphs (Rollik 1980). Blum and Kozen proved that a finite automaton with two pebbles, or two automata together, search every maze, and that no three finite automata together can search all finite planar cubic graphs (Theorem 6.1; also one automaton, Theorem 4.1, and two, Theorem 4.6), so mazes, which come with a compass, are strictly easier than graphs; whether any finite team suffices for planar cubic graphs was left open (Blum and Kozen 1978). Their automata need not halt. This confirms the account of (Fraigniaud et al. 2005a); the reading in (Cohen et al. 2008) that one agent with three pebbles cannot explore all graphs is not a result of (Blum and Kozen 1978). Fraigniaud, Ilcinkas and Pelc showed that a polynomial increase in the number of states strictly enlarges the class of explorable regular graphs (DAM 2008, not in the bibliography yet). Checked against the full text.

16.1.2 Trees, perpetual exploration

Adds + tree **to** Arbitrary graphs, perpetual exploration (p. 93) VERIFIED

On trees, perpetual exploration needs exactly $\Theta(\log \Delta)$ bits: enough to remember a port.

Claude (Anthropic) was used in cataloguing the problems.

Assumed: graph dynamics: static graph; edge direction: undirected; termination requirement: visit every node; never required to stop (perpetual); topology: tree. **Not assumed:** node identifiers: anonymous nodes; knowledge: nothing known about the graph; randomness: deterministic; markers: no markers; preprocessing: ports and nodes as given by an adversary.

Measure	Lower bound	Best upper bound	Status
Time	none known Time not stated in the sources read.	—	● unknown
Memory	$\lceil \log \Delta \rceil$ (Diks et al. 2004) $\Theta(\log \Delta)$ bits.	$O(\log \Delta)$ (Diks et al. 2004)	● tight

Results.

- Diks et al. (2004) [lower bound]: memory $\lceil \log \Delta \rceil$. *Bottleneck:* port addressing. *Holds when:* trees of maximum degree Δ; deterministic.
- Diks et al. (2004) [current best]: memory $O(\log \Delta)$. Introductory and current best. Equivalently, the sequence 1^n is a universal exploration sequence for trees (Koucký 2002): leave by the port after the entry port.

16.1.3 Exploration with stop, nothing known

Adds nothing known, no markers, adversarial ports, + with stop VERIFIED

Without a bound on n and without a way to mark a node, an agent can never be sure it has seen everything.

Assumed: graph dynamics: static graph; edge direction: undirected; termination requirement: explore, then stop anywhere. **Not assumed:** node identifiers: anonymous nodes; knowledge: nothing known about the graph; randomness: deterministic; markers: no markers; preprocessing: ports and nodes as given by an adversary; topology: arbitrary connected graph.

Measure	Lower bound	Best upper bound	Status
Time	*impossible* (Dieudonné and Pelc 2012) Feasible with one unmovable token at the start node (Dieudonné and Pelc 2012).	—	● impossible
Memory	*impossible* (Dieudonné and Pelc 2012) Impossible for any memory size.	—	● impossible

Results.

- Dieudonné and Pelc (2012) [lower bound]: time *impossible. Bottleneck:* no way to recognise the start. *Holds when:* deterministic agent, no upper bound on n, no marking of nodes; unbounded memory does not help; even cycles cannot be explored with stop without a way to mark a node (Fraigniaud et al. 2006c).
- Dieudonné and Pelc (2012) [lower bound]: memory *impossible. Bottleneck:* no way to recognise the start. *Holds when:* same conditions as for time.

Notes. The impossibility is why every stopping variant below adds either knowledge (a bound on n) or a way to mark a node (a pebble or a token). Dieudonné and Pelc also study Byzantine tokens: identical unmovable tokens, some of which may be faulty; exploration is feasible in polynomial cost if a bound on the number of tokens is known and at least one is fault-free (Dieudonné and Pelc 2012).

16.1.4 Trees, exploration with stop

Adds + tree **to** Exploration with stop, nothing known (p. 94) VERIFIED

Stopping on a tree without knowing n needs more than constant memory, but how much more is open: $\Omega(\log \log \log n)$ *against* $O(\log n)$.

Claude (Anthropic) was used in cataloguing the problems.

Assumed: graph dynamics: static graph; edge direction: undirected; termination requirement: explore, then stop anywhere; topology: tree. **Not assumed:** node identifiers: anonymous nodes; knowledge: nothing known about the graph; randomness: deterministic; markers: no markers; preprocessing: ports and nodes as given by an adversary.

Measure	Lower bound	Best upper bound	Status
Time	none known Time not studied.	—	● unknown
Memory	$\Omega(\log\log\log n)$ (Diks et al. 2004) $O(\log n)$ versus $\Omega(\log\log\log n)$: exponentially apart.	$O(\log n)$ (Ambühl et al. 2011)	● polynomial gap

Results.

- Diks et al. (2004) [lower bound]: memory $\Omega(\log\log\log n)$. *Bottleneck:* deciding when to stop. *Holds when:* bounded-degree trees built from paths; n unknown; deterministic.
- Ambühl et al. (2011) [current best]: memory $O(\log n)$. Via exploration with return, which implies stop.

Open questions.

Q1. Can exploration with stop on trees use asymptotically less memory than exploration with return? (Open problem of Diks et al. 2004.) Equivalently: close $\Omega(\log\log\log n)$ versus $O(\log n)$.

Notes. If an upper bound N on n is known, $O(\log N)$ bits suffice; if a bound m on the number of nodes of some degree is known, $O(\log\Delta + \log m)$ bits (Diks et al. 2004). "The additional memory is essentially needed to decide when to stop."

16.1.5 Exploration with return and pebbles

Adds nothing known, + pebbles VERIFIED

$\Theta(\log\log n)$ pebbles are necessary and sufficient for a constant-memory agent, which then explores in polynomial time and returns to the start without knowing n.

Assumed: graph dynamics: static graph; edge direction: undirected; markers: several pebbles; their number is a resource. **Not assumed:** node identifiers: anonymous nodes; knowledge: nothing known about the graph; randomness: deterministic; preprocessing: ports and nodes as given by an adversary; termination requirement: explore, then stop at the start node; topology: arbitrary connected graph.

Measure	Lower bound	Best upper bound	Status
Time	none known Polynomial; the exponent is not tracked.	$n^{O(1)}$ (Disser et al. 2019)	● unknown
Memory	$\Omega(\log\log n)$ pebbles (Disser et al. 2019) $\Theta(\log\log n)$ pebbles with $O(1)$ bits.	$O(1)$ bits, $O(\log\log n)$ pebbles (Disser et al. 2019)	● tight

Results.

- Disser et al. (2019) [lower bound]: memory $\Omega(\log\log n)$ pebbles. *Bottleneck:* trap graphs. *Holds when:* for any agent with $O((\log n)^{1-\varepsilon})$ bits; holds for every termination requirement; deterministic.
- Disser et al. (2019) [current best]: time $n^{O(1)}$, memory $O(1)$ bits, $O(\log\log n)$ pebbles. *Assumes:* distinguishable pebbles; n unknown. Introductory and current best (SODA 2016 for bounded degree, JACM 2019 in general). Since $p+1$ agents can simulate one agent with p pebbles, the same bounds hold for teams: $\Theta(\log\log n)$ constant-memory agents.

Open questions.

Claude (Anthropic) was used in cataloguing the problems.

Q1. How much memory is needed to encode the agent's transition function? The naive table needs $(\log n)^{O(1)}$ bits (Remark 1 of Disser et al. 2019).

Notes. The trade-off reads: $\Omega(\log n)$ bits make pebbles unnecessary; $O((\log n)^{1-\varepsilon})$ bits force $\Omega(\log\log n)$ pebbles; $\Omega(\log\log n)$ pebbles make $O(1)$ bits enough. The traps of (Disser et al. 2019) have $O(K^{2^{5k}})$ nodes for k agents with K states, improving Rollik's tower-type bound (Rollik 1980).

16.1.6 Memoryless automaton with pebbles

Adds + perpetual **to** Exploration with return and pebbles (p. 95) VERIFIED

A deterministic memoryless automaton (one state) explores every connected graph with 2 pebbles, and 2 is the minimum.

Assumed: graph dynamics: static graph; edge direction: undirected; markers: several pebbles; their number is a resource; termination requirement: visit every node; never required to stop (perpetual). **Not assumed:** node identifiers: anonymous nodes; knowledge: nothing known about the graph; randomness: deterministic; preprocessing: ports and nodes as given by an adversary; topology: arbitrary connected graph.

Statement for this setting. The agent is a deterministic finite automaton with a single state: its decision depends only on what it observes at the current node (entry port, degree, pebbles present), not on a state. It carries p pebbles labelled $1, \dots, p$, which it can drop and pick up. Exploration is complete when every edge has been traversed at least once.

Measure	Lower bound	Best upper bound	Status
Time	none known Time not studied.	—	● unknown
Memory	2 pebbles (Pattanayak and Pelc 2024a) Two pebbles are necessary and sufficient.	0 bits, 2 pebbles (Pattanayak and Pelc 2024a)	● tight

Results.

- Pattanayak and Pelc (2024a) [lower bound]: memory 2 pebbles. *Bottleneck:* trap graphs. *Holds when:* one pebble is insufficient in general, even if n is known; a memoryless automaton without pebbles cannot explore all graphs.
- Pattanayak and Pelc (2024a) [current best]: memory 0 bits, 2 pebbles. *Assumes:* n and Δ unknown. Introductory and current best. If n is known and the automaton starts at a node with a pebble, one pebble suffices.

Notes. Results from the paper's abstract, supplied by the catalogue owner (a co-author).

16.1.7 Exploration with stop when n is known

Adds deterministic, + with stop VERIFIED

Reingold's log-space universal exploration sequences give an $O(\log n)$-bit agent that explores and stops in polynomial time.

Assumed: graph dynamics: static graph; edge direction: undirected; knowledge: an upper bound on n is known; termination requirement: explore, then stop anywhere. **Not assumed:** node identifiers: anonymous nodes; randomness: deterministic; markers: no markers; preprocessing: ports and nodes as given by an adversary; topology: arbitrary connected graph.

Claude (Anthropic) was used in cataloguing the problems.

Measure	Lower bound	Best upper bound	Status
Time	none known	$n^{O(1)}$ (Reingold 2008)	● unknown
	Polynomial; the exponent is not tracked and no non-trivial lower bound is recorded.		
Memory	$\omega((\log n)^{1-\varepsilon})$ (Disser et al. 2019)	$O(\log n)$ (Reingold 2008)	● near-optimal
	$O(\log n)$ versus $\omega((\log n)^{1-\varepsilon})$: a $(\log n)^{\varepsilon}$ factor.		

Results.

- Disser et al. (2019) [lower bound]: memory $\omega((\log n)^{1-\varepsilon})$. *Bottleneck:* trap graphs. *Holds when:* for every constant $\varepsilon > 0$; the agent may depend on n: it must explore all graphs with at most n nodes; no pebbles; deterministic. Corollary 25 of (Disser et al. 2019): with $O((\log n)^{1-\varepsilon})$ bits an agent needs $\Omega(\log \log n)$ pebbles.
- Reingold (2008) [current best]: time $n^{O(1)}$, memory $O(\log n)$. Introductory and current best. A log-space constructible universal exploration sequence (Koucký 2002) guides a walk through all nodes; the consequence for agents is stated by (Disser et al. 2019).

Open questions.

Q1. Is $\Omega(\log n)$ bits necessary when n is known? The trap arguments (Fraigniaud et al. 2005a) are stated for agents that know nothing.

Notes. Whether the same agent can also return to s (by reversing the sequence) is plausible but not checked against a source.

16.1.8 Randomized exploration when n is known

Adds + private coins, + with stop VERIFIED

A random walk covers any graph in expected $2m(n-1)$ steps; with a step counter and a bound on n it stops after exploring with good probability.

Assumed: graph dynamics: static graph; edge direction: undirected; knowledge: an upper bound on n is known; randomness: randomized (agents have private coins); termination requirement: explore, then stop anywhere. **Not assumed:** node identifiers: anonymous nodes; markers: no markers; preprocessing: ports and nodes as given by an adversary; topology: arbitrary connected graph.

Measure	Lower bound	Best upper bound	Status
Time	none known	$O(mn)$ (Aleliunas et al. 1979)	● unknown
	No lower bound recorded for randomized agents.		
Memory	none known	$O(\log n)$ (Aleliunas et al. 1979)	● unknown
	No lower bound recorded for randomized agents.		

Results.

- Aleliunas et al. (1979) [current best]: time $O(mn)$, memory $O(\log n)$. *Assumes:* expected time; stops correctly with constant probability (Monte Carlo). Expected cover time at most $2m(n-1)$ from any start (Theorem 4). Running the walk for a multiple of that many steps with an $O(\log n)$-bit counter gives the Monte Carlo agent.

Open questions.

Q1. What is the memory needed by a randomized agent that must stop correctly with high probability, as a function of n?

Notes. The same paper proves by the probabilistic method that universal traversal sequences of length $O(n^3 \log n)$ exist for connected Δ-regular graphs, the constant depending on Δ (the proof uses length

Claude (Anthropic) was used in cataloguing the problems.

$2\Delta n(n-1)(\Delta n+2)\lceil \log n \rceil$) (Aleliunas et al. 1979). For directed graphs the cover time can be exponential, while Eulerian digraphs behave like undirected graphs (Aleliunas et al. 1979); Reingold later constructed universal exploration sequences in log space (Reingold 2008).

16.1.9 Exploration with stop and one pebble

Adds nothing known, + one pebble, + with stop VERIFIED

One pebble makes stopping possible; the agent then needs $\Omega(\log n)$ bits.

Assumed: graph dynamics: static graph; edge direction: undirected; markers: one pebble (a movable marker); termination requirement: explore, then stop anywhere. **Not assumed:** node identifiers: anonymous nodes; knowledge: nothing known about the graph; randomness: deterministic; preprocessing: ports and nodes as given by an adversary; topology: arbitrary connected graph.

Measure	Lower bound	Best upper bound	Status
Time	none known Time not studied.	—	● unknown
Memory	$\Omega(\log n)$ (Fraigniaud et al. 2006c) $O(D \log \Delta)$ versus $\Omega(\log n)$: polynomial when D is large.	$O(D \log \Delta)$ (Fraigniaud et al. 2005b)	● polynomial gap

Results.

- Fraigniaud et al. (2006c) [lower bound]: memory $\Omega(\log n)$. *Bottleneck:* reduced automata. *Holds when:* deterministic agent with one pebble that starts on the agent's start node; no knowledge of n; the traps are edge-coloured regular graphs of size $O(K^3)$ for a K-state agent (Theorem 5 of Fraigniaud et al. 2006c).
- Fraigniaud et al. (2005b) [current best]: memory $O(D \log \Delta)$. Introductory and current best (Theorem 3, DFS-with-stop): the pebble marks the start; depth-first phases of depth $1, 2, \ldots$ detect new nodes, and the agent stops after phase $D+1$. No knowledge of D or Δ. Time is not analysed.

Open questions.

Q1. Is $\Omega(D \log \Delta)$ memory needed for exploration with stop by an agent with one pebble (Fraigniaud et al. 2005b)?

Notes. The same authors show that k non-cooperative agents need $\Omega(\log(n/k))$ bits for perpetual exploration, with traps of size $O(kK)$ (Fraigniaud et al. 2006c). That chapter is inconsistent on tightness: a remark calls the $\Omega(\log n)$ bound tight, citing the SIROCCO paper, while its conclusion says a tight bound for terminating exploration is open. Checked against the full texts.

16.1.10 Trees, exploration with return

Adds nothing known, no markers, adversarial ports, with return, + tree VERIFIED

Returning to the start on a tree takes $\Theta(\log n)$ bits.

Assumed: graph dynamics: static graph; edge direction: undirected; topology: tree. **Not assumed:** node identifiers: anonymous nodes; knowledge: nothing known about the graph; randomness: deterministic; markers: no markers; preprocessing: ports and nodes as given by an adversary; termination requirement: explore, then stop at the start node.

Measure	Lower bound	Best upper bound	Status
Time	none known Time not stated in the abstracts read.	—	● unknown
Memory	$\Omega(\log n)$ (Diks et al. 2004) $\Theta(\log n)$ bits.	$O(\log n)$ (Ambühl et al. 2011)	● tight

Claude (Anthropic) was used in cataloguing the problems.

Results.

- Diks et al. (2004) [lower bound]: memory $\Omega(\log n)$. *Bottleneck:* pigeonhole on start nodes. *Holds when:* every n-node tree; deterministic; holds even if the agent knows the tree up to its start node, so knowing n does not help.
- Diks et al. (2004) [introductory]: memory $O(\log^2 n)$.
- Ambühl et al. (2011) [current best]: memory $O(\log n)$. Also works when ports are only circularly ordered.

Notes. (Diks et al. 2004) asked for the exact memory of exploration with return; (Ambühl et al. 2011) answered it.

16.1.11 Node labels chosen by the designer

Adds nothing known, no markers, + designer labels, + with stop VERIFIED

With three-valued labels on the nodes, an agent with fewer than 25 states explores every graph and stops after at most $20m$ edge traversals.

Assumed: graph dynamics: static graph; edge direction: undirected; preprocessing: short node labels chosen by the designer; termination requirement: explore, then stop anywhere. **Not assumed:** node identifiers: anonymous nodes; knowledge: nothing known about the graph; randomness: deterministic; markers: no markers; topology: arbitrary connected graph.

Measure	Lower bound	Best upper bound	Status
Time	none known Only the trivial $\Omega(n)$ is known against $O(m)$.	$O(m)$ (Cohen et al. 2008)	● unknown
Memory	none known Constant memory.	$O(1)$ (Cohen et al. 2008)	● tight

Results.

- Cohen et al. (2008) [current best]: time $O(m)$, memory $O(1)$. *Assumes:* labels of 2 bits (3 values), computed in linear time. Introductory and current best (Theorem 2.1): fewer than 25 states, at most $20m$ edge traversals, stops at a node chosen by the designer. Labelling while exploring gives $O(Dm)$.
- Cohen et al. (2008) [also]: time $O(\Delta^{10}m)$, memory $O(\log \Delta)$. *Assumes:* labels of 1 bit. Theorem 3.1.

Notes. The agent is a Mealy automaton: its output port depends on the input port. Agents with a single state cannot explore all graphs with such labels (Theorems 4.2 and 4.5 of Cohen et al. 2008). Recent work changes the model: zero-memory agents that do not see the entry port but may colour nodes (SIROCCO 2023, OPODIS 2025); these are not catalogued yet. Checked against the full text.

16.1.12 Port numbers chosen by the designer (periodic exploration)

Adds nothing known, no markers, + designer ports, + perpetual VERIFIED

If the designer may set the port numbers, a constant-memory agent visits every node periodically with period between $2n-2$ and $3.5n-2$.

Assumed: graph dynamics: static graph; edge direction: undirected; preprocessing: port numbers chosen by the designer; termination requirement: visit every node; never required to stop (perpetual). **Not assumed:** node identifiers: anonymous nodes; knowledge: nothing known about the graph; randomness: deterministic; markers: no markers; topology: arbitrary connected graph.

Claude (Anthropic) was used in cataloguing the problems.

Measure	Lower bound	Best upper bound	Status
Time	$2n-2$ (Czyzowicz et al. 2012c) $\Theta(n)$; the constant is open: $2n-2$ versus $3.5n-2$ (constant memory), $2.8n-O(1)$ versus $4n-2$ (oblivious).	$3.5n-2$ (Czyzowicz et al. 2012c)	• tight
Memory	none known Constant memory.	$O(1)$ (Czyzowicz et al. 2012c)	• tight

Results.

- Czyzowicz et al. (2012c) [lower bound]: time $2n-2$. *Bottleneck:* every tree edge twice. *Holds when:* any agent, whatever its memory: on a tree every edge is traversed in both directions; folklore, stated in (Czyzowicz et al. 2012c) and (Gąsieniec et al. 2008); for oblivious agents the bound is $2.8n-O(1)$: for $n=5k+l$ with $l<5$ some n-node graph has no witness cycle shorter than $14k+2l$ (Theorem 13 of Czyzowicz et al. 2012c).
- Ilcinkas (2008) [introductory]: time $4n-2$, memory $O(1)$. *Assumes:* three-state agent. Theorem 1; the abstract rounds the period to $4n$.
- Gąsieniec et al. (2008) [also]: time $3.75n-2$, memory $O(1)$. Period less than $3.75n-2$ (Theorem 2), with port numbers set along a spanning tree; this disproved Ilcinkas's conjecture that $4n-O(1)$ is optimal for any memory (Ilcinkas 2008).
- Czyzowicz et al. (2012c) [current best]: time $3.5n-2$, memory $O(1)$. For oblivious agents the same paper gives $(4+\frac{1}{3})n-4$; the best oblivious period is $4n-2$ (Kosowski and Navarra 2012).
- Kosowski and Navarra (2012) [also]: time $4n-2$, memory 0. *Assumes:* oblivious agent following the right-hand rule: it leaves by port $(i \bmod \deg(v))+1$ after entering by port i. Theorem 3; the labelling is computed in polynomial time (Theorem 4). It is the best possible for labellings in which the edges traversed twice form a connected spanning subgraph: $K_{2,l}$ needs $4n-8$ under that restriction (Theorem 5).

Open questions.

Q1. Close the constant gaps: $2n-2$ versus $3.5n-2$ for constant memory, $2.8n-O(1)$ versus $4n-2$ for oblivious agents (Czyzowicz et al. 2012c; Kosowski and Navarra 2012).

Q2. Trade-offs between period and memory: the only lower bound for agents with memory does not depend on memory.

Notes. Without the designer's help, periodic exploration by a finite automaton is impossible (Budach, as cited by Czyzowicz et al. 2012c). These papers began with Dobrev, Jansson, Sadakane and Sung (SIROCCO 2005; period $10n$, as cited by Czyzowicz et al. 2012c). Computing the shortest witness cycle for an oblivious agent is NP-hard. The period is defined differently: Ilcinkas counts returns to a node in the same state through the same port, Czyzowicz et al. returns to a node through the same port. For oblivious agents the right-hand rule is no restriction: the shortest period over all labellings is the same for every oblivious strategy, as Kosowski and Navarra note citing (Czyzowicz et al. 2012c; Kosowski and Navarra 2012).

16.1.13 Directed graphs, one pebble

Adds directed, anonymous nodes, nothing known, + one pebble, + with stop VERIFIED

An agent that cannot walk back along an arc needs $\Omega(n \log \Delta_{out})$ bits, even with up to n pebbles; with one pebble, $O(n\Delta_{out} \log n)$ bits suffice, in exponential time.

Assumed: graph dynamics: static graph; markers: one pebble (a movable marker); termination requirement: explore, then stop anywhere. **Not assumed:** edge direction: directed (strongly connected); node identifiers: anonymous nodes; knowledge: nothing known about the graph; randomness: deterministic; preprocessing: ports and nodes as given by an adversary; topology: arbitrary connected graph.

Claude (Anthropic) was used in cataloguing the problems.

Measure	Lower bound	Best upper bound	Status
Time	none known	*exponential* (Fraigniaud and Ilcinkas 2004)	● unknown
	The one-pebble algorithm takes exponential time; there is no lower bound on time beyond the trivial one.		
Memory	$\Omega(n \log \Delta_{out})$ (Fraigniaud and Ilcinkas 2004) Factor $\Delta_{out} \log n / \log \Delta_{out}$.	$O(n\Delta_{out} \log n)$ (Fraigniaud and Ilcinkas 2004)	● polynomial gap

Results.

- Fraigniaud and Ilcinkas (2004) [lower bound]: memory $\Omega(n \log \Delta_{out})$. *Bottleneck:* no backtracking. *Holds when:* strongly connected digraphs, maximum out-degree Δ_{out}; the agent is a deterministic finite automaton; a combination lock (Definition 1): nodes $u_0, \dots, u_{n-1}$, all of out-degree Δ_{out}; each u_i with $i < n-1$ has one arc to u_{i+1} and its other arcs to u_0, and every arc of u_{n-1} leads to u_0; proved for perpetual exploration (Theorem 1), so it holds for exploration with stop; holds even with up to n pebbles when $\Delta_{out} > 2$, and with up to $n/2$ pebbles when $\Delta_{out} = 2$.
- Fraigniaud and Ilcinkas (2004) [current best]: time *exponential*, memory $O(n\Delta_{out} \log n)$. Introductory and current best (Algorithm Test-all-maps, Theorem 2). The agent needs no knowledge of n: it tries $n = 1, 2, \dots$ and every map of order n in turn, checking each with its pebble, so the time is exponential in the worst case. With whiteboards instead of a pebble, constant agent memory and $O(\log \Delta_{out})$ bits per node suffice.

Open questions.

Q1. Close $\Omega(n \log \Delta_{out})$ versus $O(n\Delta_{out} \log n)$ bits for exploring digraphs with one pebble.

Q2. Determine the trade-off between time and memory: polynomial-time exploration with stop is known only with $O(n^2\Delta_{out} \log n)$ bits and $O(\log \log n)$ pebbles, far from the $\Omega(n \log \Delta_{out})$ lower bound (Section 5 of Fraigniaud and Ilcinkas 2004).

Notes. Here Δ_{out} is the maximum out-degree (d in Fraigniaud and Ilcinkas 2004). With $O(\log \log n)$ pebbles, Algorithm Compacted-Explore-and-Map, a variant of the algorithm of (Bender et al. 2002), explores with stop in polynomial time using $O(n^2\Delta_{out} \log n)$ bits (Theorem 3), and $\Omega(\log \log n)$ pebbles are necessary for polynomial time when no bound on n is known (Bender et al. 2002). Without any memory an agent cannot explore with stop even with whiteboards (Theorem 4). Bender and Slonim showed that two cooperating agents learn any unlabelled digraph in expected polynomial time (Bender and Slonim 1994). Checked against the full text of (Fraigniaud and Ilcinkas 2004) (model, Theorems 1 to 4 and the conclusion).

16.1.14 Directed graphs, one pebble, bound on n known

Adds directed, anonymous nodes, + knows a bound on n, + one pebble, + with stop VERIFIED

With a bound $N \geq n$, one pebble lets an agent map any strongly connected digraph in polynomial time.

Assumed: graph dynamics: static graph; knowledge: an upper bound on n is known; markers: one pebble (a movable marker); termination requirement: explore, then stop anywhere. **Not assumed:** edge direction: directed (strongly connected); node identifiers: anonymous nodes; randomness: deterministic; preprocessing: ports and nodes as given by an adversary; topology: arbitrary connected graph.

Measure	Lower bound	Best upper bound	Status
Time	none known No lower bound beyond the trivial $\Omega(m)$.	$O(N^2n^6\Delta_{out}^2)$ (Bender et al. 2002)	● unknown
Memory	none known Memory is not bounded in these papers.	—	● unknown

Claude (Anthropic) was used in cataloguing the problems.

Results.

- Bender et al. (2002) [current best]: time $O(N^2n^6\Delta_{out}^2)$. *Assumes:* unbounded memory; all out-degrees padded to Δ_{out} with self-loops; Δ_{out} known. Introductory and current best (Theorem 3.1; N is the paper's $\hat{n}$, Δ_{out} its d). Without a bound on n, for every constant $c > 0$, $c \log\log n$ pebbles give a deterministic polynomial-time algorithm (Theorem 4.1), and $\Omega(\log\log n)$ pebbles are necessary for any algorithm, even randomized, that learns every graph in expected polynomial time with probability greater than $1/2$ (Theorem 4.2).

Open questions.

Q1. Improve the $O(N^2n^6\Delta_{out}^2)$ running time (Section 5 of Bender et al. 2002).

Notes. Checked against the full text of (Bender et al. 2002). The agent must output a graph isomorphic to G; it sees out-ports numbered $1, \ldots, \Delta_{out}$ but not in-degrees, and cannot retrace an arc to fetch its pebble. Bender and Slonim had shown that a constant number of pebbles does not suffice for polynomial time when no bound on n is known (as cited in Bender et al. 2002). With labelled nodes a greedy search takes $O(nm)$ moves.

16.1.15 Directed graphs with labelled nodes

Adds directed, + labelled nodes, + with stop DRAFT

An agent that recognises nodes but cannot backtrack must traverse every arc; the cost depends on the deficiency ϕ of the digraph.

Assumed: graph dynamics: static graph; node identifiers: labelled nodes (an agent can recognise a node it has visited); termination requirement: explore, then stop anywhere. **Not assumed:** edge direction: directed (strongly connected); knowledge: nothing known about the graph; randomness: deterministic; markers: no markers; preprocessing: ports and nodes as given by an adversary; topology: arbitrary connected graph.

Measure	Lower bound	Best upper bound	Status
Time	$\Omega(\phi^2 m)$ (Deng and Papadimitriou 1999) $O(\phi^8 m)$ versus $\Omega(\phi^2 m)$: factor ϕ^6.	$O(\phi^8 m)$ (Fleischer and Trippen 2005)	• polynomial gap
Memory	none known Memory is not bounded in these papers.	—	• unknown

Results.

- Deng and Papadimitriou (1999) [lower bound]: time $\Omega(\phi^2 m)$. *Bottleneck:* unbalanced arcs. *Holds when:* deficiency ϕ: the number of arcs to add to make the digraph Eulerian; deterministic strategies; the argument does not apply to randomized ones; due to Koutsoupias: Corollary 3 of (Deng and Papadimitriou 1999) gives at least $\phi^2/4$ times the number of arcs. Deng and Papadimitriou's own bound (Theorem 1) is about $\phi^2 m/(2\log\phi)$; against randomized strategies they show a ratio of $\Omega(\phi/\log\phi)$ to the optimal tour (Corollary 2).
- Deng and Papadimitriou (1999) [introductory]: time $\phi^{O(\phi)}\, m$. *Assumes:* unbounded memory. Theorem 3: no arc is traversed more than $2^{O(\phi\log\phi)}$ times, without knowing ϕ. Eulerian digraphs ($\phi = 0$): ratio 2 to the optimal tour, which is optimal; deficiency one: no arc traversed more than four times (Theorem 2).
- Albers and Henzinger (2000) [also]: time $\phi^{O(\log\phi)}\, m$. *Assumes:* unbounded memory. First subexponential algorithm; the authors also show that their algorithm needs $\phi^{\Omega(\log\phi)} m$ on some graphs, and that Greedy, Depth-First and Breadth-First need $2^{\Omega(\phi)} m$.
- Fleischer and Trippen (2005) [current best]: time $O(\phi^8 m)$. *Assumes:* unbounded memory. First algorithm polynomial in ϕ; the paper itself was not read, the bound is from the authors' abstract.

Open questions.

Q1. Close $\Omega(\phi^2 m)$ versus $O(\phi^8 m)$ for online exploration of digraphs with deficiency ϕ.

Claude (Anthropic) was used in cataloguing the problems.

Notes. This is the online-algorithms tradition: nodes are recognisable and the agent builds a map; cost is compared with the optimal tour (the Chinese postman tour), and Deng and Papadimitriou note that measured in absolute terms $\Theta(nm)$ is trivially tight. Checked against the full texts of (Deng and Papadimitriou 1999) and (Albers and Henzinger 2000); the bound of (Fleischer and Trippen 2005) is still from the authors' abstract. The deficiency was introduced by Kutten (Kutten 1988). For the related objective of visiting every node (not every arc), Foerster and Wattenhofer show a competitive ratio linear in n, with deterministic and randomized upper and lower bounds tight up to small constants (abstract read) (Foerster and Wattenhofer 2016).

16.1.16 Dynamic rings

Adds anonymous nodes, nothing known, + perpetual, + ring VERIFIED

On a T-interval-connected ring whose edges may be missing for up to δ steps, the worst-case exploration time is known exactly up to additive terms.

Assumed: graph dynamics: dynamic graph, connected in every round (1-interval connected); edge direction: undirected; termination requirement: visit every node; never required to stop (perpetual); topology: ring or path. **Not assumed:** node identifiers: anonymous nodes; knowledge: nothing known about the graph; randomness: deterministic; markers: no markers; preprocessing: ports and nodes as given by an adversary.

Measure	Lower bound	Best upper bound	Status
Time	$n + \frac{n}{\max\{1,T-1\}}(\delta-1) - O(\delta)$ (Ilcinkas and Wade 2018) Matching up to $\pm\Theta(\delta)$.	$n + \frac{n}{\max\{1,T-1\}}(\delta-1) + O(\delta)$ (Ilcinkas and Wade 2018)	• tight
Memory	none known Memory not studied.	—	• unknown

Results.

- Ilcinkas and Wade (2018) [lower bound]: time $n + \frac{n}{\max\{1,T-1\}}(\delta-1) - O(\delta)$. *Bottleneck:* adversarial edge changes. *Holds when:* underlying graph a ring; the agent does not know the dynamics; T-interval connectivity; δ = longest absence of an edge.
- Ilcinkas and Wade (2018) [current best]: time $n + \frac{n}{\max\{1,T-1\}}(\delta-1) + O(\delta)$. Introductory and current best. If the agent knows the dynamics, the worst case is $2n - T - \Theta(1)$.

Notes. Multi-agent exploration of dynamic rings and graphs, where the measure is how many agents are needed (Di Luna, Dobrev, Flocchini and Santoro; Gotoh, Flocchini, Masuzawa and Santoro), is not catalogued yet.

16.1.17 Temporal graphs, schedule known

Adds anonymous nodes, + knows the map, + perpetual VERIFIED

If the whole schedule of an always-connected temporal graph is known, $\Theta(n^2)$ steps are necessary and sufficient in the worst case.

Assumed: graph dynamics: dynamic graph, connected in every round (1-interval connected); edge direction: undirected; knowledge: a complete map is known (for a dynamic graph: the whole schedule); termination requirement: visit every node; never required to stop (perpetual). **Not assumed:** node identifiers: anonymous nodes; randomness: deterministic; markers: no markers; preprocessing: ports and nodes as given by an adversary; topology: arbitrary connected graph.

Claude (Anthropic) was used in cataloguing the problems.

Measure	Lower bound	Best upper bound	Status
Time	$\Omega(n^2)$ (Erlebach et al. 2021); $\Omega(\Delta n(1+\log(n/\Delta)))$ (Carnevale 2026) [preprint]	$O(n^2)$ (Erlebach et al. 2021)	• tight
	$\Theta(n^2)$ in the worst case. When every snapshot has maximum degree at most Δ the bounds are $\Omega(\Delta n(1+\log(n/\Delta)))$ (Carnevale 2026) and $O(n^{3/2}\sqrt{\Delta \log n})$ (Bastide et al. 2025), a polynomial gap (factor about $\sqrt{n/\Delta}$).		
Memory	none known Not a memory question.	—	• unknown

Results.

- Erlebach et al. (2021) [lower bound]: time $\Omega(n^2)$. *Bottleneck:* adversarial edge changes. *Holds when:* an explicit family of always-connected temporal graphs; holds for centralised, offline exploration.
- Carnevale (2026) [preprint] [lower bound]: time $\Omega(\Delta n(1+\log(n/\Delta)))$. *Bottleneck:* adversarial edge changes. *Holds when:* underlying graph (the union of all snapshots) of maximum degree at most Δ, for $\Delta_0 \le \Delta \le n-1$ with an absolute constant Δ_0; every snapshot is a spanning tree in which only one node has degree above three; deterministic construction, polynomial in n; holds from every start node; also a lower bound when only the degree of each snapshot is at most Δ. Theorem 1.1 (arXiv v1). In each phase of $\Theta(\Delta n)$ steps, a cycle whose missing edge advances by one position per step lets the agent enter fewer than half of $\Theta(\Delta)$ branches; between phases the nodes not yet visited are reassigned among the branches along walks on an expander, which allows $\Theta(1+\log(n/\Delta))$ phases. One construction thus combines the $\Omega(\Delta n)$ of chained rotating stars (Erlebach et al. 2021) and the $\Omega(n \log n)$ known for planar underlying graphs of maximum degree 3 whose snapshots are paths (Adamson, Gusev, Malyshev and Zamaraev, strengthening degree 4 of Erlebach, Hoffmann and Kammer), both as reported by (Carnevale 2026) and (Bastide et al. 2025); for constant Δ it adds nothing to $\Omega(n \log n)$. It answers Problem 5.1 of (Bastide et al. 2025), as they state. Preprint.
- Erlebach et al. (2021) [current best]: time $O(n^2)$. The n^2 upper bound is attributed by the authors to Michail and Spirakis. They also give $O(n^{1.5}\mathrm{tw}^{1.5} \log n)$ for treewidth tw, $O(n^{1.8} \log n)$ for planar and $2n-2$ for cycles, and show inapproximability within $O(n^{1-\varepsilon})$.
- Kammer et al. (2026) [preprint] [also]: time $O(\sigma n \log^3 n)$. Parameterized by σ, the number of classes of the 2-dimensional Weisfeiler–Leman colouring of the temporal graph (their q), $1 \le \sigma \le n$; vertex-transitive and distance-regular underlying graphs have $\sigma = 1$. The walk is computed deterministically in polynomial time, for a long enough lifetime (Theorem 1.1). Some instances need $\Omega(n \log n)$ steps for constant σ and $\Omega(\sigma n)$ for every σ (Theorem 1.3), so the bound is tight up to polylogarithmic factors. It improves an earlier bound in the number of automorphism orbits (at least σ), which needed superpolynomial computation, as reported by (Kammer et al. 2026).
- Lahtin and Zamaraev (2027) [also]: time $O(\mu n \log \mu)$. *Assumes:* at most $\mu \ge 2$ edges of the underlying graph missing in each snapshot. Corollary 1.3 (arXiv v1; their k is μ here), for always-connected temporal graphs with enough snapshots, from any start node. It suffices that at most μ edges of one fixed spanning tree of the underlying graph are missing in each snapshot (Theorem 1.1). The schedule is computed by a Las Vegas algorithm in expected polynomial time, and deterministically in polynomial time when μ is constant (Theorem 1.2). It improves the $O(\mu n \log n)$ of Erlebach and Spooner and is within a factor μ of their $\Omega(n \log \mu)$ lower bound, both as reported by (Lahtin and Zamaraev 2027); for constant μ exploration takes $\Theta(n)$ steps.
- Adamson and Spirakis (2026) [also]: time $O(n^{4/3}\,\mathrm{tw}^{9/2} \log^{5/2} n)$. *Assumes:* underlying graph of treewidth tw. Theorem 2 of the full version (arXiv v2). The underlying graph is divided into parts with few boundary nodes; each part is explored by a team of agents (Theorem 1, on always S-connected temporal graphs, in which every snapshot joins every node to some node of a fixed set S), and the team schedule is turned into one for a single agent. For small tw it is below the $O(n^{3/2}\sqrt{\mathrm{tw} \log n})$ of (Bastide et al. 2025). (Carnevale 2026) cites $\mathrm{tw}^{11/2}$ for the brief announcement, which was not read.
- Bastide et al. (2025) [preprint] [also]: time $O(n^{3/2}\sqrt{\overline{\Delta} \log n})$. Theorem 1.3 (arXiv v2), from every start node, for every always-connected temporal graph with n^2 snapshots, where $\overline{\Delta}$ (their D) is the average temporal maximum degree: the average over the nodes v of the largest degree of v in any snapshot. If every snapshot has maximum degree at most Δ then $\overline{\Delta} \le \Delta$, which gives $O(n^{3/2}\sqrt{\Delta \log n})$ (Corollary 1.4), and $O(n^{3/2}\sqrt{\log n})$ for constant

Claude (Anthropic) was used in cataloguing the problems.

Δ against the $O(n^{7/4})$ of Erlebach, Kammer, Luo, Sajenko and Spooner. An underlying graph with m edges has $\overline{\Delta} \le 2m/n$ (Corollary 1.5, stated for average degree), which gives $O(n\sqrt{m \log n})$: $O(n^{3/2}\sqrt{\log n})$ for planar underlying graphs and $O(n^{3/2}\sqrt{\mathrm{tw} \log n})$ for treewidth tw, against $O(n^{1.8} \log n)$ and $O(n^{1.5}\mathrm{tw}^{1.5} \log n)$ (Erlebach et al. 2021) and $O(n^{7/4} \log n)$ and $O(\mathrm{tw}\, n^{3/2} \log n)$ by Adamson, Gusev, Malyshev and Zamaraev, the last three as reported by (Bastide et al. 2025). The bound is below n^2 when $\overline{\Delta} = o(n/\log n)$. The key lemma: any set X of nodes contains two nodes joined by a temporal walk within $2\overline{\Delta}n/|X| + 1$ steps (Lemma 1.8); in each block of n steps a walk then visits about $\sqrt{|X|/(\overline{\Delta} \log |X|)}$ of the $|X|$ nodes not yet visited (Lemma 3.2). The authors state, without a detailed proof, that the walk can be computed in polynomial time. Preprint.

Open questions.

Q1. Tight bounds when the underlying graph is a grid, has bounded degree or is planar (Erlebach et al. 2021). For bounded degree and for planar underlying graphs the best bounds known are $\Omega(n \log n)$ and $O(n^{3/2}\sqrt{\log n})$ (Bastide et al. 2025); the upper bound holds for grids too.

Q2. For underlying maximum degree Δ, close the gap between $\Omega(\Delta n(1 + \log(n/\Delta)))$ (Carnevale 2026) and $O(\min\{n^2, n^{3/2}\sqrt{\Delta \log n}\})$ (Bastide et al. 2025); for constant Δ the bounds are $\Omega(n \log n)$ and $O(n^{3/2}\sqrt{\log n})$. The upper bound needs only that every snapshot has maximum degree at most Δ.

Q3. When at most μ edges are missing in each snapshot, remove the factor μ between $O(\mu n \log \mu)$ (Lahtin and Zamaraev 2027) and $\Omega(n \log \mu)$ (as reported by Lahtin and Zamaraev 2027).

Notes. This is a centralised planning problem rather than a distributed one, but it gives the baseline for what any agent could do in a temporal graph.

The worst case $\Theta(n^2)$ needs nodes of degree $\Omega(n)$ in the snapshots and a dense underlying graph. Bastide, Groenland, Michel, Rambaud and Smolarova (Bastide et al. 2025) bound exploration by the average temporal maximum degree $\overline{\Delta}$, which is at most the maximum degree of every snapshot and at most the average degree of the underlying graph, and so treat bounded snapshot degree, planar, bounded-treewidth and sparse underlying graphs with one argument. Allowed two moves per step (a model not catalogued here), the agent explores every always-connected temporal graph within $O(n^{3/2}\sqrt{\log n})$ steps, as each snapshot squared has a spanning tree of maximum degree 3 (Bastide et al. 2025). For snapshots drawn independently from a distribution over connected graphs, the same paper gives expected exploration time $O(\overline{\Delta} n \log^3 n)$ (Theorem 1.6), with $\overline{\Delta}$ taken over all graphs of positive probability; see (Baguley et al. 2026) under further reading.

16.1.18 T-interval-connected graphs, labelled nodes

Adds + labelled nodes, + with stop

VERIFIED

On a T-interval-connected graph whose nodes carry identifiers, a deterministic agent can explore every graph with m edges only if $T = \Omega(m)$; with a slightly larger window it explores and stops within $\Theta(n^3)$ moves in the worst case, and within $\Theta(n^2)$ if it also sees its neighbours' identifiers.

Assumed: graph dynamics: dynamic graph, connected in every round (1-interval connected); edge direction: undirected; node identifiers: labelled nodes (an agent can recognise a node it has visited); termination requirement: explore, then stop anywhere. **Not assumed:** knowledge: nothing known about the graph; randomness: deterministic; markers: no markers; preprocessing: ports and nodes as given by an adversary; topology: arbitrary connected graph.

Statement for this setting. The graph is a dynamic graph $(G_t)_{t \ge 0}$ on a fixed node set; its underlying graph (all edges that ever appear) is simple and connected, with n nodes and $m \ge n$ edges. It is T-interval connected: for every t, the edges present in all of the rounds $t, \dots, t + T - 1$ form a connected spanning subgraph. Every node carries a distinct identifier. In round t the agent at node v sees the identifier of v, $\deg(v)$ in the underlying graph, which ports of v have their edge present in G_t, and the port by which it entered; it then crosses one present edge or stops (staying put only uses up the window). The agent

is deterministic and knows neither T, n nor m; it must visit every node and stop. Two quantities are asked for: the smallest window T, as a function of n and m, for which some algorithm explores every T-interval-connected graph, and the number of moves it then needs. With **1-hop view** (the model KT_1 of Sudo et al. 2026c; the setting above is their KT_0) the agent also sees the identifiers of its neighbours in G_t and the ports that lead to them.

Measure	Lower bound	Best upper bound	Status
Time	$\Omega((m-n+1)\,n)$ (Sudo et al. 2026c) [preprint]; *impossible* (Sudo et al. 2026c) [preprint]; $\Omega(m)$ (Sudo et al. 2026c) [preprint]	$O((m-n+1)\,n + n\log^2 n)$ (Sudo et al. 2026c) [preprint]	• near-optimal
	$O((m-n+1)\,n + n\log^2 n)$ against $\Omega((m-n+1)\,n)$: tight when $m-n+1=\Omega(\log^2 n)$, and $\Theta(n^3)$ as a function of n alone. With 1-hop view, $O(\zeta m + n\log^2 n)$ against $\Omega(m)$: at most a $\log^2 n$ factor, and $\Theta(n^2)$ in n alone. The window needed is between $\Omega(m)$ and $O(\zeta m + n\log^2 n)$ for both views.		
Memory	none known	—	• unknown
	Memory is not bounded: the agent keeps a map.		

GRAPHS

Results.

- Sudo et al. (2026c) [preprint] [lower bound]: time $\Omega((m-n+1)\,n)$. *Bottleneck:* adversarial edge changes. *Holds when:* deterministic agent that sees only its own node (KT_0); n large enough and $n \le m \le \binom{n}{2}$; holds for every window T: a spanning path is present in every round (∞-interval connected); the underlying graph is a path $v_1, \dots, v_n$, plus $\min(m-n+1, |V_L|\,|V_R|)$ trap edges between V_L (the first third of the path) and V_R (its last third without v_n), plus edges that are never present while the agent is at one of their ends; the agent starts at v_1; the adversary removes the trap edges at every node of V_L the agent visits, and, since the agent learns where a port leads only by crossing its edge, sends every unused port of a node of V_R along an unused trap edge. Theorem 4: each trap edge costs a walk of $\Omega(n)$ moves back to V_R before v_n is reached.

- Sudo et al. (2026c) [preprint] [lower bound]: time *impossible*. *Bottleneck:* adversarial edge changes. *Holds when:* window $T \le \kappa m$ for a small enough constant $\kappa > 0$ (Theorem 3): for every n large enough and every m with $n \le m \le \binom{n}{2}$, some T-interval-connected graph with n nodes and m edges has a node that the agent never visits; deterministic agent; holds with 1-hop view and even for perpetual exploration; for $m \ge 2n$: two copies of K_g minus one edge, $g = \lfloor\sqrt{m/8}\rfloor$, joined through three nodes and one edge, with a path attached; an agent that enters a copy can be kept inside it for $\Omega(g^2)$ rounds without disconnecting any window (Lemma 9, which holds even if the agent knows the underlying graph with its ports); for $n \le m < 2n$: a ring whose two edges at one node are removed alternately, which keeps the graph $(n-3)$-interval connected. With the algorithms below, the smallest window is $\Theta(m)$ when $m = n^{1+\Omega(1)}$, and $\Theta(n^2)$ as a function of n alone (Corollary 2), with or without 1-hop view.

- Sudo et al. (2026c) [preprint] [lower bound]: time $\Omega(m)$. *Bottleneck:* adversarial edge changes. *Holds when:* with 1-hop view (KT_1); deterministic agent; holds for every window T (∞-interval connected); a path of $n-g$ nodes attached to K_g minus one edge, $g = \lfloor\sqrt{m}\rfloor$; the agent starts inside the K_g part and is kept there for $\Omega(g^2)$ rounds (Lemma 9). Theorem 5. It bounds the time of the 1-hop-view algorithm below; the first bound is the one for this setting.

- Sudo et al. (2026c) [preprint] [current best]: time $O((m-n+1)\,n + n\log^2 n)$. *Assumes:* window $T \ge \kappa(\zeta m + n\log^2 n)$ for a large enough constant κ, where $\zeta = \zeta(n,m) = \frac{\ln n}{1+\ln m - \ln n}$; T, n and m unknown; unbounded memory: the agent keeps a map of the nodes and edges it has seen. Introductory and current best (Theorem 1, algorithm $\mathrm{GreedyExp}_0$; ζ is their $\epsilon(n,m)$). The agent walks, along edges it has never seen missing, to the nearest visited node that has an unused port whose edge was present at every visit, and crosses that edge. Every $\kappa(\zeta m' + n'\log^2 n')$ moves, with n' and m' computed from the nodes and ports seen so far, it forgets which edges it saw missing, so that each such period has a spanning tree present throughout. A period that does not end the exploration contains at least $\kappa(\zeta m' + n'\log^2 n')/(2n')$ crossings of unused edges into visited nodes (Lemma 8), and there are at most $m-n+1$ such crossings (Corollary 4).

- Sudo et al. (2026c) [preprint] [also]: time $O(\zeta m + n\log^2 n)$. *Assumes:* 1-hop view (KT_1); window $T \ge \kappa(\zeta m + n\log^2 n)$ for a large enough constant κ; T, n and m unknown; unbounded memory. Theorem 2, algorithm $\mathrm{GreedyExp}_1$: walk to the nearest node seen but not visited, in the map of edges never seen missing. Exploration ends within

Claude (Anthropic) was used in cataloguing the problems.

the window itself, so one spanning tree is present throughout. The distance to the nearest unvisited node grows by at most $2n \log n$ in total when new nodes are visited (Lemma 4) and by $O(\zeta m + n \log^2 n)$ when edges are seen missing, through the bound $|E| \le \frac{1}{2}|V|^{1+1/j} + \frac{1}{2}|V|$ for graphs of girth at least $2j+1$ (Lemmas 5 and 6).

Open questions.

Q1. Close the gap on the window, $\Omega(m)$ against $O(\zeta m + n \log^2 n)$; it is at most a $\log^2 n$ factor and remains when $m = n^{1+o(1)}$ (Sudo et al. 2026c).

Q2. How much smaller can the window be, and how much faster the exploration, for a randomized agent (Sudo et al. 2026c)?

Q3. Close the polylogarithmic gap on the time with 1-hop view, $\Omega(m)$ against $O(\zeta m + n \log^2 n)$, and determine exactly what the 1-hop view buys (Sudo et al. 2026c).

Notes. Sudo, Kitamura, Shibata, Nakamura, Tixeuil, Masuzawa and Wada study a single deterministic agent on T-interval-connected graphs of arbitrary topology (Sudo et al. 2026c). They assume $m \ge n$, since with $m = n - 1$ every edge is always present and the graph is static. Their lower bounds on time hold even if one spanning tree is present forever: the cost comes from edges that are absent while the agent stands at one of their ends, not from disconnection. On rings the window has to be large as well (the case $m < 2n$ of Theorem 3); this is why the online exploration of T-interval-connected rings (Ilcinkas and Wade 2018) also assumes that every edge appears at least once in every δ consecutive rounds, as Sudo et al. recall. Both algorithms rely on the identifiers: the agent recognises the nodes it has visited and keeps a map of the edges it has seen. Checked against the full text (arXiv v1, 2026), a preprint.

16.2 Treasure hunt

One agent must find a static treasure at an unknown node; the cost should depend on how far the treasure is, not on the size of the graph.

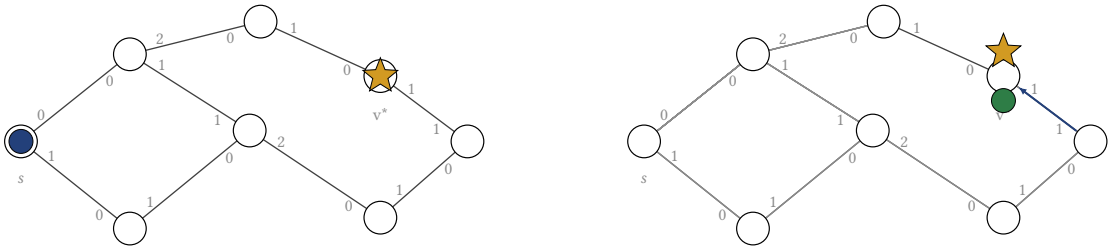
First posed in 2007 (Ta-Shma and Zwick 2014). Ta-Shma and Zwick introduced the deterministic treasure hunt problem at SODA 2007, as a variant of rendezvous in which one agent never moves.

Problem statement and measures

A single deterministic agent starts at a node s of an unknown connected graph whose edges carry **ports** (at each node, local numbers of the edges at that node); a **treasure** sits at an unknown node v^* at distance d from s. The agent must reach v^* (it recognises the treasure when it arrives). **Cost** is the number of moves (edge traversals) until the treasure is found, measured against d, against $e(d)$ (the number of edges with an endpoint at distance less than d from s) or against a path weight defined per paper. **Help** counts any external help the agent receives: bits of **advice** from an oracle that knows the instance, or **pebbles** (objects left on nodes) placed by it. Unless a variant says otherwise the agent knows nothing about the graph, d or v^*.

Measures: **Cost**, edge traversals until the treasure is reached; **Help**, bits of advice, or pebbles placed in advance by an oracle.

Claude (Anthropic) was used in cataloguing the problems.



(a) Start: agent at s, treasure at an unknown node v*

(b) Final: treasure found during the phase N = 8, after 13 moves

Figure 16.2: Treasure hunt by exploration sequences of doubling size (Ta-Shma and Zwick 2014): in the phases $N = 1, 2, 4, 8, \ldots$ the agent follows the sequence for size N from where it stands (entering by port p with term x, it leaves by port $(p + x)$ mod deg; ports are numbered from 0) and stops on the treasure v^*. The short sequences drawn here are illustrative, not actual universal sequences, which are much longer. One frame per move.

A first look

A hiker is lost in a park whose trails form a network, and a rescuer walks from the trailhead looking for her, without a map and without knowing which direction to take. The network is a **graph**: trail junctions are **nodes** and the trails between them are **edges**; the rescuer is a single **agent** that starts at a node s and walks along edges, and one crossing of one edge is a **move**. The graph is unknown and connected, and at each node the edges at that node carry local numbers, its **ports**. The hiker is the **treasure**, at an unknown node v^* at **distance** d from s (the distance is the fewest edges on a path between two nodes). The agent recognises the treasure only when it stands on it. In the basic version the graph is finite, with n nodes, and the agent knows nothing about it; it must reach v^*, and the **cost** is the number of moves it makes.

1. Algorithm: Universal exploration sequences of doubling size. (Ta-Shma and Zwick 2014) (Anonymous finite graphs, p. 112) An **exploration sequence** is a list of numbers $x_1, x_2, \ldots$; following it means: at the i-th move, if you entered the current node (of **degree** deg, the number of edges at it) by port p, leave by port $(p + x_i)$ mod deg. A sequence is **universal for size** N if following it visits every node of every connected graph with at most N nodes, from any start. Such sequences exist and can be computed (Reingold 2008).

1. For $N = 1, 2, 4, 8, \ldots$ (one **phase** per value of N):
2. follow the universal sequence for size N from wherever you are;
3. stop as soon as you stand on the treasure.

Why it works and what it costs. Once $N \geq n$, the sequence for size N visits every node, including v^*, so the search ends during that phase (Ta-Shma and Zwick 2014). Each sequence has length polynomial in N, the lengths grow geometrically, and the loop never goes past $N < 2n$, so the total cost is polynomial in n. It ignores d: a treasure next door may cost as much as one far away. With node labels, a treasure at distance d is found at cost $O(e(d) \log d)$ (Bouchard et al. 2023a), where $e(d)$ is the number of edges with an endpoint at distance less than d from s.

Why it matters. Doubling a guess you do not know (here n) is the standard way to remove knowledge from an algorithm; it reappears in rendezvous and gathering.

2. Lower bound: The treasure is behind the last port. (Bouchard et al. 2023a) (Labelled nodes, locally

finite graphs, p. 115) Let s have many neighbours, say 20, and put the treasure on one of them, so $d = 1$. The agent cannot see what is at the other end of an edge without crossing it. An adversary watches the agent and puts the treasure behind whichever of the 20 ports the agent tries last; since every node looks the same from s, the agent cannot avoid this.

Why it holds. The agent walks out and back along at least 19 edges before it tries the last one, so it pays about twice the degree of s. Here $e(1)$, the number of edges with an endpoint at distance less than 1 from s, is exactly the degree of s, and the same idea applied layer by layer gives the general bound $\Omega(e(d))$ of (Bouchard et al. 2023a), which holds even when nodes carry labels. Their algorithm costs $O(e(d) \log d)$, so the gap is a factor $\log d$. Trivially the agent also needs at least d moves.

Why it matters. It shows that the right measure of cost is the number of edges near the start, not the distance, which is why later work states bounds in terms of $e(d)$ or path weights.

What the community started looking at

Ta-Shma and Zwick introduced deterministic treasure hunt as the variant of rendezvous in which one agent never moves (Ta-Shma and Zwick 2014). The question from the start was **cost as a function of how far the treasure is**, not of the size of the graph, which led to strongly universal exploration sequences. Later work asked how much **external help** (advice bits, pebbles) reduces the cost (Miller and Pelc 2015b; Gorain et al. 2022), what is possible in graphs with infinite degrees (Pattanayak and Pelc 2024b), and whether the treasure can be found at all in a ring whose nodes an adversary reorders at every step (Ayoubi et al. 2026).

Further reading

Papers on this problem that are not yet placed in a setting.

- Gaur et al. (2025). An oblivious agent (no memory between moves) guided by pebbles that an oracle places on a shortest path to the treasure at distance d (their D): pebbles of one colour never suffice, at least $\Delta - 1$ colours are needed for a deterministic agent (Theorems 1 and 2), and $\Delta - 1$ colours suffice. Quantum pebbles, one per node of the path, each emitting qubits that encode the port to take, let the agent reach the treasure in d moves with probability tending to 1, using $O(\Delta^3(\log d + \log \Delta))$ measurements per node (Theorem 4). One wrong colour can trap an oblivious agent forever, and under such persistent errors every randomized strategy succeeds only with probability exponentially small in d (Theorem 5, Proposition 1); the quantum protocol tolerates independent errors on each emitted qubit below an explicit threshold (Theorem 6). Preprint; read (arXiv v2).
- Bhattacharya et al. (2022). Fewer pebbles than the distance: with $p < d$ pebbles placed by an oracle (their k and D), the cost becomes exponential in d/p, e.g. $O(p\Delta^{d/(p+1)})$ for $d/2 \le p < d$, against $\Omega((p/e)^{p/(p+1)}(\Delta-1)^{d/(p+1)})$ on trees for every $p < d$ (Theorem 4); also bipartite graphs and $p = cd$. Read (arXiv v1).
- Bouchard et al. (2021). Trees: the agent gets a map of the tree rooted at its start, with or without port numbers, and the distance to the treasure or not. The penalty of a less precise kind of knowledge is very large for a map without ports when the distance is known, small when it is not, and medium for not knowing the distance when a map is given. Read (arXiv v1, 2025).
- Bouchard et al. (2020). In the plane: an agent with a compass and a measure of length gets, after each move, an angle smaller than 2π that contains the treasure. Angles at most π give cost $O(D)$, which is optimal; angles at most a constant $\beta < 2\pi$ give $O(D^{2-\varepsilon})$; arbitrary angles cannot beat $\Theta(D^2)$ (their D bounds the initial distance). Read (arXiv v1).
- Pelc and Yadav (2019a). In the plane, with advice: the treasure is within distance D and is found from distance r. For every size z of advice, cost $O(\mathrm{OPT}(z, D, r))$ when $r \le 1$ or $r \ge 0.9D$, and $O(\mathrm{OPT}(z, D, r)^{1+\alpha})$ for every constant $\alpha > 0$ in between. Read (arXiv v1).

Claude (Anthropic) was used in cataloguing the problems.

- Pelc and Yadav (2021). Polygonal terrains with obstacles: the advice needed to find the treasure at cost $O(L)$, where L is the length of a shortest path to it, is determined exactly for convex terrains with convex fat obstacles, and is exponentially larger for arbitrary terrains. Read (arXiv v2).
- Bhattacharya et al. (2023). In the plane, with pebbles: an agent with a compass whose speed the adversary controls can turn only at pebbles. One pebble does not allow a finite cost, two give $O(D)$, and $k > 8$ pebbles give $O(k^2) + D(\sin\theta_0 + \cos\theta_0)$ for an angle θ_0 that shrinks as k grows, so the cost approaches D for large D (Theorem 3). Read (arXiv v1).
- Dobrev et al. (2017). Treasure hunt in another sense, from Fraigniaud, Korman and Rodeh: k deterministic agents open boxes of an infinite ordered list, and an agent can only tell whether a box was opened before. With failing agents they are 1-competitive against an algorithm that knows which agents fail; with f boxes that are black holes, between $2f + 1$ and $4f + 1$ agents are needed. Read (LIPIcs version).
- Korman and Rodeh (2016). The same box search with a treasure placed uniformly at random among many boxes: k searchers that do not coordinate achieve speed-up $k(k + 1)/(3k - 1)$, about three times worse than full coordination as k grows. Read (arXiv v2).

Relations to other problems

- **Subproblem of Graph exploration by a single agent.** An agent that explores the graph finds any treasure, so every exploration algorithm solves treasure hunt; the question is how much cheaper it can be when the treasure is close, measured against its distance d (or a weight of the path to it) rather than the size of the graph.

Assumptions specific to this problem

Graph size

Ta-Shma and Zwick posed deterministic treasure hunt in a finite connected graph, as the variant of rendezvous in which one agent never moves (Ta-Shma and Zwick 2014); Pelc notes that rendezvous with arbitrary delay contains this case, since the adversary may wake one agent late enough (Pelc 2019). Infinite graphs make sense once cost is measured locally: Bouchard, Dieudonné, Labourel and Pelc consider connected graphs, finite or countably infinite, with finite degrees, and find a treasure at distance d at cost $O(e(d)\log d)$, refuting a conjecture of Awerbuch, Betke, Rivest and Singh (Bouchard et al. 2023a). Pattanayak and Pelc observed that "all known treasure hunt and rendezvous algorithms rely on the assumption that degrees of all nodes are finite, even when the graph itself may be infinite", removed the assumption, and noted that countable degrees are the limit: with uncountable degrees neither task can be done in finite worst-case time (Pattanayak and Pelc 2024b). The three levels differ in the natural cost measure: a function of n in finite graphs, $e(d)$ in locally finite ones, and a weight $w(s, v^*)$ of the cheapest path type when $e(d)$ may be infinite (Pattanayak and Pelc 2024b).

Level	Value	Formal definition
0	**infinite degrees allowed**	G is finite or countably infinite; a node may have countably infinite degree (ports $1, 2, 3, \ldots$).
1	**locally finite**	G is finite or countably infinite, and every node has finite degree.
2	**finite graph**	G has finitely many nodes.

External help

External help measures how much information about the instance buys a fast treasure hunt. Miller and Pelc framed it in the paradigm of algorithms with advice: an oracle that knows the network, the start and the treasure gives the agent a binary string, and the question is the smallest advice that achieves a given

Claude (Anthropic) was used in cataloguing the problems.

cost, against cost $\Theta(m)$ in the worst case without information (Miller and Pelc 2015b); among the earlier advice papers they cite tree exploration with advice by Fraigniaud, Ilcinkas and Pelc (Fraigniaud et al. 2008). Their nodes carry labels that the agent reads on arrival (Miller and Pelc 2015b). Gorain, Mondal, Nayak and Pandit replaced the string by pebbles that the oracle places on nodes of an anonymous graph, at most one per node, seen only on arrival, arguing that when many agents search independently it is cheaper to put information on the nodes once than to give advice to each agent (Gorain et al. 2022). They show $O(d \log \Delta)$ pebbles give time $O(d \log \Delta + \log^3 \Delta)$ and that no number of pebbles beats $\Omega(d \log \Delta)$ (Gorain et al. 2022). The two forms of help sit on the same level because neither simulates the other: advice is global and read at once, pebbles are local and read along the walk.

Level	Value	Formal definition
0	**no help**	No information beyond the agent's own observations.
1	**advice**	An oracle that knows the whole instance gives the agent a binary string before it starts; its length (in bits) is the measure.
1	**oracle pebbles**	An oracle that knows the instance places pebbles on nodes (at most one per node) before the agent starts; their number is the measure.

Case tree

The root assumes nothing; going down, each edge adds the assumptions written in grey (+) or leaves an axis unassumed. Bold entries are catalogued settings with their status per measure (Cost, Help) and page; italic entries are branch points, named after the axis on which their branches differ. Axis order: Graph dynamics → Edge direction → Graph size → Node identifiers → External help → Topology.

No assumptions

+ static graph, + undirected
Anonymous graphs with possibly infinite degrees ● ● p. 112

+ finite graph
Anonymous finite graphs STANDARD SETTING ● ● p. 112

+ advice
Treasure hunt with advice ● ● p. 114

+ oracle pebbles
Treasure hunt guided by pebbles ● ● p. 114

+ locally finite, + labelled nodes
Labelled nodes, locally finite graphs ● ● p. 115

+ connected every round, + undirected, + finite graph
Dynamic graphs, connected at every step ● ● p. 115

What is known

The best upper bound against the best lower bound in each catalogued setting, in tree order. The settings follow in detail.

Setting	Cost	Help	p.
Anonymous graphs with possibly infinite degrees	● tight $O(w(s, v^*))$ vs $\Omega(w(s, v^*))$	● tight — vs —	112

Claude (Anthropic) was used in cataloguing the problems.

Setting	Cost	Help	p.
Anonymous finite graphs STANDARD SETTING	● unknown $O(n^\theta \log n (\log\log n)^{1+\varepsilon})$ vs —	● unknown — vs —	112
Treasure hunt with advice	● tight C vs —	● near-optimal $O(d\log(d\cdot m/C)+\log\log m)$ vs $\Omega(d\log(m/C))$	114
Treasure hunt guided by pebbles	● near-optimal $O(d\log\Delta+\log^3\Delta)$ vs $\Omega(d\log\Delta)$	● unknown $O(d\log\Delta)$ pebbles vs —	114
Labelled nodes, locally finite graphs	● near-optimal $O(e(d)\log d)$ vs $\Omega(e(d))$	● tight — vs —	115
Dynamic graphs, connected at every step	● impossible	● unknown — vs —	115

16.2.1 Anonymous graphs with possibly infinite degrees

Adds + static graph, + undirected VERIFIED

A universal algorithm whose cost is optimal among universal algorithms, measured by a weight $w(s,v^*)$ of the cheapest path type to the treasure.

Assumed: graph dynamics: static graph; edge direction: undirected. **Not assumed:** graph size: finite or countably infinite, degrees may be infinite; node identifiers: anonymous nodes; external help: no help; topology: arbitrary connected graph.

Statement for this setting. For a path with largest port q and length ρ, its type is (q,ρ) with value $\rho\, 2^\rho q^\rho$; $w(s,v^*)$ is the value of the cheapest type of a path from s to v^*.

Measure	Lower bound	Best upper bound	Status
Cost	$\Omega(w(s,v^*))$ (Pattanayak and Pelc 2024b) Optimal order of magnitude among universal algorithms.	$O(w(s,v^*))$ (Pattanayak and Pelc 2024b)	● tight
Help	none known No help is used.	—	● tight

Results.

- Pattanayak and Pelc (2024b) [lower bound]: cost $\Omega(w(s,v^*))$. *Bottleneck:* port addressing. *Holds when:* among universal algorithms (no knowledge of the graph); graphs whose nodes may have countably infinite degree.
- Pattanayak and Pelc (2024b) [current best]: cost $O(w(s,v^*))$. Introductory and current best (Theorem 2.1).

16.2.2 Anonymous finite graphs

Adds + finite graph **to** Anonymous graphs with possibly infinite degrees (p. 112) STANDARD SETTING VERIFIED

If the treasure is present from the start, concatenating universal traversal sequences finds it in polynomial time; if it may appear later, explicit solutions need backtracking and take the length of a universal exploration sequence, polynomial with a huge exponent, times a polylogarithmic factor.

Assumed: graph dynamics: static graph; edge direction: undirected; graph size: finite. **Not assumed:** node identifiers: anonymous nodes; external help: no help; topology: arbitrary connected graph.

Claude (Anthropic) was used in cataloguing the problems.

Measure	Lower bound	Best upper bound	Status
Cost	none known	$O(n^\theta \log n (\log\log n)^{1+\varepsilon})$ (Xin 2026) [preprint]	• unknown
	No lower bound in terms of n is recorded. The best upper bound exceeds the length $O(n^\theta)$ of one ordinary universal exploration sequence by a factor $\log n (\log\log n)^{1+\varepsilon}$ (preprint) (Xin 2026).		
Help	none known Memory is not studied in these papers.	—	• unknown

Results.

- Ta-Shma and Zwick (2014) [introductory]: cost $O(n^{2\theta})$. *Assumes:* the treasure may appear after the agent starts; backtracking: the agent sees its entry port; Δ-regular graphs, $\Delta \geq 3$ fixed. Theorem 3.2: an infinite strongly universal exploration sequence with cover time $O(n^{2\theta})$, θ the exponent of Reingold's universal exploration sequences; explicit when those are. Proved for Δ-regular graphs; arbitrary degrees are reduced to them by adding self-loops up to degree n (Section 1.5), which needs sequences for n-regular graphs.
- Xin (2007) [also]: cost $O(n^{\theta(1+1/a)})$. *Assumes:* the treasure may appear after the agent starts; backtracking: the agent sees its entry port; Δ-regular graphs, $\Delta \geq 3$ fixed. For any fixed integer $a \gg 1$; with backtracking (the agent sees the entry port). Proved for Δ-regular graphs (Theorem 4); arbitrary degrees by a remark. The best peer-reviewed bound; recovered as Corollary 7.1 of (Xin 2026).
- Xin (2026) [preprint] [current best]: cost $O(n^\theta \log n (\log\log n)^{1+\varepsilon})$. *Assumes:* the treasure may appear after the agent starts; backtracking: the agent sees its entry port; Δ-regular graphs, $\Delta \geq 3$ fixed. Preprint. Theorem 1.1 and Corollary 8.1: one explicit infinite strongly universal exploration sequence, for every fixed $\varepsilon > 0$; the agent knows neither n nor when the treasure appears. Theorem 6.1 gives the same with any fixed number of iterated logarithms, and the simpler periods of Section 6.1 give $O(n^\theta \log^2 n)$. Arbitrary degrees (Proposition 8.2): the agent simulates the sequence on the cubic multigraph with $2m$ nodes that replaces each node by a cycle of $\deg(v)$ nodes, one move or stay per step, and finds the treasure within $O(m^{\theta_3} \log m (\log\log m)^{1+\varepsilon})$ steps, θ_3 the exponent of universal exploration sequences for cubic multigraphs; $O(n^{2\theta_3} \log n (\log\log n)^{1+\varepsilon})$ on simple graphs.

Open questions.

Q1. Treasure hunt when the treasure may appear after the agent starts and the agent does **not** see its entry port: "We do not know if an efficient solution is possible when backtracking is not allowed" (Ta-Shma and Zwick 2014).

Q2. Do strongly universal traversal sequences exist? They exist if and only if treasure hunt has a solution independent of the start time (Ta-Shma and Zwick 2014).

Q3. Is there a strongly universal exploration sequence with cover time $O(n^\theta)$, or $O(n^\theta \log n)$, for fixed-degree graphs? (Xin 2026) (preprint) reaches $O(n^\theta \log n (\log\log n)^{1+\varepsilon})$ and shows that its recurrence cannot keep linear length with periods $\Theta(2^j j)$ (Corollary 10.2), which is "not a lower bound for arbitrary SUESs".

Notes. For a treasure present when the agent starts, Ta-Shma and Zwick note that following $U_1 U_2 U_4 \ldots$, universal traversal sequences of doubling length, solves the problem in polynomial time (Ta-Shma and Zwick 2014). The bounds above are for the harder case. All three are proved for Δ-regular graphs with fixed $\Delta \geq 3$, where θ is defined; Ta-Shma and Zwick reach arbitrary degrees by self-loops up to degree n (Ta-Shma and Zwick 2014), and Xin by a cubic simulation whose bound is in m (Xin 2026), so the three exponents compare directly only on regular graphs. Ta-Shma and Zwick build the strongly universal exploration sequence by inserting, after the i-th symbol of an ordinary one, a forward and a backward copy of a much shorter prefix whose length is about the square root of the 2-adic part of i; walking a sequence and then its reverse returns the agent to where it was, so every window of length $O(n^{2\theta})$ contains a full universal sequence for n nodes (Ta-Shma and Zwick 2014). Xin first shrank the inserted prefixes to a $1/a$ power (Xin 2007), and later chose them by periods $q_j \approx 2^j j (\log j)^{1+\varepsilon}$ with $\sum_j 2^j/q_j$

Claude (Anthropic) was used in cataloguing the problems.

finite, which keeps the whole sequence within a constant factor of the ordinary one and gives windows of length $O(n^{\theta} \log n (\log \log n)^{1+\varepsilon})$ (preprint) (Xin 2026). In that preprint the agent computes the t-th instruction with $O(\log(t+2))$ bits of memory, so its memory grows with the time since it started, not with the graph (Xin 2026). Checked against the full texts (Xin 2026: arXiv v1).

16.2.3 Treasure hunt with advice

Adds + advice **to** Anonymous finite graphs (p. 112) VERIFIED

How many bits of advice buy a given cost C: about $d \log(m/C)$.

Assumed: graph dynamics: static graph; edge direction: undirected; graph size: finite; external help: advice: bits from an oracle that knows the instance. **Not assumed:** node identifiers: anonymous nodes; topology: arbitrary connected graph.

Measure	Lower bound	Best upper bound	Status
Cost	none known Cost C is the measure to minimise.	C (Miller and Pelc 2015b)	• tight
Help	$\Omega(d \log(m/C))$ (Miller and Pelc 2015b) A $\log d$ inside the first term on arbitrary graphs.	$O(d \log(d \cdot m/C) + \log \log m)$ (Miller and Pelc 2015b)	• near-optimal

Results.

- Miller and Pelc (2015b) [lower bound]: help $\Omega(d \log(m/C))$. *Bottleneck:* information. *Holds when:* to achieve cost C; d = distance to the treasure, m = number of edges.
- Miller and Pelc (2015b) [current best]: cost C, help $O(d \log(d \cdot m/C) + \log \log m)$. Introductory and current best; on trees $O(d \log(m/C) + \log \log m)$, nearly tight.

Notes. The same paper gives the analogous trade-off for rendezvous.

16.2.4 Treasure hunt guided by pebbles

Adds + oracle pebbles **to** Anonymous finite graphs (p. 112) VERIFIED

An oracle places pebbles (at most one per node); $O(d \log \Delta)$ pebbles let the agent reach the treasure in near-optimal time.

Assumed: graph dynamics: static graph; edge direction: undirected; graph size: finite; external help: pebbles placed by an oracle that knows the instance. **Not assumed:** node identifiers: anonymous nodes; topology: arbitrary connected graph.

Measure	Lower bound	Best upper bound	Status
Cost	$\Omega(d \log \Delta)$ (Gorain et al. 2022) An additive $\log^3 \Delta$.	$O(d \log \Delta + \log^3 \Delta)$ (Gorain et al. 2022)	• near-optimal
Help	none known No lower bound on the number of pebbles recorded.	$O(d \log \Delta)$ pebbles (Gorain et al. 2022)	• unknown

Results.

- Gorain et al. (2022) [lower bound]: cost $\Omega(d \log \Delta)$. *Bottleneck:* port addressing. *Holds when:* any number of pebbles; d = distance to the treasure.
- Gorain et al. (2022) [current best]: cost $O(d \log \Delta + \log^3 \Delta)$, help $O(d \log \Delta)$ pebbles. Introductory and current best (SIROCCO 2021 bounds; whether the journal version improves them was not checked).

Claude (Anthropic) was used in cataloguing the problems.

16.2.5 Labelled nodes, locally finite graphs

Adds + locally finite, + labelled nodes **to** Anonymous graphs with possibly infinite degrees (p. 112) VERIFIED

With node labels, a treasure at distance d is found at cost $O(e(d)\log d)$, against a lower bound of $\Omega(e(d))$.

Assumed: graph dynamics: static graph; edge direction: undirected; graph size: finite or countably infinite, every degree finite; node identifiers: labelled nodes (an agent can recognise a node it has visited). **Not assumed:** external help: no help; topology: arbitrary connected graph.

Measure	Lower bound	Best upper bound	Status
Cost	$\Omega(e(d))$ (Bouchard et al. 2023a) A $\log d$ factor.	$O(e(d)\log d)$ (Bouchard et al. 2023a)	• near-optimal
Help	none known No help is used.	—	• tight

Results.

- Bouchard et al. (2023a) [lower bound]: cost $\Omega(e(d))$. *Bottleneck:* distance. *Holds when:* $e(d)$: edges with an endpoint at distance below d; holds for every algorithm on some graphs.
- Bouchard et al. (2023a) [current best]: cost $O(e(d)\log d)$. *Assumes:* unbounded memory. Introductory and current best; holds also in the fuel- and rope-restricted models, and refutes a 1999 conjecture of Awerbuch, Betke, Rivest and Singh.

Open questions.

Q1. Remove the $O(\log d)$ factor between $O(e(d)\log d)$ and $\Omega(e(d))$ (Bouchard et al. 2023a).

16.2.6 Dynamic graphs, connected at every step

Adds + connected every round, + undirected, + finite graph VERIFIED

If the edges may change at every step while every snapshot stays connected, no agent ever finds the treasure, even when every snapshot is a ring; if every two nodes become adjacent within every ξ steps and agents flag the nodes they visit, $\Theta(\xi n)$ steps are necessary and sufficient on such rings.

Assumed: graph dynamics: dynamic graph, connected in every round (1-interval connected); edge direction: undirected; graph size: finite. **Not assumed:** node identifiers: anonymous nodes; external help: no help; topology: arbitrary connected graph.

Statement for this setting. The graph is dynamic on a fixed set of n nodes: an adversary chooses the edges at every step t, and every snapshot is connected (1-interval connectivity). The known results concern **vertex-permuted rings**, a special case: every snapshot is a cycle through all n nodes, and the adversary may place the nodes on this cycle in a new order at every step. The treasure stays at one node. The agent chooses each move online, from what it sees at that step, without knowing the snapshots to come. **Flags**, used by the algorithms described in the notes, are marks that an agent plants on every node it visits and that agents on neighbouring nodes see.

Measure	Lower bound	Best upper bound	Status
Cost	*impossible* (Ayoubi et al. 2026) [preprint] Impossible already when every snapshot is a ring whose nodes the adversary may reorder freely (Theorem 1 of Ayoubi et al. 2026); the notes describe the restricted class VP(ξ), in which it becomes solvable.	—	• impossible
Help	none known External help (advice, pebbles placed by an oracle) is not studied for dynamic graphs.	—	• unknown

Claude (Anthropic) was used in cataloguing the problems.

Results.

- Ayoubi et al. (2026) [preprint] [lower bound]: cost *impossible*. *Bottleneck:* adversarial edge changes. *Holds when:* Theorem 1: vertex-permuted rings, so it holds in every class of dynamic graphs that contains them; when the agent moves from a node u to a neighbour v, the next snapshot swaps u and v, so the agent stays in the middle of the same three consecutive nodes and the treasure is elsewhere; holds for deterministic and randomized agents, even with memory, chirality, a view of the whole snapshot and knowledge of the node labels; holds for up to $n-3$ agents starting at one node.

Open questions.

Q1. With flags in VP(ξ), the lower bound of $\xi(n-3)$ steps for one deterministic agent is proved for $\xi \geq 2n$; does $\Omega(\xi n)$ hold for every $\xi \geq \lceil (n-1)/2 \rceil$, the smallest window for which VP(ξ) is non-empty? (Ayoubi et al. 2026)

Q2. What is the competitive ratio of randomized agents in VP(ξ) against an adaptive online adversary? (Ayoubi et al. 2026)

Q3. Which of these results extend to vertex-permuted graphs other than rings? (Ayoubi et al. 2026)

Notes. Ayoubi, Mans and Narayanan make the problem solvable with two restrictions. In the class VP(ξ) (their VP(δ); their δ is ξ here) every two nodes are adjacent in at least one step of each window of ξ consecutive steps; the class is non-empty if and only if $\xi \geq \lceil (n-1)/2 \rceil$ (Proposition 1, through Walecki's decomposition of the complete graph into Hamiltonian cycles). With flags, a memoryless agent that moves to an unflagged neighbour whenever it sees one finds the treasure within $\xi(n-1)$ steps (Theorem 2), and for $\xi \geq 2n$ every deterministic agent needs $\xi(n-3)$ steps on some input, even with memory, chirality, a view of the whole snapshot and knowledge of the node labels (Theorem 3); without node labels its competitive ratio is $\Omega(\xi n)$ (Theorem 4). Without flags and without node labels, no deterministic agent succeeds for $\xi \geq n$, even with memory, chirality and a view of the whole snapshot (Theorem 5), while a randomized agent that stays put with probability $1 - 1/\xi$ finds the treasure in expected $O(\xi^2)$ steps (Theorem 6), even without knowing n or ξ (Remark 1). With k agents, flags, and access to each port by mutual exclusion, $\xi\lceil 2n/k \rceil$ steps suffice for $\xi \geq \lceil (n-1)/2 \rceil$ (Theorem 8), and $(\lceil (n-2)/k \rceil - 2)\,\xi$ steps are necessary for $1 \leq k \leq n/2 - 2$ and $\xi \geq (k+1)(n-k-2)$ (Theorem 9). If each snapshot is a ring in a uniformly random order (R-VP), an agent that always moves to its clockwise neighbour finds the treasure in expected $\Theta(n)$ steps when the treasure is placed in advance, and $\Theta(n \log n)$ when an adversary that knows the sequence of snapshots places it (Theorem 7). These results need flags, the window ξ or random snapshots, which the axes do not record, so they are described here rather than listed as bounds. Checked against the full text (preprint, v1).

16.3 Rendezvous

Two agents start at different nodes of an unknown graph and must meet; how long does it take, and how much memory or symmetry breaking does it need?

First posed in 2003 (Dessmark et al. 2006). Deterministic rendezvous of labelled agents in arbitrary graphs was posed by Dessmark, Fraigniaud and Pelc at ESA 2003. Rendezvous itself is older (randomized search games); the catalogue covers the deterministic line (Pelc 2019).

Claude (Anthropic) was used in cataloguing the problems.

Problem statement and measures

Two mobile agents start at different nodes of an unknown connected port-labelled graph. Nodes are anonymous unless a setting gives them **node labels**: distinct positive integers that an agent reads when it is at the node. In the synchronous model they must be at the same node in the same round; in the asynchronous model an adversary controls their speeds and they may also meet inside an edge. Agents are either **labelled** (distinct labels from $\{1, \dots, L\}$; λ is the smaller label) or **anonymous**. They may start at different times; τ is the delay. **Time** is the number of rounds from the start of the later agent (synchronous) or the total number of moves, that is edge traversals (asynchronous cost). **Memory** is bits per agent.

Measures: **Time**, rounds after the later agent starts (synchronous); edge traversals (asynchronous); **Memory**, bits per agent.



(a) Start: agents 1 and 3 at the ends of a tree (n = 4)

(b) Final: agent 3 walks onto the waiting agent 1 in round 21

Figure 16.3: Rendezvous of two labelled agents in a tree, simultaneous start, by the explore-or-wait idea of (Dessmark et al. 2006) (simplified): after one depth-first tour to learn n, each agent reads its label in binary with every bit doubled and 01 appended, and spends one block of $2(n-1)$ rounds per bit, exploring and returning on a 1, waiting at its start node on a 0. Labels 1 and 3 give 11 01 and 11 11 01; in block 3 agent 3 explores while agent 1 waits, and they meet. One frame per round.

A first look

Two friends lose each other in a city with no phones and want to meet at the same spot. The city is a **graph**: street corners are **nodes** and the streets between them are **edges**; the friends are two **agents** that start at different nodes of this unknown connected graph. At each node the edges at that node carry local numbers, its **ports**. Time passes in synchronous **rounds**; in each round an agent stays or makes a **move** along one edge, and the agents **meet** when they are on the same node in the same round (crossing inside an edge does not count). In the basic version each agent knows its own **label**, a distinct integer from $\{1, \dots, L\}$, but not the other's label and nothing about the graph; λ is the smaller of the two labels, n the number of nodes, and the later agent may start τ rounds after the earlier one (the **delay**). The task is to meet, and the cost is the number of rounds from the start of the later agent.

1. Algorithm: Explore or wait, bit by bit (trees, simultaneous start). (Dessmark et al. 2006) (Labelled agents in trees, p. 125) If one agent knew it should wait, the other could simply search the graph ("wait for mommy" in Dessmark et al. 2006). Without knowing the other label, use the bits of your own label to take turns. This is a simplified version of their idea, for a tree (no cycles), both agents starting in the same round.

1. Explore the tree by depth-first search and return to your **start node** (the node where you began); you learn n, and this takes $2(n-1)$ rounds. Call $2(n-1)$ rounds a **block**.

Claude (Anthropic) was used in cataloguing the problems.

2. Write your label in binary, write every bit twice, and append 0 1. For example 5 = 101 becomes 11 00 11 01.
3. Go through this string one bit per block: on a bit 1, explore the tree and return to your start node; on a bit 0, wait at your start node for the whole block.

Why it works and what it costs. Both agents compute the same block length, so their blocks start and end in the same rounds. Two different labels give strings that differ at some position before the shorter one ends (the final 0 1 marks the end, and no doubled bit looks like it). In that block one agent explores while the other waits at its start node, so the exploring agent walks onto it. The cost is $O(\log \lambda)$ blocks of $O(n)$ rounds, $O(n \log \lambda)$ in all. (Dessmark et al. 2006) reach $O(n + \log \lambda)$ on trees even with a delay τ, which is optimal; on general graphs the best known is $\tilde{O}(n^5 \log \lambda)$ (Ta-Shma and Zwick 2014), against a lower bound of $\Omega(n^2)$.

Why it matters. Turning distinct labels into different move/wait patterns is the basic symmetry-breaking device of deterministic rendezvous.

2. Impossibility: Identical agents in symmetric positions. (Czyzowicz et al. 2012b) (Anonymous agents: memory, p. 126) Take a ring of 6 nodes where at every node port 0 leads clockwise and port 1 counterclockwise. Put two agents without labels (**anonymous**: same algorithm, same initial memory) on opposite nodes, 3 edges apart, starting in the same round. Each agent sees exactly what the other sees: a node of **degree** 2 (two edges at it), the same entry ports, nobody else around. So in every round they make the same choice: both stay, both go clockwise, or both go counterclockwise.

Why it holds. After any number of rounds they are still exactly 3 edges apart, so they never meet, whatever the algorithm and however much **memory** (bits kept between rounds) they have. Identical agents can meet only when their starting positions are not symmetric in this sense, and for those positions $\Theta(\log n)$ bits of memory are exactly what is needed (Czyzowicz et al. 2012b).

Why it matters. It explains why most rendezvous work gives the agents labels, and why for identical agents the question becomes which starting positions can be told apart.

3. Lower bound: Two nodes and the length of a label. (Dessmark et al. 2006) (Labelled agents, arbitrary finite graphs, p. 124) Take the smallest graph: two nodes joined by one edge, one agent at each end, both starting together. Until they meet, an agent sees nothing new (always degree 1, always port 0), so its behaviour in the first t rounds is a string of t choices, stay or move, fixed by its label alone. They meet in a round where exactly one of them moves; if both move they just swap ends.

Why it holds. There are only 2^t strings of t choices. Among $2^t + 1$ labels, two give the same string; give these labels to the two agents and they do the same thing in each of the first t rounds, so they do not meet within t rounds. Hence some pairs of labels need about log (number of labels) rounds, which gives the $\Omega(\log \lambda)$ bound of (Dessmark et al. 2006). Together with the $\Omega(n)$ needed to cover the graph it shows their $O(n + \log \lambda)$ algorithm for trees is optimal.

Why it matters. It shows that the cost of breaking symmetry is the length of a label, a counting argument that recurs in every model with identifiers.

What the community started looking at

Deterministic rendezvous in graphs started with the question of **feasibility and cost polynomial in the graph size and the label length**, for labelled agents that may start at different times (Dessmark et al. 2006). The community then removed the dependence on the delay (Kowalski and Malinowski 2008), made the algorithms constructive (Ta-Shma and Zwick 2014), asked how little **memory** anonymous agents need (Czyzowicz et al. 2012b; Fraigniaud and Pelc 2013), and moved to asynchronous agents,

Claude (Anthropic) was used in cataloguing the problems.

where even feasibility in every graph took years (De Marco et al. 2006; Czyzowicz et al. 2012a; Dieudonné et al. 2015). Pelc's survey records this evolution (Pelc 2019). Node labels are another source of symmetry breaking: on paths with labelled nodes, anonymous agents meet in time proportional to d times the iterated logarithm of a nearby node label (Bourreau et al. 2026).

Further reading

Papers on this problem that are not yet placed in a setting.

- Kammer et al. (2026). Temporal rendezvous: two synchronous agents know the whole always-connected temporal graph but share no labelling of its nodes and cannot communicate; the adversary picks the starting nodes and the labellings. They meet within $O(n \log^3 n)$ steps using polynomial computation, for a long enough lifetime (Theorem 1.4), against an $\Omega(n \log n)$ lower bound from the paper that posed the problem (Dogeas et al., ICALP 2024, not in the catalogue), as reported. Preprint; read.
- Das (2026). Footprints: every node an agent visits keeps a permanent mark, the same for both agents, which an agent sees when it is at the node. For two anonymous agents with simultaneous start, Das gives a universal algorithm, which meets in every feasible instance (a graph and two start nodes), in every connected graph, finite or countably infinite with finite degrees (Theorem 2.1, Corollary 2.1); this answers an open question of Das and Pelc (SPAA 2026, not in the bibliography). By their Proposition 2.1, recalled here, an instance is feasible only if the views from the two start nodes differ or the footprints seen along some path differ. No bound on time is given. The problem has no axis for marking nodes yet. Preprint; read.
- Das and Pelc (2026). Two anonymous agents modelled as copies of one deterministic finite automaton, synchronous, with arbitrary delay, in finite graphs whose degrees are bounded by a constant known to the designer; the automaton must meet in every feasible instance of a class. Without marking no such automaton exists even for paths (Proposition 2.1); with any finite number of pebbles that can be picked up again, none exists for all graphs (Proposition 2.2, through the graphs of Rollik 1980); with finitely many pebbles that cannot be picked up, none exists even for paths (Theorem 3.1, Corollary 3.1); with one pebble per agent that can be picked up, one automaton meets in every feasible instance on trees, within $O(n^2)$ rounds from the start of the earlier agent (Theorem 4.1, Corollary 4.1). Without pebbles, $\Theta(\log n)$ bits are needed in trees with delay (Fraigniaud and Pelc 2013), so one pebble replaces them by constant memory. The problem has no axis for marking nodes yet. Preprint; read.
- Baaziz and Pelc (2026a). Rendezvous in the plane rather than in a graph. Two anonymous agents, copies of one deterministic finite automaton with a compass, see the disc of radius 1 around them before each move, move asynchronously (the adversary sets the speed during each move), and carry pebbles that they can pick up again and that the other agent can tell apart from its own. With at most 8 pebbles each they always meet, at cost (total length of both trajectories) $O(d^2)$ when the start points are at distance at most d (their D) (Theorem 2); the authors argue in Section 5 that $\Omega(d^2)$ is necessary, since the adversary can hold one agent back until the other finds it. The same algorithm elects a leader between the two agents. Read (LIPIcs version).
- Gao and Pelc (2026). Two labelled agents that are finite automata (states depending only on L), started in the same round on distinct nodes of the infinite oriented path, can be placed so that they never meet (Theorem 3.1, for any set of such agents); teams of two or more agents per starting node gather (Theorems 4.1–6.2). Read (arXiv v1).
- Yamada et al. (2026). A different model: two metamorphic robotic systems (connected oblivious modules on the square grid) without a common compass meet until every module sees all the others, then merge into one connected system; five modules per system suffice (Theorems 1 and 2) and are necessary for rendezvous (Theorem 3). Read (arXiv v1).
- Bampas et al. (2019a). Characterises, in polynomial time, the pairs of walks that force rendezvous against an asynchronous adversary within a given number of edge traversals, with a certificate of why. Abstract read.
- Bouchard et al. (2018). Rendezvous at a node (not inside an edge) of labelled agents whose speed the adversary fixes per agent and per edge: polynomial in n, the smaller label and the largest traversal time. Abstract read.
- Chalopin et al. (2016a). Labelled agents with delay faults, measured by cost: polynomial cost with random faults; impossible with unbounded adversarial faults even on rings (cost $O(n\lambda)$ on trees, $\Omega(\lambda)$ needed); polynomial

Claude (Anthropic) was used in cataloguing the problems.

with faults bounded by an unknown c. Abstract read.

- Dereniowski et al. (2015b). Agents that know the map but traverse edges at different, private speeds: $O(n\,T_{\mathrm{OPT}})$ in general, $\Theta(T_{\mathrm{OPT}})$ if they exchange $\Theta(n)$ bits at the start, with T_{OPT} the offline meeting time. Abstract read.
- Miller and Pelc (2016). Trade-off between time and cost (edge traversals) for labelled agents, as functions of an exploration bound E: cost $O(E)$ with time $O(EL)$, or both $O(E \log L)$, with lower bounds on the trade-off. Abstract and introduction read.
- Miller and Pelc (2015a). How much advice gives rendezvous in time $O(d)$: $\Theta(d \log(n/d) + \log\log L)$ bits. Abstract and introduction read.
- Eguchi et al. (2022). Neighbourhood rendezvous (agents start at adjacent nodes) in $o(\Delta)$ rounds, with randomisation, node identifiers, neighbourhood identifiers and bounded minimum degree. Abstract read.
- Cosson (2024a). Two agents with whiteboards: exploration in m steps from a common start and rendezvous in at most $\lceil 3m/2 \rceil$ steps, also against an asynchronous adversary, with matching lower bounds there. A preprint; abstract and main results read.
- Gao and Pelc (2025). Identical agents in the infinite oriented grid that mark the nodes they visit: time $O(D)$ if they know the distance bound D or start together, and $O(D^2)$ with neither, where a superlinear lower bound separates the cases. Abstract read.
- Das et al. (2015a). Rendezvous of several asynchronous anonymous finite-state agents despite a fast malicious agent that blocks nodes: necessary conditions, solved on rings except for even numbers of agents on unoriented rings, and on oriented meshes iff agents see two hops. Abstract read.
- Dogeas et al. (2024). Temporal graphs with r automorphism orbits are explored in $O(rn^{1+\varepsilon})$ steps, with lower bounds; the paper also poses rendezvous in temporal graphs. Abstract read.
- Mironowicz (2023). Location-aware agents on small networks that share entangled quantum states: optimal strategies with a limited number of steps, with an advantage over classical agents on cubic graphs and cycles. Introduction read.
- Pelc and Yadav (2020). Gathering of anonymous agents in the plane with compasses and clocks, where delays can break symmetry: characterisation of gatherable configurations, no common algorithm for the 'bad' ones, and the knowledge of the team size needed for the others. Abstract read.
- Dieudonné et al. (2023). Rendezvous of two anonymous agents in the plane whose attributes (coordinates, clock rate, speed, wake-up) may differ: exact characterisation of feasible instances, no single algorithm for two small sets of them, and one algorithm for all others. Abstract read.
- Elouasbi and Pelc (2018). Labelled disc agents in the plane that sense their distance but not the direction to each other: deterministic rendezvous under two models of distance estimation. Abstract read.
- Czyzowicz et al. (2018b). Two anonymous robots on the line whose speeds or clock units may differ: a universal algorithm meets whenever at least one parameter differs. Abstract read.
- Georgiou et al. (2018). Randomised symmetric rendezvous of two robots at distance 2 with a common reference point (in a disk): expected time below the 4.25 of the line. Abstract read.
- Chuangpishit et al. (2017). Location-aware robots on a line, some of them Byzantine: a central authority plans trajectories with bounded competitive ratio, better when the number of faulty robots is bounded. Abstract read.
- Ozsoyeller et al. (2021). Symmetric rendezvous of two robots in planar environments with and without obstacles. Abstract read.

Relations to other problems

- **Subproblem of Graph exploration by a single agent.** If the agents can break symmetry (for example by their labels), one may wait while the other explores, so exploration solves rendezvous; the work is in doing better than that, or in meeting when neither knows who should wait (Pelc 2019).

Claude (Anthropic) was used in cataloguing the problems.

- **Derived problem:** **Gathering in graphs** (generalisation of). Rendezvous is gathering of two agents; with more agents the questions become which teams can gather, and how many faulty agents a team tolerates.

Assumptions specific to this problem

Graph size

Deterministic rendezvous in graphs was posed for finite graphs, where an agent can afford to explore the whole graph and the cost is a function of n (Dessmark et al. 2006; Pelc 2019). Bhagat and Pelc argued that this approach "is inefficient for very large graphs, and cannot be used for infinite graphs", and gave the first synchronous algorithm for connected graphs with finite degrees and finitely or countably many nodes, with time polynomial in d, $\log L$ and the number of paths of length d from the starting nodes, a quantity that also bounds rendezvous from below in trees (Bhagat and Pelc 2022). For asynchronous agents, Czyzowicz, Labourel and Pelc had already shown that rendezvous is feasible in every connected countable graph (Czyzowicz et al. 2012a) (as summarised in Pelc 2019); like all earlier work it assumed finite degrees (Pattanayak and Pelc 2024b). Pattanayak and Pelc then allowed countably infinite degrees, where the path count is infinite, and obtained a universal synchronous algorithm with time polynomial in a weight of the path types between the starting nodes and in $\log L$, matched up to a polynomial by a lower bound in the tree whose degrees are all infinite (Pattanayak and Pelc 2024b). The axis thus separates three cost measures: n, the local path count, and path-type weights.

Level	Value	Formal definition
0	**infinite degrees allowed**	G is finite or countably infinite; degrees may be infinite.
1	**locally finite**	G is finite or countably infinite with finite degrees.
2	**finite graph**	G has finitely many nodes.

Start

Dessmark, Fraigniaud, Kowalski and Pelc treated two startup scenarios for deterministic rendezvous: simultaneous startup, and an arbitrary delay τ chosen by the adversary, in which each agent counts rounds from its own wake-up without knowing τ (Dessmark et al. 2006). The delay is the stronger adversary: it can delay one agent long enough that the other must find it while it sleeps, which is treasure hunt (Pelc 2019; Ta-Shma and Zwick 2014). It changes cost: on a ring simultaneous start gives $\Theta(d \log \lambda)$, while with arbitrary delay $\Omega(n + d \log \lambda)$ is a lower bound even when n is known; and the first polynomial algorithm for arbitrary graphs had cost growing with $\sqrt{\tau}$, leaving open whether the dependence could be removed (Dessmark et al. 2006), which Kowalski and Malinowski and then Ta-Shma and Zwick did (Kowalski and Malinowski 2008; Ta-Shma and Zwick 2014). It also changes memory: Fraigniaud and Pelc showed that identical agents need $\Theta(\log n)$ bits in trees with arbitrary delay, even on a line, but only $O(\log \nu + \log \log n)$ bits with simultaneous start in trees with ν leaves, an exponential gap when ν is small (Fraigniaud and Pelc 2013).

Level	Value	Formal definition
0	**any delay**	An adversary wakes the agents at times t_1 and t_2; the delay $\tau = \lvert t_1 - t_2 \rvert$ is unknown.
1	**simultaneous start**	Both agents start in the same round ($\tau = 0$).

Claude (Anthropic) was used in cataloguing the problems.

Case tree

The root assumes nothing; going down, each edge adds the assumptions written in grey (+) or leaves an axis unassumed. Bold entries are catalogued settings with their status per measure (Time, Memory) and page; italic entries are branch points, named after the axis on which their branches differ. Axis order: Graph dynamics → Edge direction → Graph size → Node identifiers → Identifiers → Timing → Start → Topology.

No assumptions

+ static graph, + undirected

Graph size

+ finite graph

Asynchronous anonymous agents ●● p. 123

+ distinct IDs (eponymous)

Asynchronous labelled agents ●● p. 123

+ synchronous

Labelled agents, arbitrary finite graphs STANDARD SETTING ●● p. 124

any delay, + tree

Labelled agents in trees ●● p. 125

+ simultaneous start, + ring

Labelled agents in rings, simultaneous start ●● p. 126

anonymous, + synchronous

Anonymous agents: memory ●● p. 126

+ simultaneous start, + tree

Anonymous agents in trees, simultaneous start ●● p. 127

+ locally finite

Node identifiers

anonymous nodes, + distinct IDs (eponymous), + synchronous

Start

any delay, + tree

Labelled agents in infinite trees ●● p. 127

+ simultaneous start, + tree

Labelled agents in infinite trees, simultaneous start ●● p. 128

+ labelled nodes, + synchronous, + ring

Anonymous agents on paths and rings with labelled nodes ●● p. 129

infinite degrees allowed, + distinct IDs (eponymous), + synchronous

Labelled agents, graphs with infinite degrees ●● p. 130

What is known

The best upper bound against the best lower bound in each catalogued setting, in tree order. The settings follow in detail.

Setting	Time	Memory	p.
Asynchronous anonymous agents	● unknown *finite* vs —	● unknown — vs —	123
Asynchronous labelled agents	● unknown poly(n, log λ) vs —	● unknown — vs —	123

Claude (Anthropic) was used in cataloguing the problems.

Setting	Time	Memory	p.
Labelled agents, arbitrary finite graphs STANDARD SETTING	● polynomial gap $\tilde{O}(n^5 \log \lambda)$ vs $\Omega(n^2)$	● unknown — vs —	124
Labelled agents in trees	● tight $O(n + \log \lambda)$ vs $\Omega(n + \log \lambda)$	● unknown — vs —	125
Labelled agents in rings, simultaneous start	● tight $O(d \log \lambda)$ vs $\Omega(d \log \lambda)$	● unknown — vs —	126
Anonymous agents: memory	● polynomial gap $(n+\tau)^{O(n+\tau)}$ vs $2^{\Omega(d)}$	● tight $O(\log n)$ vs $\Omega(\log n)$	126
Anonymous agents in trees, simultaneous start	● unknown — vs —	● tight $O(\log \nu + \log \log n)$ vs $\Omega(\log \nu + \log \log n)$	127
Labelled agents in infinite trees	● near-optimal $O(\text{ball}(d) \log L)$ vs $\Omega(\text{ball}(d) + d \log L)$	● unknown — vs —	127
Labelled agents in infinite trees, simultaneous start	● near-optimal $O(d \log L)$ vs $\Omega(\text{ball}(d) + d \log L)$	● unknown — vs —	128
Anonymous agents on paths and rings with labelled nodes	● tight $O(d \log^* \sigma_{\min})$ vs $\Omega(d \log^* \sigma_{\max})$	● unknown — vs —	129
Labelled agents, graphs with infinite degrees	● polynomial gap $O((W \log L)^3)$ vs $\Omega(\max(W, \log L))$	● unknown — vs —	130

16.3.1 Asynchronous anonymous agents

Adds + finite graph VERIFIED

Which pairs of starting nodes allow identical asynchronous agents to meet is characterised.

Assumed: graph dynamics: static graph; edge direction: undirected; graph size: finite. **Not assumed:** node identifiers: anonymous nodes; identifiers: anonymous: all agents are identical; timing: asynchronous; start: arbitrary delay between the two starts; topology: arbitrary connected graph.

Measure	Lower bound	Best upper bound	Status
Time	none known Cost not studied.	*finite* (Guilbault and Pelc 2011)	● unknown
Memory	none known Memory is unbounded.	—	● unknown

Results.

- Guilbault and Pelc (2011) [current best]: time *finite*. Introductory and current best. Feasible if and only if the views from the two starting nodes differ or the nodes are joined by a path whose sequence of ports reads the same in both directions (Theorem 1), in finite or countably infinite graphs; randomized agents meet with probability 1 from any starting nodes (Theorem 2).

16.3.2 Asynchronous labelled agents

Adds + distinct IDs (eponymous) **to** Asynchronous anonymous agents (p. 123) VERIFIED

An adversary controls speeds; agents may meet inside an edge. Feasible everywhere, and at polynomial cost,

Claude (Anthropic) was used in cataloguing the problems.

but with very large exponents.

Assumed: graph dynamics: static graph; edge direction: undirected; graph size: finite; identifiers: distinct identifiers (eponymous): agents have pairwise distinct identifiers. **Not assumed:** node identifiers: anonymous nodes; timing: asynchronous; start: arbitrary delay between the two starts; topology: arbitrary connected graph.

Measure	Lower bound	Best upper bound	Status
Time	none known No lower bound recorded for general graphs.	poly$(n, \log \lambda)$ (Dieudonné et al. 2015)	• unknown
Memory	none known Memory is unbounded.	—	• unknown

Results.

- De Marco et al. (2006) [introductory]: time *finite*. *Assumes:* an upper bound on the size is known. On a ring of known size, cost $O(n \log \lambda)$, which is optimal.
- Czyzowicz et al. (2012a) [also]: time *finite*. Feasible in every connected countable graph without any knowledge; cost at least exponential.
- Dieudonné et al. (2015) [current best]: time poly$(n, \log \lambda)$. First polynomial cost.

Open questions.

Q1. Find an optimal-cost asynchronous rendezvous algorithm in arbitrary graphs (open problem of Pelc 2019).

16.3.3 Labelled agents, arbitrary finite graphs

Adds + synchronous **to** Asynchronous labelled agents (p. 123) STANDARD SETTING VERIFIED

Polynomial in n and $\log \lambda$ and independent of the delay, but far from the known lower bounds.

Assumed: graph dynamics: static graph; edge direction: undirected; graph size: finite; identifiers: distinct identifiers (eponymous): agents have pairwise distinct identifiers; timing: synchronous rounds. **Not assumed:** node identifiers: anonymous nodes; start: arbitrary delay between the two starts; topology: arbitrary connected graph.

Measure	Lower bound	Best upper bound	Status
Time	$\Omega(n^2)$ (Dessmark et al. 2006); $\Omega(\log \lambda)$ (Dessmark et al. 2006) $\tilde{O}(n^5 \log \lambda)$ versus $\Omega(n^2 + \log \lambda)$.	$\tilde{O}(n^5 \log \lambda)$ (Ta-Shma and Zwick 2014)	• polynomial gap
Memory	none known Memory is unbounded in these algorithms.	—	• unknown

Results.

- Dessmark et al. (2006) [lower bound]: time $\Omega(n^2)$. *Bottleneck:* exploration. *Holds when:* Theorem 4.4: for all n and all pairs of labels, a regular graph made of two cliques of $n/2$ nodes joined by two swapped edges, ports equal at both ends of each edge; holds even with simultaneous start.
- Dessmark et al. (2006) [lower bound]: time $\Omega(\log \lambda)$. *Bottleneck:* symmetry breaking. *Holds when:* the two-node graph, agents at both ends, simultaneous start: with fewer steps than label bits two labels produce the same move/stay sequence; extends to trees (Theorem 2.3).
- Dessmark et al. (2006) [introductory]: time $O(n^5 \sqrt{\tau \log \lambda}\, \log n + n^{10} \log^2 n \log \lambda)$. *Assumes:* memory unbounded; local computation free. Theorem 4.3: polynomial in n, τ and $\log \lambda$. Not constructive in the computational sense: each agent finds, by exhaustive local search, a combinatorial object whose existence is proved by the probabilistic method; only moves are counted.

Claude (Anthropic) was used in cataloguing the problems.

- Kowalski and Malinowski (2008) [also]: time $O(\log^3 \lambda + n^{15} \log^{12} n)$. *Assumes:* backtracking: the agent learns the port by which it entered; memory unbounded; local computation free. Corollary 4.1; the first bound independent of τ, answering the question of (Dessmark et al. 2006). The cover walk it uses runs shorter cover walks recursively and goes back along them, so it needs backtracking (Section 4.1.2). Like (Dessmark et al. 2006) it is deterministic but not explicit: each agent finds by exhaustive local search a walk whose existence is proved by the probabilistic method. The same algorithm gathers k labelled agents.

- Ta-Shma and Zwick (2014) [current best]: time $\tilde{O}(n^5 \log \lambda)$. Independent of τ and needs no backtracking, but not explicit: it relies on universal traversal sequences that are only known to exist. Checked in the paper. (Xin 2007 wrongly says it uses backtracking.)

Open questions.

Q1. Find an optimal-time synchronous rendezvous algorithm for labelled agents in arbitrary graphs (open problem of Pelc 2019).

Notes. ℓ in (Ta-Shma and Zwick 2014) is the length of the shorter label, our $\log \lambda$; ℓ in (Kowalski and Malinowski 2008) is the smaller label itself, our λ. Checked against the full texts of (Dessmark et al. 2006), (Kowalski and Malinowski 2008) and (Ta-Shma and Zwick 2014): agents know only their own label, learn the entry port and the degree at each node, and do not notice crossing inside an edge. With **simultaneous** start (Dessmark et al. 2006) meets in $O(d\Delta^d \log \lambda)$ rounds in any graph without knowing Δ (Theorem 4.1), optimal when d and Δ are bounded.

16.3.4 Labelled agents in trees

Adds any delay, + tree **to** Labelled agents, arbitrary finite graphs (p. 124) VERIFIED

In trees, $O(n + \log \lambda)$ even with arbitrary delay, and this cannot be improved.

Assumed: graph dynamics: static graph; edge direction: undirected; graph size: finite; identifiers: distinct identifiers (eponymous): agents have pairwise distinct identifiers; timing: synchronous rounds; topology: tree. **Not assumed:** node identifiers: anonymous nodes; start: arbitrary delay between the two starts.

Measure	Lower bound	Best upper bound	Status
Time	$\Omega(n + \log \lambda)$ (Dessmark et al. 2006) $\Theta(n + \log \lambda)$.	$O(n + \log \lambda)$ (Dessmark et al. 2006)	• tight
Memory	none known Memory is unbounded.	—	• unknown

Results.

- Dessmark et al. (2006) [lower bound]: time $\Omega(n + \log \lambda)$. *Bottleneck:* distance. *Holds when:* Theorem 2.3: two stars of $n/2 - 1$ leaves whose centres are joined, agents at the centres, for the $\Omega(n)$ part; the two-node argument for the $\Omega(\log \lambda)$ part; holds even with simultaneous start.

- Dessmark et al. (2006) [current best]: time $O(n + \log \lambda)$. *Assumes:* agents know that the graph is a tree, but not its size. Introductory and current best (Theorem 2.2), with arbitrary delay: explore by depth-first search, go to the central node or the central edge, and on a central edge run Procedure Extend-Labels, which meets on K_2 within $2 \log \lambda + 6$ rounds.

Notes. Checked against the full text of (Dessmark et al. 2006).

Claude (Anthropic) was used in cataloguing the problems.

16.3.5 Labelled agents in rings, simultaneous start

Adds + simultaneous start, + ring **to** Labelled agents, arbitrary finite graphs (p. 124) VERIFIED

With simultaneous start on a ring, rendezvous takes $\Theta(d \log \lambda)$ for initial distance d.

Assumed: graph dynamics: static graph; edge direction: undirected; graph size: finite; identifiers: distinct identifiers (eponymous): agents have pairwise distinct identifiers; timing: synchronous rounds; start: simultaneous start; topology: ring or path. **Not assumed:** node identifiers: anonymous nodes.

Measure	Lower bound	Best upper bound	Status
Time	$\Omega(d \log \lambda)$ (Dessmark et al. 2006) $\Theta(d \log \lambda)$.	$O(d \log \lambda)$ (Dessmark et al. 2006)	• tight
Memory	none known Memory is unbounded.	—	• unknown

Results.

- Dessmark et al. (2006) [lower bound]: time $\Omega(d \log \lambda)$. *Bottleneck:* symmetry breaking. *Holds when:* rings, for every initial distance d (Theorem 3.1); agents know that the graph is a ring.
- Dessmark et al. (2006) [current best]: time $O(d \log \lambda)$. *Assumes:* agents know that the graph is a ring, but not its size. Introductory and current best (Lemmas 3.1, 3.2 and Theorem 3.1).

Notes. Checked against the full text of (Dessmark et al. 2006). With **arbitrary delay** on an n-node ring, $\Omega(n + d \log \lambda)$ rounds are necessary even when n is known (Proposition 3.1); $O(n \log \lambda)$ suffices for known n (Theorem 3.2) and $O(\lambda\tau + \lambda n^2)$ for unknown n (Theorem 3.3) (Dessmark et al. 2006). The paper announces that a forthcoming paper reaches $O(n \log \lambda)$ with arbitrary delay; this is (Kowalski and Malinowski 2008) (Theorem 3.1), for unknown n, which with Proposition 3.1 of (Dessmark et al. 2006) makes it optimal (their Corollary 3.1).

16.3.6 Anonymous agents: memory

Adds anonymous, + synchronous **to** Asynchronous anonymous agents (p. 123) VERIFIED

Identical agents can meet whenever their positions are not symmetric, or are symmetric but start with a long enough delay; $\Theta(\log n)$ bits suffice when the positions are not symmetric.

Assumed: graph dynamics: static graph; edge direction: undirected; graph size: finite; timing: synchronous rounds. **Not assumed:** node identifiers: anonymous nodes; identifiers: anonymous: all agents are identical; start: arbitrary delay between the two starts; topology: arbitrary connected graph.

Measure	Lower bound	Best upper bound	Status
Time	$2^{\Omega(d)}$ (Pelc and Yadav 2019b)	$(n+\tau)^{O(n+\tau)}$ (Pelc and Yadav 2019b)	• polynomial gap
	Exponential in d (Theorem 4.1 of Pelc and Yadav 2019b) against $(n+\tau)^{O(n+\tau)}$; whether a universal algorithm can run in time polynomial in n and τ is open.		
Memory	$\Omega(\log n)$ (Czyzowicz et al. 2012b) $\Theta(\log n)$ bits.	$O(\log n)$ (Czyzowicz et al. 2012b)	• tight

Results.

- Czyzowicz et al. (2012b) [lower bound]: memory $\Omega(\log n)$. *Bottleneck:* symmetry breaking. *Holds when:* rings; holds even with simultaneous start.
- Pelc and Yadav (2019b) [lower bound]: time $2^{\Omega(d)}$. *Bottleneck:* symmetry breaking. *Holds when:* symmetric start nodes at distance d with a feasible delay, in a graph built from a regular tree whose ports give the agents no information (Theorem 4.1); for an algorithm that meets for every such start, as a universal algorithm must.

Claude (Anthropic) was used in cataloguing the problems.

- Czyzowicz et al. (2012b) [current best]: memory $O(\log n)$. Introductory and current best; any delay; also gives the first polynomial-time computation of a quotient graph.
- Pelc and Yadav (2019b) [current best]: time $(n+\tau)^{O(n+\tau)}$. *Assumes:* every feasible start: nonsymmetric nodes with any delay, or symmetric nodes u, v with delay $\tau \geq \text{Shrink}(u, v)$; no knowledge of n, τ or the start nodes. Introductory and current best for starts with a delay. Corollary 3.1: a start is feasible if and only if the nodes are nonsymmetric or $\tau \geq \text{Shrink}(u, v)$, the smallest distance at which the same sequence of ports leads from u and from v (Lemma 3.1 proves necessity); Algorithm UniversalRV meets for every feasible start (Theorem 3.1). The time bound is derived in Section 4.

Open questions.

Q1. Is there a universal algorithm that meets for every feasible start in time polynomial in n and τ? Posed in (Pelc and Yadav 2019b).

Notes. With simultaneous start, rendezvous is feasible exactly when the two start nodes have different views. With a delay, symmetric start nodes become feasible once the delay reaches $\text{Shrink}(u, v)$ (Pelc and Yadav 2019b); Pelc and Yadav credit (Czyzowicz et al. 2012b) with the nonsymmetric case, for agents that know an upper bound on n (as reported). Checked against the full text of (Pelc and Yadav 2019b) (arXiv v1).

16.3.7 Anonymous agents in trees, simultaneous start

Adds + simultaneous start, + tree **to** Anonymous agents: memory (p. 126) VERIFIED

With simultaneous start, memory drops to $\Theta(\log \nu + \log\log n)$ for trees with ν leaves; with delay it is $\Theta(\log n)$, an exponential gap.

Assumed: graph dynamics: static graph; edge direction: undirected; graph size: finite; timing: synchronous rounds; start: simultaneous start; topology: tree. **Not assumed:** node identifiers: anonymous nodes; identifiers: anonymous: all agents are identical.

Measure	Lower bound	Best upper bound	Status
Time	none known Not the focus.	—	● unknown
Memory	$\Omega(\log \nu + \log\log n)$ (Fraigniaud and Pelc 2013) Matching bounds.	$O(\log \nu + \log\log n)$ (Fraigniaud and Pelc 2013)	● tight

Results.

- Fraigniaud and Pelc (2013) [lower bound]: memory $\Omega(\log \nu + \log\log n)$. *Bottleneck:* symmetry breaking. *Holds when:* trees of maximum degree 3; ν = number of leaves.
- Fraigniaud and Pelc (2013) [current best]: memory $O(\log \nu + \log\log n)$. Introductory and current best. With arbitrary delay, $\Omega(\log n)$ even on lines.

16.3.8 Labelled agents in infinite trees

Adds any delay, + tree VERIFIED

In infinite regular trees the time grows with the number of nodes within distance d: $O(\text{ball}(d)\log L)$ against $\Omega(\text{ball}(d) + d\log L)$.

Assumed: graph dynamics: static graph; edge direction: undirected; graph size: finite or countably infinite, every degree finite; identifiers: distinct identifiers (eponymous): agents have pairwise distinct identifiers; timing: synchronous rounds; topology: tree. **Not assumed:** node identifiers: anonymous nodes; start: arbitrary delay between the two starts.

Claude (Anthropic) was used in cataloguing the problems.

Measure	Lower bound	Best upper bound	Status
Time	$\Omega(\text{ball}(d) + d \log L)$ (Bhagat and Pelc 2024) At most a factor $\log L$; tight on the infinite path ($\Delta = 2$), where $\text{ball}(d) = \Theta(d)$.	$O(\text{ball}(d) \log L)$ (Bhagat and Pelc 2024)	• near-optimal
Memory	none known Memory is unbounded.	—	• unknown

Results.

- Bhagat and Pelc (2024) [lower bound]: time $\Omega(\text{ball}(d) + d \log L)$. *Bottleneck:* port addressing. *Holds when:* regular trees of degree $\Delta \geq 2$ with an adversarial port numbering, even with simultaneous start and known d and L (Theorem 2.2 and Corollary 2.3); $\text{ball}(d)$ is the number of nodes within distance d of a node, exponential in d when $\Delta > 2$; with arbitrary delay $\Omega(\text{ball}(d))$ holds even in oriented trees, where a root is marked and port 0 leads towards it (Proposition 3.1).
- Bhagat and Pelc (2024) [current best]: time $O(\text{ball}(d) \log L)$. *Assumes:* regular trees of constant degree; no knowledge of L or d; any delay. Introductory and current best (Theorem 2.1, Algorithm URT): each agent explores balls of growing radius, at times set by its transformed label. Time is counted from the wake-up of the earlier agent.

Open questions.

Q1. Close the gap between $O(\text{ball}(d) \log L)$ and $\Omega(\text{ball}(d) + d \log L)$ in unoriented regular trees. Posed in (Bhagat and Pelc 2024).

Q2. Find efficient rendezvous algorithms for arbitrary infinite graphs without extra knowledge such as the agents' positions. Posed in (Bhagat and Pelc 2024).

Notes. Bhagat and Pelc express the time in the initial distance d instead of n, since the tree may be infinite; their algorithms work in finite trees as well. Exploring a ball takes time linear in its size in a tree, but no such exploration is known in anonymous graphs with cycles, which is why they restrict to trees. Checked against the full text (arXiv v1).

16.3.9 Labelled agents in infinite trees, simultaneous start

Adds + simultaneous start, + tree VERIFIED

With simultaneous start, orientation helps: $O(d \log L)$ in oriented trees when a bound on L or d is known, $O(d^2 + \log^2 L)$ without; unoriented regular trees still need $\Omega(\text{ball}(d))$.

Assumed: graph dynamics: static graph; edge direction: undirected; graph size: finite or countably infinite, every degree finite; identifiers: distinct identifiers (eponymous): agents have pairwise distinct identifiers; timing: synchronous rounds; start: simultaneous start; topology: tree. **Not assumed:** node identifiers: anonymous nodes.

Measure	Lower bound	Best upper bound	Status
Time	$\Omega(\text{ball}(d) + d \log L)$ (Bhagat and Pelc 2024) Unoriented regular trees: within a factor $\log L$, as with a delay. Oriented trees: $\Theta(d \log L)$ with a known bound on L or d; without one, $O(d^2 + \log^2 L)$ against $\Omega(d \log L)$.	$O(d \log L)$ (Bhagat and Pelc 2024)	• near-optimal
Memory	none known Memory is unbounded.	—	• unknown

Results.

- Bhagat and Pelc (2024) [lower bound]: time $\Omega(\text{ball}(d) + d \log L)$. *Bottleneck:* port addressing. *Holds when:* regular trees of degree $\Delta \geq 2$ with an adversarial port numbering, even with simultaneous start and known d and L

Claude (Anthropic) was used in cataloguing the problems.

(Theorem 2.2 and Corollary 2.3); ball(d) is the number of nodes within distance d of a node, exponential in d when $\Delta > 2$; in oriented trees only $\Omega(d \log L)$ is known, from (Dessmark et al. 2006) (Section 3).

- Bhagat and Pelc (2024) [current best]: time $O(d \log L)$. *Assumes:* oriented trees: a root is marked and port 0 leads towards it; the tree need not be regular; a polynomial upper bound on L, or a linear upper bound on d, is known. Theorems 3.1 and 3.2; optimal.
- Bhagat and Pelc (2024) [also]: time $O(d^2 + \log^2 L)$. *Assumes:* oriented trees; no knowledge of L or d. Theorem 3.3; optimal when d and $\log L$ are of the same order.

Open questions.

Q1. In oriented trees with simultaneous start, can agents meet in time $O(d \log L)$ without knowing anything about d or L? Posed in (Bhagat and Pelc 2024).

Notes. Orientation has no axis here; it is recorded on the results. With arbitrary delay, orientation does not help (Proposition 3.1). Checked against the full text of (Bhagat and Pelc 2024) (arXiv v1).

GRAPHS

16.3.10 Anonymous agents on paths and rings with labelled nodes

Adds + labelled nodes, + synchronous, + ring VERIFIED

On paths with labelled nodes, two anonymous agents meet in $O(d \log^* \sigma_{\min})$ rounds, $\sigma_{\min}$ the smallest node label near their start nodes, and no algorithm is faster in the worst case.

Assumed: graph dynamics: static graph; edge direction: undirected; graph size: finite or countably infinite, every degree finite; node identifiers: labelled nodes (an agent can recognise a node it has visited); timing: synchronous rounds; topology: ring or path. **Not assumed:** identifiers: anonymous: all agents are identical; start: arbitrary delay between the two starts.

Statement for this setting. The graph is a path or a ring, finite or infinite: in the main case the two-way infinite path, every node of degree 2. Every node v carries a distinct node label $\sigma(v) \in \{1, 2, \dots\}$ that an agent reads when it is at v. The two agents are anonymous: they run the same deterministic algorithm and can break symmetry only through the node labels they see. An adversary wakes them at times of its choice; an awake agent that reaches the node of a sleeping agent meets it. They know neither d (the distance between their start nodes) nor the delay τ, share no orientation, see the port by which they enter a node, and do not notice crossing inside an edge. $\sigma_{\max}$ is the larger node label of the two start nodes, and $\sigma_{\min}$ the smallest node label within distance $O(d)$ of the start nodes; $\log^*$ is the iterated logarithm, taken to be at least 1.

Measure	Lower bound	Best upper bound	Status
Time	$\Omega(d \log^* \sigma_{\max})$ (Miller and Pelc 2025) $\Theta(d \log^* \sigma_{\max})$ in the worst case: the lower bound uses paths whose node labels all have the same $\log^*$, so there $\sigma_{\min}$ and $\sigma_{\max}$ give the same bound.	$O(d \log^* \sigma_{\min})$ (Bourreau et al. 2026)	● tight
Memory	none known Memory is unbounded.	—	● unknown

Results.

- Miller and Pelc (2025) [lower bound]: time $\Omega(d \log^* \sigma_{\max})$. *Bottleneck:* symmetry breaking. *Holds when:* an algorithm that always meets within $\frac{1}{16\kappa} \log^* \sigma_{\max}$ rounds from adjacent start nodes would give a 3-colouring algorithm for labelled paths in the LOCAL model faster than Linial's lower bound allows (κ the constant of their colouring algorithm; Theorem 4.9); for $d > 1$, $d - 1$ nodes inserted between consecutive nodes reduce to $d = 1$ (Theorem 4.10); a two-way infinite path made of consecutive segments, each labelled so that 3-colouring it needs time close to Linial's bound; holds even with simultaneous start, known d and a global sense of direction. Theorems 4.9 and 4.10 of the arXiv version (their D is d and their ℓ is $\sigma_{\max}$ here). On finite paths and rings with n nodes the bound becomes $\Omega(\min(n, d \log^* \sigma_{\max}))$ (Section 1.3).

Claude (Anthropic) was used in cataloguing the problems.

- Miller and Pelc (2025) [introductory]: time $O(d^2(\log^* \sigma_{\max})^3)$. *Assumes:* unbounded memory. Theorem 5.31 of the arXiv version: each agent knows only the node label of its start node. The algorithms 3-colour the path around the start nodes with an early-stopping version of the Cole–Vishkin algorithm, so that a node with a small node label stops early (Theorem 2.1), and let the colours decide which agent waits while the other searches. On finite paths and rings with n nodes, $O(\min(n, d^2(\log^* \sigma_{\max})^3))$.
- Bourreau et al. (2026) [current best]: time $O(d \log^* \sigma_{\min})$. *Assumes:* unbounded memory. Theorem 1.1 (arXiv v2), on the two-way infinite path, counting rounds from the wake-up of the earlier agent; Corollary 1.2 gives $O(\min(n, d \log^* \sigma_{\min}))$ on finite paths and rings with n nodes, and Remark 5.1 a faster bound when a start node is near an end of a finite path. Since $\sigma_{\min} \le \sigma_{\max}$ it is never slower than $O(d \log^* \sigma_{\max})$. It improves the $O(d^2(\log^* \sigma_{\max})^3)$ algorithm of (Miller and Pelc 2025) for unknown d, and needs no knowledge of d to match their $O(d \log^* \sigma_{\max})$ algorithm for known d.
- Miller and Pelc (2025) [also]: time $O(d \log^* \sigma_{\max})$. *Assumes:* knows d; unbounded memory. Theorem 4.2: optimal when d is known. If each agent also knows its position relative to the node with the smallest node label, $O(d)$ rounds suffice for any delay (Theorem 3.4).

GRAPHS

Notes. Bourreau, Narayanan and Nolin let each agent explore a growing interval around its start node and compute from the node labels it has seen a sparse set of nodes, a ruling set of the path with spacing $\Theta(d)$, coloured with 17 colours so that nearby members get different colours (Lemma 3.5). Each agent walks to the nearest member and alternates searching and waiting according to the bits of its colour, so that one agent searches while the other waits (Algorithm 5, Lemmas 4.3 to 4.7). The computation is an early-stopping version of distributed colouring in the LOCAL model, in which a node with a small node label decides early; this is why the bound depends on $\sigma_{\min}$. Procedure CarefulWalk crosses an edge in 4 rounds, with moves that depend on whether the node label increases across the edge, so that two agents can no longer swap places inside an edge without meeting (Theorem 3.1). Without node labels, labelled agents on a ring with simultaneous start need $\Theta(d \log \lambda)$ rounds (Dessmark et al. 2006); Bourreau et al. draw this contrast in their Table 1. Miller and Pelc left open the optimal time when d is unknown (Miller and Pelc 2025); Bourreau, Narayanan and Nolin settled it. Miller and Pelc also separate three ways of learning the node labels when d is known: $\Theta(d)$ if the whole labelling is known in advance, $\Theta(d \log^* \sigma_{\max})$ if node labels are seen on visits, and $\Theta(d \log \sigma_{\max})$ if only the start node's label is seen, which is the case of labelled agents in an anonymous graph. (Bourreau et al. 2026) read in arXiv v2 (January 2026) and (Miller and Pelc 2025) in arXiv v1 (2023); theorem numbers refer to these versions. Both bounds were checked against the full texts.

16.3.11 Labelled agents, graphs with infinite degrees

Adds infinite degrees allowed, + distinct IDs (eponymous), + synchronous VERIFIED

A universal algorithm works even when nodes have infinite degree; it is within a cubic factor of the lower bound.

Assumed: graph dynamics: static graph; edge direction: undirected; identifiers: distinct identifiers (eponymous): agents have pairwise distinct identifiers; timing: synchronous rounds. **Not assumed:** graph size: finite or countably infinite, degrees may be infinite; node identifiers: anonymous nodes; start: arbitrary delay between the two starts; topology: arbitrary connected graph.

Measure	Lower bound	Best upper bound	Status
Time	$\Omega(\max(W, \log L))$ (Pattanayak and Pelc 2024b) Cubic.	$O((W \log L)^3)$ (Pattanayak and Pelc 2024b)	• polynomial gap
Memory	none known Memory is unbounded.	—	• unknown

Results.

Claude (Anthropic) was used in cataloguing the problems.

- Pattanayak and Pelc (2024b) [lower bound]: time $\Omega(\max(W, \log L))$. *Bottleneck:* port addressing. *Holds when:* $W = \max(w(v_1, v_2), w(v_2, v_1))$, the path weight of the treasure-hunt variant; e.g. the infinite tree with all degrees infinite.
- Pattanayak and Pelc (2024b) [current best]: time $O((W \log L)^3)$. Introductory and current best (Theorem 3.1).

Open questions.

Q1. Find an optimal rendezvous algorithm, "even for finite graphs" (Pattanayak and Pelc 2024b).

16.4 Gathering in graphs

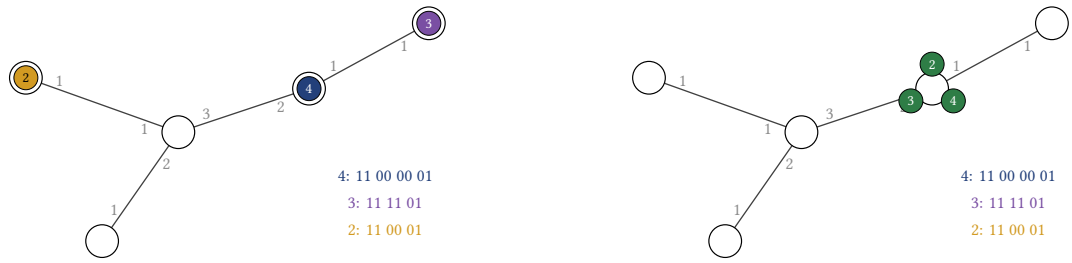
Several agents must all meet at one node of an unknown graph, possibly despite faulty agents.

First posed in 2012 (Dieudonné et al. 2014). Gathering many agents in arbitrary graphs grew out of two-agent rendezvous; the Byzantine version was posed at SODA 2012.

Problem statement and measures

$k \geq 2$ agents start at different nodes of an unknown connected graph and must all be at the same node at the same time, and (with **detection**) know it. **Time** is in rounds; **team** is the number of good agents needed when some agents are **Byzantine** (faulty agents that move arbitrarily and may lie).

Measures: **Time**, rounds until all agents meet; **Team**, good agents needed when f agents are Byzantine.



(a) Start: agents 2, 3, 4 at distinct nodes of a tree

(b) Final: leader 3 walks onto the waiting group of 4 (round 25)

Figure 16.4: Gathering of labelled agents by merging rendezvous (Dessmark et al. 2006): each leader runs the explore-or-wait rendezvous of a tree with its own label (after one tour to learn n, one block of $2(n-1)$ rounds per bit of its doubled label followed by 01: explore on 1, wait at its start node on 0); when groups meet, the largest label stays leader and the others follow it. Colours show the leader each agent follows; one frame per round. Gathering is reached, but without detection.

A first look

A group of hikers scattered over a trail network wants to get together at one spot, with no phones and no map. The network is a **graph**: trail junctions are **nodes** and the trails between them are **edges**; the hikers are $k \geq 2$ **agents** that start at different nodes of this unknown connected graph. At each node the edges at that node carry local numbers, its **ports**. Time passes in synchronous **rounds**, and in each round an agent stays or moves along one edge. All agents must be at the same node at the same time

Claude (Anthropic) was used in cataloguing the problems.

and, in the version **with detection**, know that everyone has arrived. In the basic version the agents are **anonymous** (identical: same algorithm, no **labels**, that is, no distinct integers that tell them apart), there are no faults, and n denotes the number of nodes; the question is which starting **configurations** (the sets of nodes on which the agents start) can be gathered, and how fast.

1. Algorithm: Meet in pairs, then follow the largest label. (Labelled agents, p. 136) A standard approach for agents with distinct labels that can read each other's label when they are at the same node. It uses any rendezvous algorithm for two labelled agents (an algorithm that makes two agents meet at one node) that works whatever the delay between their starts, such as the one of (Dessmark et al. 2006).

1. Every agent starts as the **leader** of a **group** containing only itself, and runs the rendezvous algorithm with its own label.
2. When groups meet at a node, everyone there adopts the largest label present; the agent with that label remains a leader and continues its own rendezvous algorithm as if nothing had happened.
3. Every other agent stops being a leader and from now on simply follows its leader's moves.

Why it works and what it costs. A leader's walk is exactly what it would do alone, so any two leaders are guaranteed to meet by the rendezvous algorithm, and at each such meeting one of them stops being a leader. The agent with the largest label is never absorbed, and every other leader meets its group at the latest when it meets that agent, so eventually all k agents move as one group. The time is at most $k-1$ rendezvous times. This gives gathering but not detection: knowing that nobody is missing needs extra knowledge, for example of n; with more than $n/2$ agents and n known, (Molla et al. 2023) gather with detection in $O(n^3)$ rounds.

Why it matters. Reducing gathering to repeated rendezvous with merging is the simplest way to see why distinct labels make gathering solvable.

2. Impossibility: Symmetric configurations of identical agents. (Dieudonné and Pelc 2016) (Anonymous agents, p. 135) Take a ring of 6 nodes, with port 0 leading clockwise and port 1 counterclockwise at every node, and put three identical agents on every second node. Starting together, each agent sees exactly what the others see, so in every round all three make the same choice. Turning the ring by two nodes maps the configuration to itself before the round and after it.

Why it holds. So the agents always stay on three different nodes, two edges apart, and never gather, whatever the algorithm. For identical agents the question is therefore which configurations can be gathered at all; (Dieudonné and Pelc 2016) characterise them and gather every such configuration in time polynomial in n, but detecting that gathering is complete needs an upper bound on n.

Why it matters. It shows that for identical agents the right question is which starting configurations are solvable, not only how fast.

3. Impossibility: Byzantine agents can impersonate the whole team. (Dieudonné et al. 2014) (Byzantine agents, p. 138) Some agents may be **Byzantine**: faulty agents that move arbitrarily and, in the strong version, may lie about their labels. Take $f = 2$ good agents, labelled 0 and 1, and 2 Byzantine ones, on a ring of 4 nodes v_0, v_1, v_2, v_3 with port 0 leading clockwise everywhere. **Run 1:** good agents 0 and 1 on v_0, v_1; the two Byzantine agents on v_2, v_3 claim labels 0 and 1 and follow the algorithm honestly. The configuration is symmetric, so when the good agents gather at some node v_i and stop, the Byzantine agents stop at the opposite node v_{i+2}. **Run 2:** the same, except good agent 0 and the Byzantine agent claiming label 0 swap starting nodes.

Why it holds. Every agent sees in run 2 exactly what it saw in run 1, so everyone does the same thing; at the end good agent 1 stands with the Byzantine agent claiming label 0 on one node and good agent 0 stands with the Byzantine agent claiming label 1 on the opposite node, and both groups stop. The good

Claude (Anthropic) was used in cataloguing the problems.

agents are not gathered, so with f strongly Byzantine agents, f good agents cannot gather, even if n is known (Dieudonné et al. 2014) (their Theorem 4.7, for every f, on a ring of $2f$ nodes). With $f+1$ good agents (n known) or $f+2$ (n unknown) gathering becomes possible (Bouchard et al. 2016).

Why it matters. This "two indistinguishable runs" argument is the standard way to prove that faulty agents make a task impossible, and it shows how many good agents a fault-tolerant algorithm must assume.

What the community started looking at

Gathering of many agents in graphs was studied as the natural extension of rendezvous (Pelc 2019): first **which configurations can gather** at all (Dieudonné and Pelc 2016), then **how many faulty agents** a team can tolerate (Dieudonné et al. 2014; Bouchard et al. 2016), and only then time (Bouchard et al. 2022; Molla et al. 2023).

Further reading

Papers on this problem that are not yet placed in a setting.

- Gao and Pelc (2026). Labelled agents that are finite automata (their number of states depends only on L) start in the same round in teams, all agents of a team on one node, on an infinite path without node names, and must gather and stop: with teams of one agent the adversary can prevent every meeting even when the two ports of every edge differ (Theorem 3.1); with equal teams of at least two agents, gathering takes time linear in the largest distance between the starting nodes of two teams when the two ports of every edge differ (Theorem 4.1), and with arbitrary ports $\Theta(\log L)$ times that distance for teams of two (Theorems 5.1, 5.2), linear time for larger teams (Theorem 5.3) and for teams of different sizes (Theorems 6.1, 6.2); the agents know L and the number of teams, which Section 7 argues is necessary, as is the simultaneous start. Read (arXiv v1).
- Bärtschi et al. (2021). Near-gathering of k agents with energy budgets in an edge-weighted graph, where gathering at one point may be impossible: minimising the radius of a ball containing all agents is polynomial; minimising the largest distance between two agents has a 2-approximation and a matching NP-hardness bound; minimising the average distance has a $2(1-1/k)$-approximation but no FPTAS unless P = NP. Abstract read.
- Pelc (2018c). Gathering on a grid by copies of a state machine: with initial input, unbounded memory and randomization every configuration is gathered with probability 1, and no two of these suffice; deterministic machines gather all connected configurations. Abstract read.
- Hirose et al. (2024). Weakly Byzantine gathering in synchronous anonymous networks without whiteboards: $O(f\,|\Lambda_{good}|\,X(N))$ rounds with $N \geq n$ known, simultaneous start and $k \geq 9f+8$, by simulating Byzantine consensus; $|\Lambda_{good}|$ is the length of the largest good identifier. Abstract and Table 1 read.
- Tsuchida et al. (2018). Gathering of k agents with unique identifiers in asynchronous arbitrary graphs with authenticated whiteboards, despite f weakly Byzantine agents: two algorithms, with and without termination detection. Abstract read.
- Miller and Saha (2020). Gathering of m synchronous robots in a graph despite f Byzantine robots that may change their identifiers, when robots can see further than their own node. Abstract read.
- Das et al. (2019b). Gathering on a ring despite a fast malicious agent that blocks paths: for synchronous agents more cases are solvable on unoriented rings than for asynchronous ones. Abstract of the ICTCS 2016 version read.
- Shibata et al. (2016). Partial gathering (g agents per meeting node suffice) in asynchronous unidirectional rings with whiteboards: three algorithms and their move complexity. Abstract read.
- Shibata et al. (2019). Partial gathering in arbitrary graphs. Abstract read.
- Bouchard (2019). A PhD thesis on deterministic gathering of mobile agents. Abstract read.
- Di Luna et al. (2020c). Gathering in dynamic rings in which one edge may be missing at each time: feasibility and complexity by chirality and cross detection. Abstract read.

Claude (Anthropic) was used in cataloguing the problems.

- Bournat et al. (2018). Gracefully degrading gathering in dynamic rings: gather when the dynamics allow, and solve a weaker task when they do not. Abstract read.
- Shibata et al. (2026). Partial gathering (g agents per meeting node) in synchronous dynamic bidirectional rings. Abstract read.
- Das et al. (2021a). Oblivious agents on dynamic rings must form a compact block. Abstract read.
- Hirose et al. (2022). Weakly Byzantine gathering of agents with unique identifiers in synchronous anonymous networks without whiteboards, with a large enough team of good agents ('strong team'), faster than the general algorithm. Abstract read.
- Saxena and Mondal (2024). Weakly Byzantine gathering of agents with unique labels in synchronous anonymous networks, with delayed starts, when the agents know n but not f, k or the range of labels: $f^2 + 4f + 9$ good agents gather and stop in the same round within $O(k^2\, X(n))$ rounds times the length of the largest label of a good agent (Theorem 4.1), where $X(n)$ is the time to explore any n-node graph; Hirose et al. needed $4f^2 + 8f + 4$ good agents. Read in full (ICDCN version).
- Shibata et al. (2020). Partial gathering (g agents or none on every node at the end, $2 \le g \le k/2$) in asynchronous unidirectional rings with whiteboards, by anonymous agents that know g but neither k nor n: the unsolvable starts are those for agents that know k, periodic configurations whose period is below g, and no agent can detect them (Theorem 2); from every other start $O(gn)$ moves in total (Theorem 3), which is optimal since $\Omega(gn)$ are needed (Theorem 1, from Shibata et al. 2016); a randomized algorithm solves every start with probability 1 in $O(gn)$ expected moves (Theorem 4). Read in full.
- Das et al. (2019a). Compacting instead of gathering on a 1-interval connected unoriented ring: synchronous Look–Compute–Move agents on distinct nodes must occupy consecutive nodes without ever sharing one, and with several colours each colour must form its own segment. Impossible from periodic starts even with global vision (Theorem 1), and with local vision even from asymmetric starts (Theorem 2). With global vision: from asymmetric starts within n rounds (Theorem 3), and from symmetric starts exactly when the axis does not pass through two empty nodes (Theorems 4, 5); the coloured version is treated in the same way, for two colours and for more (Sections 4 and 5). Read (model and theorems).

Relations to other problems

- **Generalisation of Rendezvous.** Rendezvous is gathering of two agents; with more agents the questions become which teams can gather, and how many faulty agents a team tolerates.
- **Derived problem: Gathering by oblivious robots on graphs** (variant of). The task is that of gathering in graphs, all entities on one node, but the entities are oblivious Look–Compute–Move robots, as in exploration by oblivious robots: they see the positions of all robots, remember nothing, carry no labels and send no messages. Whether a start can be gathered then depends on the symmetries of the configuration, on what a robot sees of robots sharing a node, and on the scheduler. Results do not transfer directly in either direction.
- **Related: Broadcast by agents in dynamic graphs.** Every ignorant agent must meet a source agent, so any algorithm that gathers all agents on one node in a setting also solves broadcast there; broadcast asks only for the meetings that pass the message on.
- **Related: Deciding graph properties by mobile agents.** Gathering is what makes a known number of agents decisive: after it, the agents learn the whole configuration (Fraigniaud and Pelc 2017).

Claude (Anthropic) was used in cataloguing the problems.

Case tree

The root assumes nothing; going down, each edge adds the assumptions written in grey (+) or leaves an axis unassumed. Bold entries are catalogued settings with their status per measure (Time, Team) and page; italic entries are branch points, named after the axis on which their branches differ. Axis order: Graph dynamics → Edge direction → Faults → Identifiers → Timing → Topology.

No assumptions

+ static graph, + undirected

Faults

+ no faults

Identifiers

anonymous, + synchronous

Anonymous agents STANDARD SETTING ●● p. 135

+ distinct IDs (eponymous), + synchronous

Labelled agents ●● p. 136

+ homonymous, + synchronous

Homonymous agents ●● p. 136

+ at most weak Byzantine faults, + distinct IDs (eponymous), + synchronous

Weakly Byzantine agents ●● p. 137

any faults, + distinct IDs (eponymous), + synchronous

Byzantine agents ●● p. 138

What is known

The best upper bound against the best lower bound in each catalogued setting, in tree order. The settings follow in detail.

Setting	Time	Team	p.
Anonymous agents STANDARD SETTING	● unknown $\mathrm{poly}(n)$ vs *impossible*	● unknown — vs —	135
Labelled agents	● unknown $O(n^3)$ vs —	● unknown — vs —	136
Homonymous agents	● unknown $\mathrm{poly}(n, \log \lambda)$ vs —	● unknown — vs —	136
Weakly Byzantine agents	● unknown $\mathrm{poly}(n, \log L)$ vs —	● tight $f+2$ vs $f+2$	137
Byzantine agents	● unknown $\mathrm{poly}(n, \log \lambda)$ vs —	● tight $f+1$ vs $f+1$	138

16.4.1 Anonymous agents

Adds anonymous, + synchronous STANDARD SETTING VERIFIED

Which configurations of identical agents can gather is characterised; detecting that gathering is complete needs a bound on n.

Assumed: graph dynamics: static graph; edge direction: undirected; faults: no faults; timing: synchronous rounds. **Not assumed:** identifiers: anonymous: all agents are identical; topology: arbitrary connected graph.

Claude (Anthropic) was used in cataloguing the problems.

Measure	Lower bound	Best upper bound	Status
Time	*impossible* (Dieudonné and Pelc 2016) Polynomial; exponent not tracked.	poly(n) (Dieudonné and Pelc 2016)	• unknown
Team	none known Not a team-size question here.	—	• unknown

Results.

- Dieudonné and Pelc (2016) [lower bound]: time *impossible*. *Bottleneck:* symmetry. *Holds when:* gathering with detection by a universal algorithm when no bound on n is known.
- Dieudonné and Pelc (2016) [current best]: time poly(n). *Assumes:* an upper bound on n (for detection). Introductory and current best; without a bound, gathering without detection in time polynomial in the actual size.

16.4.2 Labelled agents

Adds + distinct IDs (eponymous), + synchronous VERIFIED

Labelled agents gather even without talking; with many agents gathering with detection takes $O(n^3)$ rounds.

Assumed: graph dynamics: static graph; edge direction: undirected; faults: no faults; identifiers: distinct identifiers (eponymous): agents have pairwise distinct identifiers; timing: synchronous rounds. **Not assumed:** topology: arbitrary connected graph.

Measure	Lower bound	Best upper bound	Status
Time	none known No lower bound recorded.	$O(n^3)$ (Molla et al. 2023)	• unknown
Team	none known —	—	• unknown

Results.

- Bouchard et al. (2023b) [also]: time *finite*. Gathering with no communication at all.
- Molla et al. (2023) [current best]: time $O(n^3)$. *Assumes:* more than $n/2$ agents; n known. Gathering with detection.

Open questions.

Q1. Gathering without communication when no bound on n is known.

Notes. Two-agent gathering is rendezvous. Look–Compute–Move robots on graphs, starting with rings (Klasing et al. 2008), are treated in the problem *Gathering by oblivious robots on graphs*.

16.4.3 Homonymous agents

Adds + homonymous, + synchronous VERIFIED

When agents may share labels, the teams that can always gather are fully characterised, and gather in time polynomial in n and $\log \lambda$.

Assumed: graph dynamics: static graph; edge direction: undirected; faults: no faults; identifiers: homonymous: agents carry labels, but several may share a label; timing: synchronous rounds. **Not assumed:** topology: arbitrary connected graph.

Claude (Anthropic) was used in cataloguing the problems.

Measure	Lower bound	Best upper bound	Status
Time	none known —	poly(n, log λ) (Devismes et al. 2026b)	● unknown
Team	none known —	—	● unknown

Results.

- Devismes et al. (2026b) [current best]: time poly(n, log λ). *Assumes:* $O(\log\log\log L)$ bits of common knowledge. Introductory and current best; λ is the smallest label, L the largest.

16.4.4 Weakly Byzantine agents

Adds + at most weak Byzantine faults, + distinct IDs (eponymous), + synchronous VERIFIED

With f weakly Byzantine agents and f known, one good agent suffices when n is known and $f+2$ are necessary and sufficient when it is not; when nothing is known, no team can gather.

Assumed: graph dynamics: static graph; edge direction: undirected; faults: up to f weak Byzantine agents (arbitrary behaviour, cannot fake identifiers); identifiers: distinct identifiers (eponymous): agents have pairwise distinct identifiers; timing: synchronous rounds. **Not assumed:** topology: arbitrary connected graph.

Measure	Lower bound	Best upper bound	Status
Time	none known No lower bound on time beyond fault-free gathering is recorded.	poly(n, log L) (Dieudonné et al. 2014)	● unknown
Team	$f+2$ (Dieudonné et al. 2014); *impossible* (Hirose et al. 2025) $f+2$ for unknown n; 1 for known n; both with f known. With nothing known, no team gathers (Hirose et al. 2025).	$f+2$ (Dieudonné et al. 2014)	● tight

Results.

- Dieudonné et al. (2014) [lower bound]: team $f+2$. *Bottleneck:* indistinguishable Byzantine agents. *Holds when:* n unknown; Theorem 3.11: $f+1$ good agents cannot gather deterministically with termination in some networks.
- Hirose et al. (2025) [lower bound]: team *impossible*. *Bottleneck:* deciding when to stop. *Holds when:* no global knowledge: none of n, k, f and the length of the largest label of a good agent is known (Theorem 3.1); holds for every number of good agents; since f is unknown, a good agent may be the only one, so it must stop even if it meets no good agent; a single agent cannot visit every node of a ring of unknown size and stop (Lemma 3.1), and two good agents that each stop before visiting some node are placed in two graphs joined by an edge between those nodes, so they never meet.
- Dieudonné et al. (2014) [current best]: time poly(n, log L), team $f+2$. *Assumes:* f known to all agents. Introductory and current best (Theorem 3.10, unknown n); time polynomial in n and in the length of the largest label of a good agent. With known n, any number of good agents gather within $4n^4 \cdot P(n, \log L)$ rounds (Theorem 3.6; P is the polynomial time bound of the rendezvous algorithm of Ta-Shma and Zwick Ta-Shma and Zwick 2014).

Open questions.

Q1. Can weakly Byzantine agents solve self-stabilizing perpetual gathering (from arbitrary memory states, all good agents eventually always at one node) with no global knowledge? (Hirose et al. 2025) solve it with upper bounds on n, k and f and the length of the largest label of a good agent, and leave this open.

Notes. Checked against the full text of (Dieudonné et al. 2014). Dieudonné, Pelc and Peleg note that $f+2$ is a new threshold for tolerating f Byzantine entities, next to $f+1$ for communication through a

Claude (Anthropic) was used in cataloguing the problems.

cut and $2f + 1$ processors for consensus in a complete network.

Hirose, Eguchi and Sudo (Hirose et al. 2025) show that the algorithms of (Dieudonné et al. 2014) need what they know: with none of n, k, f and the length of the largest label of a good agent known, no number of good agents can gather and stop (Theorem 3.1). Their agents are those of (Dieudonné et al. 2014): unique labels, no marks on nodes, agents at the same node read each other's state and a Byzantine agent shows the same state to all of them, and the adversary wakes agents at different times. They also study **self-stabilizing** gathering, where every agent starts in an arbitrary, possibly corrupted, memory state: it is impossible when $k \leq 2f$, even with n, k, f and the length of the largest label of a good agent known (Theorem 3.2). For the relaxed goal of **perpetual gathering**, in which from some round on all good agents are always at the same node (they may keep moving together and never stop), they give a self-stabilizing algorithm for every k and f that knows upper bounds on n (namely N), k and f and the length of the largest label of a good agent; its time is proportional to the product of the bounds on k and f times the time of the rendezvous algorithm of Ta-Shma and Zwick (Ta-Shma and Zwick 2014) for labels of that length, that is $\tilde{O}(N^5)$ times that length (Theorem 4.1; Section 5 removes the simultaneous start). Without self-stabilization, perpetual gathering needs no knowledge: removing termination from the algorithm of (Dieudonné et al. 2014) suffices, as observed in (Hirose et al. 2025). Checked against the full text of (Hirose et al. 2025) (arXiv v1).

16.4.5 Byzantine agents

Adds any faults, + distinct IDs (eponymous), + synchronous VERIFIED

With f strongly Byzantine agents, $f + 1$ good agents (known n) or $f + 2$ (unknown n) are necessary and sufficient.

Assumed: graph dynamics: static graph; edge direction: undirected; identifiers: distinct identifiers (eponymous): agents have pairwise distinct identifiers; timing: synchronous rounds. **Not assumed:** faults: up to f strong Byzantine agents (arbitrary behaviour, can fake identifiers); topology: arbitrary connected graph.

Measure	Lower bound	Best upper bound	Status
Time	none known Polynomial only for large teams.	poly(n, log λ) (Bouchard et al. 2022)	● unknown
Team	$f + 1$ (Dieudonné et al. 2014); $f + 2$ (Dieudonné et al. 2014) $f + 1$ / $f + 2$.	$f + 1$ (Bouchard et al. 2016)	● tight

Results.

- Dieudonné et al. (2014) [lower bound]: team $f + 1$. *Bottleneck:* indistinguishable Byzantine agents. *Holds when:* Theorem 4.7: f good agents cannot gather deterministically against f strongly Byzantine agents; holds even in a ring of known size.
- Dieudonné et al. (2014) [lower bound]: team $f + 2$. *Bottleneck:* indistinguishable Byzantine agents. *Holds when:* n unknown; inherited from weakly Byzantine agents (Theorem 3.11).
- Dieudonné et al. (2014) [introductory]: team $2f + 1$. *Assumes:* f known to all agents. Introductory (Theorems 4.5 and 4.6): $2f + 1$ good agents with known n, $4f + 2$ with unknown n; deterministic but exponential time. For known n and constant f a randomized algorithm gathers $2f + 1$ good agents in polynomial time with probability at least $1 - 1/n$ (Theorem 4.4).
- Bouchard et al. (2016) [current best]: team $f + 1$. $f + 2$ when n is unknown; time exponential.
- Bouchard et al. (2022) [also]: time poly(n, log λ). *Assumes:* good agents at least quadratic in f; $O(\log \log \log n)$ bits of global knowledge. First polynomial-time Byzantine gathering. λ is the smallest label of a good agent (the paper's $|l_{min}|$ is log λ here).

Claude (Anthropic) was used in cataloguing the problems.

Open questions.

Q1. Polynomial-time Byzantine gathering with teams of size close to $f + 1$.

Notes. Checked against the full text of (Dieudonné et al. 2014) for its bounds; the agents cannot mark nodes, agents at the same node exchange all information, the adversary may wake agents at different times, and good agents must gather and terminate simultaneously. The paper's results for known n also hold with a known polynomial upper bound on n.



16.5 Gathering by oblivious robots on graphs

Oblivious Look–Compute–Move robots must all reach one node of a graph and stay there; which starts can be gathered depends on the symmetries of the graph, on multiplicity detection and on the scheduler.

First posed in 2006 (Klasing et al. 2008). Gathering of asynchronous oblivious robots on a ring (ISAAC 2006).

Problem statement and measures

$k \geq 2$ anonymous, oblivious robots stand on the nodes of a connected anonymous graph G with n nodes, which every robot sees whole. A robot runs Look–Compute–Move cycles: in a Look it takes a snapshot of the graph with its occupied nodes, it computes from that snapshot only, and it stays or moves to a neighbour; moves are instantaneous, so robots are always seen on nodes. A node with two or more robots is a **tower**. With **multiplicity detection** a snapshot shows which nodes are towers (weak) or how many robots each node holds (strong); without it a tower looks like one robot. The robots must reach a configuration in which all of them are on one node and none moves again. A setting fixes the scheduler, the multiplicity detection, whether towers may exist at the start, and the graph class; an algorithm must gather from every start of the setting except the starts listed as unsolvable. Under **round robin** the scheduler activates one robot at a time in a fixed cyclic order. **Time** is counted in epochs: an epoch is an interval in which every robot completes at least one cycle (under round robin, k consecutive activations).

Measures: **Time**, epochs until all robots are on one node and no robot moves again.

A first look

Imagine identical toy trains on a rail network that must all meet at one station, although none of them remembers anything and none can tell how many trains already stand at a station. The network is a **graph**: stations are **nodes**, and two nodes joined by an edge are **neighbours**. The $k \geq 2$ trains are **robots**. They are **anonymous** (the robots are identical and the nodes carry no names) and **oblivious** (a robot keeps nothing from one activation to the next). Each activation is a **Look–Compute–Move cycle**. The robot takes a **snapshot** of the whole graph showing which nodes are occupied, computes from that snapshot alone, and then stays or moves to a neighbour. A node with two or more robots is a **tower**. With **multiplicity detection** a snapshot shows the towers; without it a tower looks like a single robot. The **scheduler**, an adversary, decides when each robot runs its cycle. The robots must reach a **configuration** (the positions of all of them) in which they all stand on one node, and then never move again. In the standard version the graph is a ring and the robots cannot see towers. Towers may already exist at the start, and the scheduler is **round robin**: it activates one robot at a time, in a cyclic

Claude (Anthropic) was used in cataloguing the problems.



(a) Start: two centres; the left half has fewer occupied nodes (2 against 3)

(b) Final: all robots on one node

Figure 16.5: Gathering on a tree by asynchronous oblivious robots that cannot see towers, the algorithm of (D'Angelo et al. 2016): this tree has two centres, c_1 and c_2, and removing the edge between them (dashed) leaves two halves. The robots of the half with fewer occupied nodes walk towards the other centre while the others wait; when that half is empty, every robot walks to c_2. A tower looks like a single robot, which only lowers the count of the half that is emptying, so the robots never change their choice. One Move per frame, in an order chosen by the scheduler.

order that it fixes once and for all. Time is counted in **epochs**, intervals in which every robot completes at least one cycle.

1. Algorithm: On a tree, walk to the centre. (D'Angelo et al. 2016) (Trees, ASYNC, no multiplicity detection, p. 155) A tree is a graph without cycles. A **centre** of a tree is a node whose largest distance to the other nodes is as small as possible; a tree has one centre or two neighbouring centres. The scheduler is asynchronous: Look, Compute and Move take finite but unpredictable times, so a robot may move on the basis of an old snapshot. The robots start on distinct nodes and cannot see towers.

1. If the tree has one centre, every robot walks to it, one edge per cycle.
2. If it has two centres, remove the edge between them. This splits the tree into two halves, each holding one centre.
3. If one half has fewer occupied nodes, its robots walk to the centre of the other half, and the robots of the other half wait.
4. When that half is empty, every robot walks to the remaining centre.
5. If the halves have equal numbers of occupied nodes, a fixed way of writing each half as a string, occupied nodes included, picks the half that moves first.

Why it works and what it costs. Each rule depends only on the tree and on which nodes are occupied, so towers that look like single robots do no harm (D'Angelo et al. 2016). While a half empties, its number of occupied nodes can only fall and the other half's can only rise, so the robots never change their choice, even when they act on an old snapshot. The robots gather from every start except those in which the two halves are mirror images of each other, robots included (Theorem 6). There the scheduler can activate two mirror-image robots together forever, and the configuration stays symmetric. The paper does not analyse the time.

Why it matters. It shows the method of this problem in its simplest form: the robots agree on a node that the snapshot determines, and the only obstacle is a symmetry that the scheduler can preserve.

2. Impossibility: Two robots at odd distance never meet. (Klasing et al. 2008) (Rings, SSYNC, no multiplicity detection, p. 149) Put two robots on a ring (a cycle of nodes) at odd distance, and let the scheduler activate both of them in every round.

Claude (Anthropic) was used in cataloguing the problems.

1. Two robots on a ring see the same snapshot, up to a reflection, so they make the same decision: both stay or both move.
2. Fix one of the two arcs of the ring between them and count its edges. A move of one robot changes that count by one, so when both move it changes by two or not at all.
3. The count starts odd, so it stays odd and never reaches 0.

Why it holds. So two robots never gather on a ring, whatever the algorithm (Klasing et al. 2008). Without multiplicity detection the same holds for any number of robots. Let a single robot create the first tower alone. From then on every snapshot looks like one with fewer robots, and by induction they never gather. This is why Klasing, Markou and Pelc assume multiplicity detection.

Why it matters. It is the basic obstacle on rings: what the robots cannot see or tell apart, the scheduler can keep symmetric.

3. Impossibility: One robot at a time is not enough unless the order is fixed. (Navarra and Piselli 2025) (Rings, sequential scheduler, no multiplicity detection, p. 147) Let the scheduler activate one robot at a time but choose the order freely, on a ring with robots that cannot see towers. Just before gathering, all robots stand on two neighbouring nodes. A snapshot shows only two occupied neighbours, so the only useful move is onto the other occupied node (Lemma III.1).

1. Take robots r_1 and r_2 on node u and r_3 on node v.
2. The scheduler activates r_1, which moves to v.
3. It activates r_3, which moves back to u; then r_2, which moves to v.
4. It goes on like this, always taking a robot from the node that holds more, so neither node ever empties, and every robot is activated again and again.

Why it holds. So with such a scheduler $k \geq 3$ robots that cannot see towers never gather on a ring (Theorem III.1) (Navarra and Piselli 2025). Round robin removes the scheduler's freedom: the order is fixed once and for all. Navarra and Piselli first make the robots choose an empty gathering node between two groups of robots, then move both groups next to it and finally onto it. Every robot is activated once before any robot is activated again, so the last step works. Their algorithm gathers from every start except a few listed ones, in at most $n - 3$ epochs, which is optimal up to a constant factor.

Why it matters. It shows how much the scheduler matters for robots that cannot count: a fixed order of activations replaces multiplicity detection.

What the community started looking at

Klasing, Markou and Pelc posed the problem on rings for asynchronous robots and showed that it needs multiplicity detection: without it, $k \geq 2$ robots cannot gather on any ring; with weak detection an odd number of robots gathers exactly from the starts that are not periodic (Klasing et al. 2008). Later work kept these assumptions (no towers at the start, multiplicity detection) and solved one graph class at a time, as listed in (Cicerone et al. 2025d). A recent series drops multiplicity detection and allows towers at the start, and makes up for it with round robin: rings (Navarra and Piselli 2025), graphs that are not vertex-transitive (Cicerone et al. 2025d), and hypercubes and infinite grids (Cicerone et al. 2025b).

Further reading

Papers on this problem that are not yet placed in a setting.

Claude (Anthropic) was used in cataloguing the problems.

- Maiti et al. (2026e). Asynchronous oblivious robots on the infinite grid, with no common orientation or chirality, global visibility and global strong multiplicity detection, starting on distinct nodes, must gather despite up to $k-2$ crash faults at a meeting node (one of finitely many designated nodes) that minimises the sum of distances from the initial positions (their n is k here): impossible from some symmetric starts when the robots on the axis or at the centre of symmetry crash (Theorem 1) and, by the authors' argument, with only weak multiplicity detection even under FSYNC (Theorem 2); an algorithm gathers the robots that do not crash from every other start in finite time (Theorem 3); time is not analysed. Preprint. Read.
- Maiti et al. (2026d). Surplus parking gathering on the infinite grid: asynchronous oblivious disoriented robots with global strong multiplicity detection, starting on distinct nodes, must fill designated parking nodes exactly to given capacities and gather all other robots at one node; unsolvable starts, mostly adapted from parking and meeting-node gathering (Theorems 1–5, Corollaries 1, 2), an algorithm for every other start (Theorem 6), and a number of moves of the order of k times the perimeter of the smallest rectangle enclosing the robots, plus k^2, against a worst case of k times that perimeter (Theorems 7, 8; their n is k here). Preprint. Read.
- Kamei et al. (2025). Gathering on rings by myopic ASYNC robots with lights: feasible with three or four colours whenever one of two parameters of the initial configuration is odd. Abstract read.
- Goswami et al. (2025). Oblivious robots on the infinite triangular grid that see only adjacent nodes: gathering is impossible without agreement on an axis, even under FSYNC (Theorem 1); every algorithm takes $\Omega(k)$ epochs (Theorem 3; their n is k here); with one axis agreed and SSYNC, $O(k)$ epochs suffice (Theorem 4). Read (arXiv v4).
- Doan et al. (2017). A Maude model of oblivious ASYNC robots on rings with multiplicity detection; model checking the gathering algorithm of D'Angelo, Di Stefano and Navarra (Distributed Computing 2014) finds counterexamples in which the robots deadlock. Read (LIPIcs version).
- Gupta and Kulkarni (2024). The triangular-grid gathering algorithm of Goswami et al. (Goswami et al. 2025) is lattice-linear, so it also works under ASYNC and with robots that see in one direction only. Read (arXiv v2).
- Bhagat et al. (2022). Asynchronous oblivious robots in the infinite grid must gather at one of some marked meeting nodes: unsolvable configurations characterised, an algorithm for the others, and a lower bound on the number of moves. Abstract read.
- Goswami et al. (2023). Two fully synchronous oblivious robots entering a finite unoriented grid through a corner must meet at a resource that moves until a robot reaches it: an algorithm, a time lower bound, and impossibility under a semi-synchronous scheduler. A brief announcement; abstract read.
- Bose et al. (2017). Crash-tolerant gathering on a finite grid by ASYNC oblivious robots with weak multiplicity detection. Abstract read.
- Kamei et al. (2021). Gathering on a torus by ASYNC oblivious robots with local weak multiplicity detection. Abstract read.
- Castenow et al. (2020). Gathering on the grid by oblivious robots with constant viewing radius and no compass, faster than the $O(n^2)$ rounds of the Euclidean algorithm. Abstract read.
- Cord-Landwehr et al. (2016). Local gathering of a connected swarm on the grid into a 2×2 area in $O(n)$ FSYNC rounds, which is optimal, with constant vision and no compass. Abstract read.
- Abshoff et al. (2016). A closed chain of robots on the grid, which must stay connected, is gathered into a 2×2 area with local vision in FSYNC. Abstract read.
- Chakraborty et al. (2024). Gathering on the infinite grid while minimising the largest distance any robot travels: unsolvable configurations characterised and an algorithm for the rest. Abstract read.
- Shibata et al. (2022b). Gathering of seven robots on a triangular grid into a hexagon: the visibility range needed. Abstract read.
- Bramas et al. (2024a). Stand-up indulgent gathering on rings: if robots crash at one node, the others must gather there. Abstract read.
- D'Angelo et al. (2016). A unified approach to gathering and exclusive searching on rings by ASYNC oblivious robots, for more initial configurations than earlier work. Abstract read.

- Cicerone et al. (2025e). Gathering on butterfly networks by FSYNC robots with weak multiplicity detection (global), starting on distinct nodes. A placement-based leader election works exactly in the leader configurations, where one robot can be told apart, through an embedding of the butterfly in the plane; from every such start the robots gather in $\Theta(\log n)$ epochs on a butterfly with n nodes (Theorem 1; their $\Theta(d)$, d the dimension), which is optimal. Read in full.

- Ooshita and Tixeuil (2015). Self-stabilizing gathering on rings, from every start, towers included. No randomized algorithm gathers from every such start in ASYNC, even with strong multiplicity detection on an oriented ring (Theorem 1), nor in SSYNC with only local strong and global weak detection (Theorem 2); so the usual assumption that robots start on distinct nodes is a strong one. With global strong detection, a randomized SSYNC algorithm gathers in $O(n \log k)$ expected rounds and $O(kn)$ expected moves (Theorem 3), and also orients the ring and forms sets of nodes (Theorems 4, 5). Read (model and theorems).

Relations to other problems

- **Variant of Gathering in graphs.** The task is that of gathering in graphs, all entities on one node, but the entities are oblivious Look–Compute–Move robots, as in exploration by oblivious robots: they see the positions of all robots, remember nothing, carry no labels and send no messages. Whether a start can be gathered then depends on the symmetries of the configuration, on what a robot sees of robots sharing a node, and on the scheduler. Results do not transfer directly in either direction.

- **Related: Gathering.** The same robots moving freely in the plane. On a graph a robot moves one edge per cycle, and the symmetries of the graph add to those of the configuration.

- **Related: Exploration by oblivious robots.** The same robots and schedulers on graphs, with exploration as the task.

Assumptions specific to this problem

Scheduler

The schedulers come from robots in the plane (see the plane model). On graphs gathering was posed under ASYNC (Klasing et al. 2008). A sequential scheduler activates one robot at a time, so no two robots with the same snapshot move together; it was used for universal pattern formation in the plane, as reported by (Navarra and Piselli 2025), and on rings, where a general sequential scheduler still defeats every algorithm for $k \geq 3$ robots without multiplicity detection, while round robin does not (Navarra and Piselli 2025).

Claude (Anthropic) was used in cataloguing the problems.

Level	Value	Formal definition
0	**ASYNC**	**ASYNC**: each robot's Look, Compute and Move phases take finite but unbounded, adversarially chosen time; a robot may Look while others are between phases, so it may move on an outdated snapshot. A robot is always seen on a node (moves are instantaneous). The scheduler is fair: every robot is activated infinitely often.
1	**SSYNC**	**SSYNC**: time is divided into rounds; in each round a fair adversary chooses a non-empty subset of robots, which execute one atomic Look–Compute–Move cycle together.
2	**sequential**	**Sequential**: an SSYNC scheduler that activates exactly one robot per round, fairly; which robot is the adversary's choice unless the activation order is fixed (see the order axis).
2	**FSYNC**	**FSYNC**: in every round every robot executes one atomic Look–Compute–Move cycle.

Activation order

Navarra and Piselli took round robin from work on distinct gathering in the plane (as they report) to rings, where it separates the solvable from the unsolvable: under a general sequential scheduler gathering is impossible for $k \geq 3$ robots without multiplicity detection, under round robin it is characterised (Navarra and Piselli 2025). Cicerone, Di Fonso, Di Stefano and Navarra call it favourable, since it breaks every symmetry by activating a single robot, and use it to compensate for the loss of multiplicity detection (Cicerone et al. 2025b); their algorithms for graphs that are not vertex-transitive also work if the order changes in every epoch (Cicerone et al. 2025d). The order is a separate axis because round robin is a special case of the sequential scheduler only, and says nothing about FSYNC.

Level	Value	Formal definition
0	**any order**	The adversary chooses which robots to activate, subject only to the scheduler's fairness.
1	**round robin**	With the sequential scheduler: the robots are activated one per round in a cyclic order that the adversary fixes at the start, that the robots do not know, and that repeats forever; an epoch is k consecutive rounds, one activation per robot.

Multiplicity detection

Klasing, Markou and Pelc proved that without multiplicity detection $k \geq 2$ robots cannot gather on any ring, and assumed the weak form (Klasing et al. 2008). Cicerone, Di Fonso, Di Stefano and Navarra note that most work assumes some multiplicity detection and starts without towers, and drop both under round robin (Cicerone et al. 2025b,d). Strong detection is used where robots must count, as in gathering at a meeting node that minimises the total distance despite crashes (Maiti et al. 2026e). Local forms, in which a robot sees only the multiplicity of its own node, are stated per result.

Level	Value	Formal definition
0	**no multiplicity detection**	A snapshot shows which nodes are occupied, not how many robots each holds.
1	**weak detection**	A snapshot marks every occupied node as holding one robot or more than one.
2	**strong detection**	A snapshot gives the number of robots on every node.

Claude (Anthropic) was used in cataloguing the problems.

Network

Gathering was solved graph class by graph class: rings first (Klasing et al. 2008), then trees, finite and infinite grids, tori, hypercubes, complete and complete bipartite graphs and butterflies, as listed in (Cicerone et al. 2025d). On finite grids and trees no multiplicity detection is needed: in ASYNC every start can be gathered unless a symmetry of the graph and the robots forbids it (D'Angelo et al. 2016). Classes matter through their symmetries: the more automorphisms a graph has, the fewer nodes robots can tell apart. Under round robin without multiplicity detection, every start on a graph that is not vertex-transitive can be gathered (Cicerone et al. 2025d), vertex- and edge-transitive graphs need algorithms specific to the class, and complete graphs cannot be gathered at all (Cicerone et al. 2025b). This axis replaces the model's topology axis because the infinite grid and the graphs that are not vertex-transitive are not among the model's values.

Level	Value	Formal definition
0	**any graph**	Any connected anonymous graph, finite unless stated; robots see it whole but share no orientation.
1	**ring**	An anonymous, unoriented ring of n nodes.
1	**complete graph**	K_n: every two nodes are neighbours.
1	**hypercube**	The hypercube of dimension $\log n$: nodes are the n binary strings of length $\log n$, neighbours when they differ in one bit. Its diameter is $D = \log n$.
1	**infinite grid**	The infinite square grid: nodes are the points of $\mathbb{Z}^2$, neighbours at Manhattan distance one; finitely many robots.
1	**finite grid**	An anonymous $a \times b$ grid without orientation: nodes are the points (i, j) with $1 \le i \le a$ and $1 \le j \le b$, neighbours at distance one.
1	**tree**	An anonymous tree of n nodes without orientation.
1	**not vertex-transitive**	A finite connected graph whose nodes fall into at least two **orbits**, an orbit being a class of nodes that automorphisms of G map onto each other.

Case tree

The root assumes nothing; going down, each edge adds the assumptions written in grey (+) or leaves an axis unassumed. Bold entries are catalogued settings with their status per measure (Time) and page; italic entries are branch points, named after the axis on which their branches differ. Axis order: Graph dynamics → Edge direction → Faults → Network → Multiplicity detection → Initial configuration → Scheduler → Activation order.

No assumptions

+ static graph, + undirected, + no faults

Network

+ ring

Multiplicity detection

no multiplicity detection, any start, + sequential

Rings, sequential scheduler, no multiplicity detection ● p. 147

+ round robin

Rings, round robin, no multiplicity detection STANDARD SETTING ● p. 147

no multiplicity detection, + dispersed start

Scheduler

Claude (Anthropic) was used in cataloguing the problems.

+ sequential, + round robin
Rings, round robin, one robot per node at the start ● p. 148

+ SSYNC
Rings, SSYNC, no multiplicity detection ● p. 149

+ weak detection, + dispersed start
Rings, ASYNC, weak multiplicity detection ● p. 149

+ complete graph
Multiplicity detection

no multiplicity detection, any start, + sequential, + round robin
Complete graphs, round robin ● p. 150

+ strong detection, + dispersed start, + FSYNC
Complete graphs, FSYNC, strong multiplicity detection ● p. 151

+ not vertex-transitive, + sequential, + round robin
Graphs that are not vertex-transitive, round robin ● p. 152

+ hypercube, + sequential, + round robin
Hypercubes, round robin ● p. 153

+ infinite grid, + sequential, + round robin
Infinite grid, round robin ● p. 153

+ finite grid, + dispersed start
Finite grids, ASYNC, no multiplicity detection ● p. 154

+ tree, + dispersed start
Trees, ASYNC, no multiplicity detection ● p. 155

What is known

The best upper bound against the best lower bound in each catalogued setting, in tree order. The settings follow in detail.

Setting	Time	p.
Rings, sequential scheduler, no multiplicity detection	● impossible	147
Rings, round robin, no multiplicity detection STANDARD SETTING	● tight $O(n)$ vs *impossible*	147
Rings, round robin, one robot per node at the start	● tight $O(n)$ vs *impossible*	148
Rings, SSYNC, no multiplicity detection	● impossible	149
Rings, ASYNC, weak multiplicity detection	● unknown *finite* vs *impossible*	149
Complete graphs, round robin	● impossible	150
Complete graphs, FSYNC, strong multiplicity detection	● near-optimal $O(\log k)$ vs *impossible*	151
Graphs that are not vertex-transitive, round robin	● polynomial gap $O(\ell d + n)$ vs $\Omega(d)$	152
Hypercubes, round robin	● tight $O(\log n)$ vs *impossible*	153

Claude (Anthropic) was used in cataloguing the problems.

Setting	Time	p.
Infinite grid, round robin	● tight $O(d)$ vs *impossible*	153
Finite grids, ASYNC, no multiplicity detection	● unknown *finite* vs *impossible*	154
Trees, ASYNC, no multiplicity detection	● unknown *finite* vs *impossible*	155

16.5.1 Rings, sequential scheduler, no multiplicity detection

Adds no multiplicity detection, any start, + sequential VERIFIED

If the scheduler may activate the robots one at a time in any fair order, $k \geq 3$ robots that cannot see towers never gather on a ring.

Assumed: graph dynamics: static graph; edge direction: undirected; faults: no faults; network: ring; scheduler: sequential: exactly one robot is activated per round. **Not assumed:** multiplicity detection: none: a tower looks like one robot; initial configuration: arbitrary (agents on any $\ell \leq k$ nodes); activation order: any order the scheduler chooses.

Measure	Lower bound	Best upper bound	Status
Time	*impossible* (Navarra and Piselli 2025) For every $k \geq 3$.	—	● impossible

Results.

- Navarra and Piselli (2025) [lower bound]: time *impossible. Bottleneck:* adversarial activation. *Holds when:* $k \geq 3$ (Theorem III.1); just before gathering, all robots are on two adjacent nodes and one of them is a tower; every robot must move to the other node (Lemma III.1), and the adversary alternates between the two nodes.

Notes. The proof uses only the last step of an execution, which also arises from starts with one robot per node; the paper states the result for starts that may contain towers. Round robin, which fixes the order, makes the problem solvable. Checked against the full text of (Navarra and Piselli 2025) (arXiv v2).

16.5.2 Rings, round robin, no multiplicity detection

Adds + round robin **to** Rings, sequential scheduler, no multiplicity detection (p. 147) STANDARD SETTING VERIFIED

Under round robin, robots that cannot see towers gather on a ring from every start except a few listed ones, in at most $n - 3$ epochs, which is optimal up to a constant factor.

Assumed: graph dynamics: static graph; edge direction: undirected; faults: no faults; network: ring; scheduler: sequential: exactly one robot is activated per round; activation order: round robin: one robot per round in a fixed cyclic order. **Not assumed:** multiplicity detection: none: a tower looks like one robot; initial configuration: arbitrary (agents on any $\ell \leq k$ nodes).

Statement for this setting. Towers may exist at the start and are invisible. The unsolvable starts are those whose occupied nodes are exactly two adjacent nodes or three consecutive nodes, those with every node occupied, and on the 5-node ring some starts with three or four occupied nodes (Navarra and Piselli 2025).

Measure	Lower bound	Best upper bound	Status
Time	*impossible* (Navarra and Piselli 2025); $\Omega(n)$ (Navarra and Piselli 2025) $n - 3$ epochs against about $n/2$.	$O(n)$ (Navarra and Piselli 2025)	● tight

Claude (Anthropic) was used in cataloguing the problems.

Results.

- Navarra and Piselli (2025) [lower bound]: time *impossible*. *Bottleneck:* indistinguishable multiplicities. *Holds when:* $k \geq 3$ robots whose occupied nodes are exactly two adjacent nodes, three consecutive nodes, or all n nodes (Theorem III.2); on the 5-node ring with $k \geq 5$ robots: three occupied nodes with a tower next to another occupied node (Theorem III.3), or four occupied nodes (Corollary III.1); the adversary chooses where the hidden towers are and the cyclic order; on two adjacent occupied nodes every robot must move to the other node (Lemma III.1), and an order that alternates between the two nodes never empties either.
- Navarra and Piselli (2025) [lower bound]: time $\Omega(n)$. *Bottleneck:* distance. *Holds when:* a solvable start with $n-1$ occupied nodes, $n \geq 6$: every node is at distance at least $\lfloor n/2 \rfloor - 1$ from some robot; under round robin a robot moves at most one edge per epoch; the paper argues with the fully occupied ring (Section V), which can be gathered only when it starts with one robot per node.
- Navarra and Piselli (2025) [current best]: time $O(n)$. *Assumes:* the start is not one of the unsolvable starts. Introductory and current best: at most $n-3$ epochs (Theorem V.1); no earlier round-robin result on rings was found. Read in the arXiv version (v2).

Open questions.

Q1. Which sequential schedulers between round robin and an arbitrary sequential scheduler still allow gathering on rings without multiplicity detection? (Navarra and Piselli 2025) ask about other sequential schedulers; on graphs that are not vertex-transitive, a new order in every epoch suffices (Cicerone et al. 2025d).

Notes. Navarra and Piselli build the algorithm from tasks that a robot recognises in its snapshot: first form two groups of consecutive occupied nodes separated by one empty node on one side and by at least two on the other; that empty node becomes the gathering node; then shrink the two groups onto its two neighbours and move the robots onto it. Round robin makes the last step safe: once the gathering node is fixed, every robot is activated once before any robot is activated again (Navarra and Piselli 2025). The same algorithm solves the setting with one robot per node at the start. Checked against the full text of (Navarra and Piselli 2025) (arXiv v2).

16.5.3 Rings, round robin, one robot per node at the start

Adds + sequential, + round robin VERIFIED

With one robot per node at the start, only three robots on three consecutive nodes and the full 4-node and 5-node rings cannot be gathered; every other start is gathered in at most $n-3$ epochs.

Assumed: graph dynamics: static graph; edge direction: undirected; faults: no faults; network: ring; initial configuration: dispersed or scattered (one agent per occupied node); scheduler: sequential: exactly one robot is activated per round; activation order: round robin: one robot per round in a fixed cyclic order. **Not assumed:** multiplicity detection: none: a tower looks like one robot.

Statement for this setting. Every node holds at most one robot at the start; towers formed later are invisible. (Navarra and Piselli 2025) call this *distinct gathering*.

Measure	Lower bound	Best upper bound	Status
Time	*impossible* (Navarra and Piselli 2025); $\Omega(n)$ (Navarra and Piselli 2025) $n-3$ epochs against $\lfloor n/2 \rfloor$.	$O(n)$ (Navarra and Piselli 2025)	• tight

Results.

- Navarra and Piselli (2025) [lower bound]: time *impossible*. *Bottleneck:* indistinguishable multiplicities. *Holds when:* three robots on three consecutive nodes, for every n (Theorem III.2); the 4-node and 5-node rings with every node occupied (Lemmas III.2, III.3); the adversary chooses the cyclic order.
- Navarra and Piselli (2025) [lower bound]: time $\Omega(n)$. *Bottleneck:* distance. *Holds when:* the fully occupied ring with $n \geq 6$: whatever the gathering node, some robot is at distance $\lfloor n/2 \rfloor$ from it (Section V); a robot moves at most one edge per epoch.
- Navarra and Piselli (2025) [current best]: time $O(n)$. *Assumes:* the start is not one of the three unsolvable kinds. Introductory and current best as far as found: at most $n - 3$ epochs (Theorem V.1), with the algorithm of the setting that allows towers at the start.

Notes. Allowing towers at the start adds two adjacent occupied nodes, every node occupied and two kinds of starts on the 5-node ring to the unsolvable starts, since hidden towers can sit anywhere. Checked against the full text of (Navarra and Piselli 2025) (arXiv v2).

16.5.4 Rings, SSYNC, no multiplicity detection

Adds + SSYNC VERIFIED

Without multiplicity detection, $k \geq 2$ robots cannot gather on any ring when the scheduler may activate several robots in the same round.

Assumed: graph dynamics: static graph; edge direction: undirected; faults: no faults; network: ring; initial configuration: dispersed or scattered (one agent per occupied node); scheduler: semi-synchronous (SSYNC). **Not assumed:** multiplicity detection: none: a tower looks like one robot; activation order: any order the scheduler chooses.

Measure	Lower bound	Best upper bound	Status
Time	*impossible* (Klasing et al. 2008) For every $k \geq 2$.	—	• impossible

Results.

- Klasing et al. (2008) [lower bound]: time *impossible*. *Bottleneck:* indistinguishable multiplicities. *Holds when:* every $k \geq 2$ and every ring, from some start with one robot per node (Proposition 2.1); two robots at odd distance activated together stay at odd distance; with more robots, the adversary lets one robot create the first tower alone, after which the snapshots are those of $k - 1$ robots, and induction applies; stated for ASYNC; the adversary's schedules are semi-synchronous, so the proof holds under SSYNC (not stated in the paper).

Notes. Proposition 2.1 is why (Klasing et al. 2008) assume multiplicity detection. Round robin avoids the impossibility for most starts (Navarra and Piselli 2025). Checked against the model section and the proof of Proposition 2.1 in (Klasing et al. 2008).

16.5.5 Rings, ASYNC, weak multiplicity detection

Adds + weak detection, + dispersed start DRAFT

With weak multiplicity detection, an odd number of asynchronous robots gathers on a ring exactly from the starts that are not periodic; two robots never gather.

Assumed: graph dynamics: static graph; edge direction: undirected; faults: no faults; network: ring; multiplicity detection: global weak: a robot sees which nodes are towers; initial configuration: dispersed or scattered (one agent per occupied node). **Not assumed:** scheduler: asynchronous (ASYNC); activation order: any order the scheduler chooses.

Statement for this setting. Starts have one robot per node. A start is **periodic** if a rotation of the ring other than the identity maps the occupied nodes onto themselves.

Claude (Anthropic) was used in cataloguing the problems.

Measure	Lower bound	Best upper bound	Status
Time	*impossible* (Klasing et al. 2008); *impossible* (Klasing et al. 2008); *impossible* (D'Emidio et al. 2018) Time not analysed in (Klasing et al. 2008); later work on even k was not read.	*finite* (Klasing et al. 2008)	• unknown

Results.

- Klasing et al. (2008) [lower bound]: time *impossible*. *Bottleneck:* symmetry breaking. *Holds when:* periodic starts (Theorem 3.1); starts of an even number of robots on a ring of even size that are symmetric about an axis through the middles of two edges (Theorem 3.2); the adversary activates all robots together, so the symmetry persists; holds even under FSYNC.
- Klasing et al. (2008) [lower bound]: time *impossible*. *Bottleneck:* symmetry breaking. *Holds when:* $k = 2$, from a start at odd distance (Proposition 2.1).
- D'Emidio et al. (2018) [lower bound]: time *impossible*. *Bottleneck:* asynchrony. *Holds when:* $k = 4$ robots on a ring of 7 nodes, from a symmetric start whose axis passes through a node (start (i) of their Fig. 3; Theorem 3.5); computer-assisted: Lemmas 3.2–3.4 restrict the moves any algorithm may make, and all 57 600 remaining strategies were explored and fail; the same robots gather in SSYNC (Theorem 4.2), so asynchrony is the obstacle.
- Klasing et al. (2008) [introductory]: time *finite*. *Assumes:* odd $k \geq 3$; the start is not periodic. Theorem 6.1; with Theorem 3.1 this settles odd k. Time is not analysed. For even k the paper leaves open the symmetric starts whose axis passes through a node (Section 7).

Open questions.

Q1. Even k: which symmetric starts whose axis passes through a node can be gathered? Left open by (Klasing et al. 2008); four robots on rings of seven and nine nodes cannot gather from some of them (D'Emidio et al. 2018), and the proofs do not extend to other ring sizes. Later ring papers listed in (Cicerone et al. 2025d) were not read here.

Notes. Read: the model section, Proposition 2.1, Theorems 3.1, 3.2 and 6.1 and the conclusion of (Klasing et al. 2008); the algorithms were not checked in detail. D'Emidio, Di Stefano, Frigioni and Navarra show, with a computer-assisted search, that four robots on a ring of seven nodes cannot gather in ASYNC from a symmetric start whose axis passes through a node (Theorem 3.5), which disproves conjectures of (Klasing et al. 2008) and of Koren about which starts can be gathered; four robots on nine nodes are ungatherable too (the IWOCA 2015 version, as reported), and the unified ring algorithm of D'Angelo, Di Stefano and Navarra, which delegates four robots to Koren's algorithm, does not cover these starts (D'Emidio et al. 2018). On rings, FSYNC robots are strictly stronger than SSYNC ones, which are strictly stronger than ASYNC ones: four robots on a ring of five nodes gather in FSYNC but not in SSYNC (Theorem 4.1, Corollary 4.3). Model checking found deadlocks in the ring-gathering algorithm of D'Angelo, Di Stefano and Navarra (Distributed Computing 2014) for symmetric configurations with two multiplicities (Doan et al. 2017).

16.5.6 Complete graphs, round robin

Adds no multiplicity detection, any start, + sequential, + round robin VERIFIED

On a complete graph, robots that cannot see towers never gather under round robin once two nodes are occupied.

Assumed: graph dynamics: static graph; edge direction: undirected; faults: no faults; network: complete graph; scheduler: sequential: exactly one robot is activated per round; activation order: round robin: one robot per round in a fixed cyclic order. **Not assumed:** multiplicity detection: none: a tower looks like one robot; initial configuration: arbitrary (agents on any $\ell \leq k$ nodes).

Claude (Anthropic) was used in cataloguing the problems.

Measure	Lower bound	Best upper bound	Status
Time	*impossible* (Cicerone et al. 2025b) [preprint] From every start with two or more occupied nodes.	—	• impossible

Results.

- Cicerone et al. (2025b) [preprint] [lower bound]: time *impossible. Bottleneck:* indistinguishable multiplicities. *Holds when:* every start with at least two occupied nodes (Theorem 2); a robot can only move to an occupied or to an empty node, all alike; moving to empty nodes eventually occupies every node (unsolvable by Theorem 1), and with a hidden tower on every occupied node the adversary's order never empties a node; the same holds on complete bipartite graphs with equal sides when both sides are occupied (Theorem 2).

Notes. With all robots on one side of a complete bipartite graph, one epoch suffices: the first robot crosses and every later one joins it (Cicerone et al. 2025b). Checked against the full text of (Cicerone et al. 2025b) (arXiv v1).

16.5.7 Complete graphs, FSYNC, strong multiplicity detection

Adds + strong detection, + dispersed start, + FSYNC VERIFIED

On a complete graph, FSYNC robots that start on distinct nodes and see how many robots each node holds gather exactly when the least prime factor of k exceeds the number $n-k$ of empty nodes, in $O(\log k)$ epochs.

Assumed: graph dynamics: static graph; edge direction: undirected; faults: no faults; network: complete graph; multiplicity detection: global strong: a robot sees how many robots each node holds; initial configuration: dispersed or scattered (one agent per occupied node); scheduler: fully synchronous (FSYNC). **Not assumed:** activation order: any order the scheduler chooses.

Measure	Lower bound	Best upper bound	Status
Time	*impossible* (Cicerone et al. 2021b); $\Omega(\log k)$ (Cicerone et al. 2021b) $O(\log k)$ against $\Omega(\log k)$ for k a Fibonacci prime: a $\log k$ factor for other k. (Cicerone et al. 2021b) call the algorithm asymptotically optimal (Corollary 18), but their lower-bound starts are solvable only when $k = F_\ell$ is prime, and it is not known whether there are infinitely many Fibonacci primes. Own observation of this catalogue, not taken from a paper.	$O(\log k)$ (Cicerone et al. 2021b)	• near-optimal

Results.

- Cicerone et al. (2021b) [lower bound]: time *impossible. Bottleneck:* symmetry breaking. *Holds when:* every start with $k = n$, and every start in which the least prime factor p of k is at most $n-k$ (Theorems 1 and 10); all nodes of a complete graph look alike, so an algorithm must send moving robots to empty nodes (Corollary 3); the adversary sends them to p empty nodes, k/p robots on each, and every later move rebuilds such a configuration; holds even with strong multiplicity detection and FSYNC.
- Cicerone et al. (2021b) [lower bound]: time $\Omega(\log k)$. *Bottleneck:* symmetric splitting. *Holds when:* a complete graph with $F_{\ell+1}$ nodes and $k = F_\ell$ robots, F_ℓ a Fibonacci number, on distinct nodes (Theorem 16, Corollary 17); the adversary keeps the nodes in classes of equivalent nodes, each of Fibonacci size and holding equal towers, and the smallest class loses at most two Fibonacci indices per round; by Theorem 10 this start is solvable only when F_ℓ is prime.
- Cicerone et al. (2021b) [current best]: time $O(\log k)$. *Assumes:* the start is not one of the unsolvable ones. Introductory and current best (Theorems 10 and 11): in the first round every robot moves to an empty node; from then on, the robots outside the smallest class of nodes holding equal towers (the class of larger towers on a tie) move into it. As $p > n-k$, two classes remain until all robots are on one node, and the adversary can do no better than halve the class each round, so $O(\log(n-k)) = O(\log k)$ rounds.

Open questions.

Q1. Does every algorithm need $\Omega(\log k)$ epochs on complete graphs for infinitely many k, without assuming infinitely many Fibonacci primes? The lower bound of (Cicerone et al. 2021b) uses $k = F_\ell$ robots and $F_{\ell-1}$ empty nodes, solvable only when F_ℓ is prime.

Notes. Cicerone, Di Stefano and Navarra also give a condition for general graphs: a start of k SSYNC robots with strong multiplicity detection is solvable if the graph has a minimal recognizable subgraph H (an induced subgraph that every automorphism maps onto itself, minimal with this property) whose number of connected components and their size are both smaller than the least prime factor of k (Theorem 9). On complete bipartite graphs with k_1 and k_2 robots on the two sides, they characterise the solvable starts for FSYNC robots (Theorem 13; for instance, every start with both sides occupied and k_1, k_2 coprime) and gather in $O(\log k + \delta(k))$ rounds, $\delta(k)$ the number of divisors of k (Corollary 14). Diameter is the general lower bound on time. Checked against the full text of (Cicerone et al. 2021b) (journal pre-proof).

16.5.8 Graphs that are not vertex-transitive, round robin

Adds + not vertex-transitive, + sequential, + round robin VERIFIED

Under round robin, robots that cannot see towers gather from every start on a finite graph that is not vertex-transitive; time is optimal when the graph has a terminal orbit.

Assumed: graph dynamics: static graph; edge direction: undirected; faults: no faults; network: finite graphs that are not vertex-transitive; scheduler: sequential: exactly one robot is activated per round; activation order: round robin: one robot per round in a fixed cyclic order. **Not assumed:** multiplicity detection: none: a tower looks like one robot; initial configuration: arbitrary (agents on any $\ell \le k$ nodes).

Statement for this setting. An orbit O is **terminal** if every node of O can be reached from every node outside O by a path that meets O only at its end. ℓ is the number of initially occupied nodes and d the largest distance between two of them. Robots compute the orbits from their snapshot by canonical labelling of the graph.

Measure	Lower bound	Best upper bound	Status
Time	$\Omega(d)$ (Cicerone et al. 2025d)	$O(\ell\, d + n)$ (Cicerone et al. 2025d)	• polynomial gap
	Factor ℓ, plus an additive n, without a terminal orbit; $\Theta(D)$ in the worst case with one.		

Results.

- Cicerone et al. (2025d) [lower bound]: time $\Omega(d)$. *Bottleneck:* distance. *Holds when:* the two robots at distance d must meet, and each moves at most one edge per epoch (Lemma 1); for a worst-case start, $d = D$.
- Cicerone et al. (2025d) [current best]: time $O(\ell\, d + n)$. *Assumes:* unbounded local computation (canonical labelling of the snapshot). Introductory and current best (Theorem 5, graphs without a terminal orbit). With a terminal orbit, a simple algorithm gathers in $O(D)$ epochs (Theorem 1): robots leave the smallest terminal orbit until one of its nodes is occupied, then all walk to that node without entering the orbit elsewhere. Graphs with a universal node, a cut node or an orbit of twins have a terminal orbit (Theorem 3); butterflies and a family of circular graphs have none (Theorem 4). The algorithms also work if the adversary changes the order in every epoch. Read in the arXiv version (v1).

Open questions.

Q1. Can robots gather in $O(d)$ epochs on every graph that is not vertex-transitive and has no terminal orbit? Asked in (Cicerone et al. 2025d).

Claude (Anthropic) was used in cataloguing the problems.

Notes. There are no unsolvable starts: the orbits of the graph, which every robot computes, break the symmetry that defeats robots on rings. Checked against the full text of (Cicerone et al. 2025d) (arXiv v1).

16.5.9 Hypercubes, round robin

Adds + hypercube, + sequential, + round robin VERIFIED

On hypercubes of dimension at least 3, robots that cannot see towers gather under round robin in $O(\log n)$ epochs from every start except two adjacent occupied nodes, three occupied nodes forming a path, and every node occupied.

Assumed: graph dynamics: static graph; edge direction: undirected; faults: no faults; network: hypercube; scheduler: sequential: exactly one robot is activated per round; activation order: round robin: one robot per round in a fixed cyclic order. **Not assumed:** multiplicity detection: none: a tower looks like one robot; initial configuration: arbitrary (agents on any $\ell \leq k$ nodes).

Measure	Lower bound	Best upper bound	Status
Time	*impossible* (Cicerone et al. 2025b) [preprint]; $\Omega(\log n)$ (Cicerone et al. 2025b) [preprint] $\Theta(\log n)$ epochs.	$O(\log n)$ (Cicerone et al. 2025b) [preprint]	• tight

Results.

- Cicerone et al. (2025b) [preprint] [lower bound]: time *impossible*. *Bottleneck:* indistinguishable multiplicities. *Holds when:* starts whose occupied nodes are two adjacent nodes or all n nodes (Theorem 1, for every vertex- and edge-transitive graph); starts whose occupied nodes form a path of three nodes (Corollary 1): every algorithm must reach an empty node with all robots on its neighbours (Theorem 3) and then a path of three occupied nodes, and in a hypercube all such paths look alike; the adversary chooses the hidden towers and the order.
- Cicerone et al. (2025b) [preprint] [lower bound]: time $\Omega(\log n)$. *Bottleneck:* distance. *Holds when:* two robots at distance $D = \log n$ must meet, and each moves at most one edge per epoch (Lemma 1).
- Cicerone et al. (2025b) [preprint] [current best]: time $O(\log n)$. *Assumes:* dimension at least 3; the start is not one of the unsolvable ones. Introductory and current best (Theorem 4): the algorithm shrinks the smallest subcube containing all robots by one dimension at a time, in $O(1)$ epochs per dimension. Preprint. Read in the arXiv version (v1).

Notes. Checked against the full text of (Cicerone et al. 2025b) (arXiv v1).

16.5.10 Infinite grid, round robin

Adds + infinite grid, + sequential, + round robin VERIFIED

On the infinite grid, robots that cannot see towers gather under round robin in a number of epochs linear in their largest distance, from every start except two adjacent occupied nodes and one kind of three-node path.

Assumed: graph dynamics: static graph; edge direction: undirected; faults: no faults; network: infinite square grid; scheduler: sequential: exactly one robot is activated per round; activation order: round robin: one robot per round in a fixed cyclic order. **Not assumed:** multiplicity detection: none: a tower looks like one robot; initial configuration: arbitrary (agents on any $\ell \leq k$ nodes).

Statement for this setting. d is the largest distance between two initially occupied nodes.

Claude (Anthropic) was used in cataloguing the problems.

Measure	Lower bound	Best upper bound	Status
Time	*impossible* (Cicerone et al. 2025b) [preprint]; $\Omega(d)$ (Cicerone et al. 2025b) [preprint] $\Theta(d)$ epochs.	$O(d)$ (Cicerone et al. 2025b) [preprint]	• tight

Results.

- Cicerone et al. (2025b) [preprint] [lower bound]: time *impossible*. *Bottleneck:* indistinguishable multiplicities. *Holds when:* two adjacent occupied nodes (Theorem 1); one of the two kinds of three-node paths must also be excluded (Corollary 1): the algorithm of Theorem 5 excludes three occupied nodes of a 2×2 square; a variant that is only sketched excludes three consecutive nodes of a row instead (Theorem 6).
- Cicerone et al. (2025b) [preprint] [lower bound]: time $\Omega(d)$. *Bottleneck:* distance. *Holds when:* the two robots at distance d must meet, and each moves at most one edge per epoch (proof of Lemma 1).
- Cicerone et al. (2025b) [preprint] [current best]: time $O(d)$. *Assumes:* the start is neither two adjacent occupied nodes nor three occupied nodes of a 2×2 square. Introductory and current best (Theorem 5): shrink the smallest rectangle enclosing the robots, keeping a corner empty, down to two opposite nodes of a 2×2 square, then gather on one of its two empty nodes within one epoch. Stated as $O(n+m)$ epochs for the $m \times n$ enclosing rectangle (their m, n count rows and columns), which is $O(d)$. Preprint. Read in the arXiv version (v1).

Notes. The characterisation of Theorem 6 (exactly one of the two kinds of three-node paths must be excluded) depends on the sketched variant of the algorithm. Checked against the full text of (Cicerone et al. 2025b) (arXiv v1).

16.5.11 Finite grids, ASYNC, no multiplicity detection

Adds + finite grid, + dispersed start VERIFIED

On a finite grid, ASYNC robots that cannot see towers gather from every start except those invariant under a rotation of a grid with an even side, those symmetric about an axis through edges, and every start on the 2×2 grid.

Assumed: graph dynamics: static graph; edge direction: undirected; faults: no faults; network: finite grid; initial configuration: dispersed or scattered (one agent per occupied node). **Not assumed:** multiplicity detection: none: a tower looks like one robot; scheduler: asynchronous (ASYNC); activation order: any order the scheduler chooses.

Measure	Lower bound	Best upper bound	Status
Time	*impossible* (D'Angelo et al. 2016) Finite time only; no bound on epochs is recorded.	*finite* (D'Angelo et al. 2016)	• unknown

Results.

- D'Angelo et al. (2016) [lower bound]: time *impossible*. *Bottleneck:* symmetry breaking. *Holds when:* starts invariant under a rotation by 90 or 180 degrees about the centre of the grid, or symmetric about an axis that passes through edges (Theorem 2); on a grid with both sides odd the centre is a node and every start is solvable (Theorem 1); every start with two or more occupied nodes on the 2×2 grid, which behaves as a ring of four nodes (Theorem 4); the adversary activates symmetric robots together, so every move keeps the symmetry.
- D'Angelo et al. (2016) [current best]: time *finite*. *Assumes:* the start is not one of the unsolvable ones. Introductory and current best (Theorems 1, 3 and 5): on a grid with both sides odd every robot moves to the central node; otherwise the robots read the configuration as strings from the corners and agree on a gathering node, which stays the same while robots move toward it, and towers formed on the way do no harm. Time is not analysed.

Open questions.

Claude (Anthropic) was used in cataloguing the problems.

Q1. How many epochs does gathering on finite grids without multiplicity detection take?

Notes. D'Angelo, Di Stefano, Klasing and Navarra show that the unsolvable starts are exactly those listed here: a start is unsolvable if and only if it is invariant under a rotation on a grid with at least one even side, symmetric about an axis through edges, or on the 2×2 grid (Section 5). With local multiplicity detection the 2×2 grid can be gathered exactly from the starts with three occupied nodes (Theorem 4). On rings no robots gather without multiplicity detection (Klasing et al. 2008); on grids and trees they do. Checked against the full text of (D'Angelo et al. 2016).

16.5.12 Trees, ASYNC, no multiplicity detection

Adds + tree, + dispersed start VERIFIED

On a tree, ASYNC robots that cannot see towers gather from every start except those in which removing the edge between the two centres leaves two isomorphic halves, robots included.

Assumed: graph dynamics: static graph; edge direction: undirected; faults: no faults; network: tree; initial configuration: dispersed or scattered (one agent per occupied node). **Not assumed:** multiplicity detection: none: a tower looks like one robot; scheduler: asynchronous (ASYNC); activation order: any order the scheduler chooses.

Measure	Lower bound	Best upper bound	Status
Time	*impossible* (D'Angelo et al. 2016) Finite time only; no bound on epochs is recorded.	*finite* (D'Angelo et al. 2016)	● unknown

Results.

- D'Angelo et al. (2016) [lower bound]: time *impossible. Bottleneck:* symmetry breaking. *Holds when:* a tree with two centres such that removing the edge between them leaves two subtrees that are isomorphic, occupied nodes mapped to occupied nodes (Theorem 6); every move of a robot in one subtree may be mirrored by the adversary in the other, so the robots never pick one node.
- D'Angelo et al. (2016) [current best]: time *finite. Assumes:* the start is not one of the unsolvable ones. Introductory and current best (Section 4): if the tree has one centre, every robot moves to it. With two centres, removing the edge between them splits the tree into two subtrees; the robots of the subtree with fewer occupied nodes move to the other centre, and once their subtree is empty all robots move there. When both subtrees hold as many occupied nodes, a string encoding of the subtrees with their robots picks the side whose robots move first, which breaks the tie. Time is not analysed.

Notes. The unsolvable starts of Theorem 6 are the only ones: every other start on a tree is gathered without multiplicity detection (Section 5 of D'Angelo et al. 2016). Checked against the full text of (D'Angelo et al. 2016).

16.6 Map construction

Agents must not only visit every node but output a map of the anonymous graph, up to isomorphism.

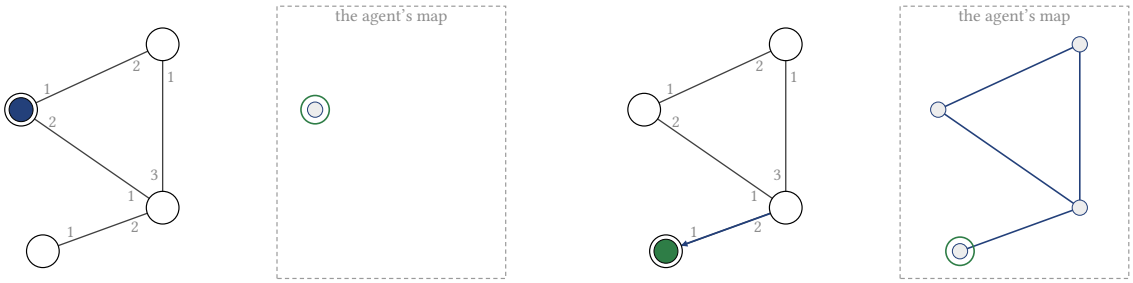
First posed in 1991 (Dudek et al. 1991). Dudek, Jenkin, Milios and Wilkes framed robotic exploration as graph construction.

Claude (Anthropic) was used in cataloguing the problems.

Problem statement and measures

Agents explore an unknown anonymous port-labelled graph and must output a **map**: a port-labelled graph, kept in the agent's memory, that is isomorphic to the explored graph (with the ports), with the agent's current node indicated. **Time** is the number of moves.

Measures: **Cost**, moves until the map is built.



(a) Start: an unknown anonymous graph; the map holds the start node

(b) Final: the last node is new; every port tried, the map is complete

Figure 16.6: Map construction with one movable marker (Dudek et al. 1991) (simplified), a setting of the problem of (Chalopin et al. 2010a): the agent crosses the edge at an untried port, drops the marker, steps back and tours the nodes of its map. If it finds the marker on a node already in the map, the edge leads there; otherwise the node is new, and the agent fetches the marker and adds the node and the edge to the map. Left: the unknown graph with its ports; right: the map, with the agent's position circled. One frame per move.

A first look

Imagine an explorer dropped into a maze of identical rooms joined by corridors, with no compass and no way to tell rooms apart; it must draw a correct map of the maze. Here the maze is a **graph** with n **nodes** (the rooms) and m **edges** (the corridors; each edge joins two nodes), and the explorer is an **agent** that walks from node to node along edges; one crossing of one edge is a **move**. The nodes are **anonymous**: they have no names the agent can read. At each node the edges at that node are numbered $1, 2, \ldots$ (these numbers are **ports**), so the agent can choose which edge to take and sees through which port it arrived. One agent starts at some node, its **start node**; it must walk around and finally output a **map**: a graph with port numbers, kept in the agent's memory, that is the same as the real graph up to renaming of the nodes, with the node where the agent stands indicated. We count the moves. In the basic catalogued version the agent cannot leave anything on a node, but it knows upper bounds on n and on the maximum **degree** Δ (the degree of a node is the number of edges at it).

1. Algorithm: Drop a marker to recognise a node. (Dudek et al. 1991) (One agent with a marker, p. 161)
The agent carries one **marker**, an object that it can put down on a node and pick up again (a simplified description of Dudek et al. 1991).

1. Keep a map of the nodes seen so far; at first it holds only the start node.
2. Choose a node of the map with a port that has not been tried yet. Walk to that node of the graph, following the map, and cross the edge at that port.
3. Put the marker down on the node of the graph just reached, step back across the same edge, and visit every node of the graph that is in the map (walking along edges of the map).

Claude (Anthropic) was used in cataloguing the problems.

4. If the marker is found on the node of the graph that a node u of the map stands for, the new edge leads to u: record that edge in the map, pick up the marker, and go to step 2.
5. If the marker is not found anywhere, the node is new: cross the edge again, pick up the marker, and add the node and the edge to the map.
6. Stop when every port of every node in the map has been tried.

Why it works and what it costs. Each edge is tried once from each end, so there are at most $2m$ tests. One test costs a walk to the edge plus a tour of the nodes in the map, and both are $O(n)$ moves if the agent walks along a tree inside its map. In total this is $O(mn)$ moves; (Dudek et al. 1991) count the moves precisely: with one marker, at most $7mn - 3n^2$ plus lower-order moves, and fewer with more markers. This is the current best bound recorded in the catalogue for the marker setting.

Why it matters. It showed that a single movable marker turns an unsolvable task into a polynomial one, and it started the study of how much marking or knowledge map construction needs.

2. Impossibility: Graphs that look the same from inside. (Chalopin et al. 2010a) (One agent, no marking, p. 160) Take a **ring** (nodes joined in a circle, each node with two edges) and number the ports so that at every node port 1 goes clockwise and port 2 goes counterclockwise. Now compare a ring of 3 nodes with a ring of 6 nodes. An agent without a marker sees the same thing at every node of both rings: two ports, and whichever port it takes, it arrives through the other one. Every walk it can make produces exactly the same observations in the two rings. So it cannot tell the rings apart, and it cannot tell when it has seen every node.

Why it holds. In general, two nodes with the same **view** (everything an agent can observe on walks that start at the node: the ports it can take and the ports through which it arrives) cannot be told apart, and the most an agent can learn is the **quotient graph**, which merges the nodes with the same view into one node (here all nodes have the same view, so the quotient graph is a single node with one edge from the node to itself, port 1 at one end and port 2 at the other). Without a marked node, (Chalopin et al. 2010a) note that no algorithm can explore graphs of arbitrary size and stop, so the agent needs extra knowledge (bounds on n and Δ) to stop, and then it builds the quotient graph, which equals the real graph only when all views are distinct.

Why it matters. It explains why every map construction algorithm needs something extra: a marker, a recognisable start node, or knowledge of the size.

3. Algorithm: Recognise a node by walking home. (Chalopin et al. 2010a) (One agent, no marking, p. 160) Suppose only the start node s is marked: the agent can recognise s when it stands on it, and nothing else.

1. Build the map breadth-first: for every node u already in the map, store the list of ports of a path from u back to s.
2. To test a port that has not been tried yet, cross its edge and arrive at some node v of the graph.
3. For each node u in the map: from v, follow the stored port list of u and see whether the walk ends on the marked node s; then walk back to v.
4. If some list ends at s, then v is u: record the edge in the map. Otherwise v is new, and the agent stores a path home for it (the edge it just crossed followed by the path of the node it came from).
5. Stop when every port has been tested.

Why it works and what it costs. The test is correct because a list of ports, read backwards from s, leads to exactly one node; if it leads home from both u and v, then $u = v$. There are at most $n\Delta$ ports to test,

Claude (Anthropic) was used in cataloguing the problems.

each test tries up to n stored paths, and each path has fewer than n moves, so the cost is $O(n^3\Delta)$ moves, the bound of (Chalopin et al. 2010a) (Proposition 9). This needs no knowledge of n, and the output is an exact copy of the graph. Without any marked node the same paper needs known bounds and universal exploration sequences, with $O(N^6\Delta^2 \log N)$ moves, where N is the known upper bound on n.

Why it matters. It shows that a marked start node, which cannot be moved, is already enough, which is much weaker than a marker the agent carries.

What the community started looking at

Map construction started in robotics as **exploration as graph construction** with markers (Dudek et al. 1991); in distributed computing the questions were **feasibility** for anonymous agents (Das et al. 2007) and then the cost for a single agent without marks (Chalopin et al. 2010a), as surveyed in (Das 2019).

Further reading

Papers on this problem that are not yet placed in a setting.

- Stopkin (2025c). One agent that paints nodes, edges and the ends of edges with a single colour reports each step to a second, stationary agent outside the graph, which builds the map; a depth-first algorithm takes $O(n^2)$ moves, $O(n^2)$ time and memory, and $O(n^2 \log n)$ bits of messages. No lower bound. In Ukrainian. Read.
- Chalopin et al. (2016b). With binoculars of radius 1, one agent explores every Weetman graph (a class containing the chordal graphs, some of them with $\Theta(n^2)$ edges) and builds its map in $O(n)$ moves, without knowing n or the diameter and for any port numbering. Read (arXiv v1).
- Stepkin and Stepkina (2021). Two finite-memory agents, one walking and marking the graph and one building the map, recognise a simple graph. In Russian; abstract read.
- Stopkin (2024). A variant of the same two-agent recognition algorithm for simple undirected graphs. In Ukrainian; abstract read.
- Stopkin (2025a). An agent-researcher that traverses and marks the graph and an agent-experimenter that builds its map explore simple undirected graphs. Abstract read.

Relations to other problems

- **Strengthening of Graph exploration by a single agent.** Building an isomorphic map implies having visited every node, so map construction is exploration with more output; it is used to solve rendezvous and leader election (Das 2019).
- **Related: Deciding graph properties by mobile agents.** Two or more agents that know their number decide every property by gathering and drawing a map, one agent using the others as a token; a lone agent with a map decides every property (Fraigniaud and Pelc 2017).

Assumptions specific to this problem

Agents

Map construction began as a single-robot problem: Dudek, Jenkin, Milios and Wilkes framed robotic exploration as building a graph, with a robot carrying one or more markers (Dudek et al. 1991). Das, Flocchini, Kutten, Nayak and Santoro posed the distributed version: k identical agents, initially dispersed among the nodes and communicating through whiteboards, must each build an identically labelled map (Das et al. 2007). They note that it "is much more difficult than exploration by a single agent, because it

Claude (Anthropic) was used in cataloguing the problems.

involves achieving cooperation among multiple agents", and that it is deterministically unsolvable in some cases (Das et al. 2007). The axis separates the single agent, whose difficulty is recognising nodes it has already seen, from the team, which must in addition break the symmetry between identical agents.

Level	Value	Formal definition
0	**one agent**	A single agent.
1	**several agents**	$k \geq 2$ identical agents.

Knowledge

In an anonymous graph an agent that cannot mark nodes cannot tell a revisited node from a new one by looking at it, so it needs information from outside before it can commit to a map. The standard information is an upper bound on n, which fixes how long a universal sequence must be followed; Chalopin, Das and Kosowski state the trade-off exactly: "in the model where no vertices of the graph are marked, for the task of map construction to be feasible, the agents must know some bound n on the number of nodes of the graph and some bound $d \leq n$ on its degree", and with these bounds they build the map by following universal exploration sequences; without any knowledge they instead assume that the starting node is marked, and then build the map in $O(n^3\Delta)$ moves with $O(n\Delta \log n)$ bits of memory (Chalopin et al. 2010a). The clearest measurement of what the bound is worth comes from directed graphs: Bender et al. map any strongly connected digraph with one pebble when an upper bound on n is known, while without it $\Theta(\log \log n)$ pebbles are necessary and sufficient (Bender et al. 2002).

Level	Value	Formal definition
0	**nothing known**	Nothing is known about G.
1	**bound on n**	An upper bound $N \geq n$ is known.

Markers

A marker lets a robot recognise one chosen node, which is exactly what an anonymous graph withholds. Dudek, Jenkin, Milios and Wilkes introduced the robot that carries $k \geq 1$ distinct markers and builds the graph as it explores; they show that without markers the task is unsolvable in general (regular graphs of the same degree look alike everywhere) and that their algorithm makes at most $(5 + 2/k)mn - 3n^2$ plus lower-order moves, that is $O(mn) \subseteq O(n^2\Delta)$ (Dudek et al. 1991). Bender, Fernández, Ron, Sahai and Vadhan showed in "The power of a pebble" that one pebble also suffices to map a directed graph in polynomial time when an upper bound on n is known (Bender et al. 2002). The axis keeps carried markers apart from whiteboards because a marker is a single movable bit, while a whiteboard stores information permanently at every node.

Level	Value	Formal definition
0	**no markers**	No markers.
1	**markers**	The agent carries markers it can drop and pick up.

Whiteboards

Whiteboards, a small memory at every node that visiting agents read and write in mutual exclusion, are the standard communication device in the work of Flocchini, Santoro and coauthors on asynchronous agents (Dobrev et al. 2006; Flocchini et al. 2012b). Das, Flocchini, Kutten, Nayak and Santoro used them for map construction by k identical agents, which communicate only through these "small public bulletin boards available at each node" (Das et al. 2007). Flocchini, Ilcinkas and Santoro note that whiteboards also "allow to break symmetry among identical agents" (Flocchini et al. 2012b), which is what a team of identical agents needs in order to agree on one labelled map.

Claude (Anthropic) was used in cataloguing the problems.

Level	Value	Formal definition
0	**no whiteboards**	No whiteboards.
1	**whiteboards**	Each node has a whiteboard.

Case tree

The root assumes nothing; going down, each edge adds the assumptions written in grey (+) or leaves an axis unassumed. Bold entries are catalogued settings with their status per measure (Cost) and page; italic entries are branch points, named after the axis on which their branches differ. Axis order: Graph dynamics → Edge direction → Timing → Agents → Knowledge → Markers → Whiteboards → Topology.

No assumptions

+ static graph, + undirected

Agents

one agent, + bound on n

One agent, no marking STANDARD SETTING ● p. 160

one agent, nothing known, + markers

One agent with a marker ● p. 161

+ several agents, + whiteboards

Identical agents with whiteboards ● p. 162

What is known

The best upper bound against the best lower bound in each catalogued setting, in tree order. The settings follow in detail.

Setting	Cost	p.
One agent, no marking STANDARD SETTING	● unknown $O(N^6\Delta^2 \log N)$ vs —	160
One agent with a marker	● unknown $O(mn)$ vs —	161
Identical agents with whiteboards	● unknown $O(m \log k)$ vs *impossible*	162

16.6.1 One agent, no marking

Adds one agent, + bound on n STANDARD SETTING VERIFIED

Knowing upper bounds on n and on the degree, one agent builds the quotient graph of an anonymous graph without marking anything, by following a universal exploration sequence.

Assumed: graph dynamics: static graph; edge direction: undirected; knowledge: an upper bound on n. **Not assumed:** timing: asynchronous; agents: a single agent; markers: no markers; whiteboards: no whiteboards; topology: arbitrary connected graph.

Statement for this setting. The agent knows an upper bound $N \geq n$ (and a bound on the maximum degree; $N - 1$ will do) and may not mark nodes. Following (Chalopin et al. 2010a), it must output the **quotient graph** of G with its port numbering, the multigraph obtained by merging nodes with the same view; this is G itself exactly when all nodes have distinct views, and nothing more can be learned,

Claude (Anthropic) was used in cataloguing the problems.

since graphs with the same quotient graph are indistinguishable to the agent. **Time** is the number of edge traversals; local computation is free.

Measure	Lower bound	Best upper bound	Status
Cost	none known No lower bound is proved in (Chalopin et al. 2010a).	$O(N^6\Delta^2 \log N)$ (Chalopin et al. 2010a)	• unknown

Results.

- Chalopin et al. (2010a) [current best]: cost $O(N^6\Delta^2 \log N)$. *Assumes:* upper bounds on n and on Δ known. Introductory and current best (Proposition 6): follow a universal exploration sequence $U(2N^2, \Delta)$ of length $O(N^6\Delta^2 \log N)$ from Aleliunas et al., record the signature, and compute the smallest graph with that signature; memory of the same order. The paper writes n and d for the known bounds. Using the routines of Czyzowicz, Kosowski and Pelc (Czyzowicz et al. 2012b) gives $O(n^{10}\Delta^5 \log^2 n)$ time with $O(n^9\Delta^4 \log^2 n)$ memory (as computed in Chalopin et al. 2010a).
- Chalopin et al. (2010a) [also]: cost $O(N^6\Delta^3 \log N)$. *Assumes:* upper bounds on n and on Δ known. Proposition 7 (class refinement, as in automaton minimisation): slightly more time for much less memory.
- Chalopin et al. (2010a) [also]: cost $O(n^3\Delta)$. *Assumes:* the starting node is marked (recognisable); no bound on n needed. Proposition 9: breadth-first construction that recognises a node by walking its path back to the marked home base; the output is an exact copy of G. With a marked start and no knowledge, Proposition 8 also computes a polynomial bound $N \geq n$ with $\Theta(\log n)$ bits, so map construction is possible in polynomial time with optimal memory.

Notes. Checked against the full text of (Chalopin et al. 2010a). Which bound needs what: without any marked node the agent must know bounds on n and on the degree (Propositions 6 and 7, $O(N^6\Delta^2 \log N)$ and $O(N^6\Delta^3 \log N)$); with a marked starting node no knowledge is needed (Propositions 8 and 9, $O(n^3\Delta)$ moves and $O(n\Delta \log n)$ bits). The paper writes that "it is not possible to perform exploration with stop in unlabelled graphs of arbitrary size and topology" without a marked node; a marked home base is "a much weaker assumption compared to allowing the agent to have a pebble". The same paper solves rendezvous of two agents with marked home bases, even asynchronously, in $O(n^4\Delta^2)$ moves per agent whenever it is deterministically possible, and detects when it is not (Propositions 10 and 11). The timing axis is not set because a single agent's moves are counted regardless of timing. (Das 2019) summarises that a full map is obtainable only for covering-minimal graphs; in the paper's terms, an exact copy is obtained exactly when all views are distinct. The $O(n^3\Delta)$ bound is the one Disser, Hackfeld and Klimm report (Disser et al. 2019).

16.6.2 One agent with a marker

Adds one agent, nothing known, + markers VERIFIED

An agent with $k \geq 1$ distinct markers it can drop and pick up builds the graph as it explores, in $O(mn)$ moves.

Assumed: graph dynamics: static graph; edge direction: undirected; markers: markers the agent carries. **Not assumed:** timing: asynchronous; agents: a single agent; knowledge: nothing known; whiteboards: no whiteboards; topology: arbitrary connected graph.

Measure	Lower bound	Best upper bound	Status
Cost	none known No lower bound recorded; (Dudek et al. 1991) reports that experiments needed 25–65% of the bound.	$O(mn)$ (Dudek et al. 1991)	• unknown

Results.

Claude (Anthropic) was used in cataloguing the problems.

- Dudek et al. (1991) [current best]: cost $O(mn)$. *Assumes:* $k \geq 1$ distinct markers; the agent sees the edges at a node in a cyclic order relative to the edge it entered by. Introductory and current best recorded here: at most $(5 + 2/k)mn - 3n^2$ plus lower-order moves (equation (4)); a variant that marks both ends of an unexplored edge makes $(3 + 4/k)mn - n^2$ plus lower-order moves. The bound counts only physical moves; the agent's computation is $O(n^2m + m^2 \log m)$.

Notes. Checked against the full text of (Dudek et al. 1991) (a scan, read through OCR). The agent cannot measure distances and has no compass; without markers the problem is unsolvable in general, because regular graphs of the same degree look the same at every node. For directed graphs, one marker and a bound on n suffice, and without a bound $\Theta(\log \log n)$ pebbles are necessary and sufficient (Bender et al. 2002); Bender et al. also cite the $O(n \cdot m)$ bound of Dudek et al. for one marker.

16.6.3 Identical agents with whiteboards

Adds + several agents, + whiteboards VERIFIED

Several identical asynchronous agents with whiteboards build a common labelled map in $O(m \log k)$ moves when n and k are co-prime; otherwise some placements are unsolvable.

Assumed: graph dynamics: static graph; edge direction: undirected; agents: several identical agents; whiteboards: whiteboards at the nodes. **Not assumed:** timing: asynchronous; knowledge: nothing known; markers: no markers; topology: arbitrary connected graph.

Statement for this setting. k identical asynchronous agents start on k distinct nodes (their homebases, which are marked) of an anonymous graph with n nodes, m edges and arbitrary ports. Every node has a whiteboard that agents read and write in fair mutual exclusion; agents at the same node can talk. Each agent knows n or k. Every agent must return to its homebase with the same labelled map: a copy of the graph with ports in which every node carries the same label in every agent's map. Cost is the total number of moves.

Measure	Lower bound	Best upper bound	Status
Cost	*impossible* (Das et al. 2007) No lower bound on the number of moves is stated in (Das et al. 2007).	$O(m \log k)$ (Das et al. 2007)	● unknown

Results.

- Das et al. (2007) [lower bound]: cost *impossible*. *Bottleneck:* symmetry. *Holds when:* $\gcd(n, k) > 1$: on a symmetric graph such as a ring the agents can be placed symmetrically, so that no deterministic algorithm elects a leader (Lemma 2.2, derived from earlier results on election and rendezvous); map construction, leader election, labelling the nodes and rendezvous are solvable on exactly the same instances (Theorem 2.1); agents that know neither n nor k cannot solve every solvable instance (Lemma 2.1); some instances with $\gcd(n, k) > 1$ are solvable, e.g. a tree with an odd number of nodes.
- Das et al. (2007) [introductory]: cost $O(km)$. *Assumes:* each agent knows n or k; $O(\log n)$ bits per whiteboard. Algorithm MERGE-TREE (Theorems 5.1 to 5.3): solves the problem whenever $\gcd(n, k) = 1$, and every agent terminates explicitly, also on instances it cannot solve.
- Das et al. (2007) [current best]: cost $O(m \log k)$. *Assumes:* each agent knows $\gcd(n, k)$ and n or k (for instance both n and k); $\gcd(n, k) = 1$; $O(\log n)$ bits per whiteboard. Algorithm Explore-&-Capture (Theorems 6.1 and 6.2): at most $\log k$ phases of $O(m)$ moves each.

Notes. Checked against the full text of (Das et al. 2007). Both algorithms first let every agent explore and mark a territory around its homebase with a depth-first search ($4m - 2n + 2k$ moves in total, Lemma 3.2), then merge territories in phases: neighbouring territories compare tokens (number of nodes and number of agents), and the smaller is captured. If all territories have the same counts, $\gcd(n, k) > 1$,

which is how the agents detect an instance the algorithm cannot solve (Lemmas 5.3 and 5.4). Each agent stores a full map in its own memory; the whiteboards need only $O(\log n)$ bits (Theorem 6.2).



16.7 Exploration by oblivious robots

Oblivious Look–Compute–Move robots that see all robots must visit every node of a ring or grid and stop, or keep visiting every node forever without two robots ever meeting; how many robots are needed?

First posed in 2007 (Flocchini et al. 2013). Ring exploration by asynchronous oblivious robots (OPODIS 2007).

Problem statement and measures

k anonymous, oblivious robots run Look–Compute–Move cycles on an anonymous graph with n nodes: in a Look a robot takes a snapshot showing which nodes hold no robot, one robot or a tower, it computes from that snapshot only, and it stays or moves to a neighbour. The robots start on distinct nodes (no tower). Two goals are studied. In **terminating exploration** every node must be visited by at least one robot and the robots must reach a configuration in which no robot ever moves again. In **exclusive perpetual exploration** every robot must visit every node infinitely often, no two robots may ever be on the same node, and no two robots may ever cross the same edge in opposite directions. The setting fixes the goal, the scheduler (ASYNC, SSYNC or FSYNC) and whether robots may flip coins; which initial configurations an algorithm must handle (every configuration without towers, or only some) is stated with each result. The primary measure is the **number of robots**; the second is the total number of moves.

Measures: **Robots**, number of robots needed; **Moves**, total moves.



(a) Start: 5 robots on distinct nodes, n = 8

(b) Walker reaches the group from the other side: every node visited, all stop

Figure 16.7: Exploration of a ring by five oblivious robots (Lamani et al. 2010) (the problem: Flocchini et al. 2013): anonymous robots without memory, on distinct nodes of an anonymous ring with $n = 8$ nodes (co-prime with 5), run Look–Compute–Move cycles, one robot activated per frame here. They first close the gaps into a block of five, then the middle robot forms a tower, which fixes a direction; one end robot, the walker, moves away from the tower round the ring, and when it is back beside the group the configuration means that every node (shaded) has been visited, and no robot moves again. The rules are simplified from (Lamani et al. 2010).

Claude (Anthropic) was used in cataloguing the problems.

A first look

Imagine a few identical toy robots on a circular track of n squares. The track is a **ring**: its squares are its n **nodes**, and two squares side by side are **neighbours**. The robots cannot talk, they have no memory at all, and they cannot tell clockwise from counterclockwise, yet together they must visit every node and then stop forever. Each robot works in **Look–Compute–Move cycles**: **Look** (take a **snapshot** showing where all robots are), **Compute** (decide, from that snapshot only, whether to move and where) and **Move** (move to a neighbour). Robots are **oblivious**: they forget everything after each cycle. The nodes are **anonymous** (they have no names), and an **adversary** (an imagined opponent) decides when each robot is activated. A robot can tell whether several robots stand on one node (such a node is called a **tower**). The positions of all robots form the **configuration**. In the basic version k robots start on distinct nodes of a ring of n nodes; they must visit every node and then all stop. We count the robots needed.

1. Algorithm: Gather in a block, build a tower, send one walker. (Lamani et al. 2010) (Rings, ASYNC, terminating, p. 168) Five robots, on a ring whose size n is not a multiple of 5 (Lamani et al. 2010). The rule a robot follows depends only on its current snapshot, so each stage must leave a configuration that announces which stage it is.

1. **Block:** while the robots are not on five consecutive nodes (a **block**), robots move closer together, carefully so that no two land on the same node.
2. **Tower:** when the five robots form one block, the middle robot moves onto a neighbour. Now there is exactly one tower, and the configuration is no longer symmetric: it shows a direction.
3. **Walk:** one robot, the **walker**, singled out by its position relative to the tower, moves node by node around the ring, away from the tower.
4. When the walker reaches the other side of the group, the robots form a special pattern (a *tower-chain*); this configuration means the ring has been explored, and nobody moves again.

Why it works and what it costs. The condition that n and 5 share no common factor is what lets the robots break symmetry in the first stage (see the third idea for what goes wrong otherwise). Each stage costs $O(n)$ moves, so the total is $O(n)$, which is optimal, since the walker must cover the ring. Three robots never suffice, even if robots may flip coins (Devismes et al. 2013). For even n, four robots cannot explore without coins (Lamani et al. 2010) (the proof uses an adversary that activates robots with the same snapshot at the same moment), so five is optimal for those rings; for odd n it is open whether four robots suffice when the adversary may also interleave the three phases of different robots. The earlier algorithm of (Flocchini et al. 2013) needed 17 or more robots.

Why it matters. It shows the main technique of the area: memoryless robots store their progress in the shape of their own configuration.

2. Impossibility: The final configuration must contain a tower. (Devismes et al. 2021) (Grids, ASYNC, terminating, p. 170) An oblivious robot decides only from its current snapshot. Suppose that the robots stop in a configuration in which every robot is on its own node. That configuration could also be a starting configuration, since robots start on distinct nodes. Started from it, the robots would see snapshots in which they stop, so they would stop at once, after visiting only their own k nodes, fewer than the n nodes of the graph. So every final configuration must contain a tower.

Why it holds. A single robot can never make a tower, so one robot can never explore and stop, on any graph with at least two nodes. With more care (Devismes et al. 2021) show that two robots are not enough on a grid with at least three nodes, and they give an algorithm with three robots, which is optimal (four for the 2×2 grid, five for the 3×3 grid).

Claude (Anthropic) was used in cataloguing the problems.

Why it matters. This one-line remark is the starting point of most lower bounds for oblivious robots: termination itself must be written in the configuration.

3. Impossibility: Evenly spaced robots stay evenly spaced. (Flocchini et al. 2013) (Rings, ASYNC, terminating, p. 168) Let k divide n, with $k < n$, for example 3 robots on a ring of 6 nodes, one on every second node. Every robot takes exactly the same snapshot, the same in both directions. The adversary activates all robots together and, whenever a robot wants to move but cannot tell its two directions apart, sends every robot clockwise. Then all robots do the same thing: either nobody moves, or every robot moves clockwise and the configuration is again evenly spaced. The robots will take this same snapshot forever, so they either stop where they are, having visited only k nodes, or never stop.

Why it holds. So k robots cannot explore a ring whose size is a multiple of k (Flocchini et al. 2013). Whether exploration is also impossible when n and k share a smaller common factor is open: it was conjectured (Lamani et al. 2010), and a computer check found the algorithm of (Flocchini et al. 2013) correct for five robots on rings of 10 and 15 nodes from starts with at most one axis of symmetry (Bérard et al. 2016). When n and k share no factor and $k \geq 17$, exploration is solvable (Flocchini et al. 2013); the five-robot algorithm above later lowered the team size.

Why it matters. Symmetry that robots cannot break is the obstacle behind the number-theoretic conditions that appear throughout the oblivious-robot literature.

What the community started looking at

Robots with no memory must encode progress in their positions, so the first question was **the minimum number of robots**. On rings, k robots cannot explore when k divides n, which forces $\Omega(\log n)$ robots for infinitely many n, and $O(\log n)$ robots always suffice (Flocchini et al. 2013). Three robots never suffice, even with coin flips; four robots that flip coins suffice under SSYNC (Devismes et al. 2013). Without coins, five robots suffice whenever n is not a multiple of 5, four do not when n is even, and four suffice for odd $n > 7$ under SSYNC (Lamani et al. 2010); the exact minimum for every n is still open. On grids three robots suffice and are optimal (Devismes et al. 2021). Exclusive perpetual exploration of rings needs three robots when $n \geq 10$ and $3 \nmid n$ (Blin et al. 2010; Bérard et al. 2016).

Further reading

Papers on this problem that are not yet placed in a setting.

- Ilcinkas (2019). Survey of terminating and exclusive perpetual exploration by oblivious robots on rings, trees, lines, grids and tori, with the schedulers and multiplicity detection each bound needs. Read.
- Flocchini et al. (2010). Trees: $\Omega(n)$ robots are needed on some trees; $O(\log n / \log\log n)$ suffice and are needed when the maximum degree is 3. As reported by (Ilcinkas 2019); not read.
- Flocchini et al. (2011). Lines: $k < n$ robots explore an n-node line if and only if $k = 3$, $k \geq 5$, or $k = 4$ with n odd. As reported by (Ilcinkas 2019); not read.
- Devismes et al. (2019). Tori: at least 5 deterministic robots; 4 probabilistic robots for large tori. As reported by (Ilcinkas 2019); not read.
- Bonnet et al. (2011). Exclusive perpetual exploration of grids without sense of direction under ASYNC: 3 robots, except the 2×2 and 2×3 grids. As reported by (Ilcinkas 2019); not read.
- Chalopin et al. (2010b). Terminating exploration of arbitrary graphs with port numbers under ASYNC from rigid configurations: impossible with fewer than 3 robots, a characterisation of the graphs explorable by 3 robots, and every graph explorable by 4 robots and by any odd $k > 3$. As reported by (Ilcinkas 2019); not read.
- Bonnet et al. (2014). Enumerates all protocols for exclusive perpetual ring exploration; five robots on a 10-node ring are impossible under SSYNC from rigid starts. Abstract read; result as reported by (Ilcinkas 2019).

Claude (Anthropic) was used in cataloguing the problems.

- Doan et al. (2016). Model checking of the three-robot exclusive perpetual algorithm of (Blin et al. 2010); finds a collision, like (Bérard et al. 2016). As reported by (Ilcinkas 2019); not read.

- Sangnier et al. (2020). Parameterized verification of ring algorithms for oblivious robots: the three-robot exclusive perpetual algorithm of (Blin et al. 2010) is collision-free under FSYNC for $n > 10$, and its $n - 5$ algorithm has collisions for 7 and 9 robots. Section 4.5 of the 2017 report read.

- Balabonski et al. (2018a). Coq framework that certifies impossibility results for robots on graphs, such as the $k \mid n$ argument for ring exploration. Abstract read.

- Sardar et al. (2022). Terminating exploration of a finite grid by k oblivious ASYNC robots with unlimited visibility and one pebble each, minimising nodes visited twice: no repeated visit on grids with an even side, at most one on grids with both sides odd, and at most $n/(k+1) - 1$ on a $1 \times n$ grid, following Hamiltonian cycles or paths of the grid. Read (model, algorithms and bounds).

Relations to other problems

- **Variant of Graph exploration by a single agent.** The robots follow the Look–Compute–Move model of the plane, moved to graphs: they see the positions of all robots but remember nothing, so the configuration itself must encode the progress of the exploration.

- **Derived problem: Exploration by luminous robots** (variant of). The same Look–Compute–Move robots, but each carries a light with a constant number c of colours that acts both as persistent memory (the robot sees its own colour in the next cycle) and as a message (every robot within range reads it); in exchange, robots usually see only within a fixed distance φ. An oblivious algorithm is a luminous one with $c = 1$, so algorithms of the parent problem with myopic vision carry over, and lower bounds proved here for any number of colours hold for oblivious robots too (Ooshita and Tixeuil 2022).

- **Related: Gathering by oblivious robots on graphs.** The same robots and schedulers on graphs, with exploration as the task.

Assumptions specific to this problem

Scheduler

Ring exploration by oblivious robots on graphs began under ASYNC with instantaneous moves (Flocchini et al. 2013). Devismes, Petit and Tixeuil proved their bounds for probabilistic robots with atomic cycles (SSYNC) (Devismes et al. 2013), and Lamani, Potop-Butucaru and Tixeuil wrote their four-robot lower bound for SSYNC (it carries over to ASYNC) and gave a four-robot algorithm for odd n only under SSYNC (Lamani et al. 2010). The scheduler changes the answer: with coin flips, a constant number of robots suffices under SSYNC, while $\Omega(\log n)$ robots are needed under ASYNC for infinitely many n, as reported by the survey of Ilcinkas, which compares the three schedulers for this problem (Ilcinkas 2019).

Claude (Anthropic) was used in cataloguing the problems.

Level	Value	Formal definition
0	**ASYNC**	**ASYNC**: each robot's Look, Compute and Move phases take finite but unbounded, adversarially chosen time; a robot may Look while others are between phases, so it may move on an outdated snapshot. A robot is always seen on a node (moves are instantaneous). The scheduler is fair: every robot is activated infinitely often. Called *CORDA* (with instantaneous moves) in (Lamani et al. 2010) and (Blin et al. 2010).
1	**SSYNC**	**SSYNC**: time is divided into rounds; in each round a fair adversary chooses a non-empty subset of robots, which execute one atomic Look–Compute–Move cycle together. Called *ATOM* in (Lamani et al. 2010).
2	**FSYNC**	**FSYNC**: in every round every robot executes one atomic Look–Compute–Move cycle.

Exploration goal

Terminating exploration of rings by oblivious robots was posed by Flocchini, Ilcinkas, Pelc and Santoro (Flocchini et al. 2013). Exclusive perpetual exploration was posed for robots with memory on partial grids by Baldoni, Bonnet, Milani and Raynal (as reported by Blin et al. 2010) and for oblivious robots on rings without chirality by Blin, Milani, Potop-Butucaru and Tixeuil (Blin et al. 2010); it models patrolling by physical robots that must not collide. The two goals do not imply each other: a terminating algorithm stops, and an exclusive perpetual one never stops but must avoid towers, which terminating algorithms use to mark the end. The survey of Ilcinkas treats the two goals side by side (Ilcinkas 2019).

Level	Value	Formal definition
0	**both goals**	Solve both terminating and exclusive perpetual exploration with the stated resources (possibly with two algorithms). A lower bound for either goal holds here, and a result here gives both. No variant is catalogued here; the value exists because the two goals do not imply each other.
1	**terminating**	From every allowed initial configuration, every node is visited by at least one robot and the robots reach a configuration in which no robot ever moves again. With coin flips, this must happen with probability 1.
1	**exclusive perpetual**	Every robot visits every node infinitely often, no two robots are ever on the same node, and no two robots ever cross the same edge in opposite directions.

Case tree

The root assumes nothing; going down, each edge adds the assumptions written in grey (+) or leaves an axis unassumed. Bold entries are catalogued settings with their status per measure (Robots, Moves) and page; italic entries are branch points, named after the axis on which their branches differ. Axis order: Graph dynamics → Edge direction → Identifiers → Randomness → Scheduler → Exploration goal → Topology.

No assumptions

+ static graph, + undirected

Randomness

Claude (Anthropic) was used in cataloguing the problems.



What is known

The best upper bound against the best lower bound in each catalogued setting, in tree order. The settings follow in detail.

Setting	Robots	Moves	p.
Rings, ASYNC, terminating STANDARD SETTING	● tight 5 vs 4	● tight $O(n)$ vs $\Omega(n)$	168
Grids, ASYNC, terminating	● tight 3 vs 3	● unknown — vs —	170
Rings, SSYNC, terminating	● tight 4 vs 4	● unknown — vs —	170
Rings, probabilistic, SSYNC, terminating	● tight 4 vs 4	● unknown — vs —	171
Rings, ASYNC, exclusive perpetual	● tight 3 vs 3	● unknown — vs —	172

16.7.1 Rings, ASYNC, terminating

Adds + ring

STANDARD SETTING VERIFIED

Five asynchronous robots explore and stop from any configuration without towers when n is not a multiple of 5; three never suffice, four never suffice for even n, and k never suffices when k divides n.

Assumed: graph dynamics: static graph; edge direction: undirected; exploration goal: terminating exploration (explore, then stop); topology: ring or path. **Not assumed:** identifiers: anonymous: all agents are identical; randomness: deterministic; scheduler: asynchronous (ASYNC).

Statement for this setting. The algorithm must work from **every** initial configuration in which each node holds at most one robot; the robots know neither n nor any orientation of the ring. Each Look shows the whole ring and, for each node, whether it holds no robot, one robot or several (global weak multiplicity detection). When a robot whose snapshot is symmetric chooses to move, the adversary chooses the direction (Flocchini et al. 2013).

Claude (Anthropic) was used in cataloguing the problems.

Measure	Lower bound	Best upper bound	Status
Robots	4 (Devismes et al. 2013); 5 (Lamani et al. 2010); *impossible* (Flocchini et al. 2013); $\Omega(\log n)$ (Flocchini et al. 2013)	5 (Lamani et al. 2010)	• tight
	Tight only in the worst case over n: $\Theta(\log n)$ (Flocchini et al. 2013). For a given n: exactly 5 when n is even and not a multiple of 5; 4 or 5 when n is odd and not a multiple of 5 (gap of one robot); for multiples of 5 (five robots impossible) the best algorithm uses the least $k \geq 17$ co-prime with n, against at least 4 robots (at least 6 when $10 \mid n$).		
Moves	$\Omega(n)$ (Lamani et al. 2010)	$O(n)$ (Lamani et al. 2010)	• tight
	$\Theta(n)$ moves with five robots; moves of the $k \geq 17$ algorithm are not analysed.		

Results.

- Devismes et al. (2013) [lower bound]: robots 4. *Bottleneck:* memory. *Holds when:* $k \leq 3$ and every $n > k$ (Corollary 2 of the 2009 report); holds for probabilistic algorithms too; proved under SSYNC, so it holds under ASYNC; a sequential scheduler suffices except for $k = 3$, $n = 4$, settled by enumerating all protocols (Theorem 2); an execution must pass through at least $n - k + 1$ pairwise distinguishable configurations that contain a tower of fewer than k robots (Corollary 1); three robots have only $\lfloor n/2 \rfloor$ such configurations (Lemma 5).
- Lamani et al. (2010) [lower bound]: robots 5. *Bottleneck:* symmetry breaking. *Holds when:* $k = 4$ and even n, deterministic (Lemma 1); initial configuration: four robots with two axes of symmetry; proved under SSYNC: robots with the same snapshot are activated together, so the four robots act as two; for odd n the bound is 4.
- Flocchini et al. (2013) [lower bound]: robots *impossible*. *Bottleneck:* symmetry breaking. *Holds when:* k divides n, $k < n$ (Lemma 2.1); initial configuration: the k robots evenly spaced; holds even under FSYNC: the adversary activates all robots and sends them the same way, so every configuration is evenly spaced and none is terminal; deterministic.
- Flocchini et al. (2013) [lower bound]: robots $\Omega(\log n)$. *Bottleneck:* symmetry breaking. *Holds when:* for infinitely many n: $n = \mathrm{lcm}(1, \ldots, m)$, which every $k \leq m$ divides (Theorem 4.1); consequence of the $k \mid n$ impossibility; deterministic; holds even under FSYNC.
- Lamani et al. (2010) [lower bound]: moves $\Omega(n)$. *Bottleneck:* every node visited. *Holds when:* each of the $n - k$ initially empty nodes must be entered by some robot; stated as trivial in (Lamani et al. 2010).
- Flocchini et al. (2013) [introductory]: robots $O(\log n)$. *Assumes:* $17 \leq k < n$ and $\gcd(n, k) = 1$ (Theorem 3.4); every initial configuration without towers. Theorem 4.1: taking the least $k \geq 17$ co-prime with n gives $O(\log n)$ robots for every n; for prime $n > 17$, 17 robots suffice. Moves are not analysed.
- Lamani et al. (2010) [current best]: robots 5, moves $O(n)$. *Assumes:* $\gcd(n, 5) = 1$; every initial configuration without towers. Introductory and current best for five robots; $O(n)$ moves, which is optimal. Read in the arXiv version (October 2009); the SIROCCO 2010 version was not read.

Open questions.

Q1. Can four robots explore and stop on every ring of odd size n under ASYNC? Four suffice under SSYNC for odd $n > 7$; the question is posed in (Lamani et al. 2010).

Q2. For which pairs (n, k) can k robots explore an n-node ring (Flocchini et al. 2013)? In particular, is exploration impossible whenever $\gcd(n, k) > 1$ and $k < n$, as conjectured in (Lamani et al. 2010)? Impossibility is proved only when k divides n (Flocchini et al. 2013), and for $k = 4$ with even n (Lamani et al. 2010).

Q3. What is the minimum number of robots when n is a multiple of 5? From model checking on rings of fewer than 18 nodes, Bérard et al. conjecture that the algorithm of (Flocchini et al. 2013) is correct for every odd k with $5 \leq k < 17$ from initial configurations with at most one axis of symmetry (Bérard et al. 2016); the lower bounds use more symmetric starts.

Notes. Flocchini, Ilcinkas, Pelc and Santoro posed the problem and showed that robots without memory or communication can still explore a ring: the configuration itself records the phase (set-up, tower

Claude (Anthropic) was used in cataloguing the problems.

creation, exploration) (Flocchini et al. 2013). Lamani, Potop-Butucaru and Tixeuil reduced the team to five robots with the same three-phase idea (a block of five, one tower, one walker) (Lamani et al. 2010). The lower bounds are proved under SSYNC (fewer than five robots) or FSYNC ($k \mid n$), so they hold here; all of them use a symmetric or evenly spaced start, which the problem allows. The four-robot probabilistic algorithm sometimes attributed to (Lamani et al. 2010) is due to Devismes, Petit and Tixeuil and needs atomic cycles (see the probabilistic SSYNC setting) (Devismes et al. 2013). Ilcinkas reports that the lower bounds hold even with global strong multiplicity detection, while the algorithms use only weak detection (Ilcinkas 2019). As reported by (Ilcinkas 2019), the $\Omega(\log n)$ bound remains true for probabilistic algorithms under ASYNC, citing the journal version of (Devismes et al. 2013); this is unverified here (the journal version was not read, and the 2009 report read does not contain it). Bérard et al. model-checked the set-up phase of the $k \geq 17$ algorithm for $17 \leq k \leq 21$, $n \leq 22$ and found it correct (Bérard et al. 2016).

16.7.2 Grids, ASYNC, terminating

Adds + grid VERIFIED

Three robots explore any grid with at least three nodes (four for 2×2, five for 3×3), which is optimal.

Assumed: graph dynamics: static graph; edge direction: undirected; exploration goal: terminating exploration (explore, then stop); topology: grid. **Not assumed:** identifiers: anonymous: all agents are identical; randomness: deterministic; scheduler: asynchronous (ASYNC).

Measure	Lower bound	Best upper bound	Status
Robots	3 (Devismes et al. 2021) Three.	3 (Devismes et al. 2021)	• tight
Moves	none known —	—	• unknown

Results.

- Devismes et al. (2021) [lower bound]: robots 3. *Bottleneck:* symmetry. *Holds when:* grids with at least 3 nodes, deterministic or probabilistic.
- Devismes et al. (2021) [current best]: robots 3. Introductory and current best.

16.7.3 Rings, SSYNC, terminating

Adds deterministic, + SSYNC, + terminating, + ring VERIFIED

With atomic cycles, four robots explore every ring of odd size $n > 7$ and stop; fewer than four never suffice, and four do not suffice for even n.

Assumed: graph dynamics: static graph; edge direction: undirected; scheduler: semi-synchronous (SSYNC); exploration goal: terminating exploration (explore, then stop); topology: ring or path. **Not assumed:** identifiers: anonymous: all agents are identical; randomness: deterministic.

Statement for this setting. As the ASYNC setting, from every initial configuration without towers and with global weak multiplicity detection, but each activated robot executes its whole Look–Compute–Move cycle atomically.

Measure	Lower bound	Best upper bound	Status
Robots	4 (Devismes et al. 2013); 5 (Lamani et al. 2010); *impossible* (Flocchini et al. 2013); $\Omega(\log n)$ (Flocchini et al. 2013)	4 (Lamani et al. 2010)	• tight
	Worst case over n: $\Theta(\log n)$, as in the ASYNC setting. For a given n: exactly four for odd $n > 7$; exactly five for even n not a multiple of 5 (the ASYNC algorithm of Lamani et al. 2010); for $10 \mid n$ at least six, against the least $k \geq 17$ co-prime with n (Flocchini et al. 2013).		
Moves	none known	—	• unknown
	Moves of the four-robot SSYNC algorithm are not analysed; $\Omega(n)$ moves are needed, and five robots use $O(n)$ moves when n is not a multiple of 5 (Lamani et al. 2010).		

Results.

- Devismes et al. (2013) [lower bound]: robots 4. *Bottleneck:* memory. *Holds when:* $k \leq 3$, every $n > k$, even for probabilistic algorithms (Corollary 2 of the 2009 report).
- Lamani et al. (2010) [lower bound]: robots 5. *Bottleneck:* symmetry breaking. *Holds when:* even n, $k = 4$ (Lemma 1); initial configuration with two axes of symmetry.
- Flocchini et al. (2013) [lower bound]: robots *impossible*. *Bottleneck:* symmetry breaking. *Holds when:* k divides n, $k < n$ (Lemma 2.1); initial configuration: the k robots evenly spaced; holds even under FSYNC.
- Flocchini et al. (2013) [lower bound]: robots $\Omega(\log n)$. *Bottleneck:* symmetry breaking. *Holds when:* for infinitely many n (Theorem 4.1); consequence of the $k \mid n$ impossibility, which holds even under FSYNC.
- Lamani et al. (2010) [current best]: robots 4. *Assumes:* odd $n > 7$; every initial configuration without towers. Algorithm Exploration-odd, in the appendix of the arXiv version (Lemma 14); (Ilcinkas 2019) also cites it from there. Introductory and current best. Whether the appendix is in the SIROCCO 2010 version is not known (not read).

Open questions.

Q1. What is the minimum number of robots under SSYNC when n is a multiple of 10 (four and five robots are impossible)? This is the SSYNC case of the characterisation of solvable pairs (n, k) asked for in (Flocchini et al. 2013).

Notes. The appendix algorithm of (Lamani et al. 2010) forms one tower from a symmetric or asymmetric configuration and lets one or two robots walk until a terminal pattern appears. Moves of this algorithm are not analysed. Under FSYNC, Ilcinkas reports that only three robots are a clear lower bound for every n, in particular for odd n (Ilcinkas 2019).

16.7.4 Rings, probabilistic, SSYNC, terminating

Adds + private coins, + SSYNC, + terminating, + ring VERIFIED

Four robots that flip coins explore any ring of more than four nodes and stop with probability 1; three never suffice, whatever n.

Assumed: graph dynamics: static graph; edge direction: undirected; randomness: randomized (agents have private coins); scheduler: semi-synchronous (SSYNC); exploration goal: terminating exploration (explore, then stop); topology: ring or path. **Not assumed:** identifiers: anonymous: all agents are identical.

Statement for this setting. As the SSYNC setting, but a robot may flip coins in Compute. From every initial configuration without towers, every execution must visit every node, and the robots must reach a configuration in which no robot moves again with probability 1 (Devismes et al. 2013).

Claude (Anthropic) was used in cataloguing the problems.

Measure	Lower bound	Best upper bound	Status
Robots	4 (Devismes et al. 2013)	4 (Devismes et al. 2013)	• tight
	Four, for every $n > 4$ (Devismes et al. 2013 for $n > 8$, Devismes 2010 for $4 < n \leq 8$).		
Moves	none known	—	• unknown
	Only finite expected termination is proved; no bound on moves or rounds is given (Devismes et al. 2013).		

Results.

- Devismes et al. (2013) [lower bound]: robots 4. *Bottleneck:* memory. *Holds when:* $k \leq 3$, every $n > k$ (Corollary 2 of the 2009 report); proved with a sequential scheduler, except $k = 3$, $n = 4$ (Theorem 2).
- Devismes et al. (2013) [current best]: robots 4. *Assumes:* $n > 8$ (Theorem 3 of the 2009 report); every initial configuration without towers; no condition linking n and k. Introductory and current best. Read in INRIA report RR-6838 (2009), which calls the robots asynchronous but defines atomic cycles; the TCS 2013 version was not read. Rings of five to eight nodes are covered by (Devismes 2010), so four probabilistic robots suffice for every $n > 4$, as Ilcinkas also reports (Ilcinkas 2019).
- Devismes (2010) [also]: robots 4. *Assumes:* $4 < n \leq 8$ (Theorem 5); every initial configuration without towers; multiplicity detection: a robot sees whether a node holds no robot, one robot or several. The same model (atomic Look–Compute–Move cycles, oblivious anonymous robots without a common orientation), with the rings that (Devismes et al. 2013) leaves out; each size from 5 to 8 is checked separately. Read in the author's copy.

Open questions.

Q1. How many moves (or rounds) do four probabilistic robots need to explore a ring of n nodes and stop, and how does the time decrease when more robots are used? Posed in (Devismes et al. 2013), where termination is only shown to be finite.

Notes. Randomization removes the number-theoretic conditions of the deterministic setting: four robots suffice for every $n > 4$ (Devismes et al. 2013; Devismes 2010), whereas deterministic robots need $\Omega(\log n)$ for infinitely many n even under FSYNC (Flocchini et al. 2013). The robots first gather on four consecutive nodes, using coin flips to break symmetry, then build an *arrow* (a tower next to the block) whose tail robot walks around the ring (Devismes et al. 2013). Under ASYNC, coin flips do not help in the worst case: $\Omega(\log n)$ robots are needed for infinitely many n, as reported by (Ilcinkas 2019) (not verified; see the ASYNC setting).

16.7.5 Rings, ASYNC, exclusive perpetual

Adds deterministic, ASYNC, + exclusive perpetual, + ring VERIFIED

Three asynchronous robots explore a ring of $n \geq 10$ nodes perpetually without collisions when $3 \nmid n$; one or two robots never can, and three cannot when $n < 10$.

Assumed: graph dynamics: static graph; edge direction: undirected; exploration goal: exclusive perpetual exploration; topology: ring or path. **Not assumed:** identifiers: anonymous: all agents are identical; randomness: deterministic; scheduler: asynchronous (ASYNC).

Statement for this setting. From every initial configuration without towers, every robot must visit every node infinitely often, no two robots may ever be on one node, and no two robots may ever cross one edge in opposite directions. Robots have no common chirality (no common sense of clockwise) (Blin et al. 2010).

Claude (Anthropic) was used in cataloguing the problems.

Measure	Lower bound	Best upper bound	Status
Robots	3 (Blin et al. 2010); *impossible* (Blin et al. 2010); *impossible* (Ilcinkas 2019)	3 (Bérard et al. 2016)	• tight
	Three, for $n \geq 10$ with $3 \nmid n$ (two robots never suffice). Three robots are impossible for $n < 10$ (claimed; proofs for $n = 7, 8$ omitted in the report read) and for $3 \mid n$ when $n > 6$ (Lemma 1 of Ilcinkas 2019).		
Moves	none known	—	• unknown
	Not a measure of perpetual exploration in these papers; no bound on the time between visits is catalogued.		

Results.

- Blin et al. (2010) [lower bound]: robots 3. *Bottleneck:* symmetry breaking. *Holds when:* $k \leq 2$, $n \geq 3$: one robot (Lemma 12) and two robots (Corollary 1) in the 2010 report; a single robot always has a symmetric snapshot, so the adversary moves it back and forth between two nodes.
- Blin et al. (2010) [lower bound]: robots *impossible*. *Bottleneck:* symmetry breaking. *Holds when:* $k = 3$ and $n < 10$ (Theorem 1); in the 2010 report the proof is given for $n = 4, 5$, the cases $n = 7, 8$ are omitted, and $n = 3, 6, 9$ are referred to the $k \mid n$ case; $(n - k) \mid n$ with $n - k > 2$ (evenly spaced empty nodes, Lemma 5), and $k \in \{n - 4, n - 3, n - 2\}$ from specific symmetric starts (Lemmas 4, 6, 7).
- Ilcinkas (2019) [lower bound]: robots *impossible*. *Bottleneck:* symmetry breaking. *Holds when:* k divides n and $k < n/2$, $n \geq 3$ (Lemma 1 of the survey, inspired by Lemma 2.1 of Flocchini et al. 2013); initial configuration: the k robots evenly spaced; holds even under FSYNC, with global strong multiplicity detection and for robots with memory; deterministic.
- Blin et al. (2010) [introductory]: robots 3. *Assumes:* $n \geq 10$, $3 \nmid n$; every initial configuration without towers. Not correct as published: model checking found a collision under ASYNC for $n = 10$ (Bérard et al. 2016), also found by Doan, Bonnet and Ogata (Doan et al. 2016) (as reported by Ilcinkas 2019); under FSYNC the algorithm was verified collision-free for $n > 10$ (Sangnier et al. 2020). Read in the 2010 research report; the DISC 2010 version was not read.
- Bérard et al. (2016) [current best]: robots 3. *Assumes:* $n \geq 10$, $\gcd(n, 3) = 1$; every initial configuration without towers. Corrects rule RC5 of (Blin et al. 2010); model-checked under ASYNC for $n \in \{10, 11, 13, 14, 16\}$ and proved for every such n (Theorems 2 and 3: from a legitimate configuration the ring is explored perpetually, and a legitimate configuration is always reached). The inductive proof was read for its statements only.
- D'Angelo et al. (2015) [also]: robots $5 \leq k \leq n - 3$. *Assumes:* rigid initial configuration (no non-trivial symmetry), without towers; $n \geq 10$, $(k, n) \neq (5, 10)$. Theorems 6 and 7: the same algorithm also clears the ring perpetually (graph searching). Large teams are possible from rigid starts, so the impossibilities for $k \geq n - 4$ in (Blin et al. 2010) come from symmetric starts.

Open questions.

Q1. What is the minimum number of robots for exclusive perpetual exploration of a ring whose size is a multiple of 3, from every configuration without towers?

Q2. Would coin flips help in the cases where deterministic exclusive perpetual exploration is impossible? Posed in (Blin et al. 2010).

Q3. What is the largest team that can explore exclusively and perpetually from every configuration without towers? (Blin et al. 2010) claim $n - 5$ robots for odd $k \geq 7$ with $k \nmid n$, but an SMT check found collisions in their algorithm for 7 and 9 robots on rings of $k + 5$ nodes under FSYNC (Sangnier et al. 2020).

Notes. Blin, Milani, Potop-Butucaru and Tixeuil posed the problem for oblivious robots without chirality; their three robots cycle through three legitimate configurations, each round shifting all robots one node (Blin et al. 2010). Bérard et al. found a collision in that algorithm by model checking under ASYNC: a move computed from an outdated snapshot meets a later move (Bérard et al. 2016). Ilcinkas reports, from the DISC version of (Blin et al. 2010), that the problem cannot be solved with an even

Claude (Anthropic) was used in cataloguing the problems.

number of robots, even under FSYNC (Ilcinkas 2019); this claim was not found in the 2010 report and is unverified. From rigid starts, D'Angelo et al. handle every $5 \le k \le n-3$ for $n \ge 10$ except $(k, n) = (5, 10)$ (D'Angelo et al. 2015); Bonnet et al. enumerated all SSYNC protocols for five robots on a ring of 10 nodes and found none (as reported by Ilcinkas 2019; abstract of Bonnet et al. 2014 read). In the text extraction of (D'Angelo et al. 2015) the relation symbol in the range of n in Theorem 6 is lost; $n \ge 10$ is taken from its contribution paragraph and Theorem 7.



16.8 Exploration by luminous robots

Myopic Look–Compute–Move robots with a light of a few colours must visit every node of a ring, grid or torus, and stop or keep visiting; how many robots and colours are needed?

First posed in 2018 (Ooshita and Tixeuil 2022). Ring exploration by myopic robots with a two-colour light (SSS 2018).

Problem statement and measures

k anonymous robots, each with a **light** that shows one of c colours, are placed on an anonymous graph $G = (V, E)$ (a ring, a finite grid, a torus or a triangular grid) in an initial configuration designated by the algorithm (positions and colours). Each robot repeats Look–Compute–Move cycles. **Look:** it sees every node within distance φ, with the multiset of colours of the robots on each (so it sees **towers**, nodes with several robots), and its own colour. **Compute:** a deterministic function of this snapshot, the same for all robots, gives a new colour and a destination, its own node or a neighbour; the robot sees no identifiers, no ports, and (without chirality) no orientation. **Move:** it moves there. A fair adversarial scheduler (FSYNC, SSYNC or ASYNC) activates the robots. **Terminating exploration:** every node is visited by some robot and eventually no robot is enabled. **Perpetual exploration:** every node is visited infinitely often. With finite φ no algorithm works from every initial configuration, so each result names its initial configurations; some papers also require *exclusive* executions (no two robots on one node or one edge) or opaque robots, recorded under each result. **Measures:** the number of robots k and the number of colours c.

Measures: **Robots**, number of robots k needed; **Colours**, number of colours c of each light; $c = 1$ means no usable light (oblivious robots).

A first look

Imagine a circular race track of n squares and a few identical toy robots on it. The track is a **ring**: its squares are its n **nodes**, and two squares side by side are **neighbours**. The robots are **myopic**: each sees only its own node and its two neighbours. They cannot talk, and they are **oblivious**: they forget everything after each cycle. Each robot carries a **light** (think of a small lamp) that can glow in one of two colours, say green (G) or white (W). The light stays on until its owner switches it, and every robot that sees the node, the owner included, sees the colour. Each robot works in **Look–Compute–Move cycles**: **Look** (take a **snapshot** of its own node and its two neighbours, with the robots and light colours on them), **Compute** (decide from that snapshot alone a new colour and whether to move to one neighbour, move to the other, or stay) and **Move**. The nodes have no names and the ring has no left or right, and an **adversary** (an imagined opponent) decides when each robot acts. In the basic version all robots act together in **rounds** (**FSYNC**, fully synchronous): k robots start on neighbouring nodes in a colour pattern chosen by the algorithm (the initial **configuration**), and must visit every node and then all stop. We count the robots k and the colours c.

Claude (Anthropic) was used in cataloguing the problems.



(a) Start: three white robots W W W on neighbouring nodes

(b) The pair meets the waiting G: W joins the front robot; no rule applies, all stop

Figure 16.8: Terminating ring exploration by myopic luminous robots in FSYNC, Algorithm 2 of (Ooshita and Tixeuil 2022): three robots with visibility one and two colours (G green, W white) start as W W W on an anonymous ring ($n = 8$). In the first round both end robots switch their lights to G; then the middle robot moves towards one end (the adversary picks which), forming a pair G W that moves round the ring, while the other G waits as a lighthouse. When the pair reaches it from the other side, every node has been visited; the white robot forms a tower with the front robot and no rule applies any more.

1. Algorithm: A coloured pair moves in step. (Ooshita and Tixeuil 2022) (Rings, FSYNC, perpetual, $\varphi = 1$, p. 194) Two robots stand on neighbouring nodes, one green and one white (Ooshita and Tixeuil 2022).

1. The green robot sees white on one side and nothing on the other: it moves away from the white robot.
2. The white robot sees green on one side and nothing on the other: it moves towards the green robot.
3. Both act in the same round, so after the move they are again on neighbouring nodes, green in front, white behind, and the snapshot each takes is the same as before. They repeat forever.

Why it works and what it costs. The pair moves one node per round in a fixed direction, so it passes every node once every n rounds: perpetual exploration with two robots and two colours. One robot cannot do it: it sees nobody, its snapshot looks the same both ways, and the adversary picks its direction each time, keeping it on two nodes. Two robots of the same colour cannot either: they take mirror-image snapshots and act as mirror images. So two robots and two colours are optimal (Ooshita and Tixeuil 2022).

Why it matters. The colours give the pair a front and a back, which the ring itself does not have: the light carries the direction, both as memory and as message.

2. Algorithm: A moving pair and a lighthouse. (Ooshita and Tixeuil 2022) (Rings, FSYNC, terminating, $\varphi = 1$, p. 198) Three robots stand on three consecutive nodes coloured G, W, G (Ooshita and Tixeuil 2022).

1. The middle robot cannot tell its sides apart and moves towards one end robot (the adversary picks which); that end robot moves away from it, and the other end robot moves away too. Now a green–white pair stands on one side and a lone green robot, the **lighthouse**, on the other, with two empty nodes between them.
2. The lighthouse sees nobody and has no rule, so it waits. The pair moves away from it as in the first idea.
3. After going round the ring the pair arrives at the lighthouse from the other side. The front robot now sees a green robot ahead and stops; the white robot moves onto its node, and no rule applies

Claude (Anthropic) was used in cataloguing the problems.

any more.

Why it works and what it costs. The pair visits each node on its way round, so all n nodes are visited in about n rounds, with three robots and two colours. The lighthouse is a message that stays put: meeting it tells the pair that the trip is complete. Without lights five robots are needed (Datta et al. 2013).

Why it matters. Stopping is the hard part of exploration with little memory; here a robot that never moves plays the role of a marker left at the start.

3. Impossibility: Robots that lose sight of each other are lost. (Ooshita and Tixeuil 2022) (Rings, ASYNC, perpetual, $\varphi = 1$, p. 197) Take two robots, any number of colours, and an adversary that may activate one robot at a time (**SSYNC**, semi-synchronous) (Ooshita and Tixeuil 2022).

1. To make progress, some robot must at some moment move away from the other.
2. The adversary activates that robot alone. Now they are two nodes apart and cannot see each other.
3. A robot that sees nobody takes a symmetric snapshot, so whenever it moves the adversary chooses the direction: it sends the robot back and forth between two nodes, its *territory*. Each robot stays in its territory forever.

Why it holds. So from then on the two robots stay on four nodes, whatever their lights. Lights do not help because a robot that sees nobody reads no message, and its own colour tells it nothing about the direction. Three robots suffice in SSYNC and ASYNC (Ooshita and Tixeuil 2022).

Why it matters. This territory argument is behind most lower bounds for myopic robots: a light is a message only for robots within sight.

4. Lower bound: Two robots cannot measure the ring. (Ooshita and Tixeuil 2022) (Rings, FSYNC, terminating, $\varphi = 1$, p. 198) Suppose two robots explore a small ring of n_1 nodes and stop, or separate, at round t (Ooshita and Tixeuil 2022). Run the same robots on a ring of more than $2(t + 1)$ nodes. While they stay together they take exactly the same snapshots as before, so they do exactly the same things.

Why it holds. On the big ring they stop (or separate, and are then stuck in their territories) after visiting at most $2(t + 1)$ nodes, fewer than all of them. This holds for any number of colours, so terminating exploration needs three robots even under FSYNC, one more than perpetual exploration.

Why it matters. A robot that sees only nearby nodes cannot know how big the graph is, so stopping needs a landmark, such as the lighthouse of the second idea.

What the community started looking at

Where it started. Oblivious robots on graphs see every robot and remember nothing, so the configuration itself must record progress; the first question was how many robots explore a ring (Flocchini et al. 2013; Lamani et al. 2010) or a grid (Devismes et al. 2021). Datta, Lamani, Larmore and Petit made the robots *myopic*: with visibility $\varphi = 1$, five robots explore a ring and stop under FSYNC, and no number below n can under SSYNC or ASYNC (Datta et al. 2013). Ooshita and Tixeuil then gave each myopic robot the smallest possible light, two colours: three robots then suffice under FSYNC and four under ASYNC, and perpetual exploration needs only two and three (Ooshita and Tixeuil 2022). This showed that "there exist some tasks that myopic luminous robots with small visibility can achieve, but that non-luminous robots with larger visibility cannot" (Ooshita and Tixeuil 2022). The community then traded robots against colours, visibility and chirality: rings with $\varphi \geq 2$ (Nagahama et al. 2023), perpetual exploration of grids with chirality (Bramas et al. 2023a) and without (Rauch et al. 2021), terminating

Claude (Anthropic) was used in cataloguing the problems.

exploration of grids (Nagahama et al. 2022), tori (Darwich et al. 2023), asynchronous grids (Bramas et al. 2026a), and crash faults on rings (Ooshita et al. 2024, 2025).

A light is a message and a memory. Think of a light as a flag that a robot raises on its own mast. It is a *message*: every robot within sight can read it, without being addressed, and the robot that raised it does not know who has read it, so it is a broadcast to the neighbourhood that nobody acknowledges. It *persists*: it stays up until its owner changes it, so a robot that looks much later still reads it, and in ASYNC it may read a flag that is about to change. And it is a *sticky note to oneself*: an oblivious robot forgets everything at the end of a cycle, but in the next cycle it sees its own flag and so recalls $\log_2 c$ bits. The light models separate the two roles (Flocchini et al. 2016a): a flag that only its owner can see is memory without messages (FSTA, an internal light); a flag on the robot's back that only the others can see is a message without memory (FCOM, an external light); a flag that everybody, owner included, can see is both (LUMI, a full light) (Das et al. 2016), which is the model of every exploration result here. Two consequences of the picture are worth keeping. First, a LUMI robot's memory is public: whatever it remembers, its neighbours know too. Second, positions are messages as well, and lights were introduced to supplement them (Peleg 2005); on a graph a myopic robot sees so little that the light is often its only memory, which is why two colours change the number of robots so much (Ooshita and Tixeuil 2022).

Further reading

Papers on this problem that are not yet placed in a setting.

- Bramas et al. (2020b). Infinite grid exploration (every node eventually visited) by synchronous, opaque, myopic luminous robots with chirality: two robots never suffice, five robots with five colours are optimal. Abstract read.
- Bramas et al. (2025a). Infinite grid exploration by synchronous, opaque, myopic luminous robots without chirality: optimal visibility and optimal number of colours cannot be achieved together; eight oblivious robots with visibility range two, six robots with two colours with visibility range one, and, in the journal version, five robots (the minimum) with twelve colours with visibility range one. Abstracts read.
- Bramas et al. (2022). Perpetual exploration of a 3D grid (cuboid) by synchronous myopic luminous robots; two optimal algorithms. Abstract read.
- Bramas et al. (2026b). Perpetual exploration of a finite 3D grid (each side at least 4) by three synchronous luminous robots without chirality, no two on one node. With visibility range $\varphi = 2$, a well-defined (unambiguous) algorithm uses four colours (Theorem 2). With $\varphi = 1$, 77 colours suffice if the adversary resolves ambiguous rules (Theorem 3), and no well-defined algorithm lets three robots explore with any number of colours (Theorem 1); three robots are necessary with $\varphi = 1$ even with chirality (Bramas et al. 2022), as reported. Read.
- Bramas et al. (2026c). Stand-up indulgent gathering (gather, and at the crash site if a robot crashes) on lines by myopic luminous FSYNC robots. Abstract read.
- Bramas et al. (2025b). Stand-up indulgent gathering on lines: oblivious robots need one more hop of visibility than robots with one bit of persistent memory, a visibility–memory trade-off on graphs. Abstract read.
- Hideg and Lukovszki (2020). Filling an arbitrary graph through a door by ASYNC myopic luminous robots ($\varphi = 1$, $\Delta + 4$ colours, $O(\log \Delta)$ bits) in $O(n^2)$ epochs. Abstract read.
- Pramanick et al. (2023). MIS filling of an arbitrary graph from one door (ASYNC, $\varphi = 3$, $\Delta + 8$ colours) or several doors (SSYNC). Abstract read.
- Kamei and Tixeuil (2024). Maximum independent set on grids by ASYNC myopic luminous robots entering from a corner: three colours with $\varphi = 2$ and port numbers, or seven colours with $\varphi = 3$. Abstract read.
- Das et al. (2025b). Maximum independent set formation on a finite grid from an arbitrary placement, SSYNC, $\varphi = 2$, three colours. Abstract read.

Claude (Anthropic) was used in cataloguing the problems.

- Datta et al. (2015). Full version of the $\varphi = 2, 3$ results for oblivious myopic robots under ASYNC: with $\varphi = 3$, five robots are necessary and suffice from consecutive nodes, seven from any configuration connected by vision; with $\varphi = 2$, seven robots. Abstract read; not checked against the paper.

- Jana et al. (2025b). Filling a minimal vertex cover (the complement of a maximal independent set) of an arbitrary graph through one or more entry nodes by ASYNC myopic luminous robots: impossible with $\varphi = 1$ (Theorem 1); $\Omega(m)$ epochs from one entry node (Theorem 2); $O(m)$ epochs from one entry node with $\varphi = 3$, 4 colours and $O(\log \Delta)$ bits (Theorem 5), and from several with $\varphi = 4$ and a number of colours linear in the number of entry nodes (Theorem 6); a minimum vertex cover of a tree from one entry node with $\varphi = 2$ and 4 colours in $O(m)$ epochs (Theorem 3). Read.

- Altisen et al. (2026). Perpetual exploration of a finite grid that contains one obstacle (a missing node or a pole the robots must go around) by two myopic luminous robots with common chirality under FSYNC. A partly automated method starts from two algorithms of (Bramas et al. 2023a), synthesises 5 algorithms for visibility 1 with three colours and 115 for visibility 2 with two colours, checks them by simulation on small grids, and proves them for larger grids by induction on the grid size. Read.

Relations to other problems

- **Variant of Exploration by oblivious robots.** The same Look–Compute–Move robots, but each carries a light with a constant number c of colours that acts both as persistent memory (the robot sees its own colour in the next cycle) and as a message (every robot within range reads it); in exchange, robots usually see only within a fixed distance φ. An oblivious algorithm is a luminous one with $c = 1$, so algorithms of the parent problem with myopic vision carry over, and lower bounds proved here for any number of colours hold for oblivious robots too (Ooshita and Tixeuil 2022).

- **Related: Exploration of recolourable graphs.** Flag on the robot versus flag on the node: a luminous robot carries its coloured light with it and shows it to the robots in sight, whereas here the colour stays planted in the node and is read by whoever arrives.

Assumptions specific to this problem

Scheduler

The three schedulers come from robots in the plane (see the plane model); on graphs the asynchronous one was used from the first ring exploration paper (Flocchini et al. 2013), and Datta, Lamani, Larmore and Petit compared all three for myopic robots (Datta et al. 2013). For myopic robots on graphs the scheduler decides solvability: oblivious robots that see one hop can explore a ring with five robots under FSYNC but not at all under SSYNC or ASYNC (Datta et al. 2013), and with two-colour lights the robots needed for perpetual (terminating) exploration of a ring are two (three) under FSYNC against three (four) under SSYNC and ASYNC (Ooshita and Tixeuil 2022). Ooshita and Tixeuil ask whether SSYNC and ASYNC still differ for myopic luminous robots, since with full visibility lights make them equivalent (Ooshita and Tixeuil 2022; Das et al. 2016). Most grid and torus results are for FSYNC only; asynchronous perpetual grid exploration is the subject of (Bramas et al. 2026a).

Claude (Anthropic) was used in cataloguing the problems.

Level	Value	Formal definition
0	**ASYNC**	**ASYNC**: each robot's Look, Compute and Move phases take finite but unbounded, adversarially chosen time; a robot may Look while others are between phases, so it may move on an outdated snapshot. A robot is always seen on a node (an edge traversal is instantaneous), and its new colour is visible from the end of its Compute phase. The scheduler is fair: every robot is activated infinitely often. Called *CORDA* in (Datta et al. 2013).
1	**SSYNC**	**SSYNC**: time is divided into rounds; in each round a fair adversary chooses a non-empty subset of robots, which execute one atomic Look–Compute–Move cycle together. Called *ATOM* in (Datta et al. 2013).
2	**FSYNC**	**FSYNC**: in every round every robot executes one atomic Look–Compute–Move cycle.

Lights (memory and messages)

Peleg suggested giving robots \"colors, flags or visible indicator lights\" and $O(1)$ bits of stable memory (Peleg 2005); Das, Flocchini, Prencipe, Santoro and Yamashita made this the LUMI model, a persistent register visible to all (Das et al. 2016), and Flocchini, Santoro, Viglietta and Yamashita split the light into its two roles: FSTA, a light only its owner sees (memory), and FCOM, a light only the others see (communication) (Flocchini et al. 2016a). On graphs, Ooshita and Tixeuil gave myopic robots a light (the first study of lights for myopic robots, by their account), noting that it \"can be used as a constant-space memory\" and that robots communicate \"by changing their color and moving\" (Ooshita and Tixeuil 2022). Every exploration result found for graphs uses full lights; the number of colours c is a measure, and an algorithm with $c = 1$ is an oblivious one. The scale orders the models by what they assume: OBLOT assumes no light; FSTA and FCOM each assume one role of the light and do not imply each other; LUMI assumes both.

Level	Value	Formal definition
0	**oblivious**	**OBLOT**: a robot's Compute phase depends only on the current snapshot; nothing persists across cycles and robots communicate only by moving. Equivalent to a light with $c = 1$ colour.
1	**FSTA**	**FSTA** (*finite-state*): each robot has a light with c colours that only it can see. The colour is set at the end of Compute and persists until the robot changes it: $\log_2 c$ bits of persistent memory, no communication (Flocchini et al. 2016a).
1	**FCOM**	**FCOM** (*finite-communication*): each robot has a light with c colours that the other robots within distance φ can see but the robot itself cannot; the robot stays oblivious, but every robot that looks reads its last colour: $\log_2 c$ bits broadcast to the neighbourhood, no memory (Flocchini et al. 2016a).
2	**luminous**	**LUMI** (*luminous* robots): each robot has a light with c colours, set at the end of Compute, persistent across cycles, and visible to the robot itself and to every robot within distance φ (Das et al. 2016). It is both memory and message. A Look returns, for every node within distance φ, the multiset of colours of the robots on it.

Visibility range

Myopia, a fixed visibility range φ, replaces the global snapshot of the oblivious-robot literature, which

Claude (Anthropic) was used in cataloguing the problems.

\"somewhat contradicts the principle of very weak mobile entities\" (Ooshita and Tixeuil 2022). Datta, Lamani, Larmore and Petit studied ring exploration by oblivious robots with $\varphi = 1, 2, 3$ (Datta et al. 2013); Ooshita and Tixeuil chose \"the weakest assumption for myopia\", $\varphi = 1$ (Ooshita and Tixeuil 2022), and later work compares $\varphi = 1$ with $\varphi \geq 2$ (Ooshita et al. 2024; Nagahama et al. 2022). With a finite range no algorithm explores from every initial configuration (robots that see nobody cannot coordinate), so algorithms start from designated or locally defined configurations (Datta et al. 2013; Bramas et al. 2023a). Unlimited visibility is the setting of the parent problem.

Level	Value	Formal definition
0	$\varphi = 1$	In a Look phase a robot at node v sees every node at distance at most $\varphi = 1$ from v (in the graph metric), with the robots on it, their colours, and how many robots each node holds. It sees no node identifiers and, without chirality, no orientation.
1	$\varphi = 2$	As for $\varphi = 1$, but the robot sees every node at distance at most $\varphi = 2$. An algorithm for $\varphi = 2$ works for every larger φ, since a robot can ignore what lies further away.
2	$\varphi = 3$	As for $\varphi = 1$, but the robot sees every node at distance at most $\varphi = 3$.

Chirality

Chirality was mostly assumed in the plane; an early use on graphs is by Bramas, Devismes and Lafourcade, for infinite grid exploration (Bramas et al. 2020b) and for perpetual exploration of finite grids (Bramas et al. 2023a), noting that only \"a few works dedicated to discrete environments\" assume it and that it changes the number of robots needed (Bramas et al. 2023a). Nagahama, Ooshita and Inoue give terminating grid algorithms with and without it (Nagahama et al. 2022); without it, two robots cannot explore a grid perpetually with $\varphi = 1$ and three robots with three colours can, under FSYNC (Rauch et al. 2021). Ring results assume an unoriented ring.

Level	Value	Formal definition
0	**no chirality**	Robots share no orientation: on a ring a robot cannot tell its two directions apart; on a grid or torus its snapshot is known only up to rotation and reflection, and the adversary may choose a different transformation in every cycle (*self-inconsistent* robots, as in Darwich et al. 2023 and Bramas et al. 2026a).
1	**chirality**	All robots agree on the clockwise direction: a snapshot is known up to rotation only (no reflection), so a robot on an axis of symmetry of its view can still tell its two sides apart. Robots still share no North.

Exploration goal

Both goals come from the oblivious-robot literature: terminating exploration of rings (Flocchini et al. 2013) and grids (Devismes et al. 2021), and perpetual (often exclusive) exploration, which models patrolling (Bramas et al. 2023a). Ooshita and Tixeuil studied both for myopic luminous robots (Ooshita and Tixeuil 2022). The two goals are incomparable: a terminating algorithm stops, and a perpetual one need never detect that it is done, so the scale puts them side by side above \"both\". In this literature the impossibility proofs for perpetual exploration show that some node is never visited, which then also rules out terminating exploration (Nagahama et al. 2022), and perpetual exploration never needed more robots than terminating exploration.

Claude (Anthropic) was used in cataloguing the problems.

Level	Value	Formal definition
0	**both goals**	Solve both terminating and perpetual exploration with the stated resources (possibly with two algorithms). A lower bound for either goal holds here, and a result here gives both. No variant is catalogued here; the value exists because the two goals do not imply each other.
1	**terminating**	Every node is visited by at least one robot, and from some time on no robot is enabled (no robot changes its position or colour again).
1	**perpetual**	Every node is visited by at least one robot infinitely often; robots never stop.

Network

Rings were the first graphs explored by oblivious robots (Flocchini et al. 2013) and by myopic ones (Datta et al. 2013; Ooshita and Tixeuil 2022). Finite grids followed for myopic luminous robots in perpetual (Bramas et al. 2023a) and terminating (Nagahama et al. 2022) form, and tori, whose missing border removes the landmark that grid algorithms use to turn, in (Darwich et al. 2023). Das, Goswami, Mondal and Sau explored the rectangle-enclosed triangular grid, noting that square-grid algorithms do not apply directly because its corner and border nodes have other degrees, and motivating it by programmable matter and by sensor coverage (Das et al. 2025c). Infinite grids and 3D grids are studied in papers listed under further reading. This axis replaces the model's topology axis because the torus and the triangular grid are not among the model's values.

Level	Value	Formal definition
0	**any graph**	A finite, connected, anonymous graph $G = (V, E)$ with n nodes and no port numbers visible to the robots.
1	**ring**	An anonymous, unoriented ring of n nodes; n is unknown to the robots.
1	**grid**	An $a \times b$ grid ($n = ab$ nodes; nodes (i, j), edges between nodes at Manhattan distance one); anonymous, with a and b unknown. A robot sees the border of the grid when it is within range, since nodes beyond the border are missing from its snapshot.
1	**torus**	An $a \times b$ torus: the grid with wrap-around edges in both dimensions, so every node has degree four and there is no border.
1	**triangular grid**	The subgraph of the infinite triangular grid (every node of degree six, every face an equilateral triangle of unit side) induced by the nodes on or inside a rectangle, one pair of whose sides is parallel to a family of grid lines; the lines parallel to those sides are its *columns*. It is anonymous and its dimensions are unknown. Corner nodes have degree two or three and other border nodes degree three, four or five, so the border is seen from within range as on a square grid, but with more cases.

Claude (Anthropic) was used in cataloguing the problems.

Case tree

The root assumes nothing; going down, each edge adds the assumptions written in grey (+) or leaves an axis unassumed. Bold entries are catalogued settings with their status per measure (Robots, Colours) and page; italic entries are branch points, named after the axis on which their branches differ. Axis order: Graph dynamics → Edge direction → Network → Exploration goal → Lights (memory and messages) → Scheduler → Visibility range → Chirality → Faults → Identifiers.

No assumptions

+ static graph, + undirected
Network

+ grid
Exploration goal

+ terminating, + luminous
Scheduler

+ FSYNC
Visibility range

+ $\varphi = 2$
Chirality

+ chirality, + no faults
Grids, FSYNC, terminating, chirality, $\varphi = 2$ ●● p. 185

no chirality, + no faults
Grids, FSYNC, terminating, no chirality, $\varphi = 2$ ●● p. 186

$\varphi = 1$, + chirality, + no faults
Grids, FSYNC, terminating, chirality, $\varphi = 1$ ●● p. 186

$\varphi = 1$, no chirality, + no faults
Grids, FSYNC, terminating, no chirality, $\varphi = 1$ ●● p. 187

ASYNC, + $\varphi = 2$
Chirality

+ chirality, + no faults
Grids, ASYNC, terminating, chirality, $\varphi = 2$ ●● p. 188

no chirality, + no faults
Grids, ASYNC, terminating, no chirality, $\varphi = 2$ ●● p. 188

ASYNC, $\varphi = 1$, + chirality, + no faults
Grids, ASYNC, terminating, chirality, $\varphi = 1$ ●● p. 189

ASYNC, $\varphi = 1$, no chirality, + no faults
Grids, ASYNC, terminating, no chirality, $\varphi = 1$ ●● p. 189

+ perpetual, + luminous
Scheduler

+ FSYNC
Visibility range

$\varphi = 1$, + chirality, + no faults
Grids, FSYNC, perpetual, chirality, $\varphi = 1$ ●● p. 190

+ $\varphi = 2$, + chirality, + no faults
Grids, FSYNC, perpetual, chirality, $\varphi = 2$ ●● p. 190

Claude (Anthropic) was used in cataloguing the problems.

$\varphi = 1$, no chirality, + no faults
Grids, FSYNC, perpetual, no chirality, $\varphi = 1$ ●● p. 191

ASYNC, + $\varphi = 2$
Chirality

+ chirality, + no faults
Grids, ASYNC, perpetual, chirality, $\varphi = 2$ ●● p. 192

no chirality, + no faults
Grids, ASYNC, perpetual, no chirality, $\varphi = 2$ ●● p. 192

ASYNC, $\varphi = 1$, + chirality, + no faults
Grids, ASYNC, perpetual, chirality, $\varphi = 1$ (exclusive) ●● p. 193

ASYNC, + $\varphi = 3$, + no faults
Grids, ASYNC, perpetual, no chirality, $\varphi = 3$ ●● p. 194

+ ring
Exploration goal

+ perpetual, + luminous
Scheduler

+ FSYNC
Faults

+ no faults
Rings, FSYNC, perpetual, $\varphi = 1$ ●● p. 194

+ at most crash faults
Rings, FSYNC, perpetual, up to f crashes ●● p. 195

ASYNC, + $\varphi = 2$
Faults

+ no faults
Rings, ASYNC, perpetual, $\varphi \geq 2$ ●● p. 195

+ at most crash faults
Rings, ASYNC, perpetual, up to f crashes, $\varphi \geq 2$ ●● p. 196

ASYNC, $\varphi = 1$, + no faults
Rings, ASYNC, perpetual, $\varphi = 1$ ●● p. 197

ASYNC, $\varphi = 1$, + at most crash faults
Rings, ASYNC, perpetual, up to f crashes, $\varphi = 1$ ●● p. 197

+ terminating
Lights (memory and messages)

+ luminous
Scheduler

+ FSYNC, + no faults
Rings, FSYNC, terminating, $\varphi = 1$ STANDARD SETTING ●● p. 198

ASYNC, $\varphi = 1$, + no faults
Rings, ASYNC, terminating, $\varphi = 1$ ●● p. 199

ASYNC, + $\varphi = 2$, + no faults
Rings, ASYNC, terminating, $\varphi \geq 2$ ●● p. 199

oblivious, + FSYNC, + no faults
Rings, oblivious, FSYNC, terminating, $\varphi = 1$ ●● p. 200

Claude (Anthropic) was used in cataloguing the problems.

oblivious, ASYNC, + no faults
Rings, oblivious, ASYNC, terminating, $\varphi = 1$ ●● p. 200

+ torus, + perpetual, + luminous, + FSYNC
Visibility range

+ $\varphi = 2$
Chirality

+ chirality, + no faults
Tori, FSYNC, perpetual, chirality, $\varphi = 2$ ●● p. 201

no chirality, + no faults
Tori, FSYNC, perpetual, no chirality, $\varphi = 2$ ●● p. 202

$\varphi = 1$, + chirality, + no faults
Tori, FSYNC, perpetual, chirality, $\varphi = 1$ ●● p. 202

$\varphi = 1$, no chirality, + no faults
Tori, FSYNC, perpetual, no chirality, $\varphi = 1$ ●● p. 203

+ triangular grid, + perpetual, + luminous, + FSYNC
Visibility range

$\varphi = 1$, no chirality, + no faults
Triangular grids, FSYNC, perpetual, no chirality, $\varphi = 1$ ●● p. 203

+ $\varphi = 2$, + chirality, + no faults
Triangular grids, FSYNC, perpetual, chirality, $\varphi = 2$ ●● p. 204

What is known

The best upper bound against the best lower bound in each catalogued setting, in tree order. The settings follow in detail.

Setting	Robots	Colours	p.
Grids, FSYNC, terminating, chirality, $\varphi = 2$	● tight 2 vs 2	● tight 2 vs 2	185
Grids, FSYNC, terminating, no chirality, $\varphi = 2$	● tight 3 vs 2	● unknown 2 vs —	186
Grids, FSYNC, terminating, chirality, $\varphi = 1$	● tight 2 vs 2	● tight 3 vs 3	186
Grids, FSYNC, terminating, no chirality, $\varphi = 1$	● tight 4 vs 2	● unknown 3 vs —	187
Grids, ASYNC, terminating, chirality, $\varphi = 2$	● tight 2 vs 2	● tight 3 vs 2	188
Grids, ASYNC, terminating, no chirality, $\varphi = 2$	● tight 3 vs 2	● unknown 3 vs —	188
Grids, ASYNC, terminating, chirality, $\varphi = 1$	● tight 3 vs 3	● unknown 3 vs —	189
Grids, ASYNC, terminating, no chirality, $\varphi = 1$	● tight 6 vs 3	● unknown 3 vs —	189
Grids, FSYNC, perpetual, chirality, $\varphi = 1$	● tight 2 vs 2	● tight 3 vs 3	190
Grids, FSYNC, perpetual, chirality, $\varphi = 2$	● tight 2 vs 2	● tight 2 vs 2	190

Claude (Anthropic) was used in cataloguing the problems.

Setting	Robots	Colours	p.
Grids, FSYNC, perpetual, no chirality, $\varphi = 1$	● tight 3 vs 3	● unknown 3 vs —	191
Grids, ASYNC, perpetual, chirality, $\varphi = 2$	● tight 3 vs 2	● tight 2 vs 2	192
Grids, ASYNC, perpetual, no chirality, $\varphi = 2$	● tight 3 vs 3	● tight 2 vs 2	192
Grids, ASYNC, perpetual, chirality, $\varphi = 1$ (exclusive)	● impossible	● impossible	193
Grids, ASYNC, perpetual, no chirality, $\varphi = 3$	● tight 3 vs 3	● tight 1 vs —	194
Rings, FSYNC, perpetual, $\varphi = 1$	● tight 2 vs 2	● tight 2 vs 2	194
Rings, FSYNC, perpetual, up to f crashes	● tight $f + 2$ vs $f + 2$	● unknown $3f + 2$ vs —	195
Rings, ASYNC, perpetual, $\varphi \geq 2$	● tight 2 vs 2	● tight 2 vs 2	195
Rings, ASYNC, perpetual, up to f crashes, $\varphi \geq 2$	● tight $2f + 2$ vs $2f + 2$	● unknown $2f + 2$ vs —	196
Rings, ASYNC, perpetual, $\varphi = 1$	● tight 3 vs 3	● unknown 2 vs —	197
Rings, ASYNC, perpetual, up to f crashes, $\varphi = 1$	● tight $3f + 3$ vs $3f + 3$	● unknown $2f + 2$ vs —	197
Rings, FSYNC, terminating, $\varphi = 1$ STANDARD SETTING	● tight 3 vs 3	● tight 2 vs 2	198
Rings, ASYNC, terminating, $\varphi = 1$	● tight 4 vs 4	● tight 2 vs —	199
Rings, ASYNC, terminating, $\varphi \geq 2$	● tight 3 vs 3	● unknown 2 vs —	199
Rings, oblivious, FSYNC, terminating, $\varphi = 1$	● tight 5 vs 5	● tight 1 vs —	200
Rings, oblivious, ASYNC, terminating, $\varphi = 1$	● impossible	● impossible	200
Tori, FSYNC, perpetual, chirality, $\varphi = 2$	● tight 3 vs 3	● tight 2 vs 2	201
Tori, FSYNC, perpetual, no chirality, $\varphi = 2$	● tight 3 vs 3	● tight 3 vs —	202
Tori, FSYNC, perpetual, chirality, $\varphi = 1$	● tight 3 vs 3	● tight 2 vs 2	202
Tori, FSYNC, perpetual, no chirality, $\varphi = 1$	● tight 3 vs 3	● tight 3 vs 2	203
Triangular grids, FSYNC, perpetual, no chirality, $\varphi = 1$	● tight 2 vs 2	● tight 3 vs 2	203
Triangular grids, FSYNC, perpetual, chirality, $\varphi = 2$	● tight 2 vs 2	● unknown 2 vs —	204

16.8.1 Grids, FSYNC, terminating, chirality, $\varphi = 2$

Adds + chirality, + no faults VERIFIED

With $\varphi = 2$ and chirality, two robots with two colours, or three oblivious robots, explore a grid and stop under FSYNC.

Claude (Anthropic) was used in cataloguing the problems.

Assumed: graph dynamics: static graph; edge direction: undirected; network: finite grid; exploration goal: terminating exploration (explore, then stop); lights (memory and messages): full light (LUMI): visible to the robot and to the others; scheduler: fully synchronous (FSYNC); visibility range: $\varphi = 2$ (results hold for every $\varphi \geq 2$); chirality: common chirality (same clockwise direction); faults: no faults. **Not assumed:** identifiers: anonymous: all agents are identical.

Measure	Lower bound	Best upper bound	Status
Robots	2 (Bramas et al. 2023a) Two.	2 (Nagahama et al. 2022)	• tight
Colours	2 (Bramas et al. 2023a) Two for two robots.	2 (Nagahama et al. 2022)	• tight

Results.

- Bramas et al. (2023a) [lower bound]: robots 2. *Bottleneck:* symmetry breaking. *Holds when:* one robot (perpetual-exploration bound that also holds for terminating exploration Nagahama et al. 2022).
- Bramas et al. (2023a) [lower bound]: colours 2. *Bottleneck:* symmetry breaking. *Holds when:* two robots with one colour, any finite φ (Theorem 2, applied to terminating exploration in Nagahama et al. 2022).
- Nagahama et al. (2022) [current best]: robots 2, colours 2. *Assumes:* robots start in a designated formation at a corner of the grid; towers allowed (not exclusive); $a \geq 2$, $b \geq 3$. Introductory; optimal in robots (Section 4.2.1).
- Nagahama et al. (2022) [also]: robots 3, colours 1. *Assumes:* robots start in a designated formation at a corner of the grid; towers allowed (not exclusive); $a \geq 2$, $b \geq 3$. Oblivious robots (Section 4.2.3); optimal for one colour.

16.8.2 Grids, FSYNC, terminating, no chirality, $\varphi = 2$

Adds no chirality, + no faults VERIFIED

Without chirality and with $\varphi = 2$, three robots with two colours, or four oblivious robots, explore a grid and stop under FSYNC.

Assumed: graph dynamics: static graph; edge direction: undirected; network: finite grid; exploration goal: terminating exploration (explore, then stop); lights (memory and messages): full light (LUMI): visible to the robot and to the others; scheduler: fully synchronous (FSYNC); visibility range: $\varphi = 2$ (results hold for every $\varphi \geq 2$); faults: no faults. **Not assumed:** chirality: no common chirality; identifiers: anonymous: all agents are identical.

Measure	Lower bound	Best upper bound	Status
Robots	2 (Bramas et al. 2023a)	3 (Nagahama et al. 2022)	• tight
	Constant on both sides, so tight up to a constant: two against three. The exact number is open.		
Colours	none known No colour lower bound for three robots.	2 (Nagahama et al. 2022)	• unknown

Results.

- Bramas et al. (2023a) [lower bound]: robots 2. *Bottleneck:* symmetry breaking. *Holds when:* one robot (perpetual-exploration bound that also holds for terminating exploration Nagahama et al. 2022).
- Nagahama et al. (2022) [current best]: robots 3, colours 2. *Assumes:* robots start in a designated formation at a corner of the grid; towers allowed (not exclusive); $a \geq 2$, $b \geq 3$. Introductory (Section 4.2.2).
- Nagahama et al. (2022) [also]: robots 4, colours 1. *Assumes:* robots start in a designated formation at a corner of the grid; towers allowed (not exclusive); $a \geq 2$, $b \geq 3$. Oblivious robots (Section 4.2.4).

16.8.3 Grids, FSYNC, terminating, chirality, $\varphi = 1$

Adds $\varphi = 1$, + chirality, + no faults VERIFIED

Two robots with three colours, or three with two, explore a grid and stop under FSYNC with chirality.

Claude (Anthropic) was used in cataloguing the problems.

Assumed: graph dynamics: static graph; edge direction: undirected; network: finite grid; exploration goal: terminating exploration (explore, then stop); lights (memory and messages): full light (LUMI): visible to the robot and to the others; scheduler: fully synchronous (FSYNC); chirality: common chirality (same clockwise direction); faults: no faults. **Not assumed:** visibility range: myopic, $\varphi = 1$: a robot sees its node and its neighbours; identifiers: anonymous: all agents are identical.

Measure	Lower bound	Best upper bound	Status
Robots	2 (Bramas et al. 2023a) Two.	2 (Nagahama et al. 2022)	• tight
Colours	3 (Bramas et al. 2023a) Three for two robots.	3 (Nagahama et al. 2022)	• tight

Results.

- Bramas et al. (2023a) [lower bound]: robots 2. *Bottleneck:* symmetry breaking. *Holds when:* one robot (perpetual-exploration bound that also holds for terminating exploration Nagahama et al. 2022).
- Bramas et al. (2023a) [lower bound]: colours 3. *Bottleneck:* symmetry breaking. *Holds when:* two robots with two colours and $\varphi = 1$ cannot explore perpetually; the proof shows some node is never visited (Theorem 3, applied to terminating exploration in Nagahama et al. 2022).
- Nagahama et al. (2022) [current best]: robots 2, colours 3. *Assumes:* robots start in a designated formation at a corner of the grid; towers allowed (not exclusive); $a \geq 2$, $b \geq 3$. Introductory; optimal in robots (Section 4.2.5).
- Nagahama et al. (2022) [also]: robots 3, colours 2. *Assumes:* robots start in a designated formation at a corner of the grid; towers allowed (not exclusive); $a \geq 2$, $b \geq 3$. Section 4.2.7; optimal in robots for two colours.

16.8.4 Grids, FSYNC, terminating, no chirality, $\varphi = 1$

Adds $\varphi = 1$, no chirality, + no faults VERIFIED

Without chirality, four robots with three colours, or five with two, explore a grid and stop under FSYNC.

Assumed: graph dynamics: static graph; edge direction: undirected; network: finite grid; exploration goal: terminating exploration (explore, then stop); lights (memory and messages): full light (LUMI): visible to the robot and to the others; scheduler: fully synchronous (FSYNC); faults: no faults. **Not assumed:** visibility range: myopic, $\varphi = 1$: a robot sees its node and its neighbours; chirality: no common chirality; identifiers: anonymous: all agents are identical.

Measure	Lower bound	Best upper bound	Status
Robots	2 (Bramas et al. 2023a) Constant on both sides, so tight up to a constant: two against four; the two-robot impossibility of (Rauch et al. 2021) is proved for perpetual exploration (Theorem 1). The exact number is open.	4 (Nagahama et al. 2022)	• tight
Colours	none known No colour lower bound for four robots.	3 (Nagahama et al. 2022)	• unknown

Results.

- Bramas et al. (2023a) [lower bound]: robots 2. *Bottleneck:* symmetry breaking. *Holds when:* one robot (perpetual-exploration bound that also holds for terminating exploration Nagahama et al. 2022).
- Nagahama et al. (2022) [current best]: robots 4, colours 3. *Assumes:* robots start in a designated formation at a corner of the grid; towers allowed (not exclusive); $a \geq 2$, $b \geq 3$. Introductory (Section 4.2.6).
- Nagahama et al. (2022) [also]: robots 5, colours 2. *Assumes:* robots start in a designated formation at a corner of the grid; towers allowed (not exclusive); $a \geq 2$, $b \geq 3$. Section 4.2.8.

Open questions.

Q1. How many robots are needed for terminating grid exploration under FSYNC with $\varphi = 1$ and no chirality: three or four?

Claude (Anthropic) was used in cataloguing the problems.

Notes. Without chirality two robots with $\varphi = 1$ cannot explore a grid perpetually (Rauch et al. 2021) (Theorem 1). The proof shows that after some time the two robots stay within a bounded distance of the walls, so nodes far from every wall are not visited again (Lemma 7 and Theorem 1); this contradicts perpetual but not terminating exploration, so the argument does not apply as written to terminating exploration.

16.8.5 Grids, ASYNC, terminating, chirality, $\varphi = 2$

Adds + chirality, + no faults VERIFIED

With $\varphi = 2$ and chirality, two robots with three colours, or three with two, explore a grid and stop under ASYNC.

Assumed: graph dynamics: static graph; edge direction: undirected; network: finite grid; exploration goal: terminating exploration (explore, then stop); lights (memory and messages): full light (LUMI): visible to the robot and to the others; visibility range: $\varphi = 2$ (results hold for every $\varphi \geq 2$); chirality: common chirality (same clockwise direction); faults: no faults. **Not assumed:** scheduler: asynchronous (ASYNC); identifiers: anonymous: all agents are identical.

Measure	Lower bound	Best upper bound	Status
Robots	2 (Bramas et al. 2023a) Two.	2 (Nagahama et al. 2022)	• tight
Colours	2 (Bramas et al. 2023a) Constant on both sides, so tight up to a constant: two against three, for two robots. The exact number is open.	3 (Nagahama et al. 2022)	• tight

Results.

- Bramas et al. (2023a) [lower bound]: robots 2. *Bottleneck:* symmetry breaking. *Holds when:* one robot (perpetual-exploration bound that also holds for terminating exploration Nagahama et al. 2022).
- Bramas et al. (2023a) [lower bound]: colours 2. *Bottleneck:* symmetry breaking. *Holds when:* two robots with one colour, any finite φ (Theorem 2, applied to terminating exploration in Nagahama et al. 2022).
- Nagahama et al. (2022) [current best]: robots 2, colours 3. *Assumes:* robots start in a designated formation at a corner of the grid; towers allowed (not exclusive); $a \geq 2$, $b \geq 3$. Introductory; optimal in robots (Section 4.3.1).
- Nagahama et al. (2022) [also]: robots 3, colours 2. *Assumes:* robots start in a designated formation at a corner of the grid; towers allowed (not exclusive); $a \geq 2$, $b \geq 3$. Section 4.3.3.

16.8.6 Grids, ASYNC, terminating, no chirality, $\varphi = 2$

Adds no chirality, + no faults VERIFIED

Without chirality and with $\varphi = 2$, three robots with three colours, or four with two, explore a grid and stop under ASYNC.

Assumed: graph dynamics: static graph; edge direction: undirected; network: finite grid; exploration goal: terminating exploration (explore, then stop); lights (memory and messages): full light (LUMI): visible to the robot and to the others; visibility range: $\varphi = 2$ (results hold for every $\varphi \geq 2$); faults: no faults. **Not assumed:** scheduler: asynchronous (ASYNC); chirality: no common chirality; identifiers: anonymous: all agents are identical.

Measure	Lower bound	Best upper bound	Status
Robots	2 (Bramas et al. 2023a) Constant on both sides, so tight up to a constant: two against three. The exact number is open.	3 (Nagahama et al. 2022)	• tight
Colours	none known No colour lower bound.	3 (Nagahama et al. 2022)	• unknown

Claude (Anthropic) was used in cataloguing the problems.

Results.

- Bramas et al. (2023a) [lower bound]: robots 2. *Bottleneck:* symmetry breaking. *Holds when:* one robot (perpetual-exploration bound that also holds for terminating exploration Nagahama et al. 2022).
- Nagahama et al. (2022) [current best]: robots 3, colours 3. *Assumes:* robots start in a designated formation at a corner of the grid; towers allowed (not exclusive); $a \geq 2$, $b \geq 3$. Introductory (Section 4.3.2).
- Nagahama et al. (2022) [also]: robots 4, colours 2. *Assumes:* robots start in a designated formation at a corner of the grid; towers allowed (not exclusive); $a \geq 2$, $b \geq 3$. Section 4.3.4.

16.8.7 Grids, ASYNC, terminating, chirality, $\varphi = 1$

Adds ASYNC, $\varphi = 1$, + chirality, + no faults VERIFIED

Three robots with three colours explore a grid and stop under ASYNC with chirality and $\varphi = 1$; two robots cannot.

Assumed: graph dynamics: static graph; edge direction: undirected; network: finite grid; exploration goal: terminating exploration (explore, then stop); lights (memory and messages): full light (LUMI): visible to the robot and to the others; chirality: common chirality (same clockwise direction); faults: no faults. **Not assumed:** scheduler: asynchronous (ASYNC); visibility range: myopic, $\varphi = 1$: a robot sees its node and its neighbours; identifiers: anonymous: all agents are identical.

Measure	Lower bound	Best upper bound	Status
Robots	3 (Nagahama et al. 2022) Three.	3 (Nagahama et al. 2022)	• tight
Colours	none known No colour lower bound for three robots.	3 (Nagahama et al. 2022)	• unknown

Results.

- Nagahama et al. (2022) [lower bound]: robots 3. *Bottleneck:* adversarial activation. *Holds when:* two robots, $\varphi = 1$, any number of colours, with or without chirality, under SSYNC (Theorem 1); grids of at least 9×9 nodes; also rules out perpetual exploration.
- Nagahama et al. (2022) [current best]: robots 3, colours 3. *Assumes:* robots start in a designated formation at a corner of the grid; towers allowed (not exclusive); $a \geq 2$, $b \geq 3$. Introductory and current best (Section 4.3.5); proceeds along a row with the ring algorithm of (Ooshita and Tixeuil 2022).

16.8.8 Grids, ASYNC, terminating, no chirality, $\varphi = 1$

Adds ASYNC, $\varphi = 1$, no chirality, + no faults VERIFIED

Without chirality, six robots with three colours explore a grid and stop under ASYNC with $\varphi = 1$; at least three are needed.

Assumed: graph dynamics: static graph; edge direction: undirected; network: finite grid; exploration goal: terminating exploration (explore, then stop); lights (memory and messages): full light (LUMI): visible to the robot and to the others; faults: no faults. **Not assumed:** scheduler: asynchronous (ASYNC); visibility range: myopic, $\varphi = 1$: a robot sees its node and its neighbours; chirality: no common chirality; identifiers: anonymous: all agents are identical.

Measure	Lower bound	Best upper bound	Status
Robots	3 (Nagahama et al. 2022)	6 (Nagahama et al. 2022)	• tight
	Constant on both sides, so tight up to a constant: three against six. The exact number is open.		
Colours	none known No colour lower bound.	3 (Nagahama et al. 2022)	• unknown

Claude (Anthropic) was used in cataloguing the problems.

Results.

- Nagahama et al. (2022) [lower bound]: robots 3. *Bottleneck:* adversarial activation. *Holds when:* two robots, $\varphi = 1$, any number of colours, with or without chirality, under SSYNC (Theorem 1); grids of at least 9×9 nodes; also rules out perpetual exploration.
- Nagahama et al. (2022) [current best]: robots 6, colours 3. *Assumes:* robots start in a designated formation at a corner of the grid; towers allowed (not exclusive); $a \geq 2$, $b \geq 3$. Introductory and current best (Section 4.3.6); needs $a, b \geq 3$.

Open questions.

Q1. Close the gap between three and six robots for terminating grid exploration under ASYNC with $\varphi = 1$ and no chirality (Nagahama et al. 2022).

16.8.9 Grids, FSYNC, perpetual, chirality, $\varphi = 1$

Adds $\varphi = 1$, + chirality, + no faults VERIFIED

With common chirality and $\varphi = 1$, two robots with three colours explore a grid perpetually, optimal in robots and colours.

Assumed: graph dynamics: static graph; edge direction: undirected; network: finite grid; exploration goal: perpetual exploration (visit every node infinitely often); lights (memory and messages): full light (LUMI): visible to the robot and to the others; scheduler: fully synchronous (FSYNC); chirality: common chirality (same clockwise direction); faults: no faults. **Not assumed:** visibility range: myopic, $\varphi = 1$: a robot sees its node and its neighbours; identifiers: anonymous: all agents are identical.

Measure	Lower bound	Best upper bound	Status
Robots	2 (Bramas et al. 2023a) Two.	2 (Bramas et al. 2023a)	• tight
Colours	3 (Bramas et al. 2023a) Three, for two robots.	3 (Bramas et al. 2023a)	• tight

Results.

- Bramas et al. (2023a) [lower bound]: robots 2. *Bottleneck:* symmetry breaking. *Holds when:* one robot, any finite φ and colours (Theorem 1); holds without exclusiveness.
- Bramas et al. (2023a) [lower bound]: colours 3. *Bottleneck:* symmetry breaking. *Holds when:* two robots with two colours and $\varphi = 1$ (Theorem 3); three robots with one colour and $\varphi = 1$ also fail (Theorem 4).
- Bramas et al. (2023a) [current best]: robots 2, colours 3. *Assumes:* exclusive: no two robots ever share a node or traverse an edge together; opaque robots; locally defined initial configurations; grid with at least k rows and k columns. Introductory and current best (algorithm V_{32}^{one}).

Notes. Lower bounds hold without exclusiveness; Nagahama, Ooshita and Inoue use them for terminating exploration as well (Nagahama et al. 2022).

16.8.10 Grids, FSYNC, perpetual, chirality, $\varphi = 2$

Adds + $\varphi = 2$, + chirality, + no faults VERIFIED

With $\varphi = 2$, two robots with two colours, or three oblivious robots, explore a grid perpetually.

Assumed: graph dynamics: static graph; edge direction: undirected; network: finite grid; exploration goal: perpetual exploration (visit every node infinitely often); lights (memory and messages): full light (LUMI): visible to the robot and to

Claude (Anthropic) was used in cataloguing the problems.

the others; scheduler: fully synchronous (FSYNC); visibility range: $\varphi = 2$ (results hold for every $\varphi \geq 2$); chirality: common chirality (same clockwise direction); faults: no faults. **Not assumed:** identifiers: anonymous: all agents are identical.

Measure	Lower bound	Best upper bound	Status
Robots	2 (Bramas et al. 2023a) Two.	2 (Bramas et al. 2023a)	• tight
Colours	2 (Bramas et al. 2023a) Two for two robots; one for three.	2 (Bramas et al. 2023a)	• tight

Results.

- Bramas et al. (2023a) [lower bound]: robots 2. *Bottleneck:* symmetry breaking. *Holds when:* one robot, any finite φ (Theorem 1).
- Bramas et al. (2023a) [lower bound]: colours 2. *Bottleneck:* symmetry breaking. *Holds when:* two robots with one colour, any finite φ (Theorem 2).
- Bramas et al. (2023a) [current best]: robots 2, colours 2. *Assumes:* exclusive: no two robots ever share a node or traverse an edge together; opaque robots. Algorithm V_{22}^{two}.
- Bramas et al. (2023a) [also]: robots 3, colours 1. *Assumes:* exclusive: no two robots ever share a node or traverse an edge together; opaque robots. Algorithm V_{13}^{two}: oblivious robots.

16.8.11 Grids, FSYNC, perpetual, no chirality, $\varphi = 1$

Adds $\varphi = 1$, no chirality, + no faults VERIFIED

Without chirality, two robots never suffice and three robots with three colours do.

Assumed: graph dynamics: static graph; edge direction: undirected; network: finite grid; exploration goal: perpetual exploration (visit every node infinitely often); lights (memory and messages): full light (LUMI): visible to the robot and to the others; scheduler: fully synchronous (FSYNC); faults: no faults. **Not assumed:** visibility range: myopic, $\varphi = 1$: a robot sees its node and its neighbours; chirality: no common chirality; identifiers: anonymous: all agents are identical.

Measure	Lower bound	Best upper bound	Status
Robots	3 (Rauch et al. 2021) Three.	3 (Rauch et al. 2021)	• tight
Colours	none known No colour lower bound is proved for three robots (Rauch et al. 2021).	3 (Rauch et al. 2021)	• unknown

Results.

- Rauch et al. (2021) [lower bound]: robots 3. *Bottleneck:* symmetry breaking. *Holds when:* two robots with $\varphi = 1$ and any finite number of colours (Theorem 1), on a large enough grid; holds without exclusiveness for edges, but the model forbids two robots on one node; self-inconsistent robots: the adversary may rotate and mirror each snapshot differently in every round.
- Rauch et al. (2021) [current best]: robots 3, colours 3. *Assumes:* exclusive: no two robots ever share a node or traverse an edge together; locally defined initial configurations (three aligned robots coloured L, L, F); grid with at least k rows and k columns. Introductory and current best (algorithm V_{33}^{one}, Theorem 2). With $\varphi = 2$, five oblivious robots suffice (algorithm V_{51}^{two}, Theorem 3).

Notes. Rauch, Bramas, Devismes, Lafourcade and Lamani state in their contributions that two robots fail for every finite φ, but the proof of Theorem 1 assumes $\varphi = 1$ throughout (Section 3) (Rauch et al. 2021). The correctness proof of V_{33}^{one} is a sketch: a simulation tool checks grids of sizes 3 and 4 in each dimension, and an argument extends it to larger grids (Theorem 2).

Claude (Anthropic) was used in cataloguing the problems.

16.8.12 Grids, ASYNC, perpetual, chirality, $\varphi = 2$

Adds + chirality, + no faults VERIFIED

Three robots with two colours, or four oblivious robots, explore a grid perpetually under ASYNC with chirality and $\varphi = 2$.

Assumed: graph dynamics: static graph; edge direction: undirected; network: finite grid; exploration goal: perpetual exploration (visit every node infinitely often); lights (memory and messages): full light (LUMI): visible to the robot and to the others; visibility range: $\varphi = 2$ (results hold for every $\varphi \geq 2$); chirality: common chirality (same clockwise direction); faults: no faults. **Not assumed:** scheduler: asynchronous (ASYNC); identifiers: anonymous: all agents are identical.

Measure	Lower bound	Best upper bound	Status
Robots	2 (Bramas et al. 2023a)	3 (Bramas et al. 2023a)	• tight
	Constant on both sides, so tight up to a constant: two against three: no impossibility for two luminous robots under ASYNC with chirality and $\varphi = 2$ was found (two robots with two colours work under FSYNC Bramas et al. 2023a). The exact number is open.		
Colours	2 (Bramas et al. 2026a)	2 (Bramas et al. 2023a)	• tight
	Two for three robots; one for four.		

Results.

- Bramas et al. (2023a) [lower bound]: robots 2. *Bottleneck:* symmetry breaking. *Holds when:* one robot, any finite φ (Theorem 1); FSYNC bound.
- Bramas et al. (2026a) [lower bound]: colours 2. *Bottleneck:* symmetry breaking. *Holds when:* three oblivious robots, $\varphi = 2$, even with chirality (Theorem 3).
- Bramas et al. (2023a) [current best]: robots 3, colours 2. *Assumes:* exclusive: no two robots ever share a node or traverse an edge together. Transformation of V_{32}^{one} to ASYNC under $\varphi = 2$, as reported in Table 1 of (Bramas et al. 2026a).
- Bramas et al. (2026a) [also]: robots 4, colours 1. *Assumes:* exclusive: no two robots ever share a node or traverse an edge together; grids with at least five rows and columns (smaller cases by special rules); initial configuration not locally defined. Algorithm A_1: oblivious robots, optimal for $c = 1$.

Open questions.

Q1. Can two luminous robots with common chirality and $\varphi = 2$ explore a finite grid perpetually under ASYNC (or SSYNC)?

16.8.13 Grids, ASYNC, perpetual, no chirality, $\varphi = 2$

Adds no chirality, + no faults VERIFIED

Three robots with two colours explore a grid perpetually under ASYNC without chirality, which is optimal.

Assumed: graph dynamics: static graph; edge direction: undirected; network: finite grid; exploration goal: perpetual exploration (visit every node infinitely often); lights (memory and messages): full light (LUMI): visible to the robot and to the others; visibility range: $\varphi = 2$ (results hold for every $\varphi \geq 2$); faults: no faults. **Not assumed:** scheduler: asynchronous (ASYNC); chirality: no common chirality; identifiers: anonymous: all agents are identical.

Measure	Lower bound	Best upper bound	Status
Robots	3 (Rauch et al. 2021)	3 (Bramas et al. 2026a)	• tight
	Three, if the two-robot impossibility holds for $\varphi \geq 2$ as (Rauch et al. 2021) and (Bramas et al. 2026a) state; the published proof covers $\varphi = 1$, so the proved range is two to three.		
Colours	2 (Bramas et al. 2026a)	2 (Bramas et al. 2026a)	• tight
	Two, for three robots.		

Results.

- Rauch et al. (2021) [lower bound]: robots 3. *Bottleneck:* symmetry breaking. *Holds when:* two robots without chirality, any finite number of colours, under FSYNC (hence ASYNC); stated for every finite φ in the contributions of (Rauch et al. 2021) and in (Bramas et al. 2026a), but the proof of Theorem 1 of (Rauch et al. 2021) assumes $\varphi = 1$; for $\varphi \geq 2$ the proved bound is two (one robot fails for every φ even with chirality Bramas et al. 2023a).
- Bramas et al. (2026a) [lower bound]: colours 2. *Bottleneck:* symmetry breaking. *Holds when:* three oblivious robots, $\varphi = 2$ (Theorem 3).
- Bramas et al. (2026a) [current best]: robots 3, colours 2. *Assumes:* exclusive: no two robots ever share a node or traverse an edge together; transparent robots (no opacity); initial L-shape coloured A, A, B. Introductory and current best (algorithm A_2).

Open questions.

Q1. Is there an optimal ASYNC algorithm (three robots, $\varphi = 2$, no chirality) for opaque robots? Posed in (Bramas et al. 2026a).

Q2. Can two robots without chirality explore a grid perpetually with $\varphi \geq 2$ under FSYNC? (Rauch et al. 2021) state that they cannot for every finite φ, but prove it for $\varphi = 1$.

16.8.14 Grids, ASYNC, perpetual, chirality, $\varphi = 1$ (exclusive)

Adds ASYNC, $\varphi = 1$, + chirality, + no faults VERIFIED

With $\varphi = 1$ no number of exclusive asynchronous robots explores a grid perpetually, whatever the number of colours.

Assumed: graph dynamics: static graph; edge direction: undirected; network: finite grid; exploration goal: perpetual exploration (visit every node infinitely often); lights (memory and messages): full light (LUMI): visible to the robot and to the others; chirality: common chirality (same clockwise direction); faults: no faults. **Not assumed:** scheduler: asynchronous (ASYNC); visibility range: myopic, $\varphi = 1$: a robot sees its node and its neighbours; identifiers: anonymous: all agents are identical.

Measure	Lower bound	Best upper bound	Status
Robots	*impossible* (Bramas et al. 2026a) For exclusive robots.	—	• impossible
Colours	none known For exclusive robots.	—	• impossible

Results.

- Bramas et al. (2026a) [lower bound]: robots *impossible*. *Bottleneck:* locality. *Holds when:* any number of robots and colours, even with common chirality (Theorem 2); robots may never share a node (the model of the paper); the adversary prevents any group of robots from crossing a fence of width two (test of the fence).

Open questions.

Q1. Is perpetual grid exploration with $\varphi = 1$ solvable under ASYNC when robots may share nodes (form towers)? Terminating exploration is, with three robots and three colours (Nagahama et al. 2022).

Notes. The impossibility assumes, as the model of (Bramas et al. 2026a) does, that two robots never occupy one node. Terminating exploration in the same scheduler and range is solvable when towers are allowed (Nagahama et al. 2022), so co-location is what the impossibility needs.

Claude (Anthropic) was used in cataloguing the problems.

16.8.15 Grids, ASYNC, perpetual, no chirality, $\varphi = 3$

Adds ASYNC, + $\varphi = 3$, + no faults VERIFIED

With $\varphi = 3$, three oblivious robots explore a grid perpetually under ASYNC.

Assumed: graph dynamics: static graph; edge direction: undirected; network: finite grid; exploration goal: perpetual exploration (visit every node infinitely often); lights (memory and messages): full light (LUMI): visible to the robot and to the others; visibility range: $\varphi = 3$; faults: no faults. **Not assumed:** scheduler: asynchronous (ASYNC); chirality: no common chirality; identifiers: anonymous: all agents are identical.

Measure	Lower bound	Best upper bound	Status
Robots	3 (Rauch et al. 2021)	3 (Bramas et al. 2026a)	• tight
	Three, if the two-robot impossibility holds for $\varphi \geq 2$ as (Rauch et al. 2021) and (Bramas et al. 2026a) state; the published proof covers $\varphi = 1$, so the proved range is two to three.		
Colours	none known	1 (Bramas et al. 2026a)	• tight
	No light needed.		

Results.

- Rauch et al. (2021) [lower bound]: robots 3. *Bottleneck:* symmetry breaking. *Holds when:* two robots without chirality, any finite number of colours, under FSYNC (hence ASYNC); stated for every finite φ in the contributions of (Rauch et al. 2021) and in (Bramas et al. 2026a), but the proof of Theorem 1 of (Rauch et al. 2021) assumes $\varphi = 1$; for $\varphi \geq 2$ the proved bound is two (one robot fails for every φ even with chirality Bramas et al. 2023a).
- Bramas et al. (2026a) [current best]: robots 3, colours 1. *Assumes:* exclusive: no two robots ever share a node or traverse an edge together. Algorithm A_3: oblivious robots.

Open questions.

Q1. Can two robots without chirality explore a grid perpetually with $\varphi \geq 2$ under FSYNC? (Rauch et al. 2021) state that they cannot for every finite φ, but prove it for $\varphi = 1$.

Notes. A larger range replaces the light: compare three robots with two colours at $\varphi = 2$.

16.8.16 Rings, FSYNC, perpetual, $\varphi = 1$

Adds + no faults VERIFIED

Two robots with two colours explore a ring perpetually under FSYNC, which is optimal.

Assumed: graph dynamics: static graph; edge direction: undirected; network: ring; exploration goal: perpetual exploration (visit every node infinitely often); lights (memory and messages): full light (LUMI): visible to the robot and to the others; scheduler: fully synchronous (FSYNC); faults: no faults. **Not assumed:** visibility range: myopic, $\varphi = 1$: a robot sees its node and its neighbours; chirality: no common chirality; identifiers: anonymous: all agents are identical.

Measure	Lower bound	Best upper bound	Status
Robots	2 (Ooshita and Tixeuil 2022)	2 (Ooshita and Tixeuil 2022)	• tight
	Two.		
Colours	2 (Ooshita and Tixeuil 2022)	2 (Ooshita and Tixeuil 2022)	• tight
	Two, for two robots. With one colour, three robots suffice (the case $f = 0$ of the one-colour algorithm of Ooshita et al. 2025).		

Results.

- Ooshita and Tixeuil (2022) [lower bound]: robots 2. *Bottleneck:* symmetry breaking. *Holds when:* one robot sees nobody, so its snapshot is symmetric and the adversary chooses its direction (any φ, any number of colours); stated in Section 3.1.

- Ooshita and Tixeuil (2022) [lower bound]: colours 2. *Bottleneck:* symmetry breaking. *Holds when:* two robots, $n \geq 6$ (Lemma 6); two robots with the same colour take the same snapshot and move symmetrically.
- Ooshita and Tixeuil (2022) [current best]: robots 2, colours 2. *Assumes:* initial configuration: the two robots on adjacent nodes, coloured G and W; $n \geq 2$. Introductory and current best. Universal for $n \geq 6$: it works from every initial configuration from which two robots with colours $\{G, W\}$ can explore (Theorem 7).

Notes. The robot coloured G steps away from the other robot and the robot coloured W steps towards it; in FSYNC both move at once, so the pair keeps its shape and walks around the ring forever (Algorithm 1 of Ooshita and Tixeuil 2022). The colours give the pair a direction that the unoriented ring does not.

16.8.17 Rings, FSYNC, perpetual, up to f crashes

Adds + at most crash faults VERIFIED

$f + 2$ robots with $3f + 2$ colours explore a ring perpetually despite f crashes, which is optimal in robots.

Assumed: graph dynamics: static graph; edge direction: undirected; network: ring; exploration goal: perpetual exploration (visit every node infinitely often); lights (memory and messages): full light (LUMI): visible to the robot and to the others; scheduler: fully synchronous (FSYNC); faults: up to f agents may crash (stop forever). **Not assumed:** visibility range: myopic, $\varphi = 1$: a robot sees its node and its neighbours; chirality: no common chirality; identifiers: anonymous: all agents are identical.

Measure	Lower bound	Best upper bound	Status
Robots	$f + 2$ (Ooshita et al. 2024) $f + 2$.	$f + 2$ (Ooshita et al. 2024)	• tight
Colours	none known No lower bound on colours; with $f + 2$ robots the known algorithm uses $3f + 2$ colours, and fewer colours cost more robots (Ooshita et al. 2025).	$3f + 2$ (Ooshita et al. 2024)	• unknown

Results.

- Ooshita et al. (2024) [lower bound]: robots $f + 2$. *Bottleneck:* trivial. *Holds when:* at least two robots must survive (fault-free bound of two robots); any φ and any number of colours.
- Ooshita et al. (2024) [current best]: robots $f + 2$, colours $3f + 2$. *Assumes:* specific initial positions and colours; holds for every $\varphi \geq 1$. Introductory and current best in robots. A crashed robot keeps its colour and cannot be told apart from a correct one.
- Ooshita et al. (2025) [also]: robots $6f+3$, colours 1. *Assumes:* $\varphi = 1$. Oblivious robots; one of three robots–colours trade-offs.
- Ooshita et al. (2025) [also]: robots $4f + 2$, colours 2. *Assumes:* $\varphi = 1$. Two colours.
- Ooshita et al. (2025) [also]: robots $2f + 2$, colours $O(\log f)$. *Assumes:* $\varphi = 1$. Logarithmically many colours.

Open questions.

Q1. With $f + 2$ robots under FSYNC, how many colours are necessary for crash-tolerant perpetual ring exploration? The best algorithm uses $3f + 2$ (Ooshita et al. 2024).

Notes. The trivial algorithm runs $f + 1$ independent pairs with distinct colours ($2f + 2$ robots, $2f + 2$ colours); (Ooshita et al. 2024) lowers the robots to $f + 2$.

16.8.18 Rings, ASYNC, perpetual, $\varphi \geq 2$

Adds + no faults VERIFIED

With visibility two, two robots with two colours explore a ring perpetually even under ASYNC.

Claude (Anthropic) was used in cataloguing the problems.

Assumed: graph dynamics: static graph; edge direction: undirected; network: ring; exploration goal: perpetual exploration (visit every node infinitely often); lights (memory and messages): full light (LUMI): visible to the robot and to the others; visibility range: $\varphi = 2$ (results hold for every $\varphi \geq 2$); faults: no faults. **Not assumed:** scheduler: asynchronous (ASYNC); chirality: no common chirality; identifiers: anonymous: all agents are identical.

Measure	Lower bound	Best upper bound	Status
Robots	2 (Nagahama et al. 2023) Two.	2 (Nagahama et al. 2023)	• tight
Colours	2 (Nagahama et al. 2023) Two, for two robots on rings of $n \geq 2\varphi + 4$ nodes.	2 (Nagahama et al. 2023)	• tight

Results.

- Nagahama et al. (2023) [lower bound]: robots 2. *Bottleneck:* symmetry breaking. *Holds when:* one robot: when it sees nobody its snapshot is symmetric and the scheduler chooses the direction of each move (Section 3); any φ, any number of colours; also stated for every φ and every number of colours in (Ooshita et al. 2024).
- Nagahama et al. (2023) [lower bound]: colours 2. *Bottleneck:* symmetry breaking. *Holds when:* two robots of the same colour on a ring of $n \geq 2\varphi + 4$ nodes, even under FSYNC (Lemma 3): the scheduler moves both at once, so they keep the same colour and move symmetrically; with one colour every configuration of two robots is of this kind.
- Nagahama et al. (2023) [current best]: robots 2, colours 2. *Assumes:* initial configurations in which the two robots have different colours and see each other (on one node, or at distance at most φ); $n \geq 5$. Introductory and current best (Algorithm 1, Theorem 1, ASYNC). It is universal for $n \geq 2\varphi + 4$ (Theorem 2): every other configuration of two robots (out of sight of each other, or of the same colour) is unsolvable even under FSYNC (Lemmas 2 and 3).

Notes. Larger visibility lets the distance between robots carry information that otherwise needs a robot, as Ooshita and Tixeuil anticipated (Ooshita and Tixeuil 2022). With $\varphi \geq 2$ the same two robots also suffice under FSYNC and SSYNC, so visibility saves one robot under SSYNC and ASYNC and none under FSYNC (Nagahama et al. 2023). In the algorithm the robot coloured W moves away from the other robot when they are at distance at most one, and the robot coloured G moves towards the other robot when they are farther apart (Section 3).

16.8.19 Rings, ASYNC, perpetual, up to f crashes, $\varphi \geq 2$

Adds + at most crash faults

VERIFIED

$2f + 2$ robots are necessary and sufficient under SSYNC and ASYNC with $\varphi \geq 2$.

Assumed: graph dynamics: static graph; edge direction: undirected; network: ring; exploration goal: perpetual exploration (visit every node infinitely often); lights (memory and messages): full light (LUMI): visible to the robot and to the others; visibility range: $\varphi = 2$ (results hold for every $\varphi \geq 2$); faults: up to f agents may crash (stop forever). **Not assumed:** scheduler: asynchronous (ASYNC); chirality: no common chirality; identifiers: anonymous: all agents are identical.

Measure	Lower bound	Best upper bound	Status
Robots	$2f + 2$ (Ooshita et al. 2024) $2f + 2$.	$2f + 2$ (Ooshita et al. 2024)	• tight
Colours	none known No lower bound on colours.	$2f + 2$ (Ooshita et al. 2024)	• unknown

Results.

- Ooshita et al. (2024) [lower bound]: robots $2f + 2$. *Bottleneck:* indistinguishable crashes. *Holds when:* any number of colours, SSYNC (hence ASYNC); $f + 1$ groups of the fault-free minimum of two.

Claude (Anthropic) was used in cataloguing the problems.

- Ooshita et al. (2024) [current best]: robots $2f + 2$, colours $2f + 2$. Introductory and current best: $f + 1$ pairs running the two-robot algorithm of (Nagahama et al. 2023).

16.8.20 Rings, ASYNC, perpetual, $\varphi = 1$

Adds ASYNC, $\varphi = 1$, + no faults VERIFIED

Three robots with two colours explore a ring perpetually under ASYNC; two robots cannot even under SSYNC.

Assumed: graph dynamics: static graph; edge direction: undirected; network: ring; exploration goal: perpetual exploration (visit every node infinitely often); lights (memory and messages): full light (LUMI): visible to the robot and to the others; faults: no faults. **Not assumed:** scheduler: asynchronous (ASYNC); visibility range: myopic, $\varphi = 1$: a robot sees its node and its neighbours; chirality: no common chirality; identifiers: anonymous: all agents are identical.

Measure	Lower bound	Best upper bound	Status
Robots	3 (Ooshita and Tixeuil 2022) Three.	3 (Ooshita and Tixeuil 2022)	• tight
Colours	none known No lower bound on colours for three robots is stated.	2 (Ooshita and Tixeuil 2022)	• unknown

Results.

- Ooshita and Tixeuil (2022) [lower bound]: robots 3. *Bottleneck:* adversarial activation. *Holds when:* two robots, any number of colours, $n \geq 6$, proved under SSYNC (Theorem 12); when a robot steps away from the other, the adversary activates it alone; the two robots are then too far apart to see each other and stay confined to two nodes each (Lemma 4).
- Ooshita and Tixeuil (2022) [current best]: robots 3, colours 2. *Assumes:* designated initial configurations of three connected robots (e.g. WWG, WGG); $n \geq 3$. Introductory and current best (Theorem 13); universal for $n \geq 9$ under SSYNC and ASYNC (Theorem 14).

Open questions.

Q1. Can three oblivious robots ($c = 1$) with $\varphi = 1$ explore a ring perpetually under SSYNC or ASYNC, or are two colours necessary?

Notes. With $\varphi \geq 2$ two robots suffice (Nagahama et al. 2023) (variant *Rings, ASYNC, perpetual, $\varphi \geq 2$*).

16.8.21 Rings, ASYNC, perpetual, up to f crashes, $\varphi = 1$

Adds ASYNC, $\varphi = 1$, + at most crash faults VERIFIED

$3f + 3$ robots are necessary and sufficient under SSYNC and ASYNC with $\varphi = 1$.

Assumed: graph dynamics: static graph; edge direction: undirected; network: ring; exploration goal: perpetual exploration (visit every node infinitely often); lights (memory and messages): full light (LUMI): visible to the robot and to the others; faults: up to f agents may crash (stop forever). **Not assumed:** scheduler: asynchronous (ASYNC); visibility range: myopic, $\varphi = 1$: a robot sees its node and its neighbours; chirality: no common chirality; identifiers: anonymous: all agents are identical.

Measure	Lower bound	Best upper bound	Status
Robots	$3f + 3$ (Ooshita et al. 2024) $3f + 3$.	$3f + 3$ (Ooshita et al. 2024)	• tight
Colours	none known No lower bound on colours.	$2f + 2$ (Ooshita et al. 2024)	• unknown

Claude (Anthropic) was used in cataloguing the problems.

Results.

- Ooshita et al. (2024) [lower bound]: robots $3f + 3$. *Bottleneck:* indistinguishable crashes. *Holds when:* any number of colours, SSYNC (hence ASYNC); a slow robot looks crashed; robots must leave a group of three behind each apparent crash, so f dispersed crashes force $f + 1$ groups of the fault-free minimum of three.

- Ooshita et al. (2024) [current best]: robots $3f + 3$, colours $2f + 2$. Introductory and current best: $f + 1$ groups of three, each running the fault-free algorithm of (Ooshita and Tixeuil 2022) with its own pair of colours.

16.8.22 Rings, FSYNC, terminating, $\varphi = 1$

Adds + FSYNC, + no faults

STANDARD SETTING VERIFIED

Three robots with two colours explore a ring and stop under FSYNC; two robots cannot, whatever the number of colours.

Assumed: graph dynamics: static graph; edge direction: undirected; network: ring; exploration goal: terminating exploration (explore, then stop); lights (memory and messages): full light (LUMI): visible to the robot and to the others; scheduler: fully synchronous (FSYNC); faults: no faults. **Not assumed:** visibility range: myopic, $\varphi = 1$: a robot sees its node and its neighbours; chirality: no common chirality; identifiers: anonymous: all agents are identical.

Measure	Lower bound	Best upper bound	Status
Robots	3 (Ooshita and Tixeuil 2022) Three.	3 (Ooshita and Tixeuil 2022)	• tight
Colours	2 (Datta et al. 2013) Two, for three robots: oblivious robots need five (Datta et al. 2013).	2 (Ooshita and Tixeuil 2022)	• tight

Results.

- Ooshita and Tixeuil (2022) [lower bound]: robots 3. *Bottleneck:* locality. *Holds when:* two robots, any number of colours (Theorem 8); while two robots stay within distance one they cannot tell a ring of n_1 nodes from one of $n_2 > 2(t + 1)$ nodes, so they stop too early in the larger ring; once they separate, each is confined to two nodes (Lemma 4).

- Datta et al. (2013) [lower bound]: colours 2. *Bottleneck:* symmetry breaking. *Holds when:* three or four oblivious robots on rings with $n > 7$ (Theorem 3.2).

- Ooshita and Tixeuil (2022) [current best]: robots 3, colours 2. *Assumes:* designated initial configurations: three robots on consecutive nodes coloured WWW, GWW, WWG or GWG; $n \geq 3$. Introductory and current best (Theorem 9). No algorithm with three robots and two colours is universal (Theorem 11).

Open questions.

Q1. Characterise the initial configurations of three robots with $c \geq 3$ colours from which terminating exploration of a ring is solvable under FSYNC with $\varphi = 1$ (with two colours no universal algorithm exists Ooshita and Tixeuil 2022).

Notes. From GWG the middle robot joins one end robot, forming a pair GW that walks away from the third robot, which waits; when the pair arrives at the waiting robot from the other side every node has been visited, and one more move forms a tower that no rule enables (Algorithm 2 of Ooshita and Tixeuil 2022). Without lights five robots are necessary and sufficient (Datta et al. 2013) (variant *Rings, oblivious, FSYNC, terminating*).

Claude (Anthropic) was used in cataloguing the problems.

16.8.23 Rings, ASYNC, terminating, $\varphi = 1$

Adds ASYNC, $\varphi = 1$, + no faults VERIFIED

Four robots with two colours explore a ring and stop under ASYNC; three cannot even under SSYNC.

Assumed: graph dynamics: static graph; edge direction: undirected; network: ring; exploration goal: terminating exploration (explore, then stop); lights (memory and messages): full light (LUMI): visible to the robot and to the others; faults: no faults. **Not assumed:** scheduler: asynchronous (ASYNC); visibility range: myopic, $\varphi = 1$: a robot sees its node and its neighbours; chirality: no common chirality; identifiers: anonymous: all agents are identical.

Measure	Lower bound	Best upper bound	Status
Robots	4 (Ooshita and Tixeuil 2022) Four.	4 (Ooshita and Tixeuil 2022)	• tight
Colours	none known Two: oblivious robots cannot explore at all under SSYNC or ASYNC with $\varphi = 1$ (Datta et al. 2013).	2 (Ooshita and Tixeuil 2022)	• tight

Results.

- Ooshita and Tixeuil (2022) [lower bound]: robots 4. *Bottleneck:* locality. *Holds when:* three robots, any number of colours, proved under SSYNC (Theorem 15); robots that stay close cannot tell a ring of $n_1 \geq 9$ nodes from one of $4(t + 1)$ nodes.
- Ooshita and Tixeuil (2022) [current best]: robots 4, colours 2. *Assumes:* designated initial configurations of four robots on consecutive nodes (WWGG, WWWG, WWGW and colour swaps); $n \geq 5$. Introductory and current best (Theorem 16). No universal algorithm with four robots and two colours exists (Theorem 18).

Open questions.

Q1. Does any task separate SSYNC from ASYNC for myopic luminous robots on rings? The known exploration bounds coincide under both (Ooshita and Tixeuil 2022).

Notes. Without lights, terminating exploration with $\varphi = 1$ is impossible under SSYNC and ASYNC for $1 \leq k < n$ (Datta et al. 2013). With $\varphi \geq 2$ three robots suffice and are necessary (Nagahama et al. 2023) (setting *Rings, ASYNC, terminating,* $\varphi \geq 2$).

16.8.24 Rings, ASYNC, terminating, $\varphi \geq 2$

Adds ASYNC, + $\varphi = 2$, + no faults VERIFIED

With visibility two, three robots with two colours explore a ring and stop even under ASYNC; two robots cannot, even under FSYNC.

Assumed: graph dynamics: static graph; edge direction: undirected; network: ring; exploration goal: terminating exploration (explore, then stop); lights (memory and messages): full light (LUMI): visible to the robot and to the others; visibility range: $\varphi = 2$ (results hold for every $\varphi \geq 2$); faults: no faults. **Not assumed:** scheduler: asynchronous (ASYNC); chirality: no common chirality; identifiers: anonymous: all agents are identical.

Measure	Lower bound	Best upper bound	Status
Robots	3 (Nagahama et al. 2023) Three.	3 (Nagahama et al. 2023)	• tight
Colours	none known No lower bound on colours for three robots is proved for every $\varphi \geq 2$; without lights and with $\varphi = 2$ or 3, five robots are necessary under SSYNC and ASYNC (Datta et al., as listed in Table 1 of Nagahama et al. 2023).	2 (Nagahama et al. 2023)	• unknown

Claude (Anthropic) was used in cataloguing the problems.

Results.

- Nagahama et al. (2023) [lower bound]: robots 3. *Bottleneck:* locality. *Holds when:* two robots, any φ, even with infinitely many colours, under FSYNC (hence ASYNC) (Theorem 3); two robots that keep seeing each other cannot tell a ring of $n_1 \geq 2\varphi + 4$ nodes from a larger one before they stop; two robots that lose sight of each other stay in disjoint territories (Lemma 1).
- Nagahama et al. (2023) [current best]: robots 3, colours 2. *Assumes:* designated initial configurations: three robots that see one another in a row coloured W, W, G with gaps of at most $\varphi - 1$ free nodes, and the mirror image; $n \geq 4$. Introductory and current best (Algorithm 2, Theorem 4). Two further algorithms cover the other configurations in which the three robots see one another and no two share a node, for $n \geq 3\varphi + 1$ (Theorems 5 and 6, Lemmas 4 to 6), but no single algorithm with three robots and two colours works from all of them (Theorem 7).

Notes. With $\varphi = 1$ four robots are needed under SSYNC and ASYNC (Ooshita and Tixeuil 2022), so a larger range saves one robot; under FSYNC three robots are needed for both ranges (Nagahama et al. 2023).

16.8.25 Rings, oblivious, FSYNC, terminating, $\varphi = 1$

Adds oblivious, + FSYNC, + no faults VERIFIED

Without lights, five robots are necessary and sufficient to explore a ring of more than seven nodes and stop under FSYNC.

Assumed: graph dynamics: static graph; edge direction: undirected; network: ring; exploration goal: terminating exploration (explore, then stop); scheduler: fully synchronous (FSYNC); faults: no faults. **Not assumed:** lights (memory and messages): oblivious, no light (OBLOT); visibility range: myopic, $\varphi = 1$: a robot sees its node and its neighbours; chirality: no common chirality; identifiers: anonymous: all agents are identical.

Measure	Lower bound	Best upper bound	Status
Robots	5 (Datta et al. 2013) Five.	5 (Datta et al. 2013)	• tight
Colours	none known No light.	1 (Datta et al. 2013)	• tight

Results.

- Datta et al. (2013) [lower bound]: robots 5. *Bottleneck:* symmetry breaking. *Holds when:* $n > 7$, $2 \leq k < 5$, deterministic (Theorem 3.2); from one block of adjacent robots, every rule either repeats an indistinguishable configuration or disperses the robots.
- Datta et al. (2013) [current best]: robots 5, colours 1. *Assumes:* initial configuration: one block of five robots on consecutive nodes; $n \geq 7$; for $3 \leq n \leq 6$ the paper uses $n - 1$ robots (4 for $n = 6$). Introductory and current best.

Notes. Baseline for the power of lights: two colours bring the number of robots from five to three (Ooshita and Tixeuil 2022).

16.8.26 Rings, oblivious, ASYNC, terminating, $\varphi = 1$

Adds oblivious, ASYNC, + no faults VERIFIED

Without lights and with $\varphi = 1$, no number of robots below n explores a ring under SSYNC or ASYNC.

Assumed: graph dynamics: static graph; edge direction: undirected; network: ring; exploration goal: terminating exploration (explore, then stop); faults: no faults. **Not assumed:** lights (memory and messages): oblivious, no light (OBLOT); scheduler:

Claude (Anthropic) was used in cataloguing the problems.

asynchronous (ASYNC); visibility range: myopic, $\varphi = 1$: a robot sees its node and its neighbours; chirality: no common chirality; identifiers: anonymous: all agents are identical.

Measure	Lower bound	Best upper bound	Status
Robots	*impossible* (Datta et al. 2013) For every $k < n$.	—	• impossible
Colours	none known No light.	—	• impossible

Results.

- Datta et al. (2013) [lower bound]: robots *impossible*. *Bottleneck:* symmetry breaking. *Holds when:* $1 \le k < n, n > 1$, deterministic, proved under SSYNC (Theorem 3.1, Corollary 3.2).

Notes. With $\varphi = 2$ the same authors explore with seven or nine robots under ASYNC, and with $\varphi = 3$ they prove that fewer than five robots cannot explore (for $n > 13$) and give five- and seven-robot algorithms (Datta et al. 2013). Two colours make $\varphi = 1$ enough with four robots (Ooshita and Tixeuil 2022).

16.8.27 Tori, FSYNC, perpetual, chirality, $\varphi = 2$

Adds + chirality, + no faults

VERIFIED

With $\varphi = 2$ and chirality, three robots with two colours or four oblivious robots explore a torus perpetually under FSYNC.

Assumed: graph dynamics: static graph; edge direction: undirected; network: torus; exploration goal: perpetual exploration (visit every node infinitely often); lights (memory and messages): full light (LUMI): visible to the robot and to the others; scheduler: fully synchronous (FSYNC); visibility range: $\varphi = 2$ (results hold for every $\varphi \ge 2$); chirality: common chirality (same clockwise direction); faults: no faults. **Not assumed:** identifiers: anonymous: all agents are identical.

Measure	Lower bound	Best upper bound	Status
Robots	3 (Darwich et al. 2023) Three.	3 (Darwich et al. 2023)	• tight
Colours	2 (Darwich et al. 2023) Two for three robots; one for four.	2 (Darwich et al. 2023)	• tight

Results.

- Darwich et al. (2023) [lower bound]: robots 3. *Bottleneck:* symmetry breaking. *Holds when:* two robots (Theorem 1).
- Darwich et al. (2023) [lower bound]: colours 2. *Bottleneck:* symmetry breaking. *Holds when:* three oblivious robots, any finite φ (Theorem 2).
- Darwich et al. (2023) [current best]: robots 3, colours 2. *Assumes:* exclusive: no two robots ever share a node or traverse an edge together; opaque robots; torus with both sides at least $k\varphi + 1$. Algorithm A_1, designed for $\varphi = 1$.
- Darwich et al. (2023) [also]: robots 4, colours 1. *Assumes:* exclusive: no two robots ever share a node or traverse an edge together; opaque robots; torus with both sides at least $k\varphi + 1$. Algorithm A_3: oblivious robots, optimal for $c = 1$.

Claude (Anthropic) was used in cataloguing the problems.

16.8.28 Tori, FSYNC, perpetual, no chirality, $\varphi = 2$

Adds no chirality, + no faults VERIFIED

Without chirality and with $\varphi = 2$, three robots with three colours or five oblivious robots explore a torus perpetually under FSYNC.

Assumed: graph dynamics: static graph; edge direction: undirected; network: torus; exploration goal: perpetual exploration (visit every node infinitely often); lights (memory and messages): full light (LUMI): visible to the robot and to the others; scheduler: fully synchronous (FSYNC); visibility range: $\varphi = 2$ (results hold for every $\varphi \geq 2$); faults: no faults. **Not assumed:** chirality: no common chirality; identifiers: anonymous: all agents are identical.

Measure	Lower bound	Best upper bound	Status
Robots	3 (Darwich et al. 2023) Three.	3 (Darwich et al. 2023)	• tight
Colours	none known Constant on both sides, so tight up to a constant: two against three, for three robots. The exact number is open.	3 (Darwich et al. 2023)	• tight

Results.

- Darwich et al. (2023) [lower bound]: robots 3. *Bottleneck:* symmetry breaking. *Holds when:* two robots (Theorem 1).
- Darwich et al. (2023) [current best]: robots 3, colours 3. *Assumes:* exclusive: no two robots ever share a node or traverse an edge together; opaque robots; torus with both sides at least $k\varphi + 1$; self-inconsistent robots. Algorithm A_2, designed for $\varphi = 1$.
- Darwich et al. (2023) [also]: robots 5, colours 1. *Assumes:* exclusive: no two robots ever share a node or traverse an edge together; opaque robots; torus with both sides at least $k\varphi + 1$; self-inconsistent robots. Algorithm A_4: oblivious robots; three oblivious robots cannot (Theorem 2), so four or five is open for $c = 1$.

Open questions.

Q1. Can four oblivious robots with $\varphi = 2$ explore a torus perpetually without chirality?

16.8.29 Tori, FSYNC, perpetual, chirality, $\varphi = 1$

Adds $\varphi = 1$, + chirality, + no faults VERIFIED

Three robots with two colours explore a torus perpetually under FSYNC with chirality, optimal in robots and colours.

Assumed: graph dynamics: static graph; edge direction: undirected; network: torus; exploration goal: perpetual exploration (visit every node infinitely often); lights (memory and messages): full light (LUMI): visible to the robot and to the others; scheduler: fully synchronous (FSYNC); chirality: common chirality (same clockwise direction); faults: no faults. **Not assumed:** visibility range: myopic, $\varphi = 1$: a robot sees its node and its neighbours; identifiers: anonymous: all agents are identical.

Measure	Lower bound	Best upper bound	Status
Robots	3 (Darwich et al. 2023) Three.	3 (Darwich et al. 2023)	• tight
Colours	2 (Darwich et al. 2023) Two.	2 (Darwich et al. 2023)	• tight

Results.

- Darwich et al. (2023) [lower bound]: robots 3. *Bottleneck:* symmetry breaking. *Holds when:* two robots, any finite colours and φ, even with chirality (Theorem 1); two robots that travel together follow an integer vector, which

fails on tori whose sides are multiples of its components.

- Darwich et al. (2023) [lower bound]: colours 2. *Bottleneck:* symmetry breaking. *Holds when:* three oblivious robots, any finite φ (Theorem 2).
- Darwich et al. (2023) [current best]: robots 3, colours 2. *Assumes:* exclusive: no two robots ever share a node or traverse an edge together; opaque robots; torus with both sides at least $k\varphi + 1$. Introductory and current best (algorithm A_1). One robot more than on grids, which have walls to turn at.

16.8.30 Tori, FSYNC, perpetual, no chirality, $\varphi = 1$

Adds $\varphi = 1$, no chirality, + no faults VERIFIED

Without chirality, three robots with three colours explore a torus perpetually under FSYNC.

Assumed: graph dynamics: static graph; edge direction: undirected; network: torus; exploration goal: perpetual exploration (visit every node infinitely often); lights (memory and messages): full light (LUMI): visible to the robot and to the others; scheduler: fully synchronous (FSYNC); faults: no faults. **Not assumed:** visibility range: myopic, $\varphi = 1$: a robot sees its node and its neighbours; chirality: no common chirality; identifiers: anonymous: all agents are identical.

Measure	Lower bound	Best upper bound	Status
Robots	3 (Darwich et al. 2023) Three.	3 (Darwich et al. 2023)	• tight
Colours	2 (Darwich et al. 2023) Constant on both sides, so tight up to a constant: two against three. The exact number is open.	3 (Darwich et al. 2023)	• tight

Results.

- Darwich et al. (2023) [lower bound]: robots 3. *Bottleneck:* symmetry breaking. *Holds when:* two robots, even with chirality (Theorem 1).
- Darwich et al. (2023) [lower bound]: colours 2. *Bottleneck:* symmetry breaking. *Holds when:* three oblivious robots (Theorem 2).
- Darwich et al. (2023) [current best]: robots 3, colours 3. *Assumes:* exclusive: no two robots ever share a node or traverse an edge together; opaque robots; torus with both sides at least $k\varphi + 1$; self-inconsistent robots. Algorithm A_2 (journal version only).

Open questions.

Q1. Can three robots with two colours and $\varphi = 1$ explore a torus perpetually without common chirality?

16.8.31 Triangular grids, FSYNC, perpetual, no chirality, $\varphi = 1$

Adds $\varphi = 1$, no chirality, + no faults DRAFT

On a triangular grid two robots with three colours and $\varphi = 1$ explore perpetually under FSYNC, if each robot keeps its own chirality; on a square grid two robots cannot.

Assumed: graph dynamics: static graph; edge direction: undirected; network: rectangle-enclosed triangular grid; exploration goal: perpetual exploration (visit every node infinitely often); lights (memory and messages): full light (LUMI): visible to the robot and to the others; scheduler: fully synchronous (FSYNC); faults: no faults. **Not assumed:** visibility range: myopic, $\varphi = 1$: a robot sees its node and its neighbours; chirality: no common chirality; identifiers: anonymous: all agents are identical.

Claude (Anthropic) was used in cataloguing the problems.

Measure	Lower bound	Best upper bound	Status
Robots	2 (Das et al. 2025c) Two.	2 (Das et al. 2025c)	• tight
Colours	2 (Das et al. 2025c) Between two and three; the one-colour impossibility is a sketch for self-inconsistent robots, while the algorithm assumes robots that keep their chirality.	3 (Das et al. 2025c)	• tight

Results.

- Das et al. (2025c) [lower bound]: robots 2. *Bottleneck:* symmetry breaking. *Holds when:* one robot, any number of colours and any finite φ (Theorem 3.1); a robot far from the border sees nothing, and the adversary sets its local orientation so that it moves between two nodes; proof sketch only.
- Das et al. (2025c) [lower bound]: colours 2. *Bottleneck:* symmetry breaking. *Holds when:* two oblivious robots with $\varphi = 2$, hence also with $\varphi = 1$ (Theorem 3.2); self-inconsistent robots: the adversary sets each robot's local orientation in every round; proof sketch only.
- Das et al. (2025c) [current best]: robots 2, colours 3. *Assumes:* each robot keeps the same chirality in every round (robots need not agree on it); initial configuration: two robots on neighbouring nodes, one coloured L or R and the other F; at least four columns of at least four nodes; no two robots share a node. Introductory and current best (algorithm A_{123}^{VRL}, Section 4): the robot coloured F follows the other, and the pair sweeps the columns one by one, turning at the border. Correctness is argued informally.

Open questions.

Q1. Is there a proof of perpetual triangular-grid exploration by two robots with $\varphi = 1$ for self-inconsistent robots, or with two colours?

Notes. Das, Goswami, Mondal and Sau contrast this with the square grid, where two robots with $\varphi = 1$ and no chirality cannot explore perpetually (Rauch et al. 2021; Das et al. 2025c). The paper is a five-page short paper: the impossibilities and the algorithm are given with proof sketches, so the setting stays a draft.

16.8.32 Triangular grids, FSYNC, perpetual, chirality, $\varphi = 2$

Adds + $\varphi = 2$, + chirality, + no faults DRAFT

With common chirality and $\varphi = 2$, two robots with two colours explore a triangular grid perpetually under FSYNC.

Assumed: graph dynamics: static graph; edge direction: undirected; network: rectangle-enclosed triangular grid; exploration goal: perpetual exploration (visit every node infinitely often); lights (memory and messages): full light (LUMI): visible to the robot and to the others; scheduler: fully synchronous (FSYNC); visibility range: $\varphi = 2$ (results hold for every $\varphi \geq 2$); chirality: common chirality (same clockwise direction); faults: no faults. **Not assumed:** identifiers: anonymous: all agents are identical.

Measure	Lower bound	Best upper bound	Status
Robots	2 (Das et al. 2025c) Two.	2 (Das et al. 2025c)	• tight
Colours	none known The one-colour impossibility of (Das et al. 2025c) (Theorem 3.2) is argued for self-inconsistent robots without chirality.	2 (Das et al. 2025c)	• unknown

Results.

- Das et al. (2025c) [lower bound]: robots 2. *Bottleneck:* symmetry breaking. *Holds when:* one robot, any number of colours and any finite φ (Theorem 3.1); proof sketch only.

Claude (Anthropic) was used in cataloguing the problems.

- Das et al. (2025c) [current best]: robots 2, colours 2. *Assumes:* initial configuration: two robots on one column or diagonal line within distance two, coloured L and F; at least four columns of at least four nodes; no two robots share a node. Introductory and current best (algorithm A_{222}^{VRL}, Section 5), built like A_{123}^{VRL} with a single leader colour.

Notes. Proof sketches only (five-page short paper), so the setting stays a draft.



16.9 Exploration of recolourable graphs

A single agent with little or no memory must visit every node of an anonymous graph whose nodes carry colours it can read and rewrite; how many colours are needed, and how much does it help to recolour, to see neighbours or to have port numbers?

First posed in 2023 (Böckenhauer et al. 2023). Böckenhauer, Frei, Unger and Wehner posed exploration by an oblivious agent without port numbers that colours nodes and sees its neighbours' colours (SIROCCO 2023; arXiv January 2023), first with each node coloured once and, in a last section, with recolouring. Fixed colours chosen by a designer (2005) and the rotor-router pointer (1996, 2003) are earlier node-side precursors.

Problem statement and measures

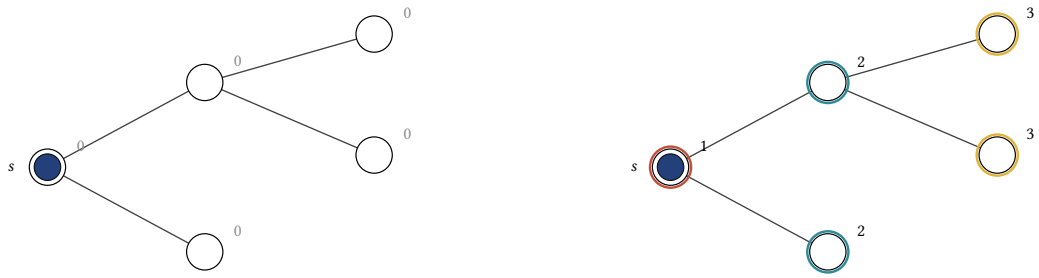
A single deterministic agent starts at a node s of a connected, simple, undirected, anonymous graph $G = (V, E)$ with n nodes, m edges, maximum degree Δ and diameter D. Every node v holds a colour $\psi(v)$ from a palette of c colours that persists between visits. In each step the agent at v observes $\psi(v)$ (and, depending on the setting, $\deg(v)$, port numbers, the entry port, and the multiset of its neighbours' colours), sets $\psi(v)$ to a new colour (**recolours** v) if the setting allows it, and then moves to a neighbour or stops. In the main model (Böckenhauer et al. 2023) there are no port numbers: the agent names a colour and the adversary moves it to any neighbour of that colour; the agent is **oblivious** (it keeps no memory between steps); all nodes start with the **initial colour** c_0; and the agent must **visit every node, return to** s **and stop there**, from every start and against every adversary. Other settings fix the colours in advance (label-guided exploration), give port numbers, or ask only for perpetual exploration. **Measures:** the number of colours c, counting c_0 (the main measure); time, the number of moves (traversals of one edge) until termination (or until every node is visited, for perpetual exploration); and the agent's memory. All known results are for one agent.

Measures: **Colours**, number c of node colours, counting the initial colour c_0 (for the rotor-router: per node); **Time**, moves until the agent stops (perpetual: until every node is visited); **Memory**, bits of agent memory; 0 for an oblivious agent.

A first look

Imagine exploring a town of identical squares joined by streets, with no street signs and no memory at all: after each step you forget everything. What you do have is a box of coloured flags, one flagpole on every square, and good eyes: standing on a square you see its flag and the flags on the squares next to it. You may change the flag where you stand, then name a colour and walk to a neighbouring square showing that colour; if several do, an **adversary** (an imagined opponent) picks which one. We model the town as a graph: squares are **nodes**, streets are **edges**, the flag on a square is the **colour** of that node, and you are the **agent**, which is **oblivious** (it remembers nothing between steps); n is the number of nodes. In the basic version every node starts with the same colour 0, the **initial colour** (a node that

Claude (Anthropic) was used in cataloguing the problems.



(a) Start: every node has colour 0

(b) Back at s: colour 1, no neighbour of colour 3, so it stops

Figure 16.9: Exploration with colourable nodes (Böckenhauer et al. 2023): an oblivious agent on a tree without port numbers sees the colour of its node and of its neighbours. On an uncoloured node it recolours the node with one more than the largest neighbouring colour (cyclically 1, 2, 3), which is its depth mod 3 plus one; it moves to an uncoloured neighbour if there is one (the adversary picks which) and otherwise to its parent, the unique neighbour of the previous colour; it stops at the start, the only node of colour 1 without a neighbour of colour 3. The coloured rings show the colours of the nodes; the agent itself remembers nothing.

still has it is **uncoloured**), you may **recolour** the node where you stand (change its colour) as often as you like, and you must visit every node, come back to where you started and stop there. The question is how many colours c are needed for every possible graph.

1. Algorithm: Colour by depth mod 3 (trees). (Böckenhauer et al. 2023) (Once-colourable nodes on trees, p. 218) For a graph without cycles (a **tree**):

1. On an uncoloured node, recolour it with the colour one more than the largest colour among its neighbours, counting cyclically 1, 2, 3, 1, ... (at the start all neighbours are 0, so the start gets 1).
2. If a neighbour is uncoloured, walk to one.
3. Otherwise, if your node has colour 1 and no neighbour of colour 3, stop.
4. Otherwise walk to the neighbour whose colour is one less than yours (cyclically): it is your **parent**, the neighbour one edge closer to the start.

Why it works and what it costs. A node at distance j from the start gets colour $(j \bmod 3) + 1$, so its parent has the previous colour and its children the next one; in a tree the parent is the only neighbour with the previous colour, so the way back is never ambiguous. The start is the only node of colour 1 without a neighbour of colour 3. Each edge is walked once down and once up, so the walk has $2(n-1)$ **moves** (one move is one traversal of one edge). Three colours plus the initial one are needed: on a path of seven nodes two colours (plus 0) always leave the agent unable to tell the way back (Böckenhauer et al. 2023). With recolouring allowed, three colours in all suffice on trees (Hiraoka et al. 2026).

Why it matters. Depth mod 3 is the basic trick of the whole area: a designer uses it in label-guided exploration (Cohen et al. 2008), and the general eight-colour algorithm is this idea with a phase colour on top (Böckenhauer et al. 2023).

2. Algorithm: The rotor-router: a pointer as the colour. (Menc et al. 2017) (Rotor-router: a pointer in every node, p. 218) Now the edges at each node are numbered $1, \dots, d$, where d is the **degree** of the node (its number of edges); these numbers are **ports**. You are not told through which port you arrived. The colour of a node of degree d is a number π between 1 and d.

Claude (Anthropic) was used in cataloguing the problems.

1. Leave through port π.
2. Before leaving, recolour the node to $\pi + 1$ (and after d back to 1).
3. Repeat forever.

Why it works and what it costs. Each node sends you out through its ports in turn, so no edge is neglected for long. Whatever the colours are at the start, after $O(mD)$ moves (m edges, D the largest distance between two nodes) the walk follows a fixed tour that uses every edge once in each direction, so every node is visited (Yanovski et al. 2003; Menc et al. 2017). A node of degree d needs d colours, since otherwise some port can never be chosen, and no oblivious rule without the arrival port is faster in the worst case, even with unlimited colours (Menc et al. 2017).

Why it matters. It shows that memory can live entirely in the nodes, and that the colours can even start in a mess (self-stabilization).

3. Lower bound: Colours that can never be changed cost n colours. (Böckenhauer et al. 2023) (Once-colourable nodes, any graph, p. 216) Suppose a colour, once set, stays forever. To return to the start the agent must always recognise which neighbour it came from. The adversary builds graphs with a long path whose nodes have extra edges between them, so that a node on the way back sees several coloured neighbours, and any repeated colour lets the adversary send the agent the wrong way or around in a loop.

Why it holds. Böckenhauer et al. turn this into a proof that for every n some n-node graph forces n colours, counting the initial one, and a depth-first search that gives each new node a colour larger than its coloured neighbours achieves it (Böckenhauer et al. 2023). With recolouring the agent can erase and reuse colours, and six colours suffice for all graphs (Takahashi et al. 2025); at least four are needed (Hiraoka et al. 2026).

Why it matters. The gap between n and a constant is the measure of what the right to recolour is worth.

What the community started looking at

A flag planted in the node. Picture every node with a flagpole. The flag shows one of c colours, and it stays up when the agent leaves: whoever arrives later, the same agent included, sees it, and in the main model an agent also sees the flags on the neighbouring nodes. Compare the three other ways the catalogue leaves information behind. A luminous robot's light is a flag on the robot's own mast: it moves with its owner and is a message to the robots in sight and a memory for the owner (Das et al. 2016) (problem *Exploration by luminous robots*); here the flag stays in place and is a memory of the *place*, read by whoever comes by. A whiteboard is the same idea with an unbounded or $O(\log n)$-bit board (Dobrev et al. 2006); a recolourable node is a whiteboard of $\log_2 c$ bits with c a constant. A pebble or token is a marker the agent carries and drops (Blum and Kozen 1978; Disser et al. 2019; Dobrev et al. 2013): there are few of them and they move, whereas every node has its own flag and none ever moves.

Where it started. Fixed flags came first: Cohen, Fraigniaud, Ilcinkas, Korman and Peleg let a designer colour the nodes in advance, and three colours (depth mod 3 from a root) let a finite automaton explore every graph (Cohen et al. 2008), although no finite automaton explores all graphs without help (Fraigniaud et al. 2005a). The rotor-router, from the Eulerian walkers of physics (Priezzhev et al. 1996) and the patrolling ants of Yanovski, Wagner and Bruckstein (Yanovski et al. 2003), is a node colour that the agent rewrites by a fixed rule: a pointer to the next port to use, which explores any graph from any initial pointers in $O(mD)$ moves, and no oblivious agent that is not told its entry port does better in the worst case (Menc et al. 2017). Böckenhauer, Frei, Unger and Wehner then posed the problem in its present form: no port numbers, no memory, the agent colours nodes and sees its neighbours' colours. If each node can be coloured only once, n colours are necessary and sufficient, four for trees;

Claude (Anthropic) was used in cataloguing the problems.

with recolouring, eight suffice for every graph (Böckenhauer et al. 2023). Takahashi, Kanaya, Hiraoka, Eguchi and Sudo brought this down to six with a depth-first search whose return path never has a chord (Takahashi et al. 2025), and Hiraoka, Imori, Takahashi and Sudo proved the first lower bound, four colours, by a computer-checked argument, which is tight for cacti, and three colours for trees (Hiraoka et al. 2026).

What the colours do. In every algorithm the colours give the agent a sense of direction that the anonymous graph lacks: depth mod 3 tells parent from child (Cohen et al. 2008; Böckenhauer et al. 2023), a path colour marks the way back (Takahashi et al. 2025), a pointer tells which port is next (Menc et al. 2017). Recolouring is what makes the palette constant: a once-coloured node must keep a colour that distinguishes it forever, and that costs n colours (Böckenhauer et al. 2023). All known results are for a single agent.

Further reading

Papers on this problem that are not yet placed in a setting.

- Bojko et al. (2022). Tree exploration with unknown entry port, constant agent memory and $O(\log \deg)$ bits per node: linear time with clean node memory or one token; quadratic time on paths with one bit at the agent and one per node if node memory is dirty (arbitrary). Abstract and introduction read.
- Inoue et al. (2022). With $O(\log \Delta)$ bits per node, an oblivious agent simulates a one-bit agent in two-edge-connected graphs with $O(n^2)$ overhead; bridges are what separates them. Abstract read.
- Sudo et al. (2026b). Self-stabilizing exploration by one agent from arbitrary agent and node memory, using whiteboards of $O(\log)$ size; cover time $O(m)$ expected (randomized) and $O(m + nD)$ (deterministic). Not read; the arXiv versions state different memory bounds.
- Shimoyama et al. (2022). One bit of agent memory suffices for snap-stabilizing perpetual exploration of cacti with distinguishable cycles (title and description in Hiraoka et al. 2026). Not read.
- Sudo et al. (2015). Trade-offs between agent memory and whiteboard size for single-agent exploration. Not read.
- Hu et al. (2012). Label-guided exploration with 1-bit labels whose black/white ratio the designer can adjust, with $O(\rho \log \Delta)$ bits of memory. Abstract read.
- Klasing et al. (2016). Several indistinguishable agents share one rotor-router system on a ring: cover time between $\Theta(n^2/k^2)$ and $\Theta(n^2/\log k)$ depending on the starting nodes, both tight, close to the range for k independent random walks; once stabilised, every node is visited at regular intervals. Read (HAL version).
- Dereniowski et al. (2016). k parallel rotor walks on any graph cover it in at most $O(mD/\log k)$ and at least $\Omega(mD/k)$ steps, a speed-up between $\log k$ and k over one walk; both extremes occur on some graph classes, for up to polynomially many walks. Read (HAL version).
- Kosowski and Pająk (2019). The exact cover time of k parallel rotor walks for every k on degree-bounded expanders, random graphs and constant-dimensional tori, and on hypercubes except for very large k, through a link with the mixing time of the random walk. Read (HAL version).
- Chalopin et al. (2015b). Parallel rotor walks need not settle into a period of length at most $2m$, as Yanovski et al. observed experimentally: the period can be superpolynomial; the limit behaviour is characterised by subcycle decompositions, and the system stabilises within $O(m^4D^2 + mD \log k)$ and $O(m^5k^2)$ steps. Read (HAL version).

Relations to other problems

- **Variant of Graph exploration by a single agent.** The same task for one agent (visit every node; with return, stop or perpetually), but every node holds a colour from a palette of c colours that the agent at the node reads and, unless the colours are fixed, overwrites; the agent itself may have no memory at all, and in the main model it has no port numbers and sees its neighbours'

Claude (Anthropic) was used in cataloguing the problems.

colours instead. With $c = 1$ the colours carry nothing: an algorithm of the parent problem that uses no markers works here whenever this setting grants its ports, entry port and memory, and a lower bound of this problem proved for any number of colours holds for the parent setting with the same view. The parent's pebbles are movable markers carried by the agent (Pattanayak and Pelc 2024a; Disser et al. 2019); here the marks stay in the nodes and there is one per node.

- **Related: Exploration by luminous robots.** Flag on the robot versus flag on the node: a luminous robot carries its coloured light with it and shows it to the robots in sight, whereas here the colour stays planted in the node and is read by whoever arrives.

Assumptions specific to this problem

Port numbers

Port numbers are the usual way an agent tells the edges at a node apart (Fraigniaud et al. 2005a; Pelc 2019), and most exploration papers also give the agent the port by which it entered. The rotor-router drops the entry port: the agent only follows and advances a pointer stored at the node, which works from any initial pointers (Yanovski et al. 2003; Menc et al. 2017). Menc, Pająk and Uznański call this model feature an *unknown inport* and prove that it costs time even with unbounded node memory (Menc et al. 2017). Böckenhauer, Frei, Unger and Wehner removed the port numbers altogether: the agent names a colour and the adversary chooses among the neighbours of that colour, so the colours are the only way to address an edge (Böckenhauer et al. 2023). Takahashi, Kanaya, Hiraoka, Eguchi and Sudo note that without port numbers \"most commonly used techniques for graph exploration and related problems are no longer applicable\" (Takahashi et al. 2025). The scale runs from no ports, through ports without the entry port, to ports with it.

Level	Value	Formal definition
0	**no ports**	Edges carry no port numbers. To move, the agent at v names a colour x; the adversary moves it to any neighbour u of v with $\psi(u) = x$ (if the agent sees no neighbour colours, to any neighbour). The agent never learns by which edge it arrived.
1	**ports, no entry port**	The edges at v carry port numbers $1, \dots, \deg(v)$, assigned by an adversary independently at each endpoint (port_v). The agent chooses the port by which it leaves, but on arrival it is not told the port by which it entered (*unknown inport*).
2	**ports + entry port**	As for port numbers, and on arrival at v the agent also reads the port $\mathrm{port}_v(e)$ of the edge e by which it entered. This is the standard convention of single-agent exploration.

View

In label-guided exploration (Cohen et al. 2008) and in the rotor-router (Menc et al. 2017) the agent reads only the node it stands on, as in all of single-agent exploration. Böckenhauer, Frei, Unger and Wehner observe that colouring alone \"is still utterly useless for an oblivious agent that has to return to the start vertex\" when it cannot tell edges apart, and therefore let the agent see the colours of the neighbouring nodes (Böckenhauer et al. 2023); every later paper on recolourable exploration keeps this one-hop view (Takahashi et al. 2025; Hiraoka et al. 2026). It is the graph analogue of the one-hop visibility of myopic luminous robots, except that what is seen are flags in nodes, not robots.

Claude (Anthropic) was used in cataloguing the problems.

Level	Value	Formal definition
0	**own node**	At node v the agent observes $\psi(v)$, $\deg(v)$ and nothing about its neighbours (and, where ports exist, what the port axis grants).
1	**neighbour colours**	At node v the agent observes $\psi(v)$ and the multiset $\{\!\{\psi(u) : u \in N(v)\}\!\}$ of its neighbours' colours (hence also $\deg(v)$), without knowing which edge leads to which colour unless ports say so.

Agent memory

How little memory an exploring agent needs is the oldest question of the area: no finite automaton explores all graphs (Budach 1978; Fraigniaud et al. 2005a), and the remedies are markers (Blum and Kozen 1978; Disser et al. 2019) or help from a designer (Cohen et al. 2008). Recolourable exploration pushes this to the extreme: the agent is oblivious and all memory lives in the nodes (Böckenhauer et al. 2023), as in the rotor-router (Menc et al. 2017) and the memoryless automaton with pebbles (Pattanayak and Pelc 2024a). Both later papers ask what a constant number of bits of agent memory would buy (Böckenhauer et al. 2023; Takahashi et al. 2025); Inoue, Kitamura, Izumi and Masuzawa compare an oblivious agent with a one-bit agent when nodes hold $O(\log \Delta)$ bits (Inoue et al. 2022).

Level	Value	Formal definition
0	**oblivious**	The agent has no internal state: its action (new colour of v, then a move, *stay* or *stop*) is a function of its current observation only. Called *zero-memory*, *memoryless* or a *1-state* robot.
1	**finite automaton**	The agent is a deterministic automaton with K states, K independent of the graph; its action and next state depend on its state and its observation.
2	$O(\log n)$ **bits**	The agent has $b = O(\log n)$ bits of memory, enough to hold a counter up to a polynomial in n. No variant is catalogued here yet.

Recolouring

Fixed colours are the label-guided exploration of Cohen, Fraigniaud, Ilcinkas, Korman and Peleg: a designer writes 2-bit labels at the nodes in advance and a finite automaton reads them (Cohen et al. 2008). Böckenhauer, Frei, Unger and Wehner let the agent write the colour itself while exploring, once per node, and showed in a last section that allowing recolouring \"radically changes our model\": seven colours (eight with the initial one) then explore every graph, against n without it (Böckenhauer et al. 2023). A recolourable node is a whiteboard of $\log_2 c$ bits; the rotor-router is the special case in which the colour is a pointer that the agent advances on every visit (Yanovski et al. 2003; Menc et al. 2017). Whether a colour may return to c_0 matters for exact counts: Böckenhauer et al. intended that it may not, Takahashi et al. allow it, and their six-colour algorithm needs seven colours without it (Takahashi et al. 2025).

Claude (Anthropic) was used in cataloguing the problems.

Level	Value	Formal definition
0	**fixed colours**	The colouring $\psi : V \to \{1, \dots, c\}$ never changes; the agent reads $\psi(v)$ (and, with neighbour view, its neighbours' colours) but cannot write.
1	**colour once**	Every node starts with the initial colour c_0. The agent at v may change $\psi(v)$ only while $\psi(v) = c_0$; once v has a colour other than c_0 it keeps it forever. This is the main model of (Böckenhauer et al. 2023).
2	**recolourable**	At every step the agent at v may set $\psi(v)$ to any colour of the palette, including back to c_0, any number of times (the convention of Takahashi et al. 2025 and Hiraoka et al. 2026).

Initial colouring

The rotor-router is studied from arbitrary pointers, which makes it self-stabilizing (Yanovski et al. 2003; Bampas et al. 2017; Menc et al. 2017); Bojko, Gotfryd, Kowalski and Pająk call arbitrary initial node memory *dirty*: with one bit at the agent and one bit per node it can force quadratic time on paths, whereas with *clean* (uniformly initialised) logarithmic node memory and constant agent memory trees are explored in linear time (Bojko et al. 2022). Recolourable exploration starts from a uniform colouring: \"at the start of exploration, all nodes are assigned a common initial color\" (Takahashi et al. 2025). A designer's colouring is the preprocessing of label-guided exploration (Cohen et al. 2008). The scale is arbitrary < uniform < designer: an algorithm that works from every colouring works from the uniform one, and a designer may choose the uniform one.

Level	Value	Formal definition
0	**arbitrary colours**	The initial colouring $\psi_0 : V \to \{1, \dots, c\}$ is chosen by the adversary; an algorithm must work from every ψ_0 (as with a rotor-router whose pointers are set arbitrarily).
1	**uniform c_0**	Initially $\psi(v) = c_0$ for every node v (*clean* node memory). The colour c_0 counts as one of the c colours.
2	**designer's colours**	Before the agent starts, a designer who knows G (and the start node) chooses ψ_0. Used with fixed colours this is label-guided exploration (Cohen et al. 2008).

Knowledge

Böckenhauer, Frei, Unger and Wehner note that their general lower bound \"relies heavily on the uniformity of the algorithm\" and study algorithms that need only explore graphs of a given size n, to see \"how useful it is to know the size of the graph in our model\" (Böckenhauer et al. 2023). As in single-agent exploration, knowing the size is the classical way to let a walk end (Aleliunas et al. 1979).

Level	Value	Formal definition
0	**nothing known**	The algorithm is the same for all graphs: it knows neither n nor Δ nor any other parameter (a *uniform* algorithm in Böckenhauer et al. 2023).
1	**n known**	The algorithm may depend on n: it must explore every graph with exactly n nodes (a *non-uniform* algorithm).

Termination requirement

The three requirements are those of single-agent exploration (Fraigniaud et al. 2005a). Recolourable exploration asks for the hardest, return: \"visit every vertex of the graph, return to the start vertex,

Claude (Anthropic) was used in cataloguing the problems.

and terminate\" (Böckenhauer et al. 2023; Takahashi et al. 2025; Hiraoka et al. 2026). Label-guided exploration asks the agent to stop (Cohen et al. 2008), and the rotor-router never stops (Yanovski et al. 2003).

Level	Value	Formal definition
0	**return**	The agent visits every node and then halts at its start node s (*exploration with return*).
1	**stop**	The agent visits every node and then halts, at any node.
2	**perpetual**	Every node is visited at least once (for the rotor-router: infinitely often); the agent never needs to halt.

Network

Colour counts depend strongly on the class of graphs: for once-colourable nodes trees need four colours and general graphs n (Böckenhauer et al. 2023), and for recolourable nodes the bounds are organised by block structure, three colours for trees and cycles and four for cacti (Hiraoka et al. 2026). This axis replaces the model's topology axis because it must order trees below cacti; the other classes studied (triangle-free, φ-free, graphs whose blocks are cycles, complete bipartite graphs or cliques, graphs of bounded circumference) are recorded in the notes.

Level	Value	Formal definition
0	**any graph**	A finite, simple, connected, undirected, anonymous graph $G = (V, E)$ with n nodes.
1	**cactus**	G is a cactus: every edge lies on at most one cycle (equivalently, every block is a cycle or a single edge).
2	**tree**	G is a tree (a cactus without cycles).

Case tree

The root assumes nothing; going down, each edge adds the assumptions written in grey (+) or leaves an axis unassumed. Bold entries are catalogued settings with their status per measure (Colours, Time, Memory) and page; italic entries are branch points, named after the axis on which their branches differ. Axis order: Graph dynamics → Edge direction → Port numbers → View → Agent memory → Recolouring → Initial colouring → Knowledge → Termination requirement → Network.

No assumptions

+ static graph, + undirected

Port numbers

no ports, + neighbour colours

Recolouring

+ recolourable, + uniform c_0

Recolourable nodes, oblivious agent, any graph STANDARD SETTING ●●● p. 214

+ cactus

Recolourable nodes on cacti ●●● p. 215

+ tree

Recolourable nodes on trees ●●● p. 216

+ colour once, + uniform c_0

Once-colourable nodes, any graph ●●● p. 216

Claude (Anthropic) was used in cataloguing the problems.

+ n known
Once-colourable nodes, n known ●●● p. 217

nothing known, + tree
Once-colourable nodes on trees ●●● p. 218

+ ports, no entry port, + recolourable
Initial colouring

arbitrary colours, + perpetual
Rotor-router: a pointer in every node ●●● p. 218

+ designer's colours, + perpetual
Rotor-router with pointers set by a designer ●●● p. 220

+ ports + entry port
Agent memory

+ finite automaton, + designer's colours, + stop
Label-guided: colours fixed by a designer ●●● p. 220

oblivious, + designer's colours, + perpetual
Label-guided, oblivious agent ●●● p. 221

What is known

The best upper bound against the best lower bound in each catalogued setting, in tree order. The settings follow in detail.

Setting	Colours	Time	Memory	p.
Recolourable nodes, oblivious agent, any graph STANDARD SETTING	● tight 6 vs 4	● unknown — vs $\Omega(n)$	● tight 0 vs —	214
Recolourable nodes on cacti	● tight 4 vs 4	● tight $O(n)$ vs $\Omega(n)$	● tight 0 vs —	215
Recolourable nodes on trees	● tight 3 vs 3	● tight $O(n)$ vs $\Omega(n)$	● tight 0 vs —	216
Once-colourable nodes, any graph	● tight n vs n	● unknown — vs —	● tight 0 vs —	216
Once-colourable nodes, n known	● tight $n-1$ vs $n-2$	● unknown — vs —	● tight 0 vs —	217
Once-colourable nodes on trees	● tight 4 vs 4	● unknown — vs —	● tight 0 vs —	218
Rotor-router: a pointer in every node	● tight $\deg(v)$ vs $\deg(v)$ at each node v	● tight $O(mD)$ vs $\Omega(n^3)$	● tight 0 vs —	218
Rotor-router with pointers set by a designer	● unknown $\deg(v)$ vs —	● unknown $O(m)$ vs —	● tight 0 vs —	220
Label-guided: colours fixed by a designer	● tight 3 vs 2	● unknown $O(m)$ vs $\Omega(n)$	● tight $O(1)$ vs —	220
Label-guided, oblivious agent	● unknown — vs $\lfloor \log \Delta \rfloor - 1$	● unknown — vs —	● tight — vs —	221

Claude (Anthropic) was used in cataloguing the problems.

16.9.1 Recolourable nodes, oblivious agent, any graph

Adds + recolourable, + uniform c_0 STANDARD SETTING VERIFIED

An oblivious agent without port numbers explores every graph and returns with six colours; at least four are needed.

Assumed: graph dynamics: static graph; edge direction: undirected; view: the agent also sees the colours of its neighbours; recolouring: recolourable: any colour, any number of times; initial colouring: uniform: every node has the initial colour c_0. **Not assumed:** port numbers: no port numbers: the agent names a colour, the adversary picks the neighbour; agent memory: oblivious (no memory, one state); knowledge: nothing known about the graph; termination requirement: explore, then stop at the start node; network: arbitrary connected simple graph.

Statement for this setting. The BFUW model: no port numbers, the agent sees the colour of its node and the multiset of its neighbours' colours, recolours its node freely and names a colour to move to; the adversary chooses among neighbours of that colour. All nodes start with c_0. The agent must visit every node, return to s and stop there, for every start and every adversary. Colours are counted including c_0.

Measure	Lower bound	Best upper bound	Status
Colours	4 (Hiraoka et al. 2026) [preprint]	6 (Takahashi et al. 2025)	• tight
	Constant: between 4 and 6 colours (counting c_0); the exact value is open.		
Time	$\Omega(n)$ (Bojko et al. 2022)	—	• unknown
	No time bound is proved for any general-graph algorithm; only the trivial $\Omega(n)$.		
Memory	none known	0 (Takahashi et al. 2025)	• tight
	Zero bits: the agent is oblivious.		

Results.

- Hiraoka et al. (2026) [preprint] [lower bound]: colours 4. *Bottleneck:* indistinguishable starts. *Holds when:* no single rule with at most three colours explores every subcubic pseudotree (graphs with maximum degree three and at most one cycle; Theorem 13); realised on a family of nine graphs with at most five nodes, among them C_3, C_4, C_5 and $K_{1,3}$; computer-assisted: the cases left after a hand reduction are refuted by a SAT solver with a DRAT-checked certificate; uses that two executions from different starts which reach the same configuration cannot both stop at their own start (Lemma 2).
- Bojko et al. (2022) [lower bound]: time $\Omega(n)$. *Bottleneck:* every node visited. *Holds when:* trivial: any agent must visit all n nodes (stated for trees in Bojko et al. 2022).
- Böckenhauer et al. (2023) [introductory]: colours 8, memory 0. Theorem 11 (stated as 7 colours: the paper does not count c_0). Breadth-first: a node at depth j gets label (j mod 3) + 1, in green or red according to the phase, plus a colour for deleted (finished) nodes. Never recolours a node back to c_0. Time not analysed.
- Takahashi et al. (2025) [current best]: colours 6, memory 0. *Assumes:* recolouring back to c_0 allowed; seven colours without it (Note 1). Theorem 1: *semi-DFS*, a depth-first search whose return path is kept an induced path, so that the predecessor of the head is its only neighbour on the path; colours init, path, fin, head1, head2, neigh. Time not analysed.

Open questions.

Q1. Is the colour complexity of recolourable exploration of arbitrary graphs 4, 5 or 6 (Takahashi et al. 2025; Hiraoka et al. 2026)?

Q2. Give a proof of the four-colour lower bound without computer assistance, or for a single graph (Hiraoka et al. 2026).

Q3. How many moves does a constant-colour algorithm need on general graphs? No bound is known; for graphs whose blocks are cliques or triangle-free, five colours and $O(n\Delta)$ moves suffice (Hiraoka et al. 2026).

Claude (Anthropic) was used in cataloguing the problems.

Q4. How many colours suffice if the agent has a constant number of bits, or even one bit, of memory, or may move at random (Böckenhauer et al. 2023; Takahashi et al. 2025)?

Q5. Does forbidding recolouring back to c_0 change the colour complexity, or only the algorithms (Takahashi et al. 2025)?

Notes. Colour counts include the initial colour c_0, as in (Takahashi et al. 2025) and (Hiraoka et al. 2026); Böckenhauer, Frei, Unger and Wehner count without it, so their 7 is 8 here. Restricted classes: five colours suffice for triangle-free graphs (Theorem 2 of the arXiv v2 of Takahashi et al. 2025) and for *φ-free* graphs, those in which, for every node v of degree at least four, each component of $G - v$ contains at most two neighbours of v; they include all graphs of maximum degree three and all cacti (Theorem 3 of Takahashi et al. 2025; the name is the paper's). Hiraoka, Imori, Takahashi and Sudo extend the five-colour algorithm to graphs whose blocks are cliques or triangle-free, with $O(n\Delta)$ moves (Theorem 11), and show that this class is exactly where its cleanup conditions hold (Hiraoka et al. 2026). All known results are for a single agent. Checked against the full texts; (Hiraoka et al. 2026) is a preprint and its lower bound is computer-assisted.

16.9.2 Recolourable nodes on cacti

Adds + cactus **to** Recolourable nodes, oblivious agent, any graph (p. 214) VERIFIED

Four colours explore every cactus in $O(n)$ moves, and four are necessary.

Assumed: graph dynamics: static graph; edge direction: undirected; view: the agent also sees the colours of its neighbours; recolouring: recolourable: any colour, any number of times; initial colouring: uniform: every node has the initial colour c_0; network: cactus (every edge on at most one cycle). **Not assumed:** port numbers: no port numbers: the agent names a colour, the adversary picks the neighbour; agent memory: oblivious (no memory, one state); knowledge: nothing known about the graph; termination requirement: explore, then stop at the start node.

Measure	Lower bound	Best upper bound	Status
Colours	4 (Hiraoka et al. 2026) [preprint] Exactly 4 (counting c_0); the lower bound is computer-assisted.	4 (Hiraoka et al. 2026) [preprint]	• tight
Time	$\Omega(n)$ (Bojko et al. 2022) $\Theta(n)$.	$O(n)$ (Hiraoka et al. 2026) [preprint]	• tight
Memory	none known Zero bits.	0 (Hiraoka et al. 2026) [preprint]	• tight

Results.

- Hiraoka et al. (2026) [preprint] [lower bound]: colours 4. *Bottleneck:* indistinguishable starts. *Holds when:* subcubic pseudotrees, which are cacti (Theorem 13); computer-assisted (SAT certificate checked with DRAT).
- Bojko et al. (2022) [lower bound]: time $\Omega(n)$. *Bottleneck:* every node visited. *Holds when:* trivial: any agent must visit all n nodes (stated for trees in Bojko et al. 2022).
- Takahashi et al. (2025) [also]: colours 5, memory 0. Theorem 3: cacti are φ-free. Uses the three colours d_0, d_1, d_2 to keep a directed acyclic graph that guides forward moves and backtracking. Time not analysed.
- Hiraoka et al. (2026) [preprint] [current best]: colours 4, time $O(n)$, memory 0. *Assumes:* holds for every graph whose blocks are cycles or complete bipartite graphs. Theorem 6. Introductory for four colours and current best.

Open questions.

Q1. Is four colours enough for φ-free graphs, or for graphs of maximum degree three? Both lie between 4 and 5 (Hiraoka et al. 2026).

Claude (Anthropic) was used in cataloguing the problems.

Notes. The four-colour algorithm treats each block on its own, either as a cycle or as a complete bipartite graph $K_{p,q}$, and handles child blocks recursively (Hiraoka et al. 2026). Checked against the full text of the preprint.

16.9.3 Recolourable nodes on trees

Adds + tree **to** Recolourable nodes, oblivious agent, any graph (p. 214) VERIFIED

Three colours explore every tree in $O(n)$ moves; two colours cannot explore the path on three nodes.

Assumed: graph dynamics: static graph; edge direction: undirected; view: the agent also sees the colours of its neighbours; recolouring: recolourable: any colour, any number of times; initial colouring: uniform: every node has the initial colour c_0; network: tree. **Not assumed:** port numbers: no port numbers: the agent names a colour, the adversary picks the neighbour; agent memory: oblivious (no memory, one state); knowledge: nothing known about the graph; termination requirement: explore, then stop at the start node.

Measure	Lower bound	Best upper bound	Status
Colours	3 (Hiraoka et al. 2026) [preprint] Exactly 3 (counting c_0).	3 (Hiraoka et al. 2026) [preprint]	• tight
Time	$\Omega(n)$ (Bojko et al. 2022) $\Theta(n)$.	$O(n)$ (Hiraoka et al. 2026) [preprint]	• tight
Memory	none known Zero bits.	0 (Hiraoka et al. 2026) [preprint]	• tight

Results.

- Hiraoka et al. (2026) [preprint] [lower bound]: colours 3. *Bottleneck:* indistinguishable starts. *Holds when:* the path P_3 on three nodes, every start node (Lemma 4); case analysis of all two-colour rules: each loops, stops early or lets two starts reach the same configuration.
- Bojko et al. (2022) [lower bound]: time $\Omega(n)$. *Bottleneck:* every node visited. *Holds when:* trivial: any agent must visit all n nodes (stated for trees in Bojko et al. 2022).
- Hiraoka et al. (2026) [preprint] [current best]: colours 3, time $O(n)$, memory 0. *Assumes:* the same algorithm explores every simple cycle. Theorem 3. Introductory and current best.

Notes. Without recolouring, trees need four colours (setting *Once-colourable nodes on trees*). Böckenhauer et al. remark that, with recolouring, a path explored from an end node needs only one colour besides c_0: colour forward, then erase on the way back (Böckenhauer et al. 2023); Lemma 4 of (Hiraoka et al. 2026) shows that two colours fail once the start may be the middle node. On simple cycles the same three-colour algorithm works, but no lower bound for cycles alone is stated. Checked against the full text of the preprint.

16.9.4 Once-colourable nodes, any graph

Adds + colour once, + uniform c_0 VERIFIED

If a coloured node can never be recoloured, n colours are necessary and sufficient.

Assumed: graph dynamics: static graph; edge direction: undirected; view: the agent also sees the colours of its neighbours; recolouring: each node may be coloured once; initial colouring: uniform: every node has the initial colour c_0. **Not assumed:** port numbers: no port numbers: the agent names a colour, the adversary picks the neighbour; agent memory: oblivious (no memory, one state); knowledge: nothing known about the graph; termination requirement: explore, then stop at the start node; network: arbitrary connected simple graph.

Claude (Anthropic) was used in cataloguing the problems.

Measure	Lower bound	Best upper bound	Status
Colours	n (Böckenhauer et al. 2023) Exactly n (counting c_0).	n (Böckenhauer et al. 2023)	• tight
Time	none known Time not studied.	—	• unknown
Memory	none known Zero bits.	0 (Böckenhauer et al. 2023)	• tight

Results.

- Böckenhauer et al. (2023) [lower bound]: colours n. *Bottleneck:* predecessor identification. *Holds when:* uniform algorithm: it must explore all graphs, n unknown (Theorem 4; stated as $n-1$ without c_0); for every n there is an n-node graph on which the algorithm uses n colours; the proof forces the agent along a path of a larger graph that it cannot tell from the small one; also holds for bipartite graphs up to one colour: $n-1$ (as summarised in Takahashi et al. 2025).
- Böckenhauer et al. (2023) [current best]: colours n, memory 0. Introductory and current best (Theorem 3, DepthFirstSearch): every newly visited node gets a colour larger than all its coloured neighbours, so the predecessor is always identifiable. Time not analysed.

Open questions.

Q1. For graphs of circumference $\operatorname{circ}(G) \leq g$, close the gap between $\min(2g-2, n)$ and $\min(2g, n)$ colours (Böckenhauer et al. 2023).

Notes. The driving parameter is the circumference $\operatorname{circ}(G)$, the length of a longest cycle (their k): for graphs of circumference at most g, $\min(2g-2, n)$ colours are necessary and $\min(2g, n)$ suffice, counting c_0 (Theorems 8 and 9 of Böckenhauer et al. 2023). A special class can be much cheaper: squares of paths need only five colours although their circumference is maximal (Theorem 10, four without c_0). Checked against the full text.

16.9.5 Once-colourable nodes, n known

Adds + n known **to** Once-colourable nodes, any graph (p. 216) VERIFIED

Knowing n saves one or two colours: $n-1$ suffice and $n-2$ are necessary.

Assumed: graph dynamics: static graph; edge direction: undirected; view: the agent also sees the colours of its neighbours; recolouring: each node may be coloured once; initial colouring: uniform: every node has the initial colour c_0; knowledge: the exact number of nodes n is known. **Not assumed:** port numbers: no port numbers: the agent names a colour, the adversary picks the neighbour; agent memory: oblivious (no memory, one state); termination requirement: explore, then stop at the start node; network: arbitrary connected simple graph.

Measure	Lower bound	Best upper bound	Status
Colours	$n-2$ (Böckenhauer et al. 2023) $n-2$ versus $n-1$: an additive gap of one colour.	$n-1$ (Böckenhauer et al. 2023)	• tight
Time	none known Time not studied.	—	• unknown
Memory	none known Zero bits.	0 (Böckenhauer et al. 2023)	• tight

Results.

- Böckenhauer et al. (2023) [lower bound]: colours $n-2$. *Bottleneck:* predecessor identification. *Holds when:* Theorem 6 (stated as $n-3$ without c_0); realised by cliques of $n-3$ nodes with two pendant leaves and a special node s.

Claude (Anthropic) was used in cataloguing the problems.

- Böckenhauer et al. (2023) [current best]: colours $n-1$, memory 0. *Assumes:* $n \geq 5$. Introductory and current best (Theorem 5, DFS_n; stated as $n-2$ without c_0): the last two nodes share a colour.

Open questions.

Q1. Is the colour complexity with known n equal to $n-2$ or $n-1$ (Böckenhauer et al. 2023)?

Notes. Checked against the full text.

16.9.6 Once-colourable nodes on trees

Adds nothing known, + tree **to** Once-colourable nodes, any graph (p. 216) VERIFIED

Colouring by depth modulo three explores every tree with four colours, and four are necessary.

Assumed: graph dynamics: static graph; edge direction: undirected; view: the agent also sees the colours of its neighbours; recolouring: each node may be coloured once; initial colouring: uniform: every node has the initial colour c_0; network: tree. **Not assumed:** port numbers: no port numbers: the agent names a colour, the adversary picks the neighbour; agent memory: oblivious (no memory, one state); knowledge: nothing known about the graph; termination requirement: explore, then stop at the start node.

Measure	Lower bound	Best upper bound	Status
Colours	4 (Böckenhauer et al. 2023) Exactly 4 (counting c_0).	4 (Böckenhauer et al. 2023)	• tight
Time	none known Not analysed in the paper.	—	• unknown
Memory	none known Zero bits.	0 (Böckenhauer et al. 2023)	• tight

Results.

- Böckenhauer et al. (2023) [lower bound]: colours 4. *Bottleneck:* predecessor identification. *Holds when:* the path on seven nodes started at an end (Theorem 2; stated as 3 without c_0); an uncoloured node cannot be told from the node two steps back, and on the way back two nodes at distance two must differ.
- Böckenhauer et al. (2023) [current best]: colours 4, memory 0. Introductory and current best (Theorem 1, TreeExploration): a node at depth j gets colour $(j \bmod 3) + 1$; the agent prefers uncoloured neighbours, otherwise returns to the parent (the unique neighbour of the previous colour), and stops at the only node of colour 1 with no neighbour of colour 3, the start. Time not analysed.

Notes. The same three-by-depth idea guides label-guided exploration (Cohen et al. 2008) and the breadth-first algorithm with recolouring (Böckenhauer et al. 2023). Checked against the full text.

16.9.7 Rotor-router: a pointer in every node

Adds arbitrary colours, + perpetual VERIFIED

A node's colour is a pointer to one of its ports; an oblivious agent that follows and advances it explores any graph in $O(mD)$ moves from arbitrary pointers, and no oblivious agent without the entry port is faster in the worst case.

Assumed: graph dynamics: static graph; edge direction: undirected; port numbers: port numbers, entry port unknown; recolouring: recolourable: any colour, any number of times; termination requirement: visit every node; never required to stop. **Not assumed:** view: the agent sees its own node only; agent memory: oblivious (no memory, one state); initial colouring: arbitrary initial colours (chosen by the adversary); knowledge: nothing known about the graph; network: arbitrary connected simple graph.

Claude (Anthropic) was used in cataloguing the problems.

Statement for this setting. Ports are numbered, the entry port is not revealed, the agent sees only its own node. The colour of v is a pointer $\pi_v \in \{1, \dots, \deg(v)\}$, initially arbitrary. The agent leaves v by port π_v and advances π_v cyclically. It must visit every node (and keeps going). So a node of degree d uses d colours, $\lceil \log_2 d \rceil$ bits.

Measure	Lower bound	Best upper bound	Status
Colours	$\deg(v)$ at each node v (Menc et al. 2017) $\deg(v)$ colours per node, the minimum.	$\deg(v)$ (Yanovski et al. 2003)	• tight
Time	$\Omega(n^3)$ (Menc et al. 2017)	$O(mD)$ (Yanovski et al. 2003)	• tight
	Worst case over n-node graphs: $O(mD) = O(n^3)$ against $\Omega(n^3)$. For a given graph the rotor-router may be slower than a cleverer rule.		
Memory	none known Zero bits.	0 (Yanovski et al. 2003)	• tight

Results.

- Menc et al. (2017) [lower bound]: colours $\deg(v)$ at each node v. *Bottleneck:* port addressing. *Holds when:* an oblivious agent without the entry port needs $\deg(v)$ distinct inputs at v to leave by every port (Observation 1.1).
- Menc et al. (2017) [lower bound]: time $\Omega(n^3)$. *Bottleneck:* unknown inport. *Holds when:* any oblivious agent without the entry port, even with unbounded node memory (Theorem 2.3); uniform initial node memory, so it holds a fortiori from arbitrary colours; on a path, $(n-1)^2$ moves are needed for a worst-case port numbering (Theorem 2.2), which the rotor-router attains.
- Yanovski et al. (2003) [current best]: colours $\deg(v)$, time $O(mD)$, memory 0. Introductory and current best: within $2|E|D$ moves every arc has been traversed and the walk repeats an Euler tour of the arcs (Corollary 7 and Theorem 3, with $|E| = 2m$ counting both directions of every edge), so every node is visited within $O(mD)$ moves. The paper starts every pointer at the first port of an arbitrary port order, which is the same as an arbitrary pointer; Bampas et al. restate the bound for any initial pointers and ports (their Theorem 1) (Bampas et al. 2017).

Open questions.

Q1. For which graphs (other than paths) is the rotor-router exactly time-optimal among oblivious agents without the entry port (Menc et al. 2017)?

Notes. The rotor-router (also *Propp machine*, *Edge Ant Walk*) goes back to the Eulerian walkers of Priezzhev, D. Dhar, A. Dhar and Krishnamurthy (as cited by Takahashi et al. 2025) (Priezzhev et al. 1996); Yanovski, Wagner and Bruckstein used it for patrolling (Yanovski et al. 2003). Bampas, Gąsieniec, Hanusse, Ilcinkas, Klasing and Kosowski study the *lock-in time*, until the walk has traversed every edge in both directions and repeats an Euler tour; this can exceed the time to visit every node (Bampas et al. 2017). When the adversary sets ports and pointers, in every graph with about m edges and diameter about D some start and setting force at least $mD/4$ moves before lock-in (Theorem 2), so the $O(mD)$ lock-in bound is tight for every such graph. If a designer who knows the ports sets the pointers, lock-in takes $O(m)$ moves (Section 2.1; setting *Rotor-router with pointers set by a designer*). If the pointers must be chosen first and the adversary then numbers the ports, lock-in takes $O(m \min(\log m, D))$ moves (Theorem 3), and some graphs need $\Omega(m \min(\log m, D))$ (Theorem 4); if the adversary sets the pointers, before or after the ports are chosen, some graphs still need $\Omega(mD)$ (Theorems 5 and 7). Once the walk follows an Euler tour, it recovers from faults: if k pointers are set arbitrarily or k edges are added, it follows a new Euler tour within $O(m \min\{k, D\})$ steps, and after an edge is removed without disconnecting the graph, within $2m$ steps times the length of a shortest cycle through that edge (journal version of Bampas et al. 2017, Theorems 10–12). It is a recolourable node whose palette grows with the degree: a deterministic recolouring rule fixed in advance.

Claude (Anthropic) was used in cataloguing the problems.

16.9.8 Rotor-router with pointers set by a designer

Adds + designer's colours, + perpetual VERIFIED

If a designer who knows the graph and its port numbers sets the initial pointers, the rotor-router follows an Euler tour from the first move, so it visits every node and locks in within $O(m)$ moves.

Assumed: graph dynamics: static graph; edge direction: undirected; port numbers: port numbers, entry port unknown; recolouring: recolourable: any colour, any number of times; initial colouring: colours chosen by a designer who knows the graph; termination requirement: visit every node; never required to stop. **Not assumed:** view: the agent sees its own node only; agent memory: oblivious (no memory, one state); knowledge: nothing known about the graph; network: arbitrary connected simple graph.

Statement for this setting. As in the rotor-router setting, the colour of a node is its pointer and the agent leaves by the pointer and advances it cyclically. The adversary numbers the ports; then a designer who knows the graph, its port numbers and the start node chooses the initial pointer at every node (case $A()P(f)$ of Bampas et al. 2017). The agent must visit every node (and keeps going).

Measure	Lower bound	Best upper bound	Status
Colours	none known	$\deg(v)$ (Bampas et al. 2017)	● unknown
	$\deg(v)$ colours per node suffice; no lower bound is known when the designer chooses the initial colours.		
Time	none known	$O(m)$ (Bampas et al. 2017)	● unknown
	Lock-in takes $\Theta(m)$ moves here, since an Euler tour of the arcs has $2m$ of them; the time to visit every node may be smaller, and no lower bound is known for it in this setting.		
Memory	none known	0 (Bampas et al. 2017)	● tight
	Zero bits.		

Results.

- Bampas et al. (2017) [current best]: colours $\deg(v)$, time $O(m)$, memory 0. *Assumes:* the designer knows the port numbers when it sets the pointers. Introductory and current best (Section 2.1, cases P-all and $A()P(f)$): the designer chooses pointers that make the walk follow an Euler tour of the arcs from the start, so every edge is traversed in both directions, and every node visited, within $2m$ moves. The bound is on the *lock-in time*, which bounds the time to visit every node.

Open questions.

Q1. How few moves does a rotor-router with designer pointers need to visit every node, as opposed to locking in? In Hamiltonian graphs the pointers can follow a Hamiltonian cycle (Bampas et al. 2017).

Notes. Bampas, Gąsieniec, Hanusse, Ilcinkas, Klasing and Kosowski let a player who wants a fast lock-in and an adversary choose the pointers and the port order in either order (Bampas et al. 2017). This setting is their case $A()P(f)$: the ports are fixed first and the player answers with the pointers, which reduces to choosing everything and gives $O(m)$ (Section 2.1). If instead the player must set the pointers before the adversary orders the ports (case $P(f)A()$), lock-in takes $O(m \min(\log m, D))$ moves, with pointers towards the child of largest Strahler rank in a breadth-first search tree (Theorem 3), and some graphs need $\Omega(m \min(\log m, D))$ moves before lock-in whatever the pointers (Theorem 4). The catalogue has no setting for that order of choices, so Theorems 3 and 4 are recorded here only; they bound lock-in, not the time to visit every node.

16.9.9 Label-guided: colours fixed by a designer

Adds + finite automaton, + designer's colours, + stop VERIFIED

If a designer colours the nodes with three colours, a finite automaton explores every graph and stops within

Claude (Anthropic) was used in cataloguing the problems.

20m moves.

Assumed: graph dynamics: static graph; edge direction: undirected; port numbers: port numbers and the entry port; agent memory: finite automaton (a constant number K of states); initial colouring: colours chosen by a designer who knows the graph; termination requirement: explore, then stop anywhere. **Not assumed:** view: the agent sees its own node only; recolouring: colours fixed: the agent reads but cannot change them; knowledge: nothing known about the graph; network: arbitrary connected simple graph.

Measure	Lower bound	Best upper bound	Status
Colours	2 (Cohen et al. 2008)	3 (Cohen et al. 2008)	• tight
	Constant: 2 versus 3; whether two colours suffice for arbitrary graphs is open (Cohen et al. 2008).		
Time	$\Omega(n)$ (Bojko et al. 2022)	$O(m)$ (Cohen et al. 2008)	• unknown
	Only the trivial $\Omega(n)$ against $O(m)$.		
Memory	none known	$O(1)$ (Cohen et al. 2008)	• tight
	Constant.		

Results.

- Cohen et al. (2008) [lower bound]: colours 2. *Bottleneck:* trap graphs. *Holds when:* with one colour the labels carry nothing, and no finite automaton explores all graphs (Fraigniaud et al. 2005a) (Section 5 of Cohen et al. 2008); holds even for bounded-degree graphs.

- Bojko et al. (2022) [lower bound]: time $\Omega(n)$. *Bottleneck:* every node visited. *Holds when:* trivial: any agent must visit all n nodes (stated for trees in Bojko et al. 2022).

- Cohen et al. (2008) [current best]: colours 3, time $O(m)$, memory $O(1)$. *Assumes:* fewer than 25 states; labels computed in linear time: colour = depth mod 3 from a root. Introductory and current best (Theorem 2.1): at most $20m$ moves. With 1-bit (two-colour) labels, $O(\log \Delta)$ bits of memory and $O(\Delta^{O(1)}m)$ moves (Theorem 3.1).

Open questions.

Q1. Is there a 1-bit (two-colour) labeling with which a finite automaton explores every graph (Cohen et al. 2008)?

Notes. This is the setting *Node labels chosen by the designer* of single-agent exploration, seen from the colour side: the flag is planted by a designer and the agent cannot change it. The designer's colouring by depth mod 3 is the same device that Böckenhauer et al. let the agent build itself. Zhang, Zhang and Tang trade the share of black nodes in 1-bit labelings against memory (Hu et al. 2012). Checked against the full text.

16.9.10 Label-guided, oblivious agent

Adds oblivious, + designer's colours, + perpetual VERIFIED

Fixed colours cannot replace memory cheaply: an oblivious agent needs more than $\lfloor \log \Delta \rfloor - 2$ designer colours.

Assumed: graph dynamics: static graph; edge direction: undirected; port numbers: port numbers and the entry port; initial colouring: colours chosen by a designer who knows the graph; termination requirement: visit every node; never required to stop. **Not assumed:** view: the agent sees its own node only; agent memory: oblivious (no memory, one state); recolouring: colours fixed: the agent reads but cannot change them; knowledge: nothing known about the graph; network: arbitrary connected simple graph.

Claude (Anthropic) was used in cataloguing the problems.

Measure	Lower bound	Best upper bound	Status
Colours	$\lfloor \log \Delta \rfloor - 1$ (Cohen et al. 2008) No algorithm is catalogued for an oblivious agent with designer colours.	—	● unknown
Time	none known No algorithm catalogued.	—	● unknown
Memory	none known Zero bits.	—	● tight

Results.

- Cohen et al. (2008) [lower bound]: colours $\lfloor \log \Delta \rfloor - 1$. *Bottleneck:* port addressing. *Holds when:* for every $\Delta > 4$ and every one-state agent with at most $\lfloor \log \Delta \rfloor - 2$ colours there is a simple graph of maximum degree Δ it does not explore (Theorem 4.5); with self-loops allowed the bound rises to $\Delta/2$ colours (Theorem 4.2); star-like graphs whose central node is entered and left by ports the agent's colour functions cannot cover; applies to visiting all nodes, so to every termination requirement.

Open questions.

Q1. How many designer colours does an oblivious agent need to explore all graphs of maximum degree Δ (Cohen et al. 2008)?

Notes. With recolourable colours the same oblivious agent needs only a constant number of colours, even without port numbers (setting *Recolourable nodes, oblivious agent, any graph*): the power lies in writing, not in reading. Checked against the full text.

16.10 Online exploration of weighted graphs

One agent must visit every node of an unknown weighted graph and return to its start node, learning the edges at each node only when it gets there; how close to a shortest tour can its cost be?

First posed in 1993 (Kalyanasundaram and Pruhs 1994). Posed by Kalyanasundaram and Pruhs at ICALP 1993 (journal version 1994), who asked whether a constant competitive ratio is possible and gave one for planar graphs, as reported by (Baligács 2026) and (Baligács et al. 2026a); the paper itself was not read.

Problem statement and measures

A single agent starts at a node s of an unknown connected undirected **graph** $G = (V, E)$ with n **nodes**; every **edge** e has a non-negative **weight** $\mathrm{wt}(e)$. Nodes and edges carry distinct identifiers. When the agent visits a node for the first time it learns the identifiers of its neighbours and the identifiers and weights of the edges to them. Crossing an edge has a **cost** equal to its weight. The agent must visit every node and return to the **start node** s; its walk is then a **tour**. An algorithm is measured by its **competitive ratio**: the supremum, over all graphs and start nodes, of the cost of its tour divided by OPT, the cost of a shortest closed walk that visits every node (an optimal travelling-salesperson tour of G); for a randomized algorithm the expected cost is used. Unless a setting says otherwise the algorithm is deterministic, the graph is arbitrary and nothing about it is known in advance. Memory and computation are unbounded: the question is what the lack of information costs. Since OPT lies between the weight of a minimum spanning tree and twice that weight, a depth-first search that ignores the weights already explores trees optimally, and the difficulty lies in the cycles.

Claude (Anthropic) was used in cataloguing the problems.

Measures: **Competitive ratio**, worst-case cost of the agent's tour divided by OPT, the cost of a shortest tour.



(a) Start at s: the agent sees only the two edges at s; it takes the lighter one

(b) Final: cost 12, which equals the shortest tour here

Figure 16.10: Online exploration of a cycle by HeavyTest (Baligács et al. 2026b): the agent learns an edge and its weight only when it reaches one of its ends (weights are shown once known; visited nodes are green). The explored part is a path with a boundary edge at each end. With b the weight of the edge ahead, a the distance back to s and d the distance from s to the far end of the other boundary edge, the agent goes on if $b \le \sqrt{3}\,a + d$ and otherwise turns back, crossing the other boundary edge. Here it refuses the edge of weight 10, which a shortest tour also avoids, and its cost equals the optimum of 12. In the worst case the ratio is $(1+\sqrt{3})/2$, the best possible for a deterministic agent on cycles (Miyazaki et al. 2009). One frame per edge crossed.

A first look

Imagine a courier in a city without a map. At each crossing the courier sees the roads that leave it, where they lead and how long they are, and must visit every crossing and come home, walking as little as possible. The city is a **graph**: crossings are **nodes** and roads are **edges**, and each edge has a **weight** ≥ 0, its length. The courier is an **agent** that starts at the **start node** s. Nodes and edges have distinct names. When the agent first visits a node, it learns the names of its neighbours and the weights of the edges to them. Crossing an edge costs its weight, and the **cost** of a walk is the total. The agent must visit every node and return to s, and its walk is then a **tour**. We compare it with OPT, the cost of a shortest tour, which someone who knew the graph could take. The **competitive ratio** is the worst case, over all graphs and start nodes, of the agent's cost divided by OPT. In the standard version the graph is arbitrary and the agent is deterministic: it tosses no coins. Memory and computation are unlimited, so the only difficulty is the missing information.

1. Algorithm: On a cycle: go on, or turn back. (Baligács et al. 2026b) (Weighted cycles, p. 231) Suppose the graph is a cycle. The explored part is then always a path, with a **boundary edge** at each end: an edge from a visited node to a node not yet visited. The agent stands at one end.

1. First cross the lighter of the two edges at s.
2. Let b be the weight of the boundary edge ahead, a the distance back to s, and d the distance from s to the far end of the other boundary edge. Going on costs b, and turning back to cross the other boundary edge costs $a + d$.
3. Go on if $b \le \sqrt{3}\,a + d$. Otherwise turn back and cross the other boundary edge.
4. When every node has been visited, return to s along a shortest path.

Why it works and what it costs. The rule is HeavyTest of (Baligács et al. 2026b). It weighs two risks: crossing a heavy edge that a shortest tour would avoid, and walking all the way back for an edge

Claude (Anthropic) was used in cataloguing the problems.

that a shortest tour would use. Their proof shows that the factor $\sqrt{3}$ gives the competitive ratio $(1+\sqrt{3})/2 \approx 1.366$. No deterministic agent does better on cycles (Miyazaki et al. 2009), whose algorithm reached the same ratio first with a rule that also uses the cost spent so far. An agent that tosses coins does better on cycles, at most 1.3144.

Why it matters. It isolates the basic dilemma of online exploration, whether to pay for the edge in front or go back, and solves it exactly.

2. Algorithm: Depth-first search, and why weights make it hard. (Miyazaki et al. 2009) (Arbitrary weighted graphs, p. 227) **Depth-first search** ignores the weights: go to an unvisited neighbour while there is one, otherwise go back the way you came, and stop when you are back at s with nothing left to try. The edges it uses to reach new nodes form a tree that contains every node, and it crosses each of them exactly twice.

Why it works and what it costs. On a tree a shortest tour also crosses every edge twice, so depth-first search is optimal there. If every edge has weight 1, depth-first search costs $2(n-1)$, while any tour crosses at least n edges, so the ratio is below 2 (Theorem 4 of Miyazaki et al. 2009). With weights it can be arbitrarily bad. On a cycle it may cross a very heavy edge twice, where a shortest tour goes out and back on the light side. A shortest tour costs between the weight of a minimum spanning tree (the lightest set of edges that connects all nodes) and twice that. The challenge is to explore at a cost close to that weight without knowing the graph.

Why it matters. It shows where the difficulty lies: not in visiting everything, but in avoiding heavy edges that are not yet known to be avoidable.

3. Algorithm: Blocking: depth-first search that waits for lighter edges. (Baligács et al. 2026a) (Graphs excluding a fixed minor, p. 229) Fix a number $\chi > 0$. A boundary edge of weight w is **blocked** if some lighter boundary edge leads to an unvisited node at distance at most $(1+\chi)\,w$ from the visited end of the first edge, along visited nodes.

1. Run depth-first search, but never cross a blocked edge.
2. Whenever a blocked edge becomes unblocked, walk to it, cross it, and start a new depth-first search from there.
3. Stop when every node has been visited, and return to s.

Why it works and what it costs. Each boundary edge that the agent crosses pays for its own trip, at most $2(\chi+2)$ times its weight (Observation 6). So the question is how heavy these edges are together compared with a minimum spanning tree. Baligács, Disser, Heinrich and Schweitzer show that they behave like a **light spanner**: a set of edges that keeps every distance within a factor $1+\chi$ and weighs at most a constant times a minimum spanning tree. Every class of graphs that excludes a fixed minor, for example planar graphs, has light spanners, so there Blocking is constant-competitive (Baligács et al. 2026a). On planar graphs the ratio is 16, the bound of Kalyanasundaram and Pruhs (Kalyanasundaram and Pruhs 1994), as reported by (Baligács et al. 2026a). On all graphs, with $\chi = \log n$, it is $O(\log n)$-competitive, but no choice of χ makes it constant. Whether any algorithm is constant-competitive on all graphs is open; no deterministic algorithm is better than 4 (Baligács 2026).

Why it matters. It is the main tool for the central open question of Kalyanasundaram and Pruhs, and it ties online exploration to spanners, a basic object of graph theory.

Claude (Anthropic) was used in cataloguing the problems.

What the community started looking at

Kalyanasundaram and Pruhs posed the problem, asked whether some algorithm has a constant competitive ratio and gave one for planar graphs (Kalyanasundaram and Pruhs 1994), as reported by (Baligács 2026) and (Baligács et al. 2026a). The question is still open. The nearest-neighbour rule (always go to the closest unvisited node), analysed as a travelling-salesperson heuristic by Rosenkrantz, Stearns and Lewis in 1977, is $\Theta(\log n)$-competitive, and no algorithm is known to do better on all graphs, as reported by (Baligács 2026). Megow, Mehlhorn and Schweitzer (2012) revisited the planar algorithm as Blocking and extended it to graphs of bounded genus (as reported by Baligács et al. 2026a); Baligács, Disser, Heinrich and Schweitzer extended it to every class that excludes a fixed minor, by relating Blocking to light spanners (Baligács et al. 2026a). Lower bounds for deterministic algorithms rose from 2, which holds on unweighted planar graphs and is matched there by depth-first search (Miyazaki et al. 2009), through 2.5 (Dobrev, Královič and Markou, 2012, as reported by Baligács 2026) and 10/3, which also holds on planar graphs (Birx et al. 2021), to 4 (Baligács 2026). On cycles the deterministic ratio is exactly $(1+\sqrt{3})/2$ (Miyazaki et al. 2009), and randomization was first shown to help there in 2026 (Baligács et al. 2026b). Gehnen, Klasing and Naquin introduced a semi-online version in which the graph is known and only the weights are uncertain (Gehnen et al. 2025).

Further reading

Papers on this problem that are not yet placed in a setting.

- Brandt et al. (2020a). Tadpole graphs (a cycle with a path attached): the greedy nearest-neighbour rule is 2-competitive for any non-negative weights, matching the lower bound $2-\varepsilon$ obtained from the construction of (Miyazaki et al. 2009); also advice complexity on cycles and tadpoles. Abstract read.
- van den Akker et al. (2024). Online exploration by 2 to 4 agents that share what they see, on cycles and tadpole graphs, for the time and for the longest walk: optimal ratios in several cases (1 with two agents on cycles for the longest walk). Abstract read.
- Böckenhauer et al. (2018). Advice complexity of online exploration in the fixed graph scenario (directed or undirected, cyclic or not, unit or individual costs): how many bits of advice an optimal tour needs. A preprint; abstract read.
- Böckenhauer et al. (2016). The agent knows the graph but sees edge weights (1 or k) only at their ends: lower bound 11/9 and a 4-competitive algorithm on ladders; greedy has a linear ratio on directed ladders and grids. Abstract read.

Relations to other problems

- **Variant of Graph exploration by a single agent.** The same task for one agent (visit every node and return to the start node), in the tradition of online algorithms: nodes carry identifiers, edges carry weights, memory and computation are unbounded, and an algorithm is judged by its competitive ratio against a shortest tour rather than by its memory. Results do not transfer in either direction without care: the parent counts moves on anonymous unweighted graphs, this problem counts weight on labelled weighted graphs. The setting directed-labelled of the parent (Deng and Papadimitriou 1999) is the analogue for unweighted directed graphs.
- **Related: Collaborative exploration.** Collective tree exploration also measures algorithms by their competitive ratio against the offline optimum, but for k agents on unweighted trees and for time; here one agent explores a weighted graph and its cost is compared with a shortest tour.

Claude (Anthropic) was used in cataloguing the problems.

Assumptions specific to this problem

Knowledge of the graph

Gehnen, Klasing and Naquin introduced weight estimates to model a traveller who knows the street map but learns the actual travel time of a road only on reaching it; the setting lies between the offline travelling-salesperson problem and online exploration (Gehnen et al. 2025). With $\vartheta = 1$ it is the offline problem; as ϑ grows it approaches online exploration, whose lower bounds apply once the intervals are wide enough (Section 2 of Gehnen et al. 2025).

Level	Value	Formal definition
0	**nothing known**	The agent knows nothing about G before it starts; the edges at a node and their weights are revealed when the agent first visits the node.
1	**weight estimates**	The agent is given G and, for every edge e, an interval $[\mathrm{wt}^-(e), \mathrm{wt}^+(e)]$ that contains $\mathrm{wt}(e)$, with $\mathrm{wt}^+(e) \leq \vartheta\, \mathrm{wt}^-(e)$ for a factor $\vartheta \geq 1$ (their α); the exact weight of e is revealed when the agent visits an end of e (Gehnen et al. 2025).

Excluded minors

Restricting the graph class is how constant competitive ratios have been obtained: Kalyanasundaram and Pruhs for planar graphs (Kalyanasundaram and Pruhs 1994), Megow, Mehlhorn and Schweitzer for graphs of bounded genus (2012, as reported by Baligács et al. 2026a), and Baligács, Disser, Heinrich and Schweitzer for every class with a forbidden minor, which they show subsumes all classes previously known to admit a ratio of $o(\log n)$ (Baligács et al. 2026a).

Level	Value	Formal definition
0	**any graph**	G is any connected graph.
1	**H-minor-free**	For a fixed graph H, no graph obtained from G by deleting nodes and edges and contracting edges is isomorphic to H. Graphs of bounded genus and of bounded treewidth are such classes; bounds may depend on H.
2	**planar**	G can be drawn in the plane without crossing edges; equivalently it has neither K_5 nor $K_{3,3}$ as a minor.

Case tree

The root assumes nothing; going down, each edge adds the assumptions written in grey (+) or leaves an axis unassumed. Bold entries are catalogued settings with their status per measure (Competitive ratio) and page; italic entries are branch points, named after the axis on which their branches differ. Axis order: Graph dynamics → Edge direction → Node identifiers → Randomness → Knowledge of the graph → Excluded minors → Topology.

No assumptions

+ static graph, + undirected, + labelled nodes

Arbitrary weighted graphs STANDARD SETTING ● p. 227

deterministic, + weight estimates

Known graph, weight estimates ● p. 228

+ complete graph

Complete graphs, uniform weight estimates ● p. 229

Claude (Anthropic) was used in cataloguing the problems.

deterministic, nothing known, + H-minor-free
Graphs excluding a fixed minor ● p. 229

deterministic, nothing known, + planar
Planar graphs ● p. 230

deterministic, nothing known, any graph, + ring
Weighted cycles ● p. 231

+ private coins, + ring
Weighted cycles, randomized agent ● p. 231

What is known

The best upper bound against the best lower bound in each catalogued setting, in tree order. The settings follow in detail.

Setting	Competitive ratio	p.
Arbitrary weighted graphs STANDARD SETTING	● polynomial gap $O(\log n)$ vs 4	227
Known graph, weight estimates	● polynomial gap ϑ vs $\min(\vartheta, 2)$	228
Complete graphs, uniform weight estimates	● tight $\frac{1+\vartheta}{2}$ vs $\frac{1+\vartheta}{2}$	229
Graphs excluding a fixed minor	● tight $O(1)$ vs 4	229
Planar graphs	● tight 16 vs 4	230
Weighted cycles	● tight $\frac{1+\sqrt{3}}{2}$ vs $\frac{1+\sqrt{3}}{2}$	231
Weighted cycles, randomized agent	● tight $1+\frac{(3-\sqrt{2})^2}{8}$ vs $1+\frac{\sqrt{2}}{8+3\sqrt{2}}$	231

16.10.1 Arbitrary weighted graphs

Adds + static graph, + undirected, + labelled nodes STANDARD SETTING VERIFIED

The question of Kalyanasundaram and Pruhs: the competitive ratio is between 4 and $O(\log n)$, and whether it is constant is open.

Assumed: graph dynamics: static graph; edge direction: undirected; node identifiers: labelled nodes (an agent can recognise a node it has visited). **Not assumed:** randomness: deterministic; knowledge of the graph: nothing known in advance; excluded minors: any connected graph; topology: arbitrary connected graph.

Measure	Lower bound	Best upper bound	Status
Competitive ratio	4 (Baligács 2026) [preprint] $O(\log n)$ against 4.	$O(\log n)$ (Baligács et al. 2026a)	● polynomial gap

Results.

- Baligács (2026) [preprint] [lower bound]: competitive ratio 4. *Bottleneck:* online graph revelation. *Holds when:* deterministic algorithms; the adversary builds the graph as the agent explores it; holds on subcubic planar graphs (Theorem 1); holds even if the agent is told n, if it learns only the identifiers and weights of the edges at

Claude (Anthropic) was used in cataloguing the problems.

a node and not the identifiers of the neighbours, and if the weights satisfy the triangle inequality: none of these changes the competitive ratio (Proposition 2, Corollary 5). Theorem 1, improving 10/3 of Birx, Disser, Hopp and Karousatou (2021), as reported. Depth-first search that ignores the weights is 4-competitive on the construction, so its analysis is tight. Additive constants do not change the ratio of the problem (Observation 3).

- Baligács et al. (2026a) [current best]: competitive ratio $O(\log n)$. Theorem 2 (Corollary 13c): Blocking with parameter $\log n$. The same ratio as the nearest-neighbour rule of Rosenkrantz, Stearns and Lewis (1977) and hierarchical depth-first search of Megow, Mehlhorn and Schweitzer (2012), as reported; neither paper is in the bibliography.

Open questions.

Q1. Is there a deterministic algorithm with a constant competitive ratio (Kalyanasundaram and Pruhs Kalyanasundaram and Pruhs 1994, as reported by Baligács 2026)?

Q2. Is there an $o(\log n)$-competitive algorithm? Blocking with parameter χ (their δ) is $\Omega(\log n)$-competitive for $\chi = o(\log n/\log\log n)$ and for $\chi = \Omega(\log n)$, and $\Omega(\log n/\log\log n)$-competitive for every χ; the range in between is open (Baligács et al. 2026a).

Q3. Does randomization lower the competitive ratio on general graphs? It does on cycles (Baligács et al. 2026b).

Notes. The restrictions of (Baligács 2026) (Proposition 2) also say what a sensible algorithm may not need: the identifiers of unvisited neighbours, knowledge of n, degrees above 3 and weights that violate the triangle inequality do not change the ratio; they are lost when proving bounds that grow with n. Theorem 3 of (Baligács et al. 2026a) rules out Blocking with any parameter as a constant-competitive algorithm. Bounds checked against the full texts (arXiv versions).

16.10.2 Known graph, weight estimates

Adds deterministic, + weight estimates **to** Arbitrary weighted graphs (p. 227) VERIFIED

If the graph is known and every weight is known up to a factor $\vartheta \le 2$, following a shortest tour for the lower estimates is ϑ-competitive, and nothing does better.

Assumed: graph dynamics: static graph; edge direction: undirected; node identifiers: labelled nodes (an agent can recognise a node it has visited); knowledge of the graph: the graph is known, each weight only up to an interval. **Not assumed:** randomness: deterministic; excluded minors: any connected graph; topology: arbitrary connected graph.

Statement for this setting. The agent knows G and, for each edge, an interval that contains its weight, with ratio at most ϑ between the ends; the weight is revealed when the agent reaches an end of the edge. In (Gehnen et al. 2025) the agent must walk from s to a given node $t \neq s$ through all nodes; the authors note that the bounds carry over to tours up to one edge.

Measure	Lower bound	Best upper bound	Status
Competitive ratio	$\min(\vartheta, 2)$ (Gehnen et al. 2025) [preprint]	ϑ (Gehnen et al. 2025) [preprint]	• polynomial gap
	Exactly ϑ for $\vartheta \le 2$; for $\vartheta > 2$, ϑ against 2.		

Results.

- Gehnen et al. (2025) [preprint] [lower bound]: competitive ratio $\min(\vartheta, 2)$. *Bottleneck:* online graph revelation. *Holds when:* a recursive instance of paths whose weights are announced in $[1, \vartheta]$; the adversary reveals each weight when the agent arrives (Theorem 3 for $\vartheta \le 2$; Remark 4 gives 2 for $\vartheta \ge 2$).

Claude (Anthropic) was used in cataloguing the problems.

- Gehnen et al. (2025) [preprint] [current best]: competitive ratio ϑ. *Assumes:* unbounded computation: the walk solves a travelling-salesperson problem in advance. Introductory and current best (Theorem 5): compute a shortest walk for the lower ends of the intervals and follow it, ignoring the revealed weights.

Open questions.

Q1. Does every graph admit a $\frac{1+\vartheta}{2}$-competitive algorithm when all weights are announced in the same interval $[1, \vartheta]$ with $\vartheta \leq 2$? Conjectured by (Gehnen et al. 2025).

Notes. Gehnen, Klasing and Naquin. Lower-bound constructions for online exploration hide the structure of the graph (dead ends), which is impossible here; they transfer only when the intervals are wide (Section 2 of Gehnen et al. 2025). Checked against the full text (arXiv v1, 2025); a LATIN 2026 version is cited by (Baligács et al. 2026b) and was not read.

16.10.3 Complete graphs, uniform weight estimates

Adds + complete graph **to** Known graph, weight estimates (p. 228) VERIFIED

On complete graphs whose weights all lie in $[1, \vartheta]$ with $\vartheta < 2$, recomputing the best walk after each step is $\frac{1+\vartheta}{2}$-competitive, which is optimal.

Assumed: graph dynamics: static graph; edge direction: undirected; node identifiers: labelled nodes (an agent can recognise a node it has visited); knowledge of the graph: the graph is known, each weight only up to an interval; topology: complete graph. **Not assumed:** randomness: deterministic; excluded minors: any connected graph.

Measure	Lower bound	Best upper bound	Status
Competitive ratio	$\frac{1+\vartheta}{2}$ (Gehnen et al. 2025) [preprint] Exactly $\frac{1+\vartheta}{2}$ for $\vartheta < 2$.	$\frac{1+\vartheta}{2}$ (Gehnen et al. 2025) [preprint]	• tight

Results.

- Gehnen et al. (2025) [preprint] [lower bound]: competitive ratio $\frac{1+\vartheta}{2}$. *Bottleneck:* online graph revelation. *Holds when:* complete graph on $2j$ nodes, every weight announced in $[1, \vartheta]$, $\vartheta \leq 2$; edges at the first j nodes visited have weight 1, edges revealed later weight ϑ (Theorem 10); for $\vartheta > 2$ the same construction gives $3/2$ (Remark 11), and an analogous one works on $K_{j,j}$.
- Gehnen et al. (2025) [preprint] [current best]: competitive ratio $\frac{1+\vartheta}{2}$. *Assumes:* every weight announced in $[1, \vartheta]$ with $\vartheta < 2$; start and end nodes distinct; unbounded computation. Introductory and current best (Theorem 6, Corollary 8): after every step, recompute a walk through the unvisited nodes that is shortest when unrevealed weights take their upper ends. On such graphs it takes the cheapest edge to an unvisited node (Lemma 7). The same holds on every graph in which each path extends to a Hamiltonian path to the end node, including $K_{j,j}$ and $K_{j+1,j}$ (Corollary 9).

Notes. On grids with uniform estimates and $\vartheta < 2$ the recomputing algorithm is no better than ϑ-competitive (Theorem 12 of Gehnen et al. 2025). Checked against the full text (arXiv v1, 2025).

16.10.4 Graphs excluding a fixed minor

Adds deterministic, nothing known, + H-minor-free **to** Arbitrary weighted graphs (p. 227) VERIFIED

Blocking is constant-competitive on every class of graphs that excludes a fixed minor, through light spanners.

Assumed: graph dynamics: static graph; edge direction: undirected; node identifiers: labelled nodes (an agent can recognise a node it has visited); excluded minors: excludes a fixed graph H as a minor. **Not assumed:** randomness: deterministic; knowledge of the graph: nothing known in advance; topology: arbitrary connected graph.

Claude (Anthropic) was used in cataloguing the problems.

Measure	Lower bound	Best upper bound	Status
Competitive ratio	4 (Baligács 2026) [preprint]	$O(1)$ (Baligács et al. 2026a)	• tight
	Both constant: at least 4 for every non-planar H, at most a constant that depends on H.		

Results.

- Baligács (2026) [preprint] [lower bound]: competitive ratio 4. *Bottleneck:* online graph revelation. *Holds when:* planar bound: holds for every non-planar excluded minor H, since planar graphs have only planar minors; deterministic algorithms.
- Baligács et al. (2026a) [current best]: competitive ratio $O(1)$. *Assumes:* the constant depends on H and on the parameter of Blocking. Introductory and current best (Theorem 1; Corollaries 11 and 13b). Blocking with parameter χ (their δ) is $2(\chi+2)$Y-competitive on every class closed under subgraphs in which every graph has a $(1+\chi)$-spanner of weight at most Y times that of a minimum spanning tree (Theorem 9), and H-minor-free graphs have such spanners with Y depending on H and χ. On graphs of genus at most $\mathfrak{g}$ the ratio is $16(1+\frac{2}{3}\mathfrak{g})$ (Corollary 13a), slightly better than $16(1+2\mathfrak{g})$ of Megow, Mehlhorn and Schweitzer, as reported.

Open questions.

Q1. How does the best constant depend on the excluded minor H, and on the genus for graphs of bounded genus (Baligács et al. 2026a)?

Notes. Baligács, Disser, Heinrich and Schweitzer run the connection both ways: the exploration analysis also gives greedy $(1+\varepsilon)$-spanners of lightness at most $(1+\frac{2}{\varepsilon})(1+\frac{2\mathfrak{g}}{1+\varepsilon})$ on graphs of genus $\mathfrak{g}$ (Theorem 4), tight for planar graphs. They suggest that a non-constant lower bound for exploration may require dense graphs of high girth or expanders (Section 4). Checked against the full text (arXiv v1, 2023); the cited version is the JCSS article.

16.10.5 Planar graphs

Adds deterministic, nothing known, + planar **to** Arbitrary weighted graphs (p. 227) DRAFT

The first class with a constant ratio: 16-competitive since 1993, against a lower bound of 4 that holds already on planar graphs.

Assumed: graph dynamics: static graph; edge direction: undirected; node identifiers: labelled nodes (an agent can recognise a node it has visited); excluded minors: planar. **Not assumed:** randomness: deterministic; knowledge of the graph: nothing known in advance; topology: arbitrary connected graph.

Measure	Lower bound	Best upper bound	Status
Competitive ratio	4 (Baligács 2026) [preprint]	16 (Kalyanasundaram and Pruhs 1994)	• tight
	Constants: between 4 and 16.		

Results.

- Baligács (2026) [preprint] [lower bound]: competitive ratio 4. *Bottleneck:* online graph revelation. *Holds when:* deterministic algorithms; subcubic planar graphs (Theorem 1).
- Kalyanasundaram and Pruhs (1994) [current best]: competitive ratio 16. Introductory and current best, as reported by (Baligács et al. 2026a) and (Gehnen et al. 2025); not read. Corollary 13a of (Baligács et al. 2026a) with genus 0 recovers 16 for Blocking with parameter 2, through planar spanners of lightness $1+2/\chi$.

Open questions.

Claude (Anthropic) was used in cataloguing the problems.

Q1. Close 4 against 16 on planar graphs.

Notes. Draft because the introductory paper was not read; the bound 16 was checked as a consequence of Corollary 10 of (Baligács et al. 2026a).

16.10.6 Weighted cycles

Adds deterministic, nothing known, any graph, + ring **to** Arbitrary weighted graphs (p. 227) VERIFIED

On a cycle the deterministic competitive ratio is exactly $(1+\sqrt{3})/2 \approx 1.366$.

Assumed: graph dynamics: static graph; edge direction: undirected; node identifiers: labelled nodes (an agent can recognise a node it has visited); topology: ring or path. **Not assumed:** randomness: deterministic; knowledge of the graph: nothing known in advance; excluded minors: any connected graph.

Measure	Lower bound	Best upper bound	Status
Competitive ratio	$\frac{1+\sqrt{3}}{2}$ (Miyazaki et al. 2009)	$\frac{1+\sqrt{3}}{2}$ (Miyazaki et al. 2009)	• tight
	Exactly $(1+\sqrt{3})/2$ for deterministic algorithms.		

Results.

- Miyazaki et al. (2009) [lower bound]: competitive ratio $\frac{1+\sqrt{3}}{2}$. *Bottleneck:* online graph revelation. *Holds when:* deterministic algorithms; Theorem 1: no $((1+\sqrt{3})/2-\varepsilon)$-competitive algorithm, for every constant $\varepsilon > 0$.
- Miyazaki et al. (2009) [current best]: competitive ratio $\frac{1+\sqrt{3}}{2}$. Theorem 2, algorithm Dist: at each step the agent crosses the boundary edge ahead if and only if its weight is at most $\sqrt{3}$ times the known distance to the unvisited end of the other boundary edge, minus the cost so far less the distance back to s; otherwise it turns back. Its choices depend on the cost so far. Before it, Asahiro, Miyano, Miyazaki and Yoshimuta showed that the nearest-neighbour rule is 3/2-competitive on cycles, against a lower bound of 5/4 (SOFSEM 2007), as reported by (Miyazaki et al. 2009).
- Baligács et al. (2026b) [preprint] [also]: competitive ratio $\frac{1+\sqrt{3}}{2}$. Theorem 1, algorithm HeavyTest: with b the weight of the boundary edge at the agent, a its distance back to s and d the distance from s to the unvisited end of the other boundary edge (so turning back costs $a+d$), cross the edge ahead if and only if $b \le \sqrt{3}\,a + d$. It depends only on the current state, and its analysis is shorter.

Notes. On a cycle the explored part is a path with two boundary edges, and the only decision is whether to continue or turn back; the difficulty is an edge heavier than all others together, which a shortest tour avoids. Trees are explored optimally by depth-first search (Baligács et al. 2026b).

16.10.7 Weighted cycles, randomized agent

Adds + private coins, + ring **to** Arbitrary weighted graphs (p. 227) VERIFIED

A randomized agent beats the deterministic ratio on cycles: at most 1.3144, *against a lower bound of* 1.1155.

Assumed: graph dynamics: static graph; edge direction: undirected; node identifiers: labelled nodes (an agent can recognise a node it has visited); randomness: randomized (agents have private coins); topology: ring or path. **Not assumed:** knowledge of the graph: nothing known in advance; excluded minors: any connected graph.

Measure	Lower bound	Best upper bound	Status
Competitive ratio	$1+\frac{\sqrt{2}}{8+3\sqrt{2}}$ (Baligács et al. 2026b) [preprint]; $\frac{1+\sqrt{2}}{2}$ (Baligács et al. 2026b) [preprint]	$1+\frac{(3-\sqrt{2})^2}{8}$ (Baligács et al. 2026b) [preprint]	• tight
	Constants: between 1.1155 and 1.3144 (between 1.2071 and 1.3144 for forward-greedy algorithms).		

Claude (Anthropic) was used in cataloguing the problems.

Results.

- Baligács et al. (2026b) [preprint] [lower bound]: competitive ratio $1 + \frac{\sqrt{2}}{8+3\sqrt{2}}$. *Bottleneck:* online graph revelation. *Holds when:* any randomized algorithm, against an adversary that fixes the cycle in advance (oblivious); Yao's principle on three cycles of four nodes that look the same from the start node (Theorem 3). About 1.1155.
- Baligács et al. (2026b) [preprint] [lower bound]: competitive ratio $\frac{1+\sqrt{2}}{2}$. *Bottleneck:* online graph revelation. *Holds when:* only for forward-greedy algorithms: those that cross the edge ahead whenever its weight b is at most $a+d$, the cost of turning back to the unvisited end of the other boundary edge; RandHeavyTest is one (Theorem 4). About 1.2071.
- Baligács et al. (2026b) [preprint] [current best]: competitive ratio $1 + \frac{(3-\sqrt{2})^2}{8}$. *Assumes:* oblivious adversary; the analysis also holds against an adversary that adapts in a limited way to the random choices. Introductory and current best (Theorem 2), about 1.3144: RandHeavyTest with parameter $1/2$ crosses the edge ahead when $b \le \frac{3}{2}a + d$ and otherwise with probability $a/(2(b-d-a))$. The proof is a supermartingale argument with a stopping time.

Open questions.

Q1. Determine the randomized competitive ratio on cycles: 1.1155 against 1.3144 (Baligács et al. 2026b).

Q2. Must an optimal randomized algorithm on cycles be forward-greedy (Baligács et al. 2026b)?

Notes. Baligács, Hązła and Volk give the first proof that randomization helps in online graph exploration; Miyazaki, Morimoto and Okabe, who settled deterministic algorithms on cycles, name randomized algorithms as future work (Miyazaki et al. 2009). The revealed part of the graph depends on the agent's earlier random choices, which is what makes randomized exploration hard to analyse. Checked against the full text (arXiv v1, 2026), a preprint.



16.11 Collision-free exploration

Agents start at distinct nodes of a graph and must explore it so that no two agents are ever at the same node; how long does it take, and what must the agents sense of each other?

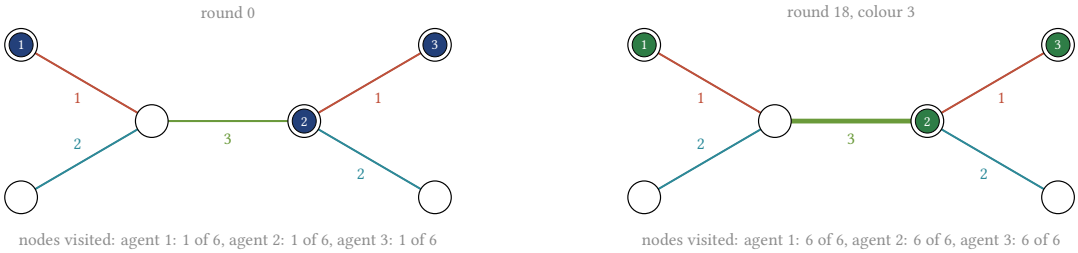
First posed in 2012 (Czyzowicz et al. 2017a). Formalised by Czyzowicz, Dereniowski, Gąsieniec, Klasing, Kosowski and Pająk for agents on graphs with labelled nodes (a 2012 report; LATIN 2014; Journal of Computer and System Sciences 2017); (Papin 2026) calls this the first formalisation. Bhagat and Pelc posed it for anonymous graphs (Bhagat and Pelc 2025).

Problem statement and measures

$k \ge 2$ **agents** start at distinct **nodes** of a connected undirected **graph** G with n nodes and move in synchronous **rounds**: in each round an agent stays or crosses one **edge**. A **collision** is two agents at the same node in the same round; two agents that cross one edge in opposite directions do not collide and do not notice it. Exploration is **collision-free** if no collision ever happens, also with an agent that an **adversary** has not yet woken up. Unless a setting says otherwise, every agent must visit every node and then stop. The agents run the same deterministic algorithm, cannot mark nodes and send no messages; what they perceive of each other (their **vision**) and what they know of G are fixed by the setting, and in some settings an oracle places coloured **pebbles** on nodes before the start. **Time** is the number of rounds from the wake-up of the later agent until every agent has stopped. $X(n)$ is the time of a single-agent exploration procedure for graphs with at most n nodes, used as a building block (polynomial in n, for instance with a universal exploration sequence).

Claude (Anthropic) was used in cataloguing the problems.

Measures: **Time**, rounds from the wake-up of the later agent until all agents have stopped; **Colours**, number of pebble colours the oracle needs, when pebbles are used.



(a) Start: a map with a 3-colouring of the edges; agents on three nodes

(b) Final, round nΔ = 18: every agent visited all 6 nodes and is back home

Figure 16.11: Collision-free exploration of a tree with a map, Algorithm Tree-Exploration of (Czyzowicz et al. 2017a): every agent computes the same colouring of the edges with $\Delta = 3$ colours (no two edges at a node share a colour). In round r the colour is $1 + (r - 1) \bmod \Delta$ (thick edges), and every agent crosses the edge of that colour at its node if there is one, otherwise it waits. The edges of one colour share no node, so two agents can only swap across an edge, which is not a collision. After $n\Delta = 18$ rounds every agent has visited every node and is back at its start node, and no algorithm is faster on this tree.

A first look

Picture night guards in a museum who must each check every room, where no two guards may ever stand in the same room at the same moment. The museum is a **graph**: its rooms are **nodes** and the doorways between them are **edges**; it has n nodes. The guards are **agents**. They start on distinct nodes and move in synchronous **rounds**: in each round an agent stays where it is or crosses one edge. A **collision** is two agents on the same node in the same round. Two agents that cross the same edge in opposite directions pass each other unnoticed, and that is not a collision. Every agent must visit every node and then stop, without any collision. An **adversary**, an opponent who knows the algorithm, chooses where the agents start and the round in which each one wakes up. The agents run the same deterministic algorithm, cannot mark nodes and send no messages. What they perceive of each other is their **vision**. In the standard version there are two identical agents that see everything within two edges of their node, and that know an upper bound on n. In other versions the nodes carry labels and the agents may have a map, or an oracle places coloured **pebbles** on some nodes before the start. We measure **time**: the number of rounds until all agents have stopped.

1. Algorithm: One colour per round. (Czyzowicz et al. 2017a) (Labelled trees, with a map, any number of agents, p. 242) Suppose the graph is a tree (a graph without cycles), its nodes carry labels, and every agent has a map but does not know where the others are. Let Δ be the largest number of edges at a node.

1. Every agent computes the same colouring of the edges with Δ colours in which no two edges at a node share a colour; a tree always has one.
2. In round r the colour is $1 + (r - 1) \bmod \Delta$, so the rounds run through the colours $1, 2, \ldots, \Delta$ and start again.
3. In each round every agent crosses the edge of that round's colour at its node, if there is one, and otherwise waits.
4. After $n\Delta$ rounds every agent stops.

Claude (Anthropic) was used in cataloguing the problems.

Why it works and what it costs. Suppose two agents ended a round on the same node w. An agent that moved came along the only edge of that colour at w, and two agents cannot both have come along it, since they started on different nodes. An agent cannot have waited at w either, since w has an edge of that colour. So no collision ever happens (Lemma 3.1). Each agent's walk goes into every branch of the tree and back. After exactly $n\Delta$ rounds it has visited every node and is back at its start node (Lemma 3.2) (Czyzowicz et al. 2017a). On any graph with labelled nodes and a map, the agents run the same procedure on a spanning tree whose largest degree Δ^* is as small as possible, in $O(n\Delta^*)$ rounds. That is optimal up to a constant factor. Finding that tree is NP-hard, but one of largest degree $\Delta^* + 1$ can be found in polynomial time.

Why it matters. Colouring the edges turns a coordination problem into a timetable that every agent can follow alone, and it is exactly optimal on trees.

2. Lower bound: Congestion: $n\Delta$ rounds on every tree. (Czyzowicz et al. 2017a) (Labelled trees, with a map, any number of agents, p. 242) Put an agent on every node of a tree and look at a node u with Δ edges.

1. In each round at most one agent enters u, otherwise two would collide there.
2. Every agent must visit all Δ branches at u and come back to its start node. In a tree each branch is reached only through u, so every agent enters u at least Δ times.
3. Together the n agents enter u at least $n\Delta$ times, at most once per round.

Why it holds. So some agent is still moving in round $n\Delta$, and the colouring algorithm is exactly optimal on every tree (Theorem 3.2) (Czyzowicz et al. 2017a). An agent does not know how many others there are, so its walk must also work when every node is occupied, and the bound applies to any number of agents. On a star, one node joined to $n - 1$ others, it is $n(n - 1)$ rounds. So agents that explore a tree with labelled nodes without a map in $O(n^2)$ rounds are optimal in the worst case.

Why it matters. The bottleneck is a single busy node, the same congestion argument that bounds routing in networks.

3. Impossibility: With vision of radius 1, two agents collide on a star. (Bhagat and Pelc 2025) (Two agents, anonymous graphs, vision of radius 1, p. 241) Take a star: a centre node joined by edges to leaves, nodes with a single edge. Two identical agents start on two leaves and see only their own node, its neighbours and any agent there (vision of radius 1).

1. From a leaf an agent sees only the centre, so it cannot see the other agent.
2. The only way out of a leaf is the centre. The algorithm must tell each agent to wait some number of rounds after waking up and then step onto the centre; the number may depend on what the agent sees.
3. The adversary wakes the agent with the shorter wait later, by the difference between the two waits.

Why it holds. Both agents then step onto the centre in the same round and collide, whatever the algorithm (Bhagat and Pelc 2025). With vision of radius 2, an agent on a leaf also sees the other leaves and any agent on them. Bhagat and Pelc show that two identical agents that know an upper bound on n then explore every graph with more than two nodes without collisions, at most a factor $\log n$ slower than a single agent. A later preprint reports a flaw in their stopping procedure and gives a faster corrected algorithm (Papin 2026). Neither claim is checked here, so the standard setting remains a draft.

Why it matters. It shows what vision buys: one more edge of sight separates impossible from possible for two agents.

Claude (Anthropic) was used in cataloguing the problems.

What the community started looking at

Czyzowicz, Dereniowski, Gąsieniec, Klasing, Kosowski and Pająk asked how long any number of agents need to explore a graph with labelled nodes without ever sharing a node, every agent returning to its start node; with a map the answer is n times the smallest maximum degree of a spanning tree, up to a constant factor (Czyzowicz et al. 2017a). Bhagat and Pelc took the problem to anonymous graphs with two identical agents, where vision of radius 2 is necessary and sufficient (Bhagat and Pelc 2025). Das, Dhar, Gorain and Mahawar replaced most of that vision by coloured pebbles placed by an oracle (Das et al. 2025a), and Papin claims to remove the logarithmic slowdown of Bhagat and Pelc and to correct their algorithm (Papin 2026).

Relations to other problems

- **Variant of Graph exploration by a single agent.** Every agent explores the graph as in exploration, and no two agents may ever be at the same node. A single-agent exploration procedure is the building block: with vision of radius 2, two agents run any such procedure with a slowdown of at most a factor $\log n$ (Bhagat and Pelc 2025) (a constant factor is claimed by Papin 2026). Lower bounds for exploring by one agent hold here whenever every agent must visit every node.
- **Related: Collaborative exploration.** When it suffices that every node is visited by some agent, the task is collaborative exploration with the added rule that agents never share a node (Das et al. 2025a).

Assumptions specific to this problem

Number of agents

Czyzowicz et al. allow any number of agents, unknown to them (Czyzowicz et al. 2017a). Bhagat and Pelc restrict to two agents and ask how to handle larger teams, for which symmetry breaking and synchronisation by observing moves become hard (Bhagat and Pelc 2025).

Level	Value	Formal definition
0	**any k**	Any number k of agents, $2 \le k \le n$, which they do not know.
1	**two agents**	Exactly two agents.

What must be visited

Czyzowicz et al. and Bhagat and Pelc require every agent to explore (Czyzowicz et al. 2017a; Bhagat and Pelc 2025); Bhagat and Pelc ask whether it helps that each node is visited by one agent only. Das et al. require only that the team visits every node (Das et al. 2025a).

Level	Value	Formal definition
0	**every agent**	Every agent must visit every node and stop (in Czyzowicz et al. 2017a also return to its start node).
1	**the team**	Every node must be visited by at least one agent, and all agents stop.

Knowledge of the graph

Claude (Anthropic) was used in cataloguing the problems.

Czyzowicz et al. compare agents with a map with agents that see only the identifiers of the current node and its neighbours (Czyzowicz et al. 2017a). In anonymous graphs an upper bound on n is necessary: without it no team of agents can explore and stop, whatever their vision (Bhagat and Pelc 2025).

Level	Value	Formal definition
0	**nothing known**	The agents know nothing about G, not even n.
1	**bound on n**	The agents know an upper bound on n (Das et al. 2025a give them n itself).
2	**map**	Every agent knows G and its own start node in it, but not the number or the start nodes of the other agents.

Vision of the other agents

With labelled nodes and a map, agents can precompute routes that never meet, so they need not see each other (Czyzowicz et al. 2017a). In anonymous graphs, Bhagat and Pelc give vision of radius 2 and prove that radius 1 does not suffice; agents then also communicate by watching each other move (Bhagat and Pelc 2025). Das et al. weaken vision to the presence of the other agent one edge away and compensate with pebbles (Das et al. 2025a).

Level	Value	Formal definition
0	**no vision**	An agent never perceives another agent.
1	**presence**	An agent notices that the other agent is on a neighbouring node, but not on which one (Das et al. 2025a).
2	**radius 1**	An agent sees the subgraph induced by the nodes at distance at most 1, with its port numbers and the agents on these nodes.
3	**radius 2**	An agent sees the subgraph induced by the nodes at distance at most 2, with its port numbers and the agents on these nodes (Bhagat and Pelc 2025).

Help from an oracle

Das, Dhar, Gorain and Mahawar let an oracle place coloured pebbles, as in advice-based models of mobile agents, and ask for the fewest colours (Das et al. 2025a).

Level	Value	Formal definition
0	**no help**	Nothing is placed on the nodes.
1	**pebbles**	Before the start an oracle that knows G, the start nodes and the wake-up rounds places at most one pebble on each node; pebbles have colours, and an agent sees the pebble of the node it is at (Das et al. 2025a).

Start

Czyzowicz et al. start all agents together (Czyzowicz et al. 2017a). Bhagat and Pelc let the adversary wake the agents; on the graph with two nodes this alone makes the task impossible (Bhagat and Pelc 2025).

Level	Value	Formal definition
0	**adversarial wake-up**	Each agent starts in a round chosen by the adversary; until then it stays at its start node, where no other agent may enter.
1	**simultaneous start**	All agents start in the same round.

Claude (Anthropic) was used in cataloguing the problems.

Case tree

The root assumes nothing; going down, each edge adds the assumptions written in grey (+) or leaves an axis unassumed. Bold entries are catalogued settings with their status per measure (Time, Colours) and page; italic entries are branch points, named after the axis on which their branches differ. Axis order: Graph dynamics → Edge direction → Node identifiers → Timing → Number of agents → What must be visited → Knowledge of the graph → Vision of the other agents → Help from an oracle → Start → Topology.

No assumptions

+ static graph, + undirected

Node identifiers

anonymous nodes, + synchronous, + two agents

What must be visited

+ the team, + bound on n, + presence

Two agents sensing presence, no pebbles ●● p. 238

+ pebbles

Two agents sensing presence, with coloured pebbles ●● p. 238

+ bipartite

Two agents sensing presence, with coloured pebbles, bipartite graphs ●● p. 239

every agent, + bound on n

Vision of the other agents

+ radius 2

Two agents, anonymous graphs, vision of radius 2 STANDARD SETTING ●● p. 240

+ radius 1

Two agents, anonymous graphs, vision of radius 1 ●● p. 241

every agent, nothing known, + radius 2, + simultaneous start

Anonymous graphs, nothing known ●● p. 241

+ labelled nodes, + synchronous

Knowledge of the graph

+ map, + simultaneous start

Labelled graphs, with a map, any number of agents ●● p. 242

+ tree

Labelled trees, with a map, any number of agents ●● p. 242

nothing known, + simultaneous start

Labelled graphs, no knowledge, any number of agents ●● p. 243

+ tree

Labelled trees, no knowledge, any number of agents ●● p. 243

What is known

The best upper bound against the best lower bound in each catalogued setting, in tree order. The settings follow in detail.

Setting	Time	Colours	p.
Two agents sensing presence, no pebbles	● impossible	● impossible	238

Claude (Anthropic) was used in cataloguing the problems.

Setting	Time	Colours	p.
Two agents sensing presence, with coloured pebbles	● unknown $O(\Delta^D)$ vs —	● tight 3 vs 2	238
Two agents sensing presence, with coloured pebbles, bipartite graphs	● unknown poly(n) vs —	● tight 2 vs 1	239
Two agents, anonymous graphs, vision of radius 2 STANDARD SETTING	● unknown $O(X(n))$ vs —	● tight — vs —	240
Two agents, anonymous graphs, vision of radius 1	● impossible	● impossible	241
Anonymous graphs, nothing known	● impossible	● impossible	241
Labelled graphs, with a map, any number of agents	● tight $O(n\Delta^*)$ vs $\Omega(n\Delta^*)$	● tight — vs —	242
Labelled trees, with a map, any number of agents	● tight $n\Delta$ vs $n\Delta$	● tight — vs —	242
Labelled graphs, no knowledge, any number of agents	● polynomial gap $O(n^5 \log n)$ vs $\Omega(n^2)$	● tight — vs —	243
Labelled trees, no knowledge, any number of agents	● tight $O(n^2)$ vs $\Omega(n^2)$	● tight — vs —	243

GRAPHS

16.11.1 Two agents sensing presence, no pebbles

Adds + the team, + bound on n, + presence VERIFIED

Two identical agents that only sense each other's presence cannot explore even a path of three nodes without help.

Assumed: graph dynamics: static graph; edge direction: undirected; timing: synchronous rounds; number of agents: two agents; what must be visited: every node is visited by some agent; knowledge of the graph: an upper bound on n; vision of the other agents: presence one edge away. **Not assumed:** node identifiers: anonymous nodes; help from an oracle: no help; start: the adversary wakes the agents; topology: arbitrary connected graph.

Measure	Lower bound	Best upper bound	Status
Time	*impossible* (Das et al. 2025a) Even with simultaneous start.	—	● impossible
Colours	none known Impossible without pebbles.	—	● impossible

Results.

- Das et al. (2025a) [lower bound]: time *impossible*. *Bottleneck:* symmetry breaking. *Holds when:* a path of three nodes with the agents on its ends, started together: they act alike, so they either collide in the middle or never visit it (Section 1.2).

Notes. Checked against the full text of (Das et al. 2025a) (arXiv v1).

16.11.2 Two agents sensing presence, with coloured pebbles

Adds + pebbles **to** Two agents sensing presence, no pebbles (p. 238) VERIFIED

With pebbles placed by an oracle, two agents that only sense each other's presence explore any graph without collisions using three colours; one colour does not suffice.

Claude (Anthropic) was used in cataloguing the problems.

Assumed: graph dynamics: static graph; edge direction: undirected; timing: synchronous rounds; number of agents: two agents; what must be visited: every node is visited by some agent; knowledge of the graph: an upper bound on n; vision of the other agents: presence one edge away; help from an oracle: coloured pebbles placed by an oracle. **Not assumed:** node identifiers: anonymous nodes; start: the adversary wakes the agents; topology: arbitrary connected graph.

Statement for this setting. The agents know n.

Measure	Lower bound	Best upper bound	Status
Time	none known Exponential; no lower bound.	$O(\Delta^D)$ (Das et al. 2025a)	● unknown
Colours	2 (Das et al. 2025a) 2 or 3.	3 (Das et al. 2025a)	● tight

Results.

- Das et al. (2025a) [lower bound]: colours 2. *Bottleneck:* symmetry breaking. *Holds when:* three graphs, each a 4-cycle with a chord between the two start nodes and differently numbered ports, which the agents cannot tell apart at the start: with pebbles of one colour, every algorithm collides or leaves a node unvisited in one of them (Lemma 2.1 for pebbles on both start nodes or on neither, Theorem 2.1).
- Das et al. (2025a) [current best]: time $O(\Delta^D)$, colours 3. Introductory and current best (Theorem 2.2): the pebbles break the symmetry between the start nodes and mark safe routes; the time is exponential in the diameter (Section 2.3).

Open questions.

Q1. Do two colours suffice on every graph (Das et al. 2025a)?

Q2. How fast can two agents explore with three colours (Das et al. 2025a)?

Notes. Read in arXiv v1 (2024); theorem numbers refer to it.

16.11.3 Two agents sensing presence, with coloured pebbles, bipartite graphs

Adds + bipartite **to** Two agents sensing presence, with coloured pebbles (p. 238) VERIFIED

On bipartite graphs two colours of pebbles suffice, in polynomial time, and then each agent even visits every node.

Assumed: graph dynamics: static graph; edge direction: undirected; timing: synchronous rounds; number of agents: two agents; what must be visited: every node is visited by some agent; knowledge of the graph: an upper bound on n; vision of the other agents: presence one edge away; help from an oracle: coloured pebbles placed by an oracle; topology: bipartite. **Not assumed:** node identifiers: anonymous nodes; start: the adversary wakes the agents.

Statement for this setting. The agents know n.

Measure	Lower bound	Best upper bound	Status
Time	none known No lower bound.	poly(n) (Das et al. 2025a)	● unknown
Colours	1 (Das et al. 2025a) 1 or 2.	2 (Das et al. 2025a)	● tight

Results.

- Das et al. (2025a) [lower bound]: colours 1. *Bottleneck:* symmetry breaking. *Holds when:* without pebbles the path of three nodes, which is bipartite, cannot be explored (Section 1.2).

Claude (Anthropic) was used in cataloguing the problems.

- Das et al. (2025a) [current best]: time poly(n), colours 2. Introductory and current best (Theorem 2.3): the agents follow a universal exploration sequence for n nodes, and the two colours and the bipartition keep them apart; every node is visited by both agents.

Notes. Read in arXiv v1 (2024).

16.11.4 Two agents, anonymous graphs, vision of radius 2

Adds + radius 2 STANDARD SETTING DRAFT

Two identical agents that see two edges around them and know a bound on n explore any graph with more than two nodes without collisions, at most a factor $\log n$ slower than one agent.

Assumed: graph dynamics: static graph; edge direction: undirected; timing: synchronous rounds; number of agents: two agents; knowledge of the graph: an upper bound on n; vision of the other agents: radius 2. **Not assumed:** node identifiers: anonymous nodes; what must be visited: every agent visits every node; help from an oracle: no help; start: the adversary wakes the agents; topology: arbitrary connected graph.

Statement for this setting. Nodes are anonymous, the adversary wakes the agents, and the graph has at least three nodes: on two nodes the adversary's wake-up alone forces a collision (Bhagat and Pelc 2025).

Measure	Lower bound	Best upper bound	Status
Time	none known No lower bound beyond exploring by one agent, which no collision-free algorithm can beat (Bhagat and Pelc 2025).	$O(X(n))$ (Papin 2026) [preprint]	● unknown
Colours	none known No pebbles.	—	● tight

Results.

- Bhagat and Pelc (2025) [introductory]: time $O(X(n)\log n)$. Theorem 3.1: each agent follows a single-agent exploration walk; when the agents come within distance 2 they synchronise and then take turns, each step of the walk costing $O(\log n)$ rounds of signalling by moves. Any single-agent procedure can be used.
- Papin (2026) [preprint] [current best]: time $O(X(n))$. *Assumes:* vision of radius 2 (the model section says nodes at distance less than 2; see the notes). Theorem 9.1: at most $6X(n) + O(1)$ rounds after the later wake-up, by a priority rule that lets both agents move in the same round. The paper also reports that the stopping procedure of (Bhagat and Pelc 2025) can lead to a collision when the agents finish their walks in different rounds (Section 3). Not checked here.

Open questions.

Q1. Can two agents explore without collisions as fast as one agent explores alone? Asked by (Bhagat and Pelc 2025); (Papin 2026) claims a constant factor.

Q2. For which number of agents below n is collision-free exploration of anonymous graphs feasible, and with which vision radius (Bhagat and Pelc 2025)?

Notes. Draft because the two papers disagree: Papin reports a collision in the stopping procedure of Bhagat and Pelc and gives a corrected algorithm, but his model section describes vision of the nodes at distance less than 2, under which two agents cannot explore a star (Bhagat and Pelc 2025); his algorithm reuses Bhagat and Pelc's synchronisation, which needs radius 2. Neither the reported collision nor the corrected algorithm was checked here. (Bhagat and Pelc 2025) read in arXiv v1 (2024); (Papin 2026) is a preprint.

Claude (Anthropic) was used in cataloguing the problems.

16.11.5 Two agents, anonymous graphs, vision of radius 1

Adds + radius 1 VERIFIED

With vision of radius 1, two identical agents cannot explore a star without colliding.

Assumed: graph dynamics: static graph; edge direction: undirected; timing: synchronous rounds; number of agents: two agents; knowledge of the graph: an upper bound on n; vision of the other agents: radius 1. **Not assumed:** node identifiers: anonymous nodes; what must be visited: every agent visits every node; help from an oracle: no help; start: the adversary wakes the agents; topology: arbitrary connected graph.

Measure	Lower bound	Best upper bound	Status
Time	*impossible* (Bhagat and Pelc 2025) On every star.	—	• impossible
Colours	none known Impossible without pebbles.	—	• impossible

Results.

- Bhagat and Pelc (2025) [lower bound]: time *impossible*. *Bottleneck:* symmetry breaking. *Holds when:* any star, with the agents on two leaves: both see the same thing until one moves, and the adversary delays one agent's wake-up so that both reach the centre in the same round (Section 4).

Notes. Checked against the full text of (Bhagat and Pelc 2025) (arXiv v1).

16.11.6 Anonymous graphs, nothing known

Adds every agent, nothing known, + radius 2, + simultaneous start VERIFIED

In anonymous graphs, agents that know no bound on n cannot explore and stop, whatever their number and their vision.

Assumed: graph dynamics: static graph; edge direction: undirected; timing: synchronous rounds; number of agents: two agents; vision of the other agents: radius 2; start: simultaneous start. **Not assumed:** node identifiers: anonymous nodes; what must be visited: every agent visits every node; knowledge of the graph: nothing known; help from an oracle: no help; topology: arbitrary connected graph.

Measure	Lower bound	Best upper bound	Status
Time	*impossible* (Bhagat and Pelc 2025) For every team and vision.	—	• impossible
Colours	none known Impossible without pebbles.	—	• impossible

Results.

- Bhagat and Pelc (2025) [lower bound]: time *impossible*. *Bottleneck:* symmetry breaking. *Holds when:* a ring whose ports are numbered $0, 1, 0, 1, \ldots$ clockwise, with the agents started together, each one more than the vision radius away from the next: in every round all agents see the same thing and none sees another, so an algorithm that stops after a fixed number of rounds leaves nodes of large enough rings unvisited (Section 4); holds for any number of agents and any vision radius.

Notes. With labelled nodes the task is solvable without knowing n (Czyzowicz et al. 2017a). Checked against the full text of (Bhagat and Pelc 2025) (arXiv v1).

Claude (Anthropic) was used in cataloguing the problems.

16.11.7 Labelled graphs, with a map, any number of agents

Adds + map, + simultaneous start VERIFIED

With a map, collision-free exploration of a graph takes $\Theta(n\Delta^*)$ rounds, where Δ^* is the smallest maximum degree of a spanning tree; computing the exact optimum is NP-hard.

Assumed: graph dynamics: static graph; edge direction: undirected; node identifiers: labelled nodes (an agent can recognise a node it has visited); timing: synchronous rounds; knowledge of the graph: a map with their own position; start: simultaneous start. **Not assumed:** number of agents: any number of agents; what must be visited: every agent visits every node; vision of the other agents: none: agents never perceive each other; help from an oracle: no help; topology: arbitrary connected graph.

Statement for this setting. As in the setting on trees. Δ^* is the smallest maximum degree of a spanning tree of G.

Measure	Lower bound	Best upper bound	Status
Time	$\Omega(n\Delta^*)$ (Czyzowicz et al. 2017a) $\Theta(n\Delta^*)$ for every graph.	$O(n\Delta^*)$ (Czyzowicz et al. 2017a)	• tight
Colours	none known No pebbles.	—	• tight

Results.

- Czyzowicz et al. (2017a) [lower bound]: time $\Omega(n\Delta^*)$. *Bottleneck:* congestion at a node. *Holds when:* n agents, one on every node; the fractional relaxation of the linear program for a spanning tree of minimum degree bounds the number of rounds (Theorem 3.4).
- Czyzowicz et al. (2017a) [current best]: time $O(n\Delta^*)$. Introductory and current best (Proposition 3.3, Theorem 3.4): explore a spanning tree of minimum degree as on trees. That tree is NP-hard to compute; one of degree at most $\Delta^* + 1$, computable in polynomial time, gives $n(\Delta^* + 1)$ rounds. Deciding whether the optimal time is at most a given number is NP-complete (Proposition 3.5).

Notes. Checked against the full text (the 2012 report on HAL).

16.11.8 Labelled trees, with a map, any number of agents

Adds + tree **to** Labelled graphs, with a map, any number of agents (p. 242) VERIFIED

Agents that know the tree and their position explore it without collisions in exactly $n\Delta$ rounds, using a colouring of the edges with Δ colours.

Assumed: graph dynamics: static graph; edge direction: undirected; node identifiers: labelled nodes (an agent can recognise a node it has visited); timing: synchronous rounds; knowledge of the graph: a map with their own position; start: simultaneous start; topology: tree. **Not assumed:** number of agents: any number of agents; what must be visited: every agent visits every node; vision of the other agents: none: agents never perceive each other; help from an oracle: no help.

Statement for this setting. Nodes carry the distinct identifiers $1, \dots, n$, and an agent sees those of its node and of the neighbours. Every agent must return to its start node.

Measure	Lower bound	Best upper bound	Status
Time	$n\Delta$ (Czyzowicz et al. 2017a) Exactly $n\Delta$ rounds.	$n\Delta$ (Czyzowicz et al. 2017a)	• tight
Colours	none known No pebbles.	—	• tight

Claude (Anthropic) was used in cataloguing the problems.

Results.

- Czyzowicz et al. (2017a) [lower bound]: time $n\Delta$. *Bottleneck:* congestion at a node. *Holds when:* n agents, one on every node; a node u of degree Δ: in every round at most one agent enters u, and every agent enters u at least Δ times because its route is closed and G is a tree (Theorem 3.2). Theorem 3.2: the bound holds for every tree.
- Czyzowicz et al. (2017a) [current best]: time $n\Delta$. Introductory and current best (Proposition 3.1, Theorem 3.2): from a colouring of the edges with Δ colours, in each round the agents cross the edges of one colour class, which form a matching, so no two agents meet.

Notes. Up to constant factors the bounds also hold if agents need not return to their start nodes (Czyzowicz et al. 2017a). Checked against the full text (the 2012 report on HAL; theorem numbers refer to it).

16.11.9 Labelled graphs, no knowledge, any number of agents

Adds nothing known, + simultaneous start VERIFIED

Without a map or a bound on n, agents on a graph with labelled nodes explore without collisions in $O(n^5 \log n)$ rounds, against $\Omega(n^2)$.

Assumed: graph dynamics: static graph; edge direction: undirected; node identifiers: labelled nodes (an agent can recognise a node it has visited); timing: synchronous rounds; start: simultaneous start. **Not assumed:** number of agents: any number of agents; what must be visited: every agent visits every node; knowledge of the graph: nothing known; vision of the other agents: none: agents never perceive each other; help from an oracle: no help; topology: arbitrary connected graph.

Statement for this setting. As in the setting on trees, on any graph.

Measure	Lower bound	Best upper bound	Status
Time	$\Omega(n^2)$ (Czyzowicz et al. 2017a) $O(n^5 \log n)$ against $\Omega(n^2)$.	$O(n^5 \log n)$ (Czyzowicz et al. 2017a)	• polynomial gap
Colours	none known No pebbles.	—	• tight

Results.

- Czyzowicz et al. (2017a) [lower bound]: time $\Omega(n^2)$. *Bottleneck:* congestion at a node. *Holds when:* a star, as on trees (Theorem 3.2).
- Czyzowicz et al. (2017a) [current best]: time $O(n^5 \log n)$. Introductory and current best (Theorem 4.2): the tree algorithm with universal exploration sequences in place of the tree traversal.

Open questions.

Q1. Close $O(n^5 \log n)$ against $\Omega(n^2)$ for agents without a map on graphs with labelled nodes.

Notes. Checked against the full text (the 2012 report on HAL).

16.11.10 Labelled trees, no knowledge, any number of agents

Adds + tree **to** Labelled graphs, no knowledge, any number of agents (p. 243) VERIFIED

Without a map or a bound on n, agents on a tree with labelled nodes explore without collisions in $O(n^2)$ rounds, which is optimal in the worst case.

Assumed: graph dynamics: static graph; edge direction: undirected; node identifiers: labelled nodes (an agent can recognise a node it has visited); timing: synchronous rounds; start: simultaneous start; topology: tree. **Not assumed:** number of agents:

Claude (Anthropic) was used in cataloguing the problems.

any number of agents; what must be visited: every agent visits every node; knowledge of the graph: nothing known; vision of the other agents: none: agents never perceive each other; help from an oracle: no help.

Statement for this setting. Nodes carry the distinct identifiers $1, \dots, n$, and an agent sees those of its node and of the neighbours; the agents are not told n or anything else about the tree. Every agent must return to its start node.

Measure	Lower bound	Best upper bound	Status
Time	$\Omega(n^2)$ (Czyzowicz et al. 2017a) $\Theta(n^2)$ in the worst case over trees with n nodes.	$O(n^2)$ (Czyzowicz et al. 2017a)	• tight
Colours	none known No pebbles.	—	• tight

Results.

- Czyzowicz et al. (2017a) [lower bound]: time $\Omega(n^2)$. *Bottleneck:* congestion at a node. *Holds when:* a star: even with a map, $n\Delta = n(n-1)$ rounds are needed (Theorem 3.2).
- Czyzowicz et al. (2017a) [current best]: time $O(n^2)$. Introductory and current best (Theorem 4.1): an edge is coloured by the sum of the identifiers of its ends, which every agent sees; in phases that double a bound on the identifiers, the agents cross the edges of one colour at a time, and at the end every agent retraces its moves.

Notes. The lower bound specialises Theorem 3.2 to a star. Checked against the full text (the 2012 report on HAL).



16.12 Collaborative exploration

k agents start together at the root of an unknown graph and must visit all its nodes; how much faster than one agent can a team be?

First posed in 2004 (Fraigniaud et al. 2006a). Collective tree exploration was posed at LATIN 2004 (journal version 2006).

Problem statement and measures

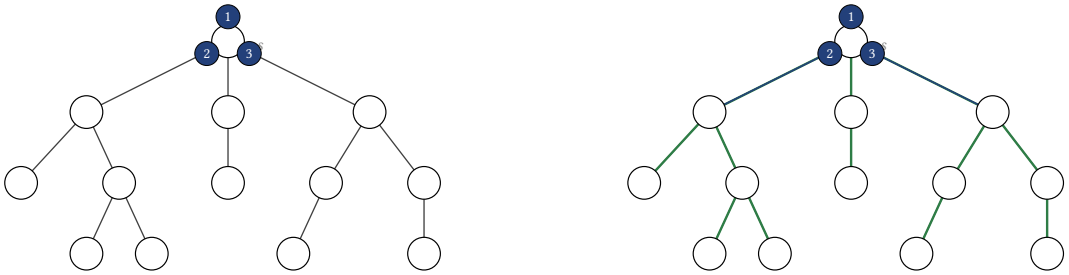
A team of k distinguishable agents starts at a node s (the root) of an unknown connected graph with n nodes; D is the largest distance from s. In synchronous rounds every agent traverses at most one edge. When an agent reaches a node it sees the incident edges. The team must **visit every node** (some papers also require all agents to return to s; the bounds coincide asymptotically).

Time is the number of rounds (in the asynchronous model: the total number of moves). The offline optimum is $\Theta(n/k + D)$, so an algorithm that explores every tree in $\mathrm{CR}(k) \cdot (n/k + D)$ rounds is called $\mathrm{CR}(k)$-competitive; a single depth-first search is $O(k)$-competitive. **Memory** is bits per agent; most of this literature lets it be unbounded.

Unless a variant says otherwise, nodes are **labelled** (an agent recognises a node it has seen), which is the standard assumption of collective tree exploration; on trees it does not matter for lower bounds (Disser et al. 2020). With anonymous nodes and $k = n$, the problem is solved by rooted dispersion.

Measures: **Time**, rounds until every node is visited (asynchronous: total moves); **Memory**, bits per agent.

Claude (Anthropic) was used in cataloguing the problems.



(a) Start: k = 3 agents at the root s of an unknown tree

(b) Final: every node visited, all agents back at s after 10 rounds

Figure 16.12: Collective tree exploration (Fraigniaud et al. 2006a), here by Breadth-First Depth-Next (Cosson et al. 2023): $k = 3$ agents start at the root s of an unknown tree and share one map. An agent at the root is sent to the shallowest node with an unexplored edge (its anchor); from there it walks down unexplored edges that no other agent takes in that round, and steps back towards the root when there is none. One frame per round; the time is compared with the offline optimum $\Theta(n/k + D)$.

A first look

Imagine k rescuers entering a cave system through one entrance; they want every chamber checked as soon as possible. We model the cave as a graph: chambers are **nodes**, tunnels between two chambers are **edges**, and each rescuer is an **agent**. The cave is a **tree** (a graph without cycles, so there is exactly one route between any two nodes) with n nodes; the entrance is its **root** s, and the **depth** D is the largest distance from s. All k agents start at s; time passes in **rounds**, and in each round every agent may walk along one edge. Nodes are **labelled** (an agent recognises a node it has seen before), and an agent at a node sees the edges leaving it. In the basic version all agents share everything they learn (**global communication**), so they act on one **map**: the part of the tree seen so far. The team must visit every node; the time is the number of rounds, and an algorithm is judged by how close it gets to the best possible time of about $n/k + D$ rounds.

1. Algorithm: Breadth-First Depth-Next. (Cosson et al. 2023) (Trees, complete communication, p. 252) The algorithm BFDN of (Cosson et al. 2023). Call an edge **unexplored** if it has been seen but no agent has walked it yet (the paper calls it *dangling*).

1. When an agent is at the root, give it an **anchor**: a node of the map with an unexplored edge, preferring nodes closest to the root and, among those, the ones with the fewest agents already anchored there.
2. The agent walks along the map to its anchor.
3. From there it does a **depth-first search**: if an unexplored edge leaves its node and no other agent has chosen it this round, it takes it; otherwise it steps one edge back towards the root.
4. Back at the root it gets a new anchor. The team stops when all agents are at the root and no unexplored edge is left.

Why it works and what it costs. Each agent mostly does what a single agent exploring alone does (walk down unexplored edges, back up when stuck), and the anchors spread the team over the shallowest unfinished parts of the tree. (Cosson et al. 2023) prove a time of $\frac{2n}{k} + D^2(\min\{\log \Delta, \log k\} + 2)$ rounds, where Δ is the maximum degree; the $2n/k$ term is what k agents need just to walk every edge twice. The best known algorithm today takes $\frac{2n}{k} + O(D \log^2 k)$ rounds (Cosson and Massoulié 2026).

Claude (Anthropic) was used in cataloguing the problems.

Why it matters. Its analysis through a simple game against an adversary opened the series of papers that brought the overhead from roughly a factor $k/\log k$ down to polylogarithmic in k.

2. Lower bound: Even knowing the tree: n/k + D. (Cosson et al. 2023) (Trees, complete communication, p. 252) Suppose the agents knew the whole tree in advance. Two things still take time.

1. Some agent must reach the deepest node, which is D edges from the root: at least D rounds.
2. Each of the $n-1$ nodes other than the root must be entered for the first time by some agent, and in one round k agents enter at most k nodes: at least $(n-1)/k$ rounds.

Why it holds. So no algorithm beats about $\max\{n/k, D\}$, which is at least half of $n/k + D$. Knowing the tree, this is also achievable up to a factor 2: cut a depth-first tour of the tree (length $2(n-1)$) into k pieces of length $2(n-1)/k$ and send one agent to each piece, for $2(n/k + D)$ rounds in total, as recalled in (Cosson et al. 2023). An online algorithm (one that discovers the tree as it goes) that is always within a factor $\mathrm{CR}(k)$ of $n/k + D$ is called $\mathrm{CR}(k)$-**competitive**; a single depth-first search by one agent is $O(k)$-competitive.

Why it matters. This simple yardstick is what every result on collaborative exploration is measured against.

3. Lower bound: An adversary that grows the tree. (Disser et al. 2020) (Trees, complete communication, p. 252) When the tree is not known in advance, it is revealed only where agents stand, so one can imagine an **adversary** (an opponent who builds the worst input) drawing the tree during the run. When the agents reach a node, the adversary decides how many children it has. Whatever way the agents split up, the adversary can make the long branches continue where few agents went and end the branches that received many agents. The full proof that this costs a factor $\Omega(\log k/\log\log k)$ is longer (Disser et al. 2020).

Why it holds. Every deterministic algorithm takes $\Omega\big(\frac{\log k}{\log\log k}(\frac{n}{k} + D)\big)$ rounds on some tree, even with global communication and unlimited memory (Disser et al. 2020); it was first shown by (Dynia et al. 2007) for $k < \sqrt{n}$. The best algorithm is $O(\log^2 k)$-competitive (Cosson and Massoulié 2026), so the remaining gap is between $\log k/\log\log k$ and $\log^2 k$.

Why it matters. It refuted the hope of a constant competitive ratio and shows that not knowing where the work is costs something no amount of communication can repair.

What the community started looking at

Collaborative exploration asked a different question from the single-agent line: not how little memory, but **how much faster** k agents can be than one. Fraigniaud, Gąsieniec, Kowalski and Pelc measured an algorithm by its competitive ratio against the offline optimum $\Theta(n/k + D)$ and showed that communication matters: without it, k agents gain nothing (Fraigniaud et al. 2006a). The following twenty years were spent narrowing the ratio from $O(k/\log k)$ towards the $\Omega(\log k/\log\log k)$ lower bound (Dynia et al. 2007; Disser et al. 2020; Cosson and Massoulié 2024), and on what very large teams can do (Dereniowski et al. 2015a).

Further reading

Papers on this problem that are not yet placed in a setting.

- Jana et al. (2025a). The offline problem behind connected collective exploration: k agents on distinct nodes that always induce a connected subgraph must visit every node of a known graph in the fewest steps. It is NP-hard; the paper gives optimal polynomial algorithms on trees for one agent and for a restricted version with k agents

Claude (Anthropic) was used in cataloguing the problems.

(Theorems 1 and 3), a $(3 - \frac{1}{21})$-approximation for two agents on arbitrary graphs, 2 without 4-cycles (Theorem 4), and a $2(1 + \ln k)$-approximation for constant k (Theorems 5 and 6). Centralised. Read (arXiv v1).

- Bampas et al. (2018). Online exploration of an unknown tree by k agents that start together and may each cross at most B edges, maximising the number of nodes visited; communication at any distance. A 3-competitive algorithm, and no algorithm is better than 2.17-competitive. Abstract read (arXiv v1, a preprint).
- Czyzowicz et al. (2021c). Exploration of a known edge-weighted graph by agents with individual budgets that may share energy when they meet: $O(n + k)$ on a path, $O(n + \ell k^2)$ on a tree with ℓ leaves, NP-hard on 3-regular graphs; total energy twice the total edge weight always suffices, which is asymptotically optimal. Abstract read (arXiv v1).
- Bramas et al. (2024b). Two energy-sharing agents explore a tree when one of them may crash: necessary and sufficient conditions on the energy, synchronous and asynchronous. Abstract read.
- Czyzowicz et al. (2017b). Exploration of a weighted graph whose nodes have deadlines by robots of which an unknown subset may crash: the optimal exploration time and the largest number of faulty robots tolerated. Abstract read.
- Gupta et al. (2023a). Collective graph exploration from a common root with k robots and budget B (all edges covered by closed walks of length at most B): NP-hard even on trees, fixed-parameter tractable by vertex cover. Abstract read.
- Osula (2019). Team exploration of rings and trees minimising the total distance travelled plus the cost of invoking agents, offline and online. Abstract read.
- Sinay et al. (2017). Multi-robot coverage of perfect N-ary trees while the team stays connected, with speedup guarantees (IJCAI). Abstract read.
- Clements et al. (2019). Online exploration of an unknown orthogonal region with rectangular holes, discretised into n grid points (their N), by k agents that talk only when adjacent: they alternate between following the walls and sweeping rooms, for $O(n)$ moves in total and $O(n/k)$ per agent (Theorem 2); when the rooms have no holes and are large, the time is $\Theta(n/k)$, which is optimal (Corollary 5). Read (model and theorems).

Relations to other problems

- **Derived problem: Dispersion** (derived from). With $k = n$ agents, a dispersed configuration has an agent on every node, so any dispersion algorithm solves n-agent collaborative exploration under the same assumptions (Augustine and Moses Jr. 2018). Consequently every lower bound for n-agent exploration is a lower bound for dispersion (this is how the $\Omega(D^2)$ tree bound of Disser et al. 2020 enters), and every dispersion algorithm is an exploration algorithm for anonymous graphs with local communication. For $k < n$ dispersion asks for less: only k nodes must be occupied.
- **Derived problem: Black hole search** (variant of). Locating the black hole means exploring every other node safely, so black hole search is exploration of a dangerous graph by a team, where agents can be lost.
- **Derived problem: Exploration of dynamic graphs by teams** (variant of). When an adversary removes edges, a single agent can be blocked forever; the question shifts from speed to **how many agents** are needed to visit every node.
- **Related: Online exploration of weighted graphs.** Collective tree exploration also measures algorithms by their competitive ratio against the offline optimum, but for k agents on unweighted trees and for time; here one agent explores a weighted graph and its cost is compared with a shortest tour.
- **Related: Collision-free exploration.** When it suffices that every node is visited by some agent, the task is collaborative exploration with the added rule that agents never share a node (Das et al. 2025a).

Claude (Anthropic) was used in cataloguing the problems.

- **Related: Delivery by energy-constrained agents.** Energy-constrained agents also explore graphs together: with budgets in unknown trees (Bampas et al. 2018), and with energy sharing in known graphs (Czyzowicz et al. 2021c); both are listed there as further reading.

Assumptions specific to this problem

Team size

Fraigniaud, Gąsieniec, Kowalski and Pelc posed collective tree exploration for any number k of agents and measured algorithms by their competitive ratio against the offline optimum (Fraigniaud et al. 2006a). Two team sizes then turned out to have sharper answers than arbitrary k. For large teams, Dereniowski, Disser, Kosowski, Pająk and Uznański explore in $O(D)$ rounds with $k = Dn^{1+\varepsilon}$ agents, a constant competitive ratio, and Disser, Mousset, Noever, Steger and Škorić showed that $k = Dn^{1+o(1)}$ agents do not suffice for that, so the threshold is essentially tight (Dereniowski et al. 2015a; Disser et al. 2020). For $k = n$, Disser et al. gave a simple algorithm taking D^2 rounds on every tree and showed that trees of arbitrarily large height need $\Omega(D^2)$ rounds (Disser et al. 2020). This team size matters beyond exploration because, as Augustine and Moses Jr. observed when they introduced dispersion, "any solution to a problem of dispersion acts as a solution to n robot exploration under the same conditions" (Augustine and Moses Jr. 2018).

Level	Value	Formal definition
0	**any** k	Any number $k \geq 2$ of agents.
1	$k = n$	Exactly $k = n$ agents.
1	$k \geq Dn^{1+\varepsilon}$	$k \geq D\,n^{1+\varepsilon}$ agents for a constant $\varepsilon > 0$ (written $k = Dn^{\gamma}$, $\gamma > 1$).

Connectivity of the team

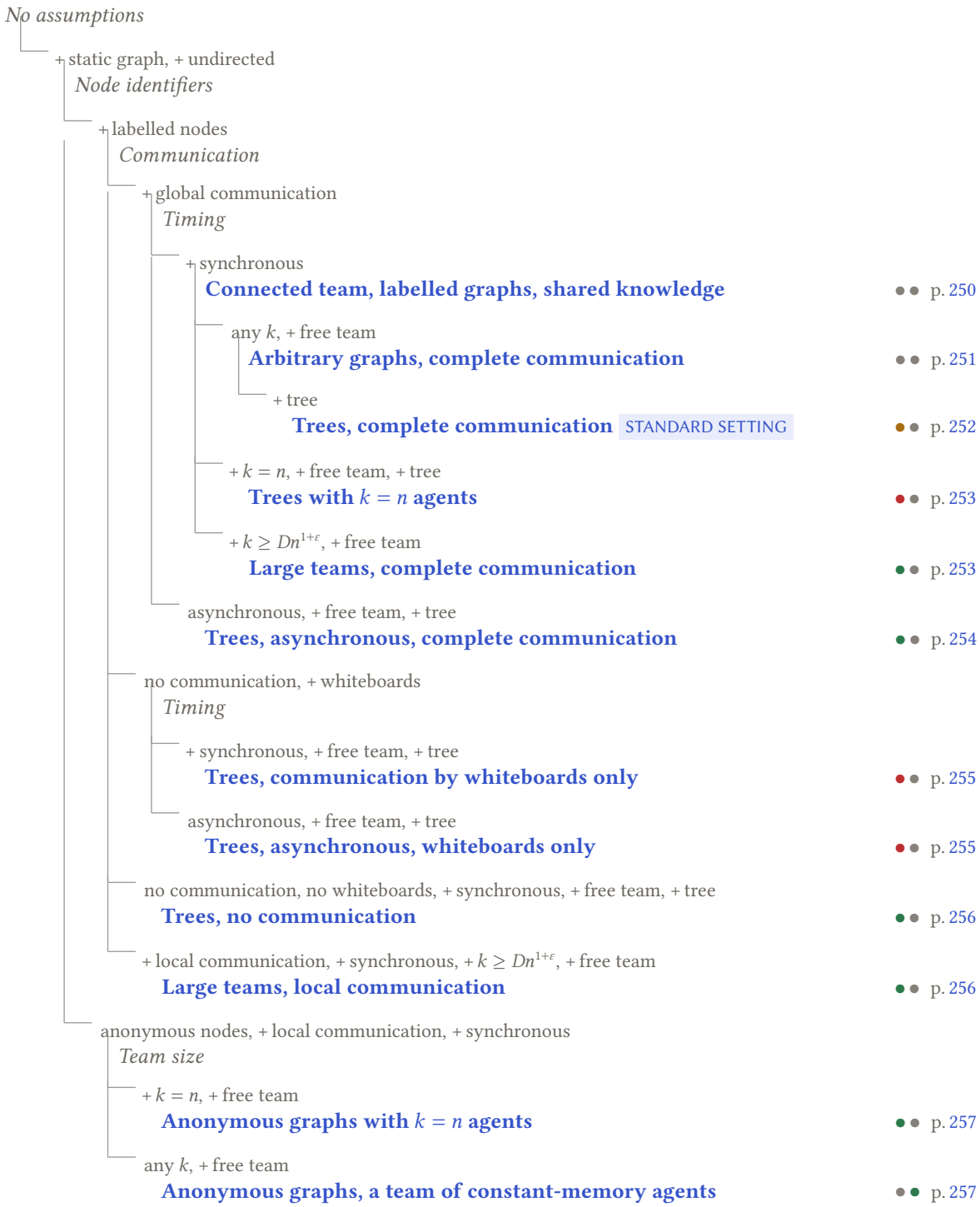
Collective exploration lets the agents spread out freely, with global communication as the usual compensation (Fraigniaud et al. 2006a). Mutzari, Aumann and Kraus require them to occupy a connected set of nodes in every round, as rescue teams and swarms that must stay within reach of each other do, and replace global communication by messages passed along the team (Mutzari et al. 2026); Jana, Italiano and Mandal study the offline version, a shortest walk of a connected team through a known graph (Jana et al. 2025a). Because the benchmark changes with the rule, competitive ratios for free teams do not carry over to connected teams.

Level	Value	Formal definition
0	**connected team**	In every round the nodes occupied by the agents induce a connected subgraph. An algorithm is then compared with the best traversal of the known graph by a team that obeys the same rule (Mutzari et al. 2026).
1	**free team**	Agents may be anywhere in the graph.

Claude (Anthropic) was used in cataloguing the problems.

Case tree

The root assumes nothing; going down, each edge adds the assumptions written in grey (+) or leaves an axis unassumed. Bold entries are catalogued settings with their status per measure (Time, Memory) and page; italic entries are branch points, named after the axis on which their branches differ. Axis order: Graph dynamics → Edge direction → Node identifiers → Randomness → Communication → Node memory → Timing → Team size → Connectivity of the team → Topology.



Claude (Anthropic) was used in cataloguing the problems.

What is known

The best upper bound against the best lower bound in each catalogued setting, in tree order. The settings follow in detail.

Setting	Time	Memory	p.
Connected team, labelled graphs, shared knowledge	● unknown $O(\min(\sqrt{D}, k^{1/3})\log k)\cdot$ optimal vs —	● unknown — vs —	250
Arbitrary graphs, complete communication	● unknown $\frac{2n}{k} + D^2(\min\{\log\Delta, \log k\}+2)$ vs —	● unknown — vs —	251
Trees, complete communication STANDARD SETTING	● near-optimal $\frac{2n}{k} + O(D\log^2 k)$ vs $\Omega(\frac{\log k}{\log\log k}(\frac{n}{k}+D))$	● unknown — vs —	252
Trees with $k = n$ agents	● polynomial gap D^2 vs $D^2/3$	● unknown — vs —	253
Large teams, complete communication	● tight $D(1+\frac{1}{\gamma-1}+o(1))$ vs $D(1+\frac{1}{\gamma}-o(1))$	● unknown — vs —	253
Trees, asynchronous, complete communication	● tight $2n + O(kD\log^2 k)$ vs $\Omega(\log^2 k)\cdot(n+kD)$	● unknown — vs —	254
Trees, communication by whiteboards only	● polynomial gap $O(D + n/\log k)$ vs —	● unknown — vs —	255
Trees, asynchronous, whiteboards only	● polynomial gap $2n + O(k^2 2^k D)$ vs —	● unknown — vs —	255
Trees, no communication	● tight $O(k)\cdot$ optimal vs $\Omega(k)\cdot$ optimal	● unknown — vs —	256
Large teams, local communication	● tight $D(1+\frac{2}{\gamma-1}+o(1))$ vs $D(1+\frac{2}{c}-o(1))$	● unknown — vs —	256
Anonymous graphs with $k = n$ agents	● tight $O(n)$ vs $\Omega(n)$	● unknown $O(\log n)$ vs —	257
Anonymous graphs, a team of constant-memory agents	● unknown $n^{O(1)}$ vs —	● tight $O(1)$ bits each, $O(\log\log n)$ agents vs $\Omega(\log\log n)$ agents	257

16.12.1 Connected team, labelled graphs, shared knowledge

Adds + synchronous VERIFIED

A team that must stay connected explores an unknown graph with $k \geq n$ agents within a factor $O(\min(\sqrt{D}, k^{1/3})\log k)$ of the best connected traversal of the known graph.

Assumed: graph dynamics: static graph; edge direction: undirected; node identifiers: labelled nodes (an agent can recognise a node it has visited); communication: global (any agent can read any other agent's state); timing: synchronous rounds. **Not assumed:** randomness: deterministic; node memory: no whiteboards; team size: any number k of agents; connectivity of the team: the agents must stay connected; topology: arbitrary connected graph.

Statement for this setting. In every round the occupied nodes induce a connected subgraph; several agents may share a node. When an agent first visits a node, the whole team learns its degree and its

unexplored edges. Time is compared with the fewest rounds in which a connected team can visit every node of the known graph.

Measure	Lower bound	Best upper bound	Status
Time	none known	$O(\min(\sqrt{D}, k^{1/3})\log k)\cdot$ optimal (Mutzari et al. 2026)	● unknown
	No lower bound for connected teams; bounds for free teams do not apply, since the optimum differs.		
Memory	none known Memory is not bounded.	—	● unknown

Results.

- Mutzari et al. (2026) [current best]: time $O(\min(\sqrt{D}, k^{1/3})\log k)\cdot$ optimal. *Assumes:* $k \geq n$; a centralised algorithm; the authors note that it can be run with messages passed along the connected team. Introductory and current best. Corollary 3 gives $O(\max(\sqrt{D}, \log k)\log k)$ by breadth-first exploration in growing layers that returns to the root after each layer; Theorem 4 gives $O(k^{1/3}\log k)$ by running that algorithm up to depth $k^{2/3}$ and, if the graph is deeper, letting the whole team perform one depth-first search as a unit, which takes $O(n) = O(k)$ rounds while the optimum needs at least the depth.

Open questions.

Q1. Is there an algorithm with sublinear overhead for $n > k$ agents? The paper gives experiments only and calls a bound an important milestone (Mutzari et al. 2026).

Notes. Mutzari, Aumann and Kraus compare their online algorithm experimentally with an offline coverage algorithm on random graphs and hotel floor plans. The offline problem is NP-hard; Jana, Italiano and Mandal approximate it for agents on distinct nodes (Jana et al. 2025a). Checked against the full text of (Mutzari et al. 2026) (IJCAI 2026).

16.12.2 Arbitrary graphs, complete communication

Adds any k, + free team **to** Connected team, labelled graphs, shared knowledge (p. 250) VERIFIED

On graphs with cycles it is not even known whether any algorithm beats depth-first search by more than a constant.

Assumed: graph dynamics: static graph; edge direction: undirected; node identifiers: labelled nodes (an agent can recognise a node it has visited); communication: global (any agent can read any other agent's state); timing: synchronous rounds; connectivity of the team: the agents may spread out. **Not assumed:** randomness: deterministic; node memory: no whiteboards; team size: any number k of agents; topology: arbitrary connected graph.

Measure	Lower bound	Best upper bound	Status
Time	none known	$\frac{2n}{k} + D^2(\min\{\log\Delta, \log k\} + 2)$ (Cosson et al. 2023)	● unknown
	No $o(k)$-competitive algorithm is known without extra knowledge; lower bounds are inherited from trees.		
Memory	none known Memory is not bounded in these papers.	—	● unknown

Results.

- Cosson et al. (2023) [current best]: time $\frac{2n}{k} + D^2(\min\{\log\Delta, \log k\} + 2)$. *Assumes:* agents know their distance to the root. Proposition 7; n counts edges here.

Open questions.

Claude (Anthropic) was used in cataloguing the problems.

Q1. Is there an $o(k)$-competitive collaborative exploration algorithm for arbitrary graphs? (Stated as open in Cosson and Massoulié 2026.)

16.12.3 Trees, complete communication

Adds + tree **to** Arbitrary graphs, complete communication (p. 251) STANDARD SETTING VERIFIED

Collective tree exploration as posed by Fraigniaud, Gąsieniec, Kowalski and Pelc: the competitive ratio is now between $\Omega(\log k/\log\log k)$ and $O(\log^2 k)$.

Assumed: graph dynamics: static graph; edge direction: undirected; node identifiers: labelled nodes (an agent can recognise a node it has visited); communication: global (any agent can read any other agent's state); timing: synchronous rounds; connectivity of the team: the agents may spread out; topology: tree. **Not assumed:** randomness: deterministic; node memory: no whiteboards; team size: any number k of agents.

Measure	Lower bound	Best upper bound	Status
Time	$\Omega\big(\frac{\log k}{\log\log k}\,(\frac{n}{k}+D)\big)$ (Disser et al. 2020)	$\frac{2n}{k}+O(D\log^2 k)$ (Cosson and Massoulié 2026) [preprint]	• near-optimal
	$O(\log^2 k)$ versus $\Omega(\log k/\log\log k)$ competitive: about a $\log k\log\log k$ factor (upper bound from a preprint; the peer-reviewed gap is $O(\sqrt{k})$ versus the same lower bound).		
Memory	none known Memory is not bounded in these papers.	—	• unknown

Results.

- Disser et al. (2020) [lower bound]: time $\Omega\big(\frac{\log k}{\log\log k}\,(\frac{n}{k}+D)\big)$. *Bottleneck:* online tree revelation. *Holds when:* deterministic; holds with global communication and unbounded memory; for $k\le n\log^\kappa n$; first shown by (Dynia et al. 2007) for $k<\sqrt{n}$; the hard instances are trees. Before that, a ratio of at least $2-1/k$ on trees of $n=xk^2+1$ nodes, so for $k<\sqrt{n}$ (Theorem 3.2 of Fraigniaud et al. 2006a, proved with an adversary that answers each choice of an unexplored edge); (Dereniowski et al. 2015a) state it for deterministic and randomized algorithms. (Dynia et al. 2007) prove their bound for deterministic algorithms on trees of $O(tk)$ nodes with $t>k$ (Theorem 1).
- Fraigniaud et al. (2006a) [introductory]: time $O(D+n/\log k)$. Works already with write-read communication at nodes; $O(k/\log k)$-competitive. The analysis is tight on a tree of (Higashikawa et al. 2014).
- Cosson et al. (2023) [also]: time $\frac{2n}{k}+D^2(\min\{\log\Delta,\log k\}+2)$. Breadth-first depth-next; optimal when $D=o(\sqrt{n})$.
- Cosson (2024b) [also]: time $\frac{2n}{k}+O(k^{\log_2 k-1}D)$. First ratio below $k/\log k$: $O(k/\exp(\sqrt{\ln 2\ln k}))$.
- Cosson and Massoulié (2024) [also]: time $\frac{2n}{k}+O(kD)$. First polynomial improvement: $O(\sqrt{k})$-competitive by using $\sqrt{k}$ of the agents. Current best among peer-reviewed results.
- Cosson and Massoulié (2026) [preprint] [current best]: time $\frac{2n}{k}+O(D\log^2 k)$. $O(\log^2 k)$-competitive. A September 2026 preprint, not yet peer-reviewed.

Open questions.

Q1. Determine the competitive ratio of synchronous collective tree exploration: between $\Omega(\log k/\log\log k)$ and $O(\log^2 k)$.

Q2. Do randomized algorithms help? A randomized lower bound of $\Omega(\sqrt{\log k}/\log\log k)$ is attributed to Ortolf and Schindelhauer (SPAA 2012), not checked here. Cosson and Massoulié conjecture they do not (Cosson and Massoulié 2025).

Notes. Fraigniaud et al. listed an algorithm with constant ratio as their first open problem (Fraigniaud et al. 2006a), and according to (Disser et al. 2020) suggested in their initial version that one exists;

Claude (Anthropic) was used in cataloguing the problems.

(Dynia et al. 2007) showed that no deterministic algorithm achieves it. Greedy algorithms cannot beat $\Omega(k/\log k)$ (Higashikawa et al. 2014), which is why progress needed new ideas: tree-mining games (Cosson 2024b) and potential functions (Cosson and Massoulié 2024).

16.12.4 Trees with $k = n$ agents

Adds + $k = n$, + free team, + tree **to** Connected team, labelled graphs, shared knowledge (p. 250) VERIFIED

With as many agents as nodes, D^2 rounds suffice, and some trees need $D^2/3$; this is the bound dispersion inherits.

Assumed: graph dynamics: static graph; edge direction: undirected; node identifiers: labelled nodes (an agent can recognise a node it has visited); communication: global (any agent can read any other agent's state); timing: synchronous rounds; team size: $k = n$ agents; connectivity of the team: the agents may spread out; topology: tree. **Not assumed:** randomness: deterministic; node memory: no whiteboards.

Measure	Lower bound	Best upper bound	Status
Time	$D^2/3$ (Disser et al. 2020); $\Omega(D)$ (Disser et al. 2020) D^2 is matched only for trees of depth about $\sqrt{\log n}$; for other depths the gap is a factor D.	D^2 (Disser et al. 2020)	• polynomial gap
Memory	none known Memory is not bounded in these papers.	—	• unknown

Results.

- Disser et al. (2020) [lower bound]: time $D^2/3$. *Bottleneck:* online tree revelation. *Holds when:* deterministic, global communication; **existential in** D: the hard tree has depth $D = \lceil\sqrt{\log n}\rceil$ (Theorem 2.5); not shown for larger heights.
- Disser et al. (2020) [lower bound]: time $\Omega(D)$. *Bottleneck:* distance. *Holds when:* the root must reach the deepest node.
- Disser et al. (2020) [current best]: time D^2. Introductory and current best: phase-wise breadth-first search, D phases of D rounds.

Open questions.

Q1. For $k = n$ and trees of depth D well above $\sqrt{\log n}$ (e.g. $D = \sqrt{n}$), is $\Omega(D^2)$ still necessary?

Notes. Dispersion papers cite this bound as "$\Omega(D^2)$ for n agents on trees" (Augustine and Moses Jr. 2018; Kshemkalyani et al. 2022); the theorem is existential in D, see the overview *From exploration to dispersion*.

16.12.5 Large teams, complete communication

Adds + $k \geq Dn^{1+\varepsilon}$, + free team **to** Connected team, labelled graphs, shared knowledge (p. 250) VERIFIED

With $k = Dn^{\gamma}$ agents ($\gamma > 1$), exploration takes $D(1 + \frac{1}{\gamma-1} + o(1))$ rounds, almost the trivial D.

Assumed: graph dynamics: static graph; edge direction: undirected; node identifiers: labelled nodes (an agent can recognise a node it has visited); communication: global (any agent can read any other agent's state); timing: synchronous rounds; team size: large teams, $k \geq Dn^{1+\varepsilon}$; connectivity of the team: the agents may spread out. **Not assumed:** randomness: deterministic; node memory: no whiteboards; topology: arbitrary connected graph.

Claude (Anthropic) was used in cataloguing the problems.

Measure	Lower bound	Best upper bound	Status
Time	$D(1+\frac{1}{\gamma}-o(1))$ (Dereniowski et al. 2015a) $\Theta(D)$; the constants differ.	$D(1+\frac{1}{\gamma-1}+o(1))$ (Dereniowski et al. 2015a)	• tight
Memory	none known Memory is not bounded in these papers.	—	• unknown

Results.

- Dereniowski et al. (2015a) [lower bound]: time $D(1+\frac{1}{\gamma}-o(1))$. *Bottleneck:* distance. *Holds when:* $k = Dn^{\gamma}$ agents; $D = n^{o(1)}$; trees.
- Dereniowski et al. (2015a) [current best]: time $D(1+\frac{1}{\gamma-1}+o(1))$. *Assumes:* n known. Introductory and current best; trees and general graphs.

Open questions.

Q1. Which team size is necessary and sufficient for $O(D)$-round exploration of graphs of larger diameter? (Stated in Dereniowski et al. 2015a.)

Notes. For $k \le Dn^{1+\varepsilon}$, some trees need $D/(5\varepsilon)$ rounds (Disser et al. 2020), so the team size of (Dereniowski et al. 2015a) is essentially needed for a constant ratio.

16.12.6 Trees, asynchronous, complete communication

Adds asynchronous, + free team, + tree VERIFIED

An adversary chooses which agent moves; the cost is the total number of moves, and the optimal regret is $\Theta(\log^2 k)$ per unit of depth.

Assumed: graph dynamics: static graph; edge direction: undirected; node identifiers: labelled nodes (an agent can recognise a node it has visited); communication: global (any agent can read any other agent's state); connectivity of the team: the agents may spread out; topology: tree. **Not assumed:** randomness: deterministic; node memory: no whiteboards; timing: asynchronous; team size: any number k of agents.

Measure	Lower bound	Best upper bound	Status
Time	$\Omega(\log^2 k)\cdot(n+kD)$ (Cosson and Massoulié 2025) [preprint] Regret $O(\log^2 k)$ matches $\Omega(\log^2 k)$; both bounds are from preprints.	$2n+O(kD\log^2 k)$ (Cosson and Massoulié 2026) [preprint]	• tight
Memory	none known Memory is not bounded in these papers.	—	• unknown

Results.

- Cosson and Massoulié (2025) [preprint] [lower bound]: time $\Omega(\log^2 k)\cdot(n+kD)$. *Bottleneck:* adversarial activation. *Holds when:* asynchronous: the adversary picks the agent that moves; centralised or distributed; from a layered-graph-traversal lower bound. Equivalently, any $\mathrm{CR}(k)(n+kD)$-move algorithm has $\mathrm{CR}(k)=\Omega(\log^2 k)$.
- Cosson et al. (2023) [introductory]: time $2n+kD^2(\log k+2)$. Proposition 6: adversarial break-downs, the setting later formalised as asynchronous exploration; stated as an average of $2n/k+D^2(\log k+2)$ moves per agent.
- Cosson and Massoulié (2024) [also]: time $2n+O(k^2D)$.
- Cosson and Massoulié (2026) [preprint] [current best]: time $2n+O(kD\log^2 k)$. Order-optimal regret. Preprint (2026).

Claude (Anthropic) was used in cataloguing the problems.

Notes. The asynchronous model was introduced by (Cosson 2024b). Both matching bounds are from preprints.

16.12.7 Trees, communication by whiteboards only

Adds + synchronous, + free team, + tree VERIFIED

Agents talk only by writing at the node they occupy; the original algorithm still gives $O(k/\log k)$.

Assumed: graph dynamics: static graph; edge direction: undirected; node identifiers: labelled nodes (an agent can recognise a node it has visited); node memory: whiteboards at nodes; timing: synchronous rounds; connectivity of the team: the agents may spread out; topology: tree. **Not assumed:** randomness: deterministic; communication: none (silent agents; only see how many agents are co-located); team size: any number k of agents.

Measure	Lower bound	Best upper bound	Status
Time	none known $O(k/\log k)$ versus the inherited $\Omega(\log k/\log\log k)$ competitive ratio.	$O(D+n/\log k)$ (Fraigniaud et al. 2006a)	• polynomial gap
Memory	none known Memory is not bounded in these papers.	—	• unknown

Results.

- Fraigniaud et al. (2006a) [current best]: time $O(D+n/\log k)$. Introductory and current best in this model.

Open questions.

Q1. Is there a separation between write-read and complete communication? (Cosson and Massoulié 2025) conjecture that centralised communication helps.

16.12.8 Trees, asynchronous, whiteboards only

Adds asynchronous, + free team, + tree VERIFIED

The fully distributed asynchronous setting: the first competitive algorithm appeared in 2025.

Assumed: graph dynamics: static graph; edge direction: undirected; node identifiers: labelled nodes (an agent can recognise a node it has visited); node memory: whiteboards at nodes; connectivity of the team: the agents may spread out; topology: tree. **Not assumed:** randomness: deterministic; communication: none (silent agents; only see how many agents are co-located); timing: asynchronous; team size: any number k of agents.

Measure	Lower bound	Best upper bound	Status
Time	none known $O(k/\log k)$ versus the inherited $\Omega(\log^2 k)$ competitive ratio.	$2n+O(k^2 2^k D)$ (Cosson and Massoulié 2025) [preprint]	• polynomial gap
Memory	none known Memory is not bounded in these papers.	—	• unknown

Results.

- Cosson and Massoulié (2025) [preprint] [current best]: time $2n+O(k^2 2^k D)$. Introductory and current best; a variant is $O(k/\log k)$-competitive. Previous distributed algorithms, including that of (Fraigniaud et al. 2006a), fail here.

Open questions.

Q1. Is there a distributed asynchronous algorithm with polylogarithmic competitive ratio?

Claude (Anthropic) was used in cataloguing the problems.

16.12.9 Trees, no communication

Adds no communication, no whiteboards, + synchronous, + free team, + tree VERIFIED

Without any communication, k agents are no faster than one.

Assumed: graph dynamics: static graph; edge direction: undirected; node identifiers: labelled nodes (an agent can recognise a node it has visited); timing: synchronous rounds; connectivity of the team: the agents may spread out; topology: tree. **Not assumed:** randomness: deterministic; communication: none (silent agents; only see how many agents are co-located); node memory: no whiteboards; team size: any number k of agents.

Measure	Lower bound	Best upper bound	Status
Time	$\Omega(k)\cdot$ optimal (Fraigniaud et al. 2006a) Tight: a single depth-first search is $O(k)$-competitive, and without communication no algorithm is better by more than a constant (Fraigniaud et al. 2006a).	$O(k)\cdot$ optimal (Fraigniaud et al. 2006a)	• tight
Memory	none known Memory is not bounded in these papers.	—	• unknown

Results.

- Fraigniaud et al. (2006a) [lower bound]: time $\Omega(k)\cdot$ optimal. *Bottleneck:* no information sharing. *Holds when:* each agent knows only what it explored itself; some trees.
- Fraigniaud et al. (2006a) [current best]: time $O(k)\cdot$ optimal. The baseline that the lower bound matches: one agent explores the whole tree by depth-first search, in $2(n-1)$ moves, while the others wait; this is within a factor $O(k)$ of the optimum $\Theta(n/k+D)$ (Fraigniaud et al. 2006a).

16.12.10 Large teams, local communication

Adds + local communication, + synchronous, + $k \geq Dn^{1+\varepsilon}$, + free team VERIFIED

If agents only talk when they meet, large teams still explore in $O(D)$ rounds, and the price of locality is visible in the constant.

Assumed: graph dynamics: static graph; edge direction: undirected; node identifiers: labelled nodes (an agent can recognise a node it has visited); communication: local (co-located agents communicate); timing: synchronous rounds; team size: large teams, $k \geq D\,n^{1+\varepsilon}$; connectivity of the team: the agents may spread out. **Not assumed:** randomness: deterministic; node memory: no whiteboards; topology: arbitrary connected graph.

Measure	Lower bound	Best upper bound	Status
Time	$D(1+\frac{2}{c}-o(1))$ (Dereniowski et al. 2015a) $\Theta(D)$; the constants differ.	$D(1+\frac{2}{\gamma-1}+o(1))$ (Dereniowski et al. 2015a)	• tight
Memory	none known Memory is not bounded in these papers.	—	• unknown

Results.

- Dereniowski et al. (2015a) [lower bound]: time $D(1+\frac{2}{c}-o(1))$. *Bottleneck:* locality of communication. *Holds when:* $k = Dn^{\gamma}$ agents; $D = n^{o(1)}$; separates local from global communication.
- Dereniowski et al. (2015a) [current best]: time $D(1+\frac{2}{\gamma-1}+o(1))$. Introductory and current best; $O(D)$ also when n and D are unknown.

Claude (Anthropic) was used in cataloguing the problems.

16.12.11 Anonymous graphs with $k = n$ agents

Adds $+k = n$, + free team VERIFIED

Nodes cannot be recognised and agents only talk when co-located: n agents explore by dispersing, in $O(n)$ rounds with $O(\log n)$ bits each.

Assumed: graph dynamics: static graph; edge direction: undirected; communication: local (co-located agents communicate); timing: synchronous rounds; team size: $k = n$ agents; connectivity of the team: the agents may spread out. **Not assumed:** node identifiers: anonymous nodes; randomness: deterministic; node memory: no whiteboards; topology: arbitrary connected graph.

Measure	Lower bound	Best upper bound	Status
Time	$\Omega(n)$ (Kshemkalyani and Ali 2019) $\Theta(n)$ in the worst case over graphs.	$O(n)$ (Kshemkalyani et al. 2025b)	• tight
Memory	none known Dispersion needs $\Omega(\log n)$ bits (Augustine and Moses Jr. 2018), but exploration need not disperse; no memory lower bound for exploration alone is recorded.	$O(\log n)$ (Kshemkalyani et al. 2025b)	• unknown

Results.

- Kshemkalyani and Ali (2019) [lower bound]: time $\Omega(n)$. *Bottleneck:* distance. *Holds when:* a path with all agents on one endpoint.
- Augustine and Moses Jr. (2018) [introductory]: time $O(m)$, memory $O(\log n)$. *Assumes:* knows n and m. Rooted dispersion; Augustine and Moses Jr. observe that any dispersion algorithm with $k = n$ solves n-agent exploration under the same conditions.
- Kshemkalyani et al. (2025b) [current best]: time $O(n)$, memory $O(\log n)$. Rooted dispersion in $O(k)$ rounds with $O(\log(k + \Delta))$ bits; exploration follows by the observation above (our inference, not stated in the paper).

Open questions.

Q1. Can n agents explore an anonymous tree in $o(n)$ rounds, e.g. $O(D^2)$, with local communication? With labelled nodes and global communication D^2 suffices (Disser et al. 2020).

Notes. See the overview *From exploration to dispersion.*

16.12.12 Anonymous graphs, a team of constant-memory agents

Adds any k, + free team VERIFIED

$\Theta(\log \log n)$ agents with constant memory are necessary and sufficient to explore any anonymous graph.

Assumed: graph dynamics: static graph; edge direction: undirected; communication: local (co-located agents communicate); timing: synchronous rounds; connectivity of the team: the agents may spread out. **Not assumed:** node identifiers: anonymous nodes; randomness: deterministic; node memory: no whiteboards; team size: any number k of agents; topology: arbitrary connected graph.

Measure	Lower bound	Best upper bound	Status
Time	none known Polynomial; exponent not tracked.	$n^{O(1)}$ (Disser et al. 2019)	• unknown
Memory	$\Omega(\log \log n)$ agents (Disser et al. 2019) $\Theta(\log \log n)$ agents with $O(1)$ bits.	$O(1)$ bits each, $O(\log \log n)$ agents (Disser et al. 2019)	• tight

Claude (Anthropic) was used in cataloguing the problems.

Results.

- Disser et al. (2019) [lower bound]: memory $\Omega(\log \log n)$ agents. *Bottleneck:* trap graphs. *Holds when:* each agent has $O((\log n)^{1-\varepsilon})$ bits; deterministic.
- Disser et al. (2019) [current best]: time $n^{O(1)}$, memory $O(1)$ bits each, $O(\log \log n)$ agents. *Assumes:* n unknown. Introductory and current best. One agent walks, the others act as pebbles.

Notes. Here the question is not speed but the size of a team of finite automata; it mirrors the single-agent pebble variant.

16.13 Dispersion

Spread $k \leq n$ agents over the nodes of an n-node anonymous graph so that no node hosts more than one agent, as fast as possible and with as little memory per agent as possible.

First posed in 2018 (Augustine and Moses Jr. 2018). Posed at ICDCN 2018, fourteen years after collaborative exploration; the introductory paper already relates dispersion to n-robot collaborative exploration and takes its tree lower bound from it.

Problem statement and measures

Given a connected graph G with n nodes and m edges, and $k \leq n$ agents placed on the nodes of G in any initial **configuration** (the node of each agent; several agents may share a node), the agents must reach a configuration in which **every node hosts at most one agent**, and stay there.

Time is the number of rounds (synchronous) or epochs (asynchronous; an **epoch** is an interval in which every agent completes at least one full Communicate–Compute–Move step, not necessarily the shortest one) until the dispersed configuration is reached; where a result also makes every agent terminate explicitly, the notes say so. **Memory** is the number of bits of persistent memory each agent needs, counting its $O(\log k)$-bit identifier. Unless a variant or a result says otherwise, agents know nothing about n, m, k, Δ or D.

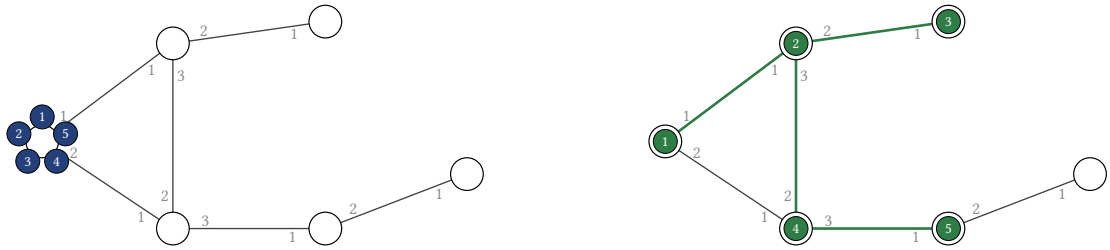
Dispersion is the natural "one agent per node" problem: it is equivalent to leaving one agent at every node of a depth-first search (DFS) tree of k nodes, which is why depth-first search bounds such as $\min(m, k\Delta)$ appear throughout, and it is a building block for exploration, leader election and self-deployment of agents.

Measures: **Time**, rounds in the synchronous model, epochs in the asynchronous model, until dispersion is reached; **Memory**, bits of persistent memory per agent, including its identifier.

A first look

Think of k electric cars arriving in a town where each charging point can serve only one car, and no car has a map. We model the town as a **graph**: charging points are **nodes**, roads between them are **edges**, and each car is an **agent**. The graph is connected and has n nodes and m edges; the nodes are **anonymous** (they have no names and nothing can be written on them). The $k \leq n$ agents have distinct **identifiers** (ID numbers) and start on nodes, several possibly on the same node. Time passes in synchronous **rounds**: in each round, agents at the same node can read each other's memory (**local communication**), and then each agent may move along one edge, choosing it by its **port** (the local number of that edge at the node). The agents know nothing about the graph, not even n. They must

Claude (Anthropic) was used in cataloguing the problems.



(a) Start: k = 5 agents on one node; ports numbered at each node

(b) Final: agent 5 settles; at most one agent per node

Figure 16.13: Dispersion from a rooted start, by the depth-first search of (Augustine and Moses Jr. 2018) and (Kshemkalyani and Ali 2019): $k = 5$ agents start on one node of an anonymous port-labelled graph and move as one group. At each new node the agent with the smallest ID settles (green) and records its parent port; the group tries the ports in order, goes back when a node is occupied, and returns through the parent port when no port is left. One frame per round; the green edges form the depth-first tree.

reach a **configuration** (the node of each agent) in which **every node holds at most one agent**; we measure the number of rounds and the number of bits of **memory** each agent needs.

1. Algorithm: Depth-first search with a settled agent per node. (Augustine and Moses Jr. 2018) (Rooted initial configuration, p. 268) When all k agents start on one node (the **rooted** case), the agents move as one group and run a **depth-first search** (DFS: go forward through an untried port while there is one, go back otherwise), using agents that stay behind on nodes (**settled agents**) as markers, as in (Augustine and Moses Jr. 2018) and (Kshemkalyani and Ali 2019).

1. At the start node, the agent with the smallest ID **settles**: it stays there for good and remembers by which port the group arrived (its **parent port**; none at the start).
2. The rest of the group leaves by the next port not yet tried at this node.
3. If the node the group reaches already has a settled agent, it has been visited: the group goes back by the port it came in and tries the next port there.
4. If the node is empty, the smallest ID in the group settles there and records the parent port; go to step 2.
5. When every port of a node has been tried, the group goes back through that node's parent port, which the settled agent tells it.
6. The group shrinks by one at each new node, and the procedure ends when the last agent settles.

Why it works and what it costs. This is a depth-first search in which the settled agents replace the "visited" marks that anonymous nodes cannot hold. Each edge is tried at most once from each of its two endpoints, and each try costs two moves (there and back) or turns into a tree edge, so the time is $O(m)$ rounds; tries happen only at the at most k nodes that receive an agent, each with at most Δ ports, which gives $O(\min(m, k\Delta))$, where Δ is the maximum degree. A settled agent remembers its ID and a few port numbers, $O(\log(k + \Delta))$ bits. The current best is $O(k)$ rounds with $O(\log(k + \Delta))$ bits, from any start (Kshemkalyani et al. 2025b): it gets rid of the cost of checking every port, which held the field at $O(\min(m, k\Delta))$ until (Sudo et al. 2026a).

Why it matters. Using the agents themselves as markers is the idea every dispersion algorithm builds on, and it ties dispersion to collaborative exploration with anonymous nodes.

Claude (Anthropic) was used in cataloguing the problems.

2. Lower bound: Distance: Omega(k) rounds. (Kshemkalyani and Ali 2019) (Arbitrary graph, arbitrary initial configuration, p. 266) Put all k agents on one end of a path (nodes in a row, each joined by an edge to the next). In the end the k agents must stand on k different nodes of the path, so one of them stands at least $k-1$ edges from the end where they all started. An agent crosses at most one edge per round.

Why it holds. So some agent needs at least $k-1$ rounds, whatever the algorithm: $\Omega(k)$ (Kshemkalyani and Ali 2019). The argument uses only distance, so it holds even for randomised algorithms, unlimited memory and global communication. The $O(k)$ algorithm of (Kshemkalyani et al. 2025b) matches it, so the time question is settled up to a constant factor.

Why it matters. It fixed the target: after it, every dispersion algorithm was measured by how far it is from k rounds.

3. Lower bound: Symmetry: Omega(log k) bits. (Augustine and Moses Jr. 2018) (Arbitrary graph, arbitrary initial configuration, p. 266) Suppose all agents run the same deterministic algorithm with the same amount of memory, and put them all on one node. If two of them start with exactly the same memory contents, they see the same things, make the same decisions and take the same ports in every round, so they stay together forever and dispersion fails. Hence the k agents on that node need k different memory contents at the start.

Why it holds. With b bits there are 2^b possible memory contents, so $2^b \geq k$, that is $b \geq \log k$ (Augustine and Moses Jr. 2018). This is why the bound counts the identifier: the ID is exactly what makes the memories differ. The best algorithms use $O(\log(k+\Delta))$ bits, so memory is optimal whenever Δ is at most polynomial in k; whether the $\log\Delta$ term is needed has no published proof for agents with identifiers.

Why it matters. It is the simplest instance of symmetry breaking, the obstacle behind the anonymous-agent results in dispersion, rendezvous and gathering.

What the community started looking at

Augustine and Moses Jr. posed dispersion as **a study of memory–time trade-offs** (Augustine and Moses Jr. 2018): the primary aim was to **minimise the number of rounds**, and then the **memory per agent**, with $\Omega(k)$ rounds and $\Omega(\log k)$ bits as the targets. Most of the later work closes these two gaps under weaker assumptions (arbitrary start, asynchrony, faults, dynamic graphs); the time-optimal $O(k)$ algorithm with $O(\log(k+\Delta))$ bits came in 2025 (Kshemkalyani et al. 2025b). Its roots are in n-agent collaborative exploration, which a dispersion with $k=n$ solves.

Further reading

Papers on this problem that are not yet placed in a setting.

- Himani et al. (2026b). Location-aware dispersion, a generalisation in which every node and every agent has a colour and each agent must settle alone on a node of its own colour (with one colour it is dispersion). Agents with distinct identifiers, local communication, synchronous rounds, $k \leq n$: a single agent cannot solve it without knowing n (Theorem 1); trees and rings take $O(n)$ rounds with $O(\log(k+\Delta))$ bits (Corollary 1); from a rooted start, $O(m)$ rounds and $O(\log k)$ bits when $k=n$ and both are known (Theorem 3), $O((n/k)\,m)$ rounds and $O((n/k)\log(k+\Delta))$ bits knowing n (Theorem 4), and $O(\log(n/k)\,(n/k)\,m)$ rounds without knowing n (Theorem 5); arbitrary and dispersed starts are reduced to these, the latter by first gathering the agents, in up to $\tilde{O}(n^5)$ rounds (Theorems 6–10). The stated lower bound of $\Omega(\min\{n^2/k, m\})$ rounds and $\Omega((n/k)\log k)$ bits (Theorem 2) is argued only for memory, and informally. Preprint. Read.
- Kaur and Mondal (2023). Distance-2-dispersion: settled agents may not be adjacent, and an agent may share a node only when no node is left to settle; with $k \geq n$ agents the settled nodes form a maximal independent set (see the MIS problem). From a rooted start with $O(\log\Delta)$ bits besides the identifier, $2\Delta(8m-3n+3)$ rounds

Claude (Anthropic) was used in cataloguing the problems.

(Theorem 3.4), against $\Omega(m\Delta)$ for agents with that memory (Theorem 4.1). An arbitrary start is solved in (Kaur et al. 2026c). Read (arXiv v1).

- Amir and Bruckstein (2020). Uniform dispersal: identical asynchronous robots enter an unknown graph over time from a source node, settle on empty nodes and guide the others; each node holds a bounded number of robots, and moving robots may crash. If at most $ct/4$ crashes occur before time t, with $0 \le c < 1$, every node is settled by time $8((1-c)^{-1} + o(1))n$ asymptotically almost surely (Theorem III.1), by comparison with a totally asymmetric simple exclusion process. Read (arXiv v3).

- Rappel et al. (2022). Simple finite-memory airborne agents explore and fill an unknown connected region of the square grid, settling one per cell. A preprint; abstract read.

- Himani et al. (2026a). Dispersion on a $\sqrt{n}\times\sqrt{n}$ torus, with oriented or unoriented port numbers, by synchronous agents that read the memory of agents on their node and know neither n nor k: $O(\sqrt{n})$ rounds with $O(\log n)$ bits per agent, from any start (Theorem 1; Section 6 removes the knowledge of n, Sections 8 and 9 the orientation); this extends the oriented-grid algorithm of (Kshemkalyani et al. 2020c). Read (model, algorithm and theorems).

Relations to other problems

- **Derived from Collaborative exploration.** With $k = n$ agents, a dispersed configuration has an agent on every node, so any dispersion algorithm solves n-agent collaborative exploration under the same assumptions (Augustine and Moses Jr. 2018). Consequently every lower bound for n-agent exploration is a lower bound for dispersion (this is how the $\Omega(D^2)$ tree bound of Disser et al. 2020 enters), and every dispersion algorithm is an exploration algorithm for anonymous graphs with local communication. For $k < n$ dispersion asks for less: only k nodes must be occupied.

- **Related family: Computing with agents.** Most algorithms first bring the n agents into a dispersed configuration, one agent per node, so that every node has an agent to compute for it (Kshemkalyani et al. 2026; Chand and Molla 2026).

- **Related: Uniform deployment on graphs.** With $k = n$ agents, dispersion and uniform deployment ask for the same final configuration, one agent on every node; dispersion also allows agents to share nodes at the start, uniform deployment does not (Shibata et al. 2022a).

Assumptions specific to this problem

Grid orientation

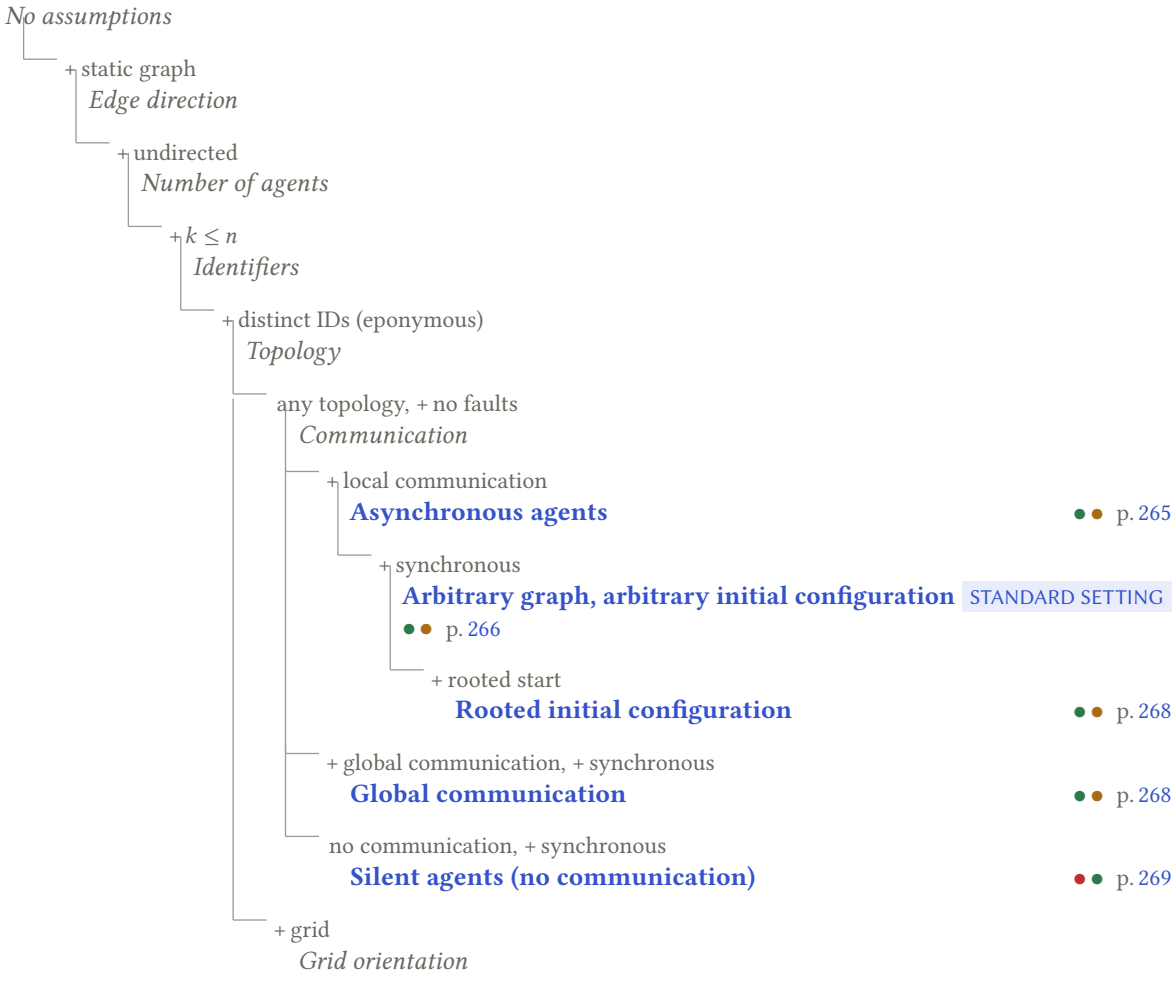
In the graph model of dispersion the port labels at a node are local and independent of those at its neighbours (Augustine and Moses Jr. 2018). Kshemkalyani, Molla and Sharma posed dispersion on grids (Kshemkalyani et al. 2020c); later papers describe their setting as an oriented ("planar") grid (Banerjee et al. 2024, 2025; Kumar et al. 2026a), although the arXiv version of (Kshemkalyani et al. 2020c) says the agents have no consistent compass (see the grid settings). Banerjee, Kumar and Molla made the distinction explicit and studied both kinds of grid, with and without crash faults (Banerjee et al. 2024). Orientation matters because a grid is symmetric and the known algorithms steer by its geometry: they walk straight to the boundary, follow it to a corner and sweep rows. In an unoriented grid, in their words, "robots cannot distinguish whether they are moving in the same direction (i.e., along a row or column) or traversing in a cycle or zigzag manner across the unoriented grid" (Banerjee et al. 2024). With a consistent compass, sweeps can start where the agents are, which gives $O(\sqrt{k})$ rounds with local communication (Kumar et al. 2026a).

Claude (Anthropic) was used in cataloguing the problems.

Level	Value	Formal definition
0	**unoriented**	Port labels carry no global meaning: at every node v the δ_v incident edges get the labels $1, \dots, \delta_v$ in an arbitrary order, chosen independently at each node, so an agent that enters v cannot tell from the labels which port continues in the same row or column. This is the default on every graph; on a grid it is called an *unoriented* grid.
1	**oriented**	The graph is a grid with nodes (x, y), and every node labels its ports North, East, South and West consistently: the port North of (x, y) leads to $(x, y+1)$, East to $(x+1, y)$, South to $(x, y-1)$ and West to $(x-1, y)$, whenever that neighbour exists. The same direction carries the same label at every node, so an agent can keep moving along a row or column; a boundary node is recognised by its missing ports.

Case tree

The root assumes nothing; going down, each edge adds the assumptions written in grey (+) or leaves an axis unassumed. Bold entries are catalogued settings with their status per measure (Time, Memory) and page; italic entries are branch points, named after the axis on which their branches differ. Axis order: Graph dynamics → Edge direction → Number of agents → Identifiers → Randomness → Topology → Grid orientation → Faults → Communication → Timing → Node memory → Node identifiers → Initial configuration.



Claude (Anthropic) was used in cataloguing the problems.

+ oriented
Faults
+ no faults, + local communication, + synchronous
Oriented grids ●● p. 270
+ at most crash faults, + local communication, + synchronous
Crash faults on oriented grids ●● p. 272
+ at most weak Byzantine faults, + local communication, + synchronous
Weak Byzantine agents on oriented grids ●● p. 272

unoriented, + no faults, + local communication, + synchronous
Unoriented grids ●● p. 273

unoriented, + at most crash faults, + local communication, + synchronous
Crash faults on unoriented grids ●● p. 274

+ ring
Faults
+ no faults
Communication
+ local communication, + synchronous
Rings and paths ●● p. 275
no communication, + synchronous
Silent agents on rings ●● p. 276

+ at most weak Byzantine faults, + local communication, + synchronous
Weak Byzantine agents on rings ●● p. 276

+ tree, + no faults, + local communication, + synchronous
Trees ●● p. 277
+ rooted start
Rooted trees ●● p. 278

any topology, + at most crash faults, + local communication, + synchronous
Crash faults ●● p. 279

any topology, + at most weak Byzantine faults, + local communication, + synchronous
Weak Byzantine agents ●● p. 280

any topology, any faults, + local communication, + synchronous
Strong Byzantine agents ●● p. 281

anonymous, + private coins, + no faults, + local communication, + synchronous
Anonymous agents with randomness ●● p. 282

any k, + private coins, + no faults, + local communication, + synchronous
More agents than nodes ●● p. 283

directed, + $k \leq n$, + distinct IDs (eponymous), + no faults
Communication
+ local communication, + synchronous
Directed graphs ●● p. 284
+ h-hop communication, + synchronous
Directed graphs, 1-hop communication ●● p. 285

+ connected every round, + undirected, + $k \leq n$, + distinct IDs (eponymous)
Topology

Claude (Anthropic) was used in cataloguing the problems.

any topology, + no faults
Communication
+ local communication, + synchronous
Dynamic graphs (1-interval connected) ●● p. 286
+ global communication, + synchronous
Dynamic graphs with global communication ●● p. 287
+ ring, + no faults, + local communication, + synchronous
Dynamic rings ●● p. 287

What is known

The best upper bound against the best lower bound in each catalogued setting, in tree order. The settings follow in detail.

Setting	Time	Memory	p.
Asynchronous agents	● tight $O(k)$ vs $\Omega(k)$	● near-optimal $O(\log(k+\Delta))$ vs $\Omega(\log k)$	265
Arbitrary graph, arbitrary initial configuration STANDARD SETTING	● tight $O(k)$ vs $\Omega(k)$	● near-optimal $O(\log(k+\Delta))$ vs $\Omega(\log k)$	266
Rooted initial configuration	● tight $O(k)$ vs $\Omega(k)$	● near-optimal $O(\log(k+\Delta))$ vs $\Omega(\log k)$	268
Global communication	● tight $O(k)$ vs $\Omega(k)$	● near-optimal $O(\log(k+\Delta))$ vs $\Omega(\log k)$	268
Silent agents (no communication)	● polynomial gap $O(k \log L + k^2 \log \Delta)$ vs $\Omega(k + \log L)$	● tight $O(\log L + \log \Delta)$ vs $\Omega(\log L + \log \Delta)$	269
Oriented grids	● tight $O(\sqrt{k})$ vs $\Omega(\sqrt{k})$	● tight $O(\log k)$ vs $\Omega(\log k)$	270
Crash faults on oriented grids	● polynomial gap $O(\sqrt{n})$ vs $\Omega(\sqrt{k})$	● tight $O(\log n)$ vs $\Omega(\log k)$	272
Weak Byzantine agents on oriented grids	● tight $O(\sqrt{n})$ vs $\Omega(\sqrt{n})$	● polynomial gap $O(n \log n)$ vs $\Omega(\log n)$	272
Unoriented grids	● polynomial gap $O(\min(k, \sqrt{n}))$ vs $\Omega(\sqrt{k})$	● tight $O(\log k)$ vs $\Omega(\log k)$	273
Crash faults on unoriented grids	● polynomial gap $O(\sqrt{n} \log n)$ vs $\Omega(\sqrt{k})$	● polynomial gap $O(\sqrt{n} \log n)$ vs $\Omega(\log k)$	274
Rings and paths	● tight $O(k)$ vs $\Omega(k)$	● tight $O(\log k)$ vs $\Omega(\log k)$	275
Silent agents on rings	● tight $O(k + \log L)$ vs $\Omega(k + \log L)$	● tight $O(\log L)$ vs $\Omega(\log L)$	276
Weak Byzantine agents on rings	● tight $O(n)$ vs $\Omega(n)$	● tight $O(\log n)$ vs $\Omega(\log n)$	276
Trees	● polynomial gap $O(k)$ vs $\Omega(D^2)$	● near-optimal $O(\log(k+\Delta))$ vs $\Omega(\log k)$	277
Rooted trees	● polynomial gap $O(D^2)$ vs $\Omega(D^2)$	● tight $O(\log n)$ vs $\Omega(\log k)$	278
Crash faults	● polynomial gap $O(\min(m, k\Delta))$ vs $\Omega(k)$	● near-optimal $O(k \log(k+\Delta))$ vs $\Omega(\log k)$	279
Weak Byzantine agents	● polynomial gap $\tilde{O}(n^9)$ vs $\Omega(k)$	● unknown — vs $\Omega(\log k)$	280

Claude (Anthropic) was used in cataloguing the problems.

Setting	Time	Memory	p.
Strong Byzantine agents	● polynomial gap $O(X(n)+n^3)$ vs $\Omega(k)$	● unknown — vs —	281
Anonymous agents with randomness	● polynomial gap $O(k^2)$ vs $\Omega(k)$	● tight $O(\log \Delta)$ vs $\Omega(\log \Delta)$	282
More agents than nodes	● unknown $O(m+k)$ vs —	● polynomial gap $O(\Delta)$ vs $\Omega(\log \Delta)$	283
Directed graphs	● polynomial gap $O(k^2\Delta_{out})$ vs $\Omega(k)$	● near-optimal $\Theta(\log(k+\Delta_{out}))$ vs $\Omega(\log k)$	284
Directed graphs, 1-hop communication	● polynomial gap $O(kD)$ vs $\Omega(k)$	● polynomial gap $O(k\log(k+\Delta_{out}))$ vs $\Omega(\log k)$	285
Dynamic graphs (1-interval connected)	● impossible	● impossible	286
Dynamic graphs with global communication	● tight $\Theta(k)$ vs $\Omega(k)$	● tight $\Theta(\log k)$ vs $\Omega(\log k)$	287
Dynamic rings	● tight $O(n)$ vs $\Omega(n)$	● tight $O(1)$ vs —	287

16.13.1 Asynchronous agents

Adds + local communication VERIFIED

Each agent's communicate–compute–move step takes finite but unbounded time; time is counted in epochs, intervals in which every agent completes at least one step.

Assumed: graph dynamics: static graph; edge direction: undirected; number of agents: $k \le n$; identifiers: distinct identifiers (eponymous): agents have pairwise distinct identifiers; faults: no faults; communication: local (co-located agents communicate).
Not assumed: randomness: deterministic; topology: arbitrary connected graph; grid orientation: unoriented (no consistent compass); timing: asynchronous; node memory: no whiteboards; node identifiers: anonymous nodes; initial configuration: arbitrary (agents on any $\ell \le k$ nodes).

Statement for this setting. Same task; an adversarial scheduler activates agents at arbitrary speeds. An **epoch** is an interval of time in which every agent performs at least one full step; it need not be the shortest one, and bounds hold for every division of the execution into consecutive epochs. Settled agents must typically stay active, because an agent that stops cannot be told that it is needed.

Measure	Lower bound	Best upper bound	Status
Time	$\Omega(k)$ (Kshemkalyani and Ali 2019) $O(k)$ epochs versus $\Omega(k)$.	$O(k)$ (Pattanayak et al. 2026)	● tight
Memory	$\Omega(\log k)$ (Augustine and Moses Jr. 2018) $O(\log(k+\Delta))$ versus $\Omega(\log k)$.	$O(\log(k+\Delta))$ (Pattanayak et al. 2026)	● near-optimal

Results.

- Kshemkalyani and Ali (2019) [lower bound]: time $\Omega(k)$. *Bottleneck:* distance. *Holds when:* worst case over inputs: a path with all k agents on one endpoint; one edge per round; holds for randomized algorithms, unbounded memory and global communication; counted in epochs. Epochs; the synchronous argument carries over.
- Augustine and Moses Jr. (2018) [lower bound]: memory $\Omega(\log k)$. *Bottleneck:* symmetry breaking. *Holds when:* deterministic; all agents have the same memory bound; an initial configuration with co-located agents (rooted suffices); memory counts the identifier.
- Kshemkalyani and Ali (2019) [introductory]: time $O(m)$, memory $O(k \log \Delta)$. *Assumes:* knows k. Three asynchronous algorithms; the only terminating one uses $O(D \log \Delta)$ bits and $O(\Delta^D)$ time. (Kshemkalyani 2025) is the journal treatment: $O(\min(m, k\Delta))$ time with $O(k \log \Delta)$ bits and no parameter knowledge.

Claude (Anthropic) was used in cataloguing the problems.

- Kshemkalyani and Sharma (2025) [also]: time $O(\min(m, k\Delta))$, memory $O(\log(k+\Delta))$. Same bounds as its synchronous algorithm, in epochs.
- Kshemkalyani et al. (2025b) [also]: time $O(k \log k)$, memory $O(\log(k+\Delta))$. First near-linear asynchronous bound, for rooted and arbitrary configurations.
- Pattanayak et al. (2026) [current best]: time $O(k)$, memory $O(\log(k+\Delta))$. Time-optimal, rooted and arbitrary configurations. Uses the **port-one tree**: following port 1 from every node defines a structure that lets an agent check a neighbour's status in $O(1)$ epochs without timing assumptions.

Open questions.

Q1. Can an asynchronous algorithm let every agent **terminate** in polynomial time without knowledge of the graph? The only terminating asynchronous algorithm known uses $O(\Delta^{\min(D,k)+1})$ time (Kshemkalyani 2025); the linear-time algorithms keep settled agents active.

Q2. Semi-synchronous schedulers sit between the two models; no separation or dedicated bound is known.

Notes. The asynchronous time gap closed in two steps in 2025: $O(k \log k)$ (Kshemkalyani et al. 2025b), then $O(k)$ (Pattanayak et al. 2026). Sudo et al. (2026a)'s rooted algorithms also work under any fair scheduler with atomic moves, but its general-configuration algorithm needs full synchrony.

16.13.2 Arbitrary graph, arbitrary initial configuration

Adds + synchronous **to** Asynchronous agents (p. 265) STANDARD SETTING VERIFIED

The standard setting: $k \le n$ agents with unique identifiers, anywhere on an arbitrary anonymous graph, synchronous rounds, local communication, no faults, no knowledge of the graph.

Assumed: graph dynamics: static graph; edge direction: undirected; number of agents: $k \le n$; identifiers: distinct identifiers (eponymous): agents have pairwise distinct identifiers; faults: no faults; communication: local (co-located agents communicate); timing: synchronous rounds. **Not assumed:** randomness: deterministic; topology: arbitrary connected graph; grid orientation: unoriented (no consistent compass); node memory: no whiteboards; node identifiers: anonymous nodes; initial configuration: arbitrary (agents on any $\ell \le k$ nodes).

Measure	Lower bound	Best upper bound	Status
Time	$\Omega(k)$ (Kshemkalyani and Ali 2019)	$O(k)$ (Kshemkalyani et al. 2025b)	• tight
	$O(k)$ matches $\Omega(k)$ as of 2025; the question raised in 2019–2021 of whether $\Omega(\min(m, k\Delta))$ could be a lower bound is settled in the negative.		
Memory	$\Omega(\log k)$ (Augustine and Moses Jr. 2018)	$O(\log(k+\Delta))$ (Kshemkalyani et al. 2025b)	• near-optimal
	$O(\log(k+\Delta))$ versus $\Omega(\log k)$: the two differ only when Δ is super-polynomial in k, and no $\Omega(\log \Delta)$ lower bound is proven.		

Results.

- Kshemkalyani and Ali (2019) [lower bound]: time $\Omega(k)$. *Bottleneck:* distance. *Holds when:* worst case over inputs: a path with all k agents on one endpoint; one edge per round; holds for randomized algorithms, unbounded memory and global communication. A path with all k agents at one end. Also $\Omega(D)$, and $\Omega(n)$ in the worst case, when $k = n$ (Augustine and Moses Jr. 2018).
- Augustine and Moses Jr. (2018) [lower bound]: memory $\Omega(\log k)$. *Bottleneck:* symmetry breaking. *Holds when:* deterministic; all agents have the same memory bound; an initial configuration with co-located agents (rooted suffices); memory counts the identifier. Two agents with identical state move in lockstep forever; stated as $\Omega(\log n)$ for $k = n$. Several papers quote $\Omega(\log(k+\Delta))$, but the $\log \Delta$ term has no published proof for deterministic agents with identifiers.

Claude (Anthropic) was used in cataloguing the problems.

- Augustine and Moses Jr. (2018) [introductory]: time $O(mn)$, memory $O(\log n)$. *Assumes:* $k = n$; knows n and m. Also $O(m)$ time with $O(n \log n)$ bits. Introduces the problem and the memory–time trade-off view.
- Kshemkalyani and Ali (2019) [also]: time $O(m)$, memory $O(k \log \Delta)$. *Assumes:* knows k. First treatment of $k \le n$. Also $O((m-n)k)$ time with $O(\log k + \log \Delta)$ bits.
- Kshemkalyani et al. (2019) [also]: time $O(\min(m, k\Delta) \log k)$, memory $O(\log n)$. *Assumes:* knows m, Δ, k. Parallel DFS traversals merged over $O(\log \ell)$ passes (ℓ = number of initially occupied nodes, so $\log \ell \le \log k$). Without knowing m: $O(k\Delta \log k)$ time with $O(\log(k+\Delta))$ bits.
- Shintaku et al. (2020) [also]: time $O(\min(m, k\Delta, k^2) \log \ell)$, memory $O(\log(k+\Delta))$. *Assumes:* knows a polynomial upper bound on identifiers. Same time as (Kshemkalyani et al. 2019) with no knowledge of m, k or Δ.
- Kshemkalyani and Sharma (2025) [also]: time $O(\min(m, k\Delta))$, memory $O(\log(k+\Delta))$. Removes the $\log \ell$ factor; matches single-source DFS. Optimal in time and memory on constant-degree graphs. Holds in the asynchronous model too.
- Sudo et al. (2026a) [also]: time $O(k \log k \log \min(k, \Delta))$, memory $O(\log(k+\Delta))$. *Assumes:* knows a polynomial upper bound on identifiers. First bound independent of m and Δ: $O(k \log^2 k)$. Terminating all agents simultaneously needs upper bounds on k and on the number of edges among any k nodes.
- Kshemkalyani et al. (2025b) [current best]: time $O(k)$, memory $O(\log(k+\Delta))$. Time-optimal. Settled agents oscillate between their node and a neighbour so that the DFS head can test whether a neighbour is occupied in $O(1)$ rounds; multiple DFS traversals are merged without the log factors.

Open questions.

Q1. Can the memory be reduced to $O(\log k)$ bits, independent of Δ, while keeping $O(k)$ time? Equivalently, is there an $\Omega(\log \Delta)$ memory lower bound for deterministic agents with identifiers? The bound $\Omega(\log(k+\Delta))$ is quoted in several papers but only the $\log k$ term is argued.

Q2. Can all agents **terminate explicitly** in $O(k)$ rounds without any knowledge of k, m or Δ? (Sudo et al. 2026a) terminates the rooted case in $O(k)$ extra rounds but needs global upper bounds in the general case; (Kshemkalyani et al. 2025b) does not address termination detection for arbitrary initial configurations.

Q3. **Ongoing dispersion** (Kshemkalyani and Ali 2019): agents that have settled must periodically leave (for example to recharge) and the configuration must be re-dispersed; no bounds beyond the original proposal are known.

Notes. Progression. (Augustine and Moses Jr. 2018) posed the problem for $k = n$ with $O(mn)$ time and $O(\log n)$ bits, or $O(m)$ time with $O(n \log n)$ bits, and asked whether $o(m)$ time was possible. (Kshemkalyani and Ali 2019) allowed $k \le n$, gave $O(m)$ time with $O(k \log \Delta)$ bits and proved the $\Omega(k)$ time bound that every later paper is measured against. (Kshemkalyani et al. 2019) reached $O(\min(m, k\Delta) \log k)$ time with logarithmic memory but needed m, Δ and k; (Shintaku et al. 2020) removed that knowledge; (Kshemkalyani and Sharma 2025) removed the log factor and reached the DFS bound $O(\min(m, k\Delta))$, optimal for constant degree. The DFS barrier was broken by (Sudo et al. 2026a) with $O(k \log^2 k)$, and (Kshemkalyani et al. 2025b) closed the time question with $O(k)$.

What the DFS barrier was. A DFS head arriving at a node v cannot tell which neighbours already hold an agent without visiting them, which costs $\Theta(\deg(v))$ rounds per node and $\Theta(\min(m, k\Delta))$ in total. The near-linear and linear algorithms avoid this by letting settled agents advertise themselves at neighbours (Sudo et al. 2026a: time slots; Kshemkalyani et al. 2025b: oscillation), or in the asynchronous case by exploiting the port-one tree structure (Pattanayak et al. 2026).

Knowledge. Where a result needs knowledge of a parameter it is listed on the result. The 2025 linear-time algorithms do not state any parameter knowledge explicitly; their descriptions use k (for example, to select $\lceil k/3 \rceil$ seekers), so treat "no knowledge" for them as unconfirmed.

Claude (Anthropic) was used in cataloguing the problems.

16.13.3 Rooted initial configuration

Adds + rooted start **to** Arbitrary graph, arbitrary initial configuration (p. 266) VERIFIED

All k agents start on the same node, so a single DFS traversal that drops one agent per new node solves the problem; the question is how fast and with how little memory.

Assumed: graph dynamics: static graph; edge direction: undirected; number of agents: $k \leq n$; identifiers: distinct identifiers (eponymous): agents have pairwise distinct identifiers; faults: no faults; communication: local (co-located agents communicate); timing: synchronous rounds; initial configuration: rooted (all agents on one node). **Not assumed:** randomness: deterministic; topology: arbitrary connected graph; grid orientation: unoriented (no consistent compass); node memory: no whiteboards; node identifiers: anonymous nodes.

Measure	Lower bound	Best upper bound	Status
Time	$\Omega(k)$ (Kshemkalyani and Ali 2019)	$O(k)$ (Kshemkalyani et al. 2025b)	• tight
	$O(k)$ versus $\Omega(k)$.		
Memory	$\Omega(\log k)$ (Augustine and Moses Jr. 2018)	$O(\log(k+\Delta))$ (Kshemkalyani et al. 2025b)	• near-optimal
	$O(\log(k+\Delta))$ versus $\Omega(\log k)$; see the base case.		

Results.

- Kshemkalyani and Ali (2019) [lower bound]: time $\Omega(k)$. *Bottleneck:* distance. *Holds when:* worst case over inputs: a path with all k agents on one endpoint; one edge per round; holds for randomized algorithms, unbounded memory and global communication. The $\Omega(k)$ argument is already rooted: all agents at one end of a path.
- Augustine and Moses Jr. (2018) [lower bound]: memory $\Omega(\log k)$. *Bottleneck:* symmetry breaking. *Holds when:* deterministic; all agents have the same memory bound; an initial configuration with co-located agents (rooted suffices); memory counts the identifier.
- Augustine and Moses Jr. (2018) [introductory]: time $O(m)$, memory $O(\log n)$. *Assumes:* $k = n$; knows n and m. DFS that leaves the smallest-identifier agent at every newly discovered node; time $O(\min(m, k\Delta))$ for $k \leq n$ in later restatements.
- Sudo et al. (2026a) [also]: time $O(k \log \min(k, \Delta))$, memory $O(\log(k+\Delta))$. Also $O(k)$ time with $O(\Delta + \log k)$ bits, the first linear-time algorithm. Both terminate: the last settled agent detects completion and relays it through the DFS tree in $O(k)$ extra rounds. Work under any fair scheduler with atomic moves.
- Kshemkalyani et al. (2025b) [current best]: time $O(k)$, memory $O(\log(k+\Delta))$. Linear time with logarithmic memory.

Open questions.

Q1. Is $O(k)$ time achievable with $O(\log k)$ bits? Memory $O(\log(k+\Delta))$ is needed by every known algorithm to remember a port number.

Notes. The rooted case is the one solved first and the one where every measure is settled up to the $\log \Delta$ term. In the asynchronous model the rooted case also has $O(k)$ epochs (Pattanayak et al. 2026) (and $O(k \log k)$ in Kshemkalyani et al. 2025b).

16.13.4 Global communication

Adds + global communication, + synchronous VERIFIED

Any agent can read the state of any other agent at any time, but still does not know the graph or where the others are.

Assumed: graph dynamics: static graph; edge direction: undirected; number of agents: $k \leq n$; identifiers: distinct identifiers (eponymous): agents have pairwise distinct identifiers; faults: no faults; communication: global (any agent can read any other

agent's state); timing: synchronous rounds. **Not assumed:** randomness: deterministic; topology: arbitrary connected graph; grid orientation: unoriented (no consistent compass); node memory: no whiteboards; node identifiers: anonymous nodes; initial configuration: arbitrary (agents on any $\ell \leq k$ nodes).

Measure	Lower bound	Best upper bound	Status
Time	$\Omega(k)$ (Kshemkalyani et al. 2022) Inherited $O(k)$ versus $\Omega(k)$.	$O(k)$ (Kshemkalyani et al. 2025b)	• tight
Memory	$\Omega(\log k)$ (Kshemkalyani et al. 2022) $O(\log(k+\Delta))$ versus $\Omega(\log k)$.	$O(\log(k+\Delta))$ (Kshemkalyani et al. 2025b)	• near-optimal

Results.

- Kshemkalyani et al. (2022) [lower bound]: time $\Omega(k)$. *Bottleneck:* distance. *Holds when:* worst case over inputs: a path with all k agents on one endpoint; one edge per round; re-proven for global communication. Re-proven for the global model; $\Omega(D^2)$ on trees when $k = n$.
- Kshemkalyani et al. (2022) [lower bound]: memory $\Omega(\log k)$. *Bottleneck:* symmetry breaking. *Holds when:* deterministic; all agents have the same memory bound; an initial configuration with co-located agents (rooted suffices); memory counts the identifier; identifiers in $[1, k]$ counted as memory.
- Kshemkalyani et al. (2022) [introductory]: time $O(k\Delta)$, memory $O(\log(k+\Delta))$. Or $O(\min(m, k\Delta))$ time with $O(\Delta + \log k)$ bits. A BFS-based algorithm gives $O((D+k)\Delta(D+\Delta))$ time with $O(\log D + \Delta \log k)$ bits. The conference version (Kshemkalyani et al. 2020b) claimed $O(\min(m, k\Delta))$ time with $O(\log(k+\Delta))$ bits; the journal version withdraws that combination.
- Kshemkalyani et al. (2025b) [current best]: time $O(k)$, memory $O(\log(k+\Delta))$. Inherited from the local-communication model, which is weaker; no algorithm specific to global communication does better.

Open questions.

Q1. Does global communication buy anything for dispersion? Since $O(k)$ time with $O(\log(k+\Delta))$ bits holds with local communication, a separation could only come from memory ($O(\log k)$ bits), from termination detection, or from the BFS-style question of achieving time close to $O(D)$-dependent bounds for small D.

Q2. (Kshemkalyani et al. 2022) asks for $O(\min(m, k\Delta))$ time with only $O(\log(k+\Delta))$ bits in the global model; this is implied by the later local-model result, so the live version of the question is the memory question above.

Notes. The global model was introduced to see how much easier dispersion becomes when agents can coordinate freely. The answer so far is: asymptotically not at all in time, once the local-model algorithms of 2025 are taken into account.

16.13.5 Silent agents (no communication)

Adds no communication, + synchronous VERIFIED

Agents cannot exchange any information; at its node an agent only learns whether it is alone and whether the number of agents there went up or down since the last round.

Assumed: graph dynamics: static graph; edge direction: undirected; number of agents: $k \leq n$; identifiers: distinct identifiers (eponymous): agents have pairwise distinct identifiers; faults: no faults; timing: synchronous rounds. **Not assumed:** randomness: deterministic; topology: arbitrary connected graph; grid orientation: unoriented (no consistent compass); communication: none (silent agents; only see how many agents are co-located); node memory: no whiteboards; node identifiers: anonymous nodes; initial configuration: arbitrary (agents on any $\ell \leq k$ nodes).

Claude (Anthropic) was used in cataloguing the problems.

Statement for this setting. No communication at all. At the start of a round an agent at v learns (i) whether it is alone at v and (ii) whether the number of agents at v increased or decreased since the previous round. Agents have identifiers in $[0, L]$ and know only their own; memory is counted as **additional** bits beyond the identifier.

Measure	Lower bound	Best upper bound	Status
Time	$\Omega(k + \log L)$ (Das et al. 2026a) $O(k \log L + k^2 \log \Delta)$ versus $\Omega(k + \log L)$.	$O(k \log L + k^2 \log \Delta)$ (Gorain et al. 2024)	• polynomial gap
Memory	$\Omega(\log L + \log \Delta)$ (Gorain et al. 2024) $\Theta(\log L + \log \Delta)$ additional bits.	$O(\log L + \log \Delta)$ (Gorain et al. 2024)	• tight

Results.

- Das et al. (2026a) [lower bound]: time $\Omega(k + \log L)$. *Bottleneck:* timing is the only channel. *Holds when:* no communication: only alone / count-change sensing; identifiers in $[0, L]$, unknown to the other agents; $\Omega(k)$ from distance; $\Omega(\log L)$ from distinguishing identifiers by waiting; proven on rings, hence for arbitrary graphs. Proven on rings, hence for arbitrary graphs.
- Gorain et al. (2024) [lower bound]: memory $\Omega(\log L + \log \Delta)$. *Bottleneck:* encoding identifier and ports. *Holds when:* additional bits beyond the identifier; rooted configuration; deterministic. Additional bits; no algorithm with o(log L + log Δ) additional bits exists in this model.
- Gorain et al. (2024) [current best]: time $O(k \log L + k^2 \log \Delta)$, memory $O(\log L + \log \Delta)$. *Assumes:* rooted. Introductory and current best for general graphs; the agents terminate. Information is conveyed only by the timing of moves and by counts of co-located agents.

Open questions.

Q1. Close the time gap on general graphs: is $O(k + \log L)$ achievable, as on rings?

Q2. Arbitrary (multi-source) initial configurations on general graphs; only rings are solved (Das et al. 2026a).

Q3. Weaker sensing: agents that only learn whether they are alone and whether the count decreased, and agents that wake up at different times (a 2026 ICDCIT paper by Gorain, Mandal and Sharma treats this; its bounds are not recorded here).

16.13.6 Oriented grids

Adds + no faults, + local communication, + synchronous VERIFIED

A $\sqrt{n} \times \sqrt{n}$ grid with consistent North, East, South and West ports: $\Theta(\sqrt{k})$ rounds and $\Theta(\log k)$ bits with local communication, by a 2026 preprint.

Assumed: graph dynamics: static graph; edge direction: undirected; number of agents: $k \leq n$; identifiers: distinct identifiers (eponymous): agents have pairwise distinct identifiers; topology: grid; grid orientation: oriented grid (consistent North, East, South, West ports); faults: no faults; communication: local (co-located agents communicate); timing: synchronous rounds. **Not assumed:** randomness: deterministic; node memory: no whiteboards; node identifiers: anonymous nodes; initial configuration: arbitrary (agents on any $\ell \leq k$ nodes).

Statement for this setting. The graph is a $\sqrt{n} \times \sqrt{n}$ grid with $\Delta = 4$ whose ports are labelled North, East, South and West consistently at every node (an **oriented** grid); a boundary node is recognised by its missing ports. Otherwise as on unoriented grids: $k \leq n$ agents with unique identifiers start in an arbitrary configuration, communicate locally and move synchronously.

Claude (Anthropic) was used in cataloguing the problems.

Measure	Lower bound	Best upper bound	Status
Time	$\Omega(\sqrt{k})$ (Kshemkalyani et al. 2020c)	$O(\sqrt{k})$ (Kumar et al. 2026a) [preprint]	• tight
	$\Theta(\sqrt{k})$ for every $k \le n$ (Kumar et al. 2026a) (preprint; proofs not independently checked, see notes). On unoriented grids a gap remains.		
Memory	$\Omega(\log k)$ (Kshemkalyani et al. 2020c)	$O(\log k)$ (Kumar et al. 2026a) [preprint]	• tight
	$\Theta(\log k)$ (Kumar et al. 2026a).		

Results.

- Kshemkalyani et al. (2020c) [lower bound]: time $\Omega(\sqrt{k})$. *Bottleneck:* distance. *Holds when:* k agents on one node: a grid ball of radius r holds at most $2r^2 + 2r + 1$ nodes; any algorithm; holds on oriented grids, hence also on unoriented ones. All k agents on one node v: after t rounds they occupy at most $2t^2 + 2t + 1$ nodes, so $t \ge \frac{1}{2}(\sqrt{2k-1} - 1)$ (restated as Theorem 1 of Kumar et al. 2026a); $\Omega(\sqrt{n})$ when $k = \Omega(n)$.
- Kshemkalyani et al. (2020c) [lower bound]: memory $\Omega(\log k)$. *Bottleneck:* symmetry breaking. *Holds when:* deterministic; all agents have the same memory bound; an initial configuration with co-located agents (rooted suffices); memory counts the identifier.
- Kumar et al. (2026a) [preprint] [current best]: time $O(\sqrt{k})$, memory $O(\log k)$. Preprint, not peer reviewed. Spread–Reverse–Collect: each group of m co-located agents sweeps a square of side $2\lceil\sqrt{m}\rceil$ from its own position and reflects it at the boundary; groups merge by label priority and at least double in size from level to level. Any initial configuration, no knowledge of n, k or the diameter, no detection of agents crossing on an edge; the authors state it runs unchanged on the infinite oriented grid. Optimal in both measures for every $k \le n$; matches with local communication the global-communication bound of (Kshemkalyani et al. 2020c).

Open questions.

Q1. Rectangular $W \times H$ oriented grids with $W \le H$: from a corner at most $W(t+1)$ nodes are reachable in t rounds, so $\lceil k/W \rceil - 1$ rounds are needed; an algorithm matching a bound that depends on W and H, with local communication and no knowledge of W and H, is open (Kumar et al. 2026a).

Q2. $O(\sqrt{k})$ rounds on oriented grids with asynchronous agents, or on the oriented torus (Kumar et al. 2026a).

Q3. Dispersion in $O(\sqrt{k})$ rounds on oriented grids with explicit termination: every agent knows when it may stop (see notes).

Notes. Kshemkalyani, Molla and Sharma left a gap between $O(\min(k, \sqrt{n}))$ rounds with local communication and $O(\sqrt{k})$ rounds with global communication (Kshemkalyani et al. 2020c); Banerjee, Kumar and Molla asked again whether local communication suffices (Banerjee et al. 2025). Kumar, Molla, Pattanayak and Pramanick answer it on oriented grids: $O(\sqrt{k})$ rounds and $O(\log k)$ bits (Kumar et al. 2026a) (arXiv preprint, v1, October 2026, not peer reviewed). Earlier grid algorithms gather the agents at the boundary or at a corner and spread from there; Spread–Reverse–Collect starts each sweep where its agents are and reflects it at the boundary, so no knowledge of n is needed. (Kshemkalyani et al. 2020c) is placed on unoriented grids (see that setting for the dispute about its orientation); its algorithm carries over here, and so do the crash-tolerant $O(\sqrt{n})$ algorithms for oriented grids (Banerjee et al. 2024, 2025). **Not independently checked.** The model, theorem statements and algorithm were read in full, but the proofs were not checked step by step, and three points need care. (1) Theorem 2 bounds the number of rounds in which each agent is active and concludes $T(k) = O(\sqrt{k})$; a settled agent can be recruited again by a later Collect, so the step from active rounds to total time needs an argument along the chain of recruiters, which is not written out. (2) The proof of Lemma 7 assumes that same-labelled settled agents stay in place during the next Spread, while Section 3.5 lets a Collect un-settle every settled agent of equal or lower label that it reaches. (3) Theorem 3 says every agent terminates, but the paper does not say how a settled agent, which may be recruited again, knows that it may stop, and on the infinite grid no whole-grid sweep happens; the result is read here as reaching a dispersed configuration that no

Claude (Anthropic) was used in cataloguing the problems.

agent leaves afterwards. The infinite-grid claim (Section 5) has no separate proof.

16.13.7 Crash faults on oriented grids

Adds + at most crash faults, + local communication, + synchronous VERIFIED

On oriented grids crashes are free for $k = \Omega(n)$: the fault-free $\Theta(\sqrt{n})$ bound survives any number of crashes.

Assumed: graph dynamics: static graph; edge direction: undirected; number of agents: $k \leq n$; identifiers: distinct identifiers (eponymous): agents have pairwise distinct identifiers; topology: grid; grid orientation: oriented grid (consistent North, East, South, West ports); faults: up to f agents may crash (stop forever); communication: local (co-located agents communicate); timing: synchronous rounds. **Not assumed:** randomness: deterministic; node memory: no whiteboards; node identifiers: anonymous nodes; initial configuration: arbitrary (agents on any $\ell \leq k$ nodes).

Measure	Lower bound	Best upper bound	Status
Time	$\Omega(\sqrt{k})$ (Kshemkalyani et al. 2020c)	$O(\sqrt{n})$ (Banerjee et al. 2025)	• polynomial gap
	$\Theta(\sqrt{n})$ for $k = \Omega(n)$; for $k = o(n)$, $O(\sqrt{n})$ versus $\Omega(\sqrt{k})$, a factor $\sqrt{n/k}$. Without faults $O(\sqrt{k})$ is known on oriented grids (Kumar et al. 2026a) (preprint).		
Memory	$\Omega(\log k)$ (Kshemkalyani et al. 2020c)	$O(\log n)$ (Banerjee et al. 2025)	• tight
	$O(\log n)$ versus $\Omega(\log k)$: $\Theta(\log n)$ when $k = n^{\Omega(1)}$.		

Results.

- Kshemkalyani et al. (2020c) [lower bound]: time $\Omega(\sqrt{k})$. *Bottleneck:* distance. *Holds when:* k agents on one node: a grid ball of radius r holds $O(r^2)$ nodes; any algorithm. $\Omega(\sqrt{n})$ when $k = \Omega(n)$.
- Kshemkalyani et al. (2020c) [lower bound]: memory $\Omega(\log k)$. *Bottleneck:* symmetry breaking. *Holds when:* deterministic; all agents have the same memory bound; an initial configuration with co-located agents (rooted suffices); memory counts the identifier.
- Banerjee et al. (2024) [introductory]: time $O(\sqrt{n})$, memory $O(\log n)$. *Assumes:* knows n. Its Theorem 1 (Section 4): $f \leq k \leq n$ crashed agents; the authors remark that the algorithm can be modified to work without knowing n. The same paper treats unoriented grids with crashes (that setting).
- Banerjee et al. (2025) [current best]: time $O(\sqrt{n})$, memory $O(\log n)$. Any $f \leq n$, no knowledge of n; stated for $k = n$ and, by its Remark 2, for every $k \leq n$; optimal in both measures for $k = \Omega(n)$.

Open questions.

Q1. Crash-tolerant dispersion in $O(\sqrt{k})$ rounds on oriented grids for $k = o(n)$; the fault-free case is solved by (Kumar et al. 2026a) (preprint), which names fault tolerance as future work.

16.13.8 Weak Byzantine agents on oriented grids

Adds + at most weak Byzantine faults, + local communication, + synchronous VERIFIED

Time-optimal $O(\sqrt{n})$ rounds on oriented grids for any number of weak Byzantine agents, at the price of $O(n \log n)$ bits.

Assumed: graph dynamics: static graph; edge direction: undirected; number of agents: $k \leq n$; identifiers: distinct identifiers (eponymous): agents have pairwise distinct identifiers; topology: grid; grid orientation: oriented grid (consistent North, East, South, West ports); faults: up to f weak Byzantine agents (arbitrary behaviour, cannot fake identifiers); communication: local (co-located agents communicate); timing: synchronous rounds. **Not assumed:** randomness: deterministic; node memory: no whiteboards; node identifiers: anonymous nodes; initial configuration: arbitrary (agents on any $\ell \leq k$ nodes).

Claude (Anthropic) was used in cataloguing the problems.

Measure	Lower bound	Best upper bound	Status
Time	$\Omega(\sqrt{n})$ (Kshemkalyani et al. 2020c) $\Theta(\sqrt{n})$.	$O(\sqrt{n})$ (Banerjee et al. 2025)	• tight
Memory	$\Omega(\log n)$ (Augustine and Moses Jr. 2018) $O(n \log n)$ versus $\Omega(\log n)$.	$O(n \log n)$ (Banerjee et al. 2025)	• polynomial gap

Results.

- Kshemkalyani et al. (2020c) [lower bound]: time $\Omega(\sqrt{n})$. *Bottleneck:* distance. *Holds when:* $k = n$: the grid has diameter $2\sqrt{n}$; any algorithm. $k = n$.
- Augustine and Moses Jr. (2018) [lower bound]: memory $\Omega(\log n)$. *Bottleneck:* symmetry breaking. *Holds when:* deterministic; all agents have the same memory bound; $k = n$.
- Banerjee et al. (2025) [current best]: time $O(\sqrt{n})$, memory $O(n \log n)$. *Assumes:* $k = n$; $f \leq n$. Introductory and current best.

Open questions.

Q1. Reduce the memory below $O(n \log n)$ bits on grids with weak Byzantine agents.

Q2. Strong Byzantine agents (able to fake identifiers) on grids (Banerjee et al. 2025).

Q3. Weak Byzantine agents on unoriented grids, and on oriented grids with $k = o(n)$ agents in $O(\sqrt{k})$ rounds; fault tolerance is named as future work in (Kumar et al. 2026a).

Notes. Every known grid result with Byzantine agents assumes an oriented grid (Banerjee et al. 2025), so this setting carries the orientation assumption; no result is known on unoriented grids.

16.13.9 Unoriented grids

Adds unoriented, + no faults, + local communication, + synchronous VERIFIED

A $\sqrt{n} \times \sqrt{n}$ anonymous grid without a consistent compass; the diameter gives an $\Omega(\sqrt{k})$ bound, and whether local communication reaches it for small k is open here, while on oriented grids it does.

Assumed: graph dynamics: static graph; edge direction: undirected; number of agents: $k \leq n$; identifiers: distinct identifiers (eponymous): agents have pairwise distinct identifiers; topology: grid; faults: no faults; communication: local (co-located agents communicate); timing: synchronous rounds. **Not assumed:** randomness: deterministic; grid orientation: unoriented (no consistent compass); node memory: no whiteboards; node identifiers: anonymous nodes; initial configuration: arbitrary (agents on any $\ell \leq k$ nodes).

Statement for this setting. The graph is a $\sqrt{n} \times \sqrt{n}$ grid with $\Delta = 4$, and the ports at each node are labelled arbitrarily: an agent has no consistent compass (an **unoriented** grid). Grids with consistent North, East, South and West ports are the setting *Oriented grids*.

Measure	Lower bound	Best upper bound	Status
Time	$\Omega(\sqrt{k})$ (Kshemkalyani et al. 2020c)	$O(\min(k, \sqrt{n}))$ (Kshemkalyani et al. 2020c)	• polynomial gap
	$O(\min(k, \sqrt{n}))$ (Kshemkalyani et al. 2020c) versus $\Omega(\sqrt{k})$: tight for $k = \Omega(n)$; a factor $\sqrt{k}$ apart for $k \leq \sqrt{n}$ and $\sqrt{n/k}$ above, at most $n^{1/4}$. If (Kshemkalyani et al. 2020c) is read as oriented, the best here is $O(\sqrt{n})$ (Banerjee et al. 2024), a factor $\sqrt{n/k}$ apart. On oriented grids the gap is closed (Kumar et al. 2026a).		
Memory	$\Omega(\log k)$ (Kshemkalyani et al. 2020c) $\Theta(\log k)$ with (Kshemkalyani et al. 2020c);	$O(\log k)$ (Kshemkalyani et al. 2020c) $O(\log n)$ bits with (Banerjee et al. 2024).	• tight

Claude (Anthropic) was used in cataloguing the problems.

Results.

- Kshemkalyani et al. (2020c) [lower bound]: time $\Omega(\sqrt{k})$. *Bottleneck:* distance. *Holds when:* k agents on one node: a grid ball of radius r holds $O(r^2)$ nodes; any algorithm; holds on oriented and unoriented grids: the argument does not use port labels. Any k agents on one node must spread over a region of diameter $\Omega(\sqrt{k})$; $\Omega(\sqrt{n})$ when $k = \Omega(n)$.
- Kshemkalyani et al. (2020c) [lower bound]: memory $\Omega(\log k)$. *Bottleneck:* symmetry breaking. *Holds when:* deterministic; all agents have the same memory bound; an initial configuration with co-located agents (rooted suffices); memory counts the identifier.
- Kshemkalyani et al. (2020c) [current best]: time $O(\min(k, \sqrt{n}))$, memory $O(\log k)$. *Assumes:* knows n and k (arXiv version). Introductory and current best for square grids with local communication; optimal in both measures when $k = \Omega(n)$. Placed on unoriented grids following the arXiv text, which says the agents have no consistent compass; later papers call the setting oriented (see notes). With global communication: $O(\sqrt{k})$ rounds and $O(\log k)$ bits for every k, as reported by (Banerjee et al. 2024) and (Kumar et al. 2026a); that result is not in the arXiv version.
- Banerjee et al. (2024) [also]: time $O(\sqrt{n})$, memory $O(\log n)$. *Assumes:* knows n. The first paper to state an unoriented grid explicitly (its Theorem 6); no knowledge of k, and the authors note the algorithm can be adapted to unknown n. If (Kshemkalyani et al. 2020c) assumes an oriented grid, this is the best result on unoriented grids. Crash-tolerant versions are under the crash-fault branch.
- Himani and Pandit (2026a) [also]: time $O(\sqrt{n})$, memory $O(\log n)$. *Assumes:* triangular grid. No knowledge of n; the agents compute $\sqrt{n}$ themselves.

Open questions.

Q1. Can $k \le n$ agents disperse on an **unoriented** grid in $O(\sqrt{k})$ rounds with **local** communication? On oriented grids $O(\sqrt{k})$ rounds are reached with local communication (Kumar et al. 2026a) (preprint), and with global communication by (Kshemkalyani et al. 2020c) (as reported); asked in (Banerjee et al. 2025), and for unoriented grids in (Kumar et al. 2026a).

Q2. A lower bound in terms of k for $k = o(\sqrt{n})$ with $O(\log n)$ memory on unoriented grids (Banerjee et al. 2024).

Q3. Unoriented grids with no knowledge of n and with faults, matching the fault-free bounds (Banerjee et al. 2024).

Notes. With global communication $O(\sqrt{k})$ rounds were known on grids, against $O(\min(k, \sqrt{n}))$ with local communication (Kshemkalyani et al. 2020c). On oriented grids Kumar, Molla, Pattanayak and Pramanick reach $O(\sqrt{k})$ rounds with local communication (Kumar et al. 2026a) (preprint), so there global communication is no longer known to help; on unoriented grids the question is open. **Orientation of (Kshemkalyani et al. 2020c).** The arXiv version (1812.05352v3) says "robots do not have access to a consistent compass to determine which direction to follow to reach boundary nodes", and its port labels at different nodes are uncorrelated; the result is placed here on that text. Banerjee, Kumar and Molla (Banerjee et al. 2024, 2025) and Kumar et al. (Kumar et al. 2026a) describe the same setting as an oriented ("planar") grid. The WALCOM 2020 text was not checked. If it assumes an oriented grid, the result belongs to *Oriented grids*, where (Kumar et al. 2026a) supersedes it, and the best result here is $O(\sqrt{n})$ rounds with $O(\log n)$ bits (Banerjee et al. 2024). Rectangular $x \times y$ grids with x, y known: $O(\min(k, \max(x, y)))$ rounds with $\Theta(\log k)$ bits (arXiv version of Kshemkalyani et al. 2020c).

16.13.10 Crash faults on unoriented grids

Adds unoriented, + at most crash faults, + local communication, + synchronous VERIFIED

Without a consistent compass the only crash-tolerant grid algorithm takes $O(\sqrt{n} \log n)$ rounds and

$O(\sqrt{n}\log n)$ bits, against $O(\sqrt{n})$ rounds and $O(\log n)$ bits on oriented grids.

Assumed: graph dynamics: static graph; edge direction: undirected; number of agents: $k \le n$; identifiers: distinct identifiers (eponymous): agents have pairwise distinct identifiers; topology: grid; faults: up to f agents may crash (stop forever); communication: local (co-located agents communicate); timing: synchronous rounds. **Not assumed:** randomness: deterministic; grid orientation: unoriented (no consistent compass); node memory: no whiteboards; node identifiers: anonymous nodes; initial configuration: arbitrary (agents on any $\ell \le k$ nodes).

Measure	Lower bound	Best upper bound	Status
Time	$\Omega(\sqrt{k})$ (Kshemkalyani et al. 2020c) $O(\sqrt{n}\log n)$ versus $\Omega(\sqrt{k})$: a $\log n$ factor for $k = \Omega(n)$, a factor $\sqrt{n/k}\log n$ for smaller k.	$O(\sqrt{n}\log n)$ (Banerjee et al. 2024)	• polynomial gap
Memory	$\Omega(\log k)$ (Kshemkalyani et al. 2020c) $O(\sqrt{n}\log n)$ versus $\Omega(\log k)$, a factor $\sqrt{n}$.	$O(\sqrt{n}\log n)$ (Banerjee et al. 2024)	• polynomial gap

Results.

- Kshemkalyani et al. (2020c) [lower bound]: time $\Omega(\sqrt{k})$. *Bottleneck:* distance. *Holds when:* k agents on one node: a grid ball of radius r holds $O(r^2)$ nodes; any algorithm. $\Omega(\sqrt{n})$ when $k = \Omega(n)$.
- Kshemkalyani et al. (2020c) [lower bound]: memory $\Omega(\log k)$. *Bottleneck:* symmetry breaking. *Holds when:* deterministic; all agents have the same memory bound; an initial configuration with co-located agents (rooted suffices); memory counts the identifier.
- Banerjee et al. (2024) [current best]: time $O(\sqrt{n}\log n)$, memory $O(\sqrt{n}\log n)$. *Assumes:* knows n. Introductory and current best (its Theorem 12, Section 6): any number of crashed agents.

Open questions.

Q1. Unoriented grids with crashes: close the gap between $O(\sqrt{n}\log n)$ rounds with $O(\sqrt{n}\log n)$ bits and the fault-free $O(\sqrt{n})$ rounds with $O(\log n)$ bits (Banerjee et al. 2024).

16.13.11 Rings and paths

Adds + local communication, + synchronous VERIFIED

Degree two everywhere; both measures are settled.

Assumed: graph dynamics: static graph; edge direction: undirected; number of agents: $k \le n$; identifiers: distinct identifiers (eponymous): agents have pairwise distinct identifiers; topology: ring or path; faults: no faults; communication: local (co-located agents communicate); timing: synchronous rounds. **Not assumed:** randomness: deterministic; grid orientation: unoriented (no consistent compass); node memory: no whiteboards; node identifiers: anonymous nodes; initial configuration: arbitrary (agents on any $\ell \le k$ nodes).

Measure	Lower bound	Best upper bound	Status
Time	$\Omega(k)$ (Kshemkalyani and Ali 2019) $O(k)$ versus $\Omega(k)$.	$O(k)$ (Kshemkalyani et al. 2025b)	• tight
Memory	$\Omega(\log k)$ (Augustine and Moses Jr. 2018) $O(\log k)$ versus $\Omega(\log k)$.	$O(\log k)$ (Kshemkalyani et al. 2025b)	• tight

Results.

- Kshemkalyani and Ali (2019) [lower bound]: time $\Omega(k)$. *Bottleneck:* distance. *Holds when:* all k agents on one node: the farthest target is $k/2$ away; any algorithm. $\Omega(n)$ when $k = n$, since $D = \Theta(n)$ (Augustine and Moses Jr. 2018).
- Augustine and Moses Jr. (2018) [lower bound]: memory $\Omega(\log k)$. *Bottleneck:* symmetry breaking. *Holds when:* deterministic; all agents have the same memory bound; an initial configuration with co-located agents (rooted suffices); memory counts the identifier.

Claude (Anthropic) was used in cataloguing the problems.

- Augustine and Moses Jr. (2018) [introductory]: time $O(n)$, memory $O(\log n)$. *Assumes:* $k = n$; knows n. One algorithm for paths, rings and trees; optimal in both measures on paths and rings for $k = n$.
- Kshemkalyani et al. (2025b) [current best]: time $O(k)$, memory $O(\log k)$. Inherited from arbitrary graphs; $O(\log(k+\Delta)) = O(\log k)$ because $\Delta = 2$.

Open questions.

Q1. Anonymous randomized agents disperse from a **rooted** ring configuration with $O(1)$ bits in $O(n)$ rounds (Molla and Moses Jr. 2019). From an **arbitrary** configuration the known algorithm never terminates; is $O(1)$-bit dispersion with termination possible on rings?

Notes. The ring is the testbed for every harder variant: silent agents, Byzantine agents and dynamic edges all have ring-specific results, catalogued under those branches.

16.13.12 Silent agents on rings

Adds no communication, + synchronous VERIFIED

On an oriented ring, silent agents from arbitrary configurations disperse in optimal $\Theta(k + \log L)$ rounds.

Assumed: graph dynamics: static graph; edge direction: undirected; number of agents: $k \le n$; identifiers: distinct identifiers (eponymous): agents have pairwise distinct identifiers; topology: ring or path; faults: no faults; timing: synchronous rounds. **Not assumed:** randomness: deterministic; grid orientation: unoriented (no consistent compass); communication: none (silent agents; only see how many agents are co-located); node memory: no whiteboards; node identifiers: anonymous nodes; initial configuration: arbitrary (agents on any $\ell \le k$ nodes).

Measure	Lower bound	Best upper bound	Status
Time	$\Omega(k + \log L)$ (Das et al. 2026a) $\Theta(k + \log L)$.	$O(k + \log L)$ (Das et al. 2026a)	• tight
Memory	$\Omega(\log L)$ (Gorain et al. 2024) $\Theta(\log L)$.	$O(\log L)$ (Das et al. 2026a)	• tight

Results.

- Das et al. (2026a) [lower bound]: time $\Omega(k + \log L)$. *Bottleneck:* timing is the only channel. *Holds when:* no communication: only alone / count-change sensing; identifiers in $[0, L]$, unknown to the other agents.
- Gorain et al. (2024) [lower bound]: memory $\Omega(\log L)$. *Bottleneck:* encoding the identifier. *Holds when:* additional bits beyond the identifier; $\Delta = 2$, so only the $\log L$ term remains. $\Delta = 2$.
- Das et al. (2026a) [current best]: time $O(k + \log L)$, memory $O(\log L)$. *Assumes:* oriented ring. Introductory and current best; arbitrary initial configurations.

Open questions.

Q1. Unoriented rings, and rings where agents cannot detect count changes.

16.13.13 Weak Byzantine agents on rings

Adds + at most weak Byzantine faults, + local communication, + synchronous VERIFIED

Optimal $O(n)$ time and $O(\log n)$ bits are possible on rings when fewer than $n/17$ agents are Byzantine and f is known.

Assumed: graph dynamics: static graph; edge direction: undirected; number of agents: $k \le n$; identifiers: distinct identifiers (eponymous): agents have pairwise distinct identifiers; topology: ring or path; faults: up to f weak Byzantine agents (arbitrary

behaviour, cannot fake identifiers); communication: local (co-located agents communicate); timing: synchronous rounds. **Not assumed:** randomness: deterministic; grid orientation: unoriented (no consistent compass); node memory: no whiteboards; node identifiers: anonymous nodes; initial configuration: arbitrary (agents on any $\ell \leq k$ nodes).

Measure	Lower bound	Best upper bound	Status
Time	$\Omega(n)$ (Molla et al. 2021a) $\Theta(n)$ under the four assumptions; $O(n^2)$ without them.	$O(n)$ (Molla et al. 2021a)	• tight
Memory	$\Omega(\log n)$ (Augustine and Moses Jr. 2018) $\Theta(\log n)$.	$O(\log n)$ (Molla et al. 2021a)	• tight

Results.

- Molla et al. (2021a) [lower bound]: time $\Omega(n)$. *Bottleneck:* distance. *Holds when:* $k = n$: the ring has diameter $n/2$; any algorithm. Diameter $n/2$, $k = n$.
- Augustine and Moses Jr. (2018) [lower bound]: memory $\Omega(\log n)$. *Bottleneck:* symmetry breaking. *Holds when:* deterministic; all agents have the same memory bound; $k = n$.
- Molla et al. (2021a) [current best]: time $O(n)$, memory $O(\log n)$. *Assumes:* $k = n$; knows n and f; $f \leq \lfloor (n-4)/17 \rfloor$; identifiers in $[1, n]$. Introductory and current best. Without those assumptions: $O(n^2)$ rounds with $O(\log n)$ bits for any $f \leq n - 1$ ($O(fn)$ if f is known), or n rounds with $O(n \log n)$ bits. From a rooted configuration, $n - 1$ rounds with $O(\log n)$ bits for any $f \leq k - 1$ without knowing n or f.

Open questions.

Q1. Keep $O(n)$ time and $O(\log n)$ bits while dropping any of: identifiers in $[1, n]$, knowledge of f, the bound $f \leq (n-4)/17$, the follow primitive (Molla et al. 2021a).

Q2. Asynchronous Byzantine dispersion on rings (Molla et al. 2021a).

16.13.14 Trees

Adds + tree, + no faults, + local communication, + synchronous VERIFIED

No cycles, so a DFS never needs to test for revisits; the difficulty is spreading agents that start far apart across long branches.

Assumed: graph dynamics: static graph; edge direction: undirected; number of agents: $k \leq n$; identifiers: distinct identifiers (eponymous): agents have pairwise distinct identifiers; topology: tree; faults: no faults; communication: local (co-located agents communicate); timing: synchronous rounds. **Not assumed:** randomness: deterministic; grid orientation: unoriented (no consistent compass); node memory: no whiteboards; node identifiers: anonymous nodes; initial configuration: arbitrary (agents on any $\ell \leq k$ nodes).

Measure	Lower bound	Best upper bound	Status
Time	$\Omega(D^2)$ (Disser et al. 2020) $O(n)$ for $k = n$ versus a lower bound of $\Omega(D^2)$ that is proved only for $D \approx \sqrt{\log n}$; in general the gap is polynomial.	$O(k)$ (Kshemkalyani et al. 2025b)	• polynomial gap
Memory	$\Omega(\log k)$ (Augustine and Moses Jr. 2018) $O(\log(k + \Delta))$ versus $\Omega(\log k)$.	$O(\log(k + \Delta))$ (Kshemkalyani et al. 2025b)	• near-optimal

Results.

- Disser et al. (2020) [lower bound]: time $\Omega(D^2)$. *Bottleneck:* collaborative tree exploration. *Holds when:* $k = n$: every node must be reached; deterministic; **existential in** D: the hard tree has height $D = \lceil \sqrt{\log n} \rceil$, so the bound is about $\log n$ rounds; it is not shown for larger heights; proved for collaborative exploration with labelled nodes and global communication, so it holds for dispersion (Augustine and Moses Jr. 2018; Kshemkalyani et al. 2022).

Claude (Anthropic) was used in cataloguing the problems.

Transferred from collaborative tree exploration (Theorem 2.5 of Disser et al. 2020) by (Augustine and Moses Jr. 2018); re-proven for the global model in (Kshemkalyani et al. 2022). In addition $\Omega(k)$ (Kshemkalyani and Ali 2019).

- Augustine and Moses Jr. (2018) [lower bound]: memory $\Omega(\log k)$. *Bottleneck:* symmetry breaking. *Holds when:* deterministic; all agents have the same memory bound; an initial configuration with co-located agents (rooted suffices); memory counts the identifier.
- Augustine and Moses Jr. (2018) [introductory]: time $O(n)$, memory $O(\log n)$. *Assumes:* $k = n$; knows n. Memory-optimal; the authors ask whether $o(n)$ rounds are possible.
- Kshemkalyani et al. (2025b) [current best]: time $O(k)$, memory $O(\log(k + \Delta))$. Inherited from arbitrary graphs. For $k = n$ this is $O(n)$, no better than 2018.

Open questions.

Q1. (Augustine and Moses Jr. 2018) asked for $o(n)$-round dispersion on arbitrary trees. Precisely: for $k = n$ and an arbitrary initial configuration, is there an algorithm whose time is bounded by a function of D alone, ideally $O(D^2)$ as in the rooted case? No tree-specific algorithm has appeared since 2018; the linear-time general algorithms give only $O(n)$.

Q2. With global communication, trees admit $O(D \cdot \max(D, k))$ rounds with $O(\max(D, \Delta \log k))$ bits (arXiv version of Kshemkalyani et al. 2020b); whether the $\max(D, k)$ factor can be reduced to D, and whether local communication suffices, is open.

Notes. Anonymous randomized agents from a rooted tree configuration: $O(n)$ rounds with $O(\log \Delta)$ bits (Molla and Moses Jr. 2019).

16.13.15 Rooted trees

Adds + rooted start **to** Trees (p. 277) VERIFIED

All agents start at one node of a tree; the time is governed by the diameter, as in collaborative tree exploration.

Assumed: graph dynamics: static graph; edge direction: undirected; number of agents: $k \leq n$; identifiers: distinct identifiers (eponymous): agents have pairwise distinct identifiers; topology: tree; faults: no faults; communication: local (co-located agents communicate); timing: synchronous rounds; initial configuration: rooted (all agents on one node). **Not assumed:** randomness: deterministic; grid orientation: unoriented (no consistent compass); node memory: no whiteboards; node identifiers: anonymous nodes.

Measure	Lower bound	Best upper bound	Status
Time	$\Omega(D^2)$ (Disser et al. 2020) $O(D^2)$ is matched only for trees of height about $\sqrt{\log n}$; for larger heights only $\Omega(D)$ is known, a factor D.	$O(D^2)$ (Augustine and Moses Jr. 2018)	• polynomial gap
Memory	$\Omega(\log k)$ (Augustine and Moses Jr. 2018) $\Theta(\log n)$ for $k = n$.	$O(\log n)$ (Augustine and Moses Jr. 2018)	• tight

Results.

- Disser et al. (2020) [lower bound]: time $\Omega(D^2)$. *Bottleneck:* collaborative tree exploration. *Holds when:* $k = n$: every node must be reached; deterministic; **existential in** D: the hard tree has height $D = \lceil\sqrt{\log n}\rceil$, so the bound is about $\log n$ rounds; it is not shown for larger heights; proved for collaborative exploration with labelled nodes and global communication, so it holds for dispersion (Augustine and Moses Jr. 2018; Kshemkalyani et al. 2022). Transferred from collaborative tree exploration (Theorem 2.5 of Disser et al. 2020) by (Augustine and Moses Jr. 2018); re-proven for the global model in (Kshemkalyani et al. 2022).
- Augustine and Moses Jr. (2018) [lower bound]: memory $\Omega(\log k)$. *Bottleneck:* symmetry breaking. *Holds when:* deterministic; all agents have the same memory bound; an initial configuration with co-located agents (rooted suffices); memory counts the identifier.

Claude (Anthropic) was used in cataloguing the problems.

- Augustine and Moses Jr. (2018) [current best]: time $O(D^2)$, memory $O(\log n)$. *Assumes:* $k = n$; knows n. Introductory and current best; the full (arXiv) version improves the memory from $O(\log n + \Delta)$ in the conference version to $O(\log n)$.
- Kshemkalyani et al. (2025b) [also]: time $O(k)$, memory $O(\log(k + \Delta))$. Inherited; better than $O(D^2)$ only when $k = o(D^2)$.

Open questions.

Q1. For $k < n$ the lower bound $\Omega(D^2)$ no longer applies (only k nodes must be reached). Whether $O(\min(k, D^2))$ time is achievable, and whether it is necessary, has not been addressed.

Q2. For $k = n$ agents rooted on a tree of height D much larger than $\sqrt{\log n}$, is $\Omega(D^2)$ time necessary? The known bound is existential in D (Disser et al. 2020).

16.13.16 Crash faults

Adds any topology, + at most crash faults, + local communication, + synchronous VERIFIED

Up to $f \leq k$ agents may crash at any time; a crashed agent disappears with everything it knew, and the surviving $k - f$ agents must still end on distinct nodes.

Assumed: graph dynamics: static graph; edge direction: undirected; number of agents: $k \leq n$; identifiers: distinct identifiers (eponymous): agents have pairwise distinct identifiers; faults: up to f agents may crash (stop forever); communication: local (co-located agents communicate); timing: synchronous rounds. **Not assumed:** randomness: deterministic; topology: arbitrary connected graph; grid orientation: unoriented (no consistent compass); node memory: no whiteboards; node identifiers: anonymous nodes; initial configuration: arbitrary (agents on any $\ell \leq k$ nodes).

Statement for this setting. An adversary may crash up to $f \leq k$ agents at arbitrary times. A crashed agent loses its memory and can no longer move or be seen by others. Dispersion is achieved when the $k - f$ non-faulty agents occupy $k - f$ distinct nodes. ℓ is the number of initially occupied nodes.

Measure	Lower bound	Best upper bound	Status
Time	$\Omega(k)$ (Kshemkalyani and Ali 2019)	$O(\min(m, k\Delta))$ (Pattanayak et al. 2023)	• polynomial gap
	$O(\min(m, k\Delta))$ versus $\Omega(k)$: a factor Δ, as in the fault-free DFS era; the fault-free $O(k)$ techniques (oscillating settled agents) have not been made crash-tolerant.		
Memory	$\Omega(\log k)$ (Augustine and Moses Jr. 2018)	$O(k \log(k + \Delta))$ (Pattanayak et al. 2023)	• near-optimal
	$O(\log(k + \Delta))$ bits are enough but only with an extra factor $(\ell + f)$ in time; the fast algorithm uses $O(k \log(k + \Delta))$.		

Results.

- Kshemkalyani and Ali (2019) [lower bound]: time $\Omega(k)$. *Bottleneck:* distance. *Holds when:* fault-free bound ($f = 0$); worst case over inputs: a path with all k agents on one endpoint; one edge per round; holds for randomized algorithms, unbounded memory and global communication. Fault-free lower bound; no crash-specific lower bound is known.
- Augustine and Moses Jr. (2018) [lower bound]: memory $\Omega(\log k)$. *Bottleneck:* symmetry breaking. *Holds when:* deterministic; all agents have the same memory bound; an initial configuration with co-located agents (rooted suffices); memory counts the identifier.
- Pattanayak et al. (2021b) [introductory]: time $O(f \cdot \min(m, k\Delta))$, memory $O(\log(k + \Delta))$. *Assumes:* rooted. First crash-tolerant dispersion on arbitrary graphs: a DFS that restarts whenever the agent carrying the traversal state crashes.
- Chand et al. (2025a) [also]: time $O(k^2)$, memory $O(\log(k + \Delta))$. *Assumes:* rooted. Independent of f. For arbitrary configurations: $O((f + \ell) \cdot \min(m, k\Delta, k^2))$ rounds with $O(\log(k + \Delta))$ bits when k, f, Δ, ℓ and m are known, and $O(k^3)$ when only k is known.

Claude (Anthropic) was used in cataloguing the problems.

- Pattanayak et al. (2023) [current best]: time $O(\min(m, k\Delta))$, memory $O(k \log(k + \Delta))$. Arbitrary configurations, no knowledge of m, Δ, ℓ, k or f, time independent of f. A second algorithm trades memory for time: $O(\min(m, k\Delta) \cdot (\ell + f))$ rounds with $O(\log(k + \Delta))$ bits.

Open questions.

Q1. Is $O(k)$ time achievable with crash faults, or does fault tolerance cost a factor of Δ? No lower bound beyond the fault-free $\Omega(k)$ is known, even with $O(\log(k + \Delta))$ memory (Chand et al. 2025a).

Q2. Simultaneously $O(\min(m, k\Delta))$ (or better) time and $O(\log(k+\Delta))$ memory for arbitrary configurations, without knowledge of f, ℓ, Δ or m (Chand et al. 2025a; Pattanayak et al. 2023).

Q3. Crash faults under asynchrony: no result exists.

Notes. On 1-interval connected dynamic graphs with global communication, crashes cost nothing: $O(k-f)$ rounds with $\Theta(\log k)$ bits (Kshemkalyani et al. 2020a). The directed-graph algorithms of (Italiano et al. 2025) also extend to crash faults (bounds not transcribed here).

16.13.17 Weak Byzantine agents

Adds any topology, + at most weak Byzantine faults, + local communication, + synchronous VERIFIED

Up to f agents behave arbitrarily and may lie, but cannot fake their identifiers; the non-faulty agents must still settle on distinct nodes and terminate.

Assumed: graph dynamics: static graph; edge direction: undirected; number of agents: $k \le n$; identifiers: distinct identifiers (eponymous): agents have pairwise distinct identifiers; faults: up to f weak Byzantine agents (arbitrary behaviour, cannot fake identifiers); communication: local (co-located agents communicate); timing: synchronous rounds. **Not assumed:** randomness: deterministic; topology: arbitrary connected graph; grid orientation: unoriented (no consistent compass); node memory: no whiteboards; node identifiers: anonymous nodes; initial configuration: arbitrary (agents on any $\ell \le k$ nodes).

Statement for this setting. Up to f agents are controlled by an adversary that knows the algorithm and the honest agents' states; they may move and communicate arbitrarily but cannot change their identifiers. The goal is a configuration in which every node holds at most one **non-Byzantine** agent, with all non-Byzantine agents terminated. Results below take $k = n$ and assume the agents know n. $X(n)$ denotes the time to explore any n-node graph, $\tilde{O}(n^5)$.

Measure	Lower bound	Best upper bound	Status
Time	$\Omega(k)$ (Kshemkalyani and Ali 2019); *impossible* (Molla et al. 2021b) Polynomial, but $\tilde{O}(n^9)$ versus $\Omega(n)$; the bottleneck is Byzantine gathering.	$\tilde{O}(n^9)$ (Molla et al. 2021b)	• polynomial gap
Memory	$\Omega(\log k)$ (Augustine and Moses Jr. 2018) Memory is not bounded in the general-graph algorithms.	—	• unknown

Results.

- Kshemkalyani and Ali (2019) [lower bound]: time $\Omega(k)$. *Bottleneck:* distance. *Holds when:* fault-free bound ($f = 0$); a path with all k agents on one endpoint. Fault-free bound; no Byzantine-specific time lower bound is known.
- Molla et al. (2021b) [lower bound]: time *impossible*. *Bottleneck:* indistinguishable Byzantine agents. *Holds when:* $\lceil k/n \rceil > \lceil (k - f)/n \rceil$; weak Byzantine agents; deterministic; even with known n, k and f. The Byzantine agents can behave exactly like honest agents and occupy nodes, so the honest agents cannot reach the balanced target.
- Augustine and Moses Jr. (2018) [lower bound]: memory $\Omega(\log k)$. *Bottleneck:* symmetry breaking. *Holds when:* deterministic; all agents have the same memory bound; an initial configuration with co-located agents (rooted suffices); memory counts the identifier.

Claude (Anthropic) was used in cataloguing the problems.

- Molla et al. (2021b) [current best]: time $\tilde{O}(n^9)$. *Assumes:* $k = n$; knows n; $f \leq \lfloor n/2 \rfloor - 1$. Introductory and current best on general graphs. Precisely $O(n^4 \cdot |\Lambda_{good}| \cdot X(n))$ rounds for arbitrary configurations, $|\Lambda_{good}|$ the length of the largest honest identifier; $O(n^4)$ if the agents start gathered; $O(n^3)$ gathered with $f \leq \lfloor n/3 \rfloor - 1$; $O((f + |\Lambda|) \cdot X(n))$ for $f = O(\sqrt{n})$. Tolerates $f \leq n - 1$ in polynomial time on graphs isomorphic to their quotient graph. Memory per agent is not analysed except for that last algorithm ($O(m \log n)$ bits).

Open questions.

Q1. Faster Byzantine dispersion on general graphs: the time is dominated by Byzantine gathering, and the authors ask whether building a map before settling is necessary (Molla et al. 2021b).

Q2. Tolerating more than $n/2$ weak Byzantine agents on general graphs (possible only on the restricted class so far) (Molla et al. 2021b).

Q3. Byzantine dispersion with bounded memory per agent, on trees, with $k < n$, or under asynchrony: nothing is known (Molla et al. 2021a; Chand et al. 2025a).

Notes. Byzantine dispersion was introduced on rings (Molla et al. 2021a) and then on general graphs (Molla et al. 2021b); no general-graph improvement has appeared since 2021.

16.13.18 Strong Byzantine agents

Adds any topology, any faults, + local communication, + synchronous VERIFIED

Byzantine agents can also fake identifiers; polynomial time is known only from a gathered start, or with a trusted shepherd agent.

Assumed: graph dynamics: static graph; edge direction: undirected; number of agents: $k \leq n$; identifiers: distinct identifiers (eponymous): agents have pairwise distinct identifiers; communication: local (co-located agents communicate); timing: synchronous rounds. **Not assumed:** randomness: deterministic; topology: arbitrary connected graph; grid orientation: unoriented (no consistent compass); faults: up to f strong Byzantine agents (arbitrary behaviour, can fake identifiers); node memory: no whiteboards; node identifiers: anonymous nodes; initial configuration: arbitrary (agents on any $\ell \leq k$ nodes).

Measure	Lower bound	Best upper bound	Status
Time	$\Omega(k)$ (Kshemkalyani and Ali 2019)	$O(X(n) + n^3)$ (Moses Jr. and Redlich 2024)	• polynomial gap
	Polynomial only with a gathered start or a shepherd; exponential otherwise.		
Memory	none known	—	• unknown

Results.

- Kshemkalyani and Ali (2019) [lower bound]: time $\Omega(k)$. *Bottleneck:* distance. *Holds when:* fault-free bound ($f = 0$). Fault-free bound.
- Molla et al. (2021b) [introductory]: time $O(n^3)$. *Assumes:* $k = n$; knows n; $f \leq \lfloor n/4 \rfloor - 1$; gathered start. From an arbitrary configuration the time is exponential in n and f must be known.
- Moses Jr. and Redlich (2024) [current best]: time $O(X(n) + n^3)$. *Assumes:* one trusted shepherd agent; $f < \lfloor (k-1)/2 \rfloor$; shepherd knows n and k. A single never-Byzantine agent recognisable by others makes arbitrary-configuration strong-Byzantine dispersion polynomial, $O(n^3)$ if gathered; also handles nodes with capacities. Memory is unbounded in this analysis. Without a shepherd, $X(n)$ rounds for $f < k$ when the total node capacity exceeds $zf + k - f$, z the number of nodes of non-zero capacity.

Open questions.

Q1. Polynomial-time strong-Byzantine dispersion from arbitrary configurations without a trusted agent, and the largest tolerable f.

Claude (Anthropic) was used in cataloguing the problems.

Q2. Asynchronous versions: a slow agent cannot be distinguished from a silent Byzantine one (Moses Jr. and Redlich 2024).

16.13.19 Anonymous agents with randomness

Adds anonymous, + private coins, + no faults, + local communication, + synchronous VERIFIED

Agents have no identifiers but private coins. Both axes change together: identical deterministic agents on one node can never behave differently, so randomness is needed to break symmetry.

Assumed: graph dynamics: static graph; edge direction: undirected; number of agents: $k \leq n$; randomness: randomized (agents have private coins); faults: no faults; communication: local (co-located agents communicate); timing: synchronous rounds. **Not assumed:** identifiers: anonymous: all agents are identical; topology: arbitrary connected graph; grid orientation: unoriented (no consistent compass); node memory: no whiteboards; node identifiers: anonymous nodes; initial configuration: arbitrary (agents on any $\ell \leq k$ nodes).

GRAPHS

Statement for this setting. Agents are identical and run a randomized algorithm; deterministic algorithms are impossible for anonymous agents (Das et al. 2023). Memory is measured without an identifier, so bounds below $\log k$ become possible. Results hold with high probability unless stated otherwise.

Measure	Lower bound	Best upper bound	Status
Time	$\Omega(k)$ (Kshemkalyani and Ali 2019)	$O(k^2)$ (Das et al. 2023)	• polynomial gap
	$\Theta(k^2)$ versus $\Omega(k)$ for rooted configurations; for arbitrary configurations only the cover time of G, up to $O(nm)$, without termination.		
Memory	$\Omega(\log \Delta)$ (Molla and Moses Jr. 2019)	$O(\log \Delta)$ (Das et al. 2023)	• tight
	$\Theta(\log \Delta)$ for rooted configurations.		

Results.

- Kshemkalyani and Ali (2019) [lower bound]: time $\Omega(k)$. *Bottleneck:* distance. *Holds when:* a path with all k agents on one endpoint; no identifiers needed; holds for randomized algorithms. The path argument does not use identifiers.
- Molla and Moses Jr. (2019) [lower bound]: memory $\Omega(\log \Delta)$. *Bottleneck:* port addressing. *Holds when:* all agents have the same memory bound; an agent with b bits can choose among at most 2^b ports; worst case over graphs: a node of degree Δ whose ports lead to the free nodes; holds for randomized algorithms. For any randomized algorithm when all agents have the same amount of memory.
- Molla and Moses Jr. (2019) [introductory]: time $O(m)$, memory $O(\max(\log \Delta, \log D))$. *Assumes:* rooted. Local leader election by coin flips replaces identifiers. Rooted graphs: $O(m)$ rounds with $O(\max(\log \Delta, \log D))$ or $O(\Delta)$ bits; rooted rings with $O(1)$ bits and rooted trees with $O(\log \Delta)$ bits in $O(n)$ rounds. Arbitrary configurations ($k = n$): a random-walk algorithm with $O(\log \Delta)$ bits disperses within the cover time of G but the agents never terminate.
- Das et al. (2023) [current best]: time $O(k^2)$, memory $O(\log \Delta)$. *Assumes:* rooted. Memory matches the $\Omega(\log \Delta)$ lower bound; the agents terminate. The time is $\Theta(k^2)$ in the worst case for this algorithm.

Open questions.

Q1. Close the gap between $\Omega(k)$ and $\Theta(k^2)$ rounds for anonymous agents from a rooted configuration with $O(\log \Delta)$ bits (Das et al. 2023).

Q2. Can anonymous randomized agents disperse from an **arbitrary** configuration and then stop, rather than staying active forever? (Molla and Moses Jr. 2019; Das et al. 2023)

Q3. On rings the memory bound is $O(1)$; is $O(1)$-bit dispersion with termination possible on rings from arbitrary configurations?

Claude (Anthropic) was used in cataloguing the problems.

Notes. The deterministic memory bound $\Omega(\log k)$ comes entirely from storing an identifier. Removing identifiers and adding coins lowers the bound to $\Omega(\log \Delta)$, achieved for rooted configurations.

16.13.20 More agents than nodes

Adds any k, + private coins, + no faults, + local communication, + synchronous VERIFIED

With $k > n$ the target is a balanced configuration: at most $\lceil k/n \rceil$ agents per node. Only randomized anonymous results exist.

Assumed: graph dynamics: static graph; edge direction: undirected; randomness: randomized (agents have private coins); faults: no faults; communication: local (co-located agents communicate); timing: synchronous rounds. **Not assumed:** number of agents: any number of agents, possibly $k > n$; identifiers: anonymous: all agents are identical; topology: arbitrary connected graph; grid orientation: unoriented (no consistent compass); node memory: no whiteboards; node identifiers: anonymous nodes; initial configuration: arbitrary (agents on any $\ell \leq k$ nodes).

Statement for this setting. $k > n$ agents must reach a configuration in which every node hosts at most $\lceil k/n \rceil$ agents (sometimes called k-dispersion). Agents do not know n.

Measure	Lower bound	Best upper bound	Status
Time	none known	$O(m+k)$ (Molla and Moses Jr. 2019)	● unknown
	The $\Omega(k)$ path argument of (Kshemkalyani and Ali 2019) is stated for $k \leq n$; no time lower bound for $k > n$ is published.		
Memory	$\Omega(\log \Delta)$ (Molla and Moses Jr. 2019)	$O(\Delta)$ (Molla and Moses Jr. 2019)	● polynomial gap
	$O(\Delta)$ or $O(\max\{\log \Delta, \log D\})$ bits on general rooted graphs against $\Omega(\log \Delta)$; tight on rooted rings and trees. Open Problem 2 of (Molla and Moses Jr. 2019).		

Results.

- Molla and Moses Jr. (2019) [lower bound]: memory $\Omega(\log \Delta)$. *Bottleneck:* port addressing. *Holds when:* all agents have the same memory bound; model assumption of (Molla and Moses Jr. 2019): an agent with b bits sees at most 2^b ports of its node, and the adversary chooses which; holds for randomized algorithms. Theorem 7 of (Molla and Moses Jr. 2019); it rests on that paper's port-visibility assumption, not on a general information argument.
- Molla and Moses Jr. (2019) [current best]: time $O(m+k)$, memory $O(\Delta)$. Introductory and current best. Rooted configurations on arbitrary graphs; the bound is a sketched modification of the rooted algorithm in Section 8, not a numbered theorem. The same paper gives $O(mk/n)$ rounds with $O(\max\{\log \Delta, \log D\})$ bits, and $O(k)$ rounds with $O(\log \Delta)$ bits on rooted rings and rooted trees.
- Molla and Moses Jr. (2019) [also]: time *cover time*, memory $O(\log \Delta)$. *Assumes:* arbitrary initial configuration; $n < k \leq \Delta^\kappa n$. With high probability; this algorithm does not terminate: settled agents stay active forever.

Open questions.

Q1. **Deterministic** dispersion with $k > n$ agents with identifiers on static graphs has not been studied: no algorithm and no lower bound beyond $\Omega(k)$ is recorded in the literature surveyed here. (Related work exists only on dynamic rings and dynamic graphs under the name balanced dispersion.)

Q2. Is $O(k)$ time possible for rooted configurations when $k > n$, matching the $k \leq n$ case?

Q3. Terminating dispersion with $k > n$ on general rooted graphs with $O(\log \Delta)$ bits per agent (Open Problem 2 of Molla and Moses Jr. 2019).

Notes. The balanced target, at most $\lceil k/n \rceil$ agents per node, is the formulation of (Molla and Moses Jr. 2019) itself. Many agents may cross one edge in the same round, so the $\Omega(k)$ argument for $k \leq n$ does not carry over. Checked against the full text (arXiv 1902.10489).

Claude (Anthropic) was used in cataloguing the problems.

16.13.21 Directed graphs

Adds + local communication, + synchronous VERIFIED

Edges can be traversed in one direction only, so an agent cannot backtrack; the problem is solvable only on strongly connected digraphs and the known time bounds are quadratic in k.

Assumed: graph dynamics: static graph; number of agents: $k \leq n$; identifiers: distinct identifiers (eponymous): agents have pairwise distinct identifiers; faults: no faults; communication: local (co-located agents communicate); timing: synchronous rounds. **Not assumed:** edge direction: directed (strongly connected); randomness: deterministic; topology: arbitrary connected graph; grid orientation: unoriented (no consistent compass); node memory: no whiteboards; node identifiers: anonymous nodes; initial configuration: arbitrary (agents on any $\ell \leq k$ nodes).

Statement for this setting. G is a directed graph with maximum out-degree Δ_{out}; at a node an agent sees only the out-ports $1, \ldots, \deg_{out}(v)$. Dispersion is not always solvable when G is not strongly connected, so results assume strong connectivity. ϕ denotes the deficiency of G, the number of arcs whose addition makes G Eulerian; $f \leq k$ agents may crash.

Measure	Lower bound	Best upper bound	Status
Time	$\Omega(k)$ (Italiano et al. 2025); $\Omega(D^2)$ (Italiano et al. 2025); $\Omega(k \cdot \min\{k, \phi\})$ (Italiano et al. 2025); *impossible* (Italiano et al. 2025)	$O(k^2\Delta_{out})$ (Italiano et al. 2025)	• polynomial gap
	$O(k^2\Delta_{out})$ versus $\Omega(k)$ in general; DFS-based algorithms need $\Omega(k \min\{k, \phi\})$, which $O(\phi k^2)$ nearly matches.		
Memory	$\Omega(\log k)$ (Italiano et al. 2025)	$\Theta(\log(k + \Delta_{out}))$ (Italiano et al. 2025)	• near-optimal
	$\Theta(\log(k + \Delta_{out}))$ is achieved; the proved lower bound is $\Omega(\log k)$.		

Results.

- Italiano et al. (2025) [lower bound]: time $\Omega(k)$. *Bottleneck:* distance. *Holds when:* Theorem 3: deterministic; a directed cycle.
- Italiano et al. (2025) [lower bound]: time $\Omega(D^2)$. *Bottleneck:* collaborative tree exploration. *Holds when:* Theorem 4: deterministic, $k = n$, $D = \omega(1)$; a strongly connected digraph built from a tree, after the exploration bound of (Disser et al. 2020).
- Italiano et al. (2025) [lower bound]: time $\Omega(k \cdot \min\{k, \phi\})$. *Bottleneck:* no backtracking. *Holds when:* Theorem 5: only for algorithms based on DFS traversal; deficiency $\phi > 0$.
- Italiano et al. (2025) [lower bound]: time *impossible*. *Bottleneck:* no backtracking. *Holds when:* Theorem 2: on some digraphs that are not strongly connected; a rooted start at a node from which some node is unreachable.
- Italiano et al. (2025) [lower bound]: memory $\Omega(\log k)$. *Bottleneck:* symmetry breaking. *Holds when:* Theorem 6: deterministic; a rooted start; memory counts the identifier.
- Italiano et al. (2025) [current best]: time $O(k^2\Delta_{out})$, memory $\Theta(\log(k + \Delta_{out}))$. Introductory and current best (Theorem 1(b)): DFS with ranked backtracking paths of at most $k - 1$ arcs; $O(fk^2\Delta_{out})$ with f crashes. Optimal memory.
- Italiano et al. (2025) [also]: time $O(\phi k^2)$, memory $O(k \log(k+\Delta_{out}))$. Theorem 1(a), simulating Kwek's exploration; $O(f\phi k^2)$ with f crashes. Better when $\phi < \Delta_{out}$.

Open questions.

Q1. Close the time gaps; in particular $O((k + f)k\Delta_{out})$ with f crashes (Italiano et al. 2025).

Q2. BFS-based dispersion in the local model, without 1-hop communication (Italiano et al. 2025).

Q3. Dispersion on dynamic directed graphs, and in semi-synchronous and asynchronous settings (Italiano et al. 2025).

Claude (Anthropic) was used in cataloguing the problems.

Notes. The algorithms do not terminate on their own: agents cannot tell that dispersion is complete. With k, Δ_{out}, f (and ϕ or D) known in advance, a round counter makes them terminate (Corollary 1). Checked against the full text of the JPDC version.

16.13.22 Directed graphs, 1-hop communication

Adds + h-hop communication, + synchronous VERIFIED

On strongly connected digraphs, agents that also talk to agents on neighbouring nodes disperse in $O(kD)$ rounds by breadth-first search, at the price of $O(k \log(k + \Delta_{out}))$ bits.

Assumed: graph dynamics: static graph; number of agents: $k \leq n$; identifiers: distinct identifiers (eponymous): agents have pairwise distinct identifiers; faults: no faults; communication: h-hop (agents within h hops communicate); timing: synchronous rounds. **Not assumed:** edge direction: directed (strongly connected); randomness: deterministic; topology: arbitrary connected graph; grid orientation: unoriented (no consistent compass); node memory: no whiteboards; node identifiers: anonymous nodes; initial configuration: arbitrary (agents on any $\ell \leq k$ nodes).

Statement for this setting. G is a directed graph with maximum out-degree Δ_{out}; at a node an agent sees only the out-ports $1, \ldots, \deg_{out}(v)$. Dispersion is not always solvable when G is not strongly connected, so results assume strong connectivity. ϕ denotes the deficiency of G, the number of arcs whose addition makes G Eulerian; $f \leq k$ agents may crash. Agents communicate with co-located agents and with agents on nodes at distance 1 ($h = 1$); communication does not follow arc directions.

Measure	Lower bound	Best upper bound	Status
Time	$\Omega(k)$ (Italiano et al. 2025); $\Omega(D^2)$ (Italiano et al. 2025); *impossible* (Italiano et al. 2025)	$O(kD)$ (Italiano et al. 2025)	• polynomial gap
	$O(kD)$ against $\Omega(k)$ (and $\Omega(D^2)$ for $k = n$): a factor up to D.		
Memory	$\Omega(\log k)$ (Italiano et al. 2025)	$O(k \log(k + \Delta_{out}))$ (Italiano et al. 2025)	• polynomial gap
	$O(k \log(k + \Delta_{out}))$ against $\Omega(\log k)$: a factor k.		

Results.

- Italiano et al. (2025) [lower bound]: time $\Omega(k)$. *Bottleneck:* distance. *Holds when:* Theorem 3: deterministic; a directed cycle; stated for any deterministic algorithm, so it holds with 1-hop communication.
- Italiano et al. (2025) [lower bound]: time $\Omega(D^2)$. *Bottleneck:* collaborative tree exploration. *Holds when:* Theorem 4: deterministic, $k = n$, $D = \omega(1)$; a strongly connected digraph built from a tree, after the exploration bound of (Disser et al. 2020); stated for any deterministic algorithm, so it holds with 1-hop communication.
- Italiano et al. (2025) [lower bound]: time *impossible*. *Bottleneck:* no backtracking. *Holds when:* Theorem 2: on some digraphs that are not strongly connected; a rooted start at a node from which some node is unreachable; stated for any deterministic algorithm, so it holds with 1-hop communication.
- Italiano et al. (2025) [lower bound]: memory $\Omega(\log k)$. *Bottleneck:* symmetry breaking. *Holds when:* Theorem 6: deterministic; a rooted start; memory counts the identifier; stated for any deterministic algorithm, so it holds with 1-hop communication.
- Italiano et al. (2025) [current best]: time $O(kD)$, memory $O(k \log(k + \Delta_{out}))$. *Assumes:* $h = 1$. Introductory and current best (Theorem 1(c)): BFS-based; in every $O(D)$ rounds at least one agent settles on an empty node. The bound also holds with $f \leq k$ crashes, with no extra time.

Open questions.

Q1. A BFS-based algorithm with $O(\log(k + \Delta_{out}))$ bits per agent, as with local communication (Italiano et al. 2025).

Q2. Does $O(kD)$ improve to $O(k + D^2)$ or $O(k)$ with 1-hop communication?

Claude (Anthropic) was used in cataloguing the problems.

Notes. The lower bounds of Theorems 2, 3, 4 and 6 of (Italiano et al. 2025) are stated for every deterministic algorithm and hold here; Theorem 5 bounds only DFS-based algorithms and does not apply to the BFS algorithm. Italiano, Pattanayak and Sharma remark that 1-hop information "is of no use in the DFS based algorithm". Checked against the full text of the JPDC version.

16.13.23 Dynamic graphs (1-interval connected)

Adds + local communication, + synchronous VERIFIED

An adversary changes the edge set every round, keeping the graph connected in every round. With only local communication the problem is impossible.

Assumed: graph dynamics: dynamic graph, connected in every round (1-interval connected); edge direction: undirected; number of agents: $k \leq n$; identifiers: distinct identifiers (eponymous): agents have pairwise distinct identifiers; faults: no faults; communication: local (co-located agents communicate); timing: synchronous rounds. **Not assumed:** randomness: deterministic; topology: arbitrary connected graph; grid orientation: unoriented (no consistent compass); node memory: no whiteboards; node identifiers: anonymous nodes; initial configuration: arbitrary (agents on any $\ell \leq k$ nodes).

Statement for this setting. The node set is fixed; in every round the adversary chooses the edge set, subject to the graph being connected in that round (1-interval connectivity). Agents see the ports of the current round. **1-neighbourhood knowledge** means an agent knows which neighbouring nodes are occupied and by whom.

Measure	Lower bound	Best upper bound	Status
Time	*impossible* (Kshemkalyani et al. 2020a) Local communication is not enough; see the global-communication branch.	—	• impossible
Memory	*impossible* (Kshemkalyani et al. 2020a)	—	• impossible

Results.

- Kshemkalyani et al. (2020a) [lower bound]: time *impossible. Bottleneck:* adversarial edge changes. *Holds when:* $k \geq 5$; deterministic; local communication; holds even with 1-neighbourhood knowledge and unbounded memory. For $k \geq 5$ with local communication, even with 1-neighbourhood knowledge and unbounded memory. Also impossible for $k \geq 3$ without 1-neighbourhood knowledge, even with global communication. $\Omega(k)$ rounds where solvable.
- Kshemkalyani et al. (2020a) [lower bound]: memory *impossible. Bottleneck:* adversarial edge changes. *Holds when:* $k \geq 5$; deterministic; local communication.

Open questions.

Q1. Which combination of communication range and visibility is **necessary** for dispersion on T-interval connected graphs with $T > 1$, where the impossibility proof for $T = 1$ does not apply? (Saxena and Mondal 2025)

Q2. Dynamic graphs with a bounded number of missing edges per round: (Saxena et al. 2026a) gives impossibilities on 1-bounded 1-interval connected graphs for several weaker communication and visibility combinations and an $O(k)$ algorithm with global communication and 1-hop visibility; the local-communication case with few missing edges is not settled.

Notes. The impossibility is the reason the dynamic-graph literature works with global communication. Rings are the exception: with full visibility of the ring, dispersion on a dynamic ring is possible with no communication model at all (see the dynamic ring branch).

Claude (Anthropic) was used in cataloguing the problems.

16.13.24 Dynamic graphs with global communication

Adds + global communication, + synchronous VERIFIED

Global communication plus 1-neighbourhood knowledge make dispersion solvable on 1-interval connected graphs, and in optimal $\Theta(k)$ time.

Assumed: graph dynamics: dynamic graph, connected in every round (1-interval connected); edge direction: undirected; number of agents: $k \leq n$; identifiers: distinct identifiers (eponymous): agents have pairwise distinct identifiers; faults: no faults; communication: global (any agent can read any other agent's state); timing: synchronous rounds. **Not assumed:** randomness: deterministic; topology: arbitrary connected graph; grid orientation: unoriented (no consistent compass); node memory: no whiteboards; node identifiers: anonymous nodes; initial configuration: arbitrary (agents on any $\ell \leq k$ nodes).

Measure	Lower bound	Best upper bound	Status
Time	$\Omega(k)$ (Kshemkalyani et al. 2020a) $\Theta(k)$; $\Theta(kT)$ in the T-path connected model.	$\Theta(k)$ (Kshemkalyani et al. 2020a)	• tight
Memory	$\Omega(\log k)$ (Kshemkalyani et al. 2020a) $\Theta(\log k)$.	$\Theta(\log k)$ (Kshemkalyani et al. 2020a)	• tight

Results.

- Kshemkalyani et al. (2020a) [lower bound]: time $\Omega(k)$. *Bottleneck:* adversarial edge changes. *Holds when:* a 1-interval connected dynamic tree with dynamic diameter $O(1)$; even with unbounded memory and global communication; $\Omega(kT)$: T-path connected graphs, even with full visibility and known k, n, T. On a 1-interval connected tree with dynamic diameter $O(1)$, even with unbounded memory. $\Omega(kT)$ on T-path connected graphs (Saxena and Mondal 2025).
- Kshemkalyani et al. (2020a) [lower bound]: memory $\Omega(\log k)$. *Bottleneck:* symmetry breaking. *Holds when:* deterministic; all agents have the same memory bound; an initial configuration with co-located agents (rooted suffices); memory counts the identifier.
- Kshemkalyani et al. (2020a) [current best]: time $\Theta(k)$, memory $\Theta(\log k)$. *Assumes:* 1-neighbourhood knowledge. Introductory and current best. With $f \leq k$ crash faults the same algorithm runs in $O(k - f)$ rounds.
- Saxena and Mondal (2025) [also]: time $\Theta(kT)$, memory $O(\log \max(k, T))$. *Assumes:* 1-hop visibility; knows T (for termination). T-path connected graphs: every pair of nodes shares a connected component at least once in every window of T rounds, so the graph may be disconnected in some rounds. Impossible on connectivity-time graphs (union of T consecutive rounds connected) for $k \geq 3$.
- Saxena et al. (2026a) [also]: time $O(k)$, memory $O(\log n)$. *Assumes:* 1-hop visibility; ℓ-bounded 1-interval connected; any k, including $k > n$. **Balanced** dispersion: every node ends with $\lfloor k/n \rfloor$ or $\lceil k/n \rceil$ agents. Impossible on temporally connected graphs even with unbounded memory and full visibility.

Open questions.

Q1. Byzantine agents on dynamic graphs (Kshemkalyani et al. 2020a).

Q2. Memory $O(\log k)$ instead of $O(\log \max(k, T))$ on T-path connected graphs when T need not be known (Saxena and Mondal 2025).

16.13.25 Dynamic rings

Adds + ring, + no faults, + local communication, + synchronous VERIFIED

A ring from which the adversary removes at most one edge per round (or permutes the nodes); solvable without communication if agents can see the whole ring.

Assumed: graph dynamics: dynamic graph, connected in every round (1-interval connected); edge direction: undirected; number of agents: $k \leq n$; identifiers: distinct identifiers (eponymous): agents have pairwise distinct identifiers; topology: ring or path; faults: no faults; communication: local (co-located agents communicate); timing: synchronous rounds. **Not**

Claude (Anthropic) was used in cataloguing the problems.

assumed: randomness: deterministic; grid orientation: unoriented (no consistent compass); node memory: no whiteboards; node identifiers: anonymous nodes; initial configuration: arbitrary (agents on any $\ell \leq k$ nodes).

Statement for this setting. Two kinds of dynamism are studied: **1-interval connectivity** (at most one edge missing per round) and **vertex permutation** (the adversary reorders the nodes every round, keeping a ring). Visibility is a parameter: with full visibility an agent sees which nodes of the ring are empty, single or multiply occupied.

Measure	Lower bound	Best upper bound	Status
Time	$\Omega(n)$ (Mondal et al. 2025a) $O(n)$ versus $\Omega(n)$ with full visibility and chirality.	$O(n)$ (Agarwalla et al. 2018)	• tight
Memory	none known Memoryless apart from identifiers.	$O(1)$ (Agarwalla et al. 2018)	• tight

Results.

- Mondal et al. (2025a) [lower bound]: time $\Omega(n)$. *Bottleneck:* distance. *Holds when:* rooted configuration on an n-node ring, static or 1-interval connected; $k = n$. For rooted configurations on an n-node ring, static or 1-interval connected; a distance argument already in (Agarwalla et al. 2018).

- Agarwalla et al. (2018) [current best]: time $O(n)$, memory $O(1)$. *Assumes:* $k = n$; full visibility; chirality. Introductory and current best. $O(n)$ rounds under vertex permutation, under 1-interval connectivity, and under both. Without chirality: $O(n)$ for rooted configurations and for odd n, $O(1)$ for $n = 4$. The algorithms are essentially memoryless. Without any visibility the problem is impossible (vertex permutation, $n \geq 3$; 1-interval connectivity, $n \geq 2$).

- Mondal et al. (2025a) [also]: time $\Theta(n)$. *Assumes:* rooted; chirality; limited visibility. Distance-k-dispersion on a 1-interval connected ring with local communication; $k = 1$ is dispersion. Impossible under a semi-synchronous scheduler if the initial configuration has a multiplicity node. Memory not recorded here.

- Mondal et al. (2025b) [preprint] [also]: time $O(\ell n)$. *Assumes:* no chirality; ℓ agents, $3 \leq \ell \leq n$. Arbitrary initial configurations without chirality; partially answers the even-ring question of (Agarwalla et al. 2018).

Open questions.

Q1. Visibility strictly between none and full: how much must an agent see of a dynamic ring for dispersion to be possible? (Agarwalla et al. 2018)

Q2. Achiral agents on even rings of size at least 6 under vertex permutation and 1-interval connectivity: is $O(n)$ time possible from arbitrary configurations? (Agarwalla et al. 2018); $O(\ell n)$ is known without vertex permutation (Mondal et al. 2025b).

16.14 Black hole search

A black hole destroys every agent that enters it, without trace; a team must locate it, losing as few agents as possible.

First posed in 2001 (Dobrev et al. 2007). Dobrev, Flocchini, Prencipe and Santoro introduced black hole search in rings (DISC 2001) and in arbitrary networks (PODC 2002) (Dobrev et al. 2006).

Claude (Anthropic) was used in cataloguing the problems.

Problem statement and measures

A connected graph contains one **black hole**: a node that destroys any agent arriving there, leaving no trace. All other nodes are **safe**. Agents start at a safe node s (the **home base**) and must determine the location of the black hole; at least one agent must survive and know it. The primary measure is the **team size** (number of agents needed); the secondary measure is the **cost** (total number of moves, a move being one traversal of one edge) or, for synchronous agents, the time. In most settings the black hole is present from the start; an **eventually emerging** black hole appears at an unknown time during the execution, and time is then counted from its appearance. Some papers also start the agents **scattered**, at distinct safe nodes; those results are recorded in the notes, because every catalogued variant starts at a home base.

Measures: **Agents**, number of agents needed (team size); **Cost**, total moves (asynchronous) or rounds (synchronous).



(a) Start: agents A, B at s; ring of 6

(b) B reads 'trying' on edge v1-v2: the black hole is v2

Figure 16.14: Black hole search on a ring by two agents with cautious walk (Dobrev et al. 2007) (the problem: Dobrev et al. 2006). Agents A and B start at the safe home base s of a ring of six nodes; one node (black) destroys every agent that enters it. Each explores half of the unexplored nodes: before crossing a new edge it marks it *trying* on the whiteboard (dashed), and after arriving it returns to mark it *safe* (green) before going on. A dies on its second new edge; B finishes its half, walks back through the explored part and finds the edge still marked *trying*: the one unexplored node beyond it is the black hole. One agent survives and knows it.

A first look

Imagine a computer network in which one computer has been taken over: every travelling program that arrives there is deleted without a trace, and nobody outside can tell. Find that computer by sending such programs, while losing as few of them as possible. We model the network as a graph: computers are **nodes**, cables between them are **edges**, and each travelling program is an **agent**. In the basic version the graph has n nodes and contains exactly one **black hole**, a node that destroys every agent that enters it; every other node is **safe**. A team of identical agents starts at a safe node s, the **home base**; they have no map, but they know n. At each node the edges leaving it are numbered $1, 2, \ldots$ (these numbers are called **ports**), each node has a **whiteboard** (a small memory on which agents can read and write notes), and the agents are **asynchronous**: every **move** (one traversal of one edge) takes a finite but unpredictable time, chosen by an **adversary** (an imagined opponent who picks the worst timing). The task: at least one agent must survive and know exactly which node is the black hole. We count the agents needed and the total number of moves.

1. Algorithm: Cautious walk and halving the unexplored ring. (Dobrev et al. 2007) (Asynchronous agents with whiteboards in a ring, p. 296) Two agents start together at s on a ring of n nodes (a cycle in which every node has two neighbours); they know n.

Claude (Anthropic) was used in cataloguing the problems.

1. **Cautious walk** (used for every step into the unknown): an agent that crosses an edge nobody has crossed before first writes *\"I am trying this edge\"* on the whiteboard where it starts. If it arrives safely, it immediately goes back and changes the note to *\"this edge is safe\"*, and only then continues. Nobody else ever enters an edge marked *\"trying\"*.
2. The unexplored nodes form one stretch of the ring. A **phase** starts: cut it into two halves of almost equal length; one agent explores the left half, the other the right half, both by cautious walk.
3. The black hole lies in at most one half, so one agent finishes its half. It walks through the explored part to the last safe edge of the other agent.
4. What is still unexplored is again one stretch, at most about half as long. A new phase starts: the finishing agent cuts it in two again, writes the new plan on the whiteboard there for its partner (in case the partner is only slow and comes back), and takes the half on the far side, away from the edge its partner is trying.
5. When only one unexplored node remains, that node is the black hole.

Why it works and what it costs. Under cautious walk an agent that does not come back has died on the very first node beyond its last safe edge, so the black hole, if it has killed the partner, lies in the near half; the finishing agent always takes the far half and never enters an edge marked *\"trying\"*, so exactly one agent dies and the other survives. Each phase at least halves the unexplored stretch, so there are about $\log n$ phases, and each phase costs $O(n)$ moves (walking around the explored part and exploring), giving $O(n \log n)$ moves. (Dobrev et al. 2007) show the exact cost $2n \log n + O(n)$, and they prove that any algorithm, with any number of agents, needs at least $(n-1)\log(n-1) + O(n)$ moves, so this simple plan is optimal up to a factor of two.

Why it matters. Cautious walk is the basic safety tool of almost every later black hole search algorithm, in rings and in arbitrary graphs.

2. Lower bound: Every unexplored port may lead to the black hole. (Dobrev et al. 2006) (Asynchronous agents with whiteboards, p. 296) Without a map, the agents do not know where a port leads. The adversary may therefore decide, as the algorithm runs, that the **first** port tried out of a node is the one that leads to the black hole. Take a graph in which the black hole is a neighbour of many nodes, for example the black hole 0 joined to every other node, with the home base at node 1. The first agent leaving node 1 through a new port is sent into the black hole. Some other agent must then reach node 2; the first agent that leaves node 2 through a new port is again sent into the black hole, and so on, node after node. Each such node costs one agent, and the search is not finished while two or more nodes are still unexplored, because a node that nobody has returned from may be the black hole or may simply be slow to reach.

Why it holds. In this example the algorithm loses an agent at almost every node adjacent to the black hole, and the black hole can have up to Δ neighbours, where Δ is the largest number of edges at any node (the **maximum degree**). With a cleverer graph (a ring of nodes all joined to a central black hole, which the adversary later replaces by a graph of maximum degree Δ that looks the same to the agents) (Dobrev et al. 2006) prove that $\Delta + 1$ agents are needed; the full proof is longer. The same paper gives an algorithm with exactly $\Delta + 1$ agents and $O(n^2)$ moves, and shows that $\Omega(n^2)$ moves are necessary for a team of this size, so both bounds are tight.

Why it matters. It shows that what limits a team without a map is the number of ports that can lead to the black hole, which is why a map or a sense of direction lowers the team to two agents.

3. Impossibility: Without knowing n, slow and dead look the same. (Dobrev et al. 2006) (Asynchronous agents with whiteboards, p. 296) Because moves take unpredictable time, an agent that has not come

Claude (Anthropic) was used in cataloguing the problems.

back may be dead or just slow. Suppose the agents do not know n. Take a graph G with the black hole at node v, and a second graph G' in which v is replaced by two nodes v_1 and v_2 joined to each other, sharing the edges that went to v. In G' the adversary makes every edge into v_1 or v_2 extremely slow. Up to the moment the algorithm stops, the agents see exactly the same things in G and in G', so they stop in G' too, with both v_1 and v_2 unexplored: they cannot know which one is the black hole.

Why it holds. The same reasoning shows that the agents can never stop while two nodes are unexplored, and that the problem is unsolvable when the graph has a node (other than s) whose removal disconnects it: if that node is the black hole, everything behind it stays unexplored (Dobrev et al. 2006). This is why the basic version assumes that n is known and that the graph stays connected after removing any one node (it is *2-connected*).

Why it matters. The indistinguishability of slow and dead agents is the reason asynchronous black hole search needs knowledge of n, and it reappears in every asynchronous lower bound for the problem.

What the community started looking at

From the start the objective was the **minimum number of agents** that guarantees success, since every agent that enters the black hole is lost; the number of moves came second (Dobrev et al. 2006). Later work replaced whiteboards by tokens (Dobrev et al. 2013; Flocchini et al. 2012b; Balamohan et al. 2016), turned to synchronous agents where time can be optimised (Czyzowicz et al. 2006; Klasing et al. 2007), to dynamic graphs (Di Luna et al. 2025b; Kaur et al. 2026b), and to Byzantine companions (Di Luna et al. 2025a). Recent variants let the black hole appear only during the execution, on rings (Bonnet et al. 2026b) and on arbitrary graphs (Kaur et al. 2026a), or destroy agents only in rounds chosen by the adversary (Bhattacharya et al. 2025).

Further reading

Papers on this problem that are not yet placed in a setting.

- Kaur and Saxena (2026). Black hole search with verification in synchronous dynamic graphs: if there is no black hole, all agents must terminate. With at most one edge missing per round, 3 co-located agents and $\deg(v_{BH}) + 1$ scattered agents cannot solve it even with 1-hop visibility and global communication (Theorems 2.1, 2.2); 4 co-located agents with 1-hop visibility and face-to-face communication solve it in $O(n^4)$ rounds (Theorem 3.2), and $\deg(v_{BH}) + 2$ agents from any start with global communication in $O(\deg(v_{BH})\, m^2)$ rounds (Theorem 4.2). In always-connected dynamic graphs $n - 2$ agents cannot solve it even with full visibility, and $n - 1$ agents with global communication and 1-hop visibility solve it in $O(n)$ rounds (Theorems 2.3, 5.1). Agents have distinct identifiers and $O(\log n)$ bits. Read (arXiv v2).
- Feletti et al. (2026d). Safe exploration of temporally connected dynamic graphs that contain any number of black holes, unknown to the agents: semi-synchronous agents start on arbitrary safe nodes, nodes have whiteboards, and every safe node must be visited (perpetually, or at least once with termination) while at least one agent survives. Without knowledge of the graph, every algorithm loses one agent per port that leads to a black hole, and no algorithm can mark exactly the ports that lead to black holes, even with synchronous agents of unbounded memory (Theorem 4). With $2\eta' + \sum_{x \in B} \deg(x) + 1$ agents, where B is the set of black holes and η' the number of transient edges between safe nodes, oblivious anonymous agents explore perpetually with whiteboards of $O(\Delta)$ bits (Theorem 11), and agents with identifiers and $O(n \log n)$ bits of memory that know the number of safe nodes explore and terminate (Theorem 16). The algorithms combine a rotor-router with cautious walk. Preprint; read.
- Bhattacharya et al. (2025). Perpetual exploration with a Byzantine black hole: in each round the adversary decides whether the black hole destroys the agents on it and those entering it, so it may never be detected. Co-located synchronous agents with distinct identifiers and face-to-face communication, without knowledge of n, must visit infinitely often every node of one component of the graph without the black hole, or of the component of the home base. On paths and trees, 4 agents (any component) and 6 agents (component of the

Claude (Anthropic) was used in cataloguing the problems.

home base) are necessary and sufficient (Theorems 4 to 7 and 36, Corollary 37; the lower bounds hold even when n is known). In arbitrary graphs, $2\Delta-1$ agents are necessary when $\Delta \geq 4$ (Theorem 10) and $3\Delta+3$ agents with $O(n\Delta \log n)$ bits each suffice for the component of the home base (Theorem 12). If the black hole destroys agents in every round, perpetual exploration needs $\Delta+1$ agents (Theorem 66) and $\Delta+2$ suffice (Remark 67). Read (arXiv v1).

- Bhattacharya et al. (2024b). Dynamic oriented $n \times m$ tori ($3 \leq n \leq m$) in which every row ring and column ring is 1-interval connected; synchronous agents with identifiers that know n and m, a whiteboard of $O(1)$ bits at each port. Co-located: $n+1$ agents cannot locate the black hole (Theorem 1), $n+c$ agents with $c \geq 2$ need $\Omega(m \log n)$ rounds (Theorem 3), $n+3$ agents take $O(nm^{1.5})$ rounds and $n+4$ agents $O(nm)$ (Theorems 7, 8). Scattered: $n+2$ agents cannot (Theorem 4), $n+c$ agents with $c \geq 3$ need $\Omega(nm)$ rounds (Theorem 5), $n+6$ agents take $O(nm^{1.5})$ rounds and $n+7$ agents $O(nm)$ (Theorems 9, 10). Read (arXiv v1).

- Bhattacharya et al. (2024a). Dynamic cactus graphs (every edge on at most one cycle) that stay connected, with whiteboards and co-located synchronous agents with identifiers. With at most one edge missing per round, 2 agents cannot locate the black hole, 3 agents need $\Omega(n^{1.5})$ moves and rounds and use $O(n^2)$, and 4 agents need $\Omega(n)$. With at most c edges missing (their k), $c+1$ agents cannot, and $2c+3$ agents use $O(cn)$ rounds and $O(c^2 n)$ moves (Theorems 9 to 11). Read (arXiv v1).

- Goswami et al. (2024b). Perpetual exploration of an oriented ring with a Byzantine black hole (it may let agents pass, destroy them, or erase what is written on it), by synchronous agents with identifiers that know n. Co-located: 3 agents are necessary and sufficient with pebbles or with whiteboards, and 5 suffice face-to-face; scattered: 4 agents with pebbles and 3 with whiteboards are necessary and sufficient (Table 1). Extended to trees and arbitrary graphs in (Bhattacharya et al. 2025). Read (arXiv v2).

- Dobrev et al. (2023). Gathering despite a black hole: k anonymous asynchronous agents, dispersed in an anonymous ring with whiteboards, must gather and stop at one node, and the question is how many of them can be guaranteed to. Knowing k but not n: $k-1$ in an oriented ring; in an unoriented ring $k-2$ for odd k and $(k-2)/2$ for even k. Knowing n but not k: $k-2$ in an oriented ring, and in an unoriented one $k-2$ unless k is even and n odd, then $(k-2)/2$. One agent more is impossible in each case (Theorems 1, 2 and 10; Table 1), at most $\lfloor k/2 \rfloor$ when neither is known (Theorem 3), and in an unoriented ring $k-2$ agents can always end within distance 1 of each other. The algorithms take $O(n)$ time and $O(nk)$ moves; the one for unknown k also locates the black hole in $\Theta(n)$ ideal time, against $O(n \log n)$ in (Dobrev et al. 2007). Read in full.

- D'Emidio et al. (2015). Explore and repair, the model of Cooper, Klasing and Radzik extended by a radius r: k synchronous labelled agents start at a safe home base of a known graph with b unknown black holes; an agent that enters a black hole repairs it and is destroyed, with every agent within distance r. Making the graph safe as fast as possible is NP-hard even for $b = k = 1$ (Theorem 3.1); for $r > 0$ every strategy needs $\Omega(n/k + D + r \min\{D, b\} + |K|)$ steps, $|K|$ the size of a largest clique (Lemma 3.4); for $r \leq 1$ three strategies (snake, scout, parallel scout) take $O(n)$ steps, which some graphs need, and are compared in experiments. Read (model and theorems).

- Czyzowicz et al. (2009a). Black hole search in a simple strongly connected directed graph with n nodes and local port labels: agents with polynomial memory start at a home base and share whiteboards. An agent that has crossed an arc cannot walk back to mark it safe, and the number of agents becomes exponential. Let $\deg_{in}(v_{BH})$ be the in-degree of the black hole v_{BH} (their Δ). For every algorithm and every $n > \deg_{in}(v_{BH})$, some n-node digraph needs $2^{\deg_{in}(v_{BH})}$ agents, even synchronous ones (Theorem 1), whereas $\deg(v_{BH})+1$ agents always suffice in undirected graphs. In planar digraphs whose planar embedding the agents know, $2^{\deg_{in}(v_{BH})}+1$ agents suffice (Theorem 2), and in some execution $2^{\deg_{in}(v_{BH})}-1$ agents enter the black hole (Theorem 3). They ask whether a number of agents that depends on $\deg_{in}(v_{BH})$ alone suffices in every digraph. Read (model and theorems).

- Kosowski et al. (2011). Synchronous agents in the setting of (Czyzowicz et al. 2009a): agents with distinct identifiers start at a home base, know n (or an upper bound on it) and the number $\deg_{in}(v_{BH})$ of arcs that lead into the black hole (their Δ), and every node has a whiteboard accessed in mutual exclusion. $O(\deg_{in}(v_{BH}) \cdot 2^{\deg_{in}(v_{BH})})$ synchronous agents suffice in every strongly connected digraph (Theorem 3.1), against the lower bound $2^{\deg_{in}(v_{BH})}$ of (Czyzowicz et al. 2009a), which answers its question for synchronous agents. With one arc into the black hole, 2 agents are necessary and sufficient, synchronous or not (Theorem 4.1). With two arcs, 4 synchronous agents are necessary and sufficient (Theorem 3.2), while asynchronous agents sometimes need 5 (Theorem 4.2): unlike in undirected graphs, synchrony changes the number of agents. Read (abstract, results and theorem statements).

Claude (Anthropic) was used in cataloguing the problems.

Relations to other problems

- **Variant of Collaborative exploration.** Locating the black hole means exploring every other node safely, so black hole search is exploration of a dangerous graph by a team, where agents can be lost.
- **Related: Network decontamination.** Both are security problems for agents in a hostile network: in black hole search the network destroys agents, in decontamination agents must clear a network that an arbitrarily fast intruder (or contamination) can re-enter.

Assumptions specific to this problem

Knowledge of the graph

Dobrev, Flocchini, Prencipe and Santoro organised black hole search in arbitrary networks around the agents' a priori knowledge: with topological ignorance, $\Delta + 1$ agents are necessary and sufficient and the cost is $\Theta(n^2)$; with sense of direction two agents suffice, still at cost $\Theta(n^2)$; with complete topological knowledge two agents suffice at cost $\Theta(n \log n)$ (Dobrev et al. 2006). Knowing n is the floor of the axis, because they prove that asynchronous black hole search is unsolvable when n is unknown (Dobrev et al. 2006). Sense of direction is the notion of Flocchini, Mans and Santoro: a globally consistent labelling of the ports from which one can decide, from the label sequences alone, whether two walks from a node end at the same node (Flocchini et al. 1998) (as defined in Dobrev et al. 2006); Dobrev et al. trace its effect on exploration back to Blum and Kozen's study of the power of a compass (Blum and Kozen 1978). The three levels thus separate team size (from $\Delta + 1$ down to 2) and, independently, cost (from n^2 down to $n \log n$).

Level	Value	Formal definition
0	**topology unknown**	Agents know n but not the topology of G.
1	**sense of direction**	Agents have a consistent sense of direction: a globally consistent labelling of edges from which they can tell whether two walks lead to the same node.
2	**map**	Agents have a map of G and know their starting position on it.

Marking nodes

The first black hole search papers gave the agents a whiteboard at every node (Dobrev et al. 2007, 2006). Flocchini, Ilcinkas and Santoro observe that whiteboards let a protocol assume FIFO links and break symmetry, whereas "the theoretical quest, on the contrary, has been for the weakest interaction mechanism allowing the problem to be solved"; that weaker mechanism is identical tokens, taken from the early work on exploring safe graphs (Flocchini et al. 2012b). Dobrev, Flocchini, Královič and Santoro showed that $\Delta + 1$ agents, each with a token that may be placed on a node or a port, locate the black hole without a map at polynomial cost (Dobrev et al. 2013) (as summarised in Flocchini et al. 2012b); with a map, Flocchini, Ilcinkas and Santoro gave the first proof that one pebble per agent, on nodes only, matches the whiteboard bounds of two agents and $\Theta(n \log n)$ moves, even on non-FIFO links (Flocchini et al. 2012b). Without a map the equivalence fails for a constant number of tokens: Balamohan, Dobrev, Flocchini and Santoro show that $\Delta + 2$ agents with three tokens are necessary and sufficient, one agent more than with whiteboards (Balamohan et al. 2016).

Claude (Anthropic) was used in cataloguing the problems.

Level	Value	Formal definition
0	**no marking**	Agents cannot leave information at nodes.
1	**tokens**	Agents carry tokens (pebbles) they can place at nodes or on ports; tokens are identical and carry no other information.
2	**whiteboards**	Every node has a whiteboard that agents read and write under mutual exclusion.

Black hole appearance

Bonnet, Bramas and Lamani let the black hole appear at any time during the execution: "To the best of our knowledge, all previous investigations of the basic Black Hole Search problem assume that the black hole is present in the initial configuration" (Bonnet et al. 2026b). An emerging black hole defeats the cautious walk alone, because agents that gather on a node to coordinate may all be destroyed when the black hole appears there (Bonnet et al. 2026b). Kaur, Saxena, Mandal and Mondal carried the question from rings to arbitrary graphs (Kaur et al. 2026a). Every other setting of this problem has the black hole present from the start.

Level	Value	Formal definition
0	**eventually emerging**	There is a time $t \geq 0$ and a node u such that from time t on u destroys every agent on it or entering it, and stays a black hole. If $t = 0$ then $u \neq s$; the home base may become the black hole after the first round (after one epoch for asynchronous agents). Time is counted from t (Bonnet et al. 2026b).
1	**present from the start**	The black hole is present at time 0, and the home base s is safe throughout.

Case tree

The root assumes nothing; going down, each edge adds the assumptions written in grey (+) or leaves an axis unassumed. Bold entries are catalogued settings with their status per measure (Agents, Cost) and page; italic entries are branch points, named after the axis on which their branches differ. Axis order: Graph dynamics → Edge direction → Faults → Black hole appearance → Timing → Knowledge of the graph → Marking nodes → Topology.

No assumptions

+ static graph, + undirected

Faults

+ no faults

Black hole appearance

+ present from the start

Timing

asynchronous, topology unknown, + whiteboards

Asynchronous agents with whiteboards STANDARD SETTING ●● p. 296

+ ring

Asynchronous agents with whiteboards in a ring ●● p. 296

asynchronous, + map

Marking nodes

+ whiteboards

Whiteboards and a map ●● p. 297

Claude (Anthropic) was used in cataloguing the problems.

+ tokens
Tokens and a map ●● p. 298

asynchronous, + sense of direction, + whiteboards
Whiteboards and sense of direction ●● p. 298

asynchronous, topology unknown, + tokens
Tokens, no map ●● p. 298

+ synchronous, + map
Synchronous agents with a map ●● p. 299

eventually emerging, + synchronous
Eventually emerging black hole, synchronous agents, arbitrary graphs ●● p. 299

+ sense of direction, + ring
Eventually emerging black hole, synchronous agents, oriented ring ●● p. 300

eventually emerging, asynchronous, + sense of direction, + ring
Eventually emerging black hole, asynchronous agents, oriented ring ●● p. 301

+ at most weak Byzantine faults, + present from the start, + synchronous
Knowledge of the graph

+ map
Byzantine companions, with a map ●● p. 302

topology unknown, + whiteboards
Byzantine companions, whiteboards, no map ●● p. 302

+ connected every round, + undirected, + no faults, + present from the start, + synchronous
Marking nodes

+ tokens, + ring
Dynamic rings ●● p. 303

+ whiteboards
Dynamic graphs with whiteboards ●● p. 303

GRAPHS

What is known

The best upper bound against the best lower bound in each catalogued setting, in tree order. The settings follow in detail.

Setting	Agents	Cost	p.
Asynchronous agents with whiteboards STANDARD SETTING	● tight $\Delta + 1$ vs $\Delta + 1$	● tight $O(n^2)$ vs $\Omega(n^2)$	296
Asynchronous agents with whiteboards in a ring	● tight 2 vs 2	● tight $2n \log n + O(n)$ vs $\Omega(n \log n)$	296
Whiteboards and a map	● tight 2 vs 2	● tight $\Theta(n \log n)$ vs —	297
Tokens and a map	● tight 2 vs 2	● tight $\Theta(n \log n)$ vs —	298
Whiteboards and sense of direction	● tight 2 vs 2	● tight $O(n^2)$ vs —	298
Tokens, no map	● tight $\Delta + 2$ vs $\Delta + 2$	● polynomial gap $O(mn)$ vs $\Omega(n^2/\Delta)$	298

Claude (Anthropic) was used in cataloguing the problems.

Setting	Agents	Cost	p.
Synchronous agents with a map	● tight 2 vs —	● polynomial gap $3\frac{3}{8}$-approx. vs *NP-hard*	299
Eventually emerging black hole, synchronous agents, arbitrary graphs	● tight 4 vs 3	● unknown poly(n) vs —	299
Eventually emerging black hole, synchronous agents, oriented ring	● tight 3 vs 3	● unknown $O(n)$ vs —	300
Eventually emerging black hole, asynchronous agents, oriented ring	● polynomial gap n vs 3	● unknown $O(n)$ vs —	301
Byzantine companions, with a map	● near-optimal $2f+2$ vs —	● unknown $O(n+f)$ vs —	302
Byzantine companions, whiteboards, no map	● near-optimal $(f+1)(\Delta+1)$ vs —	● unknown $O(m+f)$ vs —	302
Dynamic rings	● tight 3 vs 3	● tight $O(n^{1.5})$ vs $\Omega(n^{1.5})$	303
Dynamic graphs with whiteboards	● tight 6 vs 4	● unknown $O(m^2)$ vs —	303

16.14.1 Asynchronous agents with whiteboards

Adds asynchronous, topology unknown, + whiteboards STANDARD SETTING VERIFIED

Without a map, $\Delta+1$ asynchronous agents are necessary and sufficient, and they need $\Theta(n^2)$ moves.

Assumed: graph dynamics: static graph; edge direction: undirected; faults: no faults; black hole appearance: the black hole is present from the start; marking nodes: whiteboards at the nodes. **Not assumed:** timing: asynchronous; knowledge of the graph: no knowledge of the topology (n is known); topology: arbitrary connected graph.

Measure	Lower bound	Best upper bound	Status
Agents	$\Delta+1$ (Dobrev et al. 2006); *impossible* (Dobrev et al. 2006) $\Delta+1$.	$\Delta+1$ (Dobrev et al. 2006)	● tight
Cost	$\Omega(n^2)$ (Dobrev et al. 2006) $\Theta(n^2)$ moves.	$O(n^2)$ (Dobrev et al. 2006)	● tight

Results.

- Dobrev et al. (2006) [lower bound]: agents $\Delta+1$. *Bottleneck:* agents lost in the black hole. *Holds when:* generic algorithms without topological knowledge; even if Δ and n are known.
- Dobrev et al. (2006) [lower bound]: cost $\Omega(n^2)$. *Bottleneck:* asynchrony. *Holds when:* size-optimal generic solutions.
- Dobrev et al. (2006) [lower bound]: agents *impossible*. *Bottleneck:* asynchrony. *Holds when:* if n is unknown, or the graph is not 2-connected.
- Dobrev et al. (2006) [current best]: agents $\Delta+1$, cost $O(n^2)$. Introductory and current best.

Notes. The anonymous ring was treated first (Dobrev et al. 2007); see the variant for rings.

16.14.2 Asynchronous agents with whiteboards in a ring

Adds + ring **to** Asynchronous agents with whiteboards (p. 296) VERIFIED

In an anonymous ring of known size, two co-located asynchronous agents are necessary and sufficient, at

Claude (Anthropic) was used in cataloguing the problems.

optimal cost $\Theta(n \log n)$.

Assumed: graph dynamics: static graph; edge direction: undirected; faults: no faults; black hole appearance: the black hole is present from the start; marking nodes: whiteboards at the nodes; topology: ring or path. **Not assumed:** timing: asynchronous; knowledge of the graph: no knowledge of the topology (n is known).

Measure	Lower bound	Best upper bound	Status
Agents	2 (Dobrev et al. 2007); *impossible* (Dobrev et al. 2007) Two.	2 (Dobrev et al. 2007)	• tight
Cost	$\Omega(n \log n)$ (Dobrev et al. 2007) $\Theta(n \log n)$ moves; the constants differ by a factor of two.	$2n \log n + O(n)$ (Dobrev et al. 2007)	• tight

Results.

- Dobrev et al. (2007) [lower bound]: agents 2. *Bottleneck:* agents lost in the black hole. *Holds when:* at least one agent enters the black hole.
- Dobrev et al. (2007) [lower bound]: cost $\Omega(n \log n)$. *Bottleneck:* asynchrony. *Holds when:* Theorem 1: $(n-1)\log(n-1) + O(n)$ moves, whatever the number of agents; an adversarial scheduler delays moves along edges; holds also for dispersed agents in oriented rings (Theorem 8).
- Dobrev et al. (2007) [lower bound]: agents *impossible*. *Bottleneck:* asynchrony. *Holds when:* if n is unknown: no deterministic protocol always terminates correctly.
- Dobrev et al. (2007) [current best]: agents 2, cost $2n \log n + O(n)$. *Assumes:* co-located agents; n known; $O(\log n)$-bit whiteboards. Introductory and current best (Algorithm DIVIDE, Theorems 2 and 3), size-optimal and cost-optimal within a factor of two; works in unoriented rings, since co-located agents agree on a direction through the whiteboard.

Notes. Checked against the full text of (Dobrev et al. 2007). Agents are anonymous and asynchronous, and the problem is solved when at least one agent survives and all survivors know the location of the black hole. In ideal time (every move one time unit) $2n - 4$ units are necessary whatever the number of agents (Theorem 4) and $n - 1$ co-located agents achieve it (Theorems 5 and 6); $n^{1/q} + 1$ agents need $2(q+1)n - q - o(n)$ time for $1 \le q \le \log n$ (Theorem 7). With **dispersed** agents the cost is $\Omega(n \log n)$, or $\Omega(n \log(n-k))$ when the number k of agents is known (Theorems 8 and 9); in an oriented ring two dispersed agents are necessary and sufficient at cost $O(n \log n)$ (Theorem 10), and in an unoriented ring three are necessary and sufficient at the same cost (Theorem 13).

16.14.3 Whiteboards and a map

Adds + whiteboards VERIFIED

With a map, two agents locate the black hole in $\Theta(n \log n)$ *moves.*

Assumed: graph dynamics: static graph; edge direction: undirected; faults: no faults; black hole appearance: the black hole is present from the start; knowledge of the graph: a map of the graph; marking nodes: whiteboards at the nodes. **Not assumed:** timing: asynchronous; topology: arbitrary connected graph.

Measure	Lower bound	Best upper bound	Status
Agents	2 (Dobrev et al. 2006) Two agents.	2 (Dobrev et al. 2006)	• tight
Cost	none known $\Theta(n \log n)$.	$\Theta(n \log n)$ (Dobrev et al. 2006)	• tight

Results.

Claude (Anthropic) was used in cataloguing the problems.

- Dobrev et al. (2006) [lower bound]: agents 2. *Bottleneck:* agents lost in the black hole. *Holds when:* a single agent never suffices.
- Dobrev et al. (2006) [current best]: agents 2, cost $\Theta(n \log n)$. Introductory and current best.

16.14.4 Tokens and a map

Adds + tokens VERIFIED

Whiteboards can be replaced by one pebble per agent: two agents, $\Theta(n \log n)$ moves, even on non-FIFO edges.

Assumed: graph dynamics: static graph; edge direction: undirected; faults: no faults; black hole appearance: the black hole is present from the start; knowledge of the graph: a map of the graph; marking nodes: tokens (pebbles) that agents carry and drop. **Not assumed:** timing: asynchronous; topology: arbitrary connected graph.

Measure	Lower bound	Best upper bound	Status
Agents	2 (Dobrev et al. 2006) Two agents.	2 (Flocchini et al. 2012b)	• tight
Cost	none known Same as with whiteboards.	$\Theta(n \log n)$ (Flocchini et al. 2012b)	• tight

Results.

- Dobrev et al. (2006) [lower bound]: agents 2. *Bottleneck:* agents lost in the black hole. *Holds when:* a single agent never suffices.
- Flocchini et al. (2012b) [current best]: agents 2, cost $\Theta(n \log n)$. *Assumes:* one identical pebble per agent. Introductory and current best.

16.14.5 Whiteboards and sense of direction

Adds asynchronous, + sense of direction, + whiteboards VERIFIED

With sense of direction two agents suffice, still with $\Theta(n^2)$ moves.

Assumed: graph dynamics: static graph; edge direction: undirected; faults: no faults; black hole appearance: the black hole is present from the start; knowledge of the graph: sense of direction; marking nodes: whiteboards at the nodes. **Not assumed:** timing: asynchronous; topology: arbitrary connected graph.

Measure	Lower bound	Best upper bound	Status
Agents	2 (Dobrev et al. 2006) Two agents.	2 (Dobrev et al. 2006)	• tight
Cost	none known $\Theta(n^2)$.	$O(n^2)$ (Dobrev et al. 2006)	• tight

Results.

- Dobrev et al. (2006) [lower bound]: agents 2. *Bottleneck:* agents lost in the black hole. *Holds when:* a single agent never suffices.
- Dobrev et al. (2006) [current best]: agents 2, cost $O(n^2)$. Introductory and current best.

16.14.6 Tokens, no map

Adds asynchronous, topology unknown, + tokens VERIFIED

With a constant number of tokens and no map, $\Delta + 2$ agents and 3 tokens are necessary and sufficient.

Claude (Anthropic) was used in cataloguing the problems.

Assumed: graph dynamics: static graph; edge direction: undirected; faults: no faults; black hole appearance: the black hole is present from the start; marking nodes: tokens (pebbles) that agents carry and drop. **Not assumed:** timing: asynchronous; knowledge of the graph: no knowledge of the topology (n is known); topology: arbitrary connected graph.

Measure	Lower bound	Best upper bound	Status
Agents	$\Delta + 2$ (Balamohan et al. 2016) $\Delta + 2$ with $O(1)$ tokens.	$\Delta + 2$ (Balamohan et al. 2016)	• tight
Cost	$\Omega(n^2/\Delta)$ (Balamohan et al. 2016) $O(mn)$ versus $\Omega(n^2/\Delta)$; move-optimal in sparse graphs.	$O(mn)$ (Balamohan et al. 2016)	• polynomial gap

Results.

- Balamohan et al. (2016) [lower bound]: agents $\Delta + 2$. *Bottleneck:* agents lost in the black hole. *Holds when:* $O(1)$ tokens; $\Delta + 1$ agents are not enough; 2 tokens never suffice.
- Balamohan et al. (2016) [lower bound]: cost $\Omega(n^2/\Delta)$. *Bottleneck:* asynchrony. *Holds when:* teams of $(2 - \varepsilon)\Delta$ agents.
- Dobrev et al. (2013) [introductory]: agents $\Delta + 1$. *Assumes:* enhanced tokens. First token-based exploration of an unknown dangerous graph; polynomial cost.
- Balamohan et al. (2016) [current best]: agents $\Delta + 2$, cost $O(mn)$. *Assumes:* 3 tokens; the number of agents is known (5 tokens otherwise).

16.14.7 Synchronous agents with a map

Adds + synchronous, + map VERIFIED

With synchrony, two agents and a map, the question becomes the fastest exploration schedule, which is NP-hard to find.

Assumed: graph dynamics: static graph; edge direction: undirected; faults: no faults; black hole appearance: the black hole is present from the start; timing: synchronous rounds; knowledge of the graph: a map of the graph. **Not assumed:** marking nodes: no marking; topology: arbitrary connected graph.

Measure	Lower bound	Best upper bound	Status
Agents	none known Two agents.	2 (Klasing et al. 2007)	• tight
Cost	*NP-hard* (Klasing et al. 2007) Approximation ratio $3\frac{3}{8}$ versus inapproximability 389/388.	$3\frac{3}{8}$-approx. (Klasing et al. 2007)	• polynomial gap

Results.

- Klasing et al. (2007) [lower bound]: cost *NP-hard*. *Bottleneck:* combinatorial hardness. *Holds when:* computing an optimal schedule for two agents; no 389/388-approximation unless P = NP (APX-hard).
- Czyzowicz et al. (2006) [introductory]: agents 2, cost 9.3-approx..
- Klasing et al. (2007) [current best]: agents 2, cost $3\frac{3}{8}$-approx..

Notes. On trees the problem has a 5/3-approximation (CPC 2007, not yet in the bibliography).

16.14.8 Eventually emerging black hole, synchronous agents, arbitrary graphs

Adds eventually emerging, + synchronous VERIFIED

On arbitrary graphs four co-located synchronous agents locate an eventually emerging black hole by moving in a cautious chain along a universal exploration sequence; whether three suffice is open.

Claude (Anthropic) was used in cataloguing the problems.

Assumed: graph dynamics: static graph; edge direction: undirected; faults: no faults; timing: synchronous rounds. **Not assumed:** black hole appearance: the black hole may appear at any time, at any node (at the home base only after the start); knowledge of the graph: no knowledge of the topology (n is known); marking nodes: no marking; topology: arbitrary connected graph.

Statement for this setting. The black hole appears at an unknown round and node (at the home base only after round 0) and stays. Agents have distinct identifiers and face-to-face communication, and use neither pebbles nor whiteboards. Time is counted from the appearance of the black hole.

Measure	Lower bound	Best upper bound	Status
Agents	3 (Bonnet et al. 2026b) 3 or 4.	4 (Kaur et al. 2026a) [preprint]	• tight
Cost	none known No time lower bound.	poly(n) (Kaur et al. 2026a) [preprint]	• unknown

Results.

- Bonnet et al. (2026b) [lower bound]: agents 3. *Bottleneck:* agents lost in the black hole. *Holds when:* ring bound (Theorem 1 of Bonnet et al. 2026b): a ring is a graph, and the bound holds even with whiteboards, knowledge of n and an oriented ring.
- Kaur et al. (2026a) [preprint] [current best]: agents 4, cost poly(n). *Assumes:* knows n (level 0 of this problem); without it, polynomial time if the black hole appears within polynomial time (Remark 3.1); $O(\log n)$ bits of memory per agent. Introductory and current best (Theorem 3.1, Remark 3.2): the agents follow a universal exploration sequence and replace each move by a cautious chain movement, after which the two pairs of agents stand on two adjacent nodes; the time is the length of a universal exploration sequence for n nodes. Any single-agent exploration can be simulated in the same way (Remark 3.3); with depth-first search and $O(\log \Delta)$ bits of storage per node it takes $O(m)$ rounds. A preprint.

Open questions.

Q1. Do three synchronous agents suffice on arbitrary graphs (Kaur et al. 2026a)?

Notes. Kaur, Saxena, Mandal and Mondal extend the model of (Bonnet et al. 2026b) from rings to arbitrary static graphs. Checked against Section 3 of the full text of (Kaur et al. 2026a) (arXiv v2).

16.14.9 Eventually emerging black hole, synchronous agents, oriented ring

Adds + sense of direction, + ring **to** Eventually emerging black hole, synchronous agents, arbitrary graphs (p. 299) VERIFIED

If the black hole may appear at any time, even at the home base after the first round, three synchronous agents are necessary and sufficient on an oriented ring, and they find it within $4n$ rounds of its appearance.

Assumed: graph dynamics: static graph; edge direction: undirected; faults: no faults; timing: synchronous rounds; knowledge of the graph: sense of direction; topology: ring or path. **Not assumed:** black hole appearance: the black hole may appear at any time, at any node (at the home base only after the start); marking nodes: no marking.

Statement for this setting. The black hole appears at an unknown round and node (at the home base only after round 0) and stays. Agents have distinct identifiers, unbounded memory and face-to-face communication, and the ring is oriented. Time is counted from the appearance of the black hole.

Measure	Lower bound	Best upper bound	Status
Agents	3 (Bonnet et al. 2026b) Exactly 3.	3 (Bonnet et al. 2026b)	• tight
Cost	none known $4n$ rounds; no time lower bound is proved.	$O(n)$ (Bonnet et al. 2026b)	• unknown

Claude (Anthropic) was used in cataloguing the problems.

Results.

- Bonnet et al. (2026b) [lower bound]: agents 3. *Bottleneck:* agents lost in the black hole. *Holds when:* two agents may be together only at the home base, or one appearance destroys both; one agent then cannot tell whether the other died at distance 1 or at distance 2 from the home base (Theorem 1); a ring of $n \geq 5$ nodes; holds even with a safe home base, whiteboards, knowledge of n and implicit termination.
- Bonnet et al. (2026b) [current best]: agents 3, cost $O(n)$. *Assumes:* knows n (level 0 of this problem); $n \geq 5$; smaller rings are handled separately. Introductory and current best (Algorithm A3, Theorem 4): at most $4n$ rounds after the appearance. Without knowing n, three agents need $O(n^2)$ rounds (Algorithm A4, Theorem 5); three agents with one pebble need $4n$ rounds (Algorithm A2, Theorem 3); four agents need $3n$ rounds, as two pairs that never meet, each performing a cautious walk (Algorithm A1, Theorem 2).

Open questions.

Q1. Is one algorithm optimal in every respect at once: three agents, no knowledge of n, no pebble and $O(n)$ rounds? Left open by (Bonnet et al. 2026b).

Notes. Bonnet, Bramas and Lamani. Since the home base may become the black hole, it is not a safe meeting point, and agents that gather anywhere risk being destroyed together. Checked against the full text of (Bonnet et al. 2026b) (SAND 2026), which omits some proofs and pseudo-code for space.

16.14.10 Eventually emerging black hole, asynchronous agents, oriented ring

Adds eventually emerging, asynchronous, + sense of direction, + ring VERIFIED

With asynchronous agents no algorithm can announce the location of an eventually emerging black hole once and for all; with implicit termination, n agents that never communicate locate it within 2n epochs.

Assumed: graph dynamics: static graph; edge direction: undirected; faults: no faults; knowledge of the graph: sense of direction; topology: ring or path. **Not assumed:** black hole appearance: the black hole may appear at any time, at any node (at the home base only after the start); timing: asynchronous; marking nodes: no marking.

Statement for this setting. As in the synchronous setting, but agents are asynchronous. No algorithm with explicit termination exists (Theorem 6 of Bonnet et al. 2026b): a node through which every agent is delayed long enough looks like a black hole. The task is therefore **implicit termination**: eventually every surviving agent that suspects a node suspects the black hole and never changes its suspicion, and at least one agent does. Time is counted in epochs from the appearance of the black hole.

Measure	Lower bound	Best upper bound	Status
Agents	3 (Bonnet et al. 2026b) n against 3; with two pebbles four agents suffice.	n (Bonnet et al. 2026b)	• polynomial gap
Cost	none known $2n$ epochs; (Bonnet et al. 2026b) call $O(n)$ asymptotically optimal without stating a lower bound.	$O(n)$ (Bonnet et al. 2026b)	• unknown

Results.

- Bonnet et al. (2026b) [lower bound]: agents 3. *Bottleneck:* agents lost in the black hole. *Holds when:* synchronous bound (Theorem 1), which holds with implicit termination and therefore for asynchronous agents.
- Bonnet et al. (2026b) [current best]: agents n, cost $O(n)$. *Assumes:* implicit termination; time in epochs, not moves. Introductory and current best without pebbles (Algorithm A5, Theorem 7): agent i tours the ring back and forth avoiding node u_i and always suspects u_i; once the black hole appears, every agent except the one assigned to it is eventually destroyed. At most $2n$ epochs, without communication.

Open questions.

Claude (Anthropic) was used in cataloguing the problems.

Q1. How many asynchronous agents without pebbles are needed with implicit termination: between 3 and n? With two pebbles four suffice (Bonnet et al. 2026b).

Notes. With two pebbles, four agents that do not know n locate the black hole within $5n$ epochs by cautious walks in pairs (Algorithm A6, Theorem 8). The impossibility of explicit termination does not hold for a black hole present from the start, where the last unexplored node can be isolated once all others are known to be safe (Bonnet et al. 2026b). Checked against the full text of (Bonnet et al. 2026b) (SAND 2026).

16.14.11 Byzantine companions, with a map

Adds + map VERIFIED

If up to f agents are Byzantine (they collude with the black hole and survive it), $2f + 2$ agents suffice with a map.

Assumed: graph dynamics: static graph; edge direction: undirected; faults: up to f weak Byzantine agents (arbitrary behaviour, cannot fake identifiers); black hole appearance: the black hole is present from the start; timing: synchronous rounds; knowledge of the graph: a map of the graph. **Not assumed:** marking nodes: no marking; topology: arbitrary connected graph.

Measure	Lower bound	Best upper bound	Status
Agents	none known Asymptotically optimal.	$2f + 2$ (Di Luna et al. 2025a)	• near-optimal
Cost	none known No time lower bound recorded.	$O(n + f)$ (Di Luna et al. 2025a)	• unknown

Results.

- Di Luna et al. (2025a) [current best]: agents $2f + 2$, cost $O(n + f)$. Introductory and current best; also with face-to-face communication. The agent counts are asymptotically optimal.

Notes. The Byzantine agents are weakly Byzantine: they cannot forge identities (confirmed by the catalogue owner, a co-author).

16.14.12 Byzantine companions, whiteboards, no map

Adds topology unknown, + whiteboards VERIFIED

Without a map but with whiteboards, $(f + 1)(\Delta + 1)$ agents locate the black hole in $O(m + f)$ rounds.

Assumed: graph dynamics: static graph; edge direction: undirected; faults: up to f weak Byzantine agents (arbitrary behaviour, cannot fake identifiers); black hole appearance: the black hole is present from the start; timing: synchronous rounds; marking nodes: whiteboards at the nodes. **Not assumed:** knowledge of the graph: no knowledge of the topology (n is known); topology: arbitrary connected graph.

Measure	Lower bound	Best upper bound	Status
Agents	none known Asymptotically optimal.	$(f + 1)(\Delta + 1)$ (Di Luna et al. 2025a)	• near-optimal
Cost	none known No time lower bound recorded.	$O(m + f)$ (Di Luna et al. 2025a)	• unknown

Results.

Claude (Anthropic) was used in cataloguing the problems.

- Di Luna et al. (2025a) [current best]: agents $(f+1)(\Delta+1)$, cost $O(m+f)$. With face-to-face communication instead: $(f+1)(\Delta+1)+3f+1$ agents, $O(mn+f)$ rounds.

16.14.13 Dynamic rings

Adds + tokens, + ring VERIFIED

In a 1-interval-connected ring three agents are necessary and sufficient; with pebbles they need $\Theta(n^{1.5})$ moves.

Assumed: graph dynamics: dynamic graph, connected in every round (1-interval connected); edge direction: undirected; faults: no faults; black hole appearance: the black hole is present from the start; timing: synchronous rounds; marking nodes: tokens (pebbles) that agents carry and drop; topology: ring or path. **Not assumed:** knowledge of the graph: no knowledge of the topology (n is known).

Measure	Lower bound	Best upper bound	Status
Agents	3 (Di Luna et al. 2025b) Three.	3 (Di Luna et al. 2025b)	• tight
Cost	$\Omega(n^{1.5})$ (Di Luna et al. 2025b) $\Theta(n^{1.5})$.	$O(n^{1.5})$ (Di Luna et al. 2025b)	• tight

Results.

- Di Luna et al. (2025b) [lower bound]: agents 3. *Bottleneck:* adversarial edge changes. *Holds when:* fewer agents fail even when co-located with whiteboards and identifiers; n must be known.
- Di Luna et al. (2025b) [lower bound]: cost $\Omega(n^{1.5})$. *Bottleneck:* adversarial edge changes. *Holds when:* co-located agents, whiteboards.
- Di Luna et al. (2025b) [current best]: agents 3, cost $O(n^{1.5})$. *Assumes:* co-located start, pebbles. Introductory and current best; with face-to-face communication or vision $\Theta(n^2)$.

Notes. The ring is oriented and each agent carries one pebble; n must be known (Di Luna et al. 2025b). With **scattered** agents (distinct arbitrary starting nodes, with visible identifiers) three agents with pebbles are still necessary and sufficient, and the cost rises to $\Theta(n^2)$ moves and rounds; the lower bound holds even with whiteboards. With only face-to-face communication, three scattered agents fail even on a static oriented ring (Di Luna et al. 2023) (Theorems 4 to 7 of its 2026 arXiv version; the OPODIS version, and Section 6 of the 2020 arXiv version of Di Luna et al. 2025b, state this impossibility for any constant number of agents). Arbitrary dynamic graphs with whiteboards (Kaur et al. 2026b) are a setting of their own.

16.14.14 Dynamic graphs with whiteboards

Adds + whiteboards VERIFIED

In an arbitrary dynamic graph from which at most one edge is missing per round, six co-located synchronous agents with whiteboards locate the black hole in $O(m^2)$ rounds, and three are not enough.

Assumed: graph dynamics: dynamic graph, connected in every round (1-interval connected); edge direction: undirected; faults: no faults; black hole appearance: the black hole is present from the start; timing: synchronous rounds; marking nodes: whiteboards at the nodes. **Not assumed:** knowledge of the graph: no knowledge of the topology (n is known); topology: arbitrary connected graph.

Statement for this setting. The graph is the footprint G (with n nodes and m edges) of a dynamic graph that is connected in every round; the results below assume in addition that at most μ edges are missing in each round, mostly $\mu = 1$. An agent at node v sees $\deg(v)$ in G but not which incident

Claude (Anthropic) was used in cataloguing the problems.

edges are missing, and learns that an edge is missing only when its attempt to cross it fails. Agents are synchronous, have distinct identifiers, read the memory of the agents at the same node, and every node has a whiteboard. The black hole is located when an agent terminates at a safe node next to it and outputs the port that leads to it.

Measure	Lower bound	Best upper bound	Status
Agents	4 (Kaur et al. 2026b)	6 (Kaur et al. 2026b)	• tight
	Between 4 and 6 agents when at most one edge is missing per round; with up to μ missing edges, between $2\mu + 2$ and 6μ, the upper bound in exponential time.		
Cost	none known	$O(m^2)$ (Kaur et al. 2026b)	• unknown
	$O(m^2)$ rounds; no lower bound on time.		

Results.

- Kaur et al. (2026b) [lower bound]: agents 4. *Bottleneck:* adversarial edge changes. *Holds when:* Theorem 4.1 with $\mu = 1$: three agents co-located at a safe node of K_3, one node of which is the black hole, fail; holds even with unbounded agent memory and unbounded whiteboards; agents that know neither n nor μ and learn that an edge is missing only by failing to cross it; with up to μ missing edges per round, $2\mu + 1$ co-located agents fail.
- Kaur et al. (2026b) [current best]: agents 6, cost $O(m^2)$. *Assumes:* $\mu = 1$; co-located agents with distinct identifiers; $O(\log n)$ bits per agent and per whiteboard; no knowledge of n. Introductory and current best (Theorem 6.3 of arXiv v2). Theorem 3.3 gives nine agents in the same time; (Kaur et al. 2026a) and (Bileski and Miller 2026) cite this nine-agent algorithm from the ICDCN 2025 version.

Open questions.

Q1. With at most one missing edge per round, do four or five co-located agents suffice? Three do not (Kaur et al. 2026b).

Q2. With scattered agents and at most one missing edge per round, do $2\deg(v_{BH}) + 1$ or $2\deg(v_{BH}) + 2$ agents suffice? $2\deg(v_{BH})$ do not (Kaur et al. 2026b) and $2\deg(v_{BH}) + 3$ do (Bileski and Miller 2026).

Q3. With up to μ missing edges per round, can $O(\mu)$ co-located agents locate the black hole in polynomial time? The known algorithm with 6μ agents takes exponential time (Kaur et al. 2026b).

Notes. Checked against the full texts of (Kaur et al. 2026b) (arXiv v2), (Kaur et al. 2026a) (arXiv v2) and (Bileski and Miller 2026) (arXiv v1). The three papers share the model: nodes are anonymous, ports are local, an agent at node v sees $\deg(v)$ in the footprint G but not which incident edges are missing, and learns that an edge was missing only when its move fails; Kaur, Saxena, Mandal and Mondal contrast this with the dynamic rings, cactuses and tori of earlier work, where an agent sees which incident edge is missing (Kaur et al. 2026b). Co-located agents read each other's memory. Kaur et al. first give nine co-located agents in $O(m^2)$ rounds (Theorem 3.3) and then six (Theorem 6.3), both with $O(\log n)$ bits per agent and per whiteboard (Kaur et al. 2026b). With at most μ missing edges per round (their f is μ here), $2\mu + 1$ co-located agents fail even with unbounded memory and whiteboards (Theorem 4.1), and 6μ agents succeed in exponential time (Theorem 5.2) (Kaur et al. 2026b).

Scattered agents (every agent starts at an arbitrary safe node; several agents may share a node), with $\mu = 1$ and $O(\log n)$ bits per agent and per whiteboard: $2\deg(v_{BH})$ agents cannot solve the problem (Theorem 2.1 of Kaur et al. 2026b), where $\deg(v_{BH})$ is the degree of the black hole in G. Kaur, Saxena, Mandal and Mondal solve it with $2\deg(v_{BH}) + 17$ agents in $O(m^2)$ rounds (Theorem 2.2 of Kaur et al. 2026a, the extended version of Kaur et al. 2025): every agent runs a depth-first search and crosses each new edge by a cautious walk of its own, writing its outgoing port on the whiteboard before leaving and erasing it after coming back; two marks for the same port reveal the black hole, so at most $2\deg(v_{BH})$ agents die, and a group of nine agents that meets switches to the algorithm for co-located agents. Bileski and Miller drop the switch: all agents run one exploration in which at most two agents wait at a missing

Claude (Anthropic) was used in cataloguing the problems.

edge, so once $2\deg(v_{BH})$ agents have died a survivor reaches a revealing mark; $2\deg(v_{BH})+3$ agents suffice in $O(m^2\deg(v_{BH}))$ rounds (Theorem 1 of Bileski and Miller 2026, a preprint that gives proof sketches and defers the full proofs to a journal version). For rings with pebbles see the setting for dynamic rings.



16.15 Exploration of dynamic graphs by teams

An adversary changes the graph over time; how many agents are needed to explore it?

First posed in 2016 (Di Luna et al. 2020b). Live exploration of dynamic rings was posed at ICDCS 2016.

Problem statement and measures

Agents must visit every node of a **dynamic graph**: a sequence of graphs $G_t = (V, E_t)$ on one node set, whose edges are removed round by round by an adversary subject to a connectivity constraint (**1-interval connectivity**: every G_t is connected; **temporal connectivity**: every node can reach every other through edges present at non-decreasing times). The primary measure is the **number of agents**; the secondary is time in rounds.

Measures: **Agents**, number of agents needed; **Time**, rounds.



(a) Start: a and b on one node; a goes clockwise, b the other way

(b) After N - 1 = 7 rounds every node is visited; both stop

Figure 16.15: Live exploration of a dynamic ring (Di Luna et al. 2020b): in every round an adversary removes at most one edge of the ring (1-interval connectivity; the missing edge is dashed red). Two agents start on one node, know $N = n = 8$ and walk in opposite directions, waiting while their next edge is missing; the adversary can block only one of them, so every round visits a new node, and after $N-1$ rounds both stop. One agent alone can be blocked forever.

A first look

Imagine guards who must check every station on a circular railway, while the operator closes some piece of track at any moment without warning. We model the railway as a graph: stations are **nodes**, the pieces of track between neighbouring stations are **edges**, and each guard is an **agent**. In the basic version the graph is a **ring** of n nodes (the nodes form one cycle, each joined by an edge to its two neighbours). Time runs in **rounds**: in each round every agent may cross one edge or wait. An **adversary** (an imagined opponent, the operator of the picture) chooses, in every round, which edges are **missing** (removed for that round), with one rule: the graph must stay connected in every round (**1-interval connectivity**); in a ring this means that at most one edge is missing per round. A graph that changes

Claude (Anthropic) was used in cataloguing the problems.

like this is a **dynamic graph**. The agents must together visit every node; in the version recorded here they also know an upper bound $N \geq n$ and must stop. We count the agents needed and the rounds.

1. Algorithm: Two agents walking in opposite directions. (Di Luna et al. 2020b) (1-interval-connected rings, p. 309) The simplest case: two agents start on the same node and know $N \geq n$.

1. The two agents set off in opposite directions around the ring (the paper shows how two identical agents can agree on this).
2. In every round each agent tries to cross the next edge in its direction; if the edge is missing, it waits (it is **blocked**) and tries again in the next round.
3. After $N - 1$ rounds both agents stop.

Why it works and what it costs. At most one edge is missing per round, and until the two agents have between them covered the whole ring they are trying to cross two different edges, so at least one of them moves forward, onto a node that nobody has visited. Each round therefore visits at least one new node, and the $n - 1$ other nodes are all visited within $n - 1 \leq N - 1$ rounds. (Di Luna et al. 2020b) handle the harder cases (different starting nodes, no common sense of left and right) with one extra rule, turning back after being blocked for long, and both agents stop after $3N - 6$ rounds.

Why it matters. It shows that a second agent turns an impossible task into an easy one, because the adversary can only block one of them at a time.

2. Impossibility: One agent can always be blocked. (Di Luna et al. 2020b) (1-interval-connected rings, p. 309) A single agent stands on a node of the ring and, in some round, decides to cross one of the two edges at its node. The adversary removes exactly that edge in that round. Removing one edge from a ring leaves a path, which is still connected, so this is allowed. The adversary can do this in every round, so the agent never leaves its starting node.

Why it holds. So at least two agents are needed (Di Luna et al. 2020b). The same paper also shows that the adversary can prevent two agents that start on different nodes from ever meeting, by removing an edge only at the moment the two would otherwise meet: the agents cannot rely on meeting to coordinate.

Why it matters. The number of agents becomes the main measure on dynamic graphs, because a few agents may be unable to explore at all, whatever time they are given.

What the community started looking at

In dynamic graphs the community first asked **how many agents** make exploration possible at all, under each connectivity assumption (Di Luna et al. 2020b; Gotoh et al. 2021; Saxena and Mondal 2026); time came second.

Further reading

Papers on this problem that are not yet placed in a setting.

- Adamson and Spirakis (2026). Exploration by teams when the whole sequence of snapshots is given in advance (offline), each agent crossing at most one present edge per round. In an *always S-connected* temporal graph every snapshot joins every node by a path to some node of a fixed set S; $|S|$ agents, one starting on each node of S, explore every such graph whose lifetime is polynomial in n, $|S|$ and the average degree of the underlying graph (Theorem 1, Corollary 1). Through a division of the underlying graph into parts with few boundary nodes, a single agent explores an always connected temporal graph whose underlying graph has treewidth tw

Claude (Anthropic) was used in cataloguing the problems.

within $O(n^{4/3}\,\mathrm{tw}^{9/2}\log^{5/2} n)$ rounds (Theorem 2). When every edge is present in exactly one snapshot, deciding whether one agent can explore is NP-hard (Theorem 5), and the minimum number of agents that explore a path is computed in $O(n)$ time, that for a cycle in polynomial time (Theorems 6 and 7). Full version (arXiv v2) read; (Carnevale 2026) cites $\mathrm{tw}^{11/2}$ for the brief announcement, which was not read.

- Bournat et al. (2017). Perpetual exploration of connected-over-time rings by anonymous FSYNC robots with persistent memory and local weak multiplicity detection: $k \geq 3$ robots for $n > k$ (Theorem III.1); two robots fail for $n > 3$ and one for $n > 2$ (Theorems IV.1, V.1), and suffice for $n = 3$ and $n = 2$ (Theorems IV.2, V.2). Read.
- Bournat et al. (2019). Self-stabilizing perpetual exploration of connected-over-time rings by FSYNC robots with distinct identifiers: three robots for rings of at least four nodes (Theorem 4.1); two robots are impossible for $n \geq 4$ (Theorem 3.1). Read.
- Feletti et al. (2026d). Extends the temporally connected setting of (Gotoh et al. 2021) to graphs with any number of black holes, unknown to the agents, which must visit every safe node while at least one agent survives. Semi-synchronous oblivious anonymous agents starting on arbitrary safe nodes explore perpetually with $2\eta' + \sum_{x\in B}\deg(x) + 1$ agents, where B is the set of black holes and η' the number of transient edges between safe nodes, using whiteboards of $O(\Delta)$ bits (Theorem 11); without black holes this is the $2\eta + 1$ of (Gotoh et al. 2021). Agents with identifiers that know the number of safe nodes also terminate (Theorem 16). Preprint; read.
- Saxena et al. (2026b). Exploration of 1-interval connected graphs of arbitrary topology, with port numbers reassigned in every round, under a semi-synchronous scheduler: in each round an adversary deactivates up to α agents (their p), and every agent is activated infinitely often. Agents have distinct identifiers, start on arbitrary nodes and know neither n nor k; nodes have no storage. With $k \leq (n-2)(\alpha+1)$ agents exploration is impossible, even with full visibility, global communication, node identifiers and storage, unbounded memory and knowledge of every parameter (Theorem 2.1, $n \geq 3$, $\alpha \geq 1$). With $(n-2)(\alpha+1)+1$ agents it stays impossible for agents that see their neighbouring nodes but communicate only face to face (Theorem 2.2, $n \geq 9$), and for agents that see only their own node but communicate globally (Theorem 2.3). Agents that see their neighbouring nodes and communicate globally explore whenever $k \geq (n-2)(\alpha+1)+1$, that is $\alpha \leq \lfloor (k-1)/(n-2) \rfloor - 1$, without knowing n, k or α; every agent eventually knows that exploration is complete, the total number of moves is $O(k)$ times the largest diameter of a snapshot, and each agent uses $O(\max\{\log n, \log \alpha\})$ bits (Theorem 3.1, $n \geq 4$). The number of agents is therefore tight for every $\alpha \geq 1$; whether global communication is necessary is left open. For $\alpha = 0$ the threshold $n-1$ is the bound for synchronous agents of (Saxena and Mondal 2025), as reported by (Saxena et al. 2026b). The timing axis has no semi-synchronous value, so the result is not placed in a setting. Read in the arXiv version (v1); theorem numbers refer to it.
- Sudo et al. (2026c). A single deterministic agent on T-interval-connected graphs of arbitrary topology whose nodes carry identifiers: the window must be $\Omega(m)$ (Theorem 3), and with a window of $O(\zeta m + n\log^2 n)$, where $\zeta = \ln n/(1+\ln(m/n))$, the agent explores and stops within $O((m-n+1)n + n\log^2 n)$ moves, optimal when $m - n + 1 = \Omega(\log^2 n)$ (Theorems 1 and 4). Placed in the setting *T-interval-connected graphs, labelled nodes* of single-agent exploration. Preprint; read.
- Ayoubi et al. (2026). In vertex-permuted rings, where every snapshot is a cycle through all n nodes in an order the adversary chooses at each step, up to $n-3$ agents starting at one node cannot even find a treasure, so they cannot explore, even with memory, chirality, a view of the whole snapshot and knowledge of the node labels (Theorem 1). If every two nodes are adjacent within every ξ steps and agents flag the nodes they visit, k agents visit all nodes within $\xi\lceil 2n/k \rceil$ steps (proof of Theorem 8). Preprint; read.
- Gotoh et al. (2020). A single agent on a 1-interval connected ring with partial knowledge of the future: it sees which edges within H hops will be present in each of the next S rounds. It can explore exactly when $H + S \geq n$ and $S \geq \lceil n/2 \rceil$ (Theorem 2), in $O(n^2)$ rounds, $O(n^2/H + nH)$ when S is large enough, and $O(n^2/H + n\log H)$ when $S \geq n-1$ (Theorems 1, 3), against $\Omega(n^2/H)$ for every S (Theorem 4). Read in full.

Relations to other problems

- **Variant of Collaborative exploration.** When an adversary removes edges, a single agent can be blocked forever; the question shifts from speed to **how many agents** are needed to visit every node.

Claude (Anthropic) was used in cataloguing the problems.

- **Related: Broadcast by agents in dynamic graphs.** The same adversary: in every round it may remove any edges as long as the snapshot stays connected (1-interval connectivity), and the primary measure is the number of agents needed in both problems.

Case tree

The root assumes nothing; going down, each edge adds the assumptions written in grey (+) or leaves an axis unassumed. Bold entries are catalogued settings with their status per measure (Agents, Time) and page; italic entries are branch points, named after the axis on which their branches differ. Axis order: Graph dynamics → Edge direction → Timing → Topology.

No assumptions

+ connected every round, + undirected, + synchronous
Connected at every step, any graph ●● p. 308

+ ring
1-interval-connected rings STANDARD SETTING ●● p. 309

any dynamics, + undirected, + synchronous
Temporally connected graphs, arbitrary topology ●● p. 309

What is known

The best upper bound against the best lower bound in each catalogued setting, in tree order. The settings follow in detail.

Setting	Agents	Time	p.
Connected at every step, any graph	● polynomial gap $2m - n + 1$ vs $m - n + 1$	● unknown $O(n^9)$ vs —	308
1-interval-connected rings STANDARD SETTING	● tight 3 vs 2	● unknown $3N - 6$ vs —	309
Temporally connected graphs, arbitrary topology	● tight $2\eta + 1$ vs $2\eta + 1$	● unknown — vs —	309

16.15.1 Connected at every step, any graph

Adds + connected every round, + undirected, + synchronous VERIFIED

Against an adaptive adversary that keeps every snapshot connected, $2m - n + 1$ silent agents explore any graph, and $m - n + 1$ are needed even with a central offline scheduler.

Assumed: graph dynamics: dynamic graph, connected in every round (1-interval connected); edge direction: undirected; timing: synchronous rounds. **Not assumed:** topology: arbitrary connected graph.

Measure	Lower bound	Best upper bound	Status
Agents	$m - n + 1$ (Dobrev et al. 2020a) $m - n + 1$ against $2m - n + 1$; the ratio is unbounded on graphs with few cycles.	$2m - n + 1$ (Dobrev et al. 2020a)	● polynomial gap
Time	none known No lower bound; $O(n^9)$ only with $O(n^4)$ agents.	$O(n^9)$ (Dobrev et al. 2020a)	● unknown

Results.

Claude (Anthropic) was used in cataloguing the problems.

- Dobrev et al. (2020a) [lower bound]: agents $m - n + 1$. *Bottleneck:* adversarial edge changes. *Holds when:* all agents start at one node (Theorem 2); the adversary blocks one agent on each non-tree edge of a DFS tree; holds even for a central offline scheduler that sees all agents but not the future snapshots.
- Dobrev et al. (2020a) [current best]: agents $2m - n + 1$. *Assumes:* sense of direction: from the labels of traversed edges an agent can name nodes; silent agents: a mutual-exclusion mechanism lets one agent per port move, and the others learn that they failed; any starting nodes. Introductory and current best (Theorem 1); whether this algorithm explores in polynomial time is not known.
- Dobrev et al. (2020a) [also]: agents $O(n^4)$, time $O(n^9)$. *Assumes:* agents also see how many agents are at their node, and their actions. Theorem 3: polynomial time with more agents.

Open questions.

Q1. Does the algorithm with $2m - n + 1$ agents explore in polynomial time? Posed in (Dobrev et al. 2020a).

Notes. Every snapshot is a connected spanning subgraph of an underlying graph with n nodes and m edges, chosen by an adversary that sees the agents' moves. Checked against the full text of (Dobrev et al. 2020a).

16.15.2 1-interval-connected rings

Adds + ring **to** Connected at every step, any graph (p. 308) STANDARD SETTING VERIFIED

One agent cannot explore a dynamic ring; two agents with a known bound on n explore in $3N - 6$ rounds.

Assumed: graph dynamics: dynamic graph, connected in every round (1-interval connected); edge direction: undirected; timing: synchronous rounds; topology: ring or path. **Not assumed:** —.

Measure	Lower bound	Best upper bound	Status
Agents	2 (Di Luna et al. 2020b) Two (with a bound on n).	3 (Das et al. 2022)	• tight
Time	none known $O(n \log n)$ with a landmark but no chirality; whether that is necessary is open.	$3N - 6$ (Di Luna et al. 2020b)	• unknown

Results.

- Di Luna et al. (2020b) [lower bound]: agents 2. *Bottleneck:* adversarial edge changes. *Holds when:* one agent cannot explore; anonymous agents without a landmark and with n unknown cannot explore with termination at all.
- Di Luna et al. (2020b) [introductory]: agents 2, time $3N - 6$. *Assumes:* a known bound $N \geq n$.
- Mandal et al. (2023) [also]: agents 3. *Assumes:* identifiers, no landmark. Explicit termination with three agents.
- Das et al. (2022) [also]: agents 3. *Assumes:* identifiers, no landmark, no edge-crossing detection.

Open questions.

Q1. With a landmark but no chirality, is $O(n \log n)$ time necessary? (Di Luna et al. 2020b)

16.15.3 Temporally connected graphs, arbitrary topology

Adds any dynamics, + undirected, + synchronous VERIFIED

Perpetual exploration of a temporally connected graph with η transient edges needs exactly $2\eta + 1$ agents.

Claude (Anthropic) was used in cataloguing the problems.

Assumed: edge direction: undirected; timing: synchronous rounds. **Not assumed:** graph dynamics: time-varying graph, only temporally connected; topology: arbitrary connected graph.

Measure	Lower bound	Best upper bound	Status
Agents	$2\eta + 1$ (Gotoh et al. 2021) $2\eta + 1$.	$2\eta + 1$ (Gotoh et al. 2021)	• tight
Time	none known —	—	• unknown

Results.

- Gotoh et al. (2021) [lower bound]: agents $2\eta + 1$. *Bottleneck:* adversarial edge changes. *Holds when:* 2η agents fail even with identifiers, knowledge of n and k, and unbounded whiteboards.
- Gotoh et al. (2021) [current best]: agents $2\eta + 1$. Introductory and current best; whiteboards of $O(\log \deg(v))$ bits at node v.



16.16 Uniform deployment on graphs

Spread $k \leq n$ agents that start on distinct nodes of an n-node graph so that they end evenly spread, on a ring every two consecutive agents at distance $\lfloor n/k \rfloor$ or $\lceil n/k \rceil$, and stop.

First posed in 2016 (Shibata et al. 2018). Agents that have memory but see only the node they are on, in asynchronous unidirectional rings (PODC 2016), as reported by (Shibata et al. 2022a); not read. Look–Compute–Move robots that see several nodes had been deployed on rings and grids before, as listed in (Shibata et al. 2022a).

Problem statement and measures

k agents start on distinct nodes of an anonymous graph G with $n \geq k$ nodes and must reach a configuration in which they are **evenly spread**, and stop there. On a **ring**, evenly spread means that every two agents that are consecutive along the ring are at distance $\lfloor n/k \rfloor$ or $\lceil n/k \rceil$; on other graphs the target is stated with each result (one side of a complete bipartite graph, or every node at chosen depths of a perfect tree). **Almost uniform deployment** asks this of all agents but one. Measures: **time** in rounds (ideal time units for asynchronous agents), the total number of **moves**, a move being the traversal of one edge by one agent, and **memory** in bits per agent.

Measures: **Time**, rounds (ideal time units for asynchronous agents) until the agents are evenly spread and have stopped; **Moves**, total number of edge traversals by all agents; **Memory**, bits of memory per agent.

A first look

Picture k guards on a circular wall with n towers, who must spread out so that the gaps between neighbouring guards are as equal as possible, and then stand still. The wall is a **ring**: a cycle of n **nodes** (the towers), each joined by an **edge** (a stretch of wall) to the next. The guards are **agents**: mobile entities with memory that move along edges and see only the node they stand on. The nodes have no names and the agents are identical. The $k \leq n$ agents start on distinct nodes and must become **evenly spread**: every two agents that are consecutive along the ring are $\lfloor n/k \rfloor$ or $\lceil n/k \rceil$ edges apart. Then they stop. In the standard version each edge can be crossed in one direction only (the ring is unidirectional), the agents are asynchronous (each acts at its own unpredictable speed), and they know k or n. We count

Claude (Anthropic) was used in cataloguing the problems.



(a) Start: A, B, C drop their tokens (squares) and walk clockwise for 3n = 18 rounds

(b) Final: every two consecutive agents are n/k = 2 edges apart; all stop

Figure 16.16: Almost uniform deployment on a dynamic ring, the algorithm of (Shibata et al. 2022a) for agents that know n: $k = 3$ identical agents on a ring of $n = 6$ nodes agree on the clockwise direction. In the selection phase each agent drops its token on its home node and walks clockwise for $3n$ rounds, blocked whenever the edge in front of it is missing (dashed red; at most one edge is missing per round). Having gone once around, it knows the gaps between consecutive tokens. The rotation of the gap sequence that is smallest in dictionary order, here $(1, 2, 3)$, starts at the same token for every agent: the base node. An agent whose own sequence must be rotated r times to reach it has rank r and walks to the node $r \cdot n/k$ edges clockwise from the base node (green circles), where it stops (green). In this run no edge is missing after round 3; when one stays missing, the agents still walking split into two groups that walk in opposite directions, and at most one agent ends out of place. One frame per round.

moves (a **move** is one agent crossing one edge), time and memory. Time is counted in **rounds**; in a round every agent may cross one edge.

1. Algorithm: Go once around, then agree on a base node. (Shibata et al. 2022a) (Dynamic rings, all agents but one, p. 315) This version is harder. The ring is **dynamic**: in every round an **adversary**, an opponent who knows the algorithm, may remove one edge. An agent that tries to cross the missing edge is **blocked** and stays where it is. The agents move in synchronous rounds, know n, and agree on which way is clockwise. Each agent can count the agents on its node, and carries a **token**, a marker it can leave on a node. A token carries no name.

1. Every agent leaves its token on its start node, its **home node**. It then walks clockwise for $3n$ rounds, waiting whenever it is blocked.
2. On the way it writes down the distances between consecutive tokens. Once it has gone around, this list of gaps describes the start, read from its own home node.
3. Every agent rotates its list to the rotation that comes first in dictionary order. That rotation starts at the same token for every agent, unless the start repeats itself around the ring (then there is one such token per repetition). The node of that token is the **base node**. The number of steps the agent rotated is its **rank** r.
4. If missing edges stopped an agent from going once around, the agents notice it: they have all piled up on one node. They count themselves there and take ranks in the order in which they arrived; that node becomes the base node.
5. The agent of rank r walks clockwise to the node $r \cdot n/k$ edges past the base node and stops there. If an edge stays missing, the agents still walking split into two groups that walk in opposite directions. The adversary can hold back only one group, so the other group arrives.

Why it works and what it costs. In $3n$ rounds an agent either goes once around or is blocked more than $2n$ times. Only one edge is missing per round, so while it is blocked every other agent gets one edge

Claude (Anthropic) was used in cataloguing the problems.

closer to it; in the second case they all end on its node (Lemma 1). Splitting halves the number of agents still walking each time, so after about $\log k$ phases of $O(n)$ rounds at most one agent is left out of place. The goal allows exactly that: **almost uniform deployment** asks that all agents but one be evenly spread. In total this takes $O(n \log k)$ rounds and $O(kn)$ moves, since the walking agents halve from phase to phase; each agent needs $O(k \log n)$ bits to store the list of gaps (Shibata et al. 2022a). The moves are optimal, and the time is within a factor $\log k$ of the lower bound $\Omega(n)$. On a ring whose edges never go missing, the same idea of going once around and then choosing gives $\Theta(n)$ time and $\Theta(kn)$ moves (Shibata et al. 2018), as reported by (Shibata et al. 2022a).

Why it matters. Anonymous agents that see only their own node can still agree on a common reference point by reading the same circular list from different starting points. That is the core of deployment on rings.

2. Impossibility: The adversary blocks the last agent. (Shibata et al. 2022a) (Dynamic rings, all agents, p. 314) Suppose some algorithm made all k agents evenly spread on every dynamic ring, with k dividing n.

1. Run it on a ring whose edges never go missing. Let a be the last agent to stop, at round t; the proof considers an a that is still walking towards its place in its last rounds.

2. When the agents stop, a's neighbours are n/k edges away. Two agents get at most two edges closer per round, so during its last $n/2k$ rounds a shares a node with no other agent.

3. Run the algorithm again, identically until $n/2k$ rounds before t. From then on the adversary removes the edge that a tries to cross, so a never moves again. One missing edge per round is allowed.

4. No other agent meets a in these rounds, so the others behave exactly as in the first run and stop at the same nodes.

Why it holds. In the second run, a stays on the node where it stood $n/2k$ rounds before t, short of the node where it stopped in the first run, while every other agent stops where it stopped before. So a ends in the wrong place and the algorithm fails (Shibata et al. 2022a). The argument still works when the agents know n and k, agree on the clockwise direction, count the agents on their node and carry tokens. This is why the task on dynamic rings asks only for almost uniform deployment.

Why it matters. It shows the adversary's weapon, holding one agent where nobody can see it, and it fixes the relaxed goal that the algorithm above reaches.

3. Lower bound: Distance: $\Omega(kn)$ moves. (Shibata et al. 2022a) (Dynamic rings, all agents but one, p. 315) Start the k agents on k consecutive nodes, with k small compared with n. In the end, consecutive agents are about n/k edges apart, so the agents are spread over the whole ring. The nodes at least $n/4$ edges from the starting block make up more than a third of the ring once k is small. Spread evenly, the agents put a fixed fraction of themselves there.

Why it holds. So a fixed fraction of the k agents each cross at least $n/4$ edges, in total $\Omega(kn)$ moves; leaving one agent out of place changes nothing. Such an agent needs $n/4$ rounds, which gives $\Omega(n)$ time. (Shibata et al. 2022a) state these bounds for $k \leq pn$ with a constant $p < 1$, following (Shibata et al. 2018). They already hold on a ring whose edges never go missing.

Why it matters. It shows that the $O(kn)$ moves of the algorithm above cannot be improved, so the remaining question is time: $O(n \log k)$ against $\Omega(n)$.

Claude (Anthropic) was used in cataloguing the problems.

What the community started looking at

Uniform deployment asks agents that start on distinct nodes to end evenly spread. Look–Compute–Move robots that see several nodes were deployed on rings and grids first; Shibata et al. studied agents that have memory but see only their own node, on asynchronous unidirectional rings, with $\Theta(n)$ time and $\Theta(kn)$ moves (Shibata et al. 2018) (as reported by Shibata et al. 2022a). On a dynamic ring the adversary can block one agent forever, so Shibata, Sudo, Nakamura and Kim ask for almost uniform deployment and reach it with an optimal number of moves (Shibata et al. 2022a). Other graphs have their own targets: every node at chosen depths of a perfect tree (Shibata and Tixeuil 2023), and one side of a complete bipartite graph (Shibata et al. 2025).

Further reading

Papers on this problem that are not yet placed in a setting.

- Shibata and Tixeuil (2023). Semi-uniform deployment on a perfect tree (every internal node has the same number of children, all leaves at the same depth): k opaque Look–Compute–Move robots with lights, on distinct nodes, must occupy every node at depths forming an arithmetic progression, k being the number of such nodes, without collisions, and stop (Definition 1). With one colour, that is oblivious robots, no algorithm does so even under FSYNC (Theorem 2); with two colours a collision-free algorithm works under ASYNC in $O(kD)$ rounds (Theorem 3), against $\Omega(D + k)$ (Theorem 1). Read in full.
- Shibata et al. (2025). Uniform deployment on the complete bipartite graph $K_{k,k}$ (their $K_{n,n}$; anonymous nodes, local port numbers): k oblivious, anonymous Look–Compute–Move robots without lights stand on distinct nodes and must occupy every node of one side, so that any two robots are at distance 2. A start with a bijection between the two sides that preserves what every node sees is unsolvable even under FSYNC (Theorem 2); with visibility range 1 the task cannot be solved even under FSYNC (Theorem 4); with visibility range $2\lceil \log k \rceil + 2$ the robots detect the unsolvable starts (Theorem 7) and from every other one finish in $O(1)$ epochs, each robot moving at most once, even under ASYNC (Theorem 9). Whether a constant visibility range suffices is open (Section 6). Read.

Relations to other problems

- **Related: Dispersion.** With $k = n$ agents, dispersion and uniform deployment ask for the same final configuration, one agent on every node; dispersion also allows agents to share nodes at the start, uniform deployment does not (Shibata et al. 2022a).
- **Related: Uniform deployment on a ring.** The same task for robots on a continuous ring in the plane, where the target spacing is a length rather than a number of edges (Flocchini et al. 2008b).

Assumptions specific to this problem

Goal

Exact deployment cannot be reached on a dynamic ring, where the adversary may block one agent forever, so Shibata, Sudo, Nakamura and Kim relax it to all agents but one (Shibata et al. 2022a). The relaxed goal is the weaker requirement, hence the higher level.

Level	Value	Formal definition
0	**all agents**	Every agent ends evenly spread and stops.
1	**all but one**	All agents but one end evenly spread and stop; the remaining agent may stop on any node.

Claude (Anthropic) was used in cataloguing the problems.

Case tree

The root assumes nothing; going down, each edge adds the assumptions written in grey (+) or leaves an axis unassumed. Bold entries are catalogued settings with their status per measure (Time, Moves, Memory) and page; italic entries are branch points, named after the axis on which their branches differ. Axis order: Graph dynamics → Edge direction → Identifiers → Randomness → Faults → Communication → Timing → Topology → Initial configuration → Goal.

No assumptions

+ connected every round, + undirected, + no faults, + synchronous, + ring, + dispersed start

Dynamic rings, all agents ●●● p. 314

+ all but one

Dynamic rings, all agents but one ●●● p. 315

+ static graph, + no faults, + ring, + dispersed start

Asynchronous unidirectional rings STANDARD SETTING ●●● p. 316

What is known

The best upper bound against the best lower bound in each catalogued setting, in tree order. The settings follow in detail.

Setting	Time	Moves	Memory	p.
Dynamic rings, all agents	● impossible	● impossible	● impossible	314
Dynamic rings, all agents but one	● near-optimal $O(n \log k)$ vs $\Omega(n)$	● tight $O(kn)$ vs $\Omega(kn)$	● unknown $O(k \log n)$ vs —	315
Asynchronous unidirectional rings STANDARD SETTING	● tight $\Theta(n)$ vs $\Omega(n)$	● tight $\Theta(kn)$ vs $\Omega(kn)$	● unknown $O(k \log n)$ vs —	316

16.16.1 Dynamic rings, all agents

Adds + connected every round, + undirected, + no faults, + synchronous, + ring, + dispersed start VERIFIED

On a 1-interval connected ring, synchronous agents cannot all end evenly spread: the adversary can block one agent forever.

Assumed: graph dynamics: dynamic graph, connected in every round (1-interval connected); edge direction: undirected; faults: no faults; timing: synchronous rounds; topology: ring or path; initial configuration: dispersed or scattered (one agent per occupied node). **Not assumed:** identifiers: anonymous: all agents are identical; randomness: deterministic; communication: none (silent agents; only see how many agents are co-located); goal: uniform deployment.

Measure	Lower bound	Best upper bound	Status
Time	*impossible* (Shibata et al. 2022a) Whenever k divides n; this is why the task is relaxed to almost uniform deployment.	—	● impossible
Moves	none known Whenever k divides n; this is why the task is relaxed to almost uniform deployment.	—	● impossible
Memory	none known Whenever k divides n; this is why the task is relaxed to almost uniform deployment.	—	● impossible

Results.

- Shibata et al. (2022a) [lower bound]: time *impossible*. *Bottleneck:* adversarial edge changes. *Holds when:* whenever k divides n (Theorem 1); take a run with no missing edge and the last agent to stop; in a second run the

Claude (Anthropic) was used in cataloguing the problems.

adversary removes the edge in front of that agent during its last $n/2k$ rounds, in which no other agent sees it, so the others stop as before and it stops in the wrong place; holds even for agents that know n and k, agree on the clockwise direction, count the agents on their node and carry tokens.

Notes. Checked against the full text of (Shibata et al. 2022a).

16.16.2 Dynamic rings, all agents but one

Adds + all but one **to** Dynamic rings, all agents (p. 314) VERIFIED

If one agent may end out of place, synchronous agents that know n spread evenly on a 1-interval connected ring in $O(n \log k)$ rounds with $O(kn)$ moves, which is optimal in moves.

Assumed: graph dynamics: dynamic graph, connected in every round (1-interval connected); edge direction: undirected; faults: no faults; timing: synchronous rounds; topology: ring or path; initial configuration: dispersed or scattered (one agent per occupied node); goal: almost uniform deployment. **Not assumed:** identifiers: anonymous: all agents are identical; randomness: deterministic; communication: none (silent agents; only see how many agents are co-located).

Measure	Lower bound	Best upper bound	Status
Time	$\Omega(n)$ (Shibata et al. 2022a) $O(n \log k)$ against $\Omega(n)$: a $\log k$ factor.	$O(n \log k)$ (Shibata et al. 2022a)	• near-optimal
Moves	$\Omega(kn)$ (Shibata et al. 2022a) $\Theta(kn)$ moves when n is known and $k \leq pn$.	$O(kn)$ (Shibata et al. 2022a)	• tight
Memory	none known No lower bound on memory is recorded.	$O(k \log n)$ (Shibata et al. 2022a)	• unknown

Results.

- Shibata et al. (2022a) [lower bound]: moves $\Omega(kn)$. *Bottleneck:* distance. *Holds when:* when $k \leq pn$ for a constant $p < 1$ (Theorem 3); already on a static ring, from a start with all agents close together, as in (Shibata et al. 2018).
- Shibata et al. (2022a) [lower bound]: time $\Omega(n)$. *Bottleneck:* distance. *Holds when:* Theorem 2; already on a static ring.
- Shibata et al. (2022a) [current best]: time $O(n \log k)$, moves $O(kn)$, memory $O(k \log n)$. *Assumes:* n known; agents agree on the clockwise direction; an agent counts the agents on its node (strong multiplicity detection); each agent carries one token, which it may leave on a node; agents exchange no messages. Introductory and current best (Theorem 4, Section 3): each agent leaves its token at its start node and moves forward for $3n$ rounds. Either it goes once around the ring and reads the distances between tokens, from which all agents pick the same base node (one per period if the start is periodic), or missing edges have piled all agents on one node, which they detect by counting. Each agent then moves to its target relative to the base node; one agent may stay blocked, which almost uniform deployment allows.
- Shibata et al. (2022a) [also]: time $O(n^2)$, moves $O(n^2)$, memory $O(k \log n)$. *Assumes:* $k \geq 4$ known, n unknown; agents agree on the clockwise direction; an agent counts the agents on its node (strong multiplicity detection); each agent carries one token, which it may leave on a node; agents exchange no messages. Theorem 5, Section 4: knowing k instead of n costs $O(n^2)$ rounds and moves; knowing n and knowing k are equivalent on static rings but not here.

Open questions.

Q1. Is $O(n)$ time possible on a 1-interval connected ring? (Shibata et al. 2022a) reach $O(n \log k)$ against $\Omega(n)$ and leave the lower bound on time to future work.

Q2. Can almost uniform deployment be reached without chirality or without strong multiplicity detection (Shibata et al. 2022a)?

Claude (Anthropic) was used in cataloguing the problems.

Notes. Shibata, Sudo, Nakamura and Kim assume eight things (Section 1.3): anonymous agents, knowledge of k or n, chirality, strong multiplicity detection, no messages, one token per agent, distinct start nodes and synchronous rounds, and show that anonymity, no messages, tokens and distinct starts are the weakest choices. Knowing n and knowing k are not equivalent here, unlike on static rings. Checked against the full text of (Shibata et al. 2022a).

16.16.3 Asynchronous unidirectional rings

Adds + static graph, + no faults, + ring, + dispersed start STANDARD SETTING DRAFT

On a static unidirectional ring, asynchronous agents that know k or n spread evenly in $\Theta(n)$ time with $\Theta(kn)$ moves, as reported.

Assumed: graph dynamics: static graph; faults: no faults; topology: ring or path; initial configuration: dispersed or scattered (one agent per occupied node). **Not assumed:** edge direction: directed (strongly connected); identifiers: anonymous: all agents are identical; randomness: deterministic; communication: none (silent agents; only see how many agents are co-located); timing: asynchronous; goal: uniform deployment.

Measure	Lower bound	Best upper bound	Status
Time	$\Omega(n)$ (Shibata et al. 2018) $\Theta(n)$, as reported.	$\Theta(n)$ (Shibata et al. 2018)	• tight
Moves	$\Omega(kn)$ (Shibata et al. 2018) $\Theta(kn)$, as reported.	$\Theta(kn)$ (Shibata et al. 2018)	• tight
Memory	none known No lower bound on memory is recorded.	$O(k \log n)$ (Shibata et al. 2018)	• unknown

Results.

- Shibata et al. (2018) [lower bound]: moves $\Omega(kn)$. *Bottleneck:* distance. *Holds when:* as reported by (Shibata et al. 2022a) (Section 2 and Table 1); not read.
- Shibata et al. (2018) [lower bound]: time $\Omega(n)$. *Bottleneck:* distance. *Holds when:* as reported by (Shibata et al. 2022a); not read.
- Shibata et al. (2018) [current best]: time $\Theta(n)$, moves $\Theta(kn)$, memory $O(k \log n)$. *Assumes:* k or n known. Introductory and current best, as reported by (Shibata et al. 2022a) (Table 1, Result 1); not read. Knowing n and knowing k are equivalent on static rings.
- Shibata et al. (2018) [also]: time $O(n \log k)$, moves $\Theta(kn)$, memory $O(\log n)$. *Assumes:* k or n known. Less memory at the price of a $\log k$ factor in time, as reported by (Shibata et al. 2022a) (Table 1, Result 2); not read.

Notes. Draft: (Shibata et al. 2018) has no open copy and was not read; its bounds are taken from Table 1 of (Shibata et al. 2022a), which also reports algorithms without knowledge of k or n for agents that can communicate, and a follow-up on memory (Shibata, Kakugawa and Masuzawa, TCS 2020, not in this atlas).



16.17 Network decontamination

A team of agents must clean a contaminated network, or equivalently capture an invisible, arbitrarily fast intruder, using as few agents as possible.

First posed in 2002 (Barrière et al. 2002). Capture of an intruder by mobile agents (SPAA 2002).

Claude (Anthropic) was used in cataloguing the problems.

Problem statement and measures

All nodes are initially **contaminated**. A node is **cleaned** when an agent visits it and is **recontaminated** when it has a contaminated neighbour and no agent **guards** it (stands on it). Agents move along edges from a **home base** s, the node where they all start; the clean part must stay connected. **Agents** is the number needed (the primary measure); **moves** (edge traversals) and time are secondary.

Measures: **Agents**, number of agents needed; **Moves**, total moves (or time, as stated).



(a) Start: all nodes contaminated; agents 1, 2 at s

(b) Last branch: the guard joins; every node is clean with 2 agents

Figure 16.17: Decontamination of a tree (Barrière et al. 2002) with the rule "guard the node, clean its branches one by one": every node is contaminated (red) and agents start at the home base s; a node is clean (green) once visited and would be recontaminated if left unguarded next to a red node. One agent guards s while the other cleans the smaller branch and comes back; then the guard is free, both enter the other branch, and the same rule is applied at a. Nothing is ever recontaminated, the clean part stays connected, and two agents suffice.

A first look

Imagine a network of pipes filled with a contaminating gas: cleaners must clean every junction, but gas flows back into any clean junction that is left unguarded next to one still full of gas. We model the network as a graph: junctions are **nodes**, pipes are **edges**, and each cleaner is an **agent**. At the start every node is **contaminated**. An agent makes a node **clean** by standing on it; a clean node with no agent on it becomes contaminated again as soon as one of its neighbours is contaminated (**recontamination**). An agent that stays on a clean node to prevent this is a **guard**. All agents start at one node s, the **home base**, and **move** along edges (one move is one traversal of one edge), and the clean part must always be connected. In the basic version the graph is a **tree** (a graph without cycles) whose map the agents know. The task: clean every node; we count the agents needed first, and moves and time second.

1. Algorithm: Guard the node, clean its branches one by one. (Trees, p. 321) A standard approach on a tree, rooted at the home base s. Write $a(v)$ for the number of agents this method uses to clean the **branch** of node v: v together with all nodes below it.

1. A node with no children (a **leaf**) needs one agent: $a(v) = 1$.
2. At a node v, sort the branches of its children so that the first one needs the most agents.
3. Keep one agent on v as a guard. Clean the other branches one after another, each with its own team; when a branch is clean, its team comes back to v to clean the next one.
4. Finally the guard is no longer needed at v (all its other neighbours are clean), so it joins the team for the first, largest branch.

Claude (Anthropic) was used in cataloguing the problems.

5. This gives $a(v) = \max\big(a(c_1),\ a(c_2)+1\big)$, where c_1 and c_2 are the children with the largest and second-largest needs.

Why it works and what it costs. No clean node is ever left unguarded next to a contaminated one, so nothing gets recontaminated, and the clean part always contains s and stays connected. A path needs one agent; a star centred at s needs two; a complete binary tree of height h needs $h+1$ with this rule, so about $\log n$ agents for n nodes. (Barrière et al. 2002) compute the smallest possible team for any tree, and a strategy for it, in linear time; this primer does not claim that the simple rule above always matches it.

Why it matters. The idea of a guard that waits at a node until its last branch is clean underlies decontamination strategies on trees and on many structured networks.

2. Lower bound: One agent cannot clean a cycle. (Unknown graphs, monotone strategies, p. 320) A standard argument. Let the graph be a ring and let one agent start at s. As soon as the agent steps to a neighbour, s is unguarded, and its other neighbour is still contaminated, so s becomes contaminated again. The agent can never have more than its current node clean.

Why it holds. So any graph containing a cycle needs at least two agents: one guards while the other advances. More generally, the team must be large enough to guard every clean node that touches a contaminated node at every moment, which is why the answer depends on the shape of the graph (for the hypercube, $\Theta(n/\sqrt{\log n})$ agents for n nodes Flocchini et al. 2008c).

Why it matters. It shows the basic counting idea of the problem: count the boundary between clean and contaminated parts that must be guarded.

3. Lower bound: The adversary grows the tree as you clean it. (Ilcinkas et al. 2009) (Unknown graphs, monotone strategies, p. 320) Now the agents do not know the graph, and they must never let a clean part become contaminated again (the strategy is **monotone**). An **adversary** (an imagined opponent) builds the graph while the agents clean it (Ilcinkas et al. 2009). Whenever an agent crosses an edge into the unknown, the adversary attaches at the far end a new node with two more unexplored edges. Every clean node that still has an unexplored edge must be guarded, because the edge may lead to contamination. When the graph has reached n nodes, the adversary stops adding edges: the result is a tree in which each node has at most three edges.

Why it holds. At the end, every parent of a leaf in this tree has had to be guarded, and there are at least about $n/4$ such nodes, so the protocol used at least about $n/4$ agents. A team that knew the tree in advance could clean it with only $O(\log n)$ agents. So not knowing the graph costs a factor $\Omega(n/\log n)$ in team size, and (Ilcinkas et al. 2009) give an algorithm that achieves this factor, even for asynchronous agents.

Why it matters. It is a clean example of a *competitive* lower bound: comparing a distributed team without a map to the best team with a map.

What the community started looking at

From the start the objective was the **minimum number of agents** (Barrière et al. 2002), first on special topologies (Flocchini et al. 2008c) and then on unknown graphs, where the question becomes the competitive ratio against the optimal team (Ilcinkas et al. 2009). Monotone strategies in dynamic graphs, whose edges may disappear, were studied by (Bar et al. 2026).

Further reading

Papers on this problem that are not yet placed in a setting.

Claude (Anthropic) was used in cataloguing the problems.

- Saini et al. (2026). Cops and a lethal robber on a ring: synchronous agents (cops) with distinct identifiers and $O(\log n)$ bits, placed at arbitrary nodes and knowing nothing about the ring, face an invisible robber that knows their algorithm, moves arbitrarily fast along paths without agents between the agents' moves, may not enter a node with an agent, and destroys every agent that enters its node; the agents win when they occupy both neighbours of the robber. n agents never suffice, even when they start at one node (Theorem 2.1), and $\max\{\sigma + \lfloor \log \sigma \rfloor + 4,\ n - \sigma + 4\}$ agents win within $O(n^3)$ rounds, where σ is the length of the part of the ring between the nearest agents that initially contains the robber (Theorem 4.1; their ρ is σ here); from a single starting node this is $n + \lfloor \log n \rfloor + 4$ (Remark 4.1). The robber is an arbitrarily fast intruder that also destroys agents, so the problem combines intruder capture with black hole search. Preprint; read.
- Giachoudis et al. (2026). Black virus decontamination of synchronous rings by initially scattered agents: a contaminated node that, when visited, destroys the agent and infects its unguarded neighbours. Abstract of the SIROCCO 2020 version read.
- Daadaa et al. (2016). Decontamination with temporal immunity: the immunity number, the least time a clean node can stay exposed to a contaminated neighbour, for one agent on several graph classes. Abstract read.
- Flocchini et al. (2016b). Decontamination with immunity: a clean node left without an agent is recontaminated only when at least a threshold number of its neighbours are contaminated (the usual model has threshold 1). For d-dimensional meshes, tori and trees, monotone algorithms whose team shrinks as the threshold grows (on a mesh one agent once the threshold reaches d, Theorem 1), lower bounds on the team that match for some thresholds (Theorems 2, 4, 5), and $\Theta(n)$ moves, which is optimal. The agents are synchronous; with one more agent the algorithms work asynchronously. Read (model and theorems).
- Luccio and Pagli (2016). Decontamination of butterfly networks with an immunity threshold, judged by work: the number of agents times the number of parallel steps, at least n. For some thresholds a protocol with a larger team needs less work than the protocol with the smallest team known; optimality is shown only in simple cases, and lower bounds on agents and steps are left open. Read (model, protocols and costs).
- Kameda et al. (2016). A continuous relative: one agent inside a simple polygon must detect an intruder that moves arbitrarily fast. An agent with two flashlights can search every polygon that an agent with a light in all directions can, as conjectured in 1992 and proved by Park et al. in 2001; a simpler proof through a 2-link visibility diagram of the polygon (Theorems 1–3). Read (statements and outline).

Relations to other problems

- **Related: Black hole search.** Both are security problems for agents in a hostile network: in black hole search the network destroys agents, in decontamination agents must clear a network that an arbitrarily fast intruder (or contamination) can re-enter.

Assumptions specific to this problem

Knowledge

Barrière, Flocchini, Fraigniaud and Santoro introduced contiguous search by mobile agents, in which agents cannot jump and the cleared part must stay connected; they solved it on trees with a linear-time algorithm that computes the minimum team and its strategy, and stressed that the algorithm is naturally distributed, so a tree network can compute its own strategy (Barrière et al. 2002) (abstract read). Later work on specific topologies also builds the strategy from the known structure; for hypercubes, Flocchini, Huang and Luccio use a coordinator that leads the other agents (Flocchini et al. 2008c). The unknown level asks what is lost when searchers must clear a graph they discover as they go. Blin, Fraigniaud, Nisse and Vial cleared any unknown asynchronous anonymous graph with whiteboards using at most one searcher more than the monotone connected search number from the homebase, each searcher with $O(\log k)$ bits of memory (Theorem 1); the clearing is connected but not necessarily monotone, and the number of moves may be exponential (Blin et al. 2008); Ilcinkas, Nisse and Soguet showed that if clearing must also be monotone, a protocol with no a priori information about the graph has competitive

Claude (Anthropic) was used in cataloguing the problems.

ratio $\Theta(n/\log n)$ in the number of searchers, the lower bound holding even for synchronous searchers on trees of maximum degree 3 (Ilcinkas et al. 2009).

Level	Value	Formal definition
0	**unknown graph**	Agents know nothing about G.
1	**known graph**	The topology is known to the agents (or to a coordinator).

Case tree

The root assumes nothing; going down, each edge adds the assumptions written in grey (+) or leaves an axis unassumed. Bold entries are catalogued settings with their status per measure (Agents, Moves) and page; italic entries are branch points, named after the axis on which their branches differ. Axis order: Graph dynamics → Edge direction → Timing → Knowledge → Topology.

No assumptions

- + static graph, + undirected
  - **Unknown graphs, monotone strategies** ●● p. 320
  - + known graph
    Topology
    - + tree
      - **Trees** STANDARD SETTING ●● p. 321
    - + hypercube
      - **Hypercubes** ●● p. 321
- + connected every round, + undirected, + synchronous
  - **Dynamic graphs, monotone strategies** ●● p. 322

What is known

The best upper bound against the best lower bound in each catalogued setting, in tree order. The settings follow in detail.

Setting	Agents	Moves	p.
Unknown graphs, monotone strategies	● tight $O(n/\log n)\cdot$ opt vs $\Omega(n/\log n)\cdot$ opt	● unknown — vs —	320
Trees STANDARD SETTING	● tight *optimal* vs —	● unknown — vs —	321
Hypercubes	● tight $O(n/\sqrt{\log n})$ vs $\Omega(n/\sqrt{\log n})$	● unknown $O(n\log n)$ vs —	321
Dynamic graphs, monotone strategies	● polynomial gap $D+2(m-n+1)$ vs $n-1$	● unknown $O(n^2)$ vs —	322

16.17.1 Unknown graphs, monotone strategies

Adds + static graph, + undirected VERIFIED

Clearing an unknown graph monotonically costs a factor $\Theta(n/\log n)$ in agents over the optimum.

Assumed: graph dynamics: static graph; edge direction: undirected. **Not assumed:** timing: asynchronous; knowledge: the topology is unknown; topology: arbitrary connected graph.

Claude (Anthropic) was used in cataloguing the problems.

Measure	Lower bound	Best upper bound	Status
Agents	$\Omega(n/\log n)\cdot$ opt (Ilcinkas et al. 2009) Competitive ratio $\Theta(n/\log n)$.	$O(n/\log n)\cdot$ opt (Ilcinkas et al. 2009)	• tight
Moves	none known —	—	• unknown

Results.

- Ilcinkas et al. (2009) [lower bound]: agents $\Omega(n/\log n)\cdot$opt. *Bottleneck:* unknown topology. *Holds when:* monotone connected strategies; holds even for synchronous agents.
- Ilcinkas et al. (2009) [current best]: agents $O(n/\log n)\cdot$ opt. Achievable even asynchronously.

16.17.2 Trees

Adds + tree **to** Unknown graphs, monotone strategies (p. 320) STANDARD SETTING VERIFIED

On trees the minimum team and a strategy are computed in linear time.

Assumed: graph dynamics: static graph; edge direction: undirected; knowledge: the topology is known; topology: tree. **Not assumed:** timing: asynchronous.

Measure	Lower bound	Best upper bound	Status
Agents	none known Optimal team size.	*optimal* (Barrière et al. 2002)	• tight
Moves	none known —	—	• unknown

Results.

- Barrière et al. (2002) [current best]: agents *optimal*. Introductory and current best; also a decentralised version with a linear number of messages.

16.17.3 Hypercubes

Adds + hypercube **to** Unknown graphs, monotone strategies (p. 320) VERIFIED

A hypercube needs $\Theta(n/\sqrt{\log n})$ agents, matching its pathwidth.

Assumed: graph dynamics: static graph; edge direction: undirected; knowledge: the topology is known; topology: hypercube. **Not assumed:** timing: asynchronous.

Measure	Lower bound	Best upper bound	Status
Agents	$\Omega(n/\sqrt{\log n})$ (Flocchini et al. 2008c) $\Theta(n/\sqrt{\log n})$.	$O(n/\sqrt{\log n})$ (Flocchini et al. 2008c)	• tight
Moves	none known —	$O(n\log n)$ (Flocchini et al. 2008c)	• unknown

Results.

- Flocchini et al. (2008c) [lower bound]: agents $\Omega(n/\sqrt{\log n})$. *Bottleneck:* pathwidth. *Holds when:* connected monotone strategies.
- Flocchini et al. (2008c) [current best]: agents $O(n/\sqrt{\log n})$, moves $O(n\log n)$. Introductory and current best (with a coordinator); with visibility, $n/2$ agents in $O(\log n)$ time.

Claude (Anthropic) was used in cataloguing the problems.

16.17.4 Dynamic graphs, monotone strategies

Adds + connected every round, + undirected, + synchronous DRAFT

In a 1-interval connected dynamic graph whose edges may stay missing indefinitely, $D + 2(m - n + 1)$ co-located synchronous agents clean an unknown graph monotonically, $m - n + 1$ being its cyclomatic number; on wheels n agents are needed.

Assumed: graph dynamics: dynamic graph, connected in every round (1-interval connected); edge direction: undirected; timing: synchronous rounds. **Not assumed:** knowledge: the topology is unknown; topology: arbitrary connected graph.

Statement for this setting. The graph is the footprint G (with n nodes, m edges and diameter D) of a dynamic graph that is connected in every round: the adversary may remove any edges as long as the graph stays connected, and a missing edge may stay missing for ever. $m - n + 1$ is the cyclomatic number of G, the number of edges outside a spanning tree. Nodes **and edges** start contaminated: an agent cleans a node by visiting it and an edge by crossing it, and a clean node or edge becomes contaminated again when a path without agents joins it to a contaminated node or edge. The strategy must be **monotone**. Agents start at a home base s chosen by the adversary, are synchronous, have distinct identifiers, communicate only at the same node, and know nothing about G; at a node they see which incident edges are present and which are contaminated.

Measure	Lower bound	Best upper bound	Status
Agents	$n - 1$ (Bar et al. 2026); $m - n + 2$ (Bar et al. 2026) $D + 2(m - n + 1)$ agents, which is $\Theta(n^2)$ on dense graphs, against $n - 1$ on $K_{n/2,n/2}$ and $m - n + 2 = n$ on wheels; Bar et al. state that their bounds are not tight.	$D + 2(m - n + 1)$ (Bar et al. 2026)	• polynomial gap
Moves	none known No lower bound on time.	$O(n^2)$ (Bar et al. 2026)	• unknown

Results.

- Bar et al. (2026) [lower bound]: agents $n - 1$. *Bottleneck:* guarding the boundary. *Holds when:* Theorem 1: $n - 2$ co-located agents fail on $K_{n/2,n/2}$, $n > 4$; holds even when every missing edge reappears within a bounded time.
- Bar et al. (2026) [lower bound]: agents $m - n + 2$. *Bottleneck:* adversarial edge changes. *Holds when:* Theorem 2 and Corollary 1: $m - n + 1$ co-located agents fail on a wheel with $n - 1$ rim nodes, where $m - n + 1 = n - 1$, starting at the hub; the adversary keeps every rim edge missing until each rim node is guarded, then restores them all; needs edges that may stay missing indefinitely.
- Bar et al. (2026) [current best]: agents $D + 2(m - n + 1)$, moves $O(n^2)$. *Assumes:* co-located agents with distinct identifiers; agents see which incident edges are present and which are contaminated. Introductory and current best (Theorem 5; time from Lemma 6). Every node becomes clean; an edge that never reappears cannot be cleaned (Remark 5).
- Bar et al. (2026) [also]: agents $D + m - n + 1$, moves $O(\delta(n + m))$. *Assumes:* every missing edge reappears within δ rounds. Theorem 4 and Corollary 2.
- Bar et al. (2026) [also]: agents n, moves $O(\delta n^2)$. *Assumes:* every missing edge reappears within δ rounds. Theorem 3: agents settle on the nodes in breadth-first order and one agent then cleans the remaining edges; fewer agents than $D + m - n + 1$ on dense graphs.

Open questions.

Q1. How many co-located agents are necessary and sufficient to clean every 1-interval connected dynamic graph monotonically, as a function of n, m and D? $D + 2(m - n + 1)$ suffice, and $n - 1$ (on $K_{n/2,n/2}$) and n (on wheels) can be necessary (Bar et al. 2026).

Claude (Anthropic) was used in cataloguing the problems.

Notes. Checked against the full text of (Bar et al. 2026) (arXiv v1; the cited CALDAM 2026 version was not read). Bar, Barik, Bhattacharya and Mandal call this model of dynamics *indefinite edge disappearance*; in their *finite time edge reappearance* model every missing edge reappears within δ rounds (their T is δ here, their k is $m - n + 1$ and their d is D). It is a special case, so its lower bounds hold here. The algorithm of Theorem 5 keeps one agent at each node with a missing incident edge until the edge reappears, and one more agent at a node with two or more visible contaminated edges, while the rest of the team continues a depth-first search. Lemma 1 states that on some graph of diameter D fewer than D co-located agents fail, in both models; it is not recorded as a bound here, because its proof (on a complete binary tree, Fig. 3) could not be confirmed.



16.18 Broadcast by agents in dynamic graphs

One agent holds a message; an adversary changes the graph every round, keeping it connected. How many agents are needed so that the message always reaches every agent?

First posed in 2020 (Das et al. 2026b). Posed at OPODIS 2020; the journal version appeared in Discrete Applied Mathematics (2026).

Problem statement and measures

$k \geq 2$ agents with distinct identifiers stand on distinct nodes, chosen by an **adversary**, of a connected undirected **graph** $G = (V, E)$ with n nodes. One agent, the **source agent**, holds a **message**; the other $k-1$ are **ignorant agents**. Time runs in synchronous **rounds**. At the start of every round the adversary chooses the **snapshot** G_t, any connected spanning subgraph of G (the dynamic graph is 1-interval connected); every agent then sees G_t and the position and status of every agent (**global visibility**), and moves along one edge of G_t or stays. An ignorant agent that is on the same node as a source agent receives the message and becomes a source agent; no message travels in any other way. **Broadcast** is solved when every agent holds the message. The question is, for each graph G, the least k with which the agents solve broadcast against every adversary and every initial placement; **time** is the number of rounds.

Measures: **Agents**, the least number k of agents, the source agent included, with which broadcast is solved against every adversary, in the worst case over the graphs of the setting with n nodes; **Time**, rounds until every agent holds the message.

A first look

Picture a building whose rooms are joined by corridors, and a saboteur who, each minute, may lock any corridors, as long as every room can still be reached from every other. The building is a **graph**: rooms are its **nodes** and corridors its **edges**. A few **agents** (small mobile machines) stand in different rooms. One of them, the **source agent**, carries a **message**; the others are **ignorant agents**. Time runs in **rounds**. At the start of each round the saboteur, called the **adversary**, chooses which edges are open; the open edges form the **snapshot** of that round, and the snapshot must connect all nodes. Every agent sees the whole snapshot and where every agent is and whether it holds the message (**global visibility**). Then each agent crosses one open edge or stays. An ignorant agent learns the message only by standing on the same node as an agent that holds it, and then holds it too. **Broadcast** is solved when every agent holds the message. The adversary also chooses where the agents start. The question is how many agents k (the source agent included) are needed on a given graph for the agents to always win.

Claude (Anthropic) was used in cataloguing the problems.



(a) Start: n − 1 = 4 agents, one empty node; agent 1 is the source agent

(b) Final (round 5): 1 and 4 meet; every agent holds the message

Figure 16.18: Broadcast with $n-1$ agents (Das et al. 2026b) on the complete graph K_5, where $n-1$ agents are also necessary. Agent 1 is the source agent (green), agents 2 to 4 are ignorant (grey), and one node is empty. In every round the adversary opens a snapshot (thick edges), here a path through all nodes chosen to keep the source agents away from the ignorant ones. With one empty node, some source agent and some ignorant agent are still at most two edges apart, and they meet. A meeting leaves two nodes empty, so in the next round the agents shift one edge along an open path to fill one of them. One frame per round.

1. Algorithm: Fill all but one node. (Das et al. 2026b) (Dynamic graph, any topology, p. 326) Suppose there are $k = n-1$ agents on a graph with n nodes, so at most one node is empty.

1. In every snapshot there is a path from a source agent to an ignorant agent, because the snapshot connects all nodes.
2. Take a shortest such path. Its inner nodes hold no agent that could shorten it, so at most one inner node, the empty one, lies on it: the source agent and the ignorant agent are at most two edges apart.
3. Both step towards each other along that path (or one steps onto the other's node) and they meet in this round. Now there is one more source agent.
4. Now two agents share a node, so two nodes may be empty. In the next snapshot, take a path from the shared node to an empty node; every agent on it moves one edge along it, so that again at most one node is empty. Repeat until no ignorant agent is left.

Why it works and what it costs. Every repetition turns at least one more ignorant agent into a source agent, so the agents win. The argument works on every graph, so $n-1$ agents always suffice (Das et al. 2026b). On the complete graph K_n this is also necessary: with only $n-2$ agents two nodes are empty, and the adversary keeps open a spanning tree in which every ignorant agent is at least three edges from every source agent, so no meeting ever happens. On sparser graphs far fewer agents suffice: three on every ring with at least five nodes, two on every tree.

Why it matters. It gives the upper bound $n-1$ that holds on every graph, and its counting (how close the source agent is forced to come to an ignorant agent) is the measure every later bound refines.

2. Impossibility: Two agents on a ring never meet. (Das et al. 2026b) (Dynamic rings, p. 328) Take a ring with $n \geq 5$ nodes, one source agent and one ignorant agent.

1. In every round the adversary removes one edge of the shorter path between the two agents. The snapshot is still a path through all nodes, so it is connected.
2. The two agents could meet in this round only if they were at most two edges apart along an open path.

Claude (Anthropic) was used in cataloguing the problems.

3. The open path between them is the longer way around the ring, which has at least three edges because $n \geq 5$.
4. So they never meet, whatever they do.

Why it holds. With two agents broadcast is impossible on every ring with at least five nodes. A third agent defeats this adversary. The two ignorant agents walk towards the source agent from opposite sides; the adversary can remove only one edge of the ring per round, so one of them gets closer in every round and eventually meets the source agent. The two source agents then walk around the ring in opposite directions and one of them meets the last ignorant agent. So three agents always solve broadcast (Das et al. 2026b).

Why it matters. It shows the adversary's only weapon, cutting the short way between a source agent and an ignorant agent, and how extra agents beat it by approaching from several sides.

What the community started looking at

Broadcast by agents was posed for **dynamic graphs** by Das, Giachoudis, Luccio and Markou, who asked for the least number of agents per topology: two on trees, three on rings, linear in n on grids, hypercubes and complete graphs, which suggested that the number of redundant edges decides the answer (Das et al. 2026b). Moses, Redlich and Stock showed that **edge density does not decide it**, nor edge connectivity (Moses Jr. et al. 2025), and recast the problem as a game between agents and adversary in the spirit of cops and robber, with general constructions for both winners and first bounds on the time (Moses Jr. et al. 2026).

Relations to other problems

- **Related: Exploration of dynamic graphs by teams.** The same adversary: in every round it may remove any edges as long as the snapshot stays connected (1-interval connectivity), and the primary measure is the number of agents needed in both problems.
- **Related: Gathering in graphs.** Every ignorant agent must meet a source agent, so any algorithm that gathers all agents on one node in a setting also solves broadcast there; broadcast asks only for the meetings that pass the message on.

Case tree

The root assumes nothing; going down, each edge adds the assumptions written in grey (+) or leaves an axis unassumed. Bold entries are catalogued settings with their status per measure (Agents, Time) and page; italic entries are branch points, named after the axis on which their branches differ. Axis order: Graph dynamics → Edge direction → Identifiers → Communication → Timing → Initial configuration → Topology.

No assumptions

+ connected every round, + undirected, + distinct IDs (eponymous), + local communication, + synchronous, + dispersed start
Dynamic graph, any topology STANDARD SETTING ●● p. 326

+ ring
Dynamic rings ●● p. 328

+ complete graph
Dynamic complete graphs ●● p. 328

Claude (Anthropic) was used in cataloguing the problems.

+ bipartite
Dynamic bipartite graphs ●● p. 329

+ hypercube
Dynamic hypercubes ●● p. 329

+ grid
Dynamic grids ●● p. 330

+ static graph, + undirected, + distinct IDs (eponymous), + local communication, + synchronous, + dispersed start
Static graph ●● p. 330

What is known

The best upper bound against the best lower bound in each catalogued setting, in tree order. The settings follow in detail.

Setting	Agents	Time	p.
Dynamic graph, any topology STANDARD SETTING	● tight $n-1$ vs $n-1$	● unknown — vs $\lceil D/2 \rceil$	326
Dynamic rings	● tight 3 vs 3	● unknown — vs —	328
Dynamic complete graphs	● tight $n-1$ vs $n-1$	● unknown — vs —	328
Dynamic bipartite graphs	● tight $n-3$ vs $n-3$	● unknown — vs —	329
Dynamic hypercubes	● tight $n-3$ vs $n/2$	● unknown — vs —	329
Dynamic grids	● tight $n-3$ vs $(\sqrt{n}-1)^2+1$	● unknown — vs —	330
Static graph	● tight 2 vs —	● tight $O(D)$ vs $\lceil D/2 \rceil$	330

16.18.1 Dynamic graph, any topology

Adds + connected every round, + undirected, + distinct IDs (eponymous), + local communication, + synchronous, + dispersed start STANDARD SETTING VERIFIED

On a 1-interval connected graph, $n-1$ agents always suffice, and the complete graph needs them all; the least number for a given graph is not characterised.

Assumed: graph dynamics: dynamic graph, connected in every round (1-interval connected); edge direction: undirected; identifiers: distinct identifiers (eponymous): agents have pairwise distinct identifiers; communication: local (co-located agents communicate); timing: synchronous rounds; initial configuration: dispersed or scattered (one agent per occupied node). **Not assumed:** topology: arbitrary connected graph.

Measure	Lower bound	Best upper bound	Status
Agents	$n-1$ (Das et al. 2026b); $\min_v \deg(v)$ (Moses Jr. et al. 2025); $\lvert B \rvert$ (Moses Jr. et al. 2025) $n-1$ in the worst case over graphs with n nodes; for a given graph the least number lies between 2 and $n-1$ and no characterisation is known.	$n-1$ (Das et al. 2026b)	● tight
Time	$\lceil D/2 \rceil$ (Moses Jr. et al. 2026) [preprint] No time bound is stated for general graphs.	—	● unknown

Claude (Anthropic) was used in cataloguing the problems.

Results.

- Das et al. (2026b) [lower bound]: agents $n-1$. *Bottleneck:* adversarial edge changes. *Holds when:* the complete graph K_n; with at most $n-2$ agents (their $n-3$ ignorant agents) two nodes stay empty, and the adversary keeps open a spanning tree in which every ignorant agent is at least three edges from every source agent (Lemma 6.1). Theorem 6.1 (their $k = n-2$ ignorant agents is $k = n-1$ agents here).
- Moses Jr. et al. (2025) [lower bound]: agents $\min_v \deg(v)$. *Bottleneck:* adversarial edge changes. *Holds when:* a 3-vertex-connected graph; in every round the adversary connects the source agent's node to the rest only through a path of two empty nodes u, r and a spanning tree of the graph without u and the source agent's node, so every ignorant agent is at least three edges away. Theorem 11: $\min_v \deg(v) - 2$ ignorant agents fail (their δ is $\min_v \deg(v)$ here).
- Moses Jr. et al. (2025) [lower bound]: agents $|B|$. *Bottleneck:* adversarial edge changes. *Holds when:* B a bond of pairwise disjoint edges (a minimal cut that is a matching); ignorant agents on one side, the source agent on the other; the adversary removes every edge of B with an agent at an endpoint. Theorem 12 and Corollary 13 (their $k \leq |B| - 2$ ignorant agents fail).
- Moses Jr. et al. (2026) [preprint] [lower bound]: time $\lceil D/2 \rceil$. *Bottleneck:* distance. *Holds when:* the adversary places a source agent and an ignorant agent at distance D; holds whenever the agents win. Lemma 22; Theorem 25 replaces D by the largest distance between a node and a set of nodes holding the source agents.
- Das et al. (2026b) [current best]: agents $n-1$. Observation 3.4: with at most one empty node some source agent and ignorant agent are within two edges in every snapshot, so one meeting happens per round. Introductory and current best in the worst case.

Open questions.

Q1. Which graph property determines the least number of agents that solve broadcast on a graph G? Edge density, edge connectivity and the number of redundant edges do not (Moses Jr. et al. 2025); is there an analogue of dismantlability, which characterises cop-win graphs? (Moses Jr. et al. 2026)

Q2. How many rounds do the agents need when they win, as a function of n and D? Only lower bounds of order D are known (Moses Jr. et al. 2026).

Q3. Can broadcast be solved with weaker agents than global visibility, for example agents that see only the source agent? (Das et al. 2026b)

Notes. On a tree the adversary can remove no edge, so two agents suffice (Observation 3.2 of Das et al. 2026b) and the time is $\Theta(D)$ as on a static graph; a tree of diameter D forces $\lceil D/2 \rceil$ rounds with two agents (Theorem 23 of Moses Jr. et al. 2026). On cactus graphs (every edge on at most one cycle) the least number of ignorant agents is the number of cycles shorter than 5 if there are no others, and otherwise the number of cycles plus one (Theorem 4.2 of Das et al. 2026b). An upper bound for a graph is also an upper bound for each of its connected spanning subgraphs (Observation 3.3 of Das et al. 2026b). Moses, Redlich and Stock show that no simple graph property decides the least number of agents (Moses Jr. et al. 2025): graphs with edge density (edges per node) $1 + O(1/\log^* n)$ that need $\Theta(n/\log^* n)$ agents (Theorem 4, Corollary 6); for every number of agents and every density above 1, graphs below that density that need that many agents (Theorem 7); graphs with large edge connectivity, or with constant edge density and edge connectivity, that need a linear number of agents (Theorem 8, Corollaries 9–10); generalized theta graphs (two nodes joined by internally disjoint paths) need exactly one agent per path plus the source agent when every path has length at least 3 in their terms (Theorems 2–3). In (Moses Jr. et al. 2026) the winner is preserved by several constructions: adding edges cannot help the agents (Lemma 6); gluing a graph won by the adversary onto any graph at a cut node keeps it won by the adversary (Theorems 9–10); contracting cut edges does not change the winner (Theorem 14). If every set of k nodes has a spanning tree whose one-step moves can always be mapped onto an isomorphic spanning tree ("k-spanning tree symmetry"), the adversary wins with k agents even when the agents

Claude (Anthropic) was used in cataloguing the problems.

choose their start (Theorem 19). Das et al. and (Moses Jr. et al. 2025) count ignorant agents with k (their k is $k-1$ here); (Moses Jr. et al. 2026) counts all agents, as here.

16.18.2 Dynamic rings

Adds + ring **to** Dynamic graph, any topology (p. 326) VERIFIED

Three agents are necessary and sufficient on every ring with at least five nodes; two suffice on smaller rings.

Assumed: graph dynamics: dynamic graph, connected in every round (1-interval connected); edge direction: undirected; identifiers: distinct identifiers (eponymous): agents have pairwise distinct identifiers; communication: local (co-located agents communicate); timing: synchronous rounds; initial configuration: dispersed or scattered (one agent per occupied node); topology: ring or path. **Not assumed:** —.

Measure	Lower bound	Best upper bound	Status
Agents	3 (Das et al. 2026b) Exactly 3 for $n \geq 5$.	3 (Das et al. 2026b)	• tight
Time	none known Not analysed.	—	• unknown

Results.

- Das et al. (2026b) [lower bound]: agents 3. *Bottleneck:* adversarial edge changes. *Holds when:* a ring with $n \geq 5$ nodes; the adversary removes an edge of the shorter path between the two agents in every round. Theorem 4.1.
- Das et al. (2026b) [current best]: agents 3. Theorem 4.1: if and only if, for $n \geq 5$; two agents suffice for $n < 5$. Introductory and current best.

16.18.3 Dynamic complete graphs

Adds + complete graph **to** Dynamic graph, any topology (p. 326) VERIFIED

On K_n exactly $n-1$ agents are needed: all nodes but one occupied.

Assumed: graph dynamics: dynamic graph, connected in every round (1-interval connected); edge direction: undirected; identifiers: distinct identifiers (eponymous): agents have pairwise distinct identifiers; communication: local (co-located agents communicate); timing: synchronous rounds; initial configuration: dispersed or scattered (one agent per occupied node); topology: complete graph. **Not assumed:** —.

Measure	Lower bound	Best upper bound	Status
Agents	$n-1$ (Das et al. 2026b) Exactly $n-1$.	$n-1$ (Das et al. 2026b)	• tight
Time	none known No time bound is stated.	—	• unknown

Results.

- Das et al. (2026b) [lower bound]: agents $n-1$. *Bottleneck:* adversarial edge changes. *Holds when:* with at most $n-2$ agents two nodes are empty; the adversary keeps open a spanning tree with every ignorant agent at least three edges from every source agent. Theorem 6.1 with Lemma 6.1.
- Das et al. (2026b) [current best]: agents $n-1$. Theorem 6.1 (upper bound from Observation 3.4). Introductory and current best.

Claude (Anthropic) was used in cataloguing the problems.

16.18.4 Dynamic bipartite graphs

Adds + bipartite **to** Dynamic graph, any topology (p. 326) VERIFIED

On bipartite graphs whose two sides have at least three nodes each, $n-3$ agents suffice, and complete bipartite graphs need them.

Assumed: graph dynamics: dynamic graph, connected in every round (1-interval connected); edge direction: undirected; identifiers: distinct identifiers (eponymous): agents have pairwise distinct identifiers; communication: local (co-located agents communicate); timing: synchronous rounds; initial configuration: dispersed or scattered (one agent per occupied node); topology: bipartite. **Not assumed:** —.

Measure	Lower bound	Best upper bound	Status
Agents	$n-3$ (Das et al. 2026b) Exactly $n-3$ in the worst case (complete bipartite graphs).	$n-3$ (Das et al. 2026b)	• tight
Time	none known Not analysed.	—	• unknown

Results.

- Das et al. (2026b) [lower bound]: agents $n-3$. *Bottleneck:* adversarial edge changes. *Holds when:* the complete bipartite graph with both sides of at least three nodes; with fewer agents at least four nodes are empty. Lemma 6.3 and Theorem 6.3 (their $n-4$ ignorant agents).
- Das et al. (2026b) [current best]: agents $n-3$. *Assumes:* both sides have at least three nodes; with a side of at most two nodes, two agents suffice on the complete bipartite graph (Theorem 6.2). Lemma 6.2 for complete bipartite graphs; every connected bipartite graph with these sides is a connected spanning subgraph of one, so the bound carries over (Observation 3.3). Introductory and current best.

16.18.5 Dynamic hypercubes

Adds + hypercube **to** Dynamic graph, any topology (p. 326) VERIFIED

On Q_j with $j \geq 3$ between $n/2$ and $n-3$ agents are needed; exactly 5 on Q_3.

Assumed: graph dynamics: dynamic graph, connected in every round (1-interval connected); edge direction: undirected; identifiers: distinct identifiers (eponymous): agents have pairwise distinct identifiers; communication: local (co-located agents communicate); timing: synchronous rounds; initial configuration: dispersed or scattered (one agent per occupied node); topology: hypercube. **Not assumed:** —.

Measure	Lower bound	Best upper bound	Status
Agents	$n/2$ (Das et al. 2026b) $\Theta(n)$ both, but the exact number is open for $j \geq 4$: between $n/2$ and $n-3$.	$n-3$ (Das et al. 2026b)	• tight
Time	none known Not analysed.	—	• unknown

Results.

- Das et al. (2026b) [lower bound]: agents $n/2$. *Bottleneck:* adversarial edge changes. *Holds when:* Q_j with $j \geq 3$; the adversary places all ignorant agents in the half of the hypercube opposite the source agent. Theorem 6.4 (their $n/2-1$ ignorant agents); on Q_3 Lemma 6.4 raises it to 5 agents.
- Das et al. (2026b) [current best]: agents $n-3$. Hypercubes are bipartite, so the bipartite bound applies (Lemma 6.2 with Observation 3.3); on Q_3 it gives 5 agents, which is tight (Theorem 6.5). Introductory and current best.

Open questions.

Q1. How many agents are needed on Q_j for $j \geq 4$: $n/2$, $n-3$, or in between? (Das et al. 2026b)

Claude (Anthropic) was used in cataloguing the problems.

16.18.6 Dynamic grids

Adds + grid **to** Dynamic graph, any topology (p. 326) VERIFIED

On a $\sqrt{n} \times \sqrt{n}$ grid between $(\sqrt{n}-1)^2+1$ and $n-3$ agents are needed; on a $2 \times b$ grid, b or $b+1$.

Assumed: graph dynamics: dynamic graph, connected in every round (1-interval connected); edge direction: undirected; identifiers: distinct identifiers (eponymous): agents have pairwise distinct identifiers; communication: local (co-located agents communicate); timing: synchronous rounds; initial configuration: dispersed or scattered (one agent per occupied node); topology: grid. **Not assumed:** —.

Measure	Lower bound	Best upper bound	Status
Agents	$(\sqrt{n}-1)^2+1$ (Das et al. 2026b) $\Theta(n)$ both; the difference is $2\sqrt{n}-5$ agents on the square grid, and one agent on a $2 \times b$ grid.	$n-3$ (Das et al. 2026b)	• tight
Time	none known Not analysed.	—	• unknown

Results.

- Das et al. (2026b) [lower bound]: agents $(\sqrt{n}-1)^2+1$. *Bottleneck:* adversarial edge changes. *Holds when:* an $a \times b$ grid with $a, b \geq 2$ needs $(a-1)(b-1)+1$ agents; the square grid is the case $a=b=\sqrt{n}$; the adversary puts the source agent in the two bottom rows with too few ignorant agents, $b-1$ ignorant agents on every other row, and opens an edge between the two parts only at an empty node (Lemmas 5.4–5.5). Lemma 5.5 and Theorem 5.3: fewer than $(a-1)(b-1)$ ignorant agents cannot solve broadcast; on a $2 \times b$ grid, b agents (Theorem 5.1).
- Das et al. (2026b) [current best]: agents $n-3$. Grids are bipartite, so the bipartite bound applies (Lemma 6.2 with Observation 3.3); on a $2 \times b$ grid $b+1$ agents suffice (Theorem 5.2). Introductory and current best.

Open questions.

Q1. Close the gap between $(\sqrt{n}-1)^2+1$ and $n-3$ agents on the $\sqrt{n} \times \sqrt{n}$ grid (Das et al. 2026b).

Notes. The grid of this setting is the model's square grid; the bounds hold for every $a \times b$ grid with $a, b \geq 2$ (a rows, b columns).

16.18.7 Static graph

Adds + static graph, + undirected, + distinct IDs (eponymous), + local communication, + synchronous, + dispersed start
VERIFIED

Without edge changes two agents suffice on every graph and broadcast takes $\Theta(D)$ rounds.

Assumed: graph dynamics: static graph; edge direction: undirected; identifiers: distinct identifiers (eponymous): agents have pairwise distinct identifiers; communication: local (co-located agents communicate); timing: synchronous rounds; initial configuration: dispersed or scattered (one agent per occupied node). **Not assumed:** topology: arbitrary connected graph.

Measure	Lower bound	Best upper bound	Status
Agents	none known Two agents, the least possible.	2 (Das et al. 2026b)	• tight
Time	$\lceil D/2 \rceil$ (Moses Jr. et al. 2026) [preprint] $\Theta(D)$ rounds.	$O(D)$ (Das et al. 2026b)	• tight

Results.

- Moses Jr. et al. (2026) [preprint] [lower bound]: time $\lceil D/2 \rceil$. *Bottleneck:* distance. *Holds when:* the adversary places the source agent and an ignorant agent at distance D; two agents close their distance by at most two per round. Lemma 22 and Theorem 23 (stated for the dynamic game; the argument uses only the placement).

Claude (Anthropic) was used in cataloguing the problems.

- Das et al. (2026b) [current best]: agents 2, time $O(D)$. Observation 3.1: every agent walks towards the source agent. Introductory and current best.

Notes. Global visibility makes the static case a shortest-path walk; the case is listed as the base against which the adversary's power is measured.



16.19 Delivery by energy-constrained agents

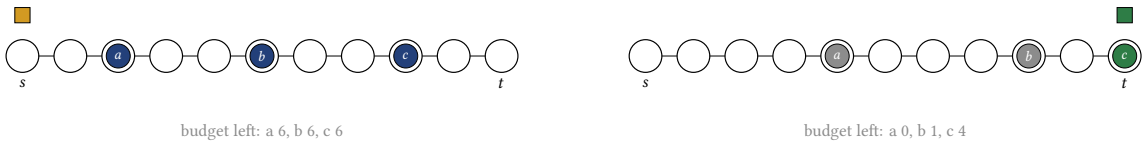
Agents with energy budgets, placed in a weighted graph, must carry a message from a source node to a target node, handing it over when they meet: when can a planner find a schedule quickly, and by how much must the budgets grow?

First posed in 2013 (Chalopin et al. 2013). Chalopin, Das, Mihalák, Penna and Widmayer first studied collaborative delivery, as (Chalopin et al. 2021) reports (ALGOSENSORS 2013). Anaya et al. had treated the transfer between two points of a path as part of convergecast with equal budgets (DISC 2012) (Anaya et al. 2016).

Problem statement and measures

k **agents** start at nodes of a connected undirected **graph** G with n **nodes** and m **edges**; every edge e has a positive **weight** $\mathrm{wt}(e)$, its length. Agent i has a **budget** B_i: it can travel a total distance of at most B_i, also along parts of edges. A **message** lies at a **source** node v_s and must reach a **target** node v_t. An agent can pick the message up where it is, carry it, and drop it anywhere (at a node or inside an edge), where another agent can pick it up; agents need not return to their start nodes. A **planner** that knows G, the start nodes and the budgets computes a **schedule**: which agents move, and where each one picks up and drops the message. The task is to decide whether a schedule exists and to find one. **Time** is the running time of the planner. The **budget factor** γ is that of a planner that runs in polynomial time and either correctly reports that no schedule exists or finds one in which every agent may travel γB_i (resource augmentation); with equal budgets $B_i = B$, it is the ratio between the budget the planner uses and the smallest budget that admits a schedule.

Measures: **Time**, running time of the planner; **Budget factor**, factor by which a polynomial-time planner enlarges the budgets (resource augmentation), or its approximation ratio for the smallest equal budget.



(a) Start: equal budgets B = 6; the message (square) lies at s

(b) Final: c brings the message to t; a schedule exists

Figure 16.19: Delivery on a path with equal budgets, by the greedy schedule of (Anaya et al. 2016): the edges have weight 1, every agent has budget $B = 6$, and the message must go from s to t. The agents are used from left to right. Each one walks to the message, picks it up and carries it to the right until its budget is used up, unless it reaches the start node of the next agent first, which then takes over with its whole budget. If any schedule exists, this one does (Lemma 1), so the planner decides the task in linear time. One frame per edge crossed; the numbers at the bottom are the budgets left.

Claude (Anthropic) was used in cataloguing the problems.

A first look

Picture couriers spread over a town, each able to walk only a limited total distance, who must carry a letter from one place to another, handing it over when two of them meet. The town is a **graph**: places are **nodes**, roads are **edges**, and each edge has a **weight**, its length. The couriers are **agents**, and agent i has a **budget** B_i, the total distance it may travel. The letter is the **message**. It lies at the **source** node v_s and must reach the **target** node v_t. An agent can pick the message up where it stands, carry it, and drop it anywhere, even partway along an edge, where another agent can pick it up. Agents need not go back to where they started. A **planner**, a program that knows the graph, the start nodes and the budgets, must decide whether a **schedule** exists and find one. A schedule says which agents move and where each one picks up and drops the message. We measure the planner's running **time**. For a fast planner that is allowed to enlarge the budgets, we also measure the **budget factor**: how much it must enlarge them. In the standard version the graph is arbitrary and every agent has its own budget.

1. Algorithm: On a path with equal budgets, push the message as far as possible. (Anaya et al. 2016) (Paths, equal budgets, p. 337) Let the graph be a path, its nodes in a row, with v_s to the left of v_t, and let all budgets be equal.

1. Take the agents in the order of their start nodes, from left to right.
2. The current agent walks to the message, picks it up and carries it to the right until its budget is used up, then drops it.
3. If it reaches the start node of the next agent on the way, it hands the message over there instead. The next agent still has its whole budget, so it can carry the message at least as far.
4. A schedule exists exactly when the message reaches v_t.

Why it works and what it costs. Each agent is used once, so once the agents are sorted the planner needs $O(k)$ steps for k agents (Anaya et al. 2016). The greedy schedule is enough because of an exchange argument (Lemma 1): if some schedule exists, then one exists in which the agents carry the message in order of position, each pushing it as far as possible. Equal budgets matter. When budgets differ, the planner must choose which agents to use, and delivery is NP-hard already on a path (Chalopin et al. 2014). That hardness is weak: it needs budgets that are large numbers.

Why it matters. It is the building block on which (Anaya et al. 2016) base convergecast and broadcast, and the easy case against which the hard ones are measured.

2. Algorithm: Triple the budgets: a chain of reaches. (Chalopin et al. 2013) (Arbitrary graphs, p. 335) Call the **reach** of agent i the set of all points within distance B_i of its start node.

1. Every hand-over point lies in the reach of both agents involved, v_s in the reach of the first agent and v_t in the reach of the last. So if no chain of agents, each reach meeting the next one, joins v_s to v_t, no schedule exists, and the planner says so.
2. Otherwise the planner takes such a chain and picks a meeting point in each pair of consecutive reaches.
3. Each agent of the chain walks to the previous meeting point (or to v_s), picks up the message, walks back to its start node, and then walks to the next meeting point (or to v_t).

Why it works and what it costs. Each of the three walks stays inside the agent's reach, so agent i travels at most $3B_i$. Finding a chain is a search in a graph whose nodes are the agents, so it takes polynomial time. Whenever the original budgets allow a schedule, the planner succeeds with budgets three times

Claude (Anthropic) was used in cataloguing the problems.

larger (Chalopin et al. 2013). Unless P = NP, no polynomial-time planner can make do with any smaller factor (Bärtschi et al. 2020). When all budgets are equal, doubling them is enough.

Why it matters. It locates the difficulty: deciding whether agents can reach each other is easy, and only the exact use of every budget is hard.

3. Lower bound: Hard choice: who carries which stretch. (Chalopin et al. 2013) (Arbitrary graphs, p. 335) Start from a puzzle known to be hard, **3-partition**: split $3z$ given numbers into z groups of three with the same sum. Build a graph from it.

1. A path from v_s to v_t is made of z stretches whose length is that sum, separated by single edges. On each separating edge sits an extra agent with budget 1, which can carry the message across it.
2. For each number a there is one agent. It spends a fixed amount to reach any node of the path, and its budget then lets it carry the message at most a edges further.
3. The path is so long that every agent must spend its whole budget on it. So the agents that carry the message across each stretch have numbers adding up to its length, and a schedule exists exactly when the numbers can be split.

Why it holds. A fast planner for delivery would therefore solve 3-partition, which is impossible unless P = NP (Chalopin et al. 2013). The numbers in 3-partition can be kept small, so delivery stays hard when every edge has weight 1 (it is strongly NP-hard). Equal budgets do not help either: each agent first walks a private path whose length makes up the difference. The hard part is choosing the order of the agents. Once the order is fixed, a planner finds a schedule in polynomial time (Bärtschi et al. 2020).

Why it matters. It is the reason this problem is studied through budget factors and special graphs rather than exact fast planners.

What the community started looking at

Anaya, Chalopin, Czyzowicz, Labourel, Pelc and Vaxès studied agents with equal budgets that gather information (convergecast) or spread it (broadcast) by meeting, and, as a building block, carrying information between two points of a path (Anaya et al. 2016). Chalopin, Das, Mihalák, Penna and Widmayer posed delivery from a source to a target with individual budgets on arbitrary graphs, showed it strongly NP-hard even with equal budgets, gave resource-augmented planners and a 2-approximation of the smallest equal budget, and introduced delivery along a fixed path (Chalopin et al. 2013); Chalopin, Jacob, Mihalák and Widmayer showed it weakly NP-hard already on a path (Chalopin et al. 2014). Bärtschi et al. showed it strongly NP-hard on planar graphs, made the factor 3 tight, and introduced returning delivery (Bärtschi et al. 2020). Later work fixed the path of the message (Chalopin et al. 2021), let the agents exchange energy (Czyzowicz et al. 2016), and minimised the total energy of agents with different consumption rates (Bärtschi et al. 2017).

Further reading

Papers on this problem that are not yet placed in a setting.

- Bärtschi et al. (2017). Minimises the total energy of agents that spend energy at different rates, without budgets, for several messages: one message in $O(|V|^3)$ time, collaboration helps by at most a factor 2, and a $4 \max_{i,j} w_i/w_j$-approximation with unit capacities (their w_i are rates). Model and contributions read (arXiv v1).
- Czyzowicz et al. (2016). Agents that may hand over energy when they meet: delivery, convergecast and broadcast take linear time on paths and polynomial time on trees, and stay NP-complete on arbitrary graphs. Theorem statements read (arXiv v1); summarised in the notes of the general setting.

Claude (Anthropic) was used in cataloguing the problems.

- Czyzowicz et al. (2019b). Broadcast without budgets: k agents at the root of an edge-weighted tree, one node holds a message that must reach every node, agents carry it on and need not return, and the total distance travelled is minimised. A greedy choice of the leaves where agents end computes the optimum in $O(n \log n)$ time for any k (Theorem 1), and in $O(n)$ time when k is at least the number of leaves (Theorem 2); the same applies to exploration. Read (model and theorems).
- Czyzowicz et al. (2021b). Gossiping with energy exchange: every node of an edge-weighted tree holds a packet, k agents placed anywhere with individual budgets carry copies of all packets they have seen and may hand over energy when they meet; can every packet reach every node? Some optimal strategy is a convergecast to one node followed by a broadcast (Theorem 7), and an $O(k^2n^2)$-time algorithm finds a cheapest strategy whenever one exists (Theorem 8). Read (model and theorems).

Relations to other problems

- **Related: Collaborative exploration.** Energy-constrained agents also explore graphs together: with budgets in unknown trees (Bampas et al. 2018), and with energy sharing in known graphs (Czyzowicz et al. 2021c); both are listed there as further reading.

Assumptions specific to this problem

Graph class

Chalopin, Das, Mihalák, Penna and Widmayer posed delivery on arbitrary graphs and left it open on paths with different budgets and on trees (Chalopin et al. 2013). Anaya et al. carried information along a path with equal budgets and asked whether the task is hard on trees (Anaya et al. 2016); Chalopin, Jacob, Mihalák and Widmayer found it hard on a path once budgets differ (Chalopin et al. 2014), and Bärtschi et al. showed hardness on planar graphs (Bärtschi et al. 2020).

Level	Value	Formal definition
0	**any graph**	G is any connected undirected graph with positive edge weights.
1	**planar**	G is planar.
2	**path**	G is a path, equivalently a segment of the line with the agents, v_s and v_t on it.

Budgets

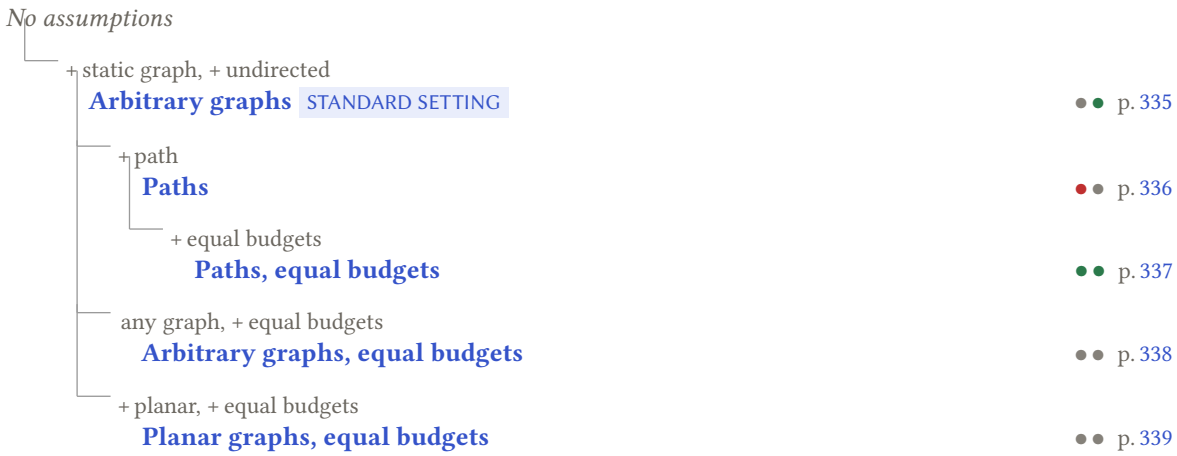
Anaya et al. give all agents the same power (Anaya et al. 2016); Chalopin et al. allow different budgets (Chalopin et al. 2013), which alone makes delivery on a path NP-hard (Chalopin et al. 2014). Chalopin, Das, Disser, Labourel and Mihalák call agents with equal budgets homogeneous (Chalopin et al. 2021).

Level	Value	Formal definition
0	**any budgets**	Each agent i has its own budget B_i.
1	**equal budgets**	All agents have the same budget B.

Claude (Anthropic) was used in cataloguing the problems.

Case tree

The root assumes nothing; going down, each edge adds the assumptions written in grey (+) or leaves an axis unassumed. Bold entries are catalogued settings with their status per measure (Time, Budget factor) and page; italic entries are branch points, named after the axis on which their branches differ. Axis order: Graph dynamics → Edge direction → Graph class → Budgets.



What is known

The best upper bound against the best lower bound in each catalogued setting, in tree order. The settings follow in detail.

Setting	Time	Budget factor	p.
Arbitrary graphs STANDARD SETTING	● unknown $\mathrm{poly}(n,m)$ vs *NP-hard*	● tight $\min\{3,\ 1+\max_{i,j} B_i/B_j\}$ vs 3	335
Paths	● polynomial gap $O\big(\mathrm{dist}(v_s,v_t)^2\, k^{1+4\log \mathrm{dist}(v_s,v_t)}\big)$ vs *weakly NP-hard*	● unknown — vs —	336
Paths, equal budgets	● tight $O(k)$ vs —	● tight — vs —	337
Arbitrary graphs, equal budgets	● unknown — vs *NP-hard*	● unknown 2 vs —	338
Planar graphs, equal budgets	● unknown — vs *NP-hard*	● unknown — vs —	339

16.19.1 Arbitrary graphs

Adds + static graph, + undirected

STANDARD SETTING VERIFIED

Deciding whether a schedule exists is strongly NP-hard; tripling the budgets makes it polynomial, and no smaller factor does unless P = NP.

Assumed: graph dynamics: static graph; edge direction: undirected. **Not assumed:** graph class: any connected graph; budgets: budgets may differ.

Claude (Anthropic) was used in cataloguing the problems.

Measure	Lower bound	Best upper bound	Status
Time	*NP-hard* (Chalopin et al. 2013)	poly(n, m) (Bärtschi et al. 2020)	● unknown
	Strongly NP-hard; exact planners are known to run in polynomial time only for a constant number of agents.		
Budget factor	3 (Bärtschi et al. 2020)	$\min\{3,\ 1 + \max_{i,j} B_i/B_j\}$ (Chalopin et al. 2013)	● tight
	Factor 3 against $3 - \varepsilon$ (unless P = NP).		

Results.

- Chalopin et al. (2013) [lower bound]: time *NP-hard*. *Bottleneck:* combinatorial hardness. *Holds when:* strongly NP-complete (Theorem 1), by a reduction from 3-partition with polynomially bounded numbers; holds even on unweighted graphs and with equal budgets.
- Bärtschi et al. (2020) [lower bound]: budget factor 3. *Bottleneck:* combinatorial hardness. *Holds when:* no polynomial-time planner with factor $3 - \varepsilon$, for any $\varepsilon > 0$, unless P = NP (Theorem 6), by a reduction from 3SAT built on the planar construction.
- Chalopin et al. (2013) [current best]: budget factor $\min\{3,\ 1 + \max_{i,j} B_i/B_j\}$. Introductory and current best (Corollary 1). Call the points within distance B_i of agent i's start node its reach. If no chain of agents with overlapping reaches joins v_s to v_t, no schedule exists; otherwise each agent of the chain fetches the message in the reach of the previous one, returns to its start node and carries the message into the reach of the next one, which costs at most $3B_i$ (Theorem 3). The factor $1 + \max_{i,j} B_i/B_j$ comes from the 2-approximation for equal budgets. Their R_i is B_i here, and their Δ is $\max_{i,j} B_i/B_j$.
- Bärtschi et al. (2020) [current best]: time poly(n, m). *Assumes:* k constant (Corollary 1). Theorem 7: if the order in which the agents carry the message is fixed, the planner computes, agent by agent, the points where the message can be, in $O(k(n+m)(n \log n + m))$ time; trying every order gives Corollary 1. (Chalopin et al. 2013) had shown this when agents hand the message over only at nodes (Theorem 2). The hardness thus lies in choosing the order.

Open questions.

Q1. What is the complexity of delivering several messages at once, for instance on paths? Raised in (Bärtschi et al. 2020).

Notes. Returning delivery. Bärtschi, Chalopin, Das, Disser, Geissmann, Graf, Labourel and Mihalák also require every agent to come back to its start node (Bärtschi et al. 2020). This variant is not a setting here, since neither variant's schedules or hardness carry over to the other. On trees it takes $O(n + k \log k)$ time: each agent is moved to the path between v_s and v_t, and the path is covered greedily by intervals of length half the remaining budget (Theorem 1). It is strongly NP-hard on planar graphs (Theorem 5), and a factor 2 suffices and is best possible unless P = NP (Theorems 2 and 6); with equal budgets a factor $2 - 2/k$ suffices (Theorem 3). **Energy exchange.** If agents may hand over energy when they meet, delivery takes linear time on paths and trees but stays NP-complete on arbitrary directed and undirected graphs (Theorems 3, 7, 13 and 14 of Czyzowicz et al. 2016). Theorem numbers of (Bärtschi et al. 2020) refer to its arXiv v1. Theorem numbers of (Chalopin et al. 2013) refer to the authors' copy.

16.19.2 Paths

Adds + path **to** Arbitrary graphs (p. 335) VERIFIED

On a path, delivery with different budgets is weakly NP-complete; a quasi-pseudo-polynomial algorithm decides it.

Assumed: graph dynamics: static graph; edge direction: undirected; graph class: paths. **Not assumed:** budgets: budgets may differ.

Claude (Anthropic) was used in cataloguing the problems.

Measure	Lower bound	Best upper bound	Status
Time	*weakly NP-hard* (Chalopin et al. 2014)	$O\big(\mathrm{dist}(v_s, v_t)^2\, k^{1+4\log \mathrm{dist}(v_s,v_t)}\big)$ (Chalopin et al. 2014)	• polynomial gap
	Weakly NP-hard against a quasi-pseudo-polynomial algorithm; whether a pseudo-polynomial algorithm exists is open (Chalopin et al. 2014).		
Budget factor	none known	—	• unknown
	The factor $\min\{3,\ 1 + \max_{i,j} B_i/B_j\}$ of arbitrary graphs applies, as (Chalopin et al. 2013) notes; no lower bound is known, and finding better planners is open (Chalopin et al. 2014).		

Results.

- Chalopin et al. (2014) [lower bound]: time *weakly NP-hard*. *Bottleneck:* combinatorial hardness. *Holds when:* NP-complete even when all positions and budgets are integers (Theorem 3), by a reduction from a weighted variant of 4-partition (Theorem 2); budgets differ: with equal budgets a path is easy (Anaya et al. 2016).
- Chalopin et al. (2014) [current best]: time $O\big(\mathrm{dist}(v_s, v_t)^2\, k^{1+4\log \mathrm{dist}(v_s,v_t)}\big)$. *Assumes:* integer positions and budgets. Introductory and current best (Theorem 1): a dynamic program over normalised schedules; the paper also states the bound $O(\mathrm{dist}(v_s, v_t)^2\, k^{1+4\log(B_{\max}+1)})$, with $B_{\max}$ the largest budget (their n is k here, and their Δ is $\mathrm{dist}(v_s, v_t)$).

Open questions.

Q1. Is there a pseudo-polynomial algorithm for delivery on a path? Posed in (Chalopin et al. 2014).

Q2. Find good resource-augmented planners for delivery on a path. Posed in (Chalopin et al. 2014).

Notes. Chalopin, Jacob, Mihalák and Widmayer answered a question of Anaya et al. and note that the hardness surprised everyone they had asked (Chalopin et al. 2014). No agent needs to carry the message twice, so a schedule is an order of agents. Returning delivery on a path (and on trees) takes $O(n + k \log k)$ time (Theorem 1 of Bärtschi et al. 2020); with energy exchange delivery on a path takes linear time (Theorem 3 of Czyzowicz et al. 2016). Checked against the authors' copy of (Chalopin et al. 2014).

16.19.3 Paths, equal budgets

Adds + equal budgets **to** Paths (p. 336) VERIFIED

With equal budgets, a greedy schedule in which each agent pushes the message as far as it can decides delivery on a path in linear time.

Assumed: graph dynamics: static graph; edge direction: undirected; graph class: paths; budgets: equal budgets. **Not assumed:** —.

Measure	Lower bound	Best upper bound	Status
Time	none known	$O(k)$ (Anaya et al. 2016)	• tight
	Linear in the number of agents.		
Budget factor	none known	—	• tight
	The planner is exact (factor 1).		

Results.

- Anaya et al. (2016) [current best]: time $O(k)$. *Assumes:* agents sorted by position. Introductory and current best (Lemma 1 and Remark 1): if any schedule exists, so does a regular one in which the agents, in order of position, each push the message as far as their budget allows; computing it agent by agent decides the task, which the paper calls carry. It is a building block of convergecast and broadcast, for which the paper computes the smallest equal budget on a path in linear time (Theorems 1 and 2).

Claude (Anthropic) was used in cataloguing the problems.

Open questions.

Q1. Is delivery with equal budgets between two nodes of a tree NP-hard or polynomial? Posed in (Anaya et al. 2016) and again in (Chalopin et al. 2013); (Chalopin et al. 2021) states that on trees delivery is known to be weakly NP-hard, without a reference, in a paper about equal budgets.

Notes. Anaya, Chalopin, Czyzowicz, Labourel, Pelc and Vaxès also show that the smallest equal budget for convergecast and for broadcast is NP-hard to compute on trees, approximate it within 2 and 4 on arbitrary graphs, and give distributed algorithms on trees that need 2 and 4 times the budget of a planner, 2 being best possible already on paths (Anaya et al. 2016). Checked against the full text (arXiv v1). Chalopin, Das, Mihalák, Penna and Widmayer state the same greedy rule: start with the agent closest to v_s, never overtake an unused agent, and use the closest unused agent next (Chalopin et al. 2013).

16.19.4 Arbitrary graphs, equal budgets

Adds any graph, + equal budgets **to** Arbitrary graphs (p. 335) VERIFIED

With equal budgets, deciding whether a schedule exists stays strongly NP-hard; the smallest budget that admits a schedule can be approximated within a factor 2, and no lower bound on the factor is known.

Assumed: graph dynamics: static graph; edge direction: undirected; budgets: equal budgets. **Not assumed:** graph class: any connected graph.

Measure	Lower bound	Best upper bound	Status
Time	*NP-hard* (Chalopin et al. 2013) Strongly NP-hard, even on unweighted graphs (Theorem 1 of Chalopin et al. 2013) and on planar graphs (Theorem 5 of Bärtschi et al. 2020).	—	• unknown
Budget factor	none known No inapproximability bound above 1 is known (Chalopin et al. 2021).	2 (Chalopin et al. 2013)	• unknown

Results.

- Chalopin et al. (2013) [lower bound]: time *NP-hard. Bottleneck:* combinatorial hardness. *Holds when:* strongly NP-complete (Theorem 1), by a reduction from 3-partition; holds even on unweighted graphs: each agent's start node gets a pendant path of length $B - B_i$, and the agent starts at its end.
- Chalopin et al. (2013) [current best]: budget factor 2. Introductory and current best (Theorem 4). The planner tries every agent as the one that fetches the message, moves it to v_s, and computes the smallest budget B' for which the reaches (the points within distance B' of the start nodes, B' minus the distance walked for the moved agent) form a chain from v_s to v_t. The smallest budget that admits a schedule is at least B', and letting each agent of the chain walk to the start node of the next costs at most $2B'$. The same idea gives budget factor 2 for the decision task.

Open questions.

Q1. Is there a constant $c > 1$ such that the smallest equal budget cannot be approximated within c in polynomial time unless P = NP? (Chalopin et al. 2021) notes that no lower bound is known.

Notes. Fixed path. Chalopin, Das, Mihalák, Penna and Widmayer also fix in advance the path along which the message must travel, to keep it out of dangerous areas: the task stays NP-hard (Theorem 5, from 3SAT; the authors note that the reduction extends to equal budgets), and for some constant $\gamma > 1$ no polynomial-time planner has budget factor γ unless P = NP (Corollary 2, whose proof is only sketched) (Chalopin et al. 2013). Chalopin, Das, Disser, Labourel and Mihalák study it with equal budgets (Chalopin et al. 2021); one agent may now carry the message on several pieces of the path. The smallest equal budget can be approximated within 3 on directed graphs (Theorem 3), not within less than 2 unless

Claude (Anthropic) was used in cataloguing the problems.

P = NP (Theorem 1), and within 2.5 on undirected graphs (Theorem 4). If every agent picks the message up at most once, the factors become 2 on directed graphs and $2 - 1/2^k$ on undirected ones (Theorem 6), with a lower bound of 1.5 on directed graphs (Theorem 5). For two agents the fixed-path task is weakly NP-hard (Theorem 7, from subset sum); with hand-overs at nodes only, a dynamic program decides it in $O(k\, n^{k+2} B^k)$ time for integer weights (Theorem 8) and yields a fully polynomial-time approximation scheme for a constant number of agents (Theorem 9). **Several sources.** With a constant number of sources, each holding data for v_t, the 2-approximation extends (Theorem 6 of Chalopin et al. 2013); with arbitrarily many sources, each holding an agent, delivery is NP-hard even with equal budgets, by the convergecast reduction of (Anaya et al. 2016) (Section 3.2 of Chalopin et al. 2013). Theorem numbers of (Chalopin et al. 2013) refer to the authors' copy.

16.19.5 Planar graphs, equal budgets

Adds + planar, + equal budgets **to** Arbitrary graphs (p. 335) VERIFIED

Delivery stays strongly NP-hard on planar graphs, even when all budgets are equal.

Assumed: graph dynamics: static graph; edge direction: undirected; graph class: planar graphs; budgets: equal budgets. **Not assumed:** —.

Measure	Lower bound	Best upper bound	Status
Time	*NP-hard* (Bärtschi et al. 2020) Strongly NP-hard.	—	● unknown
Budget factor	none known A factor 2 holds as on arbitrary graphs; no lower bound is known.	—	● unknown

Results.

- Bärtschi et al. (2020) [lower bound]: time *NP-hard. Bottleneck:* combinatorial hardness. *Holds when:* strongly NP-hard, by a reduction from planar 3SAT (Theorem 5); equal budgets: an edge of length $B - B_i$ is added at each agent's start node, and the agent moved to its end.

Open questions.

Q1. Is there an inapproximability bound for the smallest equal budget, already on planar graphs? No lower bound above 1 is known for any graph class (Chalopin et al. 2021).

Notes. The same reduction shows returning delivery strongly NP-hard on planar graphs (Bärtschi et al. 2020). Checked against the full text (arXiv v1).

Claude (Anthropic) was used in cataloguing the problems.

CHAPTER 17

Computing with agents

Agents solve graph problems on the anonymous graph they move in: leader election, independent sets, dominating sets, spanning trees, subgraph counts, and decisions about the graph itself. Most algorithms first disperse the agents, so that every node has an agent to compute for it.
This family is related to Dispersion: Most algorithms first bring the n agents into a dispersed configuration, one agent per node, so that every node has an agent to compute for it (Kshemkalyani et al. 2026; Chand and Molla 2026).

Problems in this family:

- Leader election by agents
- Maximal independent set by agents
- Minimal dominating set by agents
- Triangle and butterfly counting by agents
- Minimum spanning tree by agents
- BFS tree by agents
- Deciding graph properties by mobile agents



17.1 Leader election by agents

n agents in an anonymous graph must elect one leader; how fast, and with how little memory?

First posed in 2024 (Kshemkalyani et al. 2024). Leader election in the agent-based model, with MST as an application (DISC 2024).

Problem statement and measures

n agents with distinct identifiers (IDs) are placed on the nodes of an anonymous port-labelled graph with n nodes, m edges and maximum degree Δ. The start is rooted (all agents on one node), dispersed (one agent on each node) or arbitrary. Agents compute only by moving and by talking with agents on the same node. The agents must elect exactly one leader: every agent (or node) learns its part of the output. **Time** is in synchronous rounds; **memory** in bits per agent.

Measures: **Time**, rounds; **Memory**, bits per agent.

A first look

Picture n **agents** (small mobile computers) dropped into a building with n rooms, where the rooms carry no names and an agent can only talk to agents standing in the same room. The building is a **graph**:



(a) Start: dispersed; IDs written in 2 bits

(b) Final: after n−1 = 2 simulated rounds, 1 leads

Figure 17.1: Leader election by simulating message passing (Kshemkalyani et al. 2024): the agents, already dispersed one per node of a path, write their IDs in 2 bits. In block i (of $2\Delta = 4$ rounds) an agent whose i-th bit is 1 visits each neighbour and comes back, one edge per round, while agents with bit 0 wait at home (the ID-bit schedule); every visit to a waiting neighbour (blue edge) carries the messages of one simulated round, here the smallest ID seen ("min"). Both simulated rounds are shown round by round; after $n - 1 = 2$ of them every agent knows that agent 1 is the leader.

rooms are its **nodes** and corridors are its **edges**; there are n nodes and m edges, and from here on we say only node and edge. Two nodes joined by an edge are *neighbours*. The **degree** $\deg(v)$ of a node v is the number of edges at it, and Δ is the maximum degree. Nodes have no names, but the edges at a node v are numbered $1, 2, \ldots, \deg(v)$; these numbers are **ports**, so an agent can say "leave by port 3". Every agent carries a distinct **identifier** (an ID number). Time runs in **rounds**: in one round every agent may talk to the agents on its node and then move along at most one edge. The task is to pick exactly one agent as **leader**: at the end the leader knows it is the leader and every other agent knows it is not, and we count the rounds and the bits of **memory** each agent needs. If all agents start on one node the task is easy (the smallest ID wins), so the interesting case is when they start with one agent on each node (a **dispersed** start), or in a few clusters.

1. Algorithm: Disperse, then simulate message passing by visiting neighbours. (Kshemkalyani et al. 2024) (Knowing n and Δ, p. 345) This is the idea behind every problem in this family. In the classic **message-passing model**, a computer sits still on each node and in each round sends a message to every neighbour. Agents can pretend to be such computers, provided there is one agent per node and neighbouring agents can meet once per simulated round. Here the agents know n and Δ.

1. **Disperse.** Run a dispersion algorithm so that every node holds exactly one agent; that node is the agent's **home**. Knowing n tells the agents how long to wait before they can be sure dispersion is over.
2. **Pad the IDs.** Each agent writes its ID in binary with the same number of bits, $O(\log n)$, adding leading zeros (knowing n fixes this number).
3. **Meet every neighbour.** Split time into blocks of 2Δ rounds, one block per bit. In block i, an agent whose i-th bit is 1 visits its neighbours one by one (leave by port 1, come back, leave by port 2, come back, ...); an agent whose i-th bit is 0 stays at home. This meeting schedule is the **ID-bit schedule**.
4. Two neighbouring agents have different IDs, so in some block one of them has bit 1 and the other bit 0: the first arrives while the second is at home, and they meet. A visitor recognises the agent at home as the one whose current bit is 0.
5. **Simulate.** When two neighbouring agents meet they hand over the messages the message-passing algorithm would send along their edge. So one pass through all the blocks simulates one round of message passing.

Claude (Anthropic) was used in cataloguing the problems.

6. Run any message-passing leader election this way, for example one that needs $O(D)$ rounds, where D is the *diameter* (the largest distance between two nodes).

Why it works and what it costs. A block lasts 2Δ rounds because a visit to a neighbour and back takes 2 rounds and there are at most Δ neighbours. So one message-passing round costs $O(\log n)$ blocks of 2Δ rounds, that is $O(\Delta \log n)$ rounds, and a message-passing algorithm of R rounds costs $O(\Delta R \log n)$ rounds, plus the time to disperse ($O(n \log^2 n)$ rounds with the dispersion algorithm the paper uses Sudo et al. 2026a). With an $O(D)$-round election this gives $O(\Delta D \log n + n \log^2 n)$ rounds. The catch is memory: each agent must store whatever a message-passing node stores, which can be large; (Kshemkalyani et al. 2026) reach a bound of the same form, $O(n \log^2 n + D\Delta \log n)$, with only $O(\log n)$ bits per agent.

Why it matters. It turns decades of message-passing algorithms into agent algorithms at a cost of a factor $\Delta \log n$, and the ID-bit schedule is reused by almost every later paper on computing with agents.

2. Algorithm: Walk every edge and come home. (Kshemkalyani et al. 2024) (No knowledge, p. 344) Without knowing n or Δ, the agents cannot use blocks of fixed length, and an agent that starts alone on its node does not even know whether the others are also alone. The first algorithm for this setting works as follows (simplified here for a dispersed start).

1. Each agent visits its neighbours once and returns home. It stays a *candidate* only if its node has the smallest degree among its neighbours, with ties broken by ID; otherwise it gives up. At least one candidate survives.
2. Each surviving candidate tells a neighbouring agent where its home is, so that others do not mistake its empty home for an unvisited node.
3. Each candidate then performs a *depth-first walk* (it leaves by a port it has not tried yet when there is one, and otherwise goes back the way it came) that crosses every edge of the graph, carrying a tag (start round, ID).
4. When two walks meet, the one with the smaller tag stops and its owner gives up.
5. A candidate whose walk crosses every edge and returns home declares itself leader.

Why it works and what it costs. A depth-first walk crosses each edge at most twice (once out, once back), so it ends within $O(m)$ rounds. The delicate part, which the paper proves, is that once one candidate declares itself leader no other can; this needs the home markers of step 2 and care when walks pass each other. When the agents start in clusters the same paper first disperses them with depth-first walks, and then needs $O(n \log n)$ bits per agent. Later work brought the time down to $O(n)$ rounds with $O(\log n)$ bits (Chand and Molla 2026), building on optimal dispersion (Kshemkalyani et al. 2025b).

Why it matters. It showed that leader election, and with it MST, MIS and dominating sets, can be solved with no knowledge of the graph at all, which is the setting all later papers aim for.

3. Impossibility: Why the agents need distinct identifiers. (No knowledge, p. 344) Every setting in this catalogue gives the agents distinct IDs. Here is why. Take a ring of 6 nodes, one agent on each, and number the ports so that at every node port 1 leads clockwise and port 2 counter-clockwise. Suppose the 6 agents are identical (no IDs) and follow the same deterministic program. In round 1 they all see exactly the same thing, so they all do the same thing: all stay, or all move clockwise, or all move counter-clockwise. After the move each agent again sees exactly what every other agent sees. By the same reasoning round after round, all 6 agents are always in the same state. If one of them ever decides "I am the leader", all 6 decide it at the same moment.

Why it holds. So no deterministic algorithm elects a leader among identical agents on this ring, no matter how many rounds it uses or how much memory the agents have. The argument is a standard symmetry

Claude (Anthropic) was used in cataloguing the problems.

argument, not a result of a particular paper in this catalogue. Distinct IDs break the symmetry; the algorithms above use them both to decide who wins and, through the bits of the IDs (the ID-bit schedule), to schedule meetings.

Why it matters. Symmetry is the basic obstacle to leader election, and it explains why identifiers (or randomness) are part of every model in which leaders are elected.

What the community started looking at

The agent-based model asks what graph problems **agents** can solve on an anonymous network, with **rounds first and memory second** as the measures (Kshemkalyani et al. 2024). Leader election came first because it unlocks the rest (MST, BFS trees, gathering).

Further reading

Papers on this problem that are not yet placed in a setting.

- Kshemkalyani et al. (2025a). Full version of (Kshemkalyani et al. 2024), extended to $k < n$ agents, where each maximal connected group of occupied nodes elects its own leader: without knowledge the election stabilises (an elected leader may still be replaced) after $O((|C_{\max}| + \log^2 k)\Delta)$ rounds, $|C_{\max}|$ the number of nodes of the largest group (their notation; Theorem 12); knowing k and Δ no leader is replaced and $O(k\Delta)$ rounds suffice (Theorem 13); memory grows with the number of nodes that start with two or more agents and the number of local leaders. For $k = n$ it restates the $O(m)$-round election without knowledge (Theorem 14), with memory at most $O(n \log n)$ bits, as in (Kshemkalyani et al. 2024). Preprint. Read.

Assumptions specific to this problem

Knowledge

Whether agents know global parameters is inherited from message passing, where nodes typically know n or an upper bound on it, and possibly Δ and D; Kshemkalyani, Kumar, Molla and Sharma set the agent-based model against this and asked for algorithms that rely on no such knowledge, "neither exact nor an upper bound" (Kshemkalyani et al. 2024). Knowing n and Δ tells an agent how long to wait: in the arbitrary-start minimal dominating set algorithm of Chand, Molla and Sivasubramaniam every robot waits $\kappa n\Delta$ rounds before leader election begins (Chand et al. 2023), and the introductory maximal independent set algorithms assume both n and Δ when the agents start dispersed or arbitrarily placed, but need no knowledge from a rooted start (Pattanayak et al. 2024). Kshemkalyani et al. state the trade-off: solutions without knowledge "are more resilient to network changes and device faults", but "algorithm design becomes challenging since devices may not know how long to run a procedure to guarantee a solution" (Kshemkalyani et al. 2024). Later papers remove the assumption by first dispersing with termination detection and electing a leader, after which a spanning tree lets the agents compute Δ and the largest identifier themselves (Chand et al. 2026); knowing only that largest identifier L is a weaker assumption, recorded on results rather than as a level of this axis (Chand et al. 2025c). For leader election both levels are catalogued: without knowledge (Kshemkalyani et al. 2024; Chand and Molla 2026), and with n and Δ known (Kshemkalyani et al. 2026), whose $O(n \log^2 n + D\Delta \log n)$ rounds do not improve on the $O(n)$ rounds catalogued without knowledge (Chand and Molla 2026); from a dispersed start it needs $O(D\Delta \log n)$ rounds (Remark 1 of Kshemkalyani et al. 2026).

Level	Value	Formal definition
0	**no knowledge**	Agents know no graph parameter (n, m, Δ, D).
1	**knows** n, Δ	Every agent knows n and Δ (upper bounds suffice where stated).

Claude (Anthropic) was used in cataloguing the problems.

Case tree

The root assumes nothing; going down, each edge adds the assumptions written in grey (+) or leaves an axis unassumed. Bold entries are catalogued settings with their status per measure (Time, Memory) and page; italic entries are branch points, named after the axis on which their branches differ. Axis order: Graph dynamics → Edge direction → Identifiers → Communication → Timing → Knowledge → Initial configuration → Topology.

No assumptions

+ static graph, + undirected, + distinct IDs (eponymous), + local communication, + synchronous

No knowledge STANDARD SETTING ●● p. 344

+ knows n, Δ

Knowing n and Δ ●● p. 345

What is known

The best upper bound against the best lower bound in each catalogued setting, in tree order. The settings follow in detail.

Setting	Time	Memory	p.
No knowledge STANDARD SETTING	● unknown $O(n)$ vs —	● unknown $O(\log n)$ vs —	344
Knowing n and Δ	● unknown $O(n \log^2 n + D\Delta \log n)$ vs —	● unknown $O(\log n)$ vs —	345

17.1.1 No knowledge

Adds + static graph, + undirected, + distinct IDs (eponymous), + local communication, + synchronous
STANDARD SETTING VERIFIED

Without any knowledge of the graph, leader election takes $O(n)$ rounds with $O(\log n)$ bits.

Assumed: graph dynamics: static graph; edge direction: undirected; identifiers: distinct identifiers (eponymous): agents have pairwise distinct identifiers; communication: local (co-located agents communicate); timing: synchronous rounds. **Not assumed:** knowledge: no knowledge of graph parameters; initial configuration: arbitrary (agents on any $\ell \leq k$ nodes); topology: arbitrary connected graph.

Measure	Lower bound	Best upper bound	Status
Time	none known No time lower bound is stated in the papers.	$O(n)$ (Chand and Molla 2026)	● unknown
Memory	none known $\Omega(\log n)$ for distinct identifiers is folklore, not recorded here.	$O(\log n)$ (Chand and Molla 2026)	● unknown

Results.

- Kshemkalyani et al. (2024) [introductory]: time $O(m)$, memory $O(n \log n)$.
- Kshemkalyani et al. (2025c) [also]: time $O(n \log^2 n)$, memory $O(\log n)$. Theorem 1.1: from any start, without knowledge of any graph parameter; the dispersion algorithm of (Sudo et al. 2026a) is made terminating without knowing n.
- Chand and Molla (2026) [current best]: time $O(n)$, memory $O(\log n)$. A by-product of the minimal dominating set algorithm, built on optimal dispersion (Kshemkalyani et al. 2025b).

Notes. Spanning trees: BFS and MST with $O(\log n)$ bits from a dispersed start (Chand et al. 2025c).

Claude (Anthropic) was used in cataloguing the problems.

17.1.2 Knowing n and Δ

Adds + knows n, Δ **to** No knowledge (p. 344) VERIFIED

With n and Δ known, leader election takes $O(n \log^2 n + D\Delta \log n)$ rounds with $O(\log n)$ bits.

Assumed: graph dynamics: static graph; edge direction: undirected; identifiers: distinct identifiers (eponymous): agents have pairwise distinct identifiers; communication: local (co-located agents communicate); timing: synchronous rounds; knowledge: n and Δ are known. **Not assumed:** initial configuration: arbitrary (agents on any $\ell \leq k$ nodes); topology: arbitrary connected graph.

Measure	Lower bound	Best upper bound	Status
Time	none known	$O(n \log^2 n + D\Delta \log n)$ (Kshemkalyani et al. 2026)	• unknown
	No lower bound. From an arbitrary start, the $O(n)$ rounds without knowledge (Chand and Molla 2026) carry over to this setting and dominate the bound recorded here, as does $O(n \log^2 n)$ (Kshemkalyani et al. 2025c).		
Memory	none known —	$O(\log n)$ (Kshemkalyani et al. 2026)	• unknown

Results.

- Kshemkalyani et al. (2026) [current best]: time $O(n \log^2 n + D\Delta \log n)$, memory $O(\log n)$. Theorem 1 (Lemma 2): disperse in $O(n \log^2 n)$ rounds (Sudo et al. 2026a), knowing n to detect the end, then flood the largest ID over the ID-bit schedule. The abstract omits the $\log n$ factor of the second term. From a rooted start $O(n \log n + D\Delta \log n)$ rounds, from a dispersed start $O(D\Delta \log n)$ (Remark 1).

Notes. An algorithm that needs no knowledge also works when n and Δ are known, so the $O(n)$-round, $O(\log n)$-bit election of (Chand and Molla 2026) holds here and dominates the $O(n \log^2 n + D\Delta \log n)$ bound of (Kshemkalyani et al. 2026) from an arbitrary start. The latter is listed because it is the catalogued algorithm designed for this setting; from a dispersed start it takes $O(D\Delta \log n)$ rounds (Remark 1 of Kshemkalyani et al. 2026), which is smaller than n when $D\Delta \log n = o(n)$.

17.2 Maximal independent set by agents

n agents in an anonymous graph must mark a maximal independent set of nodes.

First posed in 2024 (Pattanayak et al. 2024). Introduced at ICDCN 2024.

Problem statement and measures

n agents with distinct identifiers (IDs) are placed on the nodes of an anonymous port-labelled graph with n nodes, m edges and maximum degree Δ. The start is rooted (all agents on one node), dispersed (one agent on each node) or arbitrary. Agents compute only by moving and by talking with agents on the same node. The agents must settle on the nodes of a maximal independent set, every other agent knowing it is not in the set: every agent (or node) learns its part of the output. **Time** is in synchronous rounds; **memory** in bits per agent.

Measures: **Time**, rounds; **Memory**, bits per agent.

Claude (Anthropic) was used in cataloguing the problems.



(a) Start: all 5 agents at node a

(b) Final: black nodes a, d, e form an MIS

Figure 17.2: MIS by a depth-first walk that colours as it settles (Chand et al. 2023) (other algorithms, e.g. from an arbitrary start, in Pattanayak et al. 2024): all agents start at node *a*; the group moves one edge per round in depth-first order (the walk's tree is drawn in grey), and at each empty node the agent with the smallest ID settles. A settled agent whose parent is black turns grey at once; otherwise it visits each neighbour and comes back, and turns black only if none is black. The black nodes form a maximal independent set.

A first look

Imagine a building with n rooms that carry no names, and n **agents** (small mobile computers) who must choose some rooms to stand in so that no two chosen rooms are next to each other, yet every room is either chosen or next to a chosen room. The building is a **graph**: rooms are its **nodes** and corridors are its **edges**; there are n nodes and m edges, and from here on we say only node and edge. Two nodes joined by an edge are *neighbours*. The **degree** $\deg(v)$ of a node v is the number of edges at it, and Δ is the maximum degree. Nodes have no names, but the edges at a node are numbered $1, 2, \ldots$; these numbers are **ports**. Each agent has a distinct **identifier** (ID). A set of nodes in which no two nodes are neighbours is *independent*; it is a **maximal independent set** (MIS) if no further node can be added to it without losing this. The agents move in synchronous **rounds** (in each round every agent may talk to the agents on its node, then move along at most one edge), and at the end every agent must know whether it stands on a chosen node. We count rounds and bits of memory per agent. As for all problems in this family, a general method is to make the agents **dispersed** (one agent on each node) and then let them imitate a classical message-passing algorithm by visiting their neighbours (explained under leader election by agents).

1. Algorithm: Depth-first walk that colours as it settles. (Chand et al. 2023) (No knowledge, p. 349) All n agents start together on one node (a *rooted* start) and know nothing about the graph.

1. The group moves through the graph in a **depth-first walk**: it leaves its current node by a port not tried yet when there is one, and otherwise goes back to the node from which it reached the current node. Whenever it reaches an empty node, the agent with the smallest ID among those still walking stays there for good: it becomes a **settled agent**, and it remembers the port by which the group came in, so the group can later walk back.
2. Before the group moves on, the new settled agent checks its neighbours. If its **parent** (the node from which the walk reached it) holds a *black* agent, it colours itself *grey*. Otherwise it visits every neighbour and comes back: if some neighbour is black it becomes grey, else black.
3. The rest of the group continues the depth-first walk; finding a settled agent on a node tells it that the node was visited before, so it turns back.
4. When every agent has settled, the black agents form the answer.

Why it works and what it costs. A black agent never has a black neighbour, so the black nodes are

Claude (Anthropic) was used in cataloguing the problems.

independent; every grey agent has a black neighbour, so no node can be added: the set is maximal. The depth-first walk crosses each edge a constant number of times, and the neighbour checks cost $2\deg(v)$ rounds at each node v, which sums to $4m$; so the total is $O(m)$ rounds with $O(\log n)$ bits per agent (Chand et al. 2023). The current best without knowledge, from any start, is $O(\max\{n \log^2 n, m\})$ rounds with $O(\log n)$ bits (Kaur et al. 2026c).

Why it matters. It shows that settling agents one at a time along a depth-first walk gives a greedy algorithm for free, the same pattern as dispersion.

2. Lower bound: Some agent must walk almost the whole diameter. (No knowledge, p. 349) Let all agents start on the end s of a path with $D + 1$ nodes, so the *diameter* D (largest distance between two nodes) is the length of the path. Maximality says that the far end u or its only neighbour must be chosen, and a chosen node needs an agent standing on it. Both nodes are at distance at least $D - 1$ from s.

Why it holds. An agent moves along at most one edge per round, so no agent reaches distance $D - 1$ before round $D - 1$: from a rooted start every algorithm needs at least $D - 1$ rounds. This simple argument is our own, not from the papers; no better lower bound is recorded in the catalogue, so the gap to the known $O(\max\{n \log^2 n, m\})$ rounds is wide open.

Why it matters. Distance is the most basic obstacle for moving agents, and the gap between this bound and the known algorithms shows how little is known about the optimal time.

3. Algorithm: Greedy by identifiers, simulated after dispersion. (Kshemkalyani et al. 2024) (Knowing n and Δ, p. 350) Here the agents know n and Δ, which lets them use the disperse-and-simulate method.

1. Disperse, so that every node holds one agent.
2. Repeat simulated message-passing rounds, each made of $O(\log n)$ blocks of 2Δ rounds in which neighbouring agents meet by the **ID-bit schedule**: in block i an agent whose i-th ID bit is 1 visits its neighbours one by one and an agent whose bit is 0 waits on its node (see leader election by agents).
3. An agent joins the set as soon as every neighbour with a larger ID has decided and none of them joined; it stays out as soon as some neighbour with a larger ID joined.
4. Each agent tells its neighbours its decision at the next meetings.

Why it works and what it costs. The agent with the largest ID decides at once, and an agent waits only for neighbours with larger IDs, so a chain of waiting agents has decreasing IDs and at most n members: at most n simulated rounds. Each costs $O(\Delta \log n)$ real rounds, so the total is $O(n\Delta \log n)$ plus the time to disperse, the introductory $O(n\Delta \log n)$ bound of (Pattanayak et al. 2024), whose dispersed-start algorithm is this greedy rule with the ID-bit schedule (Theorem 3.4). With n and Δ known the current best is $O(n \log^2 n + D\Delta \log n)$ rounds with $O(\log n)$ bits (Kshemkalyani et al. 2026).

Why it matters. It shows the cost of the generic method: a slow message-passing algorithm becomes a slow agent algorithm, which is why later work designs algorithms for agents directly.

What the community started looking at

MIS by agents was posed with **time first, memory second**: rooted, dispersed and arbitrary starts, trees and grids (Pattanayak et al. 2024); only the rooted algorithm needed no knowledge, and later work removed the knowledge of n and Δ for every start (Kshemkalyani et al. 2024; Kaur et al. 2026c).

Claude (Anthropic) was used in cataloguing the problems.

Further reading

Papers on this problem that are not yet placed in a setting.

- Jana et al. (2025b). Filling a minimal vertex cover, the complement of a maximal independent set, by Look–Compute–Move robots with lights and a visibility range of a few hops under an ASYNC scheduler: the robots enter an anonymous port-labelled graph one at a time through one or more entry nodes and must settle, never two on one node, on a minimal vertex cover. Impossible with visibility range 1 (Theorem 1); $\Omega(m)$ epochs from one entry node (Theorem 2); $O(m)$ epochs from one entry node with range 3, 4 colours and $O(\log \Delta)$ bits (Theorem 5), and from several entry nodes with range 4 and a number of colours linear in the number of entry nodes (Theorem 6); on trees from one entry node, a minimum vertex cover in $O(m)$ epochs with range 2, 4 colours and no other memory (Theorem 3). Read.
- Himani and Pandit (2026b). A maximal independent set on unoriented $x \times y$ grids and tori by $k = n + 1$ synchronous agents that start anywhere: the agents first fill every node, then the agents mark an independent set; $O(\max\{x, y\})$ rounds with $O(\log n)$ bits per agent, with or without knowing x and y, which the authors call optimal because filling every node needs as much (Section 1.2, Theorem 5.3). Read (model, algorithms and theorems).

Assumptions specific to this problem

Knowledge

Whether agents know global parameters is inherited from message passing, where nodes typically know n or an upper bound on it, and possibly Δ and D; Kshemkalyani, Kumar, Molla and Sharma set the agent-based model against this and asked for algorithms that rely on no such knowledge, "neither exact nor an upper bound" (Kshemkalyani et al. 2024). Knowing n and Δ tells an agent how long to wait: in the arbitrary-start minimal dominating set algorithm of Chand, Molla and Sivasubramaniam every robot waits $\kappa n\Delta$ rounds before leader election begins (Chand et al. 2023), and the introductory maximal independent set algorithms assume both n and Δ when the agents start dispersed or arbitrarily placed, but need no knowledge from a rooted start (Pattanayak et al. 2024). Kshemkalyani et al. state the trade-off: solutions without knowledge "are more resilient to network changes and device faults", but "algorithm design becomes challenging since devices may not know how long to run a procedure to guarantee a solution" (Kshemkalyani et al. 2024). Later papers remove the assumption by first dispersing with termination detection and electing a leader, after which a spanning tree lets the agents compute Δ and the largest identifier themselves (Chand et al. 2026); knowing only that largest identifier L is a weaker assumption, recorded on results rather than as a level of this axis (Chand et al. 2025c). For maximal independent sets the introductory algorithms assume n and Δ from dispersed and arbitrary starts and nothing from a rooted start (Theorems 3.1, 3.4 and 3.5 of Pattanayak et al. 2024); (Kshemkalyani et al. 2024) and (Kaur et al. 2026c) remove the assumption for every start.

Level	Value	Formal definition
0	**no knowledge**	Agents know no graph parameter (n, m, Δ, D).
1	**knows** n, Δ	Every agent knows n and Δ (upper bounds suffice where stated).

Claude (Anthropic) was used in cataloguing the problems.

Case tree

The root assumes nothing; going down, each edge adds the assumptions written in grey (+) or leaves an axis unassumed. Bold entries are catalogued settings with their status per measure (Time, Memory) and page; italic entries are branch points, named after the axis on which their branches differ. Axis order: Graph dynamics → Edge direction → Identifiers → Communication → Timing → Knowledge → Initial configuration → Topology.

No assumptions

+ static graph, + undirected, + distinct IDs (eponymous), + local communication, + synchronous

No knowledge ●● p. 349

+ knows n, Δ

Knowing n and Δ STANDARD SETTING ●● p. 350

no knowledge, + rooted start

Rooted start, no knowledge ●● p. 350

What is known

The best upper bound against the best lower bound in each catalogued setting, in tree order. The settings follow in detail.

Setting	Time	Memory	p.
No knowledge	● unknown $O(\max\{n \log^2 n, m\})$ vs —	● unknown $O(\log n)$ vs —	349
Knowing n and Δ STANDARD SETTING	● unknown $O(n \log^2 n + D\Delta \log n)$ vs —	● unknown $O(\log n)$ vs —	350
Rooted start, no knowledge	● unknown $O(n\Delta)$ vs —	● unknown $O(\log n)$ vs —	350

17.2.1 No knowledge

Adds + static graph, + undirected, + distinct IDs (eponymous), + local communication, + synchronous VERIFIED

Without knowledge: $O(\max\{n \log^2 n, m\})$ rounds with $O(\log n)$ bits.

Assumed: graph dynamics: static graph; edge direction: undirected; identifiers: distinct identifiers (eponymous): agents have pairwise distinct identifiers; communication: local (co-located agents communicate); timing: synchronous rounds. **Not assumed:** knowledge: no knowledge of graph parameters; initial configuration: arbitrary (agents on any $\ell \leq k$ nodes); topology: arbitrary connected graph.

Measure	Lower bound	Best upper bound	Status
Time	none known —	$O(\max\{n \log^2 n, m\})$ (Kaur et al. 2026c)	● unknown
Memory	none known —	$O(\log n)$ (Kaur et al. 2026c)	● unknown

Results.

- Kshemkalyani et al. (2024) [introductory]: time $O(n\Delta)$, memory $O(n \log n)$.
- Kaur et al. (2026c) [current best]: time $O(\max\{n \log^2 n, m\})$, memory $O(\log n)$.

Claude (Anthropic) was used in cataloguing the problems.

17.2.2 Knowing n and Δ

Adds + knows n, Δ **to** No knowledge (p. 349) STANDARD SETTING VERIFIED

With n and Δ known: $O(n \log^2 n + D\Delta \log n)$ rounds from any start with $O(\log n)$ bits; from a rooted start $O(n\Delta)$ rounds need no knowledge.

Assumed: graph dynamics: static graph; edge direction: undirected; identifiers: distinct identifiers (eponymous): agents have pairwise distinct identifiers; communication: local (co-located agents communicate); timing: synchronous rounds; knowledge: n and Δ are known. **Not assumed:** initial configuration: arbitrary (agents on any $\ell \leq k$ nodes); topology: arbitrary connected graph.

Measure	Lower bound	Best upper bound	Status
Time	none known No lower bound recorded.	$O(n \log^2 n + D\Delta \log n)$ (Kshemkalyani et al. 2026)	• unknown
Memory	none known —	$O(\log n)$ (Kshemkalyani et al. 2026)	• unknown

Results.

- Pattanayak et al. (2024) [introductory]: time $O(n\Delta \log n)$, memory $O(\log n)$. Theorems 3.4 and 3.5: $O(n\Delta \log n)$ rounds from dispersed and arbitrary starts. From a rooted start $O(n\Delta)$ rounds with no knowledge (Theorem 3.1, listed under the rooted setting). Trees: $O(D^2)$ rounds from a rooted start with $O(\log n)$ bits (Theorem 4.2); $O(D)$ rounds from a dispersed start knowing Δ, and $O(n)$ rounds from an arbitrary start knowing n and Δ, both with $O(\Delta + \log n)$ bits (Theorems 4.3 and 4.4). Rectangular grids of $x \times y$ nodes: $O(\max(x, y))$ rounds with $O(\log n)$ bits, knowing x and y (Theorem 5.1).
- Kshemkalyani et al. (2026) [current best]: time $O(n \log^2 n + D\Delta \log n)$, memory $O(\log n)$. Theorem 3: elect a leader (Theorem 1), then build the set from the dispersed configuration.

17.2.3 Rooted start, no knowledge

Adds no knowledge, + rooted start **to** No knowledge (p. 349) VERIFIED

From a rooted start without knowledge of n or Δ, an MIS takes $O(n\Delta)$ rounds with $O(\log n)$ bits.

Assumed: graph dynamics: static graph; edge direction: undirected; identifiers: distinct identifiers (eponymous): agents have pairwise distinct identifiers; communication: local (co-located agents communicate); timing: synchronous rounds; initial configuration: rooted (all agents on one node). **Not assumed:** knowledge: no knowledge of graph parameters; topology: arbitrary connected graph.

Measure	Lower bound	Best upper bound	Status
Time	none known No lower bound. From an arbitrary start, $O(\max\{n \log^2 n, m\})$ rounds with $O(\log n)$ bits (Kaur et al. 2026c) carry over and are faster when $m = o(n\Delta)$ and Δ exceeds $\log^2 n$.	$O(n\Delta)$ (Pattanayak et al. 2024)	• unknown
Memory	none known (Pattanayak et al. 2024) call $O(\log n)$ bits optimal because an agent stores its ID; no lower bound is recorded here.	$O(\log n)$ (Pattanayak et al. 2024)	• unknown

Results.

- Pattanayak et al. (2024) [current best]: time $O(n\Delta)$, memory $O(\log n)$. Introductory and current best (Theorem 3.1, Algorithm MIS-GraphRooted): a depth-first traversal that settles one agent on each new node, while the unsettled agents visit the neighbours of that node to learn whether one of them is already in the set; at most $4m$ rounds of traversal and 2δ rounds of visits at a node of degree δ. No knowledge of graph parameters. On trees, $O(D^2)$ rounds with $O(\log n)$ bits (Theorem 4.2).

Claude (Anthropic) was used in cataloguing the problems.

Notes. The rooted start is all n agents on one node (initial axis of the model). From the same start, (Kshemkalyani et al. 2024) need $O(n\Delta)$ rounds and $O(n \log n)$ bits (Theorem 20), which this result dominates. Checked against the full text of (Pattanayak et al. 2024).

17.3 Minimal dominating set by agents

n agents in an anonymous graph must mark a minimal dominating set.

First posed in 2023 (Chand et al. 2023). Introduced at ALGOSENSORS 2023.

Problem statement and measures

n agents with distinct identifiers (IDs) are placed on the nodes of an anonymous port-labelled graph with n nodes, m edges and maximum degree Δ. The start is rooted (all agents on one node), dispersed (one agent on each node) or arbitrary. Agents compute only by moving and by talking with agents on the same node. The agents must mark a minimal dominating set of nodes: every agent (or node) learns its part of the output. **Time** is in synchronous rounds; **memory** in bits per agent.

Measures: **Time**, rounds; **Memory**, bits per agent.



(a) Start: all 5 agents at node a

(b) Final: black a, c, d dominate, minimally

Figure 17.3: Minimal dominating set by a depth-first walk that colours as it settles (Chand et al. 2023): all agents start at node a; the group moves one edge per round in depth-first order (the walk's tree is drawn in grey), and at each empty node the agent with the smallest ID settles. A settled agent whose parent is black turns grey at once; otherwise it visits each neighbour and comes back, and turns black only if none is black. Every node is black or next to a black node, and no black node can be dropped.

A first look

Imagine placing first-aid kits in some rooms of a building so that every room either has a kit or is next to a room with one, and no kit can be removed without leaving some room uncovered. The building is a **graph**: rooms are its **nodes** and corridors are its **edges**; there are n nodes and m edges, and from here on we say only node and edge. Two nodes joined by an edge are *neighbours*. The **degree** $\deg(v)$ of a node v is the number of edges at it, and Δ is the maximum degree. Nodes have no names, but the edges at a node are numbered $1, 2, \ldots$; these numbers are **ports**. A set of nodes is a *dominating set* if every node is in it or is a neighbour of a node in it; it is a **minimal dominating set** if removing any one node breaks this property (it need not be the smallest possible). n **agents** (small mobile computers) with distinct **identifiers** (IDs) move in synchronous **rounds** (in each round an agent may talk to the agents on its node, then move along at most one edge) and must mark such a set: at the end every agent knows

Claude (Anthropic) was used in cataloguing the problems.

whether its node is in the set. We count rounds and bits of memory per agent. As for all problems in this family, one general method is to place one agent on each node and then imitate a message-passing algorithm by visiting neighbours (explained under leader election by agents).

1. Algorithm: Depth-first walk that colours as it settles. (Chand et al. 2023) (No knowledge, p. 353) All n agents start on one node (a *rooted* start) and know nothing about the graph.

1. The group moves through the graph in a **depth-first walk**: it leaves its current node by a port not tried yet when there is one, and otherwise goes back to the node from which it reached the current node. At each empty node it reaches, the walking agent with the smallest ID stays for good, becoming a **settled agent**, and remembers the port leading back.
2. The new settled agent decides its colour: if its **parent** (the node from which the walk reached it) holds a *black* agent it becomes *grey*; otherwise it visits all its neighbours and becomes grey if one of them is black, and black if none is.
3. The others continue; a settled agent on a node tells them the node was visited before.
4. When all agents have settled, the black nodes are the dominating set.

Why it works and what it costs. Every agent ends black or grey, and grey means "next to a black one", so the black nodes dominate the graph. They are also minimal: a black node has no black neighbour (all its neighbours are grey or settled after it and saw it), so removing it would leave it uncovered. In fact the black nodes form a maximal independent set, and every maximal independent set is a minimal dominating set. The walk crosses each edge a constant number of times and the checks cost $2\deg(v)$ rounds per node v, $4m$ in total, so the algorithm takes $O(m)$ rounds with $O(\log n)$ bits (Chand et al. 2023). The current best takes $O(n)$ rounds, from a rooted or arbitrary start, with $O(\log n)$ bits (Chand and Molla 2026), building on a faster dispersion algorithm (Kshemkalyani et al. 2025b).

Why it matters. It is the template that later dominating-set and MIS algorithms for agents refine: settle agents greedily during exploration and let each settled agent decide from its neighbourhood.

2. Lower bound: Some agent must walk almost the whole diameter. (No knowledge, p. 353) Let all agents start on the end s of a path with $D + 1$ nodes, so the *diameter* D (largest distance between two nodes) is the path's length. The far end u must be dominated: u or its only neighbour must be marked, and a marked node holds an agent. Both are at distance at least $D - 1$ from s.

Why it holds. Agents move along at most one edge per round, so the algorithm needs at least $D - 1$ rounds on this path. This is our own short argument; the catalogue records no lower bound, so whether the $O(n)$ rounds of (Chand and Molla 2026) are optimal is open. On a path $D = n - 1$, so on paths the $O(n)$ algorithm is optimal up to a constant factor.

Why it matters. It shows that linear time is the right target on long, thin graphs, and that a better lower bound would need graphs with small diameter.

What the community started looking at

Dominating sets by agents were posed with time and memory as measures (Chand et al. 2023); building on optimal dispersion, linear time without any knowledge followed (Chand and Molla 2026).

Assumptions specific to this problem

Knowledge

Whether agents know global parameters is inherited from message passing, where nodes typically know n or an upper bound on it, and possibly Δ and D; Kshemkalyani, Kumar, Molla and Sharma set the agent-based model against this and asked for algorithms that rely on no such knowledge, "neither exact nor an upper bound" (Kshemkalyani et al. 2024). Knowing n and Δ tells an agent how long to wait: in the arbitrary-start minimal dominating set algorithm of Chand, Molla and Sivasubramaniam every robot waits $\kappa n\Delta$ rounds before leader election begins (Chand et al. 2023), and the introductory maximal independent set algorithms assume both n and Δ when the agents start dispersed or arbitrarily placed, but need no knowledge from a rooted start (Pattanayak et al. 2024). Kshemkalyani et al. state the trade-off: solutions without knowledge "are more resilient to network changes and device faults", but "algorithm design becomes challenging since devices may not know how long to run a procedure to guarantee a solution" (Kshemkalyani et al. 2024). Later papers remove the assumption by first dispersing with termination detection and electing a leader, after which a spanning tree lets the agents compute Δ and the largest identifier themselves (Chand et al. 2026); knowing only that largest identifier L is a weaker assumption, recorded on results rather than as a level of this axis (Chand et al. 2025c). For minimal dominating sets the rooted algorithm of (Chand et al. 2023) already needs no knowledge, and (Chand and Molla 2026) removes it from the arbitrary start as well.

Level	Value	Formal definition
0	**no knowledge**	Agents know no graph parameter (n, m, Δ, D).
1	**knows n, Δ**	Every agent knows n and Δ (upper bounds suffice where stated).

Case tree

The root assumes nothing; going down, each edge adds the assumptions written in grey (+) or leaves an axis unassumed. Bold entries are catalogued settings with their status per measure (Time, Memory) and page; italic entries are branch points, named after the axis on which their branches differ. Axis order: Graph dynamics → Edge direction → Identifiers → Communication → Timing → Knowledge → Initial configuration → Topology.

No assumptions

+ static graph, + undirected, + distinct IDs (eponymous), + local communication, + synchronous

No knowledge STANDARD SETTING ●● p. 353

What is known

The best upper bound against the best lower bound in each catalogued setting, in tree order. The settings follow in detail.

Setting	Time	Memory	p.
No knowledge STANDARD SETTING	● unknown $O(n)$ vs —	● unknown $O(\log n)$ vs —	353

17.3.1 No knowledge

Adds + static graph, + undirected, + distinct IDs (eponymous), + local communication, + synchronous
STANDARD SETTING VERIFIED

A minimal dominating set in $O(n)$ rounds with $O(\log n)$ bits, from a rooted or arbitrary start.

Assumed: graph dynamics: static graph; edge direction: undirected; identifiers: distinct identifiers (eponymous): agents have pairwise distinct identifiers; communication: local (co-located agents communicate); timing: synchronous rounds. **Not**

Claude (Anthropic) was used in cataloguing the problems.

assumed: knowledge: no knowledge of graph parameters; initial configuration: arbitrary (agents on any $\ell \leq k$ nodes); topology: arbitrary connected graph.

Measure	Lower bound	Best upper bound	Status
Time	none known No lower bound stated.	$O(n)$ (Chand and Molla 2026)	• unknown
Memory	none known —	$O(\log n)$ (Chand and Molla 2026)	• unknown

Results.

- Chand et al. (2023) [introductory]: time $O(m)$, memory $O(\log n)$. *Assumes:* rooted start.
- Chand and Molla (2026) [current best]: time $O(n)$, memory $O(\log n)$.

17.4 Triangle and butterfly counting by agents

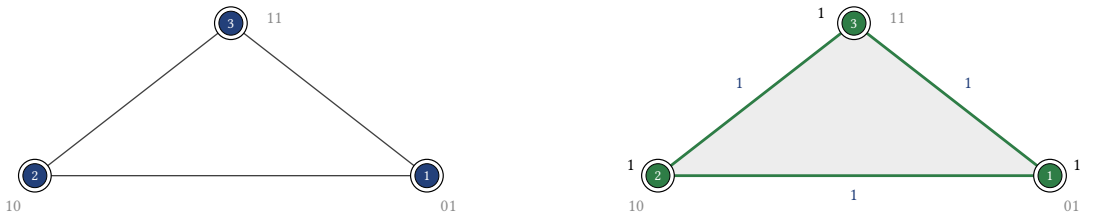
n agents in an anonymous graph must count small subgraphs (triangles; butterflies in bipartite graphs) at every node.

First posed in 2024 (Chand et al. 2026). Agent-based triangle counting (arXiv 2024; IEEE Transactions on Networking 2026).

Problem statement and measures

n agents with distinct identifiers (IDs) are placed on the nodes of an anonymous port-labelled graph with n nodes, m edges and maximum degree Δ. The start is rooted (all agents on one node), dispersed (one agent on each node) or arbitrary. Agents compute only by moving and by talking with agents on the same node. The agents must compute, at every node, the number of triangles (or butterflies) it belongs to: every agent (or node) learns its part of the output. **Time** is in synchronous rounds; **memory** in bits per agent.

Measures: **Time**, rounds; **Memory**, bits per agent.



(a) Start: dispersed; IDs written in 2 bits

(b) Final: (1 + 1) / 2 = 1 triangle at every node

Figure 17.4: Triangle counting (Chand et al. 2026) on a triangle: after dispersing (one agent per node), electing a leader and learning Δ (not shown), the agents write their IDs in 2 bits and follow the ID-bit schedule: in block i (of $2\Delta = 4$ rounds) agents whose i-th bit is 1 visit each neighbour and come back, one edge per round, while the others wait at home. In the first pass a visit to a waiting neighbour records both IDs (green edges); in the second pass, round by round with the same walks, the two ends of each edge count the IDs on both lists (blue numbers). Each agent adds its edges' counts and halves the sum: the number of triangles at its node.

Claude (Anthropic) was used in cataloguing the problems.

A first look

Imagine n **agents** (small mobile computers), one in each room of a building, who want to know how tightly the rooms are connected: for each room, how many pairs of the rooms next to it are also next to each other. The building is a **graph**: rooms are its **nodes** and corridors are its **edges**; there are n nodes and m edges, and from here on we say only node and edge. Two nodes joined by an edge are *neighbours*. The **degree** of a node is the number of edges at it, and Δ is the maximum degree. Nodes have no names, but the edges at a node are numbered $1, 2, \dots$; these numbers are **ports**. A **triangle** is three nodes all joined to each other. Agents have distinct **identifiers** (IDs) between 1 and L, can talk only to agents on the same node, and move in synchronous **rounds** (in each round: talk, then move along at most one edge). For the basic version the agents start anywhere and know nothing about the graph; at the end the agent on each node v must know the number of triangles containing v. We count rounds and bits of memory per agent. Like all problems in this family, the work starts by making the agents **dispersed** (one agent on each node, which becomes the agent's **home**) and letting neighbouring agents meet, the method explained under leader election by agents.

1. Algorithm: Learn your neighbours, then compare lists. (Chand et al. 2026) (Triangle counting, p. 357)

1. **Get organised.** The agents disperse (one per node), elect a leader and build a spanning tree, learning Δ on the way.
2. **Meet your neighbours.** Write IDs with $\log L$ bits and follow the **ID-bit schedule**: in block i (of 2Δ rounds), agents whose i-th bit is 1 visit their neighbours one by one, and agents whose bit is 0 stay at home. Every pair of neighbouring agents meets in some block, and each agent writes down the IDs of all its neighbours.
3. **Compare lists.** Repeat the meetings. When the agents on neighbouring nodes v and u meet, they count the IDs that appear on both lists: that is the number of triangles that contain the edge $\{v, u\}$.
4. **Add up.** The agent on v adds these counts over all its neighbours and divides by 2: each triangle $\{v, u, w\}$ was counted once through u and once through w.

Why it works and what it costs. Step 2 and step 3 each take $\log L$ blocks of 2Δ rounds, i.e. $O(\Delta \log L)$ rounds, and each agent stores up to Δ IDs, i.e. $O(\Delta \log L)$ bits. Step 1 costs $O(n \log^2 n)$ rounds, and it dominates the total $O(n \log^2 n + \Delta \log L)$ (Chand et al. 2026). The same counts per edge give truss decomposition and local clustering coefficients. No lower bound is proved.

Why it matters. It shows that once agents are organised, local graph statistics cost only $O(\Delta \log L)$ rounds, so the bottleneck is getting organised, not counting.

2. Lower bound: The total count needs information from both ends of the graph. (Triangle counting, p. 357) Besides the per-node counts, the algorithm also lets the agents learn the total number of triangles. Information moves at most one edge per round, because it travels only inside agents and an agent moves along at most one edge per round. So after t rounds, what an agent standing on node y knows depends only on the part of the graph within distance t of y. Take a long path and, at each end, attach two extra nodes a and b to the end node; adding the edge $\{a, b\}$ closes a triangle. If an agent announces the total after fewer than about $D/2$ rounds, where D is the *diameter* (largest distance between two nodes), one end lies outside what it can know about. Adding or removing the edge $\{a, b\}$ there changes the true total but not what this agent knows, so in one of the two graphs its answer is wrong.

Why it holds. So computing the total needs $\Omega(D)$ rounds. This is our own short argument, not a result recorded in the catalogue; the known algorithm needs $O(n + \Delta \log L)$ rounds for the total after the set-up,

Claude (Anthropic) was used in cataloguing the problems.

and on a path D is about n, so it is optimal there up to a constant factor. For the per-node counts this argument gives nothing, because they depend only on nearby nodes.

Why it matters. This "information moves one edge per round" argument is the simplest lower-bound tool for agents, and it separates local tasks (per-node counts) from global ones (totals).

3. Algorithm: Bipartite graphs: one side walks while the other waits. (Chand et al. 2025b) (Butterfly counting in bipartite graphs, p. 358) In a *bipartite* graph the nodes split into two sides V_1 and V_2 and every edge joins the two sides. A *butterfly* is a 4-cycle: two nodes a, a' on one side with two common neighbours on the other.

1. The agents, starting dispersed (one per node), elect a leader, build a spanning tree and learn which side they are on.
2. In odd rounds the agents on one side visit their neighbours while the other side waits at home; in even rounds the roles swap. Within 2Δ rounds every agent knows its neighbours' IDs.
3. The agent on a visits its neighbours again and reads their lists, so it learns, for every a' on its side, the number c of common neighbours of a and a'.
4. Each such pair lies on $\binom{c}{2}$ butterflies (choose two of the common neighbours); summing over a' gives the number of butterflies through a.

Why it works and what it costs. Because the two sides take turns, no ID-bit schedule is needed: neighbours meet within 2Δ rounds rather than $O(\Delta \log L)$, and the per-node counts take $O(\Delta)$ rounds. The set-up takes $O(n \log L)$ rounds, for a total of $O(n \log L + \Delta)$ rounds with $O(\Delta \log L)$ bits per agent (Chand et al. 2025b).

Why it matters. Knowing more structure (here, a 2-colouring of the graph) can remove the $\log L$ cost of making neighbours meet.

What the community started looking at

Counting problems extend agent-based computing from symmetry breaking to **graph analytics**: triangle counting and its applications (Chand et al. 2026), and butterfly counting in bipartite graphs (Chand et al. 2025b).

Assumptions specific to this problem

Knowledge

Whether agents know global parameters is inherited from message passing, where nodes typically know n or an upper bound on it, and possibly Δ and D; Kshemkalyani, Kumar, Molla and Sharma set the agent-based model against this and asked for algorithms that rely on no such knowledge, "neither exact nor an upper bound" (Kshemkalyani et al. 2024). Knowing n and Δ tells an agent how long to wait: in the arbitrary-start minimal dominating set algorithm of Chand, Molla and Sivasubramaniam every robot waits $\kappa n\Delta$ rounds before leader election begins (Chand et al. 2023), and the introductory maximal independent set algorithms assume both n and Δ when the agents start dispersed or arbitrarily placed, but need no knowledge from a rooted start (Pattanayak et al. 2024). Kshemkalyani et al. state the trade-off: solutions without knowledge "are more resilient to network changes and device faults", but "algorithm design becomes challenging since devices may not know how long to run a procedure to guarantee a solution" (Kshemkalyani et al. 2024). Later papers remove the assumption by first dispersing with termination detection and electing a leader, after which a spanning tree lets the agents compute

Claude (Anthropic) was used in cataloguing the problems.

Δ and the largest identifier themselves (Chand et al. 2026); knowing only that largest identifier L is a weaker assumption, recorded on results rather than as a level of this axis (Chand et al. 2025c). For triangle counting, Chand, Das and Molla remove the need to know Δ in exactly this way (Chand et al. 2026); their butterfly counting still assumes L (Chand et al. 2025b).

Level	Value	Formal definition
0	**no knowledge**	Agents know no graph parameter (n, m, Δ, D).
1	**knows n, Δ**	Every agent knows n and Δ (upper bounds suffice where stated).

Case tree

The root assumes nothing; going down, each edge adds the assumptions written in grey (+) or leaves an axis unassumed. Bold entries are catalogued settings with their status per measure (Time, Memory) and page; italic entries are branch points, named after the axis on which their branches differ. Axis order: Graph dynamics → Edge direction → Identifiers → Communication → Timing → Knowledge → Initial configuration → Topology.

No assumptions

+ static graph, + undirected, + distinct IDs (eponymous), + local communication, + synchronous
Triangle counting STANDARD SETTING ●● p. 357

+ dispersed start, + bipartite
Butterfly counting in bipartite graphs ●● p. 358

What is known

The best upper bound against the best lower bound in each catalogued setting, in tree order. The settings follow in detail.

Setting	Time	Memory	p.
Triangle counting STANDARD SETTING	● unknown $O(n \log^2 n + \Delta \log L)$ vs —	● unknown $O(\Delta \log L)$ vs —	357
Butterfly counting in bipartite graphs	● unknown $O(n \log L + \Delta)$ vs —	● unknown $O(\Delta \log L)$ vs —	358

17.4.1 Triangle counting

Adds + static graph, + undirected, + distinct IDs (eponymous), + local communication, + synchronous
STANDARD SETTING VERIFIED

From an arbitrary placement and without prior knowledge, agents count triangles per node and per edge, which yields truss decomposition, triangle centrality and local clustering coefficients.

Assumed: graph dynamics: static graph; edge direction: undirected; identifiers: distinct identifiers (eponymous): agents have pairwise distinct identifiers; communication: local (co-located agents communicate); timing: synchronous rounds. **Not assumed:** knowledge: no knowledge of graph parameters; initial configuration: arbitrary (agents on any $\ell \leq k$ nodes); topology: arbitrary connected graph.

Measure	Lower bound	Best upper bound	Status
Time	none known No lower bound is proved.	$O(n \log^2 n + \Delta \log L)$ (Chand et al. 2026)	● unknown
Memory	none known No lower bound is proved.	$O(\Delta \log L)$ (Chand et al. 2026)	● unknown

Claude (Anthropic) was used in cataloguing the problems.

Results.

- Chand et al. (2026) [current best]: time $O(n \log^2 n + \Delta \log L)$, memory $O(\Delta \log L)$. Introductory and current best (Theorem 2): dispersion, leader election and a spanning tree in $O(n \log^2 n)$ rounds, then per-node and per-edge counts in $O(\Delta \log L)$ rounds and the total in $O(n + \Delta \log L)$. Applications: truss decomposition in $O(m\Delta \log L + mD)$ rounds with $O(\Delta^2 \log L)$ bits; triangle centrality and local clustering coefficients in $O(\Delta \log L)$ rounds.

Notes. L is the largest agent identifier, at most n^κ, so $\log L = O(\log n)$. Exactly n agents. The problem is posed from a dispersed start, but Theorem 2 needs none. Checked against the full text (arXiv 2402.03653, version 3).

17.4.2 Butterfly counting in bipartite graphs

Adds + dispersed start, + bipartite **to** Triangle counting (p. 357) VERIFIED

In a bipartite graph, agents starting dispersed count butterflies ($K_{2,2}$) per node in $O(n \log L + \Delta)$ rounds.

Assumed: graph dynamics: static graph; edge direction: undirected; identifiers: distinct identifiers (eponymous): agents have pairwise distinct identifiers; communication: local (co-located agents communicate); timing: synchronous rounds; initial configuration: dispersed or scattered (one agent per occupied node); topology: bipartite. **Not assumed:** knowledge: no knowledge of graph parameters.

Measure	Lower bound	Best upper bound	Status
Time	none known —	$O(n \log L + \Delta)$ (Chand et al. 2025b)	• unknown
Memory	none known —	$O(\Delta \log L)$ (Chand et al. 2025b)	• unknown

Results.

- Chand et al. (2025b) [current best]: time $O(n \log L + \Delta)$, memory $O(\Delta \log L)$. *Assumes:* knows L, an upper bound on the identifiers. Introductory and current best. Leader election, a spanning tree and the two sides of the bipartition take $O(n \log L)$ rounds with $O(\log L)$ bits (Theorem 3.5), on any graph; per-node butterfly counts then take $O(\Delta)$ rounds and the total $O(\Delta + \min\{|V_1|, |V_2|\})$ (Theorem 4.4), with $O(\Delta \log L)$ bits (Lemma 4.3).

Notes. V_1 and V_2 are the two sides of the bipartite graph. Checked against the full text (arXiv 2506.17721).

17.5 Minimum spanning tree by agents

n agents in an anonymous weighted graph must build a minimum spanning tree.

First posed in 2024 (Kshemkalyani et al. 2024). MST in the agent-based model appeared together with leader election (DISC 2024).

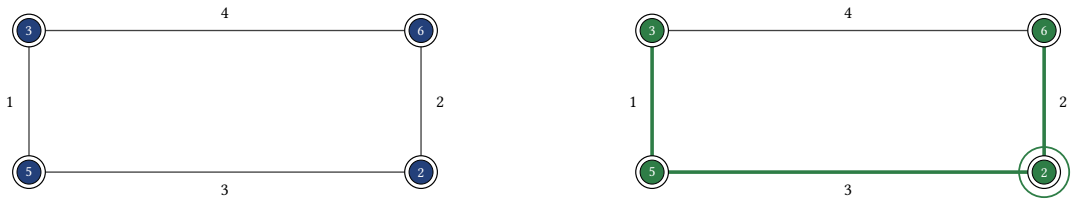
Problem statement and measures

n agents with distinct identifiers (IDs) are placed on the nodes of an anonymous port-labelled graph with n nodes, m edges and maximum degree Δ. The start is rooted (all agents on one node), dispersed (one agent on each node) or arbitrary. Agents compute only by moving and by talking with agents on the same node. The agents must build a minimum spanning tree of the edge-weighted graph, each agent

Claude (Anthropic) was used in cataloguing the problems.

knowing which incident edges belong to it: every agent (or node) learns its part of the output. **Time** is in synchronous rounds; **memory** in bits per agent.

Measures: **Time**, rounds; **Memory**, bits per agent.



(a) Start: dispersed; every node its own fragment

(b) Final: both pick weight 3; one fragment, the MST

Figure 17.5: MST by growing fragments along their lightest outgoing edges (Borůvka, Gallager-Humblet-Spira) (Chand et al. 2025c): agents dispersed on a weighted cycle meet their neighbours with the ID-bit schedule (in each block of $2\Delta = 4$ rounds, agents whose current ID bit is 1 visit each neighbour and come back, one edge per round). Each one-node fragment adds its lightest edge (weights 1 and 2), giving two fragments (colours) with leaders 3 and 2 (circled). In phase 2 the agents meet again with the same walks (shown round by round) and find the edges that leave their fragment (red); 5 and 6 walk to their leaders to report the lightest outgoing edge, both fragments pick weight 3 and merge into the minimum spanning tree (thick).

A first look

Imagine n **agents** (small mobile computers), one in each room of a building, who must choose which corridors to keep open so that every room can still reach every other, using the cheapest possible total length. The building is a **graph**: rooms are its **nodes** and corridors are its **edges**; there are n nodes and m edges, and from here on we say only node and edge. Two nodes joined by an edge are *neighbours*. Every edge has a **weight** (its cost), which the agents on its two ends can see; Δ is the maximum degree (the largest number of edges at one node). Nodes have no names, but the edges at a node are numbered $1, 2, \ldots$; these numbers are **ports**. A *spanning tree* is a set of $n - 1$ edges that connects all nodes without forming a cycle, and a **minimum spanning tree** (MST) is one with the smallest total weight. In the basic version the agents start one per node (a **dispersed** start), have distinct **identifiers** (IDs), move in synchronous **rounds** (talk to the agents on the same node, then move along at most one edge), and at the end every agent must know which of the edges at its node belong to the MST. We count rounds and bits of memory per agent. As for all problems in this family, the agents can imitate a message-passing algorithm by visiting their neighbours, the method explained under leader election by agents.

1. Algorithm: Grow fragments along their lightest outgoing edges. (Chand et al. 2025c) (Dispersed start, p. 362) This is the classic Borůvka / Gallager-Humblet-Spira idea, carried out by agents. Assume all edge weights differ (agents can break ties using their IDs and ports).

1. At the start every node is its own **fragment**: a set of nodes joined by edges already chosen for the tree, so a small tree.
2. In each phase, every agent meets its neighbours by the **ID-bit schedule** (in block i of 2Δ rounds, agents whose i-th ID bit is 1 visit their neighbours one by one while the others wait; see leader election by agents) and learns which of the edges at its node leave its fragment and how heavy they are.
3. The agents of each fragment pass the lightest such edge up their tree to the fragment's **leader** (one agent of the fragment that decides for it), which picks the **lightest outgoing edge** of the

Claude (Anthropic) was used in cataloguing the problems.

whole fragment: the edge of smallest weight with one end in the fragment and the other outside.

4. Every fragment adds its lightest outgoing edge; fragments joined by these edges merge into one, with one leader.
5. Repeat until one fragment remains: it is the MST, and its leader is also an elected leader for all agents.

Why it works and what it costs. Step 4 is safe because of the *cut property*: the lightest edge leaving any group of nodes always belongs to the MST. Every fragment merges with at least one other in each phase, so the number of fragments at least halves, and $O(\log n)$ phases suffice. In total this gives $O(n \log n + \Delta \log^2 n)$ rounds with $O(\log n)$ bits per agent, assuming the agents know L, the largest ID (Chand et al. 2025c). The first agent algorithm handled the fragments one at a time and needed $O(m + n \log n)$ rounds and $O(n \log n)$ bits (Kshemkalyani et al. 2024).

Why it matters. Halving the number of fragments in every phase is the standard way to build spanning trees in parallel, and the resulting tree gives agents a root and a structure for broadcasting, used afterwards for BFS and counting problems.

2. Lower bound: On a ring, the heaviest edge is far away. (Dispersed start, p. 362) Information moves at most one edge per round, because it travels only inside agents and each agent moves along at most one edge per round. So after t rounds, what the agent on node y knows depends only on the part of the graph within distance t of y.

Take a ring of n nodes with distinct weights. The MST of a ring is the ring minus its heaviest edge. Look at an edge $e = \{u, v\}$ and an edge f on the opposite side of the ring, at distance about $n/2$ from u and v. Give every edge other than e and f a weight below the weight of e. If f is lighter than e, then e is the heaviest edge and is not in the MST; if f is heavier, e is in the MST.

Why it holds. Within fewer than about $n/2 - 1$ rounds the agents on u and v cannot tell these two rings apart, so they answer the same way and one answer is wrong. Hence any algorithm needs $\Omega(n)$ rounds on a ring, and $\Omega(D)$ rounds in general, where D is the diameter (largest distance between two nodes). This is our own short argument; the catalogue records no lower bound. On a ring $\Delta = 2$, so the best algorithm, $O(n \log n + \Delta \log^2 n)$, is within a $\log n$ factor of it there.

Why it matters. It shows that an MST is a global object: an edge's membership can depend on an edge at the other end of the graph.

What the community started looking at

Spanning trees were the first application of agent-based leader election (Kshemkalyani et al. 2024); the question since has been **rounds, then memory**, without knowledge of graph parameters (Chand et al. 2025c).

Further reading

Papers on this problem that are not yet placed in a setting.

- Kshemkalyani et al. (2025a). Full version of (Kshemkalyani et al. 2024), extended to $k < n$ agents, which build a minimum spanning tree of each maximal connected group of occupied nodes: without knowledge, stabilising after $O(\Delta \log^2 k + |C_{\max}|(\Delta + \log |C_{\max}|))$ rounds, $|C_{\max}|$ the number of nodes of the largest group (their notation; Theorem 18); knowing k and Δ, explicit in $O(k\Delta)$ rounds (Theorem 19); memory $O(\Delta \log k)$ bits beyond that of the leader election. For $k = n$ it restates $O(m + n \log n)$ rounds without knowledge (Theorem 20), as in (Kshemkalyani et al. 2024). Preprint. Read.

Claude (Anthropic) was used in cataloguing the problems.

Assumptions specific to this problem

Knowledge

Whether agents know global parameters is inherited from message passing, where nodes typically know n or an upper bound on it, and possibly Δ and D; Kshemkalyani, Kumar, Molla and Sharma set the agent-based model against this and asked for algorithms that rely on no such knowledge, "neither exact nor an upper bound" (Kshemkalyani et al. 2024). Knowing n and Δ tells an agent how long to wait: in the arbitrary-start minimal dominating set algorithm of Chand, Molla and Sivasubramaniam every robot waits $\kappa n\Delta$ rounds before leader election begins (Chand et al. 2023), and the introductory maximal independent set algorithms assume both n and Δ when the agents start dispersed or arbitrarily placed, but need no knowledge from a rooted start (Pattanayak et al. 2024). Kshemkalyani et al. state the trade-off: solutions without knowledge "are more resilient to network changes and device faults", but "algorithm design becomes challenging since devices may not know how long to run a procedure to guarantee a solution" (Kshemkalyani et al. 2024). Later papers remove the assumption by first dispersing with termination detection and electing a leader, after which a spanning tree lets the agents compute Δ and the largest identifier themselves (Chand et al. 2026); knowing only that largest identifier L is a weaker assumption, recorded on results rather than as a level of this axis (Chand et al. 2025c). For minimum spanning trees, Kshemkalyani et al. need no parameter (Kshemkalyani et al. 2024), and Chand, Kumar and Molla reduce memory to $O(\log n)$ bits knowing only L (Chand et al. 2025c).

Level	Value	Formal definition
0	**no knowledge**	Agents know no graph parameter (n, m, Δ, D).
1	**knows n, Δ**	Every agent knows n and Δ (upper bounds suffice where stated).

Case tree

The root assumes nothing; going down, each edge adds the assumptions written in grey (+) or leaves an axis unassumed. Bold entries are catalogued settings with their status per measure (Time, Memory) and page; italic entries are branch points, named after the axis on which their branches differ. Axis order: Graph dynamics → Edge direction → Identifiers → Communication → Timing → Knowledge → Initial configuration → Topology.

No assumptions

+ static graph, + undirected, + distinct IDs (eponymous), + local communication, + synchronous

Arbitrary start ●● p. 362

+ dispersed start

Dispersed start STANDARD SETTING ●● p. 362

What is known

The best upper bound against the best lower bound in each catalogued setting, in tree order. The settings follow in detail.

Setting	Time	Memory	p.
Arbitrary start	● unknown $O(m + n \log^2 n)$ vs —	● unknown $O(\Delta \log n)$ vs —	362
Dispersed start STANDARD SETTING	● unknown $O(n \log n + \Delta \log^2 n)$ vs —	● unknown $O(\log n)$ vs —	362

Claude (Anthropic) was used in cataloguing the problems.

17.5.1 Arbitrary start

Adds + static graph, + undirected, + distinct IDs (eponymous), + local communication, + synchronous VERIFIED

From an arbitrary start without knowledge, an MST takes $O(m + n \log n)$ rounds with $O(n \log n)$ bits, or $O(m + n \log^2 n)$ rounds with $O(\Delta \log n)$ bits.

Assumed: graph dynamics: static graph; edge direction: undirected; identifiers: distinct identifiers (eponymous): agents have pairwise distinct identifiers; communication: local (co-located agents communicate); timing: synchronous rounds. **Not assumed:** knowledge: no knowledge of graph parameters; initial configuration: arbitrary (agents on any $\ell \leq k$ nodes); topology: arbitrary connected graph.

Measure	Lower bound	Best upper bound	Status
Time	none known	$O(m + n \log^2 n)$ (Kshemkalyani et al. 2025c)	• unknown
	No lower bound stated; the two algorithms trade a $\log n$ factor in time against memory.		
Memory	none known	$O(\Delta \log n)$ (Kshemkalyani et al. 2025c)	• unknown
	(Kshemkalyani et al. 2026) call $O(\Delta \log n)$ bits memory optimal for MST (Remark 2), but from a dispersed start $O(\log n)$ bits suffice when the largest identifier is known (Chand et al. 2025c); no lower bound is recorded.		

Results.

- Kshemkalyani et al. (2024) [introductory]: time $O(m + n \log n)$, memory $O(n \log n)$. Theorem 3: elect a leader in $O(m)$ rounds (Theorem 2), which also disperses the agents, then merge components along minimum outgoing edges; no knowledge of graph parameters.
- Kshemkalyani et al. (2025c) [current best]: time $O(m + n \log^2 n)$, memory $O(\Delta \log n)$. Theorem 4.4: elect a leader in $O(n \log^2 n)$ rounds with $O(\log n)$ bits (Theorem 1.1), then run the MST technique of (Kshemkalyani et al. 2024); no knowledge of graph parameters. Best in memory; it pays a $\log n$ factor on the $n \log n$ term of the time.

Open questions.

Q1. An MST from an arbitrary start in $O(m + n \log n)$ rounds, or fewer, with $O(\log n)$ bits per agent and no knowledge of graph parameters.

Notes. Knowing n and Δ, (Kshemkalyani et al. 2026) obtain $O(m + n \log^2 n + D\Delta \log n)$ rounds with $O(\Delta \log n)$ bits (Theorem 2), which the algorithm of (Kshemkalyani et al. 2025c) matches or beats without knowledge. Checked against the full text of (Kshemkalyani et al. 2024) (Theorem 3) and (Kshemkalyani et al. 2025c) (Theorem 4.4).

17.5.2 Dispersed start

Adds + dispersed start **to** Arbitrary start (p. 362) STANDARD SETTING VERIFIED

From a dispersed start, an MST takes $O(n \log n + \Delta \log^2 n)$ rounds with $O(\log n)$ bits.

Assumed: graph dynamics: static graph; edge direction: undirected; identifiers: distinct identifiers (eponymous): agents have pairwise distinct identifiers; communication: local (co-located agents communicate); timing: synchronous rounds; initial configuration: dispersed or scattered (one agent per occupied node). **Not assumed:** knowledge: no knowledge of graph parameters; topology: arbitrary connected graph.

Measure	Lower bound	Best upper bound	Status
Time	none known	$O(n \log n + \Delta \log^2 n)$ (Chand et al. 2025c)	• unknown
	No lower bound stated.		
Memory	none known	$O(\log n)$ (Chand et al. 2025c)	• unknown
	—		

Claude (Anthropic) was used in cataloguing the problems.

Results.

- Kshemkalyani et al. (2024) [introductory]: time $O(m + n \log n)$, memory $O(\log n \cdot \min\{\Delta, \log n\})$. Theorem 3 of (Kshemkalyani et al. 2024): from a dispersed start the memory is $O(\log n \cdot \min\{\Delta, \log n\})$ bits; $O(n \log n)$ bits from rooted or arbitrary starts.
- Chand et al. (2025c) [current best]: time $O(n \log n + \Delta \log^2 n)$, memory $O(\log n)$. *Assumes:* agents know L, the largest identifier.



17.6 BFS tree by agents

n agents in an anonymous graph must build a breadth-first spanning tree, without being told the root.

First posed in 2025 (Chand et al. 2025c). BFS without root knowledge in the agent-based model (SSS 2025).

Problem statement and measures

n agents with distinct identifiers (IDs) are placed on the nodes of an anonymous port-labelled graph with n nodes, m edges and maximum degree Δ. The start is rooted (all agents on one node), dispersed (one agent on each node) or arbitrary. Agents compute only by moving and by talking with agents on the same node. The agents must build a BFS tree from a root they choose, each agent knowing its parent and children: every agent (or node) learns its part of the output. **Time** is in synchronous rounds; **memory** in bits per agent.

Measures: **Time**, rounds; **Memory**, bits per agent.



(a) Start: dispersed; root 1 has depth 0

(b) Final: no new depth; BFS tree from root 1

Figure 17.6: BFS tree, grown layer by layer (Chand et al. 2025c): the agents are dispersed and the root (agent 1, chosen by leader election, not shown) has depth 0. In phase d (of $2\Delta = 6$ rounds) every agent of depth d visits each neighbour and comes back, one edge per round, while all others wait at home; a waiting agent without a depth that is visited takes depth $d + 1$ and the visitor's home as its parent (thick edges). The tree is complete when a phase gives no new depth.

A first look

Imagine n **agents** (small mobile computers), one in each room of a building, who want a plan for emergencies: one room is the exit, and every other agent should know which room next to its own is one step closer to the exit along a shortest route. The building is a **graph**: rooms are its **nodes** and corridors are its **edges**; there are n nodes and m edges, and from here on we say only node and edge. Two nodes joined by an edge are *neighbours*. The **degree** of a node is the number of edges at it, Δ is the maximum degree, and the *diameter* D is the largest distance between two nodes. Nodes have no names,

Claude (Anthropic) was used in cataloguing the problems.

but the edges at a node are numbered $1, 2, \ldots$; these numbers are **ports**. A **BFS tree** (breadth-first search tree) with **root** s (the exit of the picture) gives every other node v a **parent**: a neighbour of v that is one step closer to s, so following parents gives a shortest path to s. In the basic version the agents start one per node (a **dispersed** start; an agent's starting node is its **home**), have distinct **identifiers** (IDs), know nothing about the graph, and must choose the root themselves; they move in synchronous **rounds** (talk to the agents on the same node, then move along at most one edge), and at the end every agent knows its parent and its children. We count rounds and bits of memory per agent. As for all problems in this family, agents can imitate message-passing algorithms by visiting neighbours, the method explained under leader election by agents.

1. Algorithm: Grow the tree layer by layer from a known root. (Chand et al. 2025c) (Dispersed start, root unknown, p. 366) Suppose the agent on the root s is already known and the agents know Δ. Time is cut into phases of 2Δ rounds. Each agent will record a **depth**, a number that is unknown at the start.

1. In phase 0 the root agent takes depth 0.
2. In phase d, every agent of depth d visits its neighbours one by one (out along one edge, back along it); all other agents stay at home.
3. An agent that has no depth yet and receives such a visit takes depth $d + 1$ and records the visitor's home as its parent (if several visitors arrive, it picks the smallest ID). The visitor records its new child.
4. Repeat with phase $d + 1$ until no agent gets a new depth.

Why it works and what it costs. An agent gets depth d exactly when its distance to s is d, so the parents form a BFS tree. During phase d only depth-d agents move, so the agents they visit are at home. Each phase lasts 2Δ rounds and there are at most D phases with new nodes, so the tree is built in $O(D\Delta)$ rounds; (Chand et al. 2025c) cite this bound for a known root. Without a known root they first build an MST to get a root and the values m and Δ, which gives $O(\min(D\Delta, m \log n) + n \log n + \Delta \log^2 n)$ rounds with $O(\log n)$ bits.

Why it matters. It is the agent version of the textbook BFS, and it shows why knowing Δ matters: it fixes how long a phase must last.

2. Lower bound: A BFS tree contains an elected leader. (Dispersed start, root unknown, p. 366) When a BFS tree has been built, the agent on the root knows it has no parent, and every other agent knows it has one. So "I am the root" is exactly a leader election: one agent knows it is the leader and all others know they are not.

Why it holds. Hence building a BFS tree without a known root is at least as hard as leader election, and any lower bound for leader election also holds for BFS. This is why the best known algorithm spends $O(n \log n + \Delta \log^2 n)$ rounds electing a leader (via an MST) before the BFS itself (Chand et al. 2025c). The catalogue records no lower bound for either problem, so the time this preliminary step really needs is open.

Why it matters. It explains why leader election is the first building block in this family: trees, MIS and counting all need an agent that starts things off.

What the community started looking at

BFS trees are the next building block after leader election; the question is **rounds then memory** without knowledge of the root or of graph parameters (Chand et al. 2025c).

Assumptions specific to this problem

Knowledge

Whether agents know global parameters is inherited from message passing, where nodes typically know n or an upper bound on it, and possibly Δ and D; Kshemkalyani, Kumar, Molla and Sharma set the agent-based model against this and asked for algorithms that rely on no such knowledge, "neither exact nor an upper bound" (Kshemkalyani et al. 2024). Knowing n and Δ tells an agent how long to wait: in the arbitrary-start minimal dominating set algorithm of Chand, Molla and Sivasubramaniam every robot waits $\kappa n\Delta$ rounds before leader election begins (Chand et al. 2023), and the introductory maximal independent set algorithms assume both n and Δ when the agents start dispersed or arbitrarily placed, but need no knowledge from a rooted start (Pattanayak et al. 2024). Kshemkalyani et al. state the trade-off: solutions without knowledge "are more resilient to network changes and device faults", but "algorithm design becomes challenging since devices may not know how long to run a procedure to guarantee a solution" (Kshemkalyani et al. 2024). Later papers remove the assumption by first dispersing with termination detection and electing a leader, after which a spanning tree lets the agents compute Δ and the largest identifier themselves (Chand et al. 2026); knowing only that largest identifier L is a weaker assumption, recorded on results rather than as a level of this axis (Chand et al. 2025c). For BFS trees the catalogued algorithm uses no graph parameter, only L (Chand et al. 2025c).

Level	Value	Formal definition
0	**no knowledge**	Agents know no graph parameter (n, m, Δ, D).
1	**knows n, Δ**	Every agent knows n and Δ (upper bounds suffice where stated).

Case tree

The root assumes nothing; going down, each edge adds the assumptions written in grey (+) or leaves an axis unassumed. Bold entries are catalogued settings with their status per measure (Time, Memory) and page; italic entries are branch points, named after the axis on which their branches differ. Axis order: Graph dynamics → Edge direction → Identifiers → Communication → Timing → Knowledge → Initial configuration → Topology.

No assumptions

+ static graph, + undirected, + distinct IDs (eponymous), + local communication, + synchronous, + dispersed start

Dispersed start, root unknown STANDARD SETTING ●● p. 366

What is known

The best upper bound against the best lower bound in each catalogued setting, in tree order. The settings follow in detail.

Setting	Time	Memory	p.
Dispersed start, root unknown STANDARD SETTING	● unknown $O(\min(D\Delta, m\log n) + n\log n + \Delta\log^2 n)$ vs —	● unknown $O(\log n)$ vs —	366

Claude (Anthropic) was used in cataloguing the problems.

17.6.1 Dispersed start, root unknown

Adds + static graph, + undirected, + distinct IDs (eponymous), + local communication, + synchronous, + dispersed start
STANDARD SETTING VERIFIED

A BFS tree in $O(\min(D\Delta, m \log n) + n \log n + \Delta \log^2 n)$ rounds with $O(\log n)$ bits.

Assumed: graph dynamics: static graph; edge direction: undirected; identifiers: distinct identifiers (eponymous): agents have pairwise distinct identifiers; communication: local (co-located agents communicate); timing: synchronous rounds; initial configuration: dispersed or scattered (one agent per occupied node). **Not assumed:** knowledge: no knowledge of graph parameters; topology: arbitrary connected graph.

Measure	Lower bound	Best upper bound	Status
Time	none known —	$O(\min(D\Delta, m \log n) + n \log n + \Delta \log^2 n)$ (Chand et al. 2025c)	• unknown
Memory	none known —	$O(\log n)$ (Chand et al. 2025c)	• unknown

Results.

- Chand et al. (2025c) [current best]: time $O(\min(D\Delta, m \log n) + n \log n + \Delta \log^2 n)$, memory $O(\log n)$. Introductory and current best without knowledge of the root; $O(D\Delta)$ was known with a known root.



17.7 Deciding graph properties by mobile agents

Agents in an anonymous graph must decide together whether their starting configuration has a property, such as "the graph is a tree": which properties can they decide, with what help, and with how little memory?

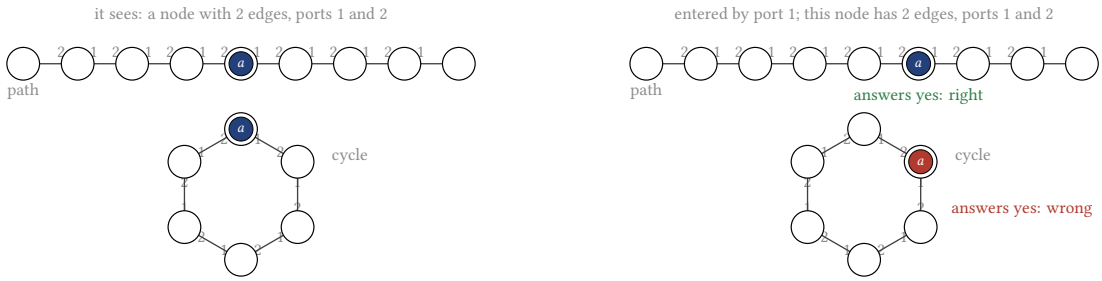
First posed in 2010 (Fraigniaud and Pelc 2017). Fraigniaud and Pelc introduced decision problems for mobile agents and the classes MAD and MAV, as (Bampas and Ilcinkas 2018) reports (arXiv 2010; LATIN 2012; Journal of Parallel and Distributed Computing 2017).

Problem statement and measures

$k \geq 1$ **agents** start at **nodes** of a connected undirected **graph** G with n nodes; the nodes are anonymous, and at each node the **edges** carry distinct **port** numbers. Each agent has a distinct identifier and an input string, and several agents may start at the same node. The agents move in synchronous **rounds**, crossing at most one edge per round, cannot mark nodes unless a setting gives **whiteboards**, and exchange everything they know when they are at the same node; no agent knows anything else in advance. A **configuration** is the graph with the agents' start nodes and inputs. A **decision problem** Π is a set of configurations, closed under automorphisms of G that preserve ports, whose membership a centralised algorithm that sees the whole configuration can decide. The agents **decide** Π if every agent eventually outputs yes or no, all agents output yes when the configuration is in Π, and at least one outputs no when it is not (Fraigniaud and Pelc 2017). In **verification** every agent also receives a **certificate**: for every configuration in Π some certificate makes all agents output yes, and for every other configuration no certificate does. The question is which decision problems the agents can decide, in particular whether all of them, and with how much memory. **Time** is in rounds; **memory** in bits per agent.

Measures: **Time**, rounds until every agent has decided; **Memory**, bits per agent.

Claude (Anthropic) was used in cataloguing the problems.



(a) Start: the same agent at the middle of a path of 2t + 3 = 9 nodes, or on a cycle

(b) Final, round t = 3: same view in both, so the same answer, wrong in one

Figure 17.7: Why agents cannot decide whether the graph is a path (Fraigniaud and Pelc 2017): a lone agent, which cannot know that it is alone, must answer after some number t of rounds (here $t = 3$; the ports it takes stand for any algorithm). Placed at the middle of a path of $2t + 3$ nodes with port 1 to the right and port 2 to the left, or on a cycle with port 1 clockwise and port 2 counter-clockwise, it sees exactly the same in every round, since the ends of the path are $t + 1$ edges away. So it gives the same answer on both graphs, and that answer is wrong on one of them. A certificate stating the number of nodes would let it check the answer.

GRAPHS

A first look

Imagine a few explorers lowered into a cave system they cannot see. Each must eventually answer yes or no to a question about the caves and the explorers in them, such as "do the caves form a single corridor?" The caves are a **graph**: chambers are **nodes** and tunnels are **edges**. The nodes have no names, but at each node the edges carry distinct **port** numbers, so an explorer can tell the exits apart. The explorers are **agents**, each with a distinct identifier and an input string; several may start on the same node. They move in synchronous **rounds**, crossing at most one edge per round. They cannot mark nodes, unless a setting gives every node a **whiteboard**, a small memory that agents there can read and write. They tell each other everything when they meet. The **configuration** is the graph with the agents' start nodes and inputs. A **decision problem** is a property of configurations that a computer seeing the whole configuration could check, such as "the graph is a tree". The agents **decide** it if every agent eventually answers, all agents answer yes when the configuration has the property, and at least one answers no when it does not. In **verification** every agent also receives a **certificate**, a string that can make all agents answer yes only when the true answer is yes. In the standard version the agents know nothing in advance, not even how many they are, and receive no certificate. The question is which decision problems they can decide, and with how much memory.

1. Impossibility: A lone agent cannot tell a long path from a cycle. (Fraigniaud and Pelc 2017) (Any number of agents, p. 371) Take the decision problem "is the graph a path?", a path being nodes in a row. A computer that sees the graph answers it at once.

1. Put a single agent on an endless path whose edges carry port 1 to the right and port 2 to the left at every node. Every algorithm must make the agent answer after some number of rounds t.
2. Now place the agent at the middle node of a path of $2t + 3$ nodes numbered the same way, or on a cycle with port 1 clockwise and port 2 counter-clockwise.
3. The ends of the path are $t + 1$ edges away, so in t rounds the agent never reaches them. On both graphs it sees, round after round, a node with two edges numbered 1 and 2, exactly as on the endless path.
4. It cannot wait for help either, since it cannot know whether other agents exist.

Claude (Anthropic) was used in cataloguing the problems.

Why it holds. So the agent gives the same answer, after the same t rounds, on both graphs, and that answer is wrong on one of them (Fraigniaud and Pelc 2017). The same holds for "is the graph a tree?" and "does the graph have exactly x nodes?". Not every problem that a computer can decide can be decided by agents; Fraigniaud and Pelc call those that can be decided MAD. A certificate helps here: given the number of nodes as certificate, the agents can check that the graph is a path, so this problem can be verified though not decided.

Why it matters. It is the basic limit of agents in anonymous graphs: what they cannot reach they cannot see, and a lone agent cannot know when it has seen enough.

2. Algorithm: Knowing the number of agents: gather, draw a map, decide. (Fraigniaud and Pelc 2017) (Two or more agents that know their number, p. 375) Suppose every agent knows the number $k \geq 2$ of agents.

1. The agents gather on one node. They try ever larger guesses of n and of the length of the identifiers, and for each guess run a procedure that makes agents meet. Since they know k, they know when all of them are together.
2. The agent with the largest identifier draws a map of the graph, using another agent as a movable token, so that it recognises a node it has already seen.
3. Now every agent knows the whole configuration and answers as the computer that sees it would.

Why it works and what it costs. So agents that know $k \geq 2$ decide every decision problem (proof of Theorem 3.2) (Fraigniaud and Pelc 2017). The time is finite but huge, because the meeting procedure uses walks of up to $2n^n$ rounds. A lone agent has no partner to use as a token, which is exactly why it fails above. A lone agent given a map of the graph, without its start node marked, also decides everything, but one that knows only n does not. Fraigniaud and Pelc conclude that knowing the number of agents is very powerful once it is larger than 1, which points to gathering as the key task.

Why it matters. It shows that gathering is not just one task among many: once agents can meet and count themselves, they can compute anything about where they are.

3. Algorithm: One bit of memory is enough with whiteboards. (Izumi et al. 2022) (One agent with whiteboards and little memory, p. 375) Now a single agent has very little memory, but every node has a whiteboard of a constant number of bits.

1. The agent runs a depth-first search in port order that needs only a constant number of bits in its own memory and on each whiteboard (Theorem 3). This puts the nodes in a fixed order.
2. Along that order, the whiteboards serve as the tape of a Turing machine, a model computer that reads and writes a long tape one cell at a time. So the agent can simulate an agent with $O(n)$ bits of memory.
3. The constant number of bits the agent still carries is reduced to one. It keeps them on the whiteboard of its node and moves them to the next node one bit at a time, with one round trip per bit.

Why it works and what it costs. So an agent with one bit of memory decides everything that an agent with $O(n)$ bits decides in polynomial time, with a polynomial slowdown (Theorem 1) (Izumi et al. 2022). Planarity, being bipartite and bounds on the diameter are examples. Zero bits are not enough. Such an agent cannot explore every graph, so on some graph it answers without visiting a node, and adding a leaf there changes the answer to "is n odd?" (Theorem 2).

Why it matters. It shows that the agent's own memory hardly matters once the graph can store information: the whiteboards together act as its memory.

Claude (Anthropic) was used in cataloguing the problems.

What the community started looking at

Fraigniaud and Pelc asked which properties of their starting configuration agents can decide in an anonymous graph, with the acceptance rule of distributed decision (all accept, or someone rejects), and set decision against verification with a certificate, by analogy with P and NP (Fraigniaud and Pelc 2017). They found a complete problem for verification, built from the number of agents and the quotient graph, and showed that knowing the number of agents decides everything once there are at least two, while a lone agent depends on what it is told about the graph. Bampas and Ilcinkas refined the classes with unanimity and with certificates for no answers, using a mobile-agent analogue of dovetailing (Bampas and Ilcinkas 2018). Izumi, Kakizawa, Kawabata, Kitamura and Masuzawa turned to a single agent with little memory and constant-size whiteboards, where one bit of agent memory decides as much as $O(n)$ bits in polynomial time (Izumi et al. 2022).

Further reading

Papers on this problem that are not yet placed in a setting.

- Pattanayak and Pelc (2023). Which functions $f : \mathbb{N}^k \to \mathbb{N}$ teams of deterministic finite automata compute on the half-line, starting in groups at the arguments. A preprint; abstract read.

Relations to other problems

- **Related: Map construction.** Two or more agents that know their number decide every property by gathering and drawing a map, one agent using the others as a token; a lone agent with a map decides every property (Fraigniaud and Pelc 2017).
- **Related: Gathering in graphs.** Gathering is what makes a known number of agents decisive: after it, the agents learn the whole configuration (Fraigniaud and Pelc 2017).
- **Related: Graph exploration by a single agent.** A lone agent with constant memory and constant-size whiteboards explores by a depth-first search in port order (Izumi et al. 2022); without memory it cannot explore every graph (Cohen et al. 2008), which is why it cannot decide whether n is odd.

Assumptions specific to this problem

Number of agents

Fraigniaud and Pelc allow any number of agents, unknown to them, and show that knowing it is very powerful when it is larger than 1 but useless for a single agent (Fraigniaud and Pelc 2017); they then study the single agent separately. Bampas and Ilcinkas name the inability to tell one agent from several as the main source of difficulty (Bampas and Ilcinkas 2018). Izumi et al. study a single agent (Izumi et al. 2022).

Level	Value	Formal definition
0	**any** k	Any number $k \geq 1$ of agents; no agent knows k, so an agent cannot tell whether it is alone.
1	**one agent**	Exactly one agent, $k = 1$: configurations with more agents are excluded.
1	**known** $k \geq 2$	$k \geq 2$ agents, and every agent knows k.

Claude (Anthropic) was used in cataloguing the problems.

Certificates

Fraigniaud and Pelc separate deciding from verifying, as P from NP, and motivate certificates by agents that repeatedly check a property (Fraigniaud and Pelc 2017); Bampas and Ilcinkas add certificates for no answers and argue that verification lets agents check the assumptions of another algorithm, or its output, before relying on it (Bampas and Ilcinkas 2018).

Level	Value	Formal definition
0	**decide**	The agents receive only their identifiers and inputs.
1	**verify**	Each agent also receives a certificate string. For every configuration in Π some certificate makes all agents output yes; for every other configuration, whatever the certificate, some agent outputs no.

Knowledge of the graph

Fraigniaud and Pelc compare, for a single agent, three kinds of information given as oracles: the quotient graph, the number of nodes and a map; they note that the size of the graph and its topology are often given to agents in network tasks, while the quotient graph matters because it is all an agent can learn (Fraigniaud and Pelc 2017).

Level	Value	Formal definition
0	**no knowledge**	The agents know nothing about G.
1	**knows** n	The agents know n (an oracle for the number of nodes in Fraigniaud and Pelc 2017).
2	**map**	The agents have a copy of G with its ports, without their start nodes marked on it.

Case tree

The root assumes nothing; going down, each edge adds the assumptions written in grey (+) or leaves an axis unassumed. Bold entries are catalogued settings with their status per measure (Time, Memory) and page; italic entries are branch points, named after the axis on which their branches differ. Axis order: Graph dynamics → Edge direction → Node identifiers → Identifiers → Communication → Timing → Node memory → Number of agents → Certificates → Knowledge of the graph.

No assumptions

+ static graph, + undirected

Identifiers

+ distinct IDs (eponymous), + local communication, + synchronous

Any number of agents STANDARD SETTING ●● p. 371

+ one agent

One agent ●● p. 372

+ knows n

One agent that knows n ●● p. 373

+ map

One agent with a map ●● p. 373

any k, + verify

Any number of agents, with certificates ●● p. 374

Claude (Anthropic) was used in cataloguing the problems.

+ known $k \geq 2$
Two or more agents that know their number ●● p. 375

anonymous, no communication, + synchronous, + whiteboards, + one agent
One agent with whiteboards and little memory ●● p. 375

What is known

The best upper bound against the best lower bound in each catalogued setting, in tree order. The settings follow in detail.

Setting	Time	Memory	p.
Any number of agents STANDARD SETTING	● impossible	● impossible	371
One agent	● impossible	● impossible	372
One agent that knows n	● impossible	● impossible	373
One agent with a map	● unknown *finite* vs —	● unknown — vs —	373
Any number of agents, with certificates	● impossible	● impossible	374
Two or more agents that know their number	● unknown *finite* vs —	● unknown — vs —	375
One agent with whiteboards and little memory	● unknown poly(n) vs —	● tight 1 vs 1	375

17.7.1 Any number of agents

Adds + distinct IDs (eponymous), + local communication, + synchronous STANDARD SETTING VERIFIED

Not every decidable property can be decided: agents that may be alone cannot tell a long path from a cycle.

Assumed: graph dynamics: static graph; edge direction: undirected; identifiers: distinct identifiers (eponymous): agents have pairwise distinct identifiers; communication: local (co-located agents communicate); timing: synchronous rounds. **Not assumed:** node identifiers: anonymous nodes; node memory: no whiteboards; number of agents: any number of agents, unknown to them; certificates: no certificate (decision); knowledge of the graph: nothing known.

Measure	Lower bound	Best upper bound	Status
Time	*impossible* (Fraigniaud and Pelc 2017) Some decidable problems cannot be decided, e.g. whether G is a path, a tree, or has a given number of nodes (Fraigniaud and Pelc 2017).	—	● impossible
Memory	none known Impossible whatever the memory of the agents.	—	● impossible

Results.

- Fraigniaud and Pelc (2017) [lower bound]: time *impossible*. *Bottleneck:* locality. *Holds when:* inherited from the single-agent setting: a configuration with one agent is a configuration of this setting; the problem whether G is a path, which a centralised algorithm decides: a lone agent that decides after t rounds behaves alike at the centre of a path of $2t + 3$ nodes and on a cycle, both with ports 1 in one direction and 2 in the other (Section 2.2).

Open questions.

Claude (Anthropic) was used in cataloguing the problems.

Q1. What is the least information about the number of agents that lets several agents reconstruct the whole configuration? Posed in (Fraigniaud and Pelc 2017).

Notes. Fraigniaud and Pelc call the decidable problems of this setting MAD (mobile agent decidable) and show that it is strictly contained in MAV, the problems that can be verified with a certificate (Fraigniaud and Pelc 2017). Some global properties are decidable: whether G is a tree with a given number x of nodes, by a depth-first search of $2(x-1)$ steps that draws a map. Bampas and Ilcinkas refine the picture (Bampas and Ilcinkas 2018). Requiring every agent to be right also in no-instances gives a class equal to the problems in both MAD and co-MAD (Theorem 1); MAD and co-MAD differ (Proposition 4, with the problem whether all inputs are empty). Theorem 5 shows that every problem of MAD also has certificates for its no-instances, which makes MAD a part of co-MAV. If only one agent must answer correctly, the class becomes too large: for every $j \geq 1$ it contains a problem complete for the j-th level Σ_j^0 of the arithmetic hierarchy (Theorem 6). Their main tool is a meta-protocol that runs infinitely many protocols in parallel, the mobile-agent analogue of dovetailing. Checked against the full texts (Fraigniaud and Pelc 2017 in arXiv v1, Bampas and Ilcinkas 2018 in its HAL version).

17.7.2 One agent

Adds + one agent **to** Any number of agents (p. 371) VERIFIED

A lone agent can decide only what its view of the graph determines; deciding is then the same as verifying both yes and no answers.

Assumed: graph dynamics: static graph; edge direction: undirected; identifiers: distinct identifiers (eponymous): agents have pairwise distinct identifiers; communication: local (co-located agents communicate); timing: synchronous rounds; number of agents: a single agent. **Not assumed:** node identifiers: anonymous nodes; node memory: no whiteboards; certificates: no certificate (decision); knowledge of the graph: nothing known.

Measure	Lower bound	Best upper bound	Status
Time	*impossible* (Fraigniaud and Pelc 2017)	—	• impossible
	Whether G is a path, a tree, or has a given number of nodes cannot be decided (Fraigniaud and Pelc 2017).		
Memory	none known	—	• impossible
	Impossible whatever the memory of the agents.		

Results.

- Fraigniaud and Pelc (2017) [lower bound]: time *impossible*. *Bottleneck:* locality. *Holds when:* the problem whether G is a path, which a centralised algorithm decides: a lone agent that decides after t rounds behaves alike at the centre of a path of $2t+3$ nodes and on a cycle, both with ports 1 in one direction and 2 in the other (Section 2.2); Fraigniaud and Pelc state the same for whether G is a tree and whether G has a given number of nodes; configurations with a single agent; holds whatever the agents' memory.

Open questions.

Q1. Give a combinatorial characterisation of the problems that a lone agent decides. Posed in (Fraigniaud and Pelc 2017).

Q2. Is it decidable whether a lone agent can decide a given problem: is there a Turing machine that, given a Turing machine deciding membership in Π, decides whether a lone agent can decide Π? Posed in (Fraigniaud and Pelc 2017).

Notes. The quotient graph of G merges nodes with the same view (the infinite tree of port-labelled walks from a node) and is the most a lone agent can learn about G (Lemma 3.1 of Fraigniaud and Pelc 2017, after Yamashita and Kameda). For one agent, the decidable problems are exactly those verifiable

Claude (Anthropic) was used in cataloguing the problems.

both for yes and for no answers (Theorem 4.1); they are strictly fewer than those verifiable for yes or for no answers, which are strictly fewer than those decidable when the agent is given its quotient graph (Theorem 4.2: the product of 'G is a consistently labelled cycle' and 'G is a sun' is decidable with the quotient graph but verifiable neither for yes nor for no answers). Bampas and Ilcinkas note that under this promise most of their classes coincide, and name the inability to tell one agent from several as the main source of difficulty (Bampas and Ilcinkas 2018). Checked against the full texts.

17.7.3 One agent that knows n

Adds + knows n **to** One agent (p. 372) VERIFIED

Knowing n lets a lone agent learn its quotient graph, but not G itself.

Assumed: graph dynamics: static graph; edge direction: undirected; identifiers: distinct identifiers (eponymous): agents have pairwise distinct identifiers; communication: local (co-located agents communicate); timing: synchronous rounds; number of agents: a single agent; knowledge of the graph: n known. **Not assumed:** node identifiers: anonymous nodes; node memory: no whiteboards; certificates: no certificate (decision).

Measure	Lower bound	Best upper bound	Status
Time	*impossible* (Fraigniaud and Pelc 2017) Whether G is a given graph cannot be decided (Fraigniaud and Pelc 2017).	—	• impossible
Memory	none known Impossible whatever the memory of the agents.	—	• impossible

Results.

- Fraigniaud and Pelc (2017) [lower bound]: time *impossible*. *Bottleneck:* symmetry breaking. *Holds when:* the problem whether G is isomorphic to a given graph: two non-isomorphic graphs with the same number of nodes and the same quotient graph (Figure 1 (c) and (d)) look the same to the agent (Theorem 4.3).

Open questions.

Q1. Is it decidable whether a lone agent that knows n can decide a given problem? Posed in (Fraigniaud and Pelc 2017).

Notes. With n known, a view of depth $2(n-1)$ gives the quotient graph (by Norris's theorem), so the agent decides strictly more than with the quotient graph alone: the number of nodes is not determined by the quotient (all consistently labelled cycles have the same one) (Theorem 4.3 of Fraigniaud and Pelc 2017). Checked against the full text (arXiv v1).

17.7.4 One agent with a map

Adds + map **to** One agent (p. 372) VERIFIED

With a map of the graph, a lone agent decides every decidable property.

Assumed: graph dynamics: static graph; edge direction: undirected; identifiers: distinct identifiers (eponymous): agents have pairwise distinct identifiers; communication: local (co-located agents communicate); timing: synchronous rounds; number of agents: a single agent; knowledge of the graph: a map of G. **Not assumed:** node identifiers: anonymous nodes; node memory: no whiteboards; certificates: no certificate (decision).

Claude (Anthropic) was used in cataloguing the problems.

Measure	Lower bound	Best upper bound	Status
Time	none known Only finiteness is shown.	*finite* (Fraigniaud and Pelc 2017)	• unknown
Memory	none known The agents have unlimited memory; no memory bound is studied.	—	• unknown

Results.

- Fraigniaud and Pelc (2017) [current best]: time *finite*. *Assumes:* every decidable problem; the map does not show the agent's start node. Introductory and current best (Theorem 4.3): with the map the agent finds its start node up to a port-preserving automorphism and so knows the whole configuration.

Notes. The chain of Theorem 4.3 of (Fraigniaud and Pelc 2017): a lone agent given its quotient graph decides strictly less than one that knows n, which decides strictly less than one with a map. Checked against the full text (arXiv v1).

17.7.5 Any number of agents, with certificates

Adds any k, + verify **to** Any number of agents (p. 371) VERIFIED

Certificates verify strictly more problems than can be decided, and one problem is complete for them, but some decidable problems still cannot be verified.

Assumed: graph dynamics: static graph; edge direction: undirected; identifiers: distinct identifiers (eponymous): agents have pairwise distinct identifiers; communication: local (co-located agents communicate); timing: synchronous rounds; certificates: a certificate for yes answers (verification). **Not assumed:** node identifiers: anonymous nodes; node memory: no whiteboards; number of agents: any number of agents, unknown to them; knowledge of the graph: nothing known.

Measure	Lower bound	Best upper bound	Status
Time	*impossible* (Bampas and Ilcinkas 2018) Some decidable problems cannot be verified (Bampas and Ilcinkas 2018).	—	• impossible
Memory	none known Impossible whatever the memory of the agents.	—	• impossible

Results.

- Bampas and Ilcinkas (2018) [lower bound]: time *impossible*. *Bottleneck:* locality. *Holds when:* the problem whether every agent's input equals the number of agents: a lone agent on a ring accepts with some certificate; two such agents, placed at the roots of their truncated views joined by an edge, run as if alone, so both accept although there are two of them (Proposition 7); the same argument shows that this problem cannot be verified for no-instances either.

Open questions.

Q1. Do certificates that may depend on the agents' identifiers verify more problems? Raised in (Bampas and Ilcinkas 2018).

Q2. Which problems become verifiable under the promise that there are at most two agents? Raised in (Bampas and Ilcinkas 2018).

Notes. Whether G is a path is verifiable with the number of nodes as certificate, so MAV is strictly larger than MAD; the problems 'some node has degree d', one for each d, form an infinite family of which none reduces to another (Fraigniaud and Pelc 2017). Fraigniaud and Pelc's main result is a complete problem for MAV: the product of teamsize (are there more than a given number of agents?) and quotient (is the quotient graph of G, which merges nodes with the same view, different from a given graph?)

Claude (Anthropic) was used in cataloguing the problems.

(Theorem 3.2). Every verifiable problem reduces to it: with two or more agents, knowing their number lets them gather and draw a map; a lone agent learns the quotient graph, which is all it can ever learn. Bampas and Ilcinkas add certificates for no-instances and unanimous acceptance, and place the nine resulting classes in a complete diagram of inclusions (Table 1 and Figures 2–3 of Bampas and Ilcinkas 2018); for example, whether some node has degree equal to an agent's input is verifiable, but not with unanimous rejection and not for no-instances (Proposition 6).

17.7.6 Two or more agents that know their number

Adds + known $k \geq 2$ **to** Any number of agents (p. 371) VERIFIED

Agents that know their number $k \geq 2$ gather, draw a map and decide every decidable property.

Assumed: graph dynamics: static graph; edge direction: undirected; identifiers: distinct identifiers (eponymous): agents have pairwise distinct identifiers; communication: local (co-located agents communicate); timing: synchronous rounds; number of agents: two or more agents that know their number. **Not assumed:** node identifiers: anonymous nodes; node memory: no whiteboards; certificates: no certificate (decision); knowledge of the graph: nothing known.

Measure	Lower bound	Best upper bound	Status
Time	none known	*finite* (Fraigniaud and Pelc 2017)	• unknown
	Only finiteness is shown; no time bound is studied.		
Memory	none known	—	• unknown
	The agents have unlimited memory; no memory bound is studied.		

Results.

- Fraigniaud and Pelc (2017) [current best]: time *finite*. *Assumes:* every decidable problem. Introductory and current best (proof of Theorem 3.2 and Section 4): the agents gather by rendezvous with growing guesses of n and of the length of identifiers (Claim 3.3), the agent with the largest identifier draws a map of G using the others as a movable token (Claim 3.4), and every agent then knows the whole configuration. The rendezvous runs DFS walks of up to $2n^n$ rounds each, so the time is finite but exponential.

Notes. Fraigniaud and Pelc conclude that knowing the number of agents is very powerful when it is larger than 1, which points to gathering as the key task in mobile agent computing (Fraigniaud and Pelc 2017). Bampas and Ilcinkas remark that the promise of two or more agents already suffices, the agents learning the configuration by rendezvous with growing guesses of n (Bampas and Ilcinkas 2018); they state this in their discussion without a proof. Checked against the full text of (Fraigniaud and Pelc 2017) (arXiv v1).

17.7.7 One agent with whiteboards and little memory

Adds anonymous, no communication, + synchronous, + whiteboards, + one agent VERIFIED

One bit of agent memory, with constant-size whiteboards, decides everything that $O(n)$ bits decide in polynomial time; zero bits do not.

Assumed: graph dynamics: static graph; edge direction: undirected; timing: synchronous rounds; node memory: whiteboards at nodes; number of agents: a single agent. **Not assumed:** node identifiers: anonymous nodes; identifiers: anonymous: all agents are identical; communication: none (silent agents; only see how many agents are co-located); certificates: no certificate (decision); knowledge of the graph: nothing known.

Statement for this setting. One agent, no inputs, whiteboards of $O(1)$ bits that are all 0 at the start. A decision problem is a set of graphs; the agent must stop within poly(n) moves, with polynomial local computation in each step, and write the answer on the whiteboard of its final node. The question is how much memory the agent needs.

Claude (Anthropic) was used in cataloguing the problems.

Measure	Lower bound	Best upper bound	Status
Time	none known Polynomial overhead; no lower bound.	poly(n) (Izumi et al. 2022) [preprint]	● unknown
Memory	1 (Izumi et al. 2022) [preprint] One bit suffices and zero bits do not.	1 (Izumi et al. 2022) [preprint]	● tight

Results.

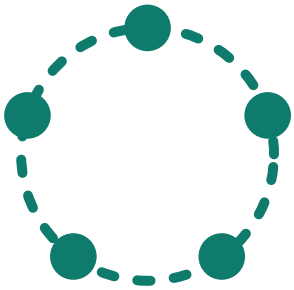
- Izumi et al. (2022) [preprint] [lower bound]: memory 1. *Bottleneck:* memory. *Holds when:* the problem whether n is odd: an agent without memory and with $O(1)$-bit whiteboards cannot explore every graph (Cohen et al. 2008), so on some graph it decides without visiting a node, and a leaf added there changes the answer (Theorem 2, Lemma 4); whiteboards of $O(1)$ bits.
- Izumi et al. (2022) [preprint] [current best]: time poly(n), memory 1. *Assumes:* every problem that an agent with $O(n)$ bits of memory decides in polynomial time with $O(1)$-bit whiteboards; no knowledge of G. Introductory and current best (Theorem 1): the agent orders the nodes by a depth-first search in port order that needs only $O(1)$ bits of memory and of whiteboard (Theorem 3), uses the whiteboards along that order as the tape of a Turing machine that simulates the $O(n)$-bit agent, and keeps paths between the positions it needs with $O(1)$ bits per node; $O(1)$ bits of agent memory are then reduced to one by carrying them across an edge one bit per round trip. The overhead is polynomial.

Notes. Izumi, Kakizawa, Kawabata, Kitamura and Masuzawa call the classes PAGT($g(n), h(n)$): problems decidable in polynomial time with $g(n)$ bits of agent memory and $h(n)$ bits per whiteboard (Izumi et al. 2022). They name planarity, bipartiteness and bounds on the diameter or the treewidth as examples of decision problems in this sense. Checked against the full text (arXiv v1, a preprint).

Claude (Anthropic) was used in cataloguing the problems.

Part IV

Mobile robots in the plane



The model and its assumption axes are in Chapter 3.

Claude (Anthropic) was used in cataloguing the problems.

Chapter 18

Formation problems

Form a geometric pattern from a starting configuration. Gathering (one point, through k-grouping), circle formation and uniform circle formation are special cases; convergence is gathering's relaxation.

Problems in this family:

- Pattern formation
- Circle formation — special case of Pattern formation
- Uniform circle formation — special case of Pattern formation
- Universal pattern formation — strengthening of Pattern formation
- Pattern sequences — generalisation of Pattern formation
- Mutual visibility — special case of Pattern formation
- k-grouping — special case of Pattern formation
- Gathering — special case of k-grouping
- Near-gathering — relaxation of Gathering
- Partitioning — special case of Pattern formation
- Convergence



18.1 Pattern formation

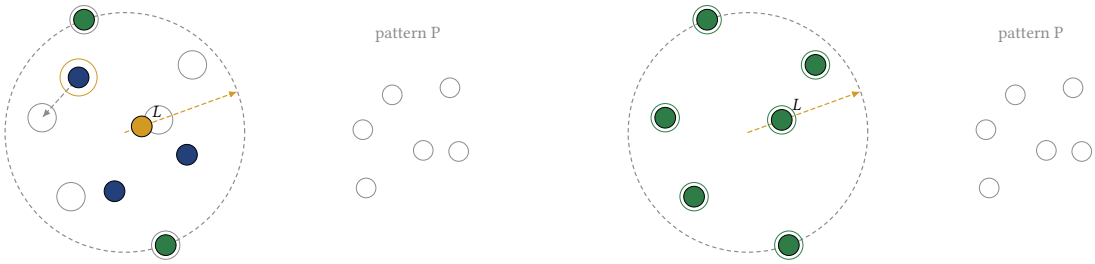
Robots start at distinct points and must form a given geometric pattern, up to similarity; which patterns are formable, and from which initial configurations?

First posed in 1999 (Suzuki and Yamashita 1999). Suzuki and Yamashita characterised the patterns that anonymous robots can form in terms of their initial configuration.

Problem statement and measures

n robots start at distinct points of the plane. Each is given the same **pattern** P, a set (or multiset) of n points in its own local coordinate system. The robots must reach a configuration similar to P (equal up to translation, rotation, uniform scaling and, without **chirality** (a common clockwise direction), reflection) and stay there. The question is for which pairs (initial configuration, pattern) this is possible under the given assumptions, and at what cost in time and colours.

Measures: **Time**, rounds (FSYNC, SSYNC) or epochs (ASYNC) until the task is done; "finite" when only solvability is known; **Colours**, colours of the light per robot (0 for oblivious robots).



(a) Start: SEC = unit circle; leader L fixes east

(b) Final: the robots form a copy of P

Figure 18.1: Pattern formation: n oblivious robots at distinct points are given the same pattern P in their local coordinate systems and must reach a configuration similar to P, up to translation, rotation and scaling (Suzuki and Yamashita 1999). Shown, with chirality, the plan of (Dieudonné et al. 2010) in SSYNC rounds: the smallest enclosing circle (dashed) gives the origin and unit, the ray to the leader L (the robot nearest the centre) gives east; P is placed accordingly (hollow points: the target points), and ringed (activated) robots walk to their target points one at a time without changing the circle or the leader; L moves last.

A first look

Picture drones at a light show that must arrange themselves into a picture, say a star, with no director and no radio. There are n **robots**, tiny points at different places in the plane; the positions of all robots at one instant form a **configuration**. Each robot repeats a **Look–Compute–Move cycle**: it takes a **snapshot** of where the other robots are (Look), decides on a destination point (Compute) and moves towards it (Move). The robots are identical, silent and **oblivious** (they forget everything at the end of each cycle), and each one sees the world in its own **local coordinate system**, with its own origin, unit of length and directions. An adversary called the **scheduler** decides the **activation** of each robot, that is, when it performs a cycle; in the standard version (ASYNC) every phase may take any finite time, so a robot may look while others are still moving. All robots receive the same **pattern** P, a list of n points, and must end in a configuration equal to P up to shifting, turning, resizing and mirroring, and then stop; the question is from which initial configurations this is possible.

1. Algorithm: Elect a leader, then build a shared coordinate system from the smallest enclosing circle. (Dieudonné et al. 2010) (ASYNC with common chirality, p. 386)

1. Every robot computes the **smallest enclosing circle** (SEC): the smallest circle that contains all robots. It is the same circle for everyone, whatever their local coordinate systems.
2. The centre of the SEC becomes the common origin and its radius the common unit of length.
3. The robots elect a **leader**, a robot whose position in the configuration looks different from every other robot's (for example the only robot closest to the centre). The ray from the centre through the leader becomes the common \"east\"; because all robots agree on what clockwise means (**chirality**), this also fixes a common \"north\".
4. Now all robots use the shared coordinate system, so they all place the pattern P in the same spot (scaled to the circle) and compute the same assignment of robots to points of P; the point assigned to a robot is its **target point**.
5. The robots move to their target points in a careful order that never changes the SEC and never lets another robot take over the leader's special position, so the shared coordinate system survives every move.

Why it works and what it costs. Once everyone agrees on coordinates, the task is just \"walk to your

Claude (Anthropic) was used in cataloguing the problems.

target point\"; the difficulty is step 5, because an oblivious robot recomputes everything in every cycle, and one careless move (a robot on the circle stepping inwards, say) can change the circle or the leader and with them every robot's target point. Dieudonné, Petit and Villain used this plan to show that, with chirality, pattern formation is solvable exactly when leader election is, for $n \geq 4$ robots in ASYNC (Dieudonné et al. 2010). The catalogue's current best drops chirality: arbitrary pattern formation is solvable exactly when a leader can be elected (Cicerone et al. 2019). Running time is not analysed in these papers.

Why it matters. It turned a geometric problem into a symmetry-breaking problem: forming any pattern is as hard as choosing one leader.

2. Impossibility: A symmetric initial configuration that the scheduler keeps symmetric forever. (Suzuki and Yamashita 1999) (ASYNC, no agreement on coordinates, p. 386) Place $n = 4$ robots on the corners of a square and ask for a pattern that does not look the same after a quarter turn, for example four points on a line. The adversary gives each robot a local coordinate system with its origin at the robot, the x-axis pointing to the centre of the square, and the same unit; the four systems are then copies of each other turned by 90°. Every robot now gets exactly the same snapshot. The scheduler activates all four together. Identical robots with identical snapshots choose the same move in their own local coordinate systems, which in the real plane is the same move turned by 90°. So after these cycles the robots again form a square (or all stand at its centre), and the scheduler repeats this forever. The line is never formed.

Why it holds. The same argument works for any initial configuration that looks the same after some rotation: whatever the algorithm, the robots can only form patterns that keep at least that much symmetry. Suzuki and Yamashita measured this with the **symmetricity** of a configuration (Suzuki and Yamashita 1999). The scheduler only needs to activate every robot at once, so the obstacle holds in the fully synchronous model (FSYNC) and therefore under every less synchronous scheduler. Hence, without any agreement between local coordinate systems, an arbitrary pattern cannot be formed from every initial configuration (Flocchini et al. 2008a); from a regular n-gon only a point and the regular n-gon itself can be formed (Flocchini et al. 2017).

Why it matters. It explains why every pattern-formation result is about breaking symmetry, and why leader election keeps appearing.

3. Algorithm: With a common compass nobody is symmetric. (Flocchini et al. 2008a) (ASYNC with a common compass, p. 389) Suppose all robots agree on north and east (a **common compass**), though not on origin or unit.

1. Each robot sorts all robots from west to east, and robots on the same north-south line from south to north. Since all robots agree on north and east, they all obtain the same order, even though their origins and units differ.
2. Each robot sorts the points of the pattern P in the same way. At first only the first robot moves: staying first in the order, it moves until the direction from it to the second robot is the direction from the first to the second point of P.
3. Every robot now places P so that its first two points fall on the first two robots. This fixes where P lies and how large it is, so all robots compute the same target points.
4. The other robots walk to the remaining target points one at a time, in an order chosen so that once a robot starts moving, no other robot moves until it has arrived.

Why it works and what it costs. A common compass removes the adversary's trick: no rotation or mirror image can map a set of points to itself while keeping north and east fixed, so every configuration has a first robot that all robots recognise. Flocchini, Prencipe, Santoro and Widmayer showed that with a common compass any number of robots form any pattern in ASYNC, and that agreement on one axis,

Claude (Anthropic) was used in cataloguing the problems.

without a common handedness, suffices for an odd number of robots (Flocchini et al. 2008a). The steps above follow their algorithm for a common compass (Theorem 4.1 of Flocchini et al. 2008a); the fact that a compass gives all robots the same order is also Lemma 2.1 of (Efrima and Peleg 2009).

Why it matters. It shows that symmetry, not obliviousness or asynchrony, is the real obstacle to forming patterns.

What the community started looking at

The founding question was **which patterns robots can form at all** (Suzuki and Yamashita 1999): Suzuki and Yamashita characterised formable patterns by the symmetry of the starting configuration, for SSYNC robots with unbounded memory and common handedness: only a point and a regular n-gon can be formed from every starting configuration. Flocchini, Prencipe, Santoro and Widmayer then asked the question that shaped the field, **how much agreement is needed**: none is not enough, one axis is enough for an odd number of robots, two axes for all (Flocchini et al. 1999, 2008a). The aim has been to remove assumptions one at a time (chirality, synchrony, rigidity), ending in the equivalence with leader election without chirality (Cicerone et al. 2019).

Further reading

Papers on this problem that are not yet placed in a setting.

- Kojima et al. (2026). Hardwired pattern formation: the robots share a unit of distance and chirality, and must form the pattern at its given size, that is, reach a configuration congruent to it (translation and rotation, no scaling), from distinct starting points. For two oblivious robots, SSYNC with non-rigid moves suffices for every positive target distance (Theorem 1; distance 0 is rendezvous, impossible, Observation 1), ASYNC does not for any target distance above δ (Theorem 2), and ASYNC robots with a light of two colours succeed by a self-stabilizing adaptation of a rendezvous algorithm of (Défago et al. 2023) (Theorem 3, Corollary 1). For $n \geq 5$ oblivious SSYNC robots, a size-adjusting algorithm brings the smallest enclosing circle to the given size without increasing the symmetricity (Theorem 4); followed by the algorithm of (Fujinaga et al. 2015), which keeps the smallest enclosing circle fixed, it forms the pattern exactly when the symmetricity of the initial configuration divides that of the pattern (Theorem 5), so this result rests on (Fujinaga et al. 2015), whose proof is disputed (see the setting with chirality). A common unit of distance is not an axis of this model. Read (arXiv v1).
- Mondal et al. (2024a). Arbitrary pattern formation on a circle: oblivious robots without chirality, confined to a circle, under SSYNC. Formable configurations are necessary, and from every one of them the robots form any pattern in $O(n)$ epochs without collisions (Theorem 36). Read (arXiv v2).
- Sil and Chaudhuri (2017). Line formation by asynchronous oblivious fat robots with limited visibility that agree on both axes, without collisions (Theorem 1). Read (arXiv v1).
- Krupke et al. (2015). A robotics relative: n robots with unique labels and a connected communication graph arrange themselves into a sorted, equally spaced array in $O(n)$ time with $O(n^2)$ messages and $O(nD)$ travel. Read (arXiv v1).
- Bose et al. (2020a). Arbitrary pattern formation on the infinite grid by ASYNC oblivious robots without common coordinates or chirality, from every asymmetric configuration. Abstract read.
- Ghosh et al. (2023a). Arbitrary pattern formation on the infinite grid by ASYNC robots that align their axes with the grid lines, from asymmetric configurations, with an optimal number of moves and optimal time (improving on $O(D^2k)$ moves). Abstract read.
- Kundu et al. (2022). Arbitrary pattern formation on the infinite grid by ASYNC opaque robots with lights and no common coordinates. A preprint; abstract read.
- Cicerone et al. (2023a). Arbitrary pattern formation on the square, triangular and hexagonal grids from asymmetric configurations. Abstract read.

Claude (Anthropic) was used in cataloguing the problems.

- Alsaedi et al. (2025a). Pattern formation by fat (unit-disc) robots with obstructed visibility and $O(1)$ bits of memory that is not communicated: mutual visibility, then leader election, then the pattern. Abstract read.
- Bose et al. (2021b). Approximate arbitrary pattern formation by robots whose moves err in direction and length: impossible without common coordinates from some symmetric configurations even in FSYNC, and solvable cases characterised. Abstract read.
- Pattanayak et al. (2020b). FSYNC oblivious robots must form a conic through f robots that crashed at the start (a line for $f = 2$, a circle for $f = 3$, a parabola for $f = 4$, an ellipse, parabola or hyperbola for $f = 5$): two rounds are needed and suffice, from asymmetric configurations with $n \geq 2f + 1$ robots. Model and theorems read.
- Tomita et al. (2017). Plane formation in three-dimensional space by FSYNC oblivious robots without chirality. Abstract read.
- Coleman et al. (2020). Pattern formation that minimises the largest distance travelled: necessary conditions and a solution for three robots. A preprint; abstract read.
- Ehresmann et al. (2019). Agents with q colours on an n-node ring rearrange so that each block of p nodes meets given colour counts; time bounds for two colours and more. Abstract read.
- Coppola et al. (2019). Automatic design and verification of stochastic local rules by which memoryless robots with local sensing and a common North form a desired pattern. Abstract read.
- Coppola et al. (2018). Preprint of the same line of work: a local proof procedure that a swarm of reactive robots reaches a desired pattern from any initial configuration. Abstract read.
- Ghosh et al. (2023b). Two teams of oblivious robots on a grid, one gathering and one forming a pattern, without any robot knowing the team of another. Abstract read.
- Asahiro et al. (2022). Monotonic self-stabilization: from any configuration, robots move to a goal formation so that a quality measure chosen by the user (distance to the goal, accuracy of the formation) never gets worse. The robots have unique identifiers, chirality and a common unit of distance, know every robot's goal position, and move a bounded distance per cycle; parameterised algorithms for FSYNC (Theorems 2 and 10) and SSYNC (Theorems 13 and 26). Read (model and theorem statements).
- Cicerone (2021). Symmetry breaking on the infinite square and triangular grids by oblivious, disoriented ASYNC robots with strong multiplicity detection (Theorem 13), used as a module for line formation, in which $n \geq 3$ robots occupy n consecutive nodes of a grid line: solvable exactly from starts with a leader and from one class of starts whose symmetry is a rotation by 180° without reflection (Theorem 31); from the others not even by FSYNC robots (Theorem 15). Read in full.

Relations to other problems

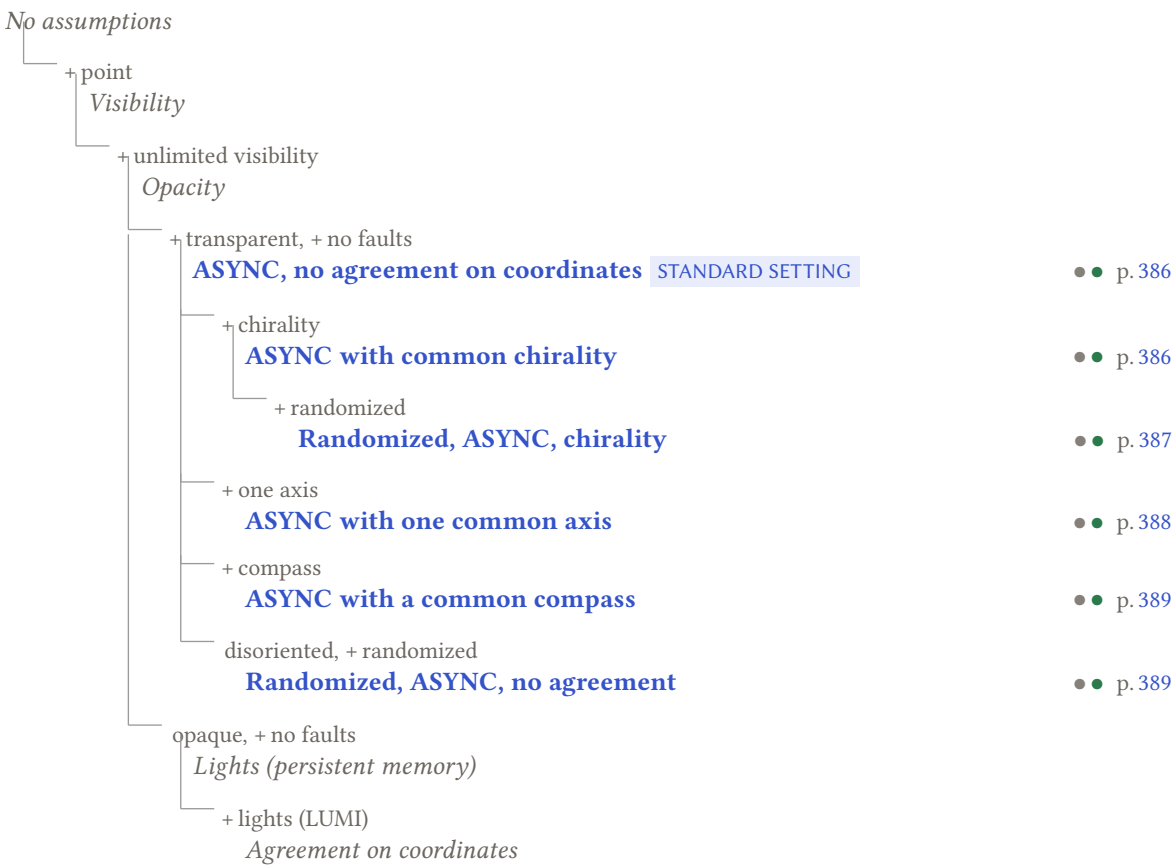
- **Derived problem: Circle formation** (special case of). A circle is a family of patterns; forming some circle is easier than forming a given pattern, and it is the precursor of uniform circle formation (Feletti et al. 2026a).
- **Derived problem: Uniform circle formation** (special case of). Uniform circle formation is "an important special case of the geometric pattern formation problem" (Feletti et al. 2026a): the target is the regular n-gon. It is also the special case of circle formation with equal spacing; circle formation is "its precursor" (Feletti et al. 2026a).
- **Derived problem: Universal pattern formation** (strengthening of). Pattern formation asks which patterns are formable from a given start; universal pattern formation asks for all patterns from all starts, which symmetry forbids under SSYNC and FSYNC but sequential schedulers allow (Flocchini et al. 2026).
- **Derived problem: Pattern sequences** (generalisation of). Forming one pattern is the case $m = 1$; a sequence needs the robots to remember (or encode in the configuration) which pattern comes next.

Claude (Anthropic) was used in cataloguing the problems.

- **Derived problem: Mutual visibility** (special case of). Mutual visibility asks for any pattern in general position, a family of patterns; it is the usual first step of pattern formation for opaque robots.
- **Derived problem: k-grouping** (special case of). When the groups sit at points, k-grouping forms a pattern of k points; which points, and how many robots each holds, is up to the robots. Gathering is its special case $k = 1$. Splitting needs symmetry breaking (Efrima and Peleg 2009).
- **Derived problem: Partitioning** (special case of). A partition into k groups at points is a pattern of k points, each of multiplicity n/k.
- **Related: Leader election in the plane.** Arbitrary pattern formation is solvable from a configuration exactly when leader election is, in ASYNC without chirality (Cicerone et al. 2019); with chirality and $n \geq 4$ this was shown by (Dieudonné et al. 2010).
- **Related: Shape formation in programmable matter.** "Shape formation is a basic distributed problem for systems of computational mobile entities. Intensively studied for systems of autonomous mobile robots, it has recently been investigated in the realm of programmable matter" (Di Luna et al. 2020e).

Case tree

The root assumes nothing; going down, each edge adds the assumptions written in grey (+) or leaves an axis unassumed. Bold entries are catalogued settings with their status per measure (Time, Colours) and page; italic entries are branch points, named after the axis on which their branches differ. Axis order: Body → Visibility → Opacity → Faults → Scheduler → Lights (persistent memory) → Agreement on coordinates → Movement → Multiplicity detection → Randomness → Identifiers.



Claude (Anthropic) was used in cataloguing the problems.

+ one axis, + rigid
Opaque robots with lights, ASYNC, one axis, rigid ●● p. 390

+ compass, + rigid
Opaque robots with lights, ASYNC, compass, rigid ●● p. 390

oblivious, + compass
Opaque oblivious robots, ASYNC, compass ●● p. 391

limited visibility, + transparent, + no faults
Limited visibility ●● p. 391

fat, + unlimited visibility, + no faults
Scheduler

ASYNC, + lights (LUMI), + one axis
Opaque fat robots with lights, ASYNC, one axis ●● p. 392

+ FSYNC, + lights (LUMI), + rigid, + randomized
Fat robots with lights, FSYNC, randomized, no agreement ●● p. 392

What is known

The best upper bound against the best lower bound in each catalogued setting, in tree order. The settings follow in detail.

Setting	Time	Colours	p.
ASYNC, no agreement on coordinates STANDARD SETTING	● unknown *finite* vs *impossible*	● tight 0 vs —	386
ASYNC with common chirality	● unknown *finite* vs —	● tight 0 vs —	386
Randomized, ASYNC, chirality	● unknown *finite* vs —	● tight 0 vs —	387
ASYNC with one common axis	● unknown *finite* vs *impossible*	● tight 0 vs —	388
ASYNC with a common compass	● unknown *finite* vs —	● tight 0 vs —	389
Randomized, ASYNC, no agreement	● unknown — vs —	● tight — vs —	389
Opaque robots with lights, ASYNC, one axis, rigid	● unknown *finite* vs —	● unknown 6 vs —	390
Opaque robots with lights, ASYNC, compass, rigid	● unknown *finite* vs —	● unknown 3 vs —	390
Opaque oblivious robots, ASYNC, compass	● impossible	● impossible	391
Limited visibility	● unknown *finite* vs —	● tight 0 vs —	391
Opaque fat robots with lights, ASYNC, one axis	● unknown *finite* vs —	● unknown 10 vs —	392
Fat robots with lights, FSYNC, randomized, no agreement	● unknown $O(n) + O(\kappa \log n)$ vs —	● unknown 7 vs —	392

Claude (Anthropic) was used in cataloguing the problems.

18.1.1 ASYNC, no agreement on coordinates

Adds + transparent, + no faults STANDARD SETTING VERIFIED

Without any common orientation, an arbitrary pattern can be formed exactly from the configurations in which a leader can be elected.

Assumed: body: point robots; visibility: unlimited; opacity: transparent: every robot sees every robot within range; faults: no faults. **Not assumed:** scheduler: asynchronous (ASYNC); lights (persistent memory): oblivious, no light (OBLOT); agreement on coordinates: no agreement on local coordinate systems; movement: non-rigid: the adversary may stop a move after some minimum distance δ; multiplicity detection: none: a point with several robots looks like one robot; randomness: deterministic; identifiers: anonymous: all robots are identical.

Measure	Lower bound	Best upper bound	Status
Time	*impossible* (Flocchini et al. 2008a) Time is not analysed in these papers.	*finite* (Cicerone et al. 2019)	● unknown
Colours	none known Oblivious robots: no colours.	0 (Cicerone et al. 2019)	● tight

Results.

- Flocchini et al. (2008a) [lower bound]: time *impossible*. *Bottleneck:* symmetry. *Holds when:* for some initial configurations: symmetric configurations cannot be broken by oblivious deterministic robots; Theorem 7.1: from a regular n-gon whose local coordinate systems are rotated by $360^\circ/n$ from one robot to the next, all robots keep the same view; holds in every scheduler.
- Flocchini et al. (2008a) [introductory]: time *finite*, colours 0. Introduces arbitrary pattern formation for asynchronous oblivious robots and shows that without agreement on both axes an arbitrary pattern cannot be formed in general (Theorem 7.1); also that forming arbitrary patterns implies leader election for $n \geq 3$ (Theorem 2.1).
- Cicerone et al. (2019) [current best]: time *finite*, colours 0. Characterisation: arbitrary pattern formation is solvable exactly when leader election is, in ASYNC without chirality. The paper also states that earlier proofs (Fujinaga et al. 2015, Bramas and Tixeuil 2016, the arXiv version of Dieudonné et al. 2010) are not correct.

Open questions.

Q1. Bound the number of epochs needed by an arbitrary pattern formation algorithm in ASYNC; no time analysis is recorded.

Notes. Patterns are formed up to translation, rotation, scaling and reflection, starting from distinct positions. The four-robot case needed a separate geometric invariant, the H-segment of a quadrilateral, with or without chirality (Bramas and Tixeuil 2018). The point (gathering) and the regular polygon (uniform circle) are the only patterns formable from every initial configuration (Flocchini et al. 2017).

18.1.2 ASYNC with common chirality

Adds + chirality **to** ASYNC, no agreement on coordinates (p. 386) VERIFIED

With common chirality, a pattern F is formable from a configuration R exactly when $\rho(R)$ divides $\rho(F)$; pattern formation and leader election are equivalent for $n \geq 4$.

Assumed: body: point robots; visibility: unlimited; opacity: transparent: every robot sees every robot within range; faults: no faults; agreement on coordinates: common chirality (same clockwise orientation). **Not assumed:** scheduler: asynchronous (ASYNC); lights (persistent memory): oblivious, no light (OBLOT); movement: non-rigid: the adversary may stop a move after some minimum distance δ; multiplicity detection: none: a point with several robots looks like one robot; randomness: deterministic; identifiers: anonymous: all robots are identical.

Claude (Anthropic) was used in cataloguing the problems.

Measure	Lower bound	Best upper bound	Status
Time	none known Time not analysed.	*finite* (Cicerone et al. 2021c)	● unknown
Colours	none known No colours.	0 (Cicerone et al. 2021c)	● tight

Results.

- Cicerone et al. (2021c) [current best]: time *finite*, colours 0. *Assumes:* global strong multiplicity detection; patterns may contain multiplicities, with $|F| = |R|$; ASYNC, oblivious robots. Current best, a characterisation (Theorem 6.12): an algorithm forms F from R if and only if $\rho(R)$ divides $\rho(F)$; necessity is the FSYNC bound of Suzuki and Yamashita (Theorem 3.3). The algorithm is split into eleven tasks, each with its precondition, and the proof checks the transitions between tasks, following the methodology of (Cicerone et al. 2021a). So ASYNC robots with chirality form exactly the patterns that FSYNC robots form.
- Cicerone et al. (2021a) [also]: time *finite*, colours 0. *Assumes:* global strong multiplicity detection; patterns may contain multiplicities, with $|F| = |R|$; ASYNC, oblivious robots; the local coordinate system of a robot may change between cycles. Theorem 3 (arXiv version): an algorithm forms F from R if and only if $\rho(R)$ divides $\rho(F)$, where the symmetricity ρ is the largest m such that the configuration splits into regular m-gons centred at the centre of its smallest enclosing circle; necessity is Suzuki and Yamashita's FSYNC bound. A new algorithm for the problem of (Fujinaga et al. 2015), whose authors, by personal communication, gave up fixing theirs. The case study of the authors' design methodology; the algorithm appeared in a journal as (Cicerone et al. 2021c).
- Dieudonné et al. (2010) [also]: time *finite*, colours 0. *Assumes:* $n \geq 4$. The equivalence of pattern formation and leader election with chirality, for $n \geq 4$.
- Fujinaga et al. (2015) [disputed proof]: time *finite*, colours 0. Patterns formable by oblivious ASYNC robots are exactly those formable by non-oblivious FSYNC robots, except a point of multiplicity 2. Its proof is reported as not correct by (Cicerone et al. 2019).

Notes. Disputed: the correctness of the proof of (Fujinaga et al. 2015) is questioned by (Cicerone et al. 2019); it is shown as disputed and never as the current best. Cicerone, Di Stefano and Navarra design a new algorithm for the same problem with a structured correctness proof and so characterise it (Cicerone et al. 2021a) (arXiv version read) (Cicerone et al. 2021c) (read in full).

18.1.3 Randomized, ASYNC, chirality

Adds + randomized **to** ASYNC with common chirality (p. 386) VERIFIED

With random bits and common handedness, $n \geq 5$ robots form any pattern without multiplicities from any initial configuration, with probability 1.

Assumed: body: point robots; visibility: unlimited; opacity: transparent: every robot sees every robot within range; faults: no faults; agreement on coordinates: common chirality (same clockwise orientation); randomness: randomized. **Not assumed:** scheduler: asynchronous (ASYNC); lights (persistent memory): oblivious, no light (OBLOT); movement: non-rigid: the adversary may stop a move after some minimum distance δ; multiplicity detection: none: a point with several robots looks like one robot; identifiers: anonymous: all robots are identical.

Statement for this setting. Robots may draw random bits in Compute. From any initial configuration of distinct points, the robots must form the given pattern with probability 1 and then stop moving.

Measure	Lower bound	Best upper bound	Status
Time	none known Time not analysed.	*finite* (Yamauchi and Yamashita 2014)	● unknown
Colours	none known No colours.	0 (Yamauchi and Yamashita 2014)	● tight

Claude (Anthropic) was used in cataloguing the problems.

Results.

- Yamauchi and Yamashita (2014) [introductory]: time *finite*, colours 0. *Assumes:* $n \geq 5$; neither the initial configuration nor the pattern has multiplicities; non-rigid moves. Theorem 5: forms any pattern from any initial configuration with probability 1. A randomized phase reaches a configuration with symmetricity 1 (Theorem 4); the robots then run the deterministic algorithm of (Fujinaga et al. 2015), whose proof (Cicerone et al. 2019) report as not correct, so this result depends on a disputed proof.

Open questions.

Q1. A proof that does not rely on the disputed algorithm of (Fujinaga et al. 2015), for example by combining the symmetry-breaking phase of (Yamauchi and Yamashita 2014) with the algorithm of (Cicerone et al. 2019).

Notes. Yamauchi and Yamashita's local coordinate systems are all right-handed (chirality). Checked against the full text of (Yamauchi and Yamashita 2014) (DISC 2014): the statements of Theorems 4 and 5 and the model; the combination with ψ_{CWM} rests on (Fujinaga et al. 2015).

18.1.4 ASYNC with one common axis

Adds + one axis **to** ASYNC, no agreement on coordinates (p. 386) VERIFIED

With a common North but no common handedness, any odd number of robots forms any pattern; an even number cannot, in the worst case.

Assumed: body: point robots; visibility: unlimited; opacity: transparent: every robot sees every robot within range; faults: no faults; agreement on coordinates: agreement on one axis (no common chirality). **Not assumed:** scheduler: asynchronous (ASYNC); lights (persistent memory): oblivious, no light (OBLOT); movement: non-rigid: the adversary may stop a move after some minimum distance δ; multiplicity detection: none: a point with several robots looks like one robot; randomness: deterministic; identifiers: anonymous: all robots are identical.

Measure	Lower bound	Best upper bound	Status
Time	*impossible* (Flocchini et al. 2008a) Time not analysed.	*finite* (Flocchini et al. 2008a)	● unknown
Colours	none known No colours.	0 (Flocchini et al. 2008a)	● tight

Results.

- Flocchini et al. (2008a) [lower bound]: time *impossible*. *Bottleneck:* symmetry. *Holds when:* even n, for some initial configurations and patterns (Corollary 2.1); leader election is impossible for even n (Theorem 2.2): robots placed symmetrically about a line parallel to the common axis, with mirrored local coordinate systems and partners activated together, stay symmetric (an SSYNC scheduler, so it holds in ASYNC too).

- Flocchini et al. (2008a) [current best]: time *finite*, colours 0. *Assumes:* odd n. Introductory and current best (Result 2, Theorem 5.1): any pattern for odd n. For even n the formable patterns are characterised exactly (Result 3, Theorem 6.2).

Notes. The robots of (Flocchini et al. 2008a) agree on the direction and orientation of one axis but have no common handedness: "knowing North does not distinguish East from West". Checked against the full text of (Flocchini et al. 2008a).

Claude (Anthropic) was used in cataloguing the problems.

18.1.5 ASYNC with a common compass

Adds + compass **to** ASYNC, no agreement on coordinates (p. 386) VERIFIED

If robots agree on both axes, any number of robots forms any pattern.

Assumed: body: point robots; visibility: unlimited; opacity: transparent: every robot sees every robot within range; faults: no faults; agreement on coordinates: agreement on both axes (common compass). **Not assumed:** scheduler: asynchronous (ASYNC); lights (persistent memory): oblivious, no light (OBLOT); movement: non-rigid: the adversary may stop a move after some minimum distance δ; multiplicity detection: none: a point with several robots looks like one robot; randomness: deterministic; identifiers: anonymous: all robots are identical.

Measure	Lower bound	Best upper bound	Status
Time	none known Time not analysed.	*finite* (Flocchini et al. 2008a)	● unknown
Colours	none known No colours.	0 (Flocchini et al. 2008a)	● tight

Results.

- Flocchini et al. (2008a) [current best]: time *finite*, colours 0. Introductory and current best (Theorem 4.1): any pattern, any n, collision-free, non-rigid moves; the pattern keeps the orientation in which it is given.

Notes. Checked against the full text of (Flocchini et al. 2008a).

18.1.6 Randomized, ASYNC, no agreement

Adds disoriented, + randomized **to** ASYNC, no agreement on coordinates (p. 386) VERIFIED

Randomness breaks symmetry: with one random bit per cycle, $n \geq 5$ robots form any pattern except gathering.

Assumed: body: point robots; visibility: unlimited; opacity: transparent: every robot sees every robot within range; faults: no faults; randomness: randomized. **Not assumed:** scheduler: asynchronous (ASYNC); lights (persistent memory): oblivious, no light (OBLOT); agreement on coordinates: no agreement on local coordinate systems; movement: non-rigid: the adversary may stop a move after some minimum distance δ; multiplicity detection: none: a point with several robots looks like one robot; identifiers: anonymous: all robots are identical.

Measure	Lower bound	Best upper bound	Status
Time	none known Time not analysed.	—	● unknown
Colours	none known No colours.	—	● tight

Results.

- Bramas and Tixeuil (2016) [disputed proof]: time *finite*, colours 0. *Assumes:* $n \geq 5$; one random bit per cycle. Patterns with multiplicity points are allowed, except gathering. Its proof is reported as not correct by (Cicerone et al. 2019).

Notes. Yamauchi and Yamashita gave a randomized algorithm that forms any pattern from any initial configuration in ASYNC with non-rigid moves, for $n \geq 5$ and patterns and initial configurations without multiplicities (Yamauchi and Yamashita 2014); their robots have right-handed local coordinate systems (chirality), so the result belongs to the setting *Randomized, ASYNC, chirality*, and its second phase runs the deterministic algorithm of (Fujinaga et al. 2015), whose proof (Cicerone et al. 2019) report as not

Claude (Anthropic) was used in cataloguing the problems.

correct. **Disputed:** the proof of (Bramas and Tixeuil 2016) is questioned by (Cicerone et al. 2019); no undisputed result is recorded for this setting.

18.1.7 Opaque robots with lights, ASYNC, one axis, rigid

Adds + one axis, + rigid VERIFIED

Opaque point robots with lights that agree on one axis form any pattern from the starts the paper characterises, with 6 colours.

Assumed: body: point robots; visibility: unlimited; faults: no faults; lights (persistent memory): full light (LUMI): every robot sees all lights, including its own; agreement on coordinates: agreement on one axis (no common chirality); movement: rigid: every move reaches its destination. **Not assumed:** opacity: opaque: collinear robots hide each other; scheduler: asynchronous (ASYNC); multiplicity detection: none: a point with several robots looks like one robot; randomness: deterministic; identifiers: anonymous: all robots are identical.

Measure	Lower bound	Best upper bound	Status
Time	none known —	*finite* (Bose et al. 2021a)	● unknown
Colours	none known No lower bound beyond trivial; the minimum is open.	6 (Bose et al. 2021a)	● unknown

Results.

- Bose et al. (2021a) [current best]: time *finite*, colours 6. Introductory and current best. Formation fails from starts symmetric about a robot-free line parallel to the agreed axis.

Open questions.

Q1. The minimum number of colours, and formation with chirality only or with no agreement (Bose et al. 2021a).

Notes. Checked against the full text.

18.1.8 Opaque robots with lights, ASYNC, compass, rigid

Adds + compass, + rigid VERIFIED

With a common compass, opaque robots with lights form any pattern with 3 colours.

Assumed: body: point robots; visibility: unlimited; faults: no faults; lights (persistent memory): full light (LUMI): every robot sees all lights, including its own; agreement on coordinates: agreement on both axes (common compass); movement: rigid: every move reaches its destination. **Not assumed:** opacity: opaque: collinear robots hide each other; scheduler: asynchronous (ASYNC); multiplicity detection: none: a point with several robots looks like one robot; randomness: deterministic; identifiers: anonymous: all robots are identical.

Measure	Lower bound	Best upper bound	Status
Time	none known —	*finite* (Bose et al. 2021a)	● unknown
Colours	none known The minimum is open.	3 (Bose et al. 2021a)	● unknown

Results.

- Bose et al. (2021a) [current best]: time *finite*, colours 3. Introductory and current best.

Claude (Anthropic) was used in cataloguing the problems.

Notes. Checked against the full text.

18.1.9 Opaque oblivious robots, ASYNC, compass

Adds oblivious, + compass VERIFIED

Without lights, opaque robots cannot form arbitrary patterns in ASYNC, even with a common compass.

Assumed: body: point robots; visibility: unlimited; faults: no faults; agreement on coordinates: agreement on both axes (common compass). **Not assumed:** opacity: opaque: collinear robots hide each other; scheduler: asynchronous (ASYNC); lights (persistent memory): oblivious, no light (OBLOT); movement: non-rigid: the adversary may stop a move after some minimum distance δ; multiplicity detection: none: a point with several robots looks like one robot; randomness: deterministic; identifiers: anonymous: all robots are identical.

Measure	Lower bound	Best upper bound	Status
Time	*impossible* (Bose et al. 2021a) Unsolvable without lights.	—	• impossible
Colours	none known Needs lights.	—	• impossible

Results.

- Bose et al. (2021a) [lower bound]: time *impossible*. *Bottleneck:* obstruction. *Holds when:* Theorem 1: oblivious opaque point robots, ASYNC; holds even with a common compass.

Notes. Checked against the full text.

18.1.10 Limited visibility

Adds limited visibility, + transparent, + no faults VERIFIED

With limited visibility, oblivious robots form a pattern under the usual symmetry condition if they start within diameter 1; the general case is open.

Assumed: body: point robots; opacity: transparent: every robot sees every robot within range; faults: no faults. **Not assumed:** visibility: limited: robots see only within a fixed radius; scheduler: asynchronous (ASYNC); lights (persistent memory): oblivious, no light (OBLOT); agreement on coordinates: no agreement on local coordinate systems; movement: non-rigid: the adversary may stop a move after some minimum distance δ; multiplicity detection: none: a point with several robots looks like one robot; randomness: deterministic; identifiers: anonymous: all robots are identical.

Measure	Lower bound	Best upper bound	Status
Time	none known Time not analysed.	*finite* (Hahn et al. 2024)	• unknown
Colours	none known No colours.	0 (Hahn et al. 2024)	• tight

Results.

- Hahn et al. (2024) [current best]: time *finite*, colours 0. *Assumes:* initial diameter at most 1 (the visibility range). Introductory and current best for oblivious robots.

Open questions.

Q1. Arbitrary pattern formation by oblivious robots with limited visibility from a general (connected) initial configuration (Hahn et al. 2024).

Claude (Anthropic) was used in cataloguing the problems.

18.1.11 Opaque fat robots with lights, ASYNC, one axis

Adds ASYNC, + lights (LUMI), + one axis VERIFIED

Opaque fat robots with lights that agree on one axis form any pattern from every start not symmetric about a robot-free line parallel to that axis, with 10 colours.

Assumed: visibility: unlimited; faults: no faults; lights (persistent memory): full light (LUMI): every robot sees all lights, including its own; agreement on coordinates: agreement on one axis (no common chirality). **Not assumed:** body: fat robots (unit discs); opacity: opaque: collinear robots hide each other; scheduler: asynchronous (ASYNC); movement: non-rigid: the adversary may stop a move after some minimum distance δ; multiplicity detection: none: a point with several robots looks like one robot; randomness: deterministic; identifiers: anonymous: all robots are identical.

Measure	Lower bound	Best upper bound	Status
Time	none known —	*finite* (Bose et al. 2020b)	• unknown
Colours	none known The minimum is open.	10 (Bose et al. 2020b)	• unknown

Results.

- Bose et al. (2020b) [current best]: time *finite*, colours 10. Introductory and current best. Theorem 1: solvable if and only if the start is not symmetric about a line parallel to the agreed axis through no robot centre. $\Theta(n)$ moves in total (Theorem 8), analysed under SSYNC with rigid moves.

Notes. Checked against the full text (arXiv 1910.02706).

18.1.12 Fat robots with lights, FSYNC, randomized, no agreement

Adds + FSYNC, + lights (LUMI), + rigid, + randomized VERIFIED

Randomized fat robots with lights and no common orientation form any pattern in FSYNC in $O(n)+O(\kappa \log n)$ rounds with probability at least $1 - n^{-\kappa}$, with 7 colours.

Assumed: visibility: unlimited; faults: no faults; scheduler: fully synchronous (FSYNC); lights (persistent memory): full light (LUMI): every robot sees all lights, including its own; movement: rigid: every move reaches its destination; randomness: randomized. **Not assumed:** body: fat robots (unit discs); opacity: opaque: collinear robots hide each other; agreement on coordinates: no agreement on local coordinate systems; multiplicity detection: none: a point with several robots looks like one robot; identifiers: anonymous: all robots are identical.

Measure	Lower bound	Best upper bound	Status
Time	none known No lower bound recorded.	$O(n) + O(\kappa \log n)$ (Alsaedi et al. 2025b)	• unknown
Colours	none known The minimum is open.	7 (Alsaedi et al. 2025b)	• unknown

Results.

- Alsaedi et al. (2025b) [current best]: time $O(n) + O(\kappa \log n)$, colours 7. *Assumes:* n unknown; the pattern is scaled by a constant (8 colours without scaling). Introductory and current best (Theorems 11, 16, 20, 21); with probability at least $1 - n^{-\kappa}$. Earlier versions used 10 and 11 colours.

Notes. With $O(1)$ words of $O(\log n)$ bits of persistent memory instead of lights, the same bound holds (Alsaedi, Gudmundsson and van Renssen, Computational Geometry 2025). Checked against the full text of (Alsaedi et al. 2025b).

Claude (Anthropic) was used in cataloguing the problems.

18.2 Circle formation

Robots must reach positions on a common circle.

First posed in 2002 (Défago and Konagaya 2002).

Problem statement and measures

n robots start at distinct points and must reach a configuration in which all of them lie on one circle, at distinct points, and stay there. Uniform circle formation additionally asks for a regular n-gon.

Measures: **Time**, rounds (FSYNC, SSYNC) or epochs (ASYNC) until the task is done; "finite" when only solvability is known; **Colours**, colours of the light per robot (0 for oblivious robots).



(a) Start: the smallest enclosing circle is fixed

(b) Final: all robots on the circle (arrows: moves; dashed circles: starting positions)

Figure 18.2: Circle formation: robots at distinct points must place themselves at distinct points of one common circle and stay there; the spacing need not be uniform (Défago and Konagaya 2002). Shown: the first phase of (Flocchini et al. 2017) in SSYNC rounds: every robot computes the smallest enclosing circle (dashed), which never changes; a ringed (activated) robot inside moves out along the ray from the centre, possibly stopping early; the robot at the centre first steps off it.

A first look

Think of dancers who must spread out on one common circle, each at a different spot on it, without anyone drawing the circle first. There are n **robots**, points at distinct places in the plane; their positions form the **configuration**. Each robot repeats a **Look–Compute–Move cycle**: it takes a **snapshot** of where the other robots are (Look), decides on a destination point (Compute) and moves towards it (Move). They are identical, silent, **oblivious** (no memory between cycles), and each uses its own **local coordinate system** (origin, axes, unit of length and handedness). A **scheduler** (an adversary) decides the **activation** of robots, that is, when each performs a cycle; in the basic version it is ASYNC, so each phase may take any finite time, and every move is a **non-rigid move**: the adversary may stop a robot after it has travelled some minimum distance δ. The robots must reach a configuration in which all of them lie on one circle, at distinct points (not necessarily evenly spaced), and stay there.

1. Algorithm: Use the smallest enclosing circle and walk out to it. (Flocchini et al. 2017) (Circle formation, p. 396)

1. Every robot computes the **smallest enclosing circle** (SEC) of all robots. It is unique and does not depend on any local coordinate system, so all robots compute the same circle.

Claude (Anthropic) was used in cataloguing the problems.

2. A robot already on the circle stays where it is.
3. A robot inside the circle moves straight outwards, along the ray from the centre through itself, to the circle.
4. A robot exactly at the centre has no ray; it first steps a little away from the centre.
5. If several robots lie on the same ray, only the outermost one moves out; the others first step sideways onto free rays, so that no two robots ever land on the same point.

Why it works and what it costs. Robots that move from inside onto the circle never change the smallest enclosing circle, so the circle stays fixed while robots move at different speeds and act on outdated snapshots. Each robot inside needs only finitely many moves, since each move covers at least δ or reaches the circle. This is the first phase of the uniform circle algorithm of Flocchini, Prencipe, Santoro and Viglietta, which forms even a regular n-gon (hence a circle) in ASYNC without **chirality** (agreement on the clockwise direction) (Flocchini et al. 2017), the catalogue's current best. Steps 4 and 5 are simplified here; the paper handles robots at the centre and robots on a common ray as separate cases.

Why it matters. The smallest enclosing circle is the standard common reference for disoriented robots, reused in pattern formation, gathering and leader election.

2. Algorithm: Move only inside your own zone, then slide along the circle. (Défago and Souissi 2008) (Circle formation, SSYNC, p. 396)

1. Compute the smallest enclosing circle and use its centre as the origin of polar coordinates.
2. A robot at the centre steps off it, by less than the distance of any other robot from the centre.
3. Otherwise, find your two neighbours in angular order around the centre and draw the rays halfway (by angle) to each of them. The region between these two rays is your **exclusive zone** for this cycle: no other robot will move into it.
4. A robot inside the circle moves outwards within its zone towards the circle; a robot on the circle slides along it towards the angle halfway between its neighbours, without ever leaving an empty arc larger than a half circle (which would shrink the circle).

Why it works and what it costs. Exclusive zones guarantee that two robots never end up at the same point. That matters: two robots on one point get the same snapshot, so a scheduler that always activates them together, with suitable local coordinate systems, can keep them together forever. Défago and Souissi show that in SSYNC the robots reach the circle and then converge towards even spacing without necessarily reaching it (Défago and Souissi 2008); reaching exactly even spacing needed later work (see uniform circle formation).

Why it matters. It introduced the exclusive-zone trick for avoiding inseparable robots and the "slide towards the middle of your neighbours" rule for even spacing.

Simulator

A simulator of Look–Compute–Move robots in the plane that runs in the browser. Each case below starts one of its algorithms from a fixed starting configuration and seed; the run is the simulator's, not a proof. On the website each case runs in the page; here it is a link.

- **Walk out to the smallest enclosing circle.** Each robot moves onto the smallest circle enclosing the robots it sees. Here 24 robots see everyone and run under the simulator's asynchronous scheduler. Open in the simulator.

Claude (Anthropic) was used in cataloguing the problems.

What the community started looking at

Circle formation was first proposed as a way to reach a uniform circle by oblivious robots without a common orientation (Défago and Konagaya 2002); the first algorithms formed a circle and only converged towards uniformity (Défago and Souissi 2008). The question was **whether disoriented, oblivious robots can form a circle at all**, under which scheduler.

Further reading

Papers on this problem that are not yet placed in a setting.

- Maiti et al. (2026a). Conflicting pattern formation: the robots form two teams, and each robot knows its own team but cannot tell which other robots belong to it; the gathering team (at least 6 robots) must meet at one point while the circle team (at least 2 robots) must reach distinct points of one circle. Oblivious SSYNC robots with no agreement on coordinates (not even chirality), non-rigid moves and distinct starting points solve both tasks at once when the gathering team has weak multiplicity detection and the circle team detects a multiplicity only at its own point and knows its own size (Theorem 1, from Lemmas 1–7). The authors attribute the problem, and an ASYNC solution with agreement on the direction of one axis and weak multiplicity detection for all robots, to Bhagat, Flocchini, Mukhopadhyaya and Santoro (ICDCN 2020), not read here. Read (arXiv v1).
- Goswami et al. (2024a). Circle formation on an infinite grid by opaque fat robots (unit disks) with lights of five colours, one-axis agreement and an ASYNC scheduler: from every initial configuration the robots form a circle and terminate (Theorem 3.27). Read (arXiv v1).

Relations to other problems

- **Special case of Pattern formation.** A circle is a family of patterns; forming some circle is easier than forming a given pattern, and it is the precursor of uniform circle formation (Feletti et al. 2026a).
- **Related: Uniform circle formation.** A uniform circle is a circle, so every uniform-circle algorithm forms a circle.

Case tree

The root assumes nothing; going down, each edge adds the assumptions written in grey (+) or leaves an axis unassumed. Bold entries are catalogued settings with their status per measure (Time, Colours) and page; italic entries are branch points, named after the axis on which their branches differ. Axis order: Body → Visibility → Opacity → Faults → Scheduler → Lights (persistent memory) → Agreement on coordinates → Movement → Multiplicity detection → Randomness → Identifiers.

No assumptions

+ point, + unlimited visibility, + transparent, + no faults

Circle formation STANDARD SETTING ●● p. 396

+ SSYNC

Circle formation, SSYNC ●● p. 396

What is known

The best upper bound against the best lower bound in each catalogued setting, in tree order. The settings follow in detail.

Claude (Anthropic) was used in cataloguing the problems.

Setting	Time	Colours	p.
Circle formation STANDARD SETTING	● unknown *finite* vs —	● tight 0 vs —	396
Circle formation, SSYNC	● unknown *finite* vs —	● tight 0 vs —	396

18.2.1 Circle formation

Adds + point, + unlimited visibility, + transparent, + no faults STANDARD SETTING VERIFIED

Robots must place themselves on a common circle, not necessarily evenly spaced; solved in ASYNC without chirality through uniform circle formation.

Assumed: body: point robots; visibility: unlimited; opacity: transparent: every robot sees every robot within range; faults: no faults. **Not assumed:** scheduler: asynchronous (ASYNC); lights (persistent memory): oblivious, no light (OBLOT); agreement on coordinates: no agreement on local coordinate systems; movement: non-rigid: the adversary may stop a move after some minimum distance δ; multiplicity detection: none: a point with several robots looks like one robot; randomness: deterministic; identifiers: anonymous: all robots are identical.

Measure	Lower bound	Best upper bound	Status
Time	none known Time not analysed.	*finite* (Flocchini et al. 2017)	● unknown
Colours	none known No colours.	0 (Flocchini et al. 2017)	● tight

Results.

- Flocchini et al. (2017) [current best]: time *finite*, colours 0. Introductory for ASYNC and current best: Theorem 6.1 forms a uniform circle, hence a circle, in ASYNC without chirality and with non-rigid moves.

Notes. The first circle formation papers are semi-synchronous; see the SSYNC setting. Checked against the full text of (Flocchini et al. 2017).

18.2.2 Circle formation, SSYNC

Adds + SSYNC **to** Circle formation (p. 396) VERIFIED

In SSYNC the first algorithms formed a circle and then converged towards a uniform one.

Assumed: body: point robots; visibility: unlimited; opacity: transparent: every robot sees every robot within range; faults: no faults; scheduler: semi-synchronous (SSYNC). **Not assumed:** lights (persistent memory): oblivious, no light (OBLOT); agreement on coordinates: no agreement on local coordinate systems; movement: non-rigid: the adversary may stop a move after some minimum distance δ; multiplicity detection: none: a point with several robots looks like one robot; randomness: deterministic; identifiers: anonymous: all robots are identical.

Measure	Lower bound	Best upper bound	Status
Time	none known Time not analysed.	*finite* (Défago and Souissi 2008)	● unknown
Colours	none known No colours.	0 (Défago and Souissi 2008)	● tight

Results.

- Défago and Konagaya (2002) [introductory]: time *finite*, colours 0. No chirality needed; proofs only sketched (as stated by Défago and Souissi 2008).

Claude (Anthropic) was used in cataloguing the problems.

- Défago and Souissi (2008) [current best]: time *finite*, colours 0. *Assumes:* chirality; a maximum step length per robot instead of adversarial non-rigid moves. Forms a circle and converges towards a uniform one without reaching it.

Notes. Défago and Souissi call their robots disoriented although they assume chirality; Flocchini et al. use disoriented to exclude chirality. Checked against the full text of (Défago and Souissi 2008); (Défago and Konagaya 2002) was not reachable.

18.3 Uniform circle formation

Robots must form a regular n-gon: a circle with equal spacing.

First posed in 2006 (Dieudonné et al. 2008). The first deterministic solution (SSS 2006); circle formation with convergence towards uniformity predates it (Défago and Konagaya 2002).

Problem statement and measures

n robots start at distinct points and must reach the vertices of a regular n-gon (in any position and size) and stay there. **Time** is in epochs (ASYNC: intervals in which every robot completes at least one full Look–Compute–Move cycle) or rounds; **colours** count light colours in the luminous model.

Measures: **Time**, rounds (FSYNC, SSYNC) or epochs (ASYNC) until the task is done; "finite" when only solvability is known; **Colours**, colours of the light per robot (0 for oblivious robots).



(a) Start: inner robots move out to the fixed circle

(b) Final: a regular hexagon (arrows: moves; dashed circles: starting positions)

Figure 18.3: Uniform circle formation: n oblivious robots at distinct points must occupy the vertices of a regular n-gon, of any position and size, and stay there (Flocchini et al. 2017). Shown in SSYNC rounds (ringed robots are activated), simplified from (Flocchini et al. 2017): the smallest enclosing circle (dashed) never changes; robots inside move out to it, then robots on it move to the vertices of a regular hexagon inscribed in it (hollow).

A first look

Picture guards who must stand evenly spaced around a circular fence that nobody has built: the circle and the spacing must come from the guards themselves. There are n **robots**, points at distinct places in the plane; their positions form the **configuration**. Each robot repeats a **Look–Compute–Move cycle**: it takes a **snapshot** of where the other robots are (Look), decides on a destination point (Compute) and moves towards it (Move). They are identical, silent, **oblivious** (no memory between cycles), and each uses its own **local coordinate system**, so they agree on no direction and no handedness. The

Claude (Anthropic) was used in cataloguing the problems.

scheduler (an adversary) decides the **activation** of robots, that is, when each performs a cycle, and is ASYNC: each phase may take any finite time. Every move is a **non-rigid move**: a robot may be stopped after a minimum distance δ. The robots must reach the **vertices** of a **regular** n**-gon** (a polygon with n equal sides and equal angles), one robot on each vertex, of any size and position, and stay there.

1. Algorithm: Fix the smallest enclosing circle, reach it, then let symmetric robots move as one. (Flocchini et al. 2017) (ASYNC, oblivious, disoriented, p. 401)

1. Compute the **smallest enclosing circle** (SEC) of all robots; every robot computes the same circle. All moves are chosen so that this circle does not change.
2. Robots inside the circle move out to it, as in circle formation.
3. With all robots on the circle, group them into **analogy classes**: two robots are analogous if some symmetry of the whole configuration maps one to the other.
4. The scheduler could make analogous robots act identically anyway, so the algorithm makes each class always move together, deliberately and symmetrically.
5. The robots compute the vertices of a regular n-gon on the circle and move to them class by class. Special configurations that are already close to regular (pre-regular, biangular) get their own procedure.

Why it works and what it costs. The overall plan is simple; the difficulty is that asynchronous robots act on outdated snapshots, so the analogy classes and destinations that one robot computes may already be stale. Flocchini, Prencipe, Santoro and Viglietta show that oblivious, disoriented robots with non-rigid moves form a regular n-gon in ASYNC from every initial configuration at distinct points, for every n (Flocchini et al. 2017); the case $n = 4$ is solved in (Mamino and Viglietta 2016). Time is not analysed; for opaque robots with lights, $O(1)$ epochs and $O(1)$ colours suffice (Feletti et al. 2026a).

Why it matters. It closed a long-standing question and showed that, for pattern formation, asynchrony, chirality and rigid moves do not add computational power.

2. Impossibility: Two robots on the same point can never be separated. (Flocchini et al. 2017) (ASYNC, oblivious, disoriented, p. 401) Suppose two robots start at the same point. They get exactly the same snapshot. The adversary gives them identical local coordinate systems and the scheduler always activates them together. Identical robots with identical snapshots and identical local coordinate systems compute the same destination, so they move together and are still on one point after every cycle. A regular n-gon has n distinct vertices, so it is never formed.

Why it holds. This is why the problem asks for distinct starting points; Flocchini et al. note that with two robots on one point the problem is \"easily seen to be unsolvable\" (Flocchini et al. 2017). The same trap is the reason careful algorithms never let two robots meet while they move.

Why it matters. It is the simplest example of a symmetry that oblivious deterministic robots can never break.

3. Impossibility: Why only the point and the regular polygon can be formed from every initial configuration. (Suzuki and Yamashita 1999) (ASYNC, oblivious, disoriented, p. 401) Start the robots on a regular n-gon. Give every robot a local coordinate system whose x-axis points to the centre, all with the same unit. All robots get the same snapshot; if the scheduler activates all of them together, they all make the same move up to rotation, so the configuration stays a regular n-gon (or collapses to its centre). So from this initial configuration the only formable patterns are the regular n-gon and the single point.

Why it holds. Suzuki and Yamashita showed that the patterns formable from an initial configuration depend on its **symmetricity**, its amount of rotational symmetry (Suzuki and Yamashita 1999); the regular n-gon has the most. Hence the point (gathering) and the regular n-gon (uniform circle) are the

Claude (Anthropic) was used in cataloguing the problems.

only patterns that might be formable from every initial configuration, and both are (Flocchini et al. 2017).

Why it matters. It explains why the uniform circle is singled out among all patterns: it is one of only two universally formable ones.

Simulator

A simulator of Look–Compute–Move robots in the plane that runs in the browser. Each case below starts one of its algorithms from a fixed starting configuration and seed; the run is the simulator's, not a proof. On the website each case runs in the page; here it is a link.

- **Spread evenly on a circle.** The simulator's uniform circle formation algorithm with 12 robots under its asynchronous scheduler. Open in the simulator.

What the community started looking at

Uniform circle formation was first asked as **solvability**: can disoriented oblivious robots form a regular polygon? It was solved for SSYNC (Dieudonné et al. 2008) and then for ASYNC without chirality or rigidity (Flocchini et al. 2017; Mamino and Viglietta 2016). Only after feasibility was settled did the community turn to **time and colours** for luminous opaque robots (Feletti et al. 2023b; Pattanayak and Sharma 2024), reaching $O(1)$ epochs with $O(1)$ colours (Feletti et al. 2026a).

Further reading

Papers on this problem that are not yet placed in a setting.

- Maiti et al. (2026b). Min-max uniform circle formation: every robot is given a circle that encloses all robots, and the robots, starting at distinct points strictly inside it, must occupy the vertices of a regular n-gon inscribed in that circle so that the largest distance travelled by any robot is as small as possible. Without agreement on axes or chirality the robots cannot agree on an optimal assignment of robots to vertices (Theorem 1); the authors also claim impossibility under FSYNC with non-rigid moves without axis agreement even with chirality and lights (Theorem 2, an informal argument that a constant number of colours cannot store the optimal value) and without lights even with a common compass (Theorem 3). For ASYNC robots with non-rigid moves, agreement on one axis and lights of six colours, an algorithm moves each robot along a straight segment to its vertex in an optimal assignment, computed with the centralised algorithms of Bhattacharya et al. and of Chen et al., without collisions and with the optimal largest distance (Theorems 4, 5). The abstract also announces min-max circle formation without equal spacing; the text treats only the uniform case. Read (arXiv v1).
- Biswas et al. (2023). Several uniform circles at once: fat robots with limited visibility, a global coordinate system, memory and messages to avoid collisions, under a synchronous scheduler, compute the positions to occupy and form several uniform circles; the authors note that the method forms any shape whose points the robots can compute. A short paper with a correctness argument and simulations. Read in full.

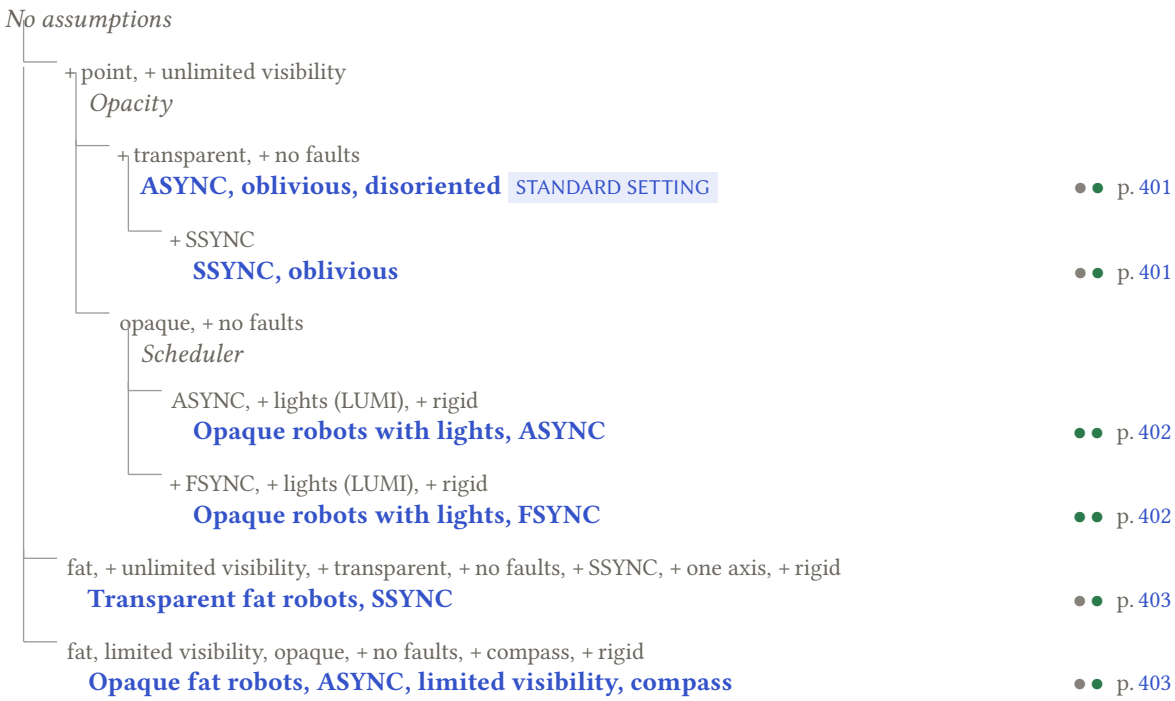
Relations to other problems

- **Special case of Pattern formation.** Uniform circle formation is "an important special case of the geometric pattern formation problem" (Feletti et al. 2026a): the target is the regular n-gon. It is also the special case of circle formation with equal spacing; circle formation is "its precursor" (Feletti et al. 2026a).
- **Related: Circle formation.** A uniform circle is a circle, so every uniform-circle algorithm forms a circle.

Claude (Anthropic) was used in cataloguing the problems.

Case tree

The root assumes nothing; going down, each edge adds the assumptions written in grey (+) or leaves an axis unassumed. Bold entries are catalogued settings with their status per measure (Time, Colours) and page; italic entries are branch points, named after the axis on which their branches differ. Axis order: Body → Visibility → Opacity → Faults → Scheduler → Lights (persistent memory) → Agreement on coordinates → Movement → Multiplicity detection → Randomness → Identifiers.



What is known

The best upper bound against the best lower bound in each catalogued setting, in tree order. The settings follow in detail.

Setting	Time	Colours	p.
ASYNC, oblivious, disoriented STANDARD SETTING	● unknown *finite* vs *impossible*	● tight 0 vs —	401
SSYNC, oblivious	● unknown *finite* vs —	● tight 0 vs —	401
Opaque robots with lights, ASYNC	● tight $O(1)$ vs $\Omega(1)$	● tight $O(1)$ vs $\Omega(1)$	402
Opaque robots with lights, FSYNC	● tight 3 vs —	● tight $O(1)$ vs —	402
Transparent fat robots, SSYNC	● unknown *finite* vs —	● tight 0 vs —	403
Opaque fat robots, ASYNC, limited visibility, compass	● unknown *finite* vs —	● tight 0 vs —	403

Claude (Anthropic) was used in cataloguing the problems.

18.3.1 ASYNC, oblivious, disoriented

Adds + transparent, + no faults STANDARD SETTING VERIFIED

Oblivious, disoriented, non-rigid robots form a regular n-gon from any configuration of distinct points, for every n.

Assumed: body: point robots; visibility: unlimited; opacity: transparent: every robot sees every robot within range; faults: no faults. **Not assumed:** scheduler: asynchronous (ASYNC); lights (persistent memory): oblivious, no light (OBLOT); agreement on coordinates: no agreement on local coordinate systems; movement: non-rigid: the adversary may stop a move after some minimum distance δ; multiplicity detection: none: a point with several robots looks like one robot; randomness: deterministic; identifiers: anonymous: all robots are identical.

Measure	Lower bound	Best upper bound	Status
Time	*impossible* (Flocchini et al. 2017) No time analysis of oblivious ASYNC uniform circle formation is recorded.	*finite* (Flocchini et al. 2017)	● unknown
Colours	none known No colours.	0 (Flocchini et al. 2017)	● tight

Results.

- Flocchini et al. (2017) [lower bound]: time *impossible*. *Bottleneck:* symmetry. *Holds when:* if two robots start at the same point.
- Flocchini et al. (2017) [current best]: time *finite*, colours 0. *Assumes:* $n \neq 4$. Introductory for ASYNC and current best. The case $n = 4$ (square formation) is solved by (Mamino and Viglietta 2016).
- Mamino and Viglietta (2016) [also]: time *finite*, colours 0. *Assumes:* $n = 4$. Completes the picture: synchrony, chirality and rigidity are computationally irrelevant for pattern formation.

Open questions.

Q1. Can oblivious robots without lights form a uniform circle in $O(1)$ epochs, with rigid or non-rigid moves? (Feletti et al. 2026a)

Notes. Uniform circle and point are the only patterns formable from every initial configuration (Flocchini et al. 2017).

18.3.2 SSYNC, oblivious

Adds + SSYNC **to** ASYNC, oblivious, disoriented (p. 401) VERIFIED

The first deterministic uniform circle formation, for $n \geq 5$ semi-synchronous robots.

Assumed: body: point robots; visibility: unlimited; opacity: transparent: every robot sees every robot within range; faults: no faults; scheduler: semi-synchronous (SSYNC). **Not assumed:** lights (persistent memory): oblivious, no light (OBLOT); agreement on coordinates: no agreement on local coordinate systems; movement: non-rigid: the adversary may stop a move after some minimum distance δ; multiplicity detection: none: a point with several robots looks like one robot; randomness: deterministic; identifiers: anonymous: all robots are identical.

Measure	Lower bound	Best upper bound	Status
Time	none known Time not analysed.	*finite* (Dieudonné et al. 2008)	● unknown
Colours	none known No colours.	0 (Dieudonné et al. 2008)	● tight

Claude (Anthropic) was used in cataloguing the problems.

Results.

- Dieudonné et al. (2008) [introductory]: time *finite*, colours 0. *Assumes:* $n \geq 5$. Concentric circles; first deterministic solution.

Notes. Dieudonné and Petit later solved $n \neq 4$ and $n = 4$ in SSYNC (SIROCCO 2007, ISAAC 2008; not yet in the bibliography).

18.3.3 Opaque robots with lights, ASYNC

Adds ASYNC, + lights (LUMI), + rigid VERIFIED

With lights, opaque robots form a uniform circle in $O(1)$ epochs with $O(1)$ colours, which is optimal in both.

Assumed: body: point robots; visibility: unlimited; faults: no faults; lights (persistent memory): full light (LUMI): every robot sees all lights, including its own; movement: rigid: every move reaches its destination. **Not assumed:** opacity: opaque: collinear robots hide each other; scheduler: asynchronous (ASYNC); agreement on coordinates: no agreement on local coordinate systems; multiplicity detection: none: a point with several robots looks like one robot; randomness: deterministic; identifiers: anonymous: all robots are identical.

Measure	Lower bound	Best upper bound	Status
Time	$\Omega(1)$ (Feletti et al. 2026a) $\Theta(1)$ epochs.	$O(1)$ (Feletti et al. 2026a)	• tight
Colours	$\Omega(1)$ (Feletti et al. 2026a) $\Theta(1)$ colours; the exact minimum is open.	$O(1)$ (Feletti et al. 2026a)	• tight

Results.

- Feletti et al. (2026a) [lower bound]: time $\Omega(1)$. *Bottleneck:* trivial. *Holds when:* trivial: at least one epoch.
- Feletti et al. (2026a) [lower bound]: colours $\Omega(1)$. *Bottleneck:* trivial. *Holds when:* trivial: at least one colour.
- Feletti et al. (2023b) [introductory]: time $O(n)$, colours $O(1)$. First ASYNC result for opaque luminous robots (ICTCS 2022; journal 2023); $O(1)$ time in FSYNC and SSYNC.
- Feletti et al. (2023a) [also]: time $O(\log n)$, colours $O(1)$.
- Pattanayak and Sharma (2024) [also]: time $O(x)$, colours $O(n^{1/2^x})$. A time–colour trade-off for $x \in [1, O(\log \log n)]$: $O(1)$ time with $O(\sqrt{n})$ colours, or $O(\log \log n)$ time with $O(1)$ colours. Abstract only.
- Feletti et al. (2026a) [current best]: time $O(1)$, colours $O(1)$. Asymptotically optimal in time and colours.

Open questions.

Q1. What is the exact minimum number of colours for $O(1)$-epoch uniform circle formation in ASYNC? (Feletti et al. 2026a)

Q2. Uniform circle formation with non-rigid moves in the luminous opaque model (Feletti et al. 2026a).

Notes. Robots are punctiform, disoriented and do not know n; collisions are avoided. The only lower bounds are trivial. Checked against the full texts of (Feletti et al. 2023a) and (Feletti et al. 2026a).

18.3.4 Opaque robots with lights, FSYNC

Adds + FSYNC, + lights (LUMI), + rigid VERIFIED

In FSYNC, opaque luminous robots form a uniform circle in 3 cycles with $O(1)$ colours.

Claude (Anthropic) was used in cataloguing the problems.

Assumed: body: point robots; visibility: unlimited; faults: no faults; scheduler: fully synchronous (FSYNC); lights (persistent memory): full light (LUMI): every robot sees all lights, including its own; movement: rigid: every move reaches its destination. **Not assumed:** opacity: opaque: collinear robots hide each other; agreement on coordinates: no agreement on local coordinate systems; multiplicity detection: none: a point with several robots looks like one robot; randomness: deterministic; identifiers: anonymous: all robots are identical.

Measure	Lower bound	Best upper bound	Status
Time	none known Constant.	3 (Feletti et al. 2023b)	• tight
Colours	none known $\Theta(1)$.	$O(1)$ (Feletti et al. 2023b)	• tight

Results.

- Feletti et al. (2018) [introductory]: time $O(1)$, colours $O(1)$. Uniform circle formation for opaque luminous robots in FSYNC, without common coordinates or chirality: a Complete Visibility phase, then five cycles (the last repeated at most twice), each with $O(1)$ colours (Section 4 and conclusion).
- Feletti et al. (2023b) [current best]: time 3, colours $O(1)$. Three cycles.

Notes. In SSYNC the same paper takes $O(1)$ rounds. The ICTCS 2022 version was read in full.

18.3.5 Transparent fat robots, SSYNC

Adds fat, + unlimited visibility, + transparent, + no faults, + SSYNC, + one axis, + rigid VERIFIED

Transparent fat robots that agree on one axis form a regular n-gon of given side in SSYNC.

Assumed: visibility: unlimited; opacity: transparent: every robot sees every robot within range; faults: no faults; scheduler: semi-synchronous (SSYNC); agreement on coordinates: agreement on one axis (no common chirality); movement: rigid: every move reaches its destination. **Not assumed:** body: fat robots (unit discs); lights (persistent memory): oblivious, no light (OBLOT); multiplicity detection: none: a point with several robots looks like one robot; randomness: deterministic; identifiers: anonymous: all robots are identical.

Measure	Lower bound	Best upper bound	Status
Time	none known —	*finite* (Mondal and Chaudhuri 2017) [preprint]	• unknown
Colours	none known No colours.	0 (Mondal and Chaudhuri 2017) [preprint]	• tight

Results.

- Mondal and Chaudhuri (2017) [preprint] [current best]: time *finite*, colours 0. *Assumes:* n and the side length are known; only the direction of the y axis is shared, without chirality. Introductory and current best (Theorem 1). A preprint.

Notes. The paper assumes agreement on one axis without chirality, which is the catalogue's one-axis value. Checked against the full text.

18.3.6 Opaque fat robots, ASYNC, limited visibility, compass

Adds fat, limited visibility, opaque, + no faults, + compass, + rigid VERIFIED

Opaque fat robots with limited visibility and a common coordinate system form a uniform circle of given

Claude (Anthropic) was used in cataloguing the problems.

centre and radius in ASYNC.

Assumed: faults: no faults; agreement on coordinates: agreement on both axes (common compass); movement: rigid: every move reaches its destination. **Not assumed:** body: fat robots (unit discs); visibility: limited: robots see only within a fixed radius; opacity: opaque: collinear robots hide each other; scheduler: asynchronous (ASYNC); lights (persistent memory): oblivious, no light (OBLOT); multiplicity detection: none: a point with several robots looks like one robot; randomness: deterministic; identifiers: anonymous: all robots are identical.

Measure	Lower bound	Best upper bound	Status
Time	none known —	*finite* (Mondal et al. 2020) [preprint]	● unknown
Colours	none known No colours.	0 (Mondal et al. 2020) [preprint]	● tight

Results.

- Mondal et al. (2020) [preprint] [current best]: time *finite*, colours 0. *Assumes:* common origin and axes; the circle's centre and radius are given. Introductory and current best (Theorem 1). A preprint.

Notes. Checked against the full text.

PLANE



18.4 Universal pattern formation

Robots must be able to form any pattern from any initial configuration, not only the ones symmetry allows.

First posed in 2025 (Flocchini et al. 2026). Introduced at SIROCCO 2025.

Problem statement and measures

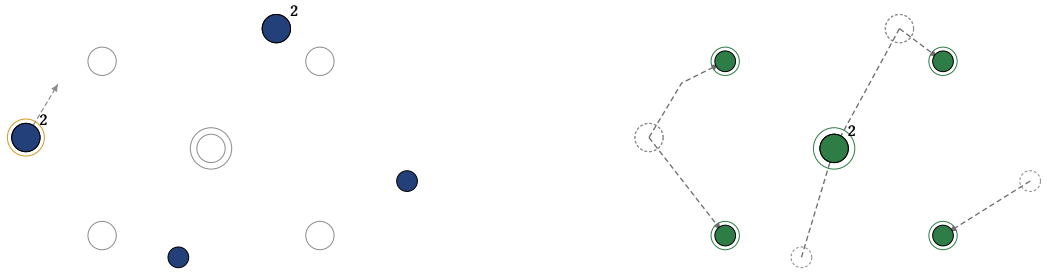
From every initial configuration of n robots (several robots may start on the same point, forming a multiplicity Flocchini et al. 2026), the robots must form every given **pattern** of n points (a multiset of points given to every robot; multiplicities allowed). With lights, universal dancing asks for any sequence of patterns.

Measures: **Time**, rounds (FSYNC, SSYNC) or epochs (ASYNC) until the task is done; "finite" when only solvability is known; **Colours**, colours of the light per robot (0 for oblivious robots).

A first look

Imagine a troupe that must be able to form every shape the director might ask for, from wherever its members happen to stand, even if some of them are standing on top of each other. There are n **robots**, points in the plane; their positions form the **configuration**, and a point that holds two or more robots is a **multiplicity**. Each robot repeats a **Look–Compute–Move cycle**: it takes a **snapshot** of where the other robots are (Look), decides on a destination point (Compute) and moves towards it (Move). They are identical, silent and **oblivious** (no memory between cycles), and each uses its own **local coordinate system**, so they agree on no direction. The **scheduler**, an adversary, decides the **activation** of robots, that is, when each performs a cycle; here it is **sequential**: time runs in **rounds**, in each round it activates exactly one robot, and every robot is activated again and again. Universal pattern formation asks for one algorithm that, from every initial configuration, forms every given **pattern** of n points (a list of points that every robot is given, to be matched up to position, rotation and size; it may contain multiplicities) and then stops.

Claude (Anthropic) was used in cataloguing the problems.



(a) Start: 6 robots on 4 points; pattern has 5

(b) Final: the pattern, with its multiplicity (arrows: moves; dashed circles: starting positions)

Figure 18.4: Universal pattern formation: from every configuration of n robots, possibly with several robots on one point (a multiplicity), the robots must be able to form every given pattern of n points, including patterns with multiplicities (Flocchini et al. 2026). Shown, simplified, the plan of (Flocchini et al. 2026) under the sequential scheduler (one ringed robot is activated per round): a robot leaves a multiplicity until the robots occupy as many points as the pattern (hollow), then the points of the pattern are filled one by one; the last robot creates the pattern's multiplicity.

1. Algorithm: One robot at a time: create enough occupied points, agree on a map, fill it in. (Flocchini et al. 2026) (Sequential schedulers, p. 406)

1. **Initialization.** If the robots occupy fewer distinct points than the pattern has, robots leave multiplicities one at a time until there are enough occupied points.
2. Scale the pattern so that its **smallest enclosing circle** (the smallest circle containing all its points) matches the smallest enclosing circle of the configuration.
3. **Leader configuration.** Move robots so that all of them agree on how the pattern is laid over the robots' positions (a common map from robots to pattern points).
4. **Partial formation.** Fill the empty points of the pattern one by one in an agreed order, moving along detours that never pass over another robot.
5. **Finalization.** When only one robot is off the pattern, the last robots move to their places.

Why it works and what it costs. Under a sequential scheduler only one robot moves at a time, so the moving robot is automatically different from all others; in particular, a multiplicity can be split, since only one of its robots moves. Flocchini, Navarra, Pattanayak, Piselli and Santoro show that this gives oblivious robots universal pattern formation with no extra assumption, except for the single point (gathering), which needs **weak multiplicity detection** (a robot sees whether a point holds more than one robot), necessary and sufficient (Flocchini et al. 2026). The steps above are their high-level plan; the actual algorithm handles many special cases, and patterns with fewer than 5 points separately. Time is not analysed.

Why it matters. It shows that sequential activation is more than a way to elect a leader: it gives power that even a leader with full synchrony does not have.

2. Impossibility: Fully synchronous robots cannot split a multiplicity. (Flocchini et al. 2026) (Sequential schedulers, p. 406) Take three robots: two on one point, the third alone elsewhere. Under FSYNC every robot is activated in every round. The two robots on the multiplicity get the same snapshot; even if they have identical local coordinate systems, move rigidly, can count robots on each point, and the third robot is a recognisable leader, the two compute the same destination and move together. They stay on one point forever, so no pattern with three distinct points is ever formed.

Claude (Anthropic) was used in cataloguing the problems.

Why it holds. This is Theorem 1 of (Flocchini et al. 2026): universal pattern formation is unsolvable under FSYNC for $n \geq 3$, even with all these extra capabilities, and hence also under SSYNC and ASYNC, where the scheduler may choose to activate every robot. Gathering alone, in contrast, is solvable under FSYNC with no extra capability.

Why it matters. It shows that the fully synchronous and the sequential schedulers are incomparable: each solves something the other cannot.

What the community started looking at

Universal pattern formation asked whether a **scheduler alone** can break the symmetry that blocks pattern formation: under sequential schedulers it can, except for a single point (Flocchini et al. 2026); with lights, robots can then perform any sequence of patterns (Feletti et al. 2026c).

Relations to other problems

- **Strengthening of Pattern formation.** Pattern formation asks which patterns are formable from a given start; universal pattern formation asks for all patterns from all starts, which symmetry forbids under SSYNC and FSYNC but sequential schedulers allow (Flocchini et al. 2026).

Case tree

The root assumes nothing; going down, each edge adds the assumptions written in grey (+) or leaves an axis unassumed. Bold entries are catalogued settings with their status per measure (Time, Colours) and page; italic entries are branch points, named after the axis on which their branches differ. Axis order: Body → Visibility → Opacity → Faults → Scheduler → Lights (persistent memory) → Agreement on coordinates → Movement → Multiplicity detection → Randomness → Identifiers.

No assumptions

+ point, + unlimited visibility, + transparent, + no faults, + sequential

Sequential schedulers STANDARD SETTING ●● p. 406

What is known

The best upper bound against the best lower bound in each catalogued setting, in tree order. The settings follow in detail.

Setting	Time	Colours	p.
Sequential schedulers STANDARD SETTING	● unknown *finite* vs *impossible*	● tight 0 vs —	406

18.4.1 Sequential schedulers

Adds + point, + unlimited visibility, + transparent, + no faults, + sequential STANDARD SETTING VERIFIED

Under any sequential scheduler, oblivious robots can form every pattern from every starting configuration, except a single point, which needs weak multiplicity detection.

Assumed: body: point robots; visibility: unlimited; opacity: transparent: every robot sees every robot within range; faults: no faults; scheduler: sequential: exactly one robot is activated per round. **Not assumed:** lights (persistent memory): oblivious,

Claude (Anthropic) was used in cataloguing the problems.

no light (OBLOT); agreement on coordinates: no agreement on local coordinate systems; movement: non-rigid: the adversary may stop a move after some minimum distance δ; multiplicity detection: none: a point with several robots looks like one robot; randomness: deterministic; identifiers: anonymous: all robots are identical.

Measure	Lower bound	Best upper bound	Status
Time	*impossible* (Flocchini et al. 2026) Time not analysed.	*finite* (Flocchini et al. 2026)	● unknown
Colours	none known No colours.	0 (Flocchini et al. 2026)	● tight

Results.

- Flocchini et al. (2026) [lower bound]: time *impossible*. *Bottleneck:* symmetry. *Holds when:* under FSYNC, even with strong extra capabilities.
- Flocchini et al. (2026) [current best]: time *finite*, colours 0. Introductory and current best; no additional assumptions except for point formation.

18.5 Pattern sequences

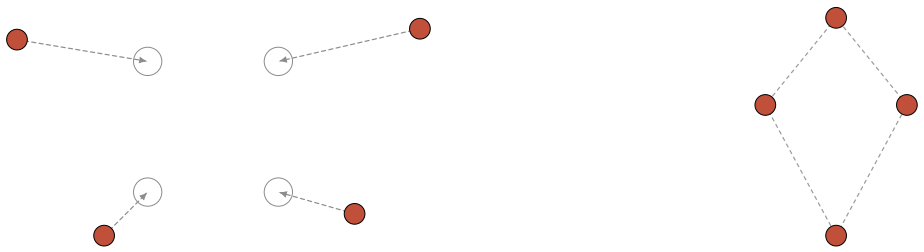
Robots must form a given sequence of patterns, one after the other, forever.

First posed in 2010 (Das et al. 2015b). Introduced at PODC 2010.

Problem statement and measures

Robots are given a sequence of **patterns** $P_1, P_2, \ldots, P_m$ (sets of points in the plane) and must form them in order, repeatedly, with all robots taking part in each.

Measures: **Time**, rounds (FSYNC, SSYNC) or epochs (ASYNC) until the task is done; "finite" when only solvability is known; **Colours**, colours of the light per robot (0 for oblivious robots).



(a) Start: lights show step 1 (square)

(b) Lights change to step 1: the cycle repeats

Figure 18.5: Pattern sequences: robots must form the patterns $P_1, P_2, \ldots, P_m$ one after another, all robots taking part in each, and repeat the sequence forever (Das et al. 2015b). Shown in FSYNC rounds: the sequence square, line, square, kite, which oblivious robots cannot form because the two squares look alike; with lights, the colour records the step, so the second square differs from the first by its colour.

Claude (Anthropic) was used in cataloguing the problems.

A first look

Picture a marching band that must form a sequence of shapes, for instance a square, then a line, then a triangle, then the square again, over and over. There are n **robots**, points in the plane; their positions form the **configuration**. Each robot repeats a **Look–Compute–Move cycle**: it takes a **snapshot** of where the other robots are (Look), decides on a destination point (Compute) and moves towards it (Move). In the basic version they are **oblivious** (no memory between cycles), agree on what clockwise means (**chirality**) but on nothing else, every move is a **rigid move** (it always reaches the computed destination), and the **scheduler**, an adversary, is SSYNC: time runs in **rounds**, and in each round it decides the **activation** of some of the robots, each of which performs one cycle. Given **patterns** $P_1, P_2, \ldots, P_m$ (m is the number of patterns; a pattern is a set of points, formed when the configuration has its shape), the robots must form P_1, then P_2, and so on up to P_m, then P_1 again, forever, with every robot taking part in each pattern.

1. Impossibility: Oblivious robots cannot repeat a pattern. (Das et al. 2015b) (Oblivious, SSYNC, chirality, p. 410) Take $n = 4$ and the sequence square, line, square, kite. When the robots stand in the square they must decide where to go next: to the line (if this is the first square) or to the kite (if it is the second). But oblivious robots use only the current snapshot, and the two squares look exactly alike to robots that see only shapes, not sizes or directions. With the same snapshot a robot computes the same destination, so it cannot follow both continuations. A second obstacle is symmetry: robots can form a pattern only if it has at least the symmetry of the configuration they start from. In a cycle $P_1 \to P_2 \to \ldots \to P_m \to P_1$ each pattern must have at least the symmetry of the one before, and going round the cycle this forces all of them to have the same symmetry.

Why it holds. Das, Flocchini, Santoro and Yamashita characterised the sequences formable by oblivious robots under these assumptions (Das et al. 2015b): as summarised in (Das et al. 2020), all patterns must have the same number of points and no pattern may appear twice, and the catalogue records that the patterns must have equal **symmetricity** (the largest q such that the points split into regular q-gons centred at the centre of their smallest enclosing circle, a point at that centre counting as a 1-gon Das et al. 2020), a multiple of that of the initial configuration. With distinct identifiers, any sequence of patterns of size at most n can be formed.

Why it matters. It isolates what memory buys: forming one pattern needs none, but following a script needs a way to know where you are in it.

2. Algorithm: Keep your place in the sequence with a coloured light. (Das et al. 2020) (Luminous robots, ASYNC, non-rigid, chirality, p. 410) Give each robot a **light**: a lamp that shows one of c colours, set by its robot, visible to all robots and kept between cycles.

1. Use the colours shown by the robots to record which step of the sequence is being formed.
2. Form the current pattern with a single-pattern algorithm.
3. When the pattern is complete, the robots change their colours to record the next step, and start forming the next pattern.
4. A repeated pattern is no longer a problem: the positions are the same, but the colours differ.

Why it works and what it costs. Lights act as a small shared memory, so repetitions, and even patterns with fewer points than robots, become possible. Das, Flocchini, Prencipe and Santoro give one protocol that forms every formable sequence, in ASYNC and with non-rigid moves (Das et al. 2020). The number of colours is bounded but not by a constant: it must grow with the number of repetitions and contractions in the sequence, by a lower bound that holds even for fully synchronous robots with rigid moves

Claude (Anthropic) was used in cataloguing the problems.

(Theorem 18 of Das et al. 2020). The steps above are the basic idea, not the paper's protocol, which must also cope with robots acting on outdated snapshots and with symmetric initial configurations.

Why it matters. It measures, in colours, exactly how much memory a choreography needs.

What the community started looking at

Forming a sequence of patterns asked how much **memory the configuration itself can hold** for oblivious robots: the answer is a symmetricity condition (Das et al. 2015b); lights remove the restriction and work asynchronously (Das et al. 2020).

Further reading

Papers on this problem that are not yet placed in a setting.

- Das et al. (2019d). Oblivious permutations: robots must permute their positions, either every robot moving to another initial position (Move-All) or every robot visiting all initial positions periodically (Visit-All). Even in FSYNC with rigid moves some configurations make them unsolvable; a characterisation for one-step algorithms with chirality (Theorem 3) and impossibilities for several steps (Theorems 4, 5). Read (arXiv v1).

Relations to other problems

- **Generalisation of Pattern formation.** Forming one pattern is the case $m = 1$; a sequence needs the robots to remember (or encode in the configuration) which pattern comes next.

Case tree

The root assumes nothing; going down, each edge adds the assumptions written in grey (+) or leaves an axis unassumed. Bold entries are catalogued settings with their status per measure (Time, Colours) and page; italic entries are branch points, named after the axis on which their branches differ. Axis order: Body → Visibility → Opacity → Faults → Scheduler → Lights (persistent memory) → Agreement on coordinates → Movement → Multiplicity detection → Randomness → Identifiers.

No assumptions

+ point, + unlimited visibility, + transparent, + no faults

Scheduler

+ SSYNC, + chirality, + rigid

Oblivious, SSYNC, chirality STANDARD SETTING ● ● p. 410

ASYNC, + lights (LUMI), + chirality

Luminous robots, ASYNC, non-rigid, chirality ● ● p. 410

+ sequential, + lights (LUMI), + strong multiplicity

Luminous robots, sequential scheduler (universal dancing) ● ● p. 411

What is known

The best upper bound against the best lower bound in each catalogued setting, in tree order. The settings follow in detail.

Claude (Anthropic) was used in cataloguing the problems.

Setting	Time	Colours	p.
Oblivious, SSYNC, chirality STANDARD SETTING	● unknown *finite* vs —	● tight 0 vs —	410
Luminous robots, ASYNC, non-rigid, chirality	● unknown *finite* vs *impossible*	● polynomial gap $\alpha(S)+\mu(S)+O(1)$ vs $\max\{\alpha(S)/\min_i \alpha(S_i), \mu(S)^{1/\alpha(S)}\}$	410
Luminous robots, sequential scheduler (universal dancing)	● unknown *finite* vs —	● near-optimal $c \geq 4$ vs with c colours,length $\leq \binom{n+c-1}{c-1}$	411

18.5.1 Oblivious, SSYNC, chirality

Adds + SSYNC, + chirality, + rigid STANDARD SETTING VERIFIED

A cyclic sequence of patterns is formable exactly when all patterns have the same symmetricity, a multiple of the start's.

Assumed: body: point robots; visibility: unlimited; opacity: transparent: every robot sees every robot within range; faults: no faults; scheduler: semi-synchronous (SSYNC); agreement on coordinates: common chirality (same clockwise orientation); movement: rigid: every move reaches its destination. **Not assumed:** lights (persistent memory): oblivious, no light (OBLOT); multiplicity detection: none: a point with several robots looks like one robot; randomness: deterministic; identifiers: anonymous: all robots are identical.

Measure	Lower bound	Best upper bound	Status
Time	none known —	*finite* (Das et al. 2015b)	● unknown
Colours	none known No colours.	0 (Das et al. 2015b)	● tight

Results.

- Das et al. (2015b) [current best]: time *finite*, colours 0. Introductory and current best; with distinct identifiers, any sequence of patterns of size at most n.

Open questions.

Q1. Form a sequence of patterns in ASYNC with oblivious robots (Das et al. 2015b).

18.5.2 Luminous robots, ASYNC, non-rigid, chirality

Adds ASYNC, + lights (LUMI), + chirality VERIFIED

With lights and common chirality, the formable pattern sequences are characterised under ASYNC and non-rigid moves; the number of colours grows with the sequence and is not known to be optimal.

Assumed: body: point robots; visibility: unlimited; opacity: transparent: every robot sees every robot within range; faults: no faults; lights (persistent memory): full light (LUMI): every robot sees all lights, including its own; agreement on coordinates: common chirality (same clockwise orientation). **Not assumed:** scheduler: asynchronous (ASYNC); movement: non-rigid: the adversary may stop a move after some minimum distance δ; multiplicity detection: none: a point with several robots looks like one robot; randomness: deterministic; identifiers: anonymous: all robots are identical.

Claude (Anthropic) was used in cataloguing the problems.

Measure	Lower bound	Best upper bound	Status
Time	*impossible* (Das et al. 2020) Only solvability.	*finite* (Das et al. 2020)	● unknown
Colours	$\max\{\alpha(S)/\min_i \alpha(S_i),\ \mu(S)^{1/\alpha(S)}\}$ (Das et al. 2020) $\alpha(S)+\mu(S)+O(1)$ against the Theorem 18 bound; closing it is called an important open question.	$\alpha(S)+\mu(S)+O(1)$ (Das et al. 2020)	● polynomial gap

Results.

- Das et al. (2020) [lower bound]: colours $\max\{\alpha(S)/\min_i \alpha(S_i),\ \mu(S)^{1/\alpha(S)}\}$. *Bottleneck:* memory. *Holds when:* Theorem 18; holds even in FSYNC with rigid moves.
- Das et al. (2020) [lower bound]: time *impossible*. *Bottleneck:* symmetry breaking. *Holds when:* Theorem 2: sequences that violate the paper's symmetry conditions.
- Das et al. (2020) [current best]: time *finite*, colours $\alpha(S)+\mu(S)+O(1)$. *Assumes:* weak chromatic multiplicity detection. Introductory and current best: a universal protocol for every formable sequence (Theorem 17). The number of colours is "bounded but not by a constant".

Open questions.

Q1. Close the gap between the colours used and the lower bound of Theorem 18 (Das et al. 2020).

Q2. Pattern sequences with lights without chirality, with faults, or self-stabilising (Das et al. 2020).

Notes. $\alpha(S)$ and $\mu(S)$ are parameters of the sequence $S = \langle S_1, \dots \rangle$ defined in (Das et al. 2020) (equation 1 and Theorem 18). Checked against the full text (HAL copy of the IEEE Access version).

18.5.3 Luminous robots, sequential scheduler (universal dancing)

Adds + sequential, + lights (LUMI), + strong multiplicity VERIFIED

Under a sequential scheduler, disoriented luminous robots form every sequence of patterns up to a length fixed by the number of colours.

Assumed: body: point robots; visibility: unlimited; opacity: transparent: every robot sees every robot within range; faults: no faults; scheduler: sequential: exactly one robot is activated per round; lights (persistent memory): full light (LUMI): every robot sees all lights, including its own; multiplicity detection: strong: a robot sees how many robots are at each point. **Not assumed:** agreement on coordinates: no agreement on local coordinate systems; movement: non-rigid: the adversary may stop a move after some minimum distance δ; randomness: deterministic; identifiers: anonymous: all robots are identical.

Measure	Lower bound	Best upper bound	Status
Time	none known —	*finite* (Feletti et al. 2026c)	● unknown
Colours	with c colours,length $\leq \binom{n+c-1}{c-1}$ (Feletti et al. 2026c) Upper and lower bounds on the sequence length differ by a shift in n and c.	$c \geq 4$ (Feletti et al. 2026c)	● near-optimal

Results.

- Feletti et al. (2026c) [lower bound]: colours with c colours,length $\leq \binom{n+c-1}{c-1}$. *Bottleneck:* memory. *Holds when:* longer sequences cannot be formed with c colours.
- Feletti et al. (2026c) [current best]: time *finite*, colours $c \geq 4$. Introductory and current best: with $c \geq 4$ colours every sequence of length up to $\binom{n+c-7}{c-4}$ is formed; non-rigid moves, disoriented robots.

Notes. Checked against the full text.

Claude (Anthropic) was used in cataloguing the problems.

18.6 Mutual visibility

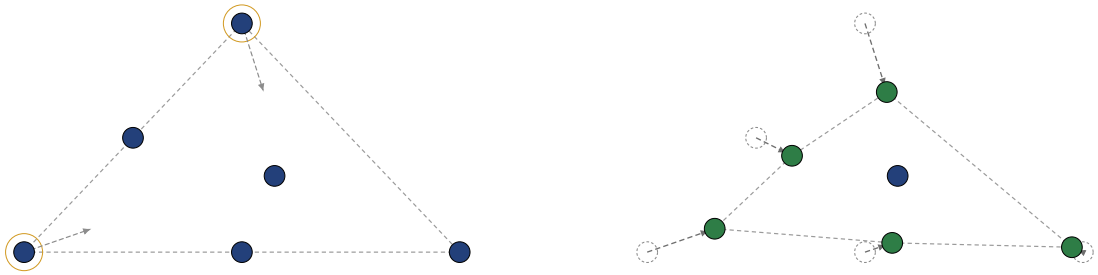
Opaque robots must reach positions where no three are collinear, so that every robot sees every other.

First posed in 2014 (Di Luna et al. 2017). Introduced at CCCG 2014 for oblivious robots and at SSS 2014 for robots with lights (Di Luna et al. 2014); journal version (Di Luna et al. 2017).

Problem statement and measures

n **opaque** point robots (or unit discs) start at distinct points: a robot hides the robots behind it on the same line of sight. They must reach a configuration in which every robot sees every other (no three collinear), without collisions.

Measures: **Time**, rounds (FSYNC, SSYNC) or epochs (ASYNC) until the task is done; "finite" when only solvability is known; **Colours**, colours of the light per robot (0 for oblivious robots).



(a) Start: hidden robots on hull edges and inside

(b) Final: all see each other; the robots stop (arrows: moves; dashed circles: starting positions)

Figure 18.6: Mutual visibility: opaque robots at distinct points, where a robot hides those behind it, must reach a configuration with no three robots collinear, so that every robot sees every other, without collisions. Shown: the Shrink idea of (Di Luna et al. 2017) in SSYNC rounds with rigid moves: a ringed (activated) robot that is at a corner of the convex hull (dashed) of the robots it sees moves a little into the triangle with its two hull neighbours, staying at a corner, and sets its light to Vertex (green); hidden robots become corner robots in turn.

A first look

Picture people at a party who want every pair of them to be able to see each other, but bodies block the view: you cannot see someone directly behind another person. There are n **robots**, points at distinct places in the plane; their positions form the **configuration**. The robots are **opaque**: robot r sees robot s only if no third robot lies on the straight segment between them. Each robot repeats a **Look–Compute–Move cycle**: it takes a **snapshot** of the robots it sees (Look), decides on a destination point (Compute) and moves towards it (Move). Each robot carries a **light** showing one of c colours, which all robots can see and which persists between cycles; otherwise robots remember nothing. In the basic version the **scheduler**, an adversary, is SSYNC: time runs in **rounds**, and in each round it decides the **activation** of some robots, each of which performs one cycle; every move is a **rigid move** (a robot always reaches its destination). The robots must reach, without two of them ever colliding, a configuration in which no three are on a line, so all see each other, and then stop.

1. Algorithm: Shrink: only corner robots move, cutting corners of the convex hull. (Di Luna et al. 2017) (Lights, SSYNC, rigid, p. 417) The **convex hull** is the shape of a rubber band stretched around all

Claude (Anthropic) was used in cataloguing the problems.

robots; its corner points are its **corners**, and a robot at one of them is a corner robot. Lights have two colours, Off and Vertex (meaning: I am at a corner), all starting Off.

1. A robot that is not at a corner of the convex hull of the robots it sees does nothing.
2. A corner robot sets its light to Vertex, finds its two neighbours a and b along the hull, and moves a little into the triangle formed by itself, a and b, cutting off its corner.
3. The move is kept inside a smaller safe triangle, so that robots moving at the same time cannot collide, and the robot stops on the nearest line parallel to ab through another robot, so that it stays at a corner.
4. A robot that sees only robots with light Vertex knows that it sees everybody, and stops.

Why it works and what it costs. Each cut shrinks the hull and can turn a robot that was hidden on an edge or inside into a new corner robot; eventually all robots are at the corners of a convex polygon, and then nobody blocks anybody. Some configurations need an inside robot to move: in a centrally symmetric configuration with a robot at the centre, symmetric robots can be made to move symmetrically forever, and the central robot keeps hiding pairs; the paper handles such cases separately. Di Luna, Flocchini, Gan Chaudhuri, Poloni, Santoro and Viglietta prove that this solves the problem in rigid SSYNC with 2 colours, which is optimal, and with no lights if n is known (Di Luna et al. 2017). In ASYNC, $O(1)$ epochs (intervals in which every robot completes a cycle) are achievable with more colours (Sharma et al. 2021a).

Why it matters. It introduced the luminous-robot approach to obstructed visibility, and its output (all robots on a convex polygon) is a starting point for other tasks on opaque robots.

2. Lower bound: Without lights, a robot cannot know when to stop. (Di Luna et al. 2017) (Lights, SSYNC, rigid, p. 417) Suppose robots have no lights (one colour) and do not know n. A robot sees, say, four other robots at the vertices of a square. Maybe that is everybody, and it should stop. Or maybe a fifth robot hides exactly behind one of the four, on the same line of sight, and the robots must still move. The two situations give the robot the same snapshot, and stopping cannot be undone, so whatever it decides is wrong in one of them.

Why it holds. Hence at least 2 colours are needed when n is unknown and robots must stop (Section 7 of Di Luna et al. 2017), and Shrink shows that 2 suffice in rigid SSYNC. If n is known, a robot can count the robots it sees and stop when it sees $n - 1$ others, so no light is needed. The authors note that if robots only need to stand still forever, without explicitly stopping, this argument does not apply.

Why it matters. It shows that the cost of detecting completion, not of reaching it, is what forces memory.

What the community started looking at

Mutual visibility arose from **obstructed visibility**: before opaque robots can coordinate they must see each other. The first questions were feasibility and the **number of colours** (Di Luna et al. 2017; Sharma et al. 2015b); then **time** with $O(1)$ colours (Vaidyanathan et al. 2015; Sharma et al. 2021a), and fat robots (Sharma et al. 2018; Bhagat et al. 2024).

Further reading

Papers on this problem that are not yet placed in a setting.

- Cicerone et al. (2023c). Geodesic mutual visibility on graphs: robots on the nodes of a grid must reach nodes such that every two are joined by a shortest path through no other robot. Time-optimal algorithms for oblivious

Claude (Anthropic) was used in cataloguing the problems.

synchronous robots with chirality on finite and infinite square grids, within the minimum area (Theorem 1). Read (arXiv v1). Also in the journal paper (Cicerone et al. 2023b).

- Badri et al. (2024). Geodesic mutual visibility on finite hexagonal grids: the largest mutual-visibility sets (Theorem 1) and an optimal algorithm for synchronous oblivious robots with chirality (Theorem 2). Read (arXiv v2).
- Cicerone et al. (2025a). The graph-theoretic side: mutual-visibility and total mutual-visibility numbers of hypercubes, Fibonacci cubes (journal version only), cube-connected cycles and butterflies: approximation algorithms for the first three, and exact formulae $2^{d+1} - 2$ and 2^d for the butterfly of dimension d. Read (arXiv v1 and the journal version).
- Kim et al. (2023). Robots on the infinite grid with lights of two colours under ASYNC reach complete visibility in $O(\Delta_0 + n)$ rounds within $O(n^2)$ area (Theorem 1, proof sketched). Read (arXiv v1). The journal version (Kim et al. 2025) proves the lemmas in full but states $O(\Delta_0 + n^2)$ rounds.
- Alsaedi et al. (2023b). A centralised motion-planning relative: move n robots inside a simple polygon along shortest paths to their targets while keeping them mutually visible; $O(mn)$ time when starts and targets lie on two disjoint segments. Read (arXiv v2).
- Dolev et al. (2019). A synchronisation tool for Look–Compute–Move robots: a self-stabilizing implementation of neighbourhood mutual remainder, used to make moves atomic and to simulate FSYNC with clocks of equal speed. Read (arXiv v1).
- Bhagat et al. (2016b). ASYNC opaque robots that agree on one axis reach general position (no three collinear). Abstract read.
- Sharma et al. (2015a). Runtime of Di Luna et al.'s algorithms with lights in FSYNC: Contain takes $\Theta(n)$ rounds and Shrink $\Omega(n^2)$; Modified Shrink takes $O(n \log n)$ and needs $\Omega(n)$. Abstract read.
- Sharma et al. (2021b). Randomized complete visibility on the infinite grid by ASYNC robots with lights and obstructed visibility, with optimal time. Abstract read.
- Kim et al. (2025). Journal version of (Kim et al. 2023): ASYNC oblivious robots on the infinite grid that agree on the direction and orientation of both axes, with lights of two colours and without knowing n, reach complete visibility from any configuration (Theorem 2, proved through Lemmas 1–13) in $O(\Delta_0 + n^2)$ asynchronous rounds (their D is Δ_0 here) within the optimal area $O(n^2)$. Read in full.
- Cicerone et al. (2023b). Poses geodesic mutual visibility: robots on distinct nodes of a graph must reach, without collisions, nodes such that every two are joined by a shortest path through no other robot. On finite grids, oblivious FSYNC robots with chirality solve it from asymmetric configurations within the smallest bounding rectangle by a time-optimal algorithm (Theorem 1), with an informal extension to infinite grids. On trees, oblivious SSYNC robots, at most one per leaf, solve it from configurations without critical nodes, where two robots would have to cross one node (Theorems 2, 3), in $O(nD)$ epochs against $\Omega(n + D)$, D the diameter of the tree (Lemmas 11, 12). Read in full.

Relations to other problems

- **Special case of Pattern formation.** Mutual visibility asks for any pattern in general position, a family of patterns; it is the usual first step of pattern formation for opaque robots.

Claude (Anthropic) was used in cataloguing the problems.

Case tree

The root assumes nothing; going down, each edge adds the assumptions written in grey (+) or leaves an axis unassumed. Bold entries are catalogued settings with their status per measure (Time, Colours) and page; italic entries are branch points, named after the axis on which their branches differ. Axis order: Body → Visibility → Opacity → Faults → Scheduler → Lights (persistent memory) → Agreement on coordinates → Movement → Multiplicity detection → Randomness → Identifiers.

No assumptions

- + point, + unlimited visibility
  Faults
  - + no faults
    Scheduler
    - + SSYNC
      Lights (persistent memory)
      - + lights (LUMI)
        Lights, SSYNC, non-rigid ●● p. 416
        - + rigid
          Lights, SSYNC, rigid ●● p. 417
      - + internal light (FSTA)
        Finite state, SSYNC, non-rigid ●● p. 417
    - ASYNC, + lights (LUMI)
      Agreement on coordinates
      - disoriented, + rigid
        Lights, ASYNC, rigid ●● p. 418
      - + one axis
        Lights, ASYNC, non-rigid, one axis ●● p. 419
    - + FSYNC, + lights (LUMI), + chirality, + rigid
      Lights, FSYNC, rigid, chirality ●● p. 419
  - + crash faults
    Scheduler
    - + SSYNC, + lights (LUMI)
      Lights, mobility faults, SSYNC, non-rigid ●● p. 420
      - + compass, + rigid
        Lights, one crash, SSYNC, rigid, compass ●● p. 421
    - ASYNC, + lights (LUMI), + one axis
      Lights, crash faults, ASYNC, one axis ●● p. 422
  - Byzantine, + SSYNC, + lights (LUMI), + compass, + rigid
    Lights, one Byzantine robot ●● p. 423
- fat, + unlimited visibility, + no faults
  Scheduler
  - ASYNC, + lights (LUMI), + one axis, + rigid
    Fat robots, ASYNC, rigid, one axis ●● p. 423
  - + FSYNC, + lights (LUMI), + rigid
    Fat robots, FSYNC, rigid ●● p. 424

Claude (Anthropic) was used in cataloguing the problems.

What is known

The best upper bound against the best lower bound in each catalogued setting, in tree order. The settings follow in detail.

Setting	Time	Colours	p.
Lights, SSYNC, non-rigid	● unknown *finite* vs —	● tight 2 vs 2	416
Lights, SSYNC, rigid	● tight *finite* vs —	● tight 2 vs 2	417
Finite state, SSYNC, non-rigid	● unknown *finite* vs —	● tight 2 vs 2	417
Lights, ASYNC, rigid	● tight $O(1)$ vs $\Omega(1)$	● tight 47 vs 2	418
Lights, ASYNC, non-rigid, one axis	● unknown *finite* vs —	● tight 2 vs 2	419
Lights, FSYNC, rigid, chirality	● unknown $O(\log n)$ vs —	● tight 12 vs —	419
Lights, mobility faults, SSYNC, non-rigid	● unknown $O(n)$ vs —	● tight 12 vs 2	420
Lights, one crash, SSYNC, rigid, compass	● unknown *finite* vs —	● tight 3 vs —	421
Lights, crash faults, ASYNC, one axis	● unknown $O(n)$ vs —	● tight 3 vs 2	422
Lights, one Byzantine robot	● impossible	● impossible	423
Fat robots, ASYNC, rigid, one axis	● unknown $O(n)$ vs —	● tight $O(1)$ vs —	423
Fat robots, FSYNC, rigid	● unknown $O(n)$ vs —	● tight 2 vs 2	424

18.6.1 Lights, SSYNC, non-rigid

Adds + lights (LUMI) VERIFIED

With non-rigid moves in SSYNC, 2 colours suffice, which is optimal.

Assumed: body: point robots; visibility: unlimited; faults: no faults; scheduler: semi-synchronous (SSYNC); lights (persistent memory): full light (LUMI): every robot sees all lights, including its own. **Not assumed:** opacity: opaque: collinear robots hide each other; agreement on coordinates: no agreement on local coordinate systems; movement: non-rigid: the adversary may stop a move after some minimum distance δ; multiplicity detection: none: a point with several robots looks like one robot; randomness: deterministic; identifiers: anonymous: all robots are identical.

Measure	Lower bound	Best upper bound	Status
Time	none known Not analysed.	*finite* (Sharma et al. 2015b)	● unknown
Colours	2 (Di Luna et al. 2017) Exactly 2 colours (Sharma et al. 2015b).	2 (Sharma et al. 2015b)	● tight

Results.

- Di Luna et al. (2017) [lower bound]: colours 2. *Bottleneck:* memory. *Holds when:* Section 7: one colour does not suffice when n is unknown and robots must terminate.
- Di Luna et al. (2017) [introductory]: time *finite*, colours 3. Introductory (Theorem 1.2): 3 colours; 2 colours if δ or n is known.

Claude (Anthropic) was used in cataloguing the problems.

- Sharma et al. (2015b) [current best]: time *finite*, colours 2. Current best (Theorem 6, part of Theorem 3(a)): 2 colours without knowing δ or n, collision-free. Many proofs are omitted in this conference version.

Notes. Di Luna et al. asked whether their 2-colour algorithm works in non-rigid SSYNC without knowing δ (Di Luna et al. 2017); Sharma, Busch and Mukhopadhyay settle the colour count with a different 2-colour algorithm that needs neither δ nor n (Sharma et al. 2015b) (Theorem 6). Checked against the full texts of (Di Luna et al. 2017) and (Sharma et al. 2015b).

18.6.2 Lights, SSYNC, rigid

Adds + rigid **to** Lights, SSYNC, non-rigid (p. 416) VERIFIED

Opaque luminous robots reach mutual visibility in rigid SSYNC with 2 colours, which is optimal; oblivious robots suffice if n is known.

Assumed: body: point robots; visibility: unlimited; faults: no faults; scheduler: semi-synchronous (SSYNC); lights (persistent memory): full light (LUMI): every robot sees all lights, including its own; movement: rigid: every move reaches its destination. **Not assumed:** opacity: opaque: collinear robots hide each other; agreement on coordinates: no agreement on local coordinate systems; multiplicity detection: none: a point with several robots looks like one robot; randomness: deterministic; identifiers: anonymous: all robots are identical.

Measure	Lower bound	Best upper bound	Status
Time	none known $O(1)$ rounds with 12 colours (Sharma et al. 2016).	*finite* (Di Luna et al. 2017)	• tight
Colours	2 (Di Luna et al. 2017) Exactly 2 colours.	2 (Di Luna et al. 2017)	• tight

Results.

- Di Luna et al. (2017) [lower bound]: colours 2. *Bottleneck:* memory. *Holds when:* Section 7: one colour does not suffice when n is unknown and robots must terminate.
- Di Luna et al. (2017) [current best]: time *finite*, colours 2. Introductory and current best (Theorem 1.1(b)); first announced at SSS 2014 (Di Luna et al. 2014). With n known, oblivious robots (no colours) suffice.
- Sharma et al. (2016) [also]: time $O(1)$, colours 12. $O(1)$ time with 12 colours in SSYNC with rigid moves, as reported by (Sharma et al. 2017) (Table I) and (Poudel et al. 2021); the paper itself is not available locally and was not read.

Notes. Checked against the full text of (Di Luna et al. 2017).

18.6.3 Finite state, SSYNC, non-rigid

Adds + internal light (FSTA) VERIFIED

One bit of persistent memory that other robots cannot see suffices in SSYNC with non-rigid moves and no common orientation.

Assumed: body: point robots; visibility: unlimited; faults: no faults; scheduler: semi-synchronous (SSYNC); lights (persistent memory): internal light only (FSTA): a robot sees its own light, not the others'. **Not assumed:** opacity: opaque: collinear robots hide each other; agreement on coordinates: no agreement on local coordinate systems; movement: non-rigid: the adversary may stop a move after some minimum distance δ; multiplicity detection: none: a point with several robots looks like one robot; randomness: deterministic; identifiers: anonymous: all robots are identical.

Claude (Anthropic) was used in cataloguing the problems.

Measure	Lower bound	Best upper bound	Status
Time	none known Not analysed.	*finite* (Bhagat and Mukhopadhyaya 2019)	● unknown
Colours	2 (Di Luna et al. 2017) Exactly 2 states.	2 (Bhagat and Mukhopadhyaya 2019)	● tight

Results.

- Di Luna et al. (2017) [lower bound]: colours 2. *Bottleneck:* memory. *Holds when:* inherited: a robot with one state is oblivious, and one colour does not suffice when n is unknown and robots must terminate (Section 7).
- Bhagat and Mukhopadhyaya (2019) [current best]: time *finite*, colours 2. Introductory and current best (Theorem 1): two internal states (one bit), collision-free, without chirality or knowledge of n or δ; the robots terminate. The colours column counts internal states here.

Open questions.

Q1. Solve mutual visibility with finite-state robots under ASYNC (Bhagat and Mukhopadhyaya 2019).

Notes. Checked against the full text of (Bhagat and Mukhopadhyaya 2019) (arXiv v1, 2017).

18.6.4 Lights, ASYNC, rigid

Adds disoriented, + rigid

VERIFIED

In rigid ASYNC, 2 colours suffice, which is optimal, and $O(1)$ epochs are achievable with 47 colours.

Assumed: body: point robots; visibility: unlimited; faults: no faults; lights (persistent memory): full light (LUMI): every robot sees all lights, including its own; movement: rigid: every move reaches its destination. **Not assumed:** opacity: opaque: collinear robots hide each other; scheduler: asynchronous (ASYNC); agreement on coordinates: no agreement on local coordinate systems; multiplicity detection: none: a point with several robots looks like one robot; randomness: deterministic; identifiers: anonymous: all robots are identical.

Measure	Lower bound	Best upper bound	Status
Time	$\Omega(1)$ (Sharma et al. 2021a) $\Theta(1)$ epochs.	$O(1)$ (Sharma et al. 2021a)	● tight
Colours	2 (Di Luna et al. 2017) $\Theta(1)$; exactly 2 colours for finite time (Sharma et al. 2015b), 47 for $O(1)$ epochs.	47 (Sharma et al. 2021a)	● tight

Results.

- Di Luna et al. (2017) [lower bound]: colours 2. *Bottleneck:* memory. *Holds when:* Section 7: one colour does not suffice when n is unknown and robots must terminate.
- Sharma et al. (2021a) [lower bound]: time $\Omega(1)$. *Bottleneck:* trivial. *Holds when:* at least one epoch.
- Di Luna et al. (2017) [introductory]: time *finite*, colours 3. Collision-free.
- Sharma et al. (2015b) [also]: time *finite*, colours 2. Fewest colours (Theorem 7, part of Theorem 3(a)): 2 colours, which is optimal; collision-free, finite time, no time analysis. Many proofs are omitted in this conference version.
- Sharma et al. (2017) [also]: time $O(\log n)$, colours 25. Theorem 1.1: $O(\log n)$ epochs, collision-free, for any $n \geq 1$; $O(1)$ colours, 25 by its Table I. Superseded in time by (Sharma et al. 2021a).
- Sharma et al. (2021a) [current best]: time $O(1)$, colours 47. Theorem 1: $O(1)$ epochs, collision-free; first appeared in 2017. Its conclusion says 60 colours.

Claude (Anthropic) was used in cataloguing the problems.

Open questions.

Q1. Fewer colours with $O(1)$ epochs (Sharma et al. 2021a).

Q2. Time bounds with non-rigid moves.

Notes. Checked against the full texts of (Sharma et al. 2015b) (Theorem 7) and (Sharma et al. 2017) (Theorem 1.1, rigid moves, robots that may be disoriented). Both are conference versions that omit many proofs.

18.6.5 Lights, ASYNC, non-rigid, one axis

Adds + one axis VERIFIED

With non-rigid moves, 2 colours suffice in ASYNC if robots agree on one axis; without axis agreement the problem is open.

Assumed: body: point robots; visibility: unlimited; faults: no faults; lights (persistent memory): full light (LUMI): every robot sees all lights, including its own; agreement on coordinates: agreement on one axis (no common chirality). **Not assumed:** opacity: opaque: collinear robots hide each other; scheduler: asynchronous (ASYNC); movement: non-rigid: the adversary may stop a move after some minimum distance δ; multiplicity detection: none: a point with several robots looks like one robot; randomness: deterministic; identifiers: anonymous: all robots are identical.

Measure	Lower bound	Best upper bound	Status
Time	none known $O(n)$ epochs with 3 colours (Poudel et al. 2021); no lower bound beyond the trivial one.	*finite* (Sharma et al. 2015b)	• unknown
Colours	2 (Di Luna et al. 2017) Exactly 2 colours (Sharma et al. 2015b).	2 (Sharma et al. 2015b)	• tight

Results.

- Di Luna et al. (2017) [lower bound]: colours 2. *Bottleneck:* memory. *Holds when:* Section 7: one colour does not suffice when n is unknown and robots must terminate.
- Di Luna et al. (2017) [introductory]: time *finite*, colours 3. Introductory (Theorem 1.2(d)): 3 colours.
- Sharma et al. (2015b) [current best]: time *finite*, colours 2. Current best for colours (Theorem 8, Theorem 3(b)): 2 colours, which is optimal. The conference version only sketches this case.
- Poudel et al. (2021) [also]: time $O(n)$, colours 3. *Assumes:* $n \geq 3$. Only time bound for this setting: its crash-tolerant algorithm (Theorem 1.1) with $f = 0$ runs in $O(n)$ epochs with 3 colours, collision-free; each robot moves at most once and the robots do not end on a convex hull.

Open questions.

Q1. Mutual visibility in non-rigid ASYNC without axis agreement, ideally with 2 colours (Di Luna et al. 2017).

Notes. Checked against the full texts of (Di Luna et al. 2017), (Sharma et al. 2015b) and (Poudel et al. 2021).

18.6.6 Lights, FSYNC, rigid, chirality

Adds + FSYNC, + lights (LUMI), + chirality, + rigid VERIFIED

In FSYNC with rigid moves and chirality, mutual visibility takes $O(\log n)$ rounds with 12 colours.

Claude (Anthropic) was used in cataloguing the problems.

Assumed: body: point robots; visibility: unlimited; faults: no faults; scheduler: fully synchronous (FSYNC); lights (persistent memory): full light (LUMI): every robot sees all lights, including its own; agreement on coordinates: common chirality (same clockwise orientation); movement: rigid: every move reaches its destination. **Not assumed:** opacity: opaque: collinear robots hide each other; multiplicity detection: none: a point with several robots looks like one robot; randomness: deterministic; identifiers: anonymous: all robots are identical.

Measure	Lower bound	Best upper bound	Status
Time	none known $O(\log n)$ rounds here; the paper proves no lower bound and asks for one. $O(1)$ time is reported for rigid SSYNC without chirality (Sharma et al. 2016) (not read), a weaker setting whose algorithm would carry over.	$O(\log n)$ (Vaidyanathan et al. 2015)	● unknown
Colours	none known $\Theta(1)$.	12 (Vaidyanathan et al. 2015)	● tight

Results.

- Vaidyanathan et al. (2015) [current best]: time $O(\log n)$, colours 12. *Assumes:* $n \geq 4$. Introductory and current best (Theorem 11): $n \geq 4$ robots, $O(\log n)$ rounds, 12 colours, n unknown. Robot paths may cross; no two robots head for the same point.

Open questions.

Q1. What is the lower bound on the number of rounds, and can chirality and crossing paths be dropped (Vaidyanathan et al. 2015)?

Notes. Vaidyanathan, Busch, Trahan, Sharma and Rai also show that the algorithm of (Di Luna et al. 2014) takes $\Theta(n)$ rounds in the worst case even in FSYNC. Checked against the full text of (Vaidyanathan et al. 2015) (model in Section IV, Theorem 11). The variant was previously recorded without rigid moves; the paper assumes that "a robot in motion cannot be stopped".

18.6.7 Lights, mobility faults, SSYNC, non-rigid

Adds + SSYNC, + lights (LUMI) VERIFIED

With any number of mobility faults, luminous robots without axis agreement reach mutual visibility in non-rigid SSYNC in $O(n)$ epochs with 12 colours, if they all see a common fixed reference point.

Assumed: body: point robots; visibility: unlimited; faults: up to f robots may crash; scheduler: semi-synchronous (SSYNC); lights (persistent memory): full light (LUMI): every robot sees all lights, including its own. **Not assumed:** opacity: opaque: collinear robots hide each other; agreement on coordinates: no agreement on local coordinate systems; movement: non-rigid: the adversary may stop a move after some minimum distance δ; multiplicity detection: none: a point with several robots looks like one robot; randomness: deterministic; identifiers: anonymous: all robots are identical.

Statement for this setting. Any number f of robots, unknown to the robots, may suffer a mobility fault at any time: a faulty robot never moves again, but keeps performing Look and Compute and changing its light, and cannot tell that it is faulty. The non-faulty robots must reach, without collisions, a configuration in which every non-faulty robot sees every other robot, faulty or not (Shukla et al. 2026a). (A weaker goal, used by Poudel et al. 2021 in ASYNC, asks only that no three non-faulty robots be collinear and that no faulty robot lie between two non-faulty ones.)

Measure	Lower bound	Best upper bound	Status
Time	none known $O(n)$ epochs; no lower bound beyond the trivial one.	$O(n)$ (Shukla et al. 2026a) [preprint]	● unknown
Colours	2 (Di Luna et al. 2017) $\Theta(1)$: between 2 and 12 colours.	12 (Shukla et al. 2026a) [preprint]	● tight

Claude (Anthropic) was used in cataloguing the problems.

Results.

- Di Luna et al. (2017) [lower bound]: colours 2. *Bottleneck:* memory. *Holds when:* fault-free bound (no robot faulty): one colour does not suffice when n is unknown and robots must terminate (Section 7 of Di Luna et al. 2017).
- Shukla et al. (2026a) [preprint] [current best]: time $O(n)$, colours 12. *Assumes:* $n \geq 3$; a common fixed reference point that every robot sees in every Look, whatever robots lie in between, and that lies outside the convex hull of the initial configuration; it gives no common direction; a faulty robot keeps changing its light: the fault stops only its movement; every robot travels a total distance of at most a budget $B > 0$, the same for all, and knows how much of it remains. Theorem 1: $O(n)$ epochs (Lemma 22), 12 colours, collision-free (Lemma 21), and each robot moves at most four times (Lemma 10); the robots are not sent to corners of a convex hull. A preprint; several proofs are only sketched.

Open questions.

Q1. Can mobility-fault-tolerant mutual visibility under a movement budget be solved without the common reference point and without any axis agreement (Shukla et al. 2026a)?

Q2. Does the budget-constrained algorithm extend to ASYNC (Shukla et al. 2026a)?

Notes. Shukla, Maiti, Kumar and Bhagat add a movement budget: each robot may travel a total distance of at most B, so robots cannot be sent to the corners of a distant convex hull. Their algorithm moves robots by small steps, at most a fixed fraction of the distances to the other robots it sees and to the lines through them, in four phases encoded by the light colours (Section 3): every non-faulty robot first makes one step off the lines through it, the robots at the ends of each line before those between them; then robots that are still between two others step towards the reference point, the farthest from it first, which breaks cyclic dependencies; finally robots at the ends of lines make one step towards it, to get past faulty robots. Their goal is stronger than that of (Poudel et al. 2021): a non-faulty robot must also see every faulty robot. As reported by (Shukla et al. 2026a) (Section 1.1 and Table 1), Pramanick, Jana and Mandal solve mobility-fault-tolerant mutual visibility without axis agreement and without a reference point: for one fault in SSYNC and ASYNC and any number in FSYNC (TCS 2025), and for any number of faults in ASYNC with 35 colours in $O(n)$ epochs (SIROCCO 2026); neither paper is in the bibliography and neither was read. An algorithm for ASYNC would carry over to this setting. Checked against the full text of (Shukla et al. 2026a) (arXiv v1: model in Section 2, Theorem 1, Lemmas 10 and 21–23) and Section 7 of (Di Luna et al. 2017).

18.6.8 Lights, one crash, SSYNC, rigid, compass

Adds + compass, + rigid **to** Lights, mobility faults, SSYNC, non-rigid (p. 420) VERIFIED

With one crash fault, robots that share a compass reach mutual visibility in rigid SSYNC with 3 colours.

Assumed: body: point robots; visibility: unlimited; faults: up to f robots may crash; scheduler: semi-synchronous (SSYNC); lights (persistent memory): full light (LUMI): every robot sees all lights, including its own; agreement on coordinates: agreement on both axes (common compass); movement: rigid: every move reaches its destination. **Not assumed:** opacity: opaque: collinear robots hide each other; multiplicity detection: none: a point with several robots looks like one robot; randomness: deterministic; identifiers: anonymous: all robots are identical.

Measure	Lower bound	Best upper bound	Status
Time	none known —	*finite* (Aljohani and Sharma 2018)	• unknown
Colours	none known $\Theta(1)$.	3 (Aljohani and Sharma 2018)	• tight

Results.

Claude (Anthropic) was used in cataloguing the problems.

- Aljohani and Sharma (2018) [current best]: time *finite*, colours 3. Introductory and current best; one crash fault.

Open questions.

Q1. Tolerating three or more crashes, and using 2 colours (Aljohani and Sharma 2018).

Notes. With up to $f \leq n$ crashes, robots that agree on one axis reach mutual visibility in non-rigid ASYNC in $O(n)$ epochs with 3 colours (setting *Lights, crash faults, ASYNC, one axis*), and in SSYNC with 2 colours (Poudel et al. 2021) (Theorems 1.1 and 4.10, $n \geq 3$). Their crash stops only movement: a crashed robot must still change its light, and Poudel, Aljohani and Sharma leave the case of lights that stop with the robot as "a major open problem". Checked against the full texts of (Aljohani and Sharma 2018) and (Poudel et al. 2021).

18.6.9 Lights, crash faults, ASYNC, one axis

Adds ASYNC, + lights (LUMI), + one axis VERIFIED

With up to n crashed robots, robots that agree on one axis reach mutual visibility in non-rigid ASYNC in $O(n)$ epochs with 3 colours.

Assumed: body: point robots; visibility: unlimited; faults: up to f robots may crash; lights (persistent memory): full light (LUMI): every robot sees all lights, including its own; agreement on coordinates: agreement on one axis (no common chirality). **Not assumed:** opacity: opaque: collinear robots hide each other; scheduler: asynchronous (ASYNC); movement: non-rigid: the adversary may stop a move after some minimum distance δ; multiplicity detection: none: a point with several robots looks like one robot; randomness: deterministic; identifiers: anonymous: all robots are identical.

Statement for this setting. Up to $f \leq n$ robots may crash, at unknown times; a crashed robot never moves again. The correct robots must reach, without collisions, a configuration in which no three correct robots are collinear and no crashed robot lies between two correct ones (Poudel et al. 2021).

Measure	Lower bound	Best upper bound	Status
Time	none known $O(n)$ epochs; no lower bound beyond the trivial one.	$O(n)$ (Poudel et al. 2021)	● unknown
Colours	2 (Di Luna et al. 2017) 2 or 3 colours.	3 (Poudel et al. 2021)	● tight

Results.

- Di Luna et al. (2017) [lower bound]: colours 2. *Bottleneck:* memory. *Holds when:* fault-free bound (no robot crashes): one colour does not suffice when n is unknown and robots must terminate (Section 7 of Di Luna et al. 2017).
- Poudel et al. (2021) [current best]: time $O(n)$, colours 3. *Assumes:* $n \geq 3$; any $f \leq n$ crashes, f unknown; a crashed robot's light keeps working: the crash stops only its movement. Introductory and current best (Theorem 1.1): collision-free, non-rigid moves, each robot moves at most once, and the robots do not end on a convex hull. Improves the 4 colours of the WALCOM 2018 version to 3. In SSYNC 2 colours suffice (Theorem 4.10).

Open questions.

Q1. Mutual visibility with several crashes when a crashed robot's light stops changing too, which (Poudel et al. 2021) call "a major open problem".

Q2. Two colours in ASYNC with crashes (Poudel et al. 2021).

Claude (Anthropic) was used in cataloguing the problems.

Notes. The fault model differs from that of (Aljohani and Sharma 2018) (one crash, SSYNC, rigid moves, compass): here faults affect only mobility, and lights must keep working after the crash. Checked against the full text of (Poudel et al. 2021) (Theorems 1.1 and 4.10).

18.6.10 Lights, one Byzantine robot

Adds Byzantine, + SSYNC, + lights (LUMI), + compass, + rigid VERIFIED

With one Byzantine robot among three, mutual visibility is impossible.

Assumed: body: point robots; visibility: unlimited; scheduler: semi-synchronous (SSYNC); lights (persistent memory): full light (LUMI): every robot sees all lights, including its own; agreement on coordinates: agreement on both axes (common compass); movement: rigid: every move reaches its destination. **Not assumed:** opacity: opaque: collinear robots hide each other; faults: up to f Byzantine robots; multiplicity detection: none: a point with several robots looks like one robot; randomness: deterministic; identifiers: anonymous: all robots are identical.

Measure	Lower bound	Best upper bound	Status
Time	*impossible* (Aljohani and Sharma 2018) For $n = 3$.	—	• impossible
Colours	none known For $n = 3$.	—	• impossible

Results.

- Aljohani and Sharma (2018) [lower bound]: time *impossible*. *Bottleneck:* indistinguishable Byzantine agents. *Holds when:* $n = 3$ with one Byzantine robot.

Notes. Checked against the full text.

18.6.11 Fat robots, ASYNC, rigid, one axis

Adds ASYNC, + lights (LUMI), + one axis, + rigid VERIFIED

Fat robots that agree on one axis reach mutual visibility in rigid ASYNC in $O(n)$ epochs with $O(1)$ colours.

Assumed: visibility: unlimited; faults: no faults; lights (persistent memory): full light (LUMI): every robot sees all lights, including its own; agreement on coordinates: agreement on one axis (no common chirality); movement: rigid: every move reaches its destination. **Not assumed:** body: fat robots (unit discs); opacity: opaque: collinear robots hide each other; scheduler: asynchronous (ASYNC); multiplicity detection: none: a point with several robots looks like one robot; randomness: deterministic; identifiers: anonymous: all robots are identical.

Measure	Lower bound	Best upper bound	Status
Time	none known No lower bound is recorded. (Bhagat et al. 2024) attribute an $\Omega(\sqrt{n})$ lower bound to (Sharma et al. 2018), but that paper (ICDCN 2018) states no lower bound.	$O(n)$ (Bhagat et al. 2024)	• unknown
Colours	none known $\Theta(1)$.	$O(1)$ (Bhagat et al. 2024)	• tight

Results.

- Bhagat et al. (2024) [current best]: time $O(n)$, colours $O(1)$. *Assumes:* $n \geq 3$; robots know their common radius. Introductory and current best for ASYNC (Theorem 1, Lemma 3.17): $n \geq 3$ unit discs, collision-free, n unknown; the robots end on a semicircle.

Open questions.

Claude (Anthropic) was used in cataloguing the problems.

Q1. Non-rigid moves, and SSYNC and ASYNC extensions of the 2-colour FSYNC algorithm (Alsaedi et al. 2023a).

Q2. Mutual visibility of fat robots in ASYNC with non-rigid moves and no axis agreement (Bhagat et al. 2024).

Notes. Checked against the full text of (Bhagat et al. 2024) (Section 2, Theorem 1). The variant was previously recorded without rigid moves; the paper assumes "rigid movement". Its count of colours is $O(1)$; the phases use 3 to 5 colours each (Lemmas 3.3, 3.7 and 3.14), and the total is not stated.

18.6.12 Fat robots, FSYNC, rigid

Adds + FSYNC, + lights (LUMI), + rigid VERIFIED

In FSYNC, fat robots without chirality reach mutual visibility in $O(n)$ rounds with 2 colours, which is optimal.

Assumed: visibility: unlimited; faults: no faults; scheduler: fully synchronous (FSYNC); lights (persistent memory): full light (LUMI): every robot sees all lights, including its own; movement: rigid: every move reaches its destination. **Not assumed:** body: fat robots (unit discs); opacity: opaque: collinear robots hide each other; agreement on coordinates: no agreement on local coordinate systems; multiplicity detection: none: a point with several robots looks like one robot; randomness: deterministic; identifiers: anonymous: all robots are identical.

Measure	Lower bound	Best upper bound	Status
Time	none known —	$O(n)$ (Alsaedi et al. 2023a)	● unknown
Colours	2 (Alsaedi et al. 2023a) Exactly 2 colours.	2 (Alsaedi et al. 2023a)	● tight

Results.

- Alsaedi et al. (2023a) [lower bound]: colours 2. *Bottleneck:* memory. *Holds when:* inherited from point robots (a lower bound for point robots holds for fat ones); not reproved for discs in (Alsaedi et al. 2023a).
- Sharma et al. (2018) [introductory]: time $O(n)$, colours 9. Theorem 1.1: unit discs, $n \geq 1$, $O(n)$ rounds, 9 colours, collision-free, no chirality, rigid moves. A short paper; details are deferred to a full version.
- Alsaedi et al. (2023a) [current best]: time $O(n)$, colours 2. Current best; no chirality. A preprint, not peer reviewed.

Notes. Checked against the full texts of (Alsaedi et al. 2023a) and (Sharma et al. 2018).

18.7 k-grouping

Robots must form k distinct groups, each group at its own point.

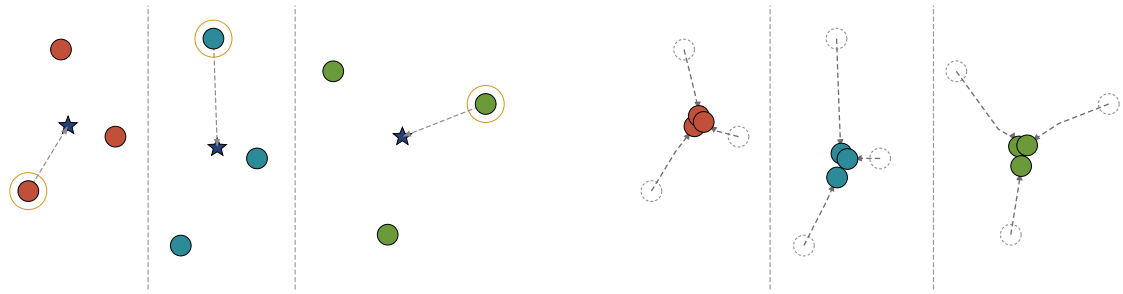
First posed in 2007 (Efrima and Peleg 2009). Splitting a swarm into a given number of groups was introduced at SIROCCO 2007, under the name partitioning. No paper titled "k-grouping" was found.

Problem statement and measures

n robots must form exactly k distinct **groups**, each group gathered at its own point or converging towards one, and stay apart. The number of groups k is given; the group sizes are free unless the variant fixes them. With $k = 1$ this is gathering (or convergence).

Claude (Anthropic) was used in cataloguing the problems.

Measures: **Time**, rounds (FSYNC, SSYNC) or epochs (ASYNC) until the task is done; "finite" when only solvability is known; **Colours**, colours of the light per robot (0 for oblivious robots).



(a) Start: $n = 9$, $k = 3$; blocks of 3 by x-order

(b) Final: 3 groups, each converging on its own point (arrows: moves; dashed circles: starting positions)

Figure 18.7: k-grouping: n robots must split into exactly k groups, each gathered at (or converging to) its own point, the points distinct. Shown: the algorithm of (Efrima and Peleg 2009) with a common compass, $n = 9$, $k = 3$, in SSYNC rounds: robots are cut into blocks of n/k by x-order (colours), and each ringed (activated) robot moves towards the centre of gravity (star) of its block, possibly stopping early; blocks never mix.

A first look

Imagine a swarm that must split into k teams of equal size, each team clustering in its own place, far from the others. In the problem the members of the swarm are n **robots**, points at distinct places in the plane, and the teams are **groups**; k divides n. The positions of all robots form the **configuration**. Each robot reads positions in its own **local coordinate system** (its private axes, unit of length and handedness) and repeats a **Look–Compute–Move cycle**: it takes a **snapshot** of where the other robots are (Look), decides on a destination point (Compute) and moves towards it (Move). Robots are identical, silent and **oblivious** (no memory between cycles). In the basic version robots share a **compass** (all local coordinate systems agree on north and east, though not on origin or unit of length); the **scheduler**, which decides the **activation** of robots (when each starts a cycle) and how long each phase takes, is ASYNC (each phase may take any finite time); and some robots may **crash** (stop forever). The robots must split into k groups of n/k robots each, every group converging towards its own point, with the groups staying a fixed distance apart; with $k = 1$ this is just convergence.

1. Algorithm: Sort by the compass, cut into blocks, converge within each block. (Efrima and Peleg 2009) (k groups of equal size, p. 427)

1. Sort the robots from west to east, and robots on one north-south line from south to north. All robots get the same order, because they share north and east.
2. Cut the order into k consecutive blocks of n/k robots: the first n/k robots, the next n/k, and so on. Each robot finds its own block.
3. Move towards the **centre of gravity** of your block, the average of the positions of its robots.

Why it works and what it costs. The region spanned by a block's robots and their destinations never grows when they move to their centre of gravity (Lemma 2.5 of Efrima and Peleg 2009, from Cohen and Peleg). The blocks start in disjoint regions ordered from west to east, so two robots of different blocks can never swap order, and the blocks cut in step 2 stay the same forever, even though robots forget them. Each block therefore converges on its own, and a crashed robot simply stays inside its block's

Claude (Anthropic) was used in cataloguing the problems.

region, so any number of crashes is tolerated (Proposition 2.7 of Efrima and Peleg 2009). This is the catalogue's introductory and current best result.

Why it matters. It shows a clean way to make oblivious robots keep an agreement forever: choose it so that the motion itself preserves it.

2. Impossibility: No compass: the regular polygon cannot be split. (Efrima and Peleg 2009) (*k* equal groups, no common orientation, deterministic, p. 429) Start the n robots on the corners of a regular n-gon, with local coordinate systems that are copies of each other turned around the centre. Every robot sees the same snapshot, so in every **round** of the fully synchronous scheduler FSYNC (a step of time in which every robot performs one full cycle) all robots make the same move up to rotation, and the configuration stays a regular polygon (or a single point). But $k \geq 2$ groups of several robots each, far apart, never look like a regular polygon or a point. So the groups never form.

Why it holds. This is Proposition 3.3 of (Efrima and Peleg 2009): deterministic robots without a compass cannot partition themselves into $k > 1$ groups, already in FSYNC and therefore in SSYNC and ASYNC. Agreeing only on the axis lines but not their directions is still not enough for some n and k (a symmetric square with $n = 4$, $k = 2$; Proposition 3.4).

Why it matters. It shows that partitioning, unlike gathering, needs a way to break symmetry.

3. Algorithm: A coin flip breaks the tie. (Efrima and Peleg 2009) (*k* equal groups, randomized, SSYNC, p. 428) Without a compass, the trouble is sets of robots that look exactly alike. Randomness helps: a robot in such a tied set chooses its direction of motion at random. Two robots that would otherwise mirror each other forever now choose differently with probability at least $1/2$ in each attempt, so the tie is broken after a few attempts on average, and the robots then proceed with the deterministic method.

Why it works and what it costs. Efrima and Peleg give a randomized algorithm, of the kind that always ends correctly and is only random in its running time, that partitions robots without any common orientation in FSYNC and SSYNC and tolerates crashes (Efrima and Peleg 2009). The description above is simplified: in the paper the random choice is used only to separate robots that lie on the same line.

Why it matters. It is the standard illustration that randomness breaks the symmetry that defeats deterministic robots.

What the community started looking at

Splitting a swarm into a given number of groups was asked as **solvability under each timing model and amount of orientation** (Efrima and Peleg 2009), who called it partitioning and required equal group sizes. Team assembling is a related task for robots of q fixed colours: a specification $(a_1, \ldots, a_q)$ says how many robots of each colour form one team, and the robots must gather into teams at distinct points from any configuration. Liu, Yamauchi, Kijima and Yamashita show that oblivious ASYNC robots with chirality and strong multiplicity detection can form a number $k > 1$ of teams from every configuration in which no two robots of one colour share a point if and only if $\gcd(a_1, \ldots, a_q) = 1$ (Theorem 1 of Liu et al. 2018; their m is k here). With one colour this means that identical robots cannot gather into $k > 1$ groups of equal size greater than one; the necessity proof keeps the symmetry of the start already in FSYNC with rigid moves (Lemma 2 of Liu et al. 2018).

Relations to other problems

- **Special case of Pattern formation.** When the groups sit at points, k-grouping forms a pattern of k points; which points, and how many robots each holds, is up to the robots. Gathering is its special case $k = 1$. Splitting needs symmetry breaking (Efrima and Peleg 2009).

Claude (Anthropic) was used in cataloguing the problems.

- **Derived problem:** Gathering (special case of). Gathering is k-grouping with $k = 1$, a single group of all n robots, that is, point formation: the pattern consisting of one point of multiplicity n (Flocchini et al. 2017, 2026). General pattern formation algorithms usually exclude it, because multiplicity points need detection (Bramas and Tixeuil 2016).
- **Related:** Partitioning. k-grouping fixes the number of groups; partitioning fixes the size of each group. With equal sizes the two coincide: k groups of n/k robots.
- **Related:** Separation. Both split the swarm into classes; in separation the classes are fixed by hidden colours, in k-grouping and partitioning they are chosen by the robots.

Case tree

The root assumes nothing; going down, each edge adds the assumptions written in grey (+) or leaves an axis unassumed. Bold entries are catalogued settings with their status per measure (Time, Colours) and page; italic entries are branch points, named after the axis on which their branches differ. Axis order: Body → Visibility → Opacity → Faults → Scheduler → Lights (persistent memory) → Agreement on coordinates → Movement → Multiplicity detection → Randomness → Identifiers.

No assumptions

+ point, + unlimited visibility, + transparent

Faults

+ crash faults

Scheduler

ASYNC, + compass

k groups of equal size STANDARD SETTING — ● ● p. 427

+ SSYNC, + randomized

k equal groups, randomized, SSYNC — ● ● p. 428

+ no faults, + FSYNC

k equal groups, no common orientation, deterministic — ● ● p. 429

What is known

The best upper bound against the best lower bound in each catalogued setting, in tree order. The settings follow in detail.

Setting	Time	Colours	p.
k groups of equal size STANDARD SETTING	● unknown *finite* vs —	● tight 0 vs —	427
k equal groups, randomized, SSYNC	● unknown *finite* vs —	● tight 0 vs —	428
k equal groups, no common orientation, deterministic	● impossible	● tight — vs —	429

18.7.1 k groups of equal size

Adds ASYNC, + compass — STANDARD SETTING VERIFIED

Robots split into k groups of n/k that converge apart; with a full compass a deterministic crash-tolerant algorithm works in ASYNC.

Claude (Anthropic) was used in cataloguing the problems.

Assumed: body: point robots; visibility: unlimited; opacity: transparent: every robot sees every robot within range; faults: up to f robots may crash; agreement on coordinates: agreement on both axes (common compass). **Not assumed:** scheduler: asynchronous (ASYNC); lights (persistent memory): oblivious, no light (OBLOT); movement: non-rigid: the adversary may stop a move after some minimum distance δ; multiplicity detection: none: a point with several robots looks like one robot; randomness: deterministic; identifiers: anonymous: all robots are identical.

Statement for this setting. Partition(n, k) of Efrima and Peleg: k is given to the robots, every group has exactly n/k robots ($\lfloor n/k \rfloor$ or $\lfloor n/k \rfloor + 1$ when k does not divide n), robots in one group **converge** (for every ε they are eventually within ε of each other) and groups stay more than 2η apart, for some fixed $\eta > 0$. Robots start at distinct points and can count the robots at a point.

Measure	Lower bound	Best upper bound	Status
Time	none known Only solvability.	*finite* (Efrima and Peleg 2009)	● unknown
Colours	none known No colours.	0 (Efrima and Peleg 2009)	● tight

Results.

- Efrima and Peleg (2009) [current best]: time *finite*, colours 0. *Assumes:* strong multiplicity detection. Introductory and current best: deterministic, crash-fault tolerant, in every timing model with a full compass. Also crash-tolerant in FSYNC and SSYNC with a half compass (both axis directions, one positive orientation, no chirality), and not crash-tolerant in ASYNC with a half compass.

Notes. Efrima and Peleg call this partitioning; the catalogue's partitioning also fixes, in advance, the k points at which the groups of n/k robots gather. Convergence is Partition$(n, 1)$. Checked against the full text.

18.7.2 k equal groups, randomized, SSYNC

Adds + SSYNC, + randomized VERIFIED

With randomness, robots without any common orientation partition in SSYNC, tolerating crashes.

Assumed: body: point robots; visibility: unlimited; opacity: transparent: every robot sees every robot within range; faults: up to f robots may crash; scheduler: semi-synchronous (SSYNC); randomness: randomized. **Not assumed:** lights (persistent memory): oblivious, no light (OBLOT); agreement on coordinates: no agreement on local coordinate systems; movement: non-rigid: the adversary may stop a move after some minimum distance δ; multiplicity detection: none: a point with several robots looks like one robot; identifiers: anonymous: all robots are identical.

Statement for this setting. Partition(n, k) of Efrima and Peleg: k is given to the robots, every group has exactly n/k robots ($\lfloor n/k \rfloor$ or $\lfloor n/k \rfloor + 1$ when k does not divide n), robots in one group **converge** (for every ε they are eventually within ε of each other) and groups stay more than 2η apart, for some fixed $\eta > 0$. Robots start at distinct points and can count the robots at a point.

Measure	Lower bound	Best upper bound	Status
Time	none known Only solvability.	*finite* (Efrima and Peleg 2009)	● unknown
Colours	none known No colours.	0 (Efrima and Peleg 2009)	● tight

Results.

- Efrima and Peleg (2009) [current best]: time *finite*, colours 0. *Assumes:* strong multiplicity detection; adaptive adversary. Introductory and current best: randomized, crash-fault tolerant, FSYNC and SSYNC.

Claude (Anthropic) was used in cataloguing the problems.

Open questions.

Q1. Partitioning in ASYNC without a common orientation, by a randomized algorithm with no extra memory.

Notes. Checked against the full text.

18.7.3 k equal groups, no common orientation, deterministic

Adds + no faults, + FSYNC VERIFIED

Without any common orientation no deterministic algorithm partitions robots into $k > 1$ groups, already in FSYNC.

Assumed: body: point robots; visibility: unlimited; opacity: transparent: every robot sees every robot within range; faults: no faults; scheduler: fully synchronous (FSYNC). **Not assumed:** lights (persistent memory): oblivious, no light (OBLOT); agreement on coordinates: no agreement on local coordinate systems; movement: non-rigid: the adversary may stop a move after some minimum distance δ; multiplicity detection: none: a point with several robots looks like one robot; randomness: deterministic; identifiers: anonymous: all robots are identical.

Statement for this setting. Partition(n, k) of Efrima and Peleg: k is given to the robots, every group has exactly n/k robots ($\lfloor n/k \rfloor$ or $\lfloor n/k \rfloor + 1$ when k does not divide n), robots in one group **converge** (for every ε they are eventually within ε of each other) and groups stay more than 2η apart, for some fixed $\eta > 0$. Robots start at distinct points and can count the robots at a point.

Measure	Lower bound	Best upper bound	Status
Time	*impossible* (Efrima and Peleg 2009) Deterministically unsolvable for $k > 1$.	—	• impossible
Colours	none known No colours.	—	• tight

Results.

- Efrima and Peleg (2009) [lower bound]: time *impossible*. *Bottleneck:* symmetry breaking. *Holds when:* Proposition 3.3: deterministic, $k > 1$; robots start at the vertices of a regular n-gon and always make the same move; holds even in FSYNC, hence in SSYNC and ASYNC; with common axes but no orientation, still unsolvable for some n, k (Proposition 3.4).

Notes. In a companion paper the same authors show that the problem becomes solvable in ASYNC without a compass if the start is asymmetric, if one robot is identifiable, or, randomized against a non-adaptive adversary, with one bit of memory and communication (as stated in Efrima and Peleg 2009). Checked against the full text.

18.8 Gathering

Robots must meet at a single point that is not fixed in advance, and stay there.

First posed in 1999 (Suzuki and Yamashita 1999). Suzuki and Yamashita gave an algorithm "moving the robots to a single point in finite steps" and showed that two oblivious robots cannot meet in SSYNC.

Claude (Anthropic) was used in cataloguing the problems.

Problem statement and measures

n robots start at distinct points and must all reach the same point in finite time and remain there. For $n = 2$ the problem is called rendezvous. **Time** is counted in rounds (FSYNC, SSYNC) or in epochs (ASYNC: intervals in which every robot completes at least one full Look–Compute–Move cycle); **colours** count light colours.

Measures: **Time**, rounds (FSYNC, SSYNC) or epochs (ASYNC) until the task is done; "finite" when only solvability is known; **Colours**, colours of the light per robot (0 for oblivious robots).



(a) Start: symmetric, no multiplicity: all aim at the centre

(b) Final: all robots on one point (arrows: moves; dashed circles: starting positions)

Figure 18.8: Gathering: anonymous oblivious robots at distinct points of the plane must meet at one point, not fixed in advance, and stay there. Shown: the strategy of (Cieliebak et al. 2012) with weak multiplicity detection, in SSYNC rounds (ringed robots are activated; a move may stop early). In a symmetric configuration every robot computes the same centre; once a single multiplicity exists, every robot walks to it.

A first look

Imagine robots scattered over a field that must all meet at one spot, although nobody has said where. There are n **robots**, points in the plane at distinct places; their positions form the **configuration**. Each robot repeats a **Look–Compute–Move cycle**: it takes a **snapshot** of where the other robots are (Look), decides on a destination point (Compute) and moves towards it (Move). They are identical, silent and **oblivious** (they remember nothing from one cycle to the next), and each uses its own **local coordinate system** (origin, axes, unit of length and handedness), so they agree on no direction. An adversary, the **scheduler**, decides the **activation** of robots, that is, when each performs a cycle: in SSYNC time runs in **rounds** and in each round it activates some of the robots; in ASYNC every phase may take any finite time. The robots must all reach the same point in finite time and stay there (for $n = 2$ this is called rendezvous). In the standard version $n \geq 3$, the scheduler is ASYNC, and robots have **weak multiplicity detection**: they can tell whether a point holds one robot or several (a **multiplicity**), but not how many.

1. Algorithm: Create one multiplicity, then every robot walks to it. (Cieliebak et al. 2012) (ASYNC with weak multiplicity detection, p. 448)

1. If exactly one point is a multiplicity, walk straight to it.
2. Otherwise, look at the shape of the configuration and choose a point that every robot identifies in the same way despite their different local coordinate systems (for example the centre of a symmetric configuration), and move so that exactly one multiplicity appears.
3. From then on every robot follows rule 1, along paths that never create a second multiplicity.

Claude (Anthropic) was used in cataloguing the problems.

Why it works and what it costs. A multiplicity is a landmark that no local coordinate system can hide: it looks special to every robot. Step 2 is the hard part, because the robots agree on no direction and the scheduler may stop a robot partway; Cieliebak, Flocchini, Prencipe and Santoro solve it with a case analysis of configuration types and gather $n \geq 3$ robots in ASYNC from every initial configuration (Cieliebak et al. 2012), the catalogue's introductory and current best result for this setting. In their algorithm a robot first checks for a single multiplicity; otherwise it checks for one symmetric shape (biangular) and moves to its centre; otherwise the robots elect some of themselves from the angles they form around the centre of the smallest enclosing circle: a single elected robot walks onto another robot, and several elected robots walk to the centre. Collisions anywhere else are avoided. Time is not analysed.

Why it matters. It shows how a tiny sensing ability, telling one robot from several, makes an unsolvable problem solvable.

2. Impossibility: Two oblivious robots cannot meet in SSYNC. (Suzuki and Yamashita 1999) (Two robots, SSYNC, p. 444) Take two robots, A and B, and look at the round in which they would finally meet.

(a) If both moved in that round, the scheduler could have activated only one: it lands where the other was heading, not where the other is, so they miss.

(b) If only A moved and landed on B, the scheduler could have activated both; unless B's rule tells it to stay put, B moves away and they miss again.

So meeting needs a situation in which, when both are activated, one robot jumps onto the other while the other stays. (c) But the robots are identical and agree on no direction, so the adversary can give them local coordinate systems in which each sees the other at the same spot, say one unit to its east. Identical robots with identical snapshots make the same choice: both stay or both move. The lopsided situation never occurs, and they never meet.

Why it holds. Suzuki and Yamashita proved this as Theorem 3.1 of (Suzuki and Yamashita 1999), even for robots with common handedness whose moves always reach their destination, so it also holds without these advantages. Natural rules fail in instructive ways: \"move to the midpoint\" brings the robots closer forever, but the scheduler activates one robot at a time so they never coincide; \"move to the other robot\" makes them swap places forever when both are activated. Extra power fixes this: with lights that both robots see, two colours suffice even in ASYNC (Heriban et al. 2018).

Why it matters. It is the founding impossibility of the field: the scheduler keeps identical robots symmetric, and most later assumptions (multiplicity detection, lights, compasses) are ways to break that symmetry.

3. Impossibility: Without multiplicity detection, a multiplicity looks like one robot. (Prencipe 2007) (No multiplicity detection, SSYNC, p. 444) Suppose the robots stand on only two points, say $n - 1$ robots on p and one robot on q. A robot that cannot detect multiplicities sees just two occupied points, exactly what it would see if there were only two robots. The robots are thrown back into the two-robot situation above, and the scheduler, by activating only some of the robots on a point, can move groups around so that they never all stand together. Prencipe shows that against any algorithm the scheduler can first drive the robots into such special configurations and then keep them from ever gathering; the full proof is a longer case analysis.

Why it holds. The result holds for deterministic oblivious robots without multiplicity detection, for every $n \geq 2$, in SSYNC and ASYNC (Prencipe 2007). In the fully synchronous model, by contrast, gathering needs no multiplicity detection (Flocchini et al. 2026), and in ASYNC weak multiplicity detection is enough (Cieliebak et al. 2012).

Why it matters. It explains why weak multiplicity detection became the standard assumption for gathering.

4. Algorithm: Short sight but a common compass: drift right, then down. (Flocchini et al. 2005)

Claude (Anthropic) was used in cataloguing the problems.

(Limited visibility with a compass, ASYNC, p. 454) Now each robot sees only the robots within distance V (its **visibility radius**), but all robots agree on north and east (a **common compass**).

1. If you see a robot to your left, or above you on your own vertical line, do nothing.
2. If you see robots only below you on your vertical line, move down towards the nearest one.
3. If you see robots only to your right, move horizontally to the vertical line of the nearest one.
4. If you see robots both below you and to your right, make a diagonal move right and down, chosen so that you lose sight of nobody.

Why it works and what it costs. Robots only move right or down and never beyond the rightmost vertical line that holds a robot in the initial configuration, so they collect on that line and then at its lowest occupied point; the delicate part of the proof is that nobody is left behind out of sight. Flocchini, Prencipe, Santoro and Widmayer prove that this gathers ASYNC robots with non-rigid moves (Theorem 4.2 of Flocchini et al. 2005), provided the initial configuration is connected (robots joined when at most V apart); a disconnected initial configuration cannot be gathered at all. With rigid moves, a common compass gathers the robots at a point in time linear in the initial diameter, which is optimal; with agreement on one axis only, the same time brings them onto a horizontal segment of length 1, not to a point (Poudel and Sharma 2021).

Why it matters. It shows how a shared direction can replace global vision: robots that see little can still agree where to go.

Simulator

A simulator of Look–Compute–Move robots in the plane that runs in the browser. Each case below starts one of its algorithms from a fixed starting configuration and seed; the run is the simulator's, not a proof. On the website each case runs in the page; here it is a link.

- **Gathering under a sequential scheduler.** Six robots with weak multiplicity detection, activated one at a time. The simulator's documentation gives the algorithm as Algorithm 8 of the arXiv version of (Flocchini et al. 2026). Open in the simulator.

What the community started looking at

Gathering started with an impossibility: two oblivious robots cannot meet in SSYNC (Suzuki and Yamashita 1999). The community's main question became **the weakest set of capabilities that makes gathering solvable**: multiplicity detection (Cieliebak et al. 2012), a compass under limited visibility (Flocchini et al. 2005), lights for two robots (Das et al. 2016; Flocchini et al. 2016a; Heriban et al. 2018), and how many faulty robots can be tolerated (Agmon and Peleg 2006; Pattanayak et al. 2019b). Running time (rounds, epochs) became a measure mainly for limited visibility (Degener et al. 2011; Castenow et al. 2022).

Further reading

Papers on this problem that are not yet placed in a setting.

- Shukla et al. (2026b). Gathering under the adversarial (n, K)-defected view model of Kim, Shibata, Sudo, Nakamura, Katayama and Masuzawa (OPODIS 2022, not in the bibliography): in each Look the adversary shows a robot only K of the occupied points other than its own (all of them if there are at most K), and may change its choice at every Look. Oblivious robots without multiplicity detection and with non-rigid moves gather in finite time for $n = 4$ and $K = 2$ in FSYNC without any agreement on coordinates (Theorem 4.1), and

for any $n \geq 3$ and $1 \leq K \leq n-2$ in ASYNC if they agree on one axis (Theorem 6.1); no time bound. The plane model has no value for such defective views, so the results have no setting here. A preprint; several proofs are sketched, and some lemma references do not match the lemmas. Read.

- Di Luna et al. (2025c). Robots confined to a circle, with chirality, rigid moves, SSYNC, and visibility limited to angular distance below ϑ, gather from every rotationally asymmetric configuration when $\vartheta = \pi$ (Theorem 4.14); for $\vartheta \leq \pi/2$ no algorithm gathers, even from asymmetric configurations with a connected visibility graph (Theorem 3.7), by a new random-perturbation argument. Read (arXiv v3).
- Ghosh et al. (2025). The same circle with $\vartheta = \pi$ under ASYNC, for robots with finite persistent memory (FSTA): gathering from every rotationally asymmetric configuration (Theorem 2). Read (arXiv v1).
- Flocchini et al. (2019a). Robots on a continuous cycle of length ℓ that see only their own position, move at unit speed and act in synchronous rounds: randomized gathering and leader election by Las Vegas algorithms when n or ℓ is known (Theorems 1 and 9) and Monte Carlo ones with upper bounds (Theorems 8 and 16); no Las Vegas algorithm exists when neither is known exactly (Theorem 19). Read (ISAAC version).
- Heriban and Tixeuil (2021). Quantitative results by simulation: the largest distance travelled when gathering two or n oblivious robots under FSYNC, SSYNC and ASYNC, the behaviour of known rendezvous and leader-election algorithms when vision is unreliable, and two algorithms (fuel-efficient convergence, leader election) that tolerate such errors. Read (arXiv v1).
- Courtieu et al. (2016). Universal gathering certified in Coq: oblivious SSYNC robots with rigid moves and strong multiplicity detection, without chirality, gather from every configuration that is not bivalent, including starts with several robots on one point. Read (arXiv v1).
- Castenow et al. (2021). A closed chain of n disoriented robots with limited visibility, each seeing its chain neighbours, gathers in $\Theta(n)$ FSYNC rounds with lights, against $\Theta(n^2)$ for the oblivious algorithms; configurations that are isogonal polygons are gathered without lights. Read (arXiv v2).
- Braun et al. (2020). Gathering with limited visibility in three dimensions: a 3D Go-To-The-Center strategy takes $\Theta(n^2)$ FSYNC rounds, and every contracting strategy in continuous time takes $O(\Delta n^{3/2})$ for initial diameter Δ, against $\Omega(\Delta n)$. Read (arXiv v1).
- Pattanayak et al. (2021a). Randomized gathering of two robots under ASYNC, by what the adversary controls (waiting time, computation delay, speed) and knows: possible against an oblivious adversary with zero computation delay or with waiting time plus delay bounded below, impossible against an adaptive one; extended to many robots with merging. Read (arXiv v1).
- Kim et al. (2022). Defected views: each robot sees only some of the others. Gathering for $N \geq 5$ robots when each sees at most $N-2$ chosen by the adversary, and for four robots that each see their two nearest; impossible when each of three robots sees only one. Read (LIPIcs version).
- Czyzowicz et al. (2020e). A central authority that knows the positions of all robots, among them Byzantine spies it cannot identify, plans trajectories so that the reliable robots gather fast: optimal competitive ratio with at most one spy, small constants when fewer than a half or a third are spies. Read (arXiv v1).
- Barel et al. (2021). Agents with simple sensors pick random headings and move only if nobody is behind them: gathering in finite expected time is proved for continuous motion with a blind zone, and observed empirically for unit jumps. Read (arXiv v2).
- Di Luna et al. (2020a). Meeting in an unknown polygon, possibly with holes, whose edges block movement and vision: $\sigma + 1$ searchers suffice and are needed when the polygon has rotational symmetry of order σ; two suffice when the barycentre is not in a hole; also for oblivious robots. Abstract read.
- Bramas et al. (2020a). Two oblivious robots in the plane must meet even if one crashes, the correct one then going to the crashed one's position (stand-up indulgent rendezvous): solvability characterised by scheduler, with algorithms. Abstract read.
- Sharma et al. (2025b). Gathering on a circle by robots with limited angular visibility that agree on the clockwise direction, asynchronous, using finite communication (lights). Abstract read.
- Aponte et al. (2026). A formal model of ASYNC executions for the Pactole library of the Rocq (Coq) prover, and a certified gathering algorithm based on Weber points: oblivious disoriented robots with non-rigid moves and

Claude (Anthropic) was used in cataloguing the problems.

strong multiplicity detection gather from every configuration that is not bivalent, starts where robots share points included. (Cieliebak et al. 2012) use Weber points only where they are unique; the authors stress that Weber points do not make gathering trivial. Read in full.

- Li et al. (2021a). Gathering in continuous time (robots adjust their unit-speed motion to what they see at every instant) with visibility radius V and a connected start: every algorithm whose target points satisfy a simple contraction criterion gathers in $O(n\Delta_0)$ time (Theorem 1, their d is Δ_0), so $O(n^2)$; two such algorithms move to the centre of the smallest circle around the relative-neighbourhood neighbours (Theorem 2), and a variant gathers without collisions in $O(n^2)$ time using lights and chirality (Theorems 3, 4). Read in full.

Relations to other problems

- **Special case of k-grouping.** Gathering is k-grouping with $k = 1$, a single group of all n robots, that is, point formation: the pattern consisting of one point of multiplicity n (Flocchini et al. 2017, 2026). General pattern formation algorithms usually exclude it, because multiplicity points need detection (Bramas and Tixeuil 2016).
- **Derived problem: Near-gathering** (relaxation of). Gathering requires a common point; near-gathering only requires being close, but forbids collisions. It is used as a first phase before forming patterns.
- **Related: Convergence.** Every gathering algorithm solves convergence, but not conversely: two oblivious robots in SSYNC cannot gather, while "two oblivious robots can converge to a point with a naive algorithm" (Izumi et al. 2012). In ASYNC the centre-of-gravity algorithm converges for any n (Cohen and Peleg 2005), whereas gathering needs extra assumptions such as multiplicity detection.
- **Related: Scattering.** Scattering is the opposite task: gathering creates a multiplicity point, scattering removes all of them. Pattern formation from configurations with multiplicities needs scattering first (Bramas and Tixeuil 2016).
- **Related: Gathering by oblivious robots on graphs.** The same robots moving freely in the plane. On a graph a robot moves one edge per cycle, and the symmetries of the graph add to those of the configuration.

Assumptions specific to this problem

Number of robots

Suzuki and Yamashita (Suzuki and Yamashita 1999) showed that two oblivious robots cannot gather deterministically in SSYNC, although the task is trivial in FSYNC; rendezvous thus became the first separator between the two schedulers (Flocchini et al. 2023; Heriban et al. 2018). For $n \geq 3$ robots the situation differs: with multiplicity detection, oblivious disoriented robots gather from any configuration even in ASYNC (Cieliebak et al. 2012), because a multiplicity point, once formed, can be kept unique and serve as the target (Bandettini et al. 2011). As Flocchini, Santoro, Viglietta and Yamashita put it, "Rendezvous is very different from the Gathering problem of having $k \geqslant 3$ robots meet" (Flocchini et al. 2016a) (their k is n here). Two robots cannot create such an asymmetry, so the two-robot case became the test bed for each new capability: unreliable compasses (Izumi et al. 2012), persistent memory (Suzuki and Yamashita 1999), lights (Viglietta 2013; Heriban et al. 2018; Okumura et al. 2023), and finite-state or finite-communication robots (Flocchini et al. 2016a). The catalogue therefore splits gathering by team size: any $n \geq 2$, exactly two (rendezvous), and $n \geq 3$.

Claude (Anthropic) was used in cataloguing the problems.

Level	Value	Formal definition
0	**any** n	Any number $n \geq 2$ of robots.
1	$n = 2$	Exactly two robots (rendezvous).
1	$n \geq 3$	$n \geq 3$ robots.

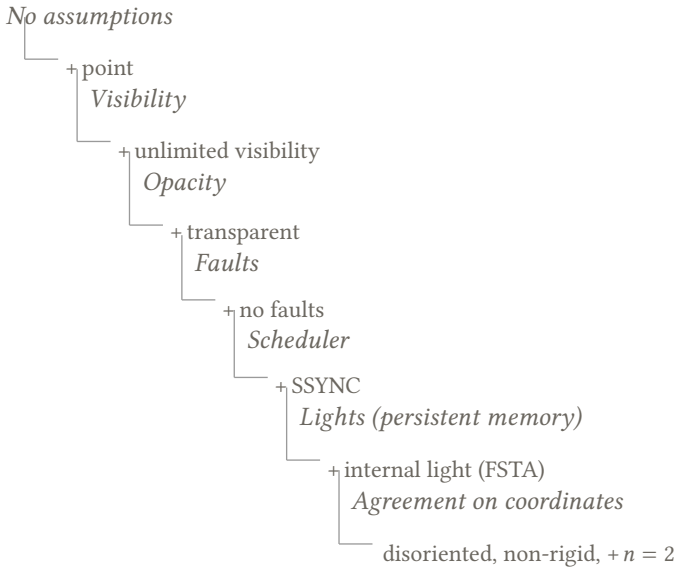
Initial configurations

Gathering algorithms often state which initial configurations they handle, and the families differ from paper to paper: Pattanayak, Mondal, Ramesh and Mandal list them in their Table 1 (Pattanayak et al. 2019b). Some restrictions are forced: with weak multiplicity detection and crashes, no wait-free algorithm gathers from configurations with more than one multiplicity (Theorem 1 of Pattanayak et al. 2019b), and bivalent configurations cannot be gathered (Bouzid et al. 2013). Others leave out solvable symmetric configurations, as the ASYNC algorithm of (Pattanayak et al. 2019b) does; Cicerone, Di Stefano and Navarra point to symmetric families that such case analyses can miss (Cicerone et al. 2021a). This axis separates results for every solvable start from results for a restricted family.

Level	Value	Formal definition
0	**any start**	The algorithm must gather from every initial configuration from which gathering is possible at all in the setting; configurations proved unsolvable there (for example bivalent ones, or several multiplicities with weak detection and crashes) are excluded by the problem, not by the algorithm.
1	**restricted start**	The algorithm gathers only from a stated family of initial configurations that leaves out some solvable ones, typically symmetric configurations; the family is named on each result.

Case tree

The root assumes nothing; going down, each edge adds the assumptions written in grey (+) or leaves an axis unassumed. Bold entries are catalogued settings with their status per measure (Time, Colours) and page; italic entries are branch points, named after the axis on which their branches differ. Axis order: Body → Visibility → Opacity → Faults → Scheduler → Lights (persistent memory) → Agreement on coordinates → Movement → Multiplicity detection → Randomness → Identifiers → Number of robots → Initial configurations.



Rendezvous, internal light (FSTA), SSYNC, non-rigid ●● p. 439

Claude (Anthropic) was used in cataloguing the problems.

- disoriented, + rigid, + $n = 2$
  Rendezvous, internal light (FSTA), SSYNC, rigid ●● p. 439
- + chirality, + $n \geq 3$
  Gathering, internal light (FSTA), SSYNC, non-rigid ●● p. 440

+ external light (FCOM)
Agreement on coordinates

- disoriented, non-rigid, + $n = 2$
  Rendezvous, external light (FCOM), SSYNC ●● p. 441
- + chirality, + rigid, + $n \geq 3$
  Gathering, external light (FCOM), SSYNC, rigid ●● p. 441

+ lights (LUMI)
Agreement on coordinates

- + chirality, + $n \geq 3$
  Gathering, full lights, SSYNC, no multiplicity detection ●● p. 442
- disoriented, + rigid, + strong multiplicity
  Self-stabilizing gathering, full lights, SSYNC, strong multiplicity detection ●● p. 443

oblivious, anonymous, + $n \geq 3$
No multiplicity detection, SSYNC ●● p. 444

oblivious, anonymous, + $n = 2$
Two robots, SSYNC ●● p. 444

oblivious, + homonymous
Homonymous robots, SSYNC ●● p. 445

ASYNC, + lights (LUMI)
Agreement on coordinates

- disoriented, + $n = 2$
  Rendezvous, full lights (LUMI), ASYNC, non-rigid ●● p. 445
- + chirality, + $n \geq 3$
  Gathering, full lights, ASYNC, no multiplicity detection ●● p. 446

ASYNC, + external light (FCOM)
Movement

- non-rigid, + $n = 2$
  Rendezvous, external light (FCOM), ASYNC, non-rigid ●● p. 447
- + rigid, + $n = 2$
  Rendezvous, external light (FCOM), ASYNC, rigid ●● p. 448

ASYNC, oblivious, + weak multiplicity, + $n \geq 3$
ASYNC with weak multiplicity detection STANDARD SETTING ●● p. 448

+ FSYNC
FSYNC, no multiplicity detection ●● p. 449

ASYNC, + internal light (FSTA), + $n = 2$
Rendezvous, internal light (FSTA), ASYNC ●● p. 450

+ sequential, + external light (FCOM), + $n \geq 3$
Gathering, external light, sequential scheduler ●● p. 450

Claude (Anthropic) was used in cataloguing the problems.

+ crash faults
Scheduler
+ SSYNC
Lights (persistent memory)
oblivious, disoriented, non-rigid, + weak multiplicity, + $n \geq 3$
Crash faults, SSYNC ●● p. 451
+ lights (LUMI), + compass, + rigid, + $n = 2$
Rendezvous, full lights, a crash, SSYNC ●● p. 451
ASYNC, + weak multiplicity, + $n \geq 3$, + restricted start
Crash faults, ASYNC, restricted start ●● p. 452
Byzantine, + SSYNC, + $n \geq 3$
Byzantine faults, SSYNC ●● p. 453
Byzantine, + FSYNC, + $n \geq 3$
Byzantine faults, FSYNC ●● p. 453
opaque, + crash faults, + one axis
Crash faults, one common axis, opaque robots ●● p. 454
limited visibility, + transparent, + no faults
Scheduler
ASYNC, + compass
Limited visibility with a compass, ASYNC ●● p. 454
+ rigid
Limited visibility with a compass, rigid, ASYNC ●● p. 455
ASYNC, + one axis, + rigid
Limited visibility, one common axis, rigid, ASYNC ●● p. 456
+ FSYNC
Limited visibility, FSYNC, disoriented ●● p. 457
fat, + unlimited visibility
Opacity
opaque, + no faults
Agreement on coordinates
disoriented, + $n \geq 3$
Fat robots, few robots, no common orientation ●● p. 457
+ chirality, + $n \geq 3$
Fat robots, chirality, any number ●● p. 458
+ transparent, + no faults, + $n \geq 3$
Transparent fat robots, no chirality ●● p. 459

What is known

The best upper bound against the best lower bound in each catalogued setting, in tree order. The settings follow in detail.

Setting	Time	Colours	p.
Rendezvous, internal light (FSTA), SSYNC, non-rigid	● unknown *finite* vs —	● tight 3 vs 3	439

Claude (Anthropic) was used in cataloguing the problems.

Setting	Time	Colours	p.
Rendezvous, internal light (FSTA), SSYNC, rigid	● unknown *finite* vs —	● tight 3 vs 3	439
Gathering, internal light (FSTA), SSYNC, non-rigid	● unknown *finite* vs —	● unknown 2 vs 3	440
Rendezvous, external light (FCOM), SSYNC	● unknown *finite* vs —	● tight 3 vs 3	441
Gathering, external light (FCOM), SSYNC, rigid	● unknown *finite* vs —	● tight 3 vs 3	441
Gathering, full lights, SSYNC, no multiplicity detection	● unknown *finite* vs —	● tight 2 vs 2	442
Self-stabilizing gathering, full lights, SSYNC, strong multiplicity detection	● unknown *finite* vs —	● tight 2 vs 2	443
No multiplicity detection, SSYNC	● impossible	● impossible	444
Two robots, SSYNC	● impossible	● impossible	444
Homonymous robots, SSYNC	● unknown *finite* vs *impossible*	● tight 0 vs —	445
Rendezvous, full lights (LUMI), ASYNC, non-rigid	● unknown *finite* vs —	● tight 2 vs 2	445
Gathering, full lights, ASYNC, no multiplicity detection	● unknown *finite* vs —	● tight 3 vs 2	446
Rendezvous, external light (FCOM), ASYNC, non-rigid	● unknown *finite* vs —	● tight 3 vs 3	447
Rendezvous, external light (FCOM), ASYNC, rigid	● unknown *finite* vs —	● tight 6 vs 3	448
ASYNC with weak multiplicity detection STANDARD SETTING	● unknown *finite* vs —	● tight 0 vs —	448
FSYNC, no multiplicity detection	● unknown *finite* vs —	● tight 0 vs —	449
Rendezvous, internal light (FSTA), ASYNC	● unknown — vs —	● unknown — vs 3	450
Gathering, external light, sequential scheduler	● unknown *finite* vs —	● tight 2 vs 2	450
Crash faults, SSYNC	● unknown *finite* vs *impossible*	● tight 0 vs —	451
Rendezvous, full lights, a crash, SSYNC	● impossible	● impossible	451
Crash faults, ASYNC, restricted start	● unknown *finite* vs —	● tight 0 vs —	452
Byzantine faults, SSYNC	● impossible	● impossible	453
Byzantine faults, FSYNC	● unknown *finite* vs —	● tight 0 vs —	453
Crash faults, one common axis, opaque robots	● unknown *finite* vs —	● tight 0 vs —	454
Limited visibility with a compass, ASYNC	● unknown *finite* vs *impossible*	● tight 0 vs —	454
Limited visibility with a compass, rigid, ASYNC	● tight $O(\Delta_0)$ vs $\Omega(\Delta_0)$	● tight 0 vs —	455
Limited visibility, one common axis, rigid, ASYNC	● unknown $O(\Delta_0)$ vs $\Omega(\Delta_0)$	● tight 0 vs —	456

Claude (Anthropic) was used in cataloguing the problems.

Setting	Time	Colours	p.
Limited visibility, FSYNC, disoriented	● tight $O(\Delta_0^2)$ vs $\Omega(\Delta_0^2)$	● tight 0 vs —	457
Fat robots, few robots, no common orientation	● unknown *finite* vs —	● tight 0 vs —	457
Fat robots, chirality, any number	● unknown *finite* vs —	● tight 0 vs —	458
Transparent fat robots, no chirality	● unknown *finite* vs —	● tight 0 vs —	459

18.8.1 Rendezvous, internal light (FSTA), SSYNC, non-rigid

Adds disoriented, non-rigid, + $n = 2$ VERIFIED

A light only its owner sees, that is constant memory, lets two robots meet in SSYNC with 3 colours if they know δ; 2 colours are impossible.

Assumed: body: point robots; visibility: unlimited; opacity: transparent: every robot sees every robot within range; faults: no faults; scheduler: semi-synchronous (SSYNC); lights (persistent memory): internal light only (FSTA): a robot sees its own light, not the others'; number of robots: $n = 2$ (rendezvous). **Not assumed:** agreement on coordinates: no agreement on local coordinate systems; movement: non-rigid: the adversary may stop a move after some minimum distance δ; multiplicity detection: none: a point with several robots looks like one robot; randomness: deterministic; identifiers: anonymous: all robots are identical; initial configurations: every solvable initial configuration.

Measure	Lower bound	Best upper bound	Status
Time	none known Time is not analysed; only solvability.	*finite* (Flocchini et al. 2016a)	● unknown
Colours	3 (Otaka et al. 2026) Exactly 3 colours when δ is known.	3 (Flocchini et al. 2016a)	● tight

Results.

- Otaka et al. (2026) [lower bound]: colours 3. *Bottleneck:* memory. *Holds when:* Theorem 1: 2 colours are impossible, for every $n \geq 2$; holds even with rigid moves, chirality and a common unit.
- Flocchini et al. (2016a) [current best]: time *finite*, colours 3. *Assumes:* the robots know the minimum move length δ. Introductory and current best (Theorem 18).

Open questions.

Q1. Rendezvous with internal lights in non-rigid SSYNC when δ is unknown.

Notes. No self-stabilizing algorithm with internal lights exists, for any number of colours (Otaka et al. 2026). No colour-only (class L) algorithm works with internal lights, whatever the memory (Theorem 1 of Flocchini et al. 2016a). If the adversary may rescale a robot's frame at every Look, rendezvous with internal lights is impossible for any number of colours (Theorem 1(i) of Wada et al. 2026). Checked against the full texts.

18.8.2 Rendezvous, internal light (FSTA), SSYNC, rigid

Adds disoriented, + rigid, + $n = 2$ VERIFIED

With rigid moves, robots with internal lights meet in SSYNC with exactly 3 colours.

Claude (Anthropic) was used in cataloguing the problems.

Assumed: body: point robots; visibility: unlimited; opacity: transparent: every robot sees every robot within range; faults: no faults; scheduler: semi-synchronous (SSYNC); lights (persistent memory): internal light only (FSTA): a robot sees its own light, not the others'; movement: rigid: every move reaches its destination; number of robots: $n = 2$ (rendezvous). **Not assumed:** agreement on coordinates: no agreement on local coordinate systems; multiplicity detection: none: a point with several robots looks like one robot; randomness: deterministic; identifiers: anonymous: all robots are identical; initial configurations: every solvable initial configuration.

Measure	Lower bound	Best upper bound	Status
Time	none known Time is not analysed; only solvability.	*finite* (Wada et al. 2026) [preprint]	● unknown
Colours	3 (Otaka et al. 2026) Exactly 3 colours.	3 (Wada et al. 2026) [preprint]	● tight

Results.

- Otaka et al. (2026) [lower bound]: colours 3. *Bottleneck:* memory. *Holds when:* Theorem 1: 2 colours are impossible.
- Flocchini et al. (2016a) [introductory]: time *finite*, colours 6. Theorem 13, from a preset state; each robot uses its own unit of distance.
- Wada et al. (2026) [preprint] [current best]: time *finite*, colours 3. *Assumes:* each robot keeps a fixed unit of distance, while its rotation and reflection may change at every Look. Theorem 8, from a designated colour. A preprint, not peer reviewed.

Notes. Checked against the full texts.

18.8.3 Gathering, internal light (FSTA), SSYNC, non-rigid

Adds + chirality, + $n \geq 3$ VERIFIED

With internal lights, 2 colours gather $n \geq 3$ robots in SSYNC only from restricted starts; 2 colours do not suffice in general, and whether more do is open.

Assumed: body: point robots; visibility: unlimited; opacity: transparent: every robot sees every robot within range; faults: no faults; scheduler: semi-synchronous (SSYNC); lights (persistent memory): internal light only (FSTA): a robot sees its own light, not the others'; agreement on coordinates: common chirality (same clockwise orientation); number of robots: $n \geq 3$. **Not assumed:** movement: non-rigid: the adversary may stop a move after some minimum distance δ; multiplicity detection: none: a point with several robots looks like one robot; randomness: deterministic; identifiers: anonymous: all robots are identical; initial configurations: every solvable initial configuration.

Measure	Lower bound	Best upper bound	Status
Time	none known Time is not analysed; only solvability.	*finite* (Otaka et al. 2026)	● unknown
Colours	3 (Otaka et al. 2026) Unconditional gathering with more colours is open.	2 (Otaka et al. 2026)	● unknown

Results.

- Otaka et al. (2026) [lower bound]: colours 3. *Bottleneck:* memory. *Holds when:* Theorem 1: 2 colours are impossible without restrictions on the start.
- Terai et al. (2023) [introductory]: time *finite*, colours 2. *Assumes:* δ known and common; restricted initial configurations. Theorem 8, from starts in which robots are at most 2δ apart.
- Otaka et al. (2026) [current best]: time *finite*, colours 2. *Assumes:* δ known and common; some two-point starts excluded. Theorem 3: far fewer excluded starts; the remaining exclusions cannot all be removed (Theorem 2(3)).

Claude (Anthropic) was used in cataloguing the problems.

Open questions.

Q1. Gathering with internal lights in SSYNC from every start, with more than 2 colours, and in ASYNC (Otaka et al. 2026).

Notes. Checked against the full texts.

18.8.4 Rendezvous, external light (FCOM), SSYNC

Adds disoriented, non-rigid, $+n = 2$ VERIFIED

With a light only the other robot sees, two robots meet in SSYNC with 3 colours, and 2 colours are impossible.

Assumed: body: point robots; visibility: unlimited; opacity: transparent: every robot sees every robot within range; faults: no faults; scheduler: semi-synchronous (SSYNC); lights (persistent memory): external light only (FCOM): a robot sees the others' lights, not its own; number of robots: $n = 2$ (rendezvous). **Not assumed:** agreement on coordinates: no agreement on local coordinate systems; movement: non-rigid: the adversary may stop a move after some minimum distance δ; multiplicity detection: none: a point with several robots looks like one robot; randomness: deterministic; identifiers: anonymous: all robots are identical; initial configurations: every solvable initial configuration.

Measure	Lower bound	Best upper bound	Status
Time	none known Time is not analysed; only solvability.	*finite* (Flocchini et al. 2016a)	● unknown
Colours	3 (Otaka et al. 2026); 3 (Okumura et al. 2023) Exactly 3 colours.	3 (Flocchini et al. 2016a)	● tight

Results.

- Otaka et al. (2026) [lower bound]: colours 3. *Bottleneck:* memory. *Holds when:* Theorem 1: 2 colours are impossible for external lights, for every $n \geq 2$; holds even with rigid moves, chirality and a common unit.
- Okumura et al. (2023) [lower bound]: colours 3. *Bottleneck:* memory. *Holds when:* Theorem 8: for colour-only (class L) algorithms, rigid SSYNC.
- Flocchini et al. (2016a) [current best]: time *finite*, colours 3. *Assumes:* class L: the destination is a point on the segment to the other robot and depends only on the colours. Introductory and current best (Theorem 16): self-stabilizing, non-rigid moves, no common unit.

Notes. Checked against the full texts.

18.8.5 Gathering, external light (FCOM), SSYNC, rigid

Adds + chirality, + rigid, $+n \geq 3$ VERIFIED

With external lights, $n \geq 3$ robots gather in rigid SSYNC with 3 colours, which is optimal; 2 suffice if a robot sees who shares its point.

Assumed: body: point robots; visibility: unlimited; opacity: transparent: every robot sees every robot within range; faults: no faults; scheduler: semi-synchronous (SSYNC); lights (persistent memory): external light only (FCOM): a robot sees the others' lights, not its own; agreement on coordinates: common chirality (same clockwise orientation); movement: rigid: every move reaches its destination; number of robots: $n \geq 3$. **Not assumed:** multiplicity detection: none: a point with several robots looks like one robot; randomness: deterministic; identifiers: anonymous: all robots are identical; initial configurations: every solvable initial configuration.

Claude (Anthropic) was used in cataloguing the problems.

Measure	Lower bound	Best upper bound	Status
Time	none known Time is not analysed; only solvability.	*finite* (Terai et al. 2023)	● unknown
Colours	3 (Otaka et al. 2026) Exactly 3 colours.	3 (Terai et al. 2023)	● tight

Results.

- Otaka et al. (2026) [lower bound]: colours 3. *Bottleneck:* memory. *Holds when:* Theorem 2(1); without seeing the robots at its own point.
- Terai et al. (2023) [current best]: time *finite*, colours 3. Introductory and current best (Theorem 6). With local awareness 2 colours suffice (Theorem 7), and local awareness cannot be dropped (Theorem 2(2) of Otaka et al. 2026).

Open questions.

Q1. Gathering with external lights in ASYNC (Otaka et al. 2026).

Notes. Checked against the full texts.

18.8.6 Gathering, full lights, SSYNC, no multiplicity detection

Adds + chirality, + $n \geq 3$ VERIFIED

With lights and chirality, $n \geq 3$ robots that cannot detect multiplicities gather in SSYNC with 2 colours, which is optimal.

Assumed: body: point robots; visibility: unlimited; opacity: transparent: every robot sees every robot within range; faults: no faults; scheduler: semi-synchronous (SSYNC); lights (persistent memory): full light (LUMI): every robot sees all lights, including its own; agreement on coordinates: common chirality (same clockwise orientation); number of robots: $n \geq 3$. **Not assumed:** movement: non-rigid: the adversary may stop a move after some minimum distance δ; multiplicity detection: none: a point with several robots looks like one robot; randomness: deterministic; identifiers: anonymous: all robots are identical; initial configurations: every solvable initial configuration.

Measure	Lower bound	Best upper bound	Status
Time	none known Time is not analysed; only solvability.	*finite* (Terai et al. 2023)	● unknown
Colours	2 (Prencipe 2007) Exactly 2 colours.	2 (Terai et al. 2023)	● tight

Results.

- Prencipe (2007) [lower bound]: colours 2. *Bottleneck:* indistinguishable multiplicities. *Holds when:* oblivious robots without multiplicity detection cannot gather in SSYNC (one colour).
- Terai et al. (2023) [current best]: time *finite*, colours 2. Introductory and current best (Theorem 5); non-rigid moves; a robot sees the set of colours at a point, not the number of robots.

Open questions.

Q1. Gathering with lights without chirality (Terai et al. 2023).

Notes. Checked against the full text.

Claude (Anthropic) was used in cataloguing the problems.

18.8.7 Self-stabilizing gathering, full lights, SSYNC, strong multiplicity detection

Adds disoriented, + rigid, + strong multiplicity VERIFIED

With 2 colours and strong multiplicity detection, any number of robots gather in SSYNC with rigid moves from any configuration and any initial colours; one colour (oblivious robots) does not suffice.

Assumed: body: point robots; visibility: unlimited; opacity: transparent: every robot sees every robot within range; faults: no faults; scheduler: semi-synchronous (SSYNC); lights (persistent memory): full light (LUMI): every robot sees all lights, including its own; movement: rigid: every move reaches its destination; multiplicity detection: strong: a robot sees how many robots are at each point. **Not assumed:** agreement on coordinates: no agreement on local coordinate systems; randomness: deterministic; identifiers: anonymous: all robots are identical; number of robots: any n; initial configurations: every solvable initial configuration.

Statement for this setting. The robots start from an arbitrary configuration with arbitrary colours: several robots may share a point (a multiplicity), and the configuration may be bivalent, with $n/2$ robots on each of two points. An algorithm that gathers from every such start is self-stabilizing (Bonnet et al. 2026a).

Measure	Lower bound	Best upper bound	Status
Time	none known Time is not analysed; only solvability.	*finite* (Bonnet et al. 2026a)	● unknown
Colours	2 (Suzuki and Yamashita 1999) Exactly 2 colours.	2 (Bonnet et al. 2026a)	● tight

Results.

- Suzuki and Yamashita (1999) [lower bound]: colours 2. *Bottleneck:* symmetry. *Holds when:* $n = 2$: two oblivious robots at distinct points cannot gather in SSYNC (Theorem 3.1 of Suzuki and Yamashita 1999), already with rigid moves and common handedness; with two robots, multiplicity detection gives no information; for every even n from a bivalent configuration, oblivious robots cannot gather in SSYNC, by a proof certified in Coq by Courtieu, Rieg, Tixeuil and Urbain (IPL 2015), as reported by (Bonnet et al. 2026a).
- Bonnet et al. (2026a) [current best]: time *finite*, colours 2. Introductory and current best (the final theorem of the Coq development, Section 4.4 of the SIROCCO 2025 version): any n, deterministic, no chirality, any initial positions and colours; the proof is certified in the Coq proof assistant. The authors call it the first fully self-stabilizing deterministic uniform gathering algorithm in SSYNC.

Open questions.

Q1. Does self-stabilizing gathering with 2 colours extend to ASYNC (Bonnet et al. 2026a)?

Q2. Is self-stabilizing SSYNC gathering with lights solvable with weak or no multiplicity detection, or with non-rigid moves? The algorithm of (Bonnet et al. 2026a) uses strong multiplicity detection and rigid moves.

Notes. Bonnet, Bramas, Courtieu, Défago, Rieg, Tixeuil and Urbain call a bivalent configuration colour-bivalent when each of its two points holds as many White as Black robots. There, an activated White robot moves to the midpoint and turns Black, and an activated Black robot turns White without moving. In a bivalent configuration that is not colour-bivalent, every activated robot moves to the point with more Black robots. In any other configuration the robots run the oblivious SSYNC algorithm of Courtieu, Rieg, Tixeuil and Urbain (DISC 2016), which never creates a bivalent configuration; lights do not change there. The proof uses the Pactole library and a measure that decreases whenever a robot changes its state (Section 4.2). Earlier deterministic self-stabilizing SSYNC solutions for any n were for oblivious robots with odd n (Dieudonné and Petit), or used randomization (Défago et al. 2020) or two different algorithms (Asahiro and Yamashita 2025); the 2-colour SSYNC algorithm of (Terai et al. 2023) needs all

Claude (Anthropic) was used in cataloguing the problems.

robots to start with the same colour (all as reported by Bonnet et al. 2026a). Checked against the full text of the SIROCCO 2025 version (model in Section 2, final theorem in Section 4.4) and of (Suzuki and Yamashita 1999) (Theorem 3.1). The journal version presents two certified algorithms and was not read.

18.8.8 No multiplicity detection, SSYNC

Adds oblivious, anonymous, $+n \geq 3$ VERIFIED

Without multiplicity detection, deterministic oblivious robots cannot gather, for any $n \geq 2$, in SSYNC or ASYNC.

Assumed: body: point robots; visibility: unlimited; opacity: transparent: every robot sees every robot within range; faults: no faults; scheduler: semi-synchronous (SSYNC); number of robots: $n \geq 3$. **Not assumed:** lights (persistent memory): oblivious, no light (OBLOT); agreement on coordinates: no agreement on local coordinate systems; movement: non-rigid: the adversary may stop a move after some minimum distance δ; multiplicity detection: none: a point with several robots looks like one robot; randomness: deterministic; identifiers: anonymous: all robots are identical; initial configurations: every solvable initial configuration.

Measure	Lower bound	Best upper bound	Status
Time	*impossible* (Prencipe 2007) Impossible in general.	—	• impossible
Colours	none known Impossible for oblivious robots.	—	• impossible

Results.

- Prencipe (2007) [lower bound]: time *impossible*. *Bottleneck:* indistinguishable multiplicities. *Holds when:* Theorem 2: deterministic oblivious robots, no multiplicity detection; $n \geq 2$: Prencipe proves $n \geq 3$ (Lemmas 5 and 6) and takes $n = 2$ from Suzuki and Yamashita; SSYNC (Atom) and ASYNC (Corda); non-rigid moves, unlimited visibility, no common coordinate system.

Notes. Prencipe uses chirality to mean agreement on North and East (a compass), not handedness. (Agmon and Peleg 2006) attribute the necessity of multiplicity detection in SSYNC and ASYNC to Prencipe's thesis (2002). Checked against the SIROCCO 2005 version; the TCS 2007 version was not read.

18.8.9 Two robots, SSYNC

Adds oblivious, anonymous, $+n = 2$ VERIFIED

Two oblivious robots without a common orientation cannot meet in SSYNC: rendezvous needs extra power.

Assumed: body: point robots; visibility: unlimited; opacity: transparent: every robot sees every robot within range; faults: no faults; scheduler: semi-synchronous (SSYNC); number of robots: $n = 2$ (rendezvous). **Not assumed:** lights (persistent memory): oblivious, no light (OBLOT); agreement on coordinates: no agreement on local coordinate systems; movement: non-rigid: the adversary may stop a move after some minimum distance δ; multiplicity detection: none: a point with several robots looks like one robot; randomness: deterministic; identifiers: anonymous: all robots are identical; initial configurations: every solvable initial configuration.

Measure	Lower bound	Best upper bound	Status
Time	*impossible* (Flocchini et al. 2016a) Impossible.	—	• impossible
Colours	none known Oblivious robots cannot do it; see the luminous variant.	—	• impossible

Claude (Anthropic) was used in cataloguing the problems.

Results.

- Flocchini et al. (2016a) [lower bound]: time *impossible*. *Bottleneck:* symmetry. *Holds when:* $n = 2$, oblivious; without additional assumptions, already in SSYNC; first shown by (Suzuki and Yamashita 1999) (Theorem 3.1), in a model with common handedness and moves that reach their computed destination, so it holds without these.

Notes. Two robots with unreliable compasses can gather if the deviation is below $\pi/2$ (static compasses, SSYNC and ASYNC), $\pi/4$ (dynamic, SSYNC) or $\pi/6$ (dynamic, ASYNC, sufficient only) (Izumi et al. 2012).

18.8.10 Homonymous robots, SSYNC

Adds oblivious, + homonymous VERIFIED

Two distinct identifiers (two different algorithms) are necessary and sufficient for gathering in SSYNC.

Assumed: body: point robots; visibility: unlimited; opacity: transparent: every robot sees every robot within range; faults: no faults; scheduler: semi-synchronous (SSYNC); identifiers: homonymous: robots carry labels, but several may share a label. **Not assumed:** lights (persistent memory): oblivious, no light (OBLOT); agreement on coordinates: no agreement on local coordinate systems; movement: non-rigid: the adversary may stop a move after some minimum distance δ; multiplicity detection: none: a point with several robots looks like one robot; randomness: deterministic; number of robots: any n; initial configurations: every solvable initial configuration.

Measure	Lower bound	Best upper bound	Status
Time	*impossible* (Asahiro and Yamashita 2025) Time not analysed.	*finite* (Asahiro and Yamashita 2025)	● unknown
Colours	none known No colours.	0 (Asahiro and Yamashita 2025)	● tight

Results.

- Asahiro and Yamashita (2025) [lower bound]: time *impossible*. *Bottleneck:* symmetry. *Holds when:* with one identifier (anonymous robots); with one crash fault, for any number of identifiers.
- Asahiro and Yamashita (2025) [current best]: time *finite*, colours 0. *Assumes:* 2 identifiers. The minimum algorithm size of gathering is 2; gathering the non-faulty robots under crashes needs 3.

Open questions.

Q1. What is the minimum number of identifiers for Byzantine-tolerant gathering? (Asahiro and Yamashita 2025)

Notes. Homonymous robots were first studied for pattern formation (Bouzid and Lamani 2011); the term comes from Byzantine agreement with homonyms (Delporte-Gallet et al. 2013).

18.8.11 Rendezvous, full lights (LUMI), ASYNC, non-rigid

Adds disoriented, + $n = 2$ VERIFIED

Two robots whose lights both can see meet in ASYNC with non-rigid moves using 2 colours, which is optimal, from any initial colours.

Assumed: body: point robots; visibility: unlimited; opacity: transparent: every robot sees every robot within range; faults: no faults; lights (persistent memory): full light (LUMI): every robot sees all lights, including its own; number of robots: $n = 2$ (rendezvous). **Not assumed:** scheduler: asynchronous (ASYNC); agreement on coordinates: no agreement on local coordinate systems; movement: non-rigid: the adversary may stop a move after some minimum distance δ; multiplicity detection: none:

Claude (Anthropic) was used in cataloguing the problems.

a point with several robots looks like one robot; randomness: deterministic; identifiers: anonymous: all robots are identical; initial configurations: every solvable initial configuration.

Measure	Lower bound	Best upper bound	Status
Time	none known Time is not analysed; only solvability.	*finite* (Heriban et al. 2018)	● unknown
Colours	2 (Viglietta 2013) Exactly 2 colours.	2 (Heriban et al. 2018)	● tight

Results.

- Viglietta (2013) [lower bound]: colours 2. *Bottleneck:* symmetry breaking. *Holds when:* Proposition 1.1: one colour is an oblivious robot, which cannot meet in SSYNC already with rigid moves (Suzuki and Yamashita).
- Das et al. (2016) [introductory]: time *finite*, colours 4. Theorem 3.3; the authors call it self-stabilizing.
- Viglietta (2013) [also]: time *finite*, colours 3. *Assumes:* class L: the destination is a point on the segment to the other robot and depends only on the colours. Theorem 2.7: self-stabilizing, colours only; 3 colours are optimal among such algorithms (Theorems 4.10 and 4.11).
- Heriban et al. (2018) [current best]: time *finite*, colours 2. *Assumes:* a robot can tell whether the other robot is at its position. Theorem 1: self-stabilizing; a robot only tests whether the other robot is at its own position.

Notes. Easier settings were solved first with 2 colours: SSYNC with colours only, from any initial colours (Theorem 2.2 of Viglietta 2013); rigid ASYNC from a preset colour (Theorem 2.4 of Viglietta 2013, Theorem 3 of Okumura et al. 2017); ASYNC in which no robot looks between another's Look and Compute, or with δ known (Okumura et al. 2017). A model checker confirms the 2- and 3-colour ASYNC algorithms and finds Viglietta's 2-colour SSYNC algorithm incorrect in ASYNC (Défago et al. 2023). Checked against the full texts.

18.8.12 Gathering, full lights, ASYNC, no multiplicity detection

Adds + chirality, + $n \geq 3$ VERIFIED

With lights and chirality, $n \geq 3$ robots gather in ASYNC with non-rigid moves and no multiplicity detection using 3 colours; whether 2 suffice is open.

Assumed: body: point robots; visibility: unlimited; opacity: transparent: every robot sees every robot within range; faults: no faults; lights (persistent memory): full light (LUMI): every robot sees all lights, including its own; agreement on coordinates: common chirality (same clockwise orientation); number of robots: $n \geq 3$. **Not assumed:** scheduler: asynchronous (ASYNC); movement: non-rigid: the adversary may stop a move after some minimum distance δ; multiplicity detection: none: a point with several robots looks like one robot; randomness: deterministic; identifiers: anonymous: all robots are identical; initial configurations: every solvable initial configuration.

Measure	Lower bound	Best upper bound	Status
Time	none known Time is not analysed; only solvability.	*finite* (Nakai et al. 2021)	● unknown
Colours	2 (Prencipe 2007) $\Theta(1)$; the exact number is 2 or 3.	3 (Nakai et al. 2021)	● tight

Results.

- Prencipe (2007) [lower bound]: colours 2. *Bottleneck:* indistinguishable multiplicities. *Holds when:* inherited: one colour is an oblivious robot.
- Terai et al. (2023) [introductory]: time *finite*, colours 10. Concluding remarks: the SSYNC algorithm through the $5c$-colour simulation of a c-colour algorithm (Das et al. 2016).

Claude (Anthropic) was used in cataloguing the problems.

- Nakai et al. (2021) [current best]: time *finite*, colours 3. Theorem 6; 6 colours by a new $3c$-colour simulation (for a c-colour algorithm) of unfair SSYNC (Theorem 4). The authors conjecture that 3 is optimal.

Open questions.

Q1. Is 3 the minimum number of colours for ASYNC gathering with lights, and can chirality be dropped (Nakai et al. 2021)?

Q2. Self-stabilizing gathering of n robots with lights (Heriban et al. 2018).

Notes. In SSYNC with strong multiplicity detection and rigid moves, 2 colours suffice from any configuration and any initial colours, for any n, without chirality (setting *Self-stabilizing gathering, full lights, SSYNC, strong multiplicity detection*, Bonnet et al. 2026a). Checked against the full texts.

18.8.13 Rendezvous, external light (FCOM), ASYNC, non-rigid

Adds non-rigid, $+n = 2$ VERIFIED

In ASYNC with non-rigid moves, robots with external lights meet with 3 colours if they know the minimum move δ; without δ the question is open, and colour-only algorithms cannot work.

Assumed: body: point robots; visibility: unlimited; opacity: transparent: every robot sees every robot within range; faults: no faults; lights (persistent memory): external light only (FCOM): a robot sees the others' lights, not its own; number of robots: $n = 2$ (rendezvous). **Not assumed:** scheduler: asynchronous (ASYNC); agreement on coordinates: no agreement on local coordinate systems; movement: non-rigid: the adversary may stop a move after some minimum distance δ; multiplicity detection: none: a point with several robots looks like one robot; randomness: deterministic; identifiers: anonymous: all robots are identical; initial configurations: every solvable initial configuration.

Measure	Lower bound	Best upper bound	Status
Time	none known Time is not analysed; only solvability.	*finite* (Flocchini et al. 2016a)	● unknown
Colours	3 (Otaka et al. 2026) 3 colours when δ is known.	3 (Flocchini et al. 2016a)	● tight

Results.

- Otaka et al. (2026) [lower bound]: colours 3. *Bottleneck:* memory. *Holds when:* SSYNC bound (Theorem 1), hence also in ASYNC.

- Flocchini et al. (2016a) [current best]: time *finite*, colours 3. *Assumes:* the robots know the minimum move length δ. Introductory and current best (Theorem 19), from a preset colour.

Open questions.

Q1. Rendezvous with external lights in non-rigid ASYNC when δ is unknown (Flocchini et al. 2016a; Wada et al. 2026).

Notes. No colour-only (class L) algorithm with external lights works in ASYNC, for any number of colours (Theorem 14 of Flocchini et al. 2016a). If no robot looks between another robot's Look and Compute, colour-only algorithms with 3, 4 and 5 colours are optimal for rigid moves, non-rigid moves from equal colours, and self-stabilization respectively (Okumura et al. 2023). Checked against the full texts.

Claude (Anthropic) was used in cataloguing the problems.

18.8.14 Rendezvous, external light (FCOM), ASYNC, rigid

Adds + rigid, + $n = 2$ VERIFIED

With rigid moves, robots with external lights meet in ASYNC; the best count is 6 colours against a lower bound of 3.

Assumed: body: point robots; visibility: unlimited; opacity: transparent: every robot sees every robot within range; faults: no faults; lights (persistent memory): external light only (FCOM): a robot sees the others' lights, not its own; movement: rigid: every move reaches its destination; number of robots: $n = 2$ (rendezvous). **Not assumed:** scheduler: asynchronous (ASYNC); agreement on coordinates: no agreement on local coordinate systems; multiplicity detection: none: a point with several robots looks like one robot; randomness: deterministic; identifiers: anonymous: all robots are identical; initial configurations: every solvable initial configuration.

Measure	Lower bound	Best upper bound	Status
Time	none known Time is not analysed; only solvability.	*finite* (Wada et al. 2026) [preprint]	● unknown
Colours	3 (Otaka et al. 2026) $\Theta(1)$ colours; the exact number is between 3 and 6.	6 (Wada et al. 2026) [preprint]	● tight

Results.

- Otaka et al. (2026) [lower bound]: colours 3. *Bottleneck:* memory. *Holds when:* SSYNC bound (Theorem 1), hence also in ASYNC.
- Flocchini et al. (2016a) [introductory]: time *finite*, colours 12. Theorem 15, from a preset colour; each robot uses its own unit of distance.
- Wada et al. (2026) [preprint] [current best]: time *finite*, colours 6. *Assumes:* each robot keeps a fixed unit of distance, while its rotation and reflection may change at every Look. Theorem 10, from a designated colour. A preprint, not peer reviewed.

Open questions.

Q1. Close the gap between 3 and 6 colours (Wada et al. 2026).

Notes. If the adversary may also rescale a robot's frame at every Look, rendezvous with external lights is impossible in ASYNC for any number of colours (Theorem 1(ii) of Wada et al. 2026). Checked against the full texts.

18.8.15 ASYNC with weak multiplicity detection

Adds ASYNC, oblivious, + weak multiplicity, + $n \geq 3$ STANDARD SETTING VERIFIED

With weak multiplicity detection, $n \geq 3$ oblivious, disoriented, asynchronous robots gather from any initial configuration.

Assumed: body: point robots; visibility: unlimited; opacity: transparent: every robot sees every robot within range; faults: no faults; multiplicity detection: weak: a robot sees whether a point holds more than one robot; number of robots: $n \geq 3$. **Not assumed:** scheduler: asynchronous (ASYNC); lights (persistent memory): oblivious, no light (OBLOT); agreement on coordinates: no agreement on local coordinate systems; movement: non-rigid: the adversary may stop a move after some minimum distance δ; randomness: deterministic; identifiers: anonymous: all robots are identical; initial configurations: every solvable initial configuration.

Claude (Anthropic) was used in cataloguing the problems.

Measure	Lower bound	Best upper bound	Status
Time	none known No epoch bound is recorded.	*finite* (Cieliebak et al. 2012)	● unknown
Colours	none known No colours.	0 (Cieliebak et al. 2012)	● tight

Results.

- Cieliebak et al. (2012) [current best]: time *finite*, colours 0. Introductory and current best: solvable "for any $n > 2$ and any initial configuration", with non-rigid moves and weak multiplicity detection (a robot sees "none, one, or more than one" robot at a point). Theorem 5.36 covers $n \geq 5$; for $n = 3, 4$ the paper relies on earlier work of Cieliebak and Prencipe (SIROCCO 2002).

Open questions.

Q1. How many epochs does oblivious ASYNC gathering with weak multiplicity detection need?

Notes. Gathering is point formation, the pattern of a single point with multiplicity n (Flocchini et al. 2017). Under sequential schedulers, weak multiplicity detection is necessary and sufficient for gathering, and under FSYNC no multiplicity detection is needed (Flocchini et al. 2026).

18.8.16 FSYNC, no multiplicity detection

Adds + FSYNC

VERIFIED

Under FSYNC, oblivious disoriented robots gather from any start of distinct points without detecting multiplicities, even with non-rigid moves.

Assumed: body: point robots; visibility: unlimited; opacity: transparent: every robot sees every robot within range; faults: no faults; scheduler: fully synchronous (FSYNC). **Not assumed:** lights (persistent memory): oblivious, no light (OBLOT); agreement on coordinates: no agreement on local coordinate systems; movement: non-rigid: the adversary may stop a move after some minimum distance δ; multiplicity detection: none: a point with several robots looks like one robot; randomness: deterministic; identifiers: anonymous: all robots are identical; number of robots: any n; initial configurations: every solvable initial configuration.

Measure	Lower bound	Best upper bound	Status
Time	none known Only solvability is certified; time is not analysed.	*finite* (Balabonski et al. 2018b)	● unknown
Colours	none known No colours.	0 (Balabonski et al. 2018b)	● tight

Results.

- Balabonski et al. (2018b) [current best]: time *finite*, colours 0. Introductory and current best: a variant of the centre-of-gravity algorithm of Cohen and Peleg, which needed multiplicity detection because non-rigid moves can make robots meet early; the proof follows theirs and is certified in the Coq proof assistant (Pactole library). The question was open before.

Notes. Without multiplicity detection gathering is impossible under SSYNC and ASYNC (setting *no-multiplicity*), so FSYNC is the only scheduler of the model in which it is solvable without extra assumptions (Balabonski et al. 2018b). (Flocchini et al. 2026) also state that under FSYNC no multiplicity detection is needed. Checked against the HAL version of (Balabonski et al. 2018b).

Claude (Anthropic) was used in cataloguing the problems.

18.8.17 Rendezvous, internal light (FSTA), ASYNC

Adds ASYNC, + internal light (FSTA), + $n = 2$ VERIFIED

Whether robots with internal lights can meet in ASYNC is open.

Assumed: body: point robots; visibility: unlimited; opacity: transparent: every robot sees every robot within range; faults: no faults; lights (persistent memory): internal light only (FSTA): a robot sees its own light, not the others'; number of robots: $n = 2$ (rendezvous). **Not assumed:** scheduler: asynchronous (ASYNC); agreement on coordinates: no agreement on local coordinate systems; movement: non-rigid: the adversary may stop a move after some minimum distance δ; multiplicity detection: none: a point with several robots looks like one robot; randomness: deterministic; identifiers: anonymous: all robots are identical; initial configurations: every solvable initial configuration.

Measure	Lower bound	Best upper bound	Status
Time	none known Open.	—	● unknown
Colours	3 (Otaka et al. 2026) Open.	—	● unknown

Results.

- Otaka et al. (2026) [lower bound]: colours 3. *Bottleneck:* memory. *Holds when:* SSYNC bound (Theorem 1), hence also in ASYNC.

Open questions.

Q1. Can robots with internal lights solve rendezvous in ASYNC (Flocchini et al. 2016a; Wada et al. 2026)?

Notes. If no robot looks while another moves (Move-atomic ASYNC), 4 colours suffice with rigid moves (Theorem 9 of Wada et al. 2026, a preprint). Checked against the full texts.

18.8.18 Gathering, external light, sequential scheduler

Adds + sequential, + external light (FCOM), + $n \geq 3$ VERIFIED

If one robot moves per round, 2 external colours gather $n \geq 3$ robots, self-stabilizing, without chirality; oblivious robots cannot.

Assumed: body: point robots; visibility: unlimited; opacity: transparent: every robot sees every robot within range; faults: no faults; scheduler: sequential: exactly one robot is activated per round; lights (persistent memory): external light only (FCOM): a robot sees the others' lights, not its own; number of robots: $n \geq 3$. **Not assumed:** agreement on coordinates: no agreement on local coordinate systems; movement: non-rigid: the adversary may stop a move after some minimum distance δ; multiplicity detection: none: a point with several robots looks like one robot; randomness: deterministic; identifiers: anonymous: all robots are identical; initial configurations: every solvable initial configuration.

Measure	Lower bound	Best upper bound	Status
Time	none known Time is not analysed; only solvability.	*finite* (Terai et al. 2023)	● unknown
Colours	2 (Terai et al. 2023) Exactly 2 colours.	2 (Terai et al. 2023)	● tight

Results.

- Terai et al. (2023) [lower bound]: colours 2. *Bottleneck:* memory. *Holds when:* oblivious robots cannot gather under this scheduler.
- Terai et al. (2023) [current best]: time *finite*, colours 2. Introductory and current best (Theorem 9); non-rigid moves. With internal lights and a round-robin scheduler, 2 colours suffice with rigid moves (Theorem 10).

Claude (Anthropic) was used in cataloguing the problems.

Notes. Checked against the full text.

18.8.19 Crash faults, SSYNC

Adds oblivious, disoriented, non-rigid, + weak multiplicity, + $n \geq 3$ VERIFIED

With weak multiplicity detection and no common orientation, the correct robots gather despite up to $n-1$ crashes.

Assumed: body: point robots; visibility: unlimited; opacity: transparent: every robot sees every robot within range; faults: up to f robots may crash; scheduler: semi-synchronous (SSYNC); multiplicity detection: weak: a robot sees whether a point holds more than one robot; number of robots: $n \geq 3$. **Not assumed:** lights (persistent memory): oblivious, no light (OBLOT); agreement on coordinates: no agreement on local coordinate systems; movement: non-rigid: the adversary may stop a move after some minimum distance δ; randomness: deterministic; identifiers: anonymous: all robots are identical; initial configurations: every solvable initial configuration.

Measure	Lower bound	Best upper bound	Status
Time	*impossible* (Pattanayak et al. 2019b) Time not analysed.	*finite* (Pattanayak et al. 2019b)	● unknown
Colours	none known No colours.	0 (Pattanayak et al. 2019b)	● tight

Results.

- Pattanayak et al. (2019b) [lower bound]: time *impossible*. *Bottleneck:* indistinguishable multiplicities. *Holds when:* from a configuration with more than one multiplicity, no wait-free deterministic algorithm gathers with weak multiplicity detection (Theorem 1); the paper shows its model is minimal: relaxing the capabilities further makes gathering impossible.
- Agmon and Peleg (2006) [introductory]: time *finite*, colours 0. *Assumes:* one crash; common chirality; $n \geq 3$; a Look returns a multiset (multiplicities are seen); initial configuration with at most one point with a multiplicity. Theorem 3.8, SSYNC, non-rigid moves of at least a fixed distance S; checked against the full text.
- Bouzid et al. (2013) [also]: time *finite*, colours 0. *Assumes:* strong multiplicity detection; chirality; any $f < n$; initial configuration not bivalent. Wait-free; every initial configuration except the bivalent one (two points holding $n/2$ robots each), from which gathering is impossible.
- Bramas and Tixeuil (2015) [also]: time *finite*, colours 0. *Assumes:* strong multiplicity detection; no chirality; $f \leq n-1$; initial configuration not bivalent. Wait-free without chirality, in SSYNC with non-rigid moves (minimum distance δ), from every configuration that is not bivalent. Checked against the full text.
- Pattanayak et al. (2019b) [current best]: time *finite*, colours 0. *Assumes:* $f \leq n-1$; initial configurations with at most one multiplicity, which Theorem 1 shows necessary with weak detection. Wait-free (Theorem 2), weak multiplicity detection, no agreement on coordinates. The ASYNC algorithm of the same paper is in the setting with a restricted start.

Open questions.

Q1. "The maximum number of faults under which a solution is still feasible, for gathering and other tasks, has yet to be established" (Agmon and Peleg 2006), in particular in ASYNC.

18.8.20 Rendezvous, full lights, a crash, SSYNC

Adds + lights (LUMI), + compass, + rigid, + $n = 2$ VERIFIED

If the correct robot must reach a crashed one, two robots cannot meet in SSYNC, even with unboundedly many colours, rigid moves and a common coordinate system.

Claude (Anthropic) was used in cataloguing the problems.

Assumed: body: point robots; visibility: unlimited; opacity: transparent: every robot sees every robot within range; faults: up to f robots may crash; scheduler: semi-synchronous (SSYNC); lights (persistent memory): full light (LUMI): every robot sees all lights, including its own; agreement on coordinates: agreement on both axes (common compass); movement: rigid: every move reaches its destination; number of robots: $n = 2$ (rendezvous). **Not assumed:** multiplicity detection: none: a point with several robots looks like one robot; randomness: deterministic; identifiers: anonymous: all robots are identical; initial configurations: every solvable initial configuration.

Statement for this setting. Stand-up indulgent rendezvous: if one robot crashes, the other must reach the crashed robot's position.

Measure	Lower bound	Best upper bound	Status
Time	*impossible* (Bramas et al. 2023b) Unsolvable.	—	• impossible
Colours	none known Unsolvable for any number of colours.	—	• impossible

Results.

- Bramas et al. (2023b) [lower bound]: time *impossible*. *Bottleneck:* indistinguishable crashes. *Holds when:* Theorem 1: SSYNC; holds with full lights of infinitely many colours, rigid moves and a common coordinate system.

Notes. With mobility faults, where a faulty robot stops moving but its light keeps working, 3 colours are necessary and sufficient for two robots in ASYNC (Pramanick et al. 2025), a preprint. Checked against the full texts.

PLANE

18.8.21 Crash faults, ASYNC, restricted start

Adds ASYNC, + weak multiplicity, + $n \geq 3$, + restricted start VERIFIED

In ASYNC with instantaneous computation, up to $\lfloor n/2 \rfloor - 2$ crashes are tolerated for $n \geq 7$, from initial configurations that exclude some symmetric families.

Assumed: body: point robots; visibility: unlimited; opacity: transparent: every robot sees every robot within range; faults: up to f robots may crash; multiplicity detection: weak: a robot sees whether a point holds more than one robot; number of robots: $n \geq 3$; initial configurations: a restricted family of initial configurations. **Not assumed:** scheduler: asynchronous (ASYNC); lights (persistent memory): oblivious, no light (OBLOT); agreement on coordinates: no agreement on local coordinate systems; movement: non-rigid: the adversary may stop a move after some minimum distance δ; randomness: deterministic; identifiers: anonymous: all robots are identical.

Measure	Lower bound	Best upper bound	Status
Time	none known Time not analysed.	*finite* (Pattanayak et al. 2019b)	• unknown
Colours	none known No colours.	0 (Pattanayak et al. 2019b)	• tight

Results.

- Pattanayak et al. (2019b) [current best]: time *finite*, colours 0. *Assumes:* ASYNC with instantaneous computation (Look and Compute happen at one instant); $f \leq \lfloor n/2 \rfloor - 2$ crashes and $n \geq 7$; initial configurations with at most one multiplicity, excluding the symmetric families $C(0)$, $C(1/k)$, $C(1/2)$ and $C(1/2 + 1/k)$ of the paper (Table 1). Introductory and current best (Theorem 3), without agreement on coordinates. Cicerone, Di Stefano and Navarra state that the discussion after this theorem does not consider configurations with more than one axis of symmetry, and that it lets configurations with a robot at the centre of the smallest enclosing circle elect a leader only when n is even (Cicerone et al. 2021a) (arXiv version).

Claude (Anthropic) was used in cataloguing the problems.

Open questions.

Q1. Gather despite crashes in ASYNC from every solvable initial configuration, symmetric ones included, with weak multiplicity detection.

Notes. Checked against the full text of (Pattanayak et al. 2019b) (Problem 2, Table 1 and Theorem 3).

18.8.22 Byzantine faults, SSYNC

Adds Byzantine, + SSYNC, + $n \geq 3$ VERIFIED

Three robots with one Byzantine robot can neither gather nor converge in SSYNC.

Assumed: body: point robots; visibility: unlimited; opacity: transparent: every robot sees every robot within range; scheduler: semi-synchronous (SSYNC); number of robots: $n \geq 3$. **Not assumed:** faults: up to f Byzantine robots; lights (persistent memory): oblivious, no light (OBLOT); agreement on coordinates: no agreement on local coordinate systems; movement: non-rigid: the adversary may stop a move after some minimum distance δ; multiplicity detection: none: a point with several robots looks like one robot; randomness: deterministic; identifiers: anonymous: all robots are identical; initial configurations: every solvable initial configuration.

Measure	Lower bound	Best upper bound	Status
Time	*impossible* (Agmon and Peleg 2006) Impossible for $n = 3$, $f = 1$.	—	• impossible
Colours	none known Impossible.	—	• impossible

Results.

- Agmon and Peleg (2006) [lower bound]: time *impossible. Bottleneck:* indistinguishable Byzantine agents. *Holds when:* $n = 3$, $f = 1$ (Theorem 4.4); proved for SSYNC, hence also for ASYNC, which is what the abstract states; impossible for convergence too; holds even with common chirality and multiplicity detection (the paper's model); the adversary moves the Byzantine robot to make the configurations seen by two correct robots equivalent, and stops correct robots early (non-rigid moves).

Notes. Checked against the full text of (Agmon and Peleg 2006): Theorem 4.4 is stated for SSYNC; the abstract says "in an asynchronous environment", and the impossibility carries over to ASYNC because, as the paper notes citing Prencipe, every problem solvable in ASYNC is solvable in SSYNC. Theorem 4.1 first shows that any algorithm that sometimes keeps an active robot still fails.

18.8.23 Byzantine faults, FSYNC

Adds Byzantine, + FSYNC, + $n \geq 3$ VERIFIED

In FSYNC, gathering tolerates f Byzantine robots when $n \geq 3f + 1$.

Assumed: body: point robots; visibility: unlimited; opacity: transparent: every robot sees every robot within range; scheduler: fully synchronous (FSYNC); number of robots: $n \geq 3$. **Not assumed:** faults: up to f Byzantine robots; lights (persistent memory): oblivious, no light (OBLOT); agreement on coordinates: no agreement on local coordinate systems; movement: non-rigid: the adversary may stop a move after some minimum distance δ; multiplicity detection: none: a point with several robots looks like one robot; randomness: deterministic; identifiers: anonymous: all robots are identical; initial configurations: every solvable initial configuration.

Claude (Anthropic) was used in cataloguing the problems.

Measure	Lower bound	Best upper bound	Status
Time	none known Whether $3f+1$ is necessary is open.	*finite* (Agmon and Peleg 2006)	● unknown
Colours	none known No colours.	0 (Agmon and Peleg 2006)	● tight

Results.

- Agmon and Peleg (2006) [current best]: time *finite*, colours 0. *Assumes:* $n \geq 3f+1$; common chirality; a Look returns a multiset (multiplicities are seen). Introductory and current best (Theorem 5.10, with $n \geq 3f+1$); $n = 3$ tolerates $f = 1$ (Theorem 5.3). Checked against the full text.

Notes. Défago et al. study self-stabilizing gathering under crash or Byzantine faults (Distributed Computing 2020; bounds not checked).

18.8.24 Crash faults, one common axis, opaque robots

Adds opaque, + crash faults, + one axis VERIFIED

Robots that agree on one axis gather in ASYNC with non-rigid moves and no multiplicity detection despite up to $n-1$ crashes, even when robots block each other's view.

Assumed: body: point robots; visibility: unlimited; faults: up to f robots may crash; agreement on coordinates: agreement on one axis (no common chirality). **Not assumed:** opacity: opaque: collinear robots hide each other; scheduler: asynchronous (ASYNC); lights (persistent memory): oblivious, no light (OBLOT); movement: non-rigid: the adversary may stop a move after some minimum distance δ; multiplicity detection: none: a point with several robots looks like one robot; randomness: deterministic; identifiers: anonymous: all robots are identical; number of robots: any n; initial configurations: every solvable initial configuration.

Measure	Lower bound	Best upper bound	Status
Time	none known Finite time only; no bound on epochs is recorded.	*finite* (Bhagat et al. 2016a)	● unknown
Colours	none known No colours.	0 (Bhagat et al. 2016a)	● tight

Results.

- Bhagat et al. (2016a) [current best]: time *finite*, colours 0. *Assumes:* up to $f = n-1$ crashes; $n \geq 2$; non-rigid moves of at least δ. Introductory and current best (Theorem 3 for opaque robots; Theorems 1 and 2 for transparent ones). Robots order the horizontal lines through occupied points from north to south. If the northernmost line holds one occupied point, every robot moves to it; otherwise the target is the apex, north of that line, of the equilateral triangle on its two outermost occupied points. The target stays the same while robots move toward it (Lemma 3), so every correct robot reaches it, whichever robots crash. Wait-free: no robot can tell a crashed robot from an idle one.

Notes. With one common axis even two robots gather, which disoriented oblivious robots cannot do in SSYNC (Suzuki and Yamashita 1999). Bhagat, Gan Chaudhuri and Mukhopadhyaya assume no multiplicity detection and arbitrary initial configurations, with robots that may share points. Checked against the full text of (Bhagat et al. 2016a) (JDA version).

18.8.25 Limited visibility with a compass, ASYNC

Adds ASYNC, + compass VERIFIED

With a common compass, asynchronous robots with limited visibility gather in finite time.

Claude (Anthropic) was used in cataloguing the problems.

Assumed: body: point robots; opacity: transparent: every robot sees every robot within range; faults: no faults; agreement on coordinates: agreement on both axes (common compass). **Not assumed:** visibility: limited: robots see only within a fixed radius; scheduler: asynchronous (ASYNC); lights (persistent memory): oblivious, no light (OBLOT); movement: non-rigid: the adversary may stop a move after some minimum distance δ; multiplicity detection: none: a point with several robots looks like one robot; randomness: deterministic; identifiers: anonymous: all robots are identical; number of robots: any n; initial configurations: every solvable initial configuration.

Measure	Lower bound	Best upper bound	Status
Time	*impossible* (Flocchini et al. 2005) Time not analysed.	*finite* (Flocchini et al. 2005)	● unknown
Colours	none known No colours.	0 (Flocchini et al. 2005)	● tight

Results.

- Flocchini et al. (2005) [lower bound]: time *impossible*. *Bottleneck:* symmetry. *Holds when:* if the initial distance graph is disconnected (Lemma 4.7); two robots at distance more than V under a fully synchronous scheduler see the same empty view and keep their distance; holds for any deterministic algorithm, with a compass. Not a bound on the variant itself: it shows that connectivity of the initial distance graph is necessary.
- Flocchini et al. (2005) [current best]: time *finite*, colours 0. *Assumes:* initial distance graph (edges between robots at distance at most V) connected; no multiplicity detection; agreement on the directions and orientations of both axes, not on unit or origin. Introductory and current best (Theorem 4.2): all robots gather at a point on the rightmost initial vertical axis. Non-rigid moves with a minimum distance δ.

Notes. Checked against the full text of (Flocchini et al. 2005) (TCS 2005; preliminary version STACS 2001): robots are oblivious, anonymous, fully asynchronous (ASYNC), with non-rigid moves that cover at least a distance $\delta_r > 0$, no multiplicity detection, and "orientation": agreement on the directions and orientations of both axes, but not on the unit distance or the origin. The authors conclude that "with respect to gathering, orientation is at least as powerful as instantaneous movements". With rigid moves added (setting *Limited visibility with a compass, rigid, ASYNC*), Poudel and Sharma gather the robots at a point in $O(\Delta_0)$ epochs, which is optimal (Theorem 3 of Poudel and Sharma 2021, visibility radius $V = \sqrt{10}$).

18.8.26 Limited visibility with a compass, rigid, ASYNC

Adds + rigid **to** Limited visibility with a compass, ASYNC (p. 454) VERIFIED

With a common compass and rigid moves, asynchronous robots with limited visibility gather at a point in time linear in the initial diameter, which is optimal.

Assumed: body: point robots; opacity: transparent: every robot sees every robot within range; faults: no faults; agreement on coordinates: agreement on both axes (common compass); movement: rigid: every move reaches its destination. **Not assumed:** visibility: limited: robots see only within a fixed radius; scheduler: asynchronous (ASYNC); lights (persistent memory): oblivious, no light (OBLOT); multiplicity detection: none: a point with several robots looks like one robot; randomness: deterministic; identifiers: anonymous: all robots are identical; number of robots: any n; initial configurations: every solvable initial configuration.

Measure	Lower bound	Best upper bound	Status
Time	$\Omega(\Delta_0)$ (Poudel and Sharma 2021) $\Theta(\Delta_0)$ epochs.	$O(\Delta_0)$ (Poudel and Sharma 2021)	● tight
Colours	none known No colours.	0 (Poudel and Sharma 2021)	● tight

Results.

Claude (Anthropic) was used in cataloguing the problems.

- Poudel and Sharma (2021) [lower bound]: time $\Omega(\Delta_0)$. *Bottleneck:* distance. *Holds when:* Δ_0 = Euclidean diameter of the initial configuration; called the trivial lower bound in (Poudel and Sharma 2021): a robot moves at most a constant distance per cycle, inside its axis-aligned 2×2 square.
- Poudel and Sharma (2021) [current best]: time $O(\Delta_0)$, colours 0. *Assumes:* visibility radius $V = \sqrt{10}$; two robots are joined when each lies in the axis-aligned 2×2 square centred at the other; the initial graph is connected. Current best (Theorem 3): all robots gather at one point in $O(\Delta_0)$ epochs; robots that occupy one point are not merged. Finite-time gathering in this setting already follows from the non-rigid algorithm of (Flocchini et al. 2005).

Open questions.

Q1. Does $O(\Delta_0)$-epoch gathering need rigid moves, or a visibility radius larger than the connectivity range (Poudel and Sharma 2021)?

Notes. Poudel and Sharma conjecture that with viewing and connectivity ranges of 1 no $O(\Delta_0)$ algorithm exists, even with a compass. Checked against the full text of (Poudel and Sharma 2021) (Information 2021; Theorems 2 and 3, model in Section 2).

PLANE

18.8.27 Limited visibility, one common axis, rigid, ASYNC

Adds ASYNC, + one axis, + rigid VERIFIED

With one common axis and rigid moves, asynchronous robots with limited visibility reach a horizontal segment of length 1 in time linear in the initial diameter; gathering at a point is not shown.

Assumed: body: point robots; opacity: transparent: every robot sees every robot within range; faults: no faults; agreement on coordinates: agreement on one axis (no common chirality); movement: rigid: every move reaches its destination. **Not assumed:** visibility: limited: robots see only within a fixed radius; scheduler: asynchronous (ASYNC); lights (persistent memory): oblivious, no light (OBLOT); multiplicity detection: none: a point with several robots looks like one robot; randomness: deterministic; identifiers: anonymous: all robots are identical; number of robots: any n; initial configurations: every solvable initial configuration.

Measure	Lower bound	Best upper bound	Status
Time	$\Omega(\Delta_0)$ (Poudel and Sharma 2021) $\Theta(\Delta_0)$ epochs for the relaxed task (a segment of length 1); no bound is recorded for gathering at a point with one common axis.	$O(\Delta_0)$ (Poudel and Sharma 2021)	● unknown
Colours	none known No colours.	0 (Poudel and Sharma 2021)	● tight

Results.

- Poudel and Sharma (2021) [lower bound]: time $\Omega(\Delta_0)$. *Bottleneck:* distance. *Holds when:* Δ_0 = Euclidean diameter of the initial configuration.
- Poudel and Sharma (2021) [current best]: time $O(\Delta_0)$, colours 0. *Assumes:* visibility radius $V = \sqrt{10}$; two robots are joined when each lies in the axis-aligned 2×2 square centred at the other; the initial graph is connected. Introductory and current best (Theorem 1): a relaxed gathering, all robots on a horizontal segment of length 1 not fixed in advance, in $O(\Delta_0)$ epochs; robots that occupy one point are not merged. FSYNC $O(g)$, g the diameter (in hops) of the initial visibility graph, was known before.

Open questions.

Q1. Can asynchronous robots with one common axis and limited visibility gather at a point, and in $O(\Delta_0)$ epochs? (Poudel and Sharma 2021) reach only a horizontal segment of length 1, because two robots on a horizontal line, disagreeing on east and west, may swap places forever.

Claude (Anthropic) was used in cataloguing the problems.

Notes. Checked against the full text of (Poudel and Sharma 2021) (Information 2021). The variant was previously recorded without rigid moves and as gathering at a point; the paper assumes rigid moves and proves only relaxed gathering for one common axis (Definition 3, Theorem 1). With a common compass the same algorithm gathers at a point in $O(\Delta_0)$ epochs (Theorem 3), (setting *Limited visibility with a compass, rigid, ASYNC*).

18.8.28 Limited visibility, FSYNC, disoriented

Adds + FSYNC VERIFIED

Disoriented robots with limited visibility gather in $O(\Delta_0^2)$ synchronous rounds, Δ_0 the initial diameter.

Assumed: body: point robots; opacity: transparent: every robot sees every robot within range; faults: no faults; scheduler: fully synchronous (FSYNC). **Not assumed:** visibility: limited: robots see only within a fixed radius; lights (persistent memory): oblivious, no light (OBLOT); agreement on coordinates: no agreement on local coordinate systems; movement: non-rigid: the adversary may stop a move after some minimum distance δ; multiplicity detection: none: a point with several robots looks like one robot; randomness: deterministic; identifiers: anonymous: all robots are identical; number of robots: any n; initial configurations: every solvable initial configuration.

Measure	Lower bound	Best upper bound	Status
Time	$\Omega(\Delta_0^2)$ (Castenow et al. 2022) Tight within the class of local convex-hull protocols; for arbitrary protocols only the trivial bound is known.	$O(\Delta_0^2)$ (Castenow et al. 2022)	• tight
Colours	none known No colours.	0 (Castenow et al. 2022)	• tight

Results.

- Castenow et al. (2022) [lower bound]: time $\Omega(\Delta_0^2)$. *Bottleneck:* locality. *Holds when:* only for protocols that move inside the local convex hull of the neighbourhood; Δ_0 = largest initial distance between two robots.
- Degener et al. (2011) [introductory]: time $O(n^2)$, colours 0. The algorithm of Ando, Oasa, Suzuki and Yamashita (move towards the centre of the smallest enclosing circle of the visible robots, as far as all of them stay visible), called Go-To-The-Center in (Castenow et al. 2022): $\Theta(n^2)$ rounds for that algorithm (Theorems 5.1 and 6.14).
- Castenow et al. (2022) [current best]: time $O(\Delta_0^2)$, colours 0. Every λ-contracting protocol, including Go-To-The-Center; works in two and three dimensions.

Open questions.

Q1. Is $\Omega(\Delta_0^2)$ necessary for every protocol, not only local convex-hull ones?

Notes. Δ_0 here is a distance, not the maximum degree.

18.8.29 Fat robots, few robots, no common orientation

Adds disoriented, + $n \geq 3$ VERIFIED

Three and four oblivious fat robots, which block each other's view and motion, gather in ASYNC without any common orientation.

Assumed: visibility: unlimited; faults: no faults; number of robots: $n \geq 3$. **Not assumed:** body: fat robots (unit discs); opacity: opaque: collinear robots hide each other; scheduler: asynchronous (ASYNC); lights (persistent memory): oblivious, no light (OBLOT); agreement on coordinates: no agreement on local coordinate systems; movement: non-rigid: the adversary may stop a move after some minimum distance δ; multiplicity detection: none: a point with several robots looks like one robot; randomness: deterministic; identifiers: anonymous: all robots are identical; initial configurations: every solvable initial configuration.

Claude (Anthropic) was used in cataloguing the problems.

Statement for this setting. Fat robots (unit discs) cannot share a point: gathering asks for a connected configuration of touching discs in which every robot sees all others.

Measure	Lower bound	Best upper bound	Status
Time	none known —	*finite* (Czyzowicz et al. 2009b)	● unknown
Colours	none known No colours.	0 (Czyzowicz et al. 2009b)	● tight

Results.

- Czyzowicz et al. (2009b) [current best]: time *finite*, colours 0. *Assumes:* $n \leq 4$; robots know n. Introductory and current best for few robots: Theorem 3.1 ($n = 3$) and the four-robot algorithm of Section 4.

Open questions.

Q1. Gathering any number of fat robots without chirality (Agathangelou et al. 2013).

Notes. Checked against the full text of (Czyzowicz et al. 2009b) (TCS 2009; OPODIS 2006). Model: closed unit discs, identical and anonymous, oblivious, silent, no common coordinate system (but a common unit, the radius), ASYNC. A robot sees another if some segment between their boundary circles meets no third robot; a robot that touches another stops and ends its cycle; the adversary may stop a move after a distance δ unknown to the robots. Gathering means a connected union of discs in which every robot sees all others. The authors could not generalise to more robots ("We were unable to do it").

18.8.30 Fat robots, chirality, any number

Adds + chirality, + $n \geq 3$ VERIFIED

With chirality and known n, any number of opaque fat robots gather in ASYNC with non-rigid moves.

Assumed: visibility: unlimited; faults: no faults; agreement on coordinates: common chirality (same clockwise orientation); number of robots: $n \geq 3$. **Not assumed:** body: fat robots (unit discs); opacity: opaque: collinear robots hide each other; scheduler: asynchronous (ASYNC); lights (persistent memory): oblivious, no light (OBLOT); movement: non-rigid: the adversary may stop a move after some minimum distance δ; multiplicity detection: none: a point with several robots looks like one robot; randomness: deterministic; identifiers: anonymous: all robots are identical; initial configurations: every solvable initial configuration.

Statement for this setting. Gathering is Definition 1 of (Agathangelou et al. 2013): a connected configuration (every disc tangent to another), fully visible (every robot sees all others) and terminal, with all centres on the convex hull.

Measure	Lower bound	Best upper bound	Status
Time	none known Time is not analysed.	*finite* (Agathangelou et al. 2013)	● unknown
Colours	none known No colours.	0 (Agathangelou et al. 2013)	● tight

Results.

- Agathangelou et al. (2013) [current best]: time *finite*, colours 0. *Assumes:* n known; the radius is the common unit of distance; robots that become tangent stop. Introductory for any n and current best (Theorem 26).

Open questions.

Claude (Anthropic) was used in cataloguing the problems.

Q1. Remove the chirality assumption (Agathangelou et al. 2013).

Notes. Checked against the full text (arXiv 1209.3904).

18.8.31 Transparent fat robots, no chirality

Adds + transparent, + no faults, + $n \geq 3$ VERIFIED

Transparent fat robots without chirality gather into a layered pattern from every start whose robots can be ordered, which is exactly the asymmetric starts.

Assumed: visibility: unlimited; opacity: transparent: every robot sees every robot within range; faults: no faults; number of robots: $n \geq 3$. **Not assumed:** body: fat robots (unit discs); scheduler: asynchronous (ASYNC); lights (persistent memory): oblivious, no light (OBLOT); agreement on coordinates: no agreement on local coordinate systems; movement: non-rigid: the adversary may stop a move after some minimum distance δ; multiplicity detection: none: a point with several robots looks like one robot; randomness: deterministic; identifiers: anonymous: all robots are identical; initial configurations: every solvable initial configuration.

Statement for this setting. Transparent fat robots do not block sight but block movement. Gathering asks for a layered pattern around the centre of the smallest enclosing circle, with a unique robot at the centre (Chaudhuri and Mukhopadhyaya 2015).

Measure	Lower bound	Best upper bound	Status
Time	none known —	*finite* (Chaudhuri and Mukhopadhyaya 2015)	● unknown
Colours	none known No colours.	0 (Chaudhuri and Mukhopadhyaya 2015)	● tight

Results.

- Chaudhuri and Mukhopadhyaya (2010) [introductory]: time *finite*, colours 0. *Assumes:* $n \geq 5$; robots start separated. A short paper with no main theorem: an algorithm for $n \geq 5$ asynchronous oblivious transparent fat robots that gathers them "if no symmetric configuration [is] encountered during run time"; excluding such configurations is left open.

- Chaudhuri and Mukhopadhyaya (2015) [current best]: time *finite*, colours 0. *Assumes:* $n \geq 5$; robots start separated (no two discs touching); rigid moves. A configuration can be ordered, and a leader elected, if and only if it is asymmetric (Theorem 6); the algorithm gathers from every orderable start.

Open questions.

Q1. Gathering transparent fat robots from symmetric starts; the paper proves no general impossibility (Chaudhuri and Mukhopadhyaya 2015).

Notes. Gathering at a given point with limited visibility (dense packing around the point) is a different goal; it is studied by Cord-Landwehr et al. (SOFSEM 2011) and Sharma et al. (TAAS 2017), not yet read. Checked against the full texts of (Chaudhuri and Mukhopadhyaya 2015) (arXiv 1208.4484) and (Chaudhuri and Mukhopadhyaya 2010).

Claude (Anthropic) was used in cataloguing the problems.

18.9 Near-gathering

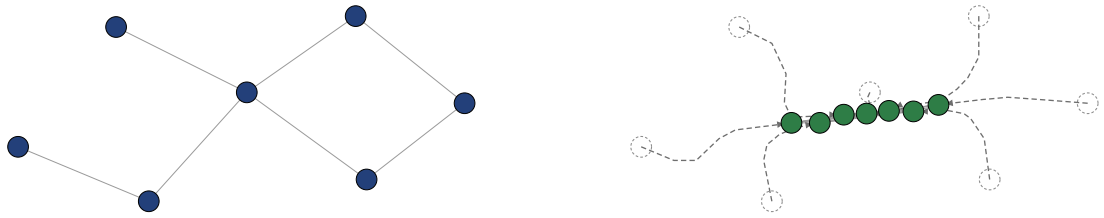
Robots must get within a small distance of each other without ever colliding.

First posed in 2012 (Pagli et al. 2015). Introduced at SIROCCO 2012.

Problem statement and measures

Robots with limited visibility, each seeing only the robots within its **visibility radius** V, start at distinct points whose **visibility graph** (two robots joined by an edge when they are within distance V) is connected. They must reach a configuration in which every robot sees all others, without a **collision** (two robots at the same point) ever happening.

Measures: **Time**, rounds (FSYNC, SSYNC) or epochs (ASYNC) until the task is done; "finite" when only solvability is known; **Colours**, colours of the light per robot (0 for oblivious robots).



(a) Start: neighbours within V share an edge

(b) Final: every robot sees all, none collide (arrows: moves; dashed circles: starting positions)

Figure 18.9: Near-gathering: robots with limited visibility, starting at distinct points whose visibility graph is connected, must come close enough that every robot sees all others, without two robots ever occupying the same point (Pagli et al. 2015). Shown in FSYNC rounds: the Go-To-the-Center rule of (Castenow et al. 2022): each robot moves part of the way towards the centre of the smallest circle enclosing the robots it sees, staying within $V/2$ of the midpoint with each neighbour, so edges of the visibility graph are never lost.

A first look

Imagine short-sighted people in fog who must come close enough that everybody can see everybody, without bumping into each other. In the problem the people are n **robots**, points at distinct places in the plane, and each sees only the robots within distance V (its **visibility radius**). Join two robots by an **edge** when they are within distance V, and call them **neighbours**; the robots and these edges form the **visibility graph**, and at the start it is **connected** (every robot can be reached from every other along edges). Each robot reads positions in its own **local coordinate system** (its private axes and unit of length) and repeats a **Look–Compute–Move cycle**: it takes a **snapshot** of where the robots it sees are (Look), decides on a destination point (Compute) and moves towards it (Move). Robots are identical, silent and **oblivious** (no memory between cycles); in the basic version their local coordinate systems agree on the direction of one axis, and the **scheduler**, controlled by an **adversary** (an imagined opponent) that decides when each robot acts and how long each phase takes, is ASYNC (each phase may take any finite time). The robots must reach a **configuration** (the positions of all robots) in which every robot sees all others, and two robots may never be at the same point (a **collision**).

Claude (Anthropic) was used in cataloguing the problems.

1. Algorithm: Go to the centre of what you see, in small steps, keeping your neighbours. (Castenow et al. 2022) (SSYNC, limited visibility, disoriented, p. 463)

1. Compute the **smallest enclosing circle** of the robots you can see, and aim for its centre (the **Go-To-the-Center** rule).
2. For every robot you see (every neighbour), stay inside the disc of radius $V/2$ around the midpoint between you and it. Your neighbour obeys the same rule, so the two of you stay within distance V, your edge is kept, and the visibility graph stays connected.
3. To avoid collisions, move only a short distance and only part of the way to your target. Here robots see a little farther than V: edges still join robots within distance V (the **connectivity range**), but each robot sees every robot within distance $V + \tau$ (its **viewing range**), for a constant $0 < \tau \leq 2V/3$ (Castenow et al. 2022). So a robot can also compute where its nearby neighbours are heading and keep clear of them.
4. Stop when everyone you see is very close: since the visibility graph is connected, then everyone is close.

Why it works and what it costs. Collisions are the new difficulty compared with gathering. Even for two robots, the simple rule "move half-way to the centre of gravity" can collide in ASYNC: robot r at $(0, 0)$ heads for $(781, 0)$; the scheduler pauses it at $(52, 0)$ while robot s, starting at $(3124, 0)$ and always seeing r at $(52, 0)$, makes five cycles and arrives at $(781, 0)$; then r finishes its move onto s (Pagli et al. 2015). Castenow, Harbig, Jung, Kling, Knollmann and Meyer auf der Heide show that any gathering rule of a broad class, including Go-To-the-Center, can be made collision-free and solves near-gathering for disoriented robots in $O(\Delta_0^2)$ **epochs** (an epoch is an interval of time in which every robot completes at least one full cycle) in SSYNC, where Δ_0 is the initial diameter (largest distance between two robots) and V is the unit (Theorem 18 of Castenow et al. 2022). The steps above simplify their construction.

Why it matters. It lets short-sighted robots reach a state where they all see each other, after which any algorithm for robots with unlimited visibility can take over.

2. Lower bound: A big regular polygon shrinks slowly. (Castenow et al. 2022) (SSYNC, limited visibility, disoriented, p. 463) Place n robots on a regular n-gon with sides of length 1, the visibility radius, so each robot sees only its two neighbours. The adversary chooses local coordinate systems so that all robots see the same thing and the polygon stays regular forever. Each robot sees itself and two neighbours nearly on a straight line: the triangle they form has height at most about π/n. If a robot always moves inside the triangle of what it sees, it moves towards the centre by at most about π/n per round (a step of time in which each robot that the scheduler activates performs one full cycle). The polygon's radius is about $n/(2\pi)$, so shrinking it to two thirds of its size takes about n^2 rounds.

Why it holds. The diameter of this start is $\Delta_0 \approx n/\pi$, so $\Omega(\Delta_0^2)$ rounds are needed by every protocol whose robots always move inside the region spanned by what they see (Theorem 4 of Castenow et al. 2022); Go-To-the-Center and its collision-free version are of this kind, so their $O(\Delta_0^2)$ bound is tight for them. Whether moving outside that region can be faster is open.

Why it matters. It shows that local rules pay quadratically in the size of the swarm, and points to where a faster algorithm would have to do something different.

3. Impossibility: Robots that see nobody stay apart. (Pagli et al. 2015) (ASYNC, limited visibility, one common axis, p. 463) Why must the start be connected? Take two robots farther apart than V. Each sees only itself, in every cycle. A deterministic oblivious robot with the same snapshot always computes the same move in its own coordinates. The adversary gives both robots the same orientation and unit and wakes them together: they make exactly the same move in the plane, so their distance never changes. They never see each other.

Claude (Anthropic) was used in cataloguing the problems.

Why it holds. So near-gathering cannot be solved from a disconnected start, which Pagli, Prencipe and Viglietta note is "easily seen" (Pagli et al. 2015); the same holds for gathering (Flocchini et al. 2005). From a connected start, Pagli et al. solve near-gathering in ASYNC, without collisions and with termination, for robots that agree on one axis (Pagli et al. 2015).

Why it matters. It is the reason every limited-visibility algorithm assumes a connected start and works hard to keep it connected.

What the community started looking at

Near-gathering was introduced to **get robots close without collisions**, a safe first phase for other tasks (Pagli et al. 2015); later work asked how fast disoriented robots can do it (Castenow et al. 2022).

Further reading

Papers on this problem that are not yet placed in a setting.

- Di Luna et al. (2021). TuringMobile: $3m + 3k$ oblivious, non-rigid ASYNC robots with limited visibility in $\mathbb{R}^m$ can simulate a rigid robot with k registers that each hold a real number; three robots suffice in the plane for one register, and fewer cannot. Applications: near-gathering and pattern formation with limited visibility in any dimension, even from a disconnected visibility graph, when a few robots form such a machine. Read (arXiv v2).
- Gerlach et al. (2024). Near-gathering that preserves the rotational symmetry of the swarm, for disoriented oblivious robots with limited visibility: a variant of Go-To-The-Average always preserves symmetry but may split the swarm, and a symmetry-preserving near-gathering algorithm works for swarms with a convex boundary and no holes; the analysis uses dynamical systems. Read (arXiv v1).
- Li et al. (2021a). In continuous time (robots adjust their motion to what they see at every instant), a variant of the relative-centre algorithm brings a connected start into a small circle without collisions in $O(n^2)$ time (Theorem 5); the authors say they trade the common orientation assumed by (Pagli et al. 2015) for chirality, lights and a clock. Read in full.

Relations to other problems

- **Relaxation of Gathering.** Gathering requires a common point; near-gathering only requires being close, but forbids collisions. It is used as a first phase before forming patterns.
- **Related: Convergence.** Near-gathering is also a relaxation of gathering: robots must get close (within visibility of each other) without ever colliding (Pagli et al. 2015), while convergence lets them approach a point only in the limit.

Case tree

The root assumes nothing; going down, each edge adds the assumptions written in grey (+) or leaves an axis unassumed. Bold entries are catalogued settings with their status per measure (Time, Colours) and page; italic entries are branch points, named after the axis on which their branches differ. Axis order: Body → Visibility → Opacity → Faults → Scheduler → Lights (persistent memory) → Agreement on coordinates → Movement → Multiplicity detection → Randomness → Identifiers.

No assumptions

+ point, + transparent, + no faults
Scheduler

Claude (Anthropic) was used in cataloguing the problems.

ASYNC, + one axis
ASYNC, limited visibility, one common axis STANDARD SETTING ●● p. 463

+ SSYNC
SSYNC, limited visibility, disoriented ●● p. 463

What is known

The best upper bound against the best lower bound in each catalogued setting, in tree order. The settings follow in detail.

Setting	Time	Colours	p.
ASYNC, limited visibility, one common axis STANDARD SETTING	● unknown *finite* vs —	● tight 0 vs —	463
SSYNC, limited visibility, disoriented	● tight $O(\Delta_0^2)$ vs $\Omega(\Delta_0^2)$	● tight 0 vs —	463

18.9.1 ASYNC, limited visibility, one common axis

Adds ASYNC, + one axis STANDARD SETTING VERIFIED

Robots get close together without ever colliding, from a connected start, if they agree on one axis.

Assumed: body: point robots; opacity: transparent: every robot sees every robot within range; faults: no faults; agreement on coordinates: agreement on one axis (no common chirality). **Not assumed:** visibility: limited: robots see only within a fixed radius; scheduler: asynchronous (ASYNC); lights (persistent memory): oblivious, no light (OBLOT); movement: non-rigid: the adversary may stop a move after some minimum distance δ; multiplicity detection: none: a point with several robots looks like one robot; randomness: deterministic; identifiers: anonymous: all robots are identical.

Measure	Lower bound	Best upper bound	Status
Time	none known —	*finite* (Pagli et al. 2015)	● unknown
Colours	none known No colours.	0 (Pagli et al. 2015)	● tight

Results.

- Pagli et al. (2015) [current best]: time *finite*, colours 0. Introductory and current best; collision-free with termination.

Open questions.

Q1. Pagli, Prencipe and Viglietta conjecture that near-gathering is impossible without agreement on one axis.

18.9.2 SSYNC, limited visibility, disoriented

Adds + SSYNC VERIFIED

Disoriented robots near-gather in $O(\Delta_0^2)$ epochs in SSYNC.

Assumed: body: point robots; opacity: transparent: every robot sees every robot within range; faults: no faults; scheduler: semi-synchronous (SSYNC). **Not assumed:** visibility: limited: robots see only within a fixed radius; lights (persistent memory):

Claude (Anthropic) was used in cataloguing the problems.

oblivious, no light (OBLOT); agreement on coordinates: no agreement on local coordinate systems; movement: non-rigid: the adversary may stop a move after some minimum distance δ; multiplicity detection: none: a point with several robots looks like one robot; randomness: deterministic; identifiers: anonymous: all robots are identical.

Measure	Lower bound	Best upper bound	Status
Time	$\Omega(\Delta_0^2)$ (Castenow et al. 2022) Within contracting protocols.	$O(\Delta_0^2)$ (Castenow et al. 2022)	• tight
Colours	none known No colours.	0 (Castenow et al. 2022)	• tight

Results.

- Castenow et al. (2022) [lower bound]: time $\Omega(\Delta_0^2)$. *Bottleneck:* locality. *Holds when:* only for λ-contracting protocols.
- Castenow et al. (2022) [current best]: time $O(\Delta_0^2)$, colours 0. Theorem 18; first near-gathering protocols for disoriented robots.

Open questions.

Q1. Near-gathering in $O(\Delta_0)$ epochs?

PLANE

18.10 Partitioning

Robots must partition themselves into k groups of exactly n/k robots each, each group at its own point.

First posed in 2021 (Bhagat et al. 2021). The fixed-point version (k-EPF) was posed with k-circle formation in 2021; the earlier partitioning of Efrima and Peleg fixes the number of groups instead and is recorded under k-grouping.

Problem statement and measures

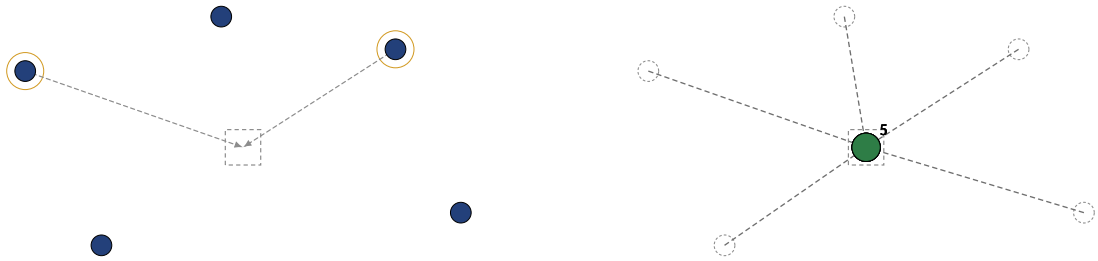
n robots, with k dividing n, must partition themselves into k groups of exactly n/k robots, each group gathered at its own point, and stay there. The k points, and so the group size n/k, are given (the papers' k-EPF and k-circle formation name the group size k, which is n/k here).

Measures: **Time**, rounds (FSYNC, SSYNC) or epochs (ASYNC) until the task is done; "finite" when only solvability is known; **Colours**, colours of the light per robot (0 for oblivious robots).

A first look

Imagine cars that must park in equal groups at a few marked charging stations. In the problem the cars are n **robots**, points in the plane, and the stations are k **fixed points** that every robot can see; k divides n. Each robot reads positions in its own **local coordinate system** (its private axes, unit of length and handedness) and repeats a **Look–Compute–Move cycle**: it takes a **snapshot** of where the other robots and the fixed points are (Look), decides on a destination point (Compute) and moves towards it (Move). Robots are identical, silent and **oblivious** (no memory between cycles); in the basic version their local coordinate systems agree on the direction of one axis (say north), all robots see everything, and the **scheduler**, which decides when each robot acts and how long each phase takes, is ASYNC (each phase may take any finite time). The robots must end with exactly n/k robots, a **group**, on each fixed point,

Claude (Anthropic) was used in cataloguing the problems.



(a) Start: $n = 5$, $k = 1$: one fixed point

(b) Final: all $n = 5$ robots on the fixed point (arrows: moves; dashed circles: starting positions)

Figure 18.10: Partitioning (k-EPF): n robots, k dividing n, must gather in groups of exactly n/k robots at the k given fixed points, each point receiving one group (Bhagat et al. 2021). Shown: the simplest case $k = 1$, one fixed point (dashed square), in SSYNC rounds; ringed robots are activated and move straight towards the point, a move possibly stopping early, until all are on it.

and stay there. The question is for which starting **configurations** (positions of all robots) and numbers k of fixed points this is possible.

1. Algorithm: The easy extreme: one fixed point for everybody. (Exactly n/k robots at each fixed point, p. 467) A standard approach for the simplest case, $k = 1$, where there is only one fixed point:

1. Look for the fixed point.
2. Move straight towards it.
3. Once on it, stay.

Why it works and what it costs. No coordination is needed, because the fixed point is a landmark all robots recognise: every move either reaches the point or covers at least a minimum distance, so each robot arrives after finitely many cycles. With more than one fixed point the robots must agree who goes where, and robots that look alike may be unable to choose differently; Bhagat, Das, Chakraborty and Mukhopadhyaya characterise the configurations and numbers k of fixed points for which the problem is solvable with one common axis (Bhagat et al. 2021), the catalogue's introductory and current best result (not described in detail here). The opposite extreme, $k = n$, asks for one robot per fixed point.

Why it matters. It shows the role of fixed points: they are landmarks that break much of the symmetry robots cannot break themselves.

2. Impossibility: Mirror twins arrive in pairs. (Exactly n/k robots at each fixed point, p. 467) A standard symmetry argument, for robots that cannot tell left from right. Let $n = 6$ and $k = 2$ (groups of 3), and put both fixed points on a vertical line L. Place three robots left of L and their three mirror images to the right, none on L; a robot and its mirror image are **mirror twins**. The **adversary**, an imagined opponent that chooses the local coordinate systems and when robots act, gives each pair of mirror twins mirror-image local coordinate systems and always activates them together. Twins see mirror-image snapshots, so they make mirror-image moves and stay mirror images forever. A robot can reach a point of L only together with its twin, so every point of L always holds an even number of robots. It can never hold exactly 3.

Why it holds. The same parity argument rules out every odd group size n/k whenever all fixed points lie on a line of symmetry of the start that holds no robot. This is why the catalogue's setting assumes a common axis, and why the answer depends on the start; the exact characterisation is in (Bhagat et al.

2021).

Why it matters. It is a compact example of how symmetry and counting combine to make a placement task impossible.

What the community started looking at

Partitioning into groups of a **given size** (n/k here; the papers write k) was first studied with the group locations fixed in advance (k-EPF), together with k-circle formation; the question was **for which configurations and group sizes it is solvable** when robots share only one axis (Bhagat et al. 2021).

Further reading

Papers on this problem that are not yet placed in a setting.

- Pramanick et al. (2024). Another partitioning: opaque oblivious robots with lights, under ASYNC, divide a known rectangle, square or circle into regions of equal area, one robot in each; $O(n)$ epochs with 2 colours (optimal) for rectangles and 5 for squares, $O(n^2)$ epochs with 8 colours for circles. Read (arXiv v2).
- Chakraborty and Mukhopadhyaya (2025). A discrete relative: oblivious ASYNC robots on an infinite grid, without common axes or chirality, must fill given parking nodes exactly to their capacities. All unsolvable initial configurations are characterised, and an algorithm solves the others. Read (arXiv v1).
- Mondal et al. (2024b). k-circle formation on the infinite square grid, a circle being drawn on the grid by Bresenham's algorithm: n asynchronous oblivious robots with one common axis form k disjoint circles of n/k robots each (their m circles of k robots); an algorithm with a correctness proof, and simulations. A short paper. Read in full.

Relations to other problems

- **Special case of Pattern formation.** A partition into k groups at points is a pattern of k points, each of multiplicity n/k.
- **Related: k-grouping.** Partitioning fixes the points, and so the number k of groups and their size n/k; k-grouping fixes only the number of groups. They coincide for equal-sized groups (Efrima and Peleg 2009).

Case tree

The root assumes nothing; going down, each edge adds the assumptions written in grey (+) or leaves an axis unassumed. Bold entries are catalogued settings with their status per measure (Time, Colours) and page; italic entries are branch points, named after the axis on which their branches differ. Axis order: Body → Visibility → Opacity → Faults → Scheduler → Lights (persistent memory) → Agreement on coordinates → Movement → Multiplicity detection → Randomness → Identifiers.

No assumptions

+ point, + unlimited visibility, + transparent, + no faults, + one axis

Exactly n/k robots at each fixed point STANDARD SETTING ●● p. 467

Claude (Anthropic) was used in cataloguing the problems.

What is known

The best upper bound against the best lower bound in each catalogued setting, in tree order. The settings follow in detail.

Setting	Time	Colours	p.
Exactly n/k robots at each fixed point STANDARD SETTING	● unknown *finite* vs —	● tight 0 vs —	467

18.10.1 Exactly n/k robots at each fixed point

Adds + point, + unlimited visibility, + transparent, + no faults, + one axis STANDARD SETTING VERIFIED

With one common axis, the configurations and numbers k of fixed points for which exactly n/k robots can gather at each of given fixed points are characterised.

Assumed: body: point robots; visibility: unlimited; opacity: transparent: every robot sees every robot within range; faults: no faults; agreement on coordinates: agreement on one axis (no common chirality). **Not assumed:** scheduler: asynchronous (ASYNC); lights (persistent memory): oblivious, no light (OBLOT); movement: non-rigid: the adversary may stop a move after some minimum distance δ; multiplicity detection: none: a point with several robots looks like one robot; randomness: deterministic; identifiers: anonymous: all robots are identical.

Measure	Lower bound	Best upper bound	Status
Time	none known —	*finite* (Bhagat et al. 2021)	● unknown
Colours	none known No colours.	0 (Bhagat et al. 2021)	● tight

Results.

- Bhagat et al. (2021) [current best]: time *finite*, colours 0. Introductory and current best. Also k-circle formation: disjoint circles of n/k robots each, centred at the fixed points.

Open questions.

Q1. Partitioning into k groups of size n/k at points chosen by the robots (no fixed points given).

Notes. The k-EPF problem generalises embedded pattern formation: the fixed points are visible to all robots and each must end with exactly n/k robots (their k is n/k here) (Bhagat et al. 2021).



18.11 Convergence

Robots must get arbitrarily close to a common point; they need not ever meet.

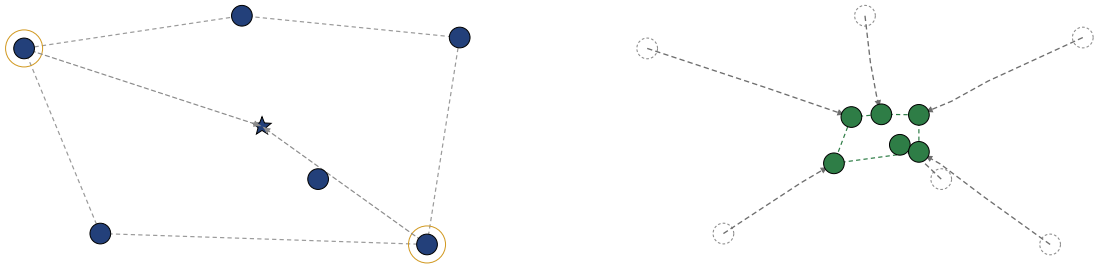
First posed in 1999 (Suzuki and Yamashita 1999). Suzuki and Yamashita gave algorithms for converging to a point and for reaching a point; Ando et al. treated limited visibility the same year (Ando et al. 1999).

Problem statement and measures

n robots start at arbitrary points. For every $\varepsilon > 0$ there must be a time after which all robots stay within distance ε of each other (equivalently, of a common limit point). Robots never need to coincide.

Claude (Anthropic) was used in cataloguing the problems.

Measures: **Time**, rounds (FSYNC, SSYNC) or epochs (ASYNC) until the task is done; "finite" when only solvability is known; **Colours**, colours of the light per robot (0 for oblivious robots).



(a) Start: ringed robots move towards the centre of gravity

(b) After 9 rounds: the convex hull keeps shrinking (arrows: moves; dashed circles: starting positions)

Figure 18.11: Convergence: robots at arbitrary points must, for every $\varepsilon > 0$, eventually stay within distance ε of each other; they approach a common limit point but need never meet (Suzuki and Yamashita 1999). Shown: the centre-of-gravity rule of (Cohen and Peleg 2005) in SSYNC rounds; ringed robots are activated and move towards the centre of gravity (star) of all robots, some stopping early.

A first look

Think of a flock that must squeeze ever closer together, without ever needing to land on one spot. There are n **robots**, points in the plane; their positions form the **configuration**. Each robot repeats a **Look–Compute–Move cycle**: it takes a **snapshot** of where the other robots are (Look), decides on a destination point (Compute) and moves towards it (Move). They are identical, silent, **oblivious** (no memory between cycles), and each has its own **local coordinate system**; the **scheduler**, an adversary, decides the **activation** of robots, that is, when each performs a cycle. In the basic version the scheduler is ASYNC (each phase may take any finite time, so a robot may act on an outdated snapshot) and every robot sees all others. The robots must converge: for every distance $\varepsilon > 0$, there must come a time after which all robots stay within ε of each other. Unlike gathering, they never have to meet exactly.

1. Algorithm: Go to the centre of gravity. (Cohen and Peleg 2005) (ASYNC, unlimited visibility, p. 472)

1. Look at the positions of all robots.
2. Compute their **centre of gravity**: the average of all positions (add up the x-coordinates and divide by n, and the same for y).
3. Move towards it.

Why it works and what it costs. The centre of gravity does not depend on the local coordinate system, so robots with different local coordinate systems aim at the same point. In FSYNC (every robot is activated in every **round**) with **rigid moves** (moves that reach their destination), all robots compute the same point and meet in one round. In ASYNC the snapshots are outdated, but each destination is an average of positions that were already inside the **convex hull** of the robots (the smallest convex region containing them), so the convex hull of the robots and their destinations never grows; Cohen and Peleg prove that it actually shrinks to a point (Cohen and Peleg 2005). A later analysis in ASYNC, counting a **round** as a stretch of time in which every robot completes a Look–Compute–Move cycle, shows that with rigid moves $O(n)$ rounds halve the diameter, improving $O(n^2)$, and that this is tight for the centre of gravity (Cord-Landwehr et al. 2011).

Claude (Anthropic) was used in cataloguing the problems.

Why it matters. It is the simplest convergence rule, and the never-growing convex hull is reused as a building block, for example to split robots into groups (Efrima and Peleg 2009).

2. Algorithm: Short sight: go to the centre of what you see, but keep your neighbours. (Ando et al. 1999) (Limited visibility, SSYNC, rigid moves, p. 471) Now each robot sees only the robots within distance V (its **visibility radius**).

1. Compute the **smallest enclosing circle** of the robots you see, and head for its centre.
2. Do not go farther than a fixed maximum step.
3. For every robot you see, stay inside the disc of radius $V/2$ around the midpoint between you and it.

Why it works and what it costs. If two robots that see each other both obey rule 3, both end in the same disc of radius $V/2$, so they are still within V and still see each other (Lemma 2 of Ando et al. 1999). Groups therefore never split, the convex hull of each group never grows, and Ando, Oasa, Suzuki and Yamashita prove that each connected group converges to a point in SSYNC with rigid moves (Theorem 1). The argument uses that a whole cycle happens at once; in ASYNC a robot may move on outdated information and break a visibility link. The catalogue records limited-visibility convergence as solved for every bounded amount of asynchrony and open for full ASYNC (Kirkpatrick et al. 2024).

Why it matters. Its "stay within $V/2$ of the midpoint" trick for keeping neighbours is reused by many limited-visibility algorithms, including Go-To-the-Center gathering (Castenow et al. 2022).

Simulator

A simulator of Look–Compute–Move robots in the plane that runs in the browser. Each case below starts one of its algorithms from a fixed starting configuration and seed; the run is the simulator's, not a proof. On the website each case runs in the page; here it is a link.

- **Go to the centre of gravity.** Each robot moves to the average position of the robots it sees, the rule of (Cohen and Peleg 2005). Here 30 robots see everyone and run under the simulator's asynchronous scheduler. Open in the simulator.
- **Limited visibility: go to the centre, keep your neighbours.** Each robot heads for the centre of the smallest circle enclosing the robots it sees, without losing sight of a neighbour, the rule of (Ando et al. 1999). Six robots with visibility radius 100 start on a chain and move one at a time; the lines join robots that see each other. Open in the simulator.

What the community started looking at

Convergence was introduced alongside gathering as its relaxation (Suzuki and Yamashita 1999). The first questions were **which simple rules converge** under each scheduler (centre of gravity in ASYNC Cohen and Peleg 2005, limited visibility in SSYNC Ando et al. 1999); then the **rate** of convergence (Cord-Landwehr et al. 2011), and how much asynchrony is compatible with limited visibility (Kirkpatrick et al. 2024).

Further reading

Papers on this problem that are not yet placed in a setting.

- Asahiro and Yamashita (2023). Compatibility: robots may run different target functions; a set of them is compatible for a task if every mix solves it. Every set of functions that always shrink the convex hull is

Claude (Anthropic) was used in cataloguing the problems.

compatible for convergence and for convergence with one crash, but no such set is compatible for gathering; fault-tolerant convergence with $f \geq 2$ crashes lies in between. Read (arXiv v7).

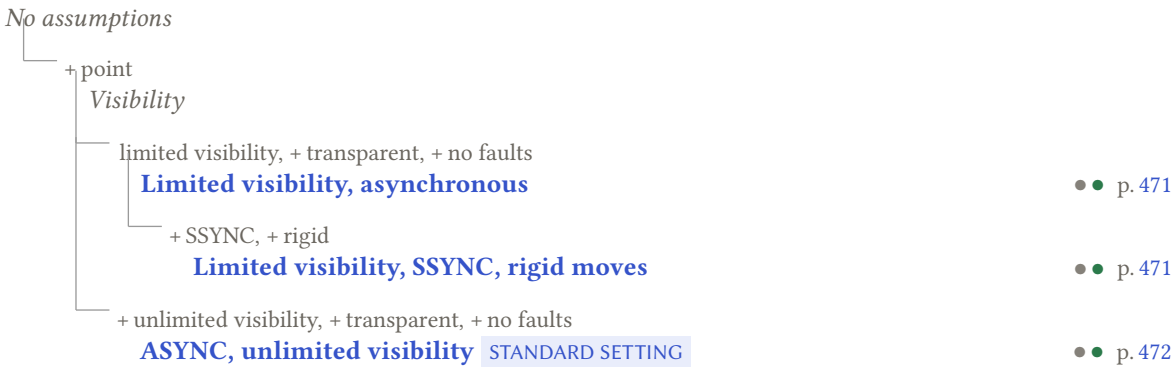
- Pattanayak et al. (2017). Convergence by 'monoculus' robots that cannot measure distances, in two weaker sensing models. Abstract read.
- De Carufel and Flocchini (2017). What the convergence algorithm of Ando et al. does on a line when some robots have crashed undetectably. Abstract read.

Relations to other problems

- **Related: Gathering.** Every gathering algorithm solves convergence, but not conversely: two oblivious robots in SSYNC cannot gather, while "two oblivious robots can converge to a point with a naive algorithm" (Izumi et al. 2012). In ASYNC the centre-of-gravity algorithm converges for any n (Cohen and Peleg 2005), whereas gathering needs extra assumptions such as multiplicity detection.
- **Related: Near-gathering.** Near-gathering is also a relaxation of gathering: robots must get close (within visibility of each other) without ever colliding (Pagli et al. 2015), while convergence lets them approach a point only in the limit.

Case tree

The root assumes nothing; going down, each edge adds the assumptions written in grey (+) or leaves an axis unassumed. Bold entries are catalogued settings with their status per measure (Time, Colours) and page; italic entries are branch points, named after the axis on which their branches differ. Axis order: Body → Visibility → Opacity → Faults → Scheduler → Lights (persistent memory) → Agreement on coordinates → Movement → Multiplicity detection → Randomness → Identifiers.



What is known

The best upper bound against the best lower bound in each catalogued setting, in tree order. The settings follow in detail.

Setting	Time	Colours	p.
Limited visibility, asynchronous	● unknown *finite* vs —	● tight 0 vs —	471
Limited visibility, SSYNC, rigid moves	● unknown *finite* vs —	● tight 0 vs —	471
ASYNC, unlimited visibility STANDARD SETTING	● unknown *finite* vs —	● tight 0 vs —	472

Claude (Anthropic) was used in cataloguing the problems.

Setting	Time	Colours	p.

18.11.1 Limited visibility, asynchronous

Adds limited visibility, + transparent, + no faults VERIFIED

With limited visibility, convergence is solved for every bounded asynchrony (k-ASYNC) but open for full ASYNC.

Assumed: body: point robots; opacity: transparent: every robot sees every robot within range; faults: no faults. **Not assumed:** visibility: limited: robots see only within a fixed radius; scheduler: asynchronous (ASYNC); lights (persistent memory): oblivious, no light (OBLOT); agreement on coordinates: no agreement on local coordinate systems; movement: non-rigid: the adversary may stop a move after some minimum distance δ; multiplicity detection: none: a point with several robots looks like one robot; randomness: deterministic; identifiers: anonymous: all robots are identical.

Measure	Lower bound	Best upper bound	Status
Time	none known Full ASYNC is open.	*finite* (Kirkpatrick et al. 2024)	● unknown
Colours	none known No colours.	0 (Kirkpatrick et al. 2024)	● tight

Results.

- Katreniak (2011) [introductory]: time *finite*, colours 0. *Assumes:* 1-ASYNC. Theorem 7, in 1-bounded asynchrony: while a robot goes from its Look to the end of its move, no other robot makes more than one Look; non-rigid moves, no multiplicity detection. The same paper solves unlimited-visibility ASYNC convergence without multiplicity detection (move halfway towards the furthest robot), and it states that the appendix of its full version (not read) proves that the algorithm of (Ando et al. 1999) fails in 1-bounded asynchrony.
- Kirkpatrick et al. (2024) [current best]: time *finite*, colours 0. *Assumes:* k-ASYNC for any fixed bound on the asynchrony. Also shows that cohesive convergence is impossible in ASYNC for hull-diminishing or coherent algorithms when perception has errors.

Open questions.

Q1. Is point convergence with limited visibility solvable in full ASYNC, without perception or motion errors? (Kirkpatrick et al. 2024)

Q2. Is there a problem that separates k-ASYNC from SSYNC? (Kirkpatrick et al. 2024)

18.11.2 Limited visibility, SSYNC, rigid moves

Adds + SSYNC, + rigid **to** Limited visibility, asynchronous (p. 471) VERIFIED

With limited visibility, semi-synchronous oblivious robots whose moves reach their computed destination converge within each connected component of the visibility graph.

Assumed: body: point robots; opacity: transparent: every robot sees every robot within range; faults: no faults; scheduler: semi-synchronous (SSYNC); movement: rigid: every move reaches its destination. **Not assumed:** visibility: limited: robots see only within a fixed radius; lights (persistent memory): oblivious, no light (OBLOT); agreement on coordinates: no agreement on local coordinate systems; multiplicity detection: none: a point with several robots looks like one robot; randomness: deterministic; identifiers: anonymous: all robots are identical.

Claude (Anthropic) was used in cataloguing the problems.

Measure	Lower bound	Best upper bound	Status
Time	none known Time not analysed here.	*finite* (Ando et al. 1999)	● unknown
Colours	none known No colours.	0 (Ando et al. 1999)	● tight

Results.

- Ando et al. (1999) [current best]: time *finite*, colours 0. *Assumes:* each connected component of the initial visibility graph converges to its own point; each move reaches its computed destination, at most a small constant away; V known (the algorithm keeps visible robots within distance V). Introductory and current best (Theorem 1).

Notes. Checked against the full text of (Ando et al. 1999). The formal model (assumptions A1–A6) has point robots that do not obstruct each other, phases of "negligibly small" duration, so that a robot observes others only when they are stationary (an SSYNC scheduler, although the paper calls the robots asynchronous), every robot active infinitely often, and exact sensing and motion. There is no common coordinate system. A robot moves towards the centre of the smallest enclosing circle of the robots it sees, but no farther than keeps every visible robot within distance V (Lemma 2: mutually visible robots stay mutually visible). Whether the same algorithm works with non-rigid moves is not addressed in the paper.

18.11.3 ASYNC, unlimited visibility

Adds + unlimited visibility, + transparent, + no faults

STANDARD SETTING VERIFIED

Moving to the centre of gravity makes oblivious robots converge in ASYNC, for any n.

Assumed: body: point robots; visibility: unlimited; opacity: transparent: every robot sees every robot within range; faults: no faults. **Not assumed:** scheduler: asynchronous (ASYNC); lights (persistent memory): oblivious, no light (OBLOT); agreement on coordinates: no agreement on local coordinate systems; movement: non-rigid: the adversary may stop a move after some minimum distance δ; multiplicity detection: none: a point with several robots looks like one robot; randomness: deterministic; identifiers: anonymous: all robots are identical.

Measure	Lower bound	Best upper bound	Status
Time	none known See the notes for convergence-time results.	*finite* (Cohen and Peleg 2005)	● unknown
Colours	none known No colours.	0 (Cohen and Peleg 2005)	● tight

Results.

- Cohen and Peleg (2005) [current best]: time *finite*, colours 0. Introductory and current best for ASYNC; also tolerates crash faults (the exact bound on f was not checked).

Notes. Convergence time of the centre-of-gravity algorithm, in epochs (a round of Cord-Landwehr et al. 2011 ends once every robot has completed a cycle): with rigid moves, $O(n)$ epochs halve the diameter of the robots' positions and target points, improving the $O(n^2)$ of (Cohen and Peleg 2005), and the $\Omega(n)$ construction of (Cohen and Peleg 2005) makes this tight for that algorithm; with non-rigid moves the bound is $O(n + n\Delta_0/\delta)$ epochs (Theorem 3 of Cord-Landwehr et al. 2011; their S is δ and their h is Δ_0 here). Their ASYNC model treats Look, Compute and Move as instantaneous operations that other robots' operations may interleave. With axes common to all robots, moving to the centre of the axis-aligned bounding box halves the diameter in $O(1)$ epochs with rigid moves (Theorem 4). Checked against the full text of (Cord-Landwehr et al. 2011).

Claude (Anthropic) was used in cataloguing the problems.

Chapter 19

Symmetry breaking and spreading

Tasks whose difficulty is breaking the symmetry of anonymous robots: electing a leader, scattering from shared points, and separating by colour.

Problems in this family:

- Scattering
- Leader election in the plane
- Separation
- Flocking
- Uniform deployment on a ring



19.1 Scattering

Robots that may start at shared points must move to pairwise distinct points.

First posed in 2009 (Dieudonné and Petit 2009). Dieudonné and Petit showed scattering is deterministically impossible and gave a randomized solution (preprint 2007).

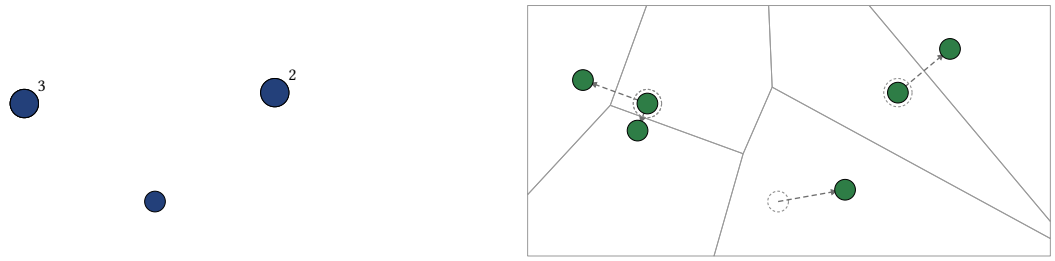
Problem statement and measures

n robots start at points of the plane, several possibly at the same point (a **multiplicity**). They must reach a configuration in which all positions are distinct, and stay there.

Measures: **Time**, rounds (FSYNC, SSYNC) or epochs (ASYNC) until the task is done; "finite" when only solvability is known; **Colours**, colours of the light per robot (0 for oblivious robots).

A first look

Imagine a box of marbles that were all tipped onto the same spot of a table, and you want them to roll apart so that no two touch the same point. In the scattering problem the marbles are n **robots**: they start in the plane, several possibly at the same point (a **multiplicity**), and must reach a **configuration** (the positions of all robots) in which all positions are distinct, and then stay distinct. Robots are **oblivious** (they forget everything after each move) and work in **Look–Compute–Move cycles**: take a **snapshot** of everyone, compute a destination, move. In each **round** an **adversary** (an opponent who controls everything the algorithm does not) chooses which robots act, and each of them performs one full cycle; this is the SSYNC **scheduler**. In the basic catalogued version robots may flip coins (**randomized**), and a robot can see how many robots share each point (**strong multiplicity detection**).



(a) Start: points of multiplicity 3, 2 and 1

(b) Final: all positions distinct (arrows: moves; dashed circles: starting positions)

Figure 19.1: Scattering: robots start at points of the plane, several possibly at the same point (multiplicities shown as numbers), and must reach distinct positions and stay there; deterministically this is impossible (Dieudonné and Petit 2009). The animation runs the randomized SSYNC algorithm of (Dieudonné and Petit 2009): in each round the adversary activates some robots (coin shown under their point), and a robot that gets heads moves to a point of its own Voronoi cell (thin lines), so robots on different points never meet again.

1. Impossibility: Two robots on one spot can never be told apart. (Dieudonné and Petit 2009) (Randomized, SSYNC, strong multiplicity detection, p. 476) Start with all robots on the same point, give them all the same **local coordinate system** (the axes, unit and handedness in which a robot reads its snapshot), and let the adversary activate all of them in every round. Each robot sees exactly the same snapshot, so a **deterministic** rule (one without coin flips) sends every robot to the same destination. After the move they are again all on one point, with identical snapshots, and so on forever.

Why it holds. By induction, there is an execution in which the robots never separate, so no deterministic algorithm solves scattering (Dieudonné and Petit 2009). The argument does not care whether robots remember the past or can count robots at a point: the robots' memories are equal as well, so their histories never differ. Only randomness can make two robots on the same point behave differently.

Why it matters. It is the cleanest example of why some robot problems need randomness at all, and it explains why scattering is studied only with coin flips.

2. Algorithm: Flip a coin, then move inside your own Voronoi cell. (Dieudonné and Petit 2009) (Randomized, SSYNC, strong multiplicity detection, p. 476) Each time a robot is activated:

1. Look at the occupied points and compute the *Voronoi diagram*: the **Voronoi cell** of an occupied point is the set of places strictly closer to it than to any other occupied point.
2. Flip a fair coin.
3. Heads: move to some point of your own cell other than where you stand. Tails: stay.

Why it works and what it costs. Robots on different points never collide again, because each moves only inside its own cell and the cells do not overlap. Two robots on the same point are separated in a round where exactly one of them moves; if both are active this happens with probability $1/2$, so with probability 1 they eventually separate (Dieudonné and Petit 2009). If all movers from a point pick the same destination, a crowd of q robots is at best split in two per round, so about $\log_2 q$ rounds are needed; robots that can count the robots on their point scatter in $O(1)$ expected rounds (Clement et al. 2010), and with weaker counting $O(\log \log n)$ rounds, which is optimal for algorithms that use the fewest random bits (Bramas and Tixeuil 2017).

Why it matters. Moving only inside your own Voronoi cell is a reusable trick for keeping already separated robots apart, and randomized scattering is used to break symmetry before running deterministic algorithms such as

Claude (Anthropic) was used in cataloguing the problems.

pattern formation.

What the community started looking at

Scattering was first a **feasibility** question: it cannot be solved deterministically, so randomness is needed (Dieudonné and Petit 2009). The measures then became **expected rounds and random bits** (Clement et al. 2010; Bramas and Tixeuil 2017), and keeping the swarm connected under limited visibility (Izumi et al. 2018). Léchine and Tixeuil gave the first solutions under ASYNC, with weak multiplicity detection, in a preprint (Léchine and Tixeuil 2019).

Further reading

Papers on this problem that are not yet placed in a setting.

- Mondal and Chaudhuri (2022). Despite the name, a deployment task: oblivious SSYNC robots with unlimited visibility on distinct nodes of an infinite grid spread out to occupy alternate nodes of a square block, without collision. Read (arXiv v1).

Relations to other problems

- **Related: Gathering.** Scattering is the opposite task: gathering creates a multiplicity point, scattering removes all of them. Pattern formation from configurations with multiplicities needs scattering first (Bramas and Tixeuil 2016).
- **Related: Uniform deployment on a ring.** Both spread robots out; scattering only needs distinct positions, deployment needs equal spacing.
- **Related: Scattering of particles.** Robots in the plane scatter to distinct points; particles must scatter so that no two are neighbours.

Case tree

The root assumes nothing; going down, each edge adds the assumptions written in grey (+) or leaves an axis unassumed. Bold entries are catalogued settings with their status per measure (Time, Colours) and page; italic entries are branch points, named after the axis on which their branches differ. Axis order: Body → Visibility → Opacity → Faults → Scheduler → Lights (persistent memory) → Agreement on coordinates → Movement → Multiplicity detection → Randomness → Identifiers.

No assumptions
- + point
  Visibility
  - + unlimited visibility, + transparent, + no faults
    Scheduler
    - + SSYNC
      Multiplicity detection
      - + strong multiplicity, + randomized
        Randomized, SSYNC, strong multiplicity detection STANDARD SETTING ●● p. 476
      - + weak multiplicity, + randomized
        Randomized, SSYNC, weak multiplicity detection ●● p. 477

Claude (Anthropic) was used in cataloguing the problems.

ASYNC, no multiplicity detection, + randomized
Randomized, ASYNC ●● p. 477

ASYNC, + weak multiplicity, + randomized
Randomized, ASYNC, weak multiplicity detection ●● p. 478

limited visibility, + transparent, + no faults
Scheduler

ASYNC, + rigid, + weak multiplicity, + randomized
Randomized, ASYNC, rigid, weak multiplicity detection, no vision needed ●● p. 479

+ SSYNC, + randomized
Limited visibility, connectivity preserving ●● p. 479

What is known

The best upper bound against the best lower bound in each catalogued setting, in tree order. The settings follow in detail.

Setting	Time	Colours	p.
Randomized, SSYNC, strong multiplicity detection STANDARD SETTING	● tight $O(1)$ vs *impossible*	● tight 0 vs —	476
Randomized, SSYNC, weak multiplicity detection	● tight $O(\log \log n)$ vs not $O(1)$	● tight 0 vs —	477
Randomized, ASYNC	● unknown — vs —	● unknown — vs —	477
Randomized, ASYNC, weak multiplicity detection	● unknown *finite* vs —	● tight 0 vs —	478
Randomized, ASYNC, rigid, weak multiplicity detection, no vision needed	● unknown $O(2^{2n^2})$ vs —	● tight 0 vs —	479
Limited visibility, connectivity preserving	● tight $O(\min\{n, g^2 + \log n\})$ vs $\Omega(n)$	● tight 0 vs —	479

19.1.1 Randomized, SSYNC, strong multiplicity detection

Adds + strong multiplicity, + randomized STANDARD SETTING VERIFIED

Deterministic scattering is impossible; with randomness and strong multiplicity detection robots scatter in $O(1)$ expected rounds.

Assumed: body: point robots; visibility: unlimited; opacity: transparent: every robot sees every robot within range; faults: no faults; scheduler: semi-synchronous (SSYNC); multiplicity detection: strong: a robot sees how many robots are at each point; randomness: randomized. **Not assumed:** lights (persistent memory): oblivious, no light (OBLOT); agreement on coordinates: no agreement on local coordinate systems; movement: non-rigid: the adversary may stop a move after some minimum distance δ; identifiers: anonymous: all robots are identical.

Measure	Lower bound	Best upper bound	Status
Time	*impossible* (Dieudonné and Petit 2009) $O(1)$ expected rounds.	$O(1)$ (Clement et al. 2010)	● tight
Colours	none known No colours.	0 (Clement et al. 2010)	● tight

Claude (Anthropic) was used in cataloguing the problems.

Results.

- Dieudonné and Petit (2009) [lower bound]: time *impossible*. *Bottleneck:* symmetry. *Holds when:* deterministic algorithms; robots at the same point are indistinguishable.
- Dieudonné and Petit (2009) [introductory]: time *finite*, colours 0. Randomized and self-stabilizing.
- Clement et al. (2010) [current best]: time $O(1)$, colours 0. Expected rounds; $O(\log n \log\log n)$-type bounds with weak multiplicity detection.

19.1.2 Randomized, SSYNC, weak multiplicity detection

Adds + weak multiplicity, + randomized VERIFIED

Without strong multiplicity detection no algorithm scatters in $O(1)$ expected rounds; with an optimal number of random bits $O(\log\log n)$ rounds suffice.

Assumed: body: point robots; visibility: unlimited; opacity: transparent: every robot sees every robot within range; faults: no faults; scheduler: semi-synchronous (SSYNC); multiplicity detection: weak: a robot sees whether a point holds more than one robot; randomness: randomized. **Not assumed:** lights (persistent memory): oblivious, no light (OBLOT); agreement on coordinates: no agreement on local coordinate systems; movement: non-rigid: the adversary may stop a move after some minimum distance δ; identifiers: anonymous: all robots are identical.

Measure	Lower bound	Best upper bound	Status
Time	not $O(1)$ (Bramas and Tixeuil 2017); $\Omega(\log\log n)$ (Bramas and Tixeuil 2017) $\Theta(\log\log n)$ among bit-optimal algorithms; in general only "not $O(1)$" is known.	$O(\log\log n)$ (Bramas and Tixeuil 2017)	• tight
Colours	none known No colours.	0 (Bramas and Tixeuil 2017)	• tight

Results.

- Bramas and Tixeuil (2017) [lower bound]: time not $O(1)$. *Bottleneck:* symmetry breaking. *Holds when:* Theorem 4.1: no algorithm runs in $O(1)$ expected rounds without strong multiplicity detection.
- Bramas and Tixeuil (2017) [lower bound]: time $\Omega(\log\log n)$. *Bottleneck:* symmetry breaking. *Holds when:* Theorem 4.5: only for algorithms that use the optimal number of random bits.
- Bramas and Tixeuil (2017) [current best]: time $O(\log\log n)$, colours 0. Corollary 4.1, with an optimal number of random bits. The analysis is in FSYNC; the algorithms are valid in SSYNC. Without any multiplicity detection, algorithm SA_ϕ (their SA_f) scatters in $O(\phi(n))$ expected rounds for any increasing surjective function ϕ (Theorem 4.2), but the robots never stop moving.

Notes. $n \log n$ random bits are necessary in any setting (Bramas and Tixeuil 2017). Bramas and Tixeuil split multiplicity detection into local and global. Checked against arXiv 1309.6603 version 2, not the journal version.

19.1.3 Randomized, ASYNC

Adds ASYNC, no multiplicity detection, + randomized DRAFT

Without multiplicity detection, ASYNC scattering in the plane is open; a preprint solves it with weak multiplicity detection, and shows it impossible on the line.

Assumed: body: point robots; visibility: unlimited; opacity: transparent: every robot sees every robot within range; faults: no faults; randomness: randomized. **Not assumed:** scheduler: asynchronous (ASYNC); lights (persistent memory): oblivious, no light (OBLOT); agreement on coordinates: no agreement on local coordinate systems; movement: non-rigid: the adversary may stop a move after some minimum distance δ; multiplicity detection: none: a point with several robots looks like one robot; identifiers: anonymous: all robots are identical.

Claude (Anthropic) was used in cataloguing the problems.

Measure	Lower bound	Best upper bound	Status
Time	none known Open.	—	● unknown
Colours	none known Open.	—	● unknown

Open questions.

Q1. Solve scattering in ASYNC without multiplicity detection: Bramas and Tixeuil call ASYNC scattering "the open problem of ASYNC scattering" (Bramas and Tixeuil 2016, 2017); Léchine and Tixeuil solve it with detection of multiplicity at the robot's own point and leave the plane without detection open (Léchine and Tixeuil 2019).

Notes. On the line, ASYNC scattering without multiplicity detection is impossible even with full vision and rigid moves (Theorem 9 of Léchine and Tixeuil 2019, a preprint); blind robots without multiplicity detection cannot scatter in the plane (Theorem 10).

19.1.4 Randomized, ASYNC, weak multiplicity detection

Adds ASYNC, + weak multiplicity, + randomized VERIFIED

With weak multiplicity detection, oblivious robots scatter in ASYNC with probability 1, even when the adversary stops their moves early.

Assumed: body: point robots; visibility: unlimited; opacity: transparent: every robot sees every robot within range; faults: no faults; multiplicity detection: weak: a robot sees whether a point holds more than one robot; randomness: randomized. **Not assumed:** scheduler: asynchronous (ASYNC); lights (persistent memory): oblivious, no light (OBLOT); agreement on coordinates: no agreement on local coordinate systems; movement: non-rigid: the adversary may stop a move after some minimum distance δ; identifiers: anonymous: all robots are identical.

Measure	Lower bound	Best upper bound	Status
Time	none known	*finite* (Léchine and Tixeuil 2019) [preprint]	● unknown
	Only solvability is known for this algorithm.		
Colours	none known No colours.	0 (Léchine and Tixeuil 2019) [preprint]	● tight

Results.

- Léchine and Tixeuil (2019) [preprint] [current best]: time *finite*, colours 0. *Assumes:* each robot only needs to know whether another robot shares its point. Introductory and current best (Algorithm 3, Theorem 4): a robot that shares its point moves in a random direction chosen from a sequence; the robots need not share a chirality or know n. The paper's non-rigid moves may stop anywhere on the way, a stronger adversary than here.

Open questions.

Q1. Bound the expected time of ASYNC scattering with weak multiplicity detection.

Notes. On the line, any algorithm in which a robot tosses a coin whenever it picks a destination uses $\Omega(n^2)$ coin tosses (Theorem 7 of Léchine and Tixeuil 2019), against $\Theta(n \log n)$ in SSYNC. Checked against the full text of (Léchine and Tixeuil 2019) (arXiv v1, a preprint).

Claude (Anthropic) was used in cataloguing the problems.

19.1.5 Randomized, ASYNC, rigid, weak multiplicity detection, no vision needed

Adds ASYNC, + rigid, + weak multiplicity, + randomized VERIFIED

With rigid moves, robots that see nothing but whether their own point is shared scatter in ASYNC in expected $O(2^{2n^2})$ rounds.

Assumed: body: point robots; opacity: transparent: every robot sees every robot within range; faults: no faults; movement: rigid: every move reaches its destination; multiplicity detection: weak: a robot sees whether a point holds more than one robot; randomness: randomized. **Not assumed:** visibility: limited: robots see only within a fixed radius; scheduler: asynchronous (ASYNC); lights (persistent memory): oblivious, no light (OBLOT); agreement on coordinates: no agreement on local coordinate systems; identifiers: anonymous: all robots are identical.

Measure	Lower bound	Best upper bound	Status
Time	none known No lower bound; the bound is exponential.	$O(2^{2n^2})$ (Léchine and Tixeuil 2019) [preprint]	● unknown
Colours	none known No colours.	0 (Léchine and Tixeuil 2019) [preprint]	● tight

Results.

- Léchine and Tixeuil (2019) [preprint] [current best]: time $O(2^{2n^2})$, colours 0. *Assumes:* blind robots: only whether another robot shares its point. Introductory and current best (Algorithm 1, Theorems 2 and 5): a robot that shares its point moves to the i-th point of a fixed sequence on a circle, with probability 2^{-i}; the bound is on the expected number of rounds. Since the robots ignore the others, the algorithm works under any visibility.

Open questions.

Q1. Is there a polynomial bound on the expected time of blind rigid ASYNC scattering in the plane?

Notes. On the line, blind robots cannot scatter even with rigid moves and weak multiplicity detection (Theorem 8 of Léchine and Tixeuil 2019). Checked against the full text of (Léchine and Tixeuil 2019) (arXiv v1, a preprint).

19.1.6 Limited visibility, connectivity preserving

Adds + SSYNC, + randomized VERIFIED

Robots with limited visibility scatter without ever disconnecting their visibility graph, in $O(\min\{n, g^2+\log n\})$ expected rounds, g the diameter of the initial visibility graph.

Assumed: body: point robots; opacity: transparent: every robot sees every robot within range; faults: no faults; scheduler: semi-synchronous (SSYNC); randomness: randomized. **Not assumed:** visibility: limited: robots see only within a fixed radius; lights (persistent memory): oblivious, no light (OBLOT); agreement on coordinates: no agreement on local coordinate systems; movement: non-rigid: the adversary may stop a move after some minimum distance δ; multiplicity detection: none: a point with several robots looks like one robot; identifiers: anonymous: all robots are identical.

Measure	Lower bound	Best upper bound	Status
Time	$\Omega(n)$ (Izumi et al. 2018); $\Omega(g^2)$ (Izumi et al. 2018) Optimal within conservative algorithms.	$O(\min\{n, g^2 + \log n\})$ (Izumi et al. 2018)	● tight
Colours	none known No colours.	0 (Izumi et al. 2018)	● tight

Results.

Claude (Anthropic) was used in cataloguing the problems.

- Izumi et al. (2018) [lower bound]: time $\Omega(n)$. *Bottleneck:* distance. *Holds when:* any algorithm, some configurations.
- Izumi et al. (2018) [lower bound]: time $\Omega(g^2)$. *Bottleneck:* locality. *Holds when:* only for conservative algorithms.
- Izumi et al. (2018) [current best]: time $O(\min\{n, g^2 + \log n\})$, colours 0. Introduced at SSS 2010.

Notes. g is the diameter (in hops) of the initial visibility graph.

19.2 Leader election in the plane

Anonymous robots must agree on one of them, using only their positions.

First posed in 2007 (Dieudonné et al. 2013). Posed for disoriented robots at OPODIS 2007 (Dieudonné and Petit); characterised in (Dieudonné et al. 2013).

Problem statement and measures

n **anonymous** robots (identical, with no names) at distinct points must agree on a single robot (the **leader**), each deciding from its own snapshot.

Measures: **Time**, rounds (FSYNC, SSYNC) or epochs (ASYNC) until the task is done; "finite" when only solvability is known; **Colours**, colours of the light per robot (0 for oblivious robots).



(a) Start: anonymous robots at distinct points

(b) Final: the robot on it nearest O is the leader

Figure 19.2: Leader election: anonymous robots at distinct points must agree on one robot, each deciding from its own snapshot, without moving. With chirality: draw the smallest enclosing circle, a ray from its centre O through every robot, read the configuration clockwise from each ray as a word (distances along the ray, angle to the next ray); the ray with the unique smallest word carries the leader, its robot nearest O. A leader can be elected exactly when this word is unique, that is, when the configuration has no rotational symmetry (Dieudonné et al. 2013).

A first look

Imagine n identical drones hovering at different spots in a field. They carry no name tags, they cannot talk, and each one has its own private idea of which way is north, what counts as one metre, and possibly even which way is clockwise. In the problem the drones are **robots**, points at distinct places in the plane, and a robot's private north, unit of length and sense of clockwise form its **local coordinate system**. Each robot takes a **snapshot**, a photograph of where all the robots are in its local coordinate system, and, using the same program as every other robot, must decide "I am the leader" or "I am not", so that exactly one robot says yes; that robot is the **leader**. The robots are **anonymous** (they look the same

Claude (Anthropic) was used in cataloguing the problems.

and run the same program) and *disoriented* (no shared compass, unit of length or, without **chirality**, shared sense of clockwise). The question is: for which **configurations** (arrangements of the robots' positions) can this be done at all?

1. Algorithm: Read the configuration as a word around the smallest enclosing circle. (Dieudonné et al. 2013) (Disoriented robots, with or without chirality, p. 482) This is the version where the robots agree on clockwise (chirality).

1. Draw the smallest circle that encloses all robots, the **smallest enclosing circle**; every robot finds the same circle. Call its centre O. If a robot sits exactly at O, it is the leader; stop.
2. Draw a **ray** (a half-line) from O through every robot (several robots may share a ray).
3. For each ray, list the distances between consecutive robots on it, starting from O, measured as fractions of the circle's radius (so the private unit of length does not matter).
4. Starting at a ray and walking clockwise around the circle, write down a **word**: the list for this ray, then the angle to the next ray, then the list for the next ray, and so on, once around.
5. Every robot computes the word of every ray and picks the ray whose word is alphabetically smallest. If only one ray has that word, the leader is the robot on that ray nearest to O.

Why it works and what it costs. All robots compute the same words, because everything is measured relative to the circle and the shared clockwise direction, so they all pick the same robot. The words of different rays are rotations of one another; two rays can have the same word only if turning the whole picture around O moves one ray onto the other, that is, if the configuration has **rotational symmetry**. (Dieudonné et al. 2013) prove that this is the exact borderline: with chirality, a leader can be elected exactly when the smallest word is unique (they phrase it with Lyndon words), and they give a similar but more delicate rule, reading in both directions, when robots do not share clockwise.

Why it matters. Turning a picture into a word and comparing words is the standard way robot papers break symmetry, and electing a leader this way is the first step of arbitrary pattern formation (Dieudonné et al. 2010).

2. Impossibility: Symmetric pictures look the same to everybody. (Dieudonné et al. 2013) (Disoriented robots, with or without chirality, p. 482) Place six robots at the corners of a regular hexagon. The **adversary** (an imagined opponent who chooses everything the algorithm does not control, here the robots' private coordinate systems) gives each robot a local coordinate system whose north points from the centre through that robot, and the same unit and handedness for all. Now every robot's snapshot is literally the same list of coordinates. The same happens for any $n = 2$: two points are always swapped by a half-turn around their midpoint.

Why it holds. A **deterministic** program (no coin flips) that receives the same input gives the same output. So either every robot declares itself leader or none does; exactly one is impossible. The same argument works for any configuration that some rotation maps onto itself while moving robots, which is why the characterisation of (Dieudonné et al. 2013) is phrased in terms of symmetry.

Why it matters. Almost every impossibility for robots in the plane, from pattern formation to gathering, is this argument: build a symmetric start and choose coordinate systems so that symmetric robots see identical snapshots.

What the community started looking at

Leader election in the plane was asked as **a characterisation of the configurations from which anonymous robots can agree on one of them** (Dieudonné et al. 2013); its importance comes from its equivalence with arbitrary pattern formation (Dieudonné et al. 2010; Cicerone et al. 2019).

Claude (Anthropic) was used in cataloguing the problems.

Relations to other problems

- **Related: Pattern formation.** Arbitrary pattern formation is solvable from a configuration exactly when leader election is, in ASYNC without chirality (Cicerone et al. 2019); with chirality and $n \geq 4$ this was shown by (Dieudonné et al. 2010).
- **Related: Leader election in programmable matter.** Both elect a leader among anonymous entities from geometry alone; particles on the lattice can also move and talk to neighbours, robots in the plane only see (Dufoulon et al. 2021).

Case tree

The root assumes nothing; going down, each edge adds the assumptions written in grey (+) or leaves an axis unassumed. Bold entries are catalogued settings with their status per measure (Time, Colours) and page; italic entries are branch points, named after the axis on which their branches differ. Axis order: Body → Visibility → Opacity → Faults → Scheduler → Lights (persistent memory) → Agreement on coordinates → Movement → Multiplicity detection → Randomness → Identifiers.

No assumptions

+ point, + unlimited visibility, + transparent, + no faults
Disoriented robots, with or without chirality STANDARD SETTING ●● p. 482

fat, + unlimited visibility, + transparent
Transparent fat robots, no chirality ●● p. 483

What is known

The best upper bound against the best lower bound in each catalogued setting, in tree order. The settings follow in detail.

Setting	Time	Colours	p.
Disoriented robots, with or without chirality STANDARD SETTING	● unknown *finite* vs *impossible*	● tight 0 vs —	482
Transparent fat robots, no chirality	● unknown *finite* vs *impossible*	● tight 0 vs —	483

19.2.1 Disoriented robots, with or without chirality

Adds + point, + unlimited visibility, + transparent, + no faults STANDARD SETTING VERIFIED

Whether anonymous robots can agree on a leader from their positions is characterised by Lyndon words around the smallest enclosing circle.

Assumed: body: point robots; visibility: unlimited; opacity: transparent: every robot sees every robot within range; faults: no faults. **Not assumed:** scheduler: asynchronous (ASYNC); lights (persistent memory): oblivious, no light (OBLOT); agreement on coordinates: no agreement on local coordinate systems; movement: non-rigid: the adversary may stop a move after some minimum distance δ; multiplicity detection: none: a point with several robots looks like one robot; randomness: deterministic; identifiers: anonymous: all robots are identical.

Claude (Anthropic) was used in cataloguing the problems.

Measure	Lower bound	Best upper bound	Status
Time	*impossible* (Dieudonné et al. 2013) Leader election here is a static decision.	*finite* (Dieudonné et al. 2013)	● unknown
Colours	none known No colours.	0 (Dieudonné et al. 2013)	● tight

Results.

- Dieudonné et al. (2013) [lower bound]: time *impossible*. *Bottleneck:* symmetry. *Holds when:* $n = 2$; symmetric configurations.
- Dieudonné et al. (2013) [current best]: time *finite*, colours 0. Complete characterisation for every $n > 1$, with and without chirality. Introduced at OPODIS 2007 by Dieudonné and Petit.

19.2.2 Transparent fat robots, no chirality

Adds fat, + unlimited visibility, + transparent VERIFIED

Transparent fat robots without chirality can agree on an ordering, and so on a leader, exactly when their configuration is asymmetric.

Assumed: visibility: unlimited; opacity: transparent: every robot sees every robot within range. **Not assumed:** body: fat robots (unit discs); faults: up to f Byzantine robots; scheduler: asynchronous (ASYNC); lights (persistent memory): oblivious, no light (OBLOT); agreement on coordinates: no agreement on local coordinate systems; movement: non-rigid: the adversary may stop a move after some minimum distance δ; multiplicity detection: none: a point with several robots looks like one robot; randomness: deterministic; identifiers: anonymous: all robots are identical.

Measure	Lower bound	Best upper bound	Status
Time	*impossible* (Chaudhuri and Mukhopadhyaya 2015) Solvable exactly on asymmetric configurations.	*finite* (Chaudhuri and Mukhopadhyaya 2015)	● unknown
Colours	none known No colours.	0 (Chaudhuri and Mukhopadhyaya 2015)	● tight

Results.

- Chaudhuri and Mukhopadhyaya (2015) [lower bound]: time *impossible*. *Bottleneck:* symmetry breaking. *Holds when:* symmetric configurations (Theorem 6); deterministic.
- Chaudhuri and Mukhopadhyaya (2015) [current best]: time *finite*, colours 0. Introductory and current best (Theorems 1–6, algorithm MakeAsym).

Notes. Checked against the full text (arXiv 1208.4484).

19.3 Separation

Robots of different (hidden) colours must separate so that each colour class occupies its own region.

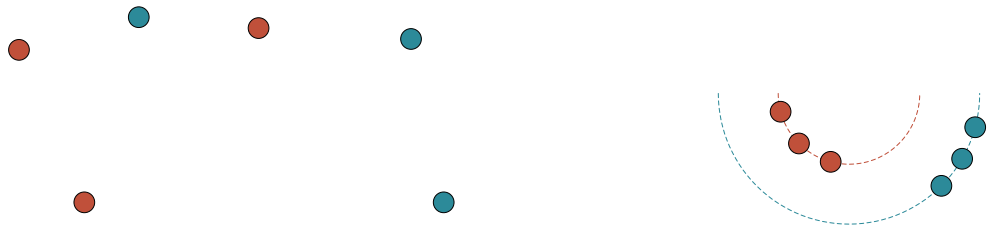
First posed in 2023 (Seike and Yamauchi 2023). Introduced at SSS 2023.

Claude (Anthropic) was used in cataloguing the problems.

Problem statement and measures

Each robot has a fixed **colour** that the others can see but it cannot (an **unconscious** robot). The robots must arrange themselves so that the **colour classes** (the sets of robots of one colour) are separated (into points, lines or circles, depending on the variant).

Measures: **Time**, rounds (FSYNC, SSYNC) or epochs (ASYNC) until the task is done; "finite" when only solvability is known; **Colours**, colours of the light per robot (0 for oblivious robots).



(a) Start: robots see the others' colours, not their own (b) Final: one semicircle per colour

Figure 19.3: Separation: each robot carries a fixed colour that the others see but it cannot; the robots must arrange themselves so that the colour classes are separated, into parallel lines in the standard ASYNC setting (Flocchini et al. 2025). The animation sketches the SSYNC algorithm for opaque robots (Pyati et al. 2026) (read in summary only): the robots move to a semicircle, one sector per colour; a signaller points at a misplaced robot, which then moves to its sector; finally each colour class moves onto its own concentric semicircle.

A first look

Imagine a party where each guest wears a coloured hat they cannot see, but they can see everybody else's hat, and nobody may speak. The guests must sort themselves so that all red hats stand on one line, all blue hats on another, and so on. In the separation problem the guests are n point **robots** in the plane, and a hat is a robot's **colour**: fixed, seen by the others but not by the robot itself (an **unconscious** robot). Each robot repeats a **Look–Compute–Move cycle**: it looks at the others, computes a destination and moves there. In the basic version robots are **oblivious** (they forget everything after each move), see everybody, and their **local coordinate systems** (each robot's private axes and unit of length) agree on the direction of one axis; the **scheduler**, an adversary that decides when each robot acts, is asynchronous (ASYNC): a robot may look while others are still moving. They must end up separated into parallel lines, one **colour class** (all robots of one colour) per line.

1. Algorithm: Others tell you where to go. (Pyati et al. 2026) (Opaque robots, SSYNC, p. 486) The key difficulty is that a robot does not know its own colour, but every other robot does. One way out, used with **opaque** robots (robots that block the view of robots behind them) under SSYNC (time is divided into **rounds**, and in each round some robots each perform one full cycle):

1. Bring all robots to the lower half of a circle, where everybody can see everybody.
2. Divide this semicircle into **sectors**, one per colour.
3. A few chosen robots act as **signallers**: by moving to a carefully chosen point they encode "the robot of colour c at this position should go to sector j".
4. The robot addressed reads the signal and moves to its sector, without ever learning its colour directly.

Claude (Anthropic) was used in cataloguing the problems.

5. Finally each sector is moved onto its own semicircle, so colour classes lie on concentric semicircles.

Why it works and what it costs. Positions are the only way robots can pass information, so the algorithm turns positions into messages. (Pyati et al. 2026) reach concentric semicircles in $O(n)$ epochs (an **epoch** is an interval of time in which every robot completes at least one full cycle) without collisions and without knowing n. We read only a summary of this paper's arXiv version, so the steps above give the idea, not its details. The algorithms for the standard ASYNC variant are different and are not described here.

Why it matters. Using movement as a message channel is how silent robots share information they cannot sense themselves, the same idea behind signalling with lights.

2. Impossibility: A colour with only one robot. (Flocchini et al. 2025) (ASYNC, one common axis, p. 486) Asynchronous separation into parallel lines is solvable when robots share one axis, provided no colour class has a single robot. If some robot is the only one of its colour, (Flocchini et al. 2025) show that no algorithm works. The proof is a symmetry argument in the paper and is not reproduced here.

Why it holds. The catalogue records the condition but not its proof, and we have not read the full text. A student should take away the shape of the result: solvability depends on how many robots share each colour, because a robot can learn about its own colour only from what the other robots do.

Why it matters. It marks the exact boundary between possible and impossible for the asynchronous algorithm of (Flocchini et al. 2025).

What the community started looking at

Separation of unconscious coloured robots asked whether robots that **cannot see their own colour** can still sort themselves by colour, first synchronously (Seike and Yamauchi 2023), then asynchronously (Flocchini et al. 2025) and with opaque robots (Pyati et al. 2026).

Relations to other problems

- **Related: k-grouping.** Both split the swarm into classes; in separation the classes are fixed by hidden colours, in k-grouping and partitioning they are chosen by the robots.
- **Related: Separation of coloured particles.** Robots in the plane separate by hidden colours with deterministic algorithms; here a stochastic local rule separates two visible colours.

Case tree

The root assumes nothing; going down, each edge adds the assumptions written in grey (+) or leaves an axis unassumed. Bold entries are catalogued settings with their status per measure (Time, Colours) and page; italic entries are branch points, named after the axis on which their branches differ. Axis order: Body → Visibility → Opacity → Faults → Scheduler → Lights (persistent memory) → Agreement on coordinates → Movement → Multiplicity detection → Randomness → Identifiers.

No assumptions

+ point, + unlimited visibility

Opacity

+ transparent, + no faults, + one axis

ASYNC, one common axis STANDARD SETTING ●● p. 486

Claude (Anthropic) was used in cataloguing the problems.

opaque, + no faults, + SSYNC, + one axis
Opaque robots, SSYNC ●● p. 486

What is known

The best upper bound against the best lower bound in each catalogued setting, in tree order. The settings follow in detail.

Setting	Time	Colours	p.
ASYNC, one common axis STANDARD SETTING	● unknown *finite* vs *impossible*	● tight 0 vs —	486
Opaque robots, SSYNC	● unknown $O(n)$ vs —	● tight 0 vs —	486

19.3.1 ASYNC, one common axis

Adds + transparent, + no faults, + one axis STANDARD SETTING VERIFIED

Robots that carry a colour visible to others but not to themselves separate by colour into parallel lines, under ASYNC, if they share one axis.

Assumed: body: point robots; visibility: unlimited; opacity: transparent: every robot sees every robot within range; faults: no faults; agreement on coordinates: agreement on one axis (no common chirality). **Not assumed:** scheduler: asynchronous (ASYNC); lights (persistent memory): oblivious, no light (OBLOT); movement: non-rigid: the adversary may stop a move after some minimum distance δ; multiplicity detection: none: a point with several robots looks like one robot; randomness: deterministic; identifiers: anonymous: all robots are identical.

Measure	Lower bound	Best upper bound	Status
Time	*impossible* (Flocchini et al. 2025) —	*finite* (Flocchini et al. 2025)	● unknown
Colours	none known The colours are given, not chosen.	0 (Flocchini et al. 2025)	● tight

Results.

- Flocchini et al. (2025) [lower bound]: time *impossible*. *Bottleneck:* symmetry. *Holds when:* if some robot is uniquely coloured.
- Seike and Yamauchi (2023) [introductory]: time *finite*, colours 0. *Assumes:* synchronous setting. As reported by (Flocchini et al. 2025); the SSS 2023 paper was not read.
- Flocchini et al. (2025) [current best]: time *finite*, colours 0. Separation into parallel lines.

19.3.2 Opaque robots, SSYNC

Adds opaque, + no faults, + SSYNC, + one axis VERIFIED

Opaque unconscious robots separate into concentric semicircles in $O(n)$ epochs.

Assumed: body: point robots; visibility: unlimited; faults: no faults; scheduler: semi-synchronous (SSYNC); agreement on coordinates: agreement on one axis (no common chirality). **Not assumed:** opacity: opaque: collinear robots hide each other; lights (persistent memory): oblivious, no light (OBLOT); movement: non-rigid: the adversary may stop a move after some minimum distance δ; multiplicity detection: none: a point with several robots looks like one robot; randomness: deterministic; identifiers: anonymous: all robots are identical.

Claude (Anthropic) was used in cataloguing the problems.

Measure	Lower bound	Best upper bound	Status
Time	none known No lower bound recorded.	$O(n)$ (Pyati et al. 2026)	● unknown
Colours	none known —	0 (Pyati et al. 2026)	● tight

Results.

- Pyati et al. (2026) [current best]: time $O(n)$, colours 0. Collision-free; robots do not know n.

19.4 Flocking

Robots must move together as a group, following a leader while roughly keeping a formation.

First posed in 2004 (Gervasi and Prencipe 2004). Coordination without communication: the flocking problem.

Problem statement and measures

Robots, the **followers**, follow a designated robot, the **leader**, whose path is not known in advance, while keeping (approximately) a given **formation** (a shape the robots must hold, placed relative to the leader) and avoiding **collisions** (two robots at the same position).

Measures: **Time**, rounds (FSYNC, SSYNC) or epochs (ASYNC) until the task is done; "finite" when only solvability is known; **Colours**, colours of the light per robot (0 for oblivious robots).



(a) Start: followers scattered behind the leader L

(b) Final: L stops; the followers hold the V

Figure 19.4: Flocking: the robots follow a leader (L) whose path is not known in advance, keeping approximately a given formation and avoiding collisions (Gervasi and Prencipe 2004). The animation shows the basic slot-chasing idea (not the method of Gervasi and Prencipe 2004): in each round the leader moves; each follower places the V formation behind the leader's last observed heading, takes the nearest free slot (dashed circles) and moves towards it at a speed higher than the leader's.

A first look

Think of a flock of birds following a leader whose route nobody knows in advance, keeping roughly the same shape and never bumping into each other. In the flocking problem the birds are **robots** in the plane: the **followers** must follow a designated **leader** robot along an unpredictable path while keeping approximately a given **formation**, a shape placed relative to the leader (for example, a V shape), and avoiding **collisions** (two robots at the same position). The robots are **oblivious** (they forget

Claude (Anthropic) was used in cataloguing the problems.

everything after each move) and silent: they coordinate only by looking at each other's positions. The catalogue records only the first formal treatment (Gervasi and Prencipe 2004) and has not checked its exact assumptions, so this page describes the problem and the simplest reasoning about it.

1. Algorithm: Keep your slot relative to the leader. (Oblivious robots following a leader, p. 489) A standard approach, assuming the followers can recognise the leader and know the target formation:

1. Look at all robots and find the leader.
2. Place the target formation with the leader at its tip, oriented, say, along the leader's last observed direction of motion.
3. Choose the free **slot** of the formation (a position in it reserved for one follower) that is closest to you (with a fixed tie-break rule).
4. Move towards that slot, but never closer to another robot than a safety distance.

Why it works and what it costs. Each follower chases a target that moves with the leader, so if followers are fast enough they stay within a bounded distance of their slots. The details that make this hard are exactly those the model hides: an oblivious follower cannot remember the leader's past direction, and slot assignment must not change from look to look. We do not attribute this sketch to (Gervasi and Prencipe 2004), whose method we have not read.

Why it matters. Reducing a moving formation to "chase your slot" is the basic design idea of leader-follower formation control.

2. Lower bound: Followers must be faster than the leader. (Oblivious robots following a leader, p. 489) Suppose followers can move no faster than the leader. Let the leader run at full speed in a straight line. A follower that is a distance $\eta > 0$ behind its slot moves at most as fast as the slot moves, so the gap never shrinks.

Why it holds. So exact formation keeping is impossible in general with equal top speeds; a follower can only stay within the error it started with. This simple observation, our own and not taken from a paper, is why flocking asks for approximate formations or gives followers extra speed.

Why it matters. It explains why "approximately" in the problem statement is not a detail but a necessity.

What the community started looking at

Flocking was posed as a **coordination problem without communication** for oblivious robots (Gervasi and Prencipe 2004); the questions were feasibility and, later, fault tolerance and self-stabilisation.

Case tree

The root assumes nothing; going down, each edge adds the assumptions written in grey (+) or leaves an axis unassumed. Bold entries are catalogued settings with their status per measure (Time, Colours) and page; italic entries are branch points, named after the axis on which their branches differ. Axis order: Body → Visibility → Opacity → Faults → Scheduler → Lights (persistent memory) → Agreement on coordinates → Movement → Multiplicity detection → Randomness → Identifiers.

No assumptions

+ point, + unlimited visibility, + transparent, + no faults

Oblivious robots following a leader STANDARD SETTING ●● p. 489

Claude (Anthropic) was used in cataloguing the problems.

What is known

The best upper bound against the best lower bound in each catalogued setting, in tree order. The settings follow in detail.

Setting	Time	Colours	p.
Oblivious robots following a leader STANDARD SETTING	● unknown *finite* vs —	● tight 0 vs —	489

19.4.1 Oblivious robots following a leader

Adds + point, + unlimited visibility, + transparent, + no faults STANDARD SETTING DRAFT

The first formal treatment of flocking by oblivious robots that communicate only by observing each other.

Assumed: body: point robots; visibility: unlimited; opacity: transparent: every robot sees every robot within range; faults: no faults. **Not assumed:** scheduler: asynchronous (ASYNC); lights (persistent memory): oblivious, no light (OBLOT); agreement on coordinates: no agreement on local coordinate systems; movement: non-rigid: the adversary may stop a move after some minimum distance δ; multiplicity detection: none: a point with several robots looks like one robot; randomness: deterministic; identifiers: anonymous: all robots are identical.

Measure	Lower bound	Best upper bound	Status
Time	none known —	*finite* (Gervasi and Prencipe 2004)	● unknown
Colours	none known No colours.	0 (Gervasi and Prencipe 2004)	● tight

Results.

- Gervasi and Prencipe (2004) [current best]: time *finite*, colours 0. Introductory; the exact assumptions and guarantees were not read in the paper.

Notes. Self-stabilising and crash-tolerant flocking followed (SSS 2007, OPODIS 2008; not yet in the bibliography).

19.5 Uniform deployment on a ring

Robots on a circle must spread out to equal spacing.

First posed in 2006 (Flocchini et al. 2008b). Self-deployment of mobile sensors on a ring (ALGOSENSORS 2006).

Problem statement and measures

n robots on a **ring** (a circle of length L) must reach positions at equal distance $d = L/n$ from their **neighbours** (the next robots along the ring in either direction) and stop. Spreading on a segment is the one-dimensional analogue.

Measures: **Time**, rounds (FSYNC, SSYNC) or epochs (ASYNC) until the task is done; "finite" when only solvability is known; **Colours**, colours of the light per robot (0 for oblivious robots).

Claude (Anthropic) was used in cataloguing the problems.



(a) Start: oriented ring, $n = 6$, robots know $d = L/6$ (b) Final: every gap equals d; nobody moves

Figure 19.5: Uniform deployment on a ring: robots on a circle of length L must reach positions at distance exactly $d = L/n$ from their neighbours and stop (Flocchini et al. 2008b). The animation runs their rule for an oriented ring with known d: an activated robot whose clockwise gap (red) exceeds d moves clockwise along the ring until it is exactly d behind the next robot; here the asynchronous adversary activates one robot at a time.

A first look

Imagine n guards spread at random along a circular wall who must spread out so that every two neighbours are the same distance apart. In **uniform deployment** the guards are n **robots** at distinct points of a **ring**, a circle of length L. Consecutive robots along the ring are **neighbours**, and the **gap** of a robot is the distance along the ring to its clockwise neighbour. The robots must reach positions where every gap is exactly $d = L/n$, and then stop. In the basic version robots are **oblivious** (they forget everything after each move), the **scheduler** is asynchronous (ASYNC: an **adversary**, an imagined opponent, decides the **activation** of each robot, that is, when it looks and moves, and a robot may see others in mid-move), and robots agree on which way is clockwise (an **oriented ring**).

1. Algorithm: Close the gap in front of you, only clockwise. (Flocchini et al. 2008b) (Oriented ring, asynchronous, oblivious, p. 492) Robots know d. Each time a robot is activated:

1. Look clockwise and measure the distance to the next robot.
2. If that distance is at most d, do nothing.
3. If it is more than d, move clockwise until you are exactly d behind that robot.

Why it works and what it costs. All gaps add up to $L = nd$, so if some gap is smaller than d, another is larger. Robots only move clockwise, so the robot in front never comes closer; a robot whose gap is at least d therefore never again has a gap below d, and the number of such robots never decreases. (Flocchini et al. 2008b) prove that this protocol (Uniform Known) reaches exact deployment in finite time, even with limited visibility $2d$; when d is unknown, a "go to the half" rule reaches every gap within any ε of d. No running-time bound is recorded in the catalogue.

Why it matters. It shows that a shared sense of direction plus one number (d) turns an impossible task into a three-line rule.

2. Impossibility: Two mirror-image teams that always overshoot. (Flocchini et al. 2008b) (Oriented ring, asynchronous, oblivious, p. 492) Without a shared clockwise direction exact deployment is impossible, even with full memory and SSYNC activation (time is divided into **rounds**; in each round the adversary activates some robots, and each looks and moves once). Take $n = 4$: black robots at two opposite points, white robots at two opposite points, the white pair rotated by a small angle from the uniform position; the two black robots form one **team** and the two white robots the other. Black robots think clockwise is

Claude (Anthropic) was used in cataloguing the problems.

one way, white robots think it is the other, so each black robot sees exactly what each white robot sees, in mirror image. The adversary then plays:

1. If moving only the blacks would not finish the job, move only the blacks (the **configuration**, the positions of all robots, keeps the same two-team shape).
2. If moving only the blacks would finish, then by the mirror image moving only the whites would also finish; so move both. Each team closes the same gap, together they overshoot, and again the configuration has the same shape.
3. Alternate between starting with the blacks and the whites, so every robot moves infinitely often.

Why it holds. Every round leaves the robots in two rotated teams, never equally spaced, so the algorithm never finishes (Flocchini et al. 2008b). The argument works for every even n with two interleaved regular $n/2$-gons. It is why the catalogued variant assumes an oriented ring.

Why it matters. It is a model example of an adversary that exploits mirror symmetry and the freedom to choose who moves.

What the community started looking at

Self-deployment asked **under which assumptions robots can spread uniformly** around a ring (Flocchini et al. 2008b) or along a segment (Cohen and Peleg 2008): orientation and knowledge of the target distance turned out to be the deciding factors.

Further reading

Papers on this problem that are not yet placed in a setting.

- Maiti et al. (2026c). Min-sum uniform deployment: the robots start at distinct points of a given segment or of a given circle (the ring), move only along it, and must reach equally spaced positions (on the segment the points at $(i-1/2)/n$ of its length from one end, $i = 1, \dots, n$; on the ring every gap equal to d) so that the total distance travelled is as small as possible. The robots are oblivious, agree on no direction (not even on the orientation of the ring), have no multiplicity detection and run under ASYNC with non-rigid moves. On the segment one robot with a free path moves at a time (Lemmas 2, 3). On the ring an optimal solution keeps one robot fixed, as in the centralised result of Bhattacharya et al.; the problem is solvable from every initial configuration except the symmetric ones in which no robot whose fixing is optimal lies on a line of symmetry (Theorem 1). The impossibility is proved for a single line of symmetry with no robot on it (Lemma 10) and only sketched for several lines. Read (arXiv v1).

Relations to other problems

- **Related: Scattering.** Both spread robots out; scattering only needs distinct positions, deployment needs equal spacing.
- **Related: Uniform deployment on graphs.** The same task for robots on a continuous ring in the plane, where the target spacing is a length rather than a number of edges (Flocchini et al. 2008b).

Claude (Anthropic) was used in cataloguing the problems.

Case tree

The root assumes nothing; going down, each edge adds the assumptions written in grey (+) or leaves an axis unassumed. Bold entries are catalogued settings with their status per measure (Time, Colours) and page; italic entries are branch points, named after the axis on which their branches differ. Axis order: Body → Visibility → Opacity → Faults → Scheduler → Lights (persistent memory) → Agreement on coordinates → Movement → Multiplicity detection → Randomness → Identifiers.

No assumptions

+ point, + unlimited visibility, + transparent, + no faults, + chirality

Oriented ring, asynchronous, oblivious STANDARD SETTING ●● p. 492

What is known

The best upper bound against the best lower bound in each catalogued setting, in tree order. The settings follow in detail.

Setting	Time	Colours	p.
Oriented ring, asynchronous, oblivious STANDARD SETTING	● unknown *finite* vs *impossible*	● tight 0 vs —	492

19.5.1 Oriented ring, asynchronous, oblivious

Adds + point, + unlimited visibility, + transparent, + no faults, + chirality STANDARD SETTING VERIFIED

On an oriented ring oblivious asynchronous robots deploy exactly if they know d, and within any ε if they do not.

Assumed: body: point robots; visibility: unlimited; opacity: transparent: every robot sees every robot within range; faults: no faults; agreement on coordinates: common chirality (same clockwise orientation). **Not assumed:** scheduler: asynchronous (ASYNC); lights (persistent memory): oblivious, no light (OBLOT); movement: non-rigid: the adversary may stop a move after some minimum distance δ; multiplicity detection: none: a point with several robots looks like one robot; randomness: deterministic; identifiers: anonymous: all robots are identical.

Measure	Lower bound	Best upper bound	Status
Time	*impossible* (Flocchini et al. 2008b) —	*finite* (Flocchini et al. 2008b)	● unknown
Colours	none known No colours.	0 (Flocchini et al. 2008b)	● tight

Results.

- Flocchini et al. (2008b) [lower bound]: time *impossible*. *Bottleneck:* symmetry. *Holds when:* exact deployment without orientation, even semi-synchronous.
- Flocchini et al. (2008b) [current best]: time *finite*, colours 0. *Assumes: d* known for exact deployment. Introductory and current best; ε-approximate deployment without knowing d.

Open questions.

Q1. Is knowledge of d necessary for exact deployment on an oriented ring? Which capabilities allow asynchronous exact deployment on unoriented rings? (Flocchini et al. 2008b)

Claude (Anthropic) was used in cataloguing the problems.

SEARCH

Notes. The chirality value stands for an oriented ring. Spreading on a segment: (Cohen and Peleg 2008).

Claude (Anthropic) was used in cataloguing the problems.

SEARCH

Part V

Search in geometric domains



The model and its assumption axes are in Chapter 4.

Claude (Anthropic) was used in cataloguing the problems.

SEARCH

Chapter 20

Search, evacuation and patrolling

Find a target, or get every robot out through an exit, at an unknown point of a known continuous domain. Evacuation builds on linear search (the cow-path problem).

Problems in this family:

- Linear search (cow path)
- Evacuation — generalisation of Linear search (cow path)
- Chauffeuring a crashed robot — variant of Evacuation
- Search for a moving target — generalisation of Linear search (cow path)
- Patrolling
- Search with segment visibility



20.1 Linear search (cow path)

A robot on a line (or on m rays from a point) must find a target at unknown distance; how small can the competitive ratio be?

First posed in 1964 (Beck 1964). Bellman posed the problem in 1963 and Beck analysed it in 1964; computer scientists rediscovered it in the late 1980s as the cow-path problem (Kupavskii and Welzl 2021).

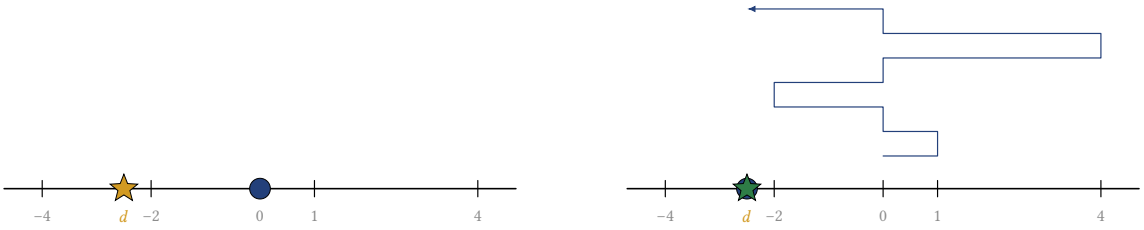
Problem statement and measures

One or more unit-speed robots start at the origin of a line (or of m rays sharing an endpoint). A target lies at an unknown point at distance $d \geq 1$. **Time** is the competitive ratio: the worst case, over target positions, of the time until the target is found (in **search**: by the first robot; in **evacuation**: by all robots) divided by d.

Measures: **Competitive ratio**, worst-case time divided by the distance to the target.

A first look

A cow stands at a gate on an endless straight fence and knows that a hole is somewhere along the fence, but not on which side or how far. She walks at unit speed and sees the hole only when she reaches it. In linear search the cow is a **robot**, a point that moves at unit speed and sees the target only when it reaches it; the gate is the **origin**, the point 0 of an infinite line where the robot starts; and the hole is the **target**, at an unknown distance $d \geq 1$ on an unknown side. We measure the **competitive ratio**: the worst case, over all target positions, of (time to find the target) divided by d. A competitive ratio of 9 means the robot never walks more than about nine times the distance it would walk if it knew where the target was.



(a) Start: robot at 0; the target is somewhere

(b) Final: found at time 16.5 = 6.6 d (worst case < 9d)

Figure 20.1: Linear search (cow path): a robot starts at the origin of a line and must reach a target at unknown distance $d \geq 1$ on an unknown side. The animation runs the doubling strategy at unit speed, with turning points $1, -2, 4, -8, \ldots$ (trace above the line, one level per trip); the target, at $d = 2.5$ just beyond the turning point -2, is found on the next trip to the left at time $16.5 = 6.6\,d$. In the worst case the strategy finds the target within time $9d$, which is optimal (Baeza-Yates et al. 1993).

1. Algorithm: Doubling: go 1 right, 2 left, 4 right, ... (Baeza-Yates et al. 1993) (One robot on a line, p. 501) A **strategy** is the rule that fixes where the robot walks. The doubling strategy:

1. Set $i = 0$.
2. **Trip** i: walk distance 2^i to one side (right when i is even, left when i is odd), checking for the target; the point where the robot turns back is a **turning point**.
3. If the target is not there, walk back to the origin; this ends trip i.
4. Increase i by one and repeat.

Why it works and what it costs. The worst place for the target is just past a turning point, say at $d = 2^i + \varepsilon$: the robot turned back there and finds it only on the next trip to that side, trip $i+2$. By then it has walked $2(1 + 2 + \cdots + 2^{i+1}) = 2(2^{i+2} - 1) < 8 \cdot 2^i < 8d$, plus the final d, in total less than $9d$. So the competitive ratio is at most 9, and (Baeza-Yates et al. 1993) prove that this is optimal for **deterministic** strategies, those that toss no coins.

Why it matters. Doubling is the universal answer to "I do not know how far it is": it reappears in search on rays, in online algorithms, and in algorithms that do not know n.

2. Lower bound: Why no deterministic strategy beats 9. (Baeza-Yates et al. 1993) (One robot on a line, p. 501) Any deterministic strategy can be described by its turning points $x_1, x_2, x_3, \ldots$, alternating sides, and one can show it is enough to consider growing ones. The **adversary**, an opponent who knows the strategy, puts the target just beyond x_i, on that side. The robot finds it only after the next turning point on the other side, so its time is at least $2(x_1 + \cdots + x_{i+1}) + x_i$. Write $S_j = x_1 + \cdots + x_j$. A competitive ratio at most $1 + 2c$ for every i needs $S_{i+1} \leq c\,x_i = c(S_i - S_{i-1})$.

Why it holds. Let $q_i = S_{i+1}/S_i$, which is greater than 1. Dividing the inequality by S_i gives $q_i \leq c - c/q_{i-1}$. Because $q + c/q \geq 2\sqrt{c}$ for every $q > 0$, we get $q_i \leq q_{i-1} - (2\sqrt{c} - c)$. If $c < 4$, the amount $2\sqrt{c} - c$ is a fixed positive number, so q_i drops by at least that much at every step and eventually falls below 1, a contradiction. Hence $c \geq 4$ and the competitive ratio is at least $1 + 2 \cdot 4 = 9$, matching doubling (up to small terms in d) (Baeza-Yates et al. 1993), (Beck 1964).

Why it matters. It is the prototype adversary argument for online search: hide the target just past a turning point of the robot.

Claude (Anthropic) was used in cataloguing the problems.

3. Algorithm: Randomize the first side and the scale. (Kao et al. 1996) (One randomized robot on a line, p. 501) A **randomized** strategy tosses coins:

1. Toss a coin to choose which side to try first.
2. Choose a random distance for the first turning point.
3. From then on, multiply the distance of the turning point by the same fixed factor on every trip, alternating sides.

Why it works and what it costs. The adversary, who must fix the target before the coins are tossed, no longer knows where the turning points are, so it cannot always place the target just past one. The expected competitive ratio drops to about 4.5911, and (Kao et al. 1996) prove that no randomized strategy does better on the line. On m **rays** (half-lines that share the origin as their endpoint) the optimal deterministic competitive ratio is $1 + 2m^m/(m-1)^{m-1}$ (Baeza-Yates et al. 1993), which gives 9 for $m = 2$.

Why it matters. It shows how randomness weakens an adversary that relies on knowing your strategy.

What the community started looking at

The first question was the **optimal search plan** for one searcher on a line (Beck 1964), posed by Bellman. Computer scientists recast it as competitive analysis and found that doubling is optimal with ratio 9, and the exact ratio for m rays, where the turning distances grow by the factor $m/(m-1)$ (Baeza-Yates et al. 1993); randomization lowers the ratio on the line to about 4.5911 (Kao et al. 1996). Recent work asks for **group search with faulty robots** (Czyzowicz et al. 2019c; Kupavskii and Welzl 2021; Czyzowicz et al. 2021a) and robots of different speeds (Bampas et al. 2019b).

Further reading

Papers on this problem that are not yet placed in a setting.

- Coleman and Morales-Ponce (2025). One robot on the line that moves either at speed 1, detecting the target with probability q when it passes it (they write p), or at a slower speed $v < 1$, always detecting it. Expected ratio $8/q + q/(2-q)$ using speed 1 only (Theorem 1, a version of which they attribute to Alpern and Gal); $3 + 2/v + 2\sqrt{2 + 2/v}$ when nothing is detected at speed 1 (Theorem 2), optimal in that case (Theorem 3); and a hybrid strategy that scouts ahead at speed 1, whose ratio (Theorem 4) is optimised numerically. A matching lower bound for speed 1 only is open. Read (arXiv v1, the full version).
- Czyzowicz et al. (2017d). One robot searches the line for a static target when its speed depends on the terrain: a faster speed in one direction (better than doubling, Theorem 1) or toward the origin (doubling is optimal, Theorem 2), faster travel over searched parts (Theorem 3), constant acceleration (Theorems 4–7) and inclined terrain. Read (arXiv v1).
- Angelopoulos and Panagiotou (2017). Weighted search on m rays: each ray holds a target with a weight, and the robot must find targets of total weight at least W. An adaptive strategy has ratio $1 + 2m(\frac{m}{m-1})^{m-1}$, the optimum for one target on m rays (Theorem 1), and better ratios when many targets must be found (Theorems 2 and 3). Read (arXiv v2, 2021).
- Bansal et al. (2023). The D-dimensional generalisation: a robot at the origin of $\mathbb{R}^D$ must reach an unknown hyperplane at distance at least 1. Every search path has ratio $\Omega(D^{3/2}/\log D)$ (Theorem 6), against the known $O(D^{3/2})$ of Antoniadis et al. (their d is D here). Read (arXiv v1).
- Czyzowicz et al. (2015a). k robots start at one end of a segment; each walks fast or searches slowly, at its own two speeds. An optimal schedule is computed in $O(k \log k)$ time when the length is known (Theorem 1), and an online algorithm is 2-competitive when it is not (Theorem 4), better when the walking speeds are equal (Theorem 5) (their n is k here). Read (arXiv v1, 2013).

Claude (Anthropic) was used in cataloguing the problems.

- Jawhar (2020). A master's thesis on linear search and evacuation by two robots sharing a bike that one robot at a time can ride; the basis of (Jawhar and Kranakis 2021). Abstract read.

Relations to other problems

- **Derived problem: Evacuation** (generalisation of). Evacuation is a form of search with an extra requirement: "optimality in evacuation is measured by the time it takes for the last robot to reach the exit, whereas in traditional search, optimality is measured by the time it takes the first robot to reach the exit" (Czyzowicz et al. 2017c). The introductory paper notes its similarity with the cow-path problem, and that evacuating uniform agents on a line "is as hard as the cow-path problem" (Czyzowicz et al. 2014).
- **Derived problem: Search for a moving target** (generalisation of). A static target is the case $u = 0$; then capture by every robot is evacuation on the line or on m rays, and the ratio of (Jawhar and Kranakis 2025b) reduces to $3 + 2\sqrt{2}$ (Czyzowicz et al. 2021d). Algorithms carry over from linear search only for $u = 0$; lower bounds for a static target hold for a target of speed u chosen by the adversary.

Assumptions specific to this problem

Domain

Bellman and Beck posed search for a target on the real line (Bellman 1963; Beck 1964) (as reported by Czyzowicz et al. 2021a). Baeza-Yates, Culberson and Rawlins treated the line as two rays and generalised to m rays from a common origin, with the optimal deterministic competitive ratio $1 + 2m^m/(m-1)^{m-1}$, attained by turning at distances $(m/(m-1))^i$ while visiting the rays cyclically (Theorem 2.3 of Baeza-Yates et al. 1993). Searching on m concurrent rays has, in the words of Hammar et al. quoted by Brandt et al., "paradigmatic status" in competitive analysis (Brandt et al. 2020b). The ray star is level 0 here because it contains the line ($m = 2$) as a special case.

Level	Value	Formal definition
0	m **rays**	$m \geq 2$ half-lines sharing their endpoint, the origin, where the robots start.
1	**line**	The real line, robots start at 0; the target is at an unknown x with $\lvert x\rvert \geq 1$.

Number of robots

The classical problem has one searcher (Beck 1964; Baeza-Yates et al. 1993). Chrobak et al. and Czyzowicz et al. proposed parallel versions with k robots on the line, where communication and faults make the team size matter: Chrobak et al. show that robots that communicate only when they meet need time $9d - o(d)$ to evacuate, whatever their number (Chrobak et al. 2015), and Czyzowicz et al. let up to f robots fail to detect the target (Czyzowicz et al. 2019c). With k fault-free robots, two of them already finish in time equal to the distance by going in opposite directions, so the multi-robot questions are interesting only with faults, with evacuation, or with unequal speeds (Kupavskii and Welzl 2021; Bampas et al. 2019b). Two robots are the smallest team in which speeds and communication interact, and Bampas et al. solve that case for every speed ratio (Bampas et al. 2019b).

Claude (Anthropic) was used in cataloguing the problems.

Level	Value	Formal definition
0	**any k**	$k \geq 1$ robots.
1	**one robot**	A single robot.
1	**two robots**	Two robots.

Goal

In search the time is the first arrival at the target, while in group evacuation it is the arrival of the last robot (Bampas et al. 2019b; Czyzowicz et al. 2021d). Chrobak et al. showed that evacuating any number of unit-speed robots that communicate only when they meet takes time $9d - o(d)$ on the line, as for one robot in the cow-path problem (Theorem 3, Corollary 1) (Chrobak et al. 2015). Bampas et al. compare search with rendezvous: in both the time is set by the last robot to arrive, but in search the meeting point is chosen by the adversary (Bampas et al. 2019b). With faults the two goals are closely linked: when $k = 2f + 1$ all agents may have to reach the target, and crash evacuation is then "a sub-problem of the more difficult byzantine search problem" (Czyzowicz et al. 2021e).

Level	Value	Formal definition
0	**evacuation**	The task ends when every robot has reached the target.
1	**search**	The task ends when the first robot reaches the target (and, with faults, the others can be sure of it).

Case tree

The root assumes nothing; going down, each edge adds the assumptions written in grey (+) or leaves an axis unassumed. Bold entries are catalogued settings with their status per measure (Competitive ratio) and page; italic entries are branch points, named after the axis on which their branches differ. Axis order: Domain → Number of robots → Goal → Communication → Faults → Speeds → Randomness.

No assumptions

- + line
  - *Number of robots*
    - + one robot, + search, + no faults, + equal speeds
      - **One robot on a line** STANDARD SETTING — ● p. 501
      - + randomized
        - **One randomized robot on a line** — ● p. 501
    - any k, + search, + wireless
      - *Faults*
        - + crash faults, + equal speeds
          - **k robots on a line, f crash faults** — ● p. 502
        - Byzantine, + equal speeds
          - **k robots on a line, Byzantine faults** — ● p. 503
    - + two robots, + no faults
      - **Two robots of different speeds, both must reach the target** — ● p. 503
- m rays, + one robot, + search, + no faults, + equal speeds
  - **One robot on m rays** — ● p. 504
  - + randomized
    - **One randomized robot on m rays** — ● p. 504

Claude (Anthropic) was used in cataloguing the problems.

What is known

The best upper bound against the best lower bound in each catalogued setting, in tree order. The settings follow in detail.

Setting	Competitive ratio	p.
One robot on a line STANDARD SETTING	• tight 9 vs 9	501
One randomized robot on a line	• tight ≈ 4.5911 vs ≈ 4.5911	501
k robots on a line, f crash faults	• tight $2\rho^\rho/(\rho-1)^{\rho-1}+1$ vs $2\rho^\rho/(\rho-1)^{\rho-1}+1$	502
k robots on a line, Byzantine faults	• polynomial gap depends on f/k vs ≈ 5.23	503
Two robots of different speeds, both must reach the target	• tight *optimal for every speed ratio* vs *matching*	503
One robot on m rays	• tight $1+\frac{2m^m}{(m-1)^{m-1}}$ vs $1+\frac{2m^m}{(m-1)^{m-1}}$	504
One randomized robot on m rays	• tight $\min_{\gamma>1}\left(1+\frac{2}{m}\cdot\frac{1+\gamma+\cdots+\gamma^{m-1}}{\ln\gamma}\right)$ vs $\min_{\gamma>1}\left(1+\frac{2}{m}\cdot\frac{1+\gamma+\cdots+\gamma^{m-1}}{\ln\gamma}\right)$	504

20.1.1 One robot on a line

Adds + one robot, + search, + no faults, + equal speeds STANDARD SETTING VERIFIED

Doubling the turning point each time gives competitive ratio 9, and nothing deterministic does better.

Assumed: domain: the line (two rays); number of robots: a single robot; goal: search: the first robot to reach the target ends the task; faults: no faults; speeds: all robots have the same unit speed. **Not assumed:** communication: face-to-face: robots exchange information only when they meet; randomness: deterministic.

Measure	Lower bound	Best upper bound	Status
Competitive ratio	9 (Baeza-Yates et al. 1993) Exactly 9.	9 (Baeza-Yates et al. 1993)	• tight

Results.

- Baeza-Yates et al. (1993) [lower bound]: competitive ratio 9. *Bottleneck:* unknown distance. *Holds when:* deterministic; target at an unknown integer distance d on an unknown side; ratio up to lower-order terms in d. Theorem 2.1 of (Baeza-Yates et al. 1993). Also due to (Beck 1964) in the original formulation.
- Baeza-Yates et al. (1993) [current best]: competitive ratio 9. Introductory (in the competitive formulation) and current best: linear spiral search (turning points 2^i) walks at most $9d$ steps; there are algorithms with $9d-\Theta((\log d)^i)$ steps for any i.

Notes. Checked against the full text of (Baeza-Yates et al. 1993) (scan, Section 2.1). The bound is on the constant multiple of the distance; lower-order terms are not optimised.

20.1.2 One randomized robot on a line

Adds + randomized **to** One robot on a line (p. 501) VERIFIED

Randomization lowers the optimal expected ratio on the line to about 4.5911.

SEARCH

Claude (Anthropic) was used in cataloguing the problems.

Assumed: domain: the line (two rays); number of robots: a single robot; goal: search: the first robot to reach the target ends the task; faults: no faults; speeds: all robots have the same unit speed; randomness: randomized. **Not assumed:** communication: face-to-face: robots exchange information only when they meet.

Measure	Lower bound	Best upper bound	Status
Competitive ratio	≈ 4.5911 (Kao et al. 1996) Optimal among randomized algorithms.	≈ 4.5911 (Kao et al. 1996)	• tight

Results.

- Kao et al. (1996) [lower bound]: competitive ratio ≈ 4.5911. *Bottleneck:* unknown distance. *Holds when:* expected competitive ratio; any randomized algorithm; goal position fixed in advance (oblivious adversary); two paths (the line). Theorem 4.1 of (Kao et al. 1996).
- Kao et al. (1996) [current best]: competitive ratio ≈ 4.5911. Introductory and current best (SmartCow): randomness only in a random permutation of the paths and a random initial distance; ratio ≈ 4.5911 on the line, and about $3.088\,m + o(m)$ on m rays.

Notes. Checked against the full text of (Kao et al. 1996). The ratio is the fixed point of an equation and is computed numerically (Theorem 5.1). Kao, Reif and Tate say Gal had obtained many of these results earlier in a game-theoretic setting, and that optimality for every number $m \geq 2$ of rays was proved afterwards by Kao et al. (SODA 1994; not in the bibliography); see the setting on m rays, where (Ma and Yin 2026) complete that proof.

20.1.3 k robots on a line, f crash faults

Adds + crash faults, + equal speeds VERIFIED

With k wireless robots of which f may fail to report the target, the optimal ratio is known exactly.

Assumed: domain: the line (two rays); goal: search: the first robot to reach the target ends the task; communication: wireless: instant communication at any distance; faults: up to f crash-faulty robots (fail to report the target); speeds: all robots have the same unit speed. **Not assumed:** number of robots: any number k of robots; randomness: deterministic.

Measure	Lower bound	Best upper bound	Status
Competitive ratio	$2\rho^\rho/(\rho-1)^{\rho-1}+1$ (Kupavskii and Welzl 2021) Matching bounds.	$2\rho^\rho/(\rho-1)^{\rho-1}+1$ (Czyzowicz et al. 2019c)	• tight

Results.

- Kupavskii and Welzl (2021) [lower bound]: competitive ratio $2\rho^\rho/(\rho-1)^{\rho-1}+1$. *Bottleneck:* faulty reports. *Holds when:* $\rho = 2(f+1)/k$, for $k < 2(f+1)$; ratio 1 if $k \geq 2f+2$.
- Czyzowicz et al. (2019c) [current best]: competitive ratio $2\rho^\rho/(\rho-1)^{\rho-1}+1$. Introductory and current best (proportional schedules, Theorem 1); the algorithm needs no communication; optimality proved by (Kupavskii and Welzl 2021).

Notes. On m rays the same formula holds with $\rho = m(f+1)/k$ (Kupavskii and Welzl 2021). Czyzowicz et al. write the ratio as $(\frac{4f+4}{k})^{(2f+2)/k}(\frac{4f+4}{k}-2)^{1-(2f+2)/k}+1$, the same expression; their own lower bound, which holds even with communication, is about 3.76 for $(k,f)=(3,1)$ (Theorem 2) and $3+2\ln k/k - 2\ln\ln k/k$ asymptotically (Corollary 2), so for $k = 2f+1$ the ratio tends to 3 (Czyzowicz et al. 2019c). Checked against the full text of (Czyzowicz et al. 2019c) (PODC 2016 version).

Claude (Anthropic) was used in cataloguing the problems.

20.1.4 k robots on a line, Byzantine faults

Adds Byzantine, + equal speeds VERIFIED

Byzantine robots may lie about the target; the optimal ratio is known only for some densities f/k.

Assumed: domain: the line (two rays); goal: search: the first robot to reach the target ends the task; communication: wireless: instant communication at any distance; speeds: all robots have the same unit speed. **Not assumed:** number of robots: any number k of robots; faults: up to f Byzantine robots (may lie or stay silent); randomness: deterministic.

Measure	Lower bound	Best upper bound	Status
Competitive ratio	≈ 5.23 (Kupavskii and Welzl 2021)	depends on f/k (Czyzowicz et al. 2021a)	• polynomial gap
	For $k = 3$, $f = 1$ the upper bound is not matched.		

Results.

- Kupavskii and Welzl (2021) [lower bound]: competitive ratio ≈ 5.23. *Bottleneck:* indistinguishable Byzantine agents. *Holds when:* $k = 3$, $f = 1$; improves 3.93 of (Czyzowicz et al. 2021a).
- Czyzowicz et al. (2021a) [current best]: competitive ratio depends on f/k. Introductory and current best; optimal for some densities of faulty robots.

Open questions.

Q1. Determine the optimal competitive ratio $B(k, f)$ for Byzantine search on the line; already $B(3, 1)$ is open.

Notes. For three robots with one Byzantine robot the upper bound is about 7.437 (Czyzowicz et al. 2021e), improving 8.653 (Theorem 3 of Sun et al. 2020; its abstract says 8.53).

20.1.5 Two robots of different speeds, both must reach the target

Adds + two robots, + no faults VERIFIED

For two robots with different maximum speeds that must both reach the target, the optimal strategy is known for every speed ratio when they communicate by meeting.

Assumed: domain: the line (two rays); number of robots: two robots; faults: no faults. **Not assumed:** goal: evacuation: every robot must reach the target; communication: face-to-face: robots exchange information only when they meet; speeds: robots may have different maximum speeds; randomness: deterministic.

Measure	Lower bound	Best upper bound	Status
Competitive ratio	*matching* (Bampas et al. 2019b)	*optimal for every speed ratio* (Bampas et al. 2019b)	• tight
	Optimal (face-to-face); the wireless case is optimal within the stated speed range.		

Results.

- Bampas et al. (2019b) [lower bound]: competitive ratio *matching*. *Bottleneck:* unknown distance. *Holds when:* face-to-face communication; all speed ratios.
- Chrobak et al. (2015) [introductory]: competitive ratio 9 ($v \geq 1/3$). Face-to-face, speeds 1 and $v \geq 1/3$: the slower robot follows its own doubling path at speed $1/3$ and is fetched by the faster one (Theorem 4).
- Bampas et al. (2019b) [current best]: competitive ratio *optimal for every speed ratio*. Wireless communication helps only while the faster robot is at most $\sqrt{17} + 4 \approx 8.12$ times faster.

Claude (Anthropic) was used in cataloguing the problems.

Notes. Introduced by Chrobak, Gąsieniec, Gorry and Martin, who also show that two unit-speed robots and further robots of speed at least $1/5$ still finish in time $9d$ (Section 3.2) (Chrobak et al. 2015). Checked against the full texts of (Chrobak et al. 2015) and (Bampas et al. 2019b).

20.1.6 One robot on m rays

Adds m rays, + one robot, + search, + no faults, + equal speeds VERIFIED

On m rays the optimal deterministic ratio is $1 + 2m^m/(m-1)^{m-1}$.

Assumed: number of robots: a single robot; goal: search: the first robot to reach the target ends the task; faults: no faults; speeds: all robots have the same unit speed. **Not assumed:** domain: m rays from a common origin; communication: face-to-face: robots exchange information only when they meet; randomness: deterministic.

Measure	Lower bound	Best upper bound	Status
Competitive ratio	$1 + \frac{2m^m}{(m-1)^{m-1}}$ (Baeza-Yates et al. 1993) Exact.	$1 + \frac{2m^m}{(m-1)^{m-1}}$ (Baeza-Yates et al. 1993)	• tight

Results.

- Baeza-Yates et al. (1993) [lower bound]: competitive ratio $1 + \frac{2m^m}{(m-1)^{m-1}}$. *Bottleneck:* unknown distance. *Holds when:* deterministic; target at an unknown integer distance on one of m rays; ratio up to lower-order terms; also the special case $k = 1$, $f = 0$ of (Kupavskii and Welzl 2021). Theorem 2.3 of (Baeza-Yates et al. 1993).
- Baeza-Yates et al. (1993) [current best]: competitive ratio $1 + \frac{2m^m}{(m-1)^{m-1}}$. Introductory and current best: generalized linear spiral search, turning points $(m/(m-1))^i$ and rays visited cyclically.

Notes. Checked against the full text of (Baeza-Yates et al. 1993) (scan, Section 2.3).

20.1.7 One randomized robot on m rays

Adds + randomized **to** One robot on m rays (p. 504) DRAFT

A randomized robot that visits the m rays in a random cyclic order, with turning points growing by a fixed factor from a random start, has the optimal expected ratio, about $3.088\,m$ for large m.

Assumed: number of robots: a single robot; goal: search: the first robot to reach the target ends the task; faults: no faults; speeds: all robots have the same unit speed; randomness: randomized. **Not assumed:** domain: m rays from a common origin; communication: face-to-face: robots exchange information only when they meet.

Measure	Lower bound	Best upper bound	Status
Competitive ratio	$\min_{\gamma>1}\left(1 + \frac{2}{m}\cdot\frac{1+\gamma+\cdots+\gamma^{m-1}}{\ln\gamma}\right)$ (Ma and Yin 2026) [preprint] Exact, if the bound of Kao, Ma, Sipser and Yin for cyclic strategies (not read) is accepted; the step from cyclic to all strategies is in a preprint (Ma and Yin 2026).	$\min_{\gamma>1}\left(1 + \frac{2}{m}\cdot\frac{1+\gamma+\cdots+\gamma^{m-1}}{\ln\gamma}\right)$ (Kao et al. 1996)	• tight

Results.

- Ma and Yin (2026) [preprint] [lower bound]: competitive ratio $\min_{\gamma>1}\left(1 + \frac{2}{m}\cdot\frac{1+\gamma+\cdots+\gamma^{m-1}}{\ln\gamma}\right)$. *Bottleneck:* unknown distance. *Holds when:* expected competitive ratio against an oblivious adversary; Yao's principle: the target lies on a ray chosen uniformly at random, at a distance $x \geq 1$ with density $\varepsilon x^{-(1+\varepsilon)}$, and $\varepsilon \to 0$; for strategies that visit the rays in a fixed cyclic order the bound is due to Kao, Ma, Sipser and Yin (Journal of Algorithms 29, 1998; not in the bibliography, not read), as reported by (Kao et al. 1996) and (Ma and Yin 2026); (Ma and Yin 2026) show that some optimal deterministic strategy against this distribution is cyclic, so the bound holds for

Claude (Anthropic) was used in cataloguing the problems.

every randomized strategy. Theorem of Section 6 of (Ma and Yin 2026) (with Propositions 1–3 and Appendix A), a preprint. It writes out the proof of Fact 4 of Kao, Ma, Sipser and Yin, which was left to an unpublished manuscript of 1994; the authors say they rebuilt it from hand-written notes. γ is the factor by which the turning distance grows from one trip to the next.

- Kao et al. (1996) [current best]: competitive ratio $\min_{\gamma>1}\left(1+\frac{2}{m}\cdot\frac{1+\gamma+\cdots+\gamma^{m-1}}{\ln\gamma}\right)$. Introductory and current best (SmartCow): a random permutation of the rays, a random initial distance, then the rays in cyclic order with turning distances growing by the factor γ; ratio $1+\frac{2}{m}\cdot\frac{1+\gamma+\cdots+\gamma^{m-1}}{\ln\gamma}$ for every $\gamma>1$ (Theorem 3.1), minimised at the unique root of an equation in γ (Theorem 5.1), and about $3.088\,m+o(m)$ (Theorem 6.1). For $m=2$ it is about 4.5911.

Notes. Kao, Reif and Tate proved SmartCow optimal for $m=2$ (Theorem 4.1; see the setting on the line) and report that Kao, Ma, Sipser and Yin later proved it optimal for every $m\geq 2$ (Kao et al. 1996). According to Ma and Yin, that proof assumed that it suffices to consider strategies that visit the rays in a fixed cyclic order (Fact 4 of the 1998 paper) and left the argument to an unpublished manuscript (Ma and Yin 2026). Their note supplies it for the distribution used in the lower bound: an optimal strategy exists (Appendix A), always extends a ray that has been explored least far (Proposition 1), extends it by more than a fixed factor (Proposition 2) and uses nondecreasing turning distances (Proposition 3), and can therefore be made cyclic. They leave open whether the same holds for the simpler truncated distribution of Schuierer (2003; not in the bibliography), whose proof of the lower bound assumes cyclic strategies; numerics for $m=3$ suggest it does (Ma and Yin 2026). Both papers write w for m. Checked against the full texts of (Kao et al. 1996) (Theorems 3.1, 5.1, 6.1) and (Ma and Yin 2026); the 1998 lower bound for cyclic strategies was not read.

20.2 Evacuation

Robots start at the centre of a disk (or a polygon) and must all leave through an exit at an unknown point of the boundary, as fast as possible.

First posed in 2014 (Czyzowicz et al. 2014). Czyzowicz et al. posed evacuation of k robots from a disk through an unknown exit on its boundary at DISC 2014.

Problem statement and measures

k unit-speed robots start together at the centre of a unit disk (or at the centroid, or at a given point of the boundary, of a unit equilateral triangle or square). An exit lies at an unknown point of the boundary; a robot recognises it only on reaching it. **Time** is the worst case, over exit positions, of the time until the last robot reaches the exit (in priority evacuation, where one robot is the **queen**: until the queen does; in weighted evacuation of two robots, a weighted average of their arrival times; in search, until a non-faulty robot has reached the exit and every non-faulty robot knows where it is).

Measures: **Evacuation time**, worst-case time until the last robot (or the queen) reaches the exit, or the weighted average or search time where the goal says so; unit disk, unit speed.

A first look

Two people stand in the middle of a dark round room of radius 1 and know there is one door somewhere along its edge, but not where; they can talk by walkie-talkie. In evacuation the people are **robots**, points that move at unit speed; the room is a unit **disk**, the points at distance at most 1 from its centre; its edge

Claude (Anthropic) was used in cataloguing the problems.

SEARCH



(a) Start: both robots at the centre O

(b) Final: robot 2 arrives by the chord at 1 + 2π/3 + √3

Figure 20.2: Evacuation from a disk: two robots start at the centre of a unit disk and must both reach an exit at an unknown point of the boundary. The animation runs the wireless algorithm of (Czyzowicz et al. 2014) at unit speed: both walk to the boundary together, then search it in opposite directions; the one that finds the exit tells the other by wireless communication, and the other walks straight across the disk along the chord. With the exit placed at arc distance $2\pi/3$ (the worst case) the time is $1 + 2\pi/3 + \sqrt{3} \approx 4.83$.

is the **boundary**, the circle of radius 1; and the door is the **exit**, a point of the boundary that a robot recognises only when it stands on it. k robots start at the centre of the disk. The goal is to minimise, in the worst case over exit positions, the time until the **last** robot reaches the exit. In the basic version there are two robots with **wireless** communication: a robot that finds the exit tells the other instantly, wherever it is.

SEARCH

1. Algorithm: Go to the boundary together, then split. (Czyzowicz et al. 2014) (Two wireless robots, disk, p. 517)

1. Both robots walk together from the centre to the same point A of the boundary (time 1).
2. One walks clockwise along the boundary, the other counter-clockwise, at full speed.
3. The robot that finds the exit tells the other by wireless communication.
4. The other walks straight across the disk to the exit, along a **chord** (the straight segment between two points of the boundary).

Why it works and what it costs. If the exit is found after each robot has walked an **arc** (a piece of the boundary) of length x (so $x \le \pi$), the robots are $2x$ apart along the boundary, and the chord between them has length $2\sin x$. The total time is $1 + x + 2\sin x$. Its derivative $1 + 2\cos x$ is zero at $x = 2\pi/3$, so the worst case is $1 + 2\pi/3 + \sqrt{3} \approx 4.826$ (Czyzowicz et al. 2014). This is exactly optimal (next idea).

Why it matters. Searching the boundary in opposite directions and cutting across on a chord is the template for almost every later disk evacuation algorithm.

2. Lower bound: Two unexplored points far apart. (Czyzowicz et al. 2014) (Two wireless robots, disk, p. 517) Look at time $1 + 2\pi/3$. Each robot needed time 1 just to reach the boundary, so together they have walked at most $4\pi/3$ of the boundary, and at least $2\pi/3$ of the boundary (minus a tiny ε) is unexplored. (Czyzowicz et al. 2014) show that then there are two unexplored points at arc distance about $2\pi/3$ (measured along the boundary), so the chord between them has length $\sqrt{3}$. The **adversary**, an opponent who watches the robots and chooses the exit position, waits until some robot first reaches one of these two points and places the exit at the other one.

Why it holds. That robot is then at least $\sqrt{3}$ away from the exit and must still get there, and this happens no earlier than time $1 + 2\pi/3$. So every algorithm needs at least $1 + 2\pi/3 + \sqrt{3}$ in the worst case, which matches the algorithm above (Czyzowicz et al. 2014).

Claude (Anthropic) was used in cataloguing the problems.

Why it matters. Counting how much of the boundary can be explored by a given time, then hiding the exit at the far end of an unexplored chord, is the basic lower-bound tool for evacuation.

3. Algorithm: Without wireless communication: catch the other robot. (Czyzowicz et al. 2014) (Two face-to-face robots, disk, p. 519) If robots exchange information only when they meet (**face-to-face** communication):

1. Go to the boundary together and split in opposite directions, as before.
2. The robot that finds the exit cannot call; it walks along a chord to the point where the other robot will be, meets it,
3. and both walk back along a chord to the exit.

Why it works and what it costs. The robot that finds the exit must cover the distance to the other robot and back, so the cost is larger: about 5.74 for this simple version (Czyzowicz et al. 2014). Cleverer paths that leave the boundary and come back improved it to about 5.6234 (Disser and Schmitt 2019), while the best lower bound is $3 + \pi/6 + \sqrt{3} \approx 5.255$ (Czyzowicz et al. 2020d). This gap has been open since 2019.

Why it matters. It shows how much communication is worth, and it is one of the simplest-to-state geometric questions still open in the area.

What the community started looking at

Evacuation was introduced to capture the tension between **spreading the robots** to find the exit fast and **keeping them close** so that the others can reach it quickly (Czyzowicz et al. 2014). The first results already separated wireless from face-to-face communication. The community then tightened the two-robot face-to-face constants (Czyzowicz et al. 2020d; Brandt et al. 2017; Disser and Schmitt 2019) and added faults (Czyzowicz et al. 2017c), priority (one robot must leave) (Czyzowicz et al. 2020b), asymmetric communication (Georgiou et al. 2022), other domains (Chuangpishit et al. 2020b) and several exits (Pattanayak et al. 2018). Beyond disks, triangles and squares, the question was asked on the line and on rays (Chrobak et al. 2015; Brandt et al. 2020b), in regular polygons with priority (Georgiou et al. 2025b), on known trees as a planning problem that is NP-hard to solve exactly (Devillez et al. 2021), and in the plane for an unknown shoreline (Acharjee et al. 2019) or an unknown convex region to escape (WEI et al. 2016).

Further reading

Papers on this problem that are not yet placed in a setting.

- Czyzowicz et al. (2019d). Survey chapter (2019) on group search and evacuation: line search with crash and Byzantine faults, linear terrains, disk, triangle and square evacuation, faulty robots, several exits on a circle, and search-and-fetch. Read; it reports results of the papers placed in the settings.
- Kranakis (2024). Invited survey (SSS 2024) on how knowledge of the communication model, of faults and of the target's distance and speed changes the competitive ratio of linear group search and evacuation, with open problems. Read; no new results.
- Czyzowicz et al. (2021f). Time–energy trade-offs for evacuating two wireless robots from the line.
- Jawhar and Kranakis (2021). Two robots and one bike on the line: walking speed 1, riding speed $v > 1$, one rider at a time; three algorithms in the wireless model and three face-to-face (Theorems 1–6), and a lower bound for both models (Theorem 8). Read.

Claude (Anthropic) was used in cataloguing the problems.

SEARCH

- Pattanayak et al. (2020a). One-page extended abstract (ICDCN 2020) summarising two-robot disk evacuation with two exits at known arc distance, labelled or unlabelled, wireless or face-to-face, and carrying a crashed robot to the exit at reduced speed; it states no bounds and defers to the full papers. Read.
- Chuangpishit et al. (2020a). Journal version of (Chuangpishit et al. 2018b): two face-to-face robots in the disk; families of algorithms minimising the average evacuation time over the exit position subject to a bound on the worst-case time, for every feasible bound (Theorems 6, 7); conjectures that $1 + \pi$ is the optimal average. Read.
- Liu et al. (2022). Evacuation of k wireless robots from the centre of a unit square whose exit is at one of its corners, with crash faults that stop a robot, which a non-faulty robot must then carry to the exit at reduced speed; competitive ratios for $(k, f) = (2, 1)$ (Theorem 1), $(3, 1)$ and $(3, 2)$ as functions of the slowdown. Read; neither the exit at a corner nor carrying is in the tree.
- Fekete et al. (2010). Evacuating rectilinear polygons; background from before the disk line of work.
- Borowiecki et al. (2025). Discrete evacuation of agents from distinct nodes of a known graph to known exits, at most one agent per node: an optimal offline strategy in polynomial time (Theorem 3.1); on trees, with communication at distance two, no $(2 - \Theta(1/k))$-competitive online algorithm (Theorem 4.1) and an $O(1)$-competitive one (Theorem 4.2); NP-complete with agents restricted to subgraphs (Theorems 5.1, 5.2). A different sense of evacuation. Read.
- Czyzowicz et al. (2019e). Search for a point in the plane by k robots of speeds v_i with unit visibility, ending when the last reliable robot reaches the target: $\pi d^2/(2V) + o(d^2)$ with V the sum of the speeds, asymptotically optimal (Theorem 1); asymptotically optimal bounds with f crash faults (Theorem 2, Lemma 6); with f Byzantine faults asymptotically optimal when wireless, and $(f + 1)\pi d^2/(2k) + o(d^2)$ face-to-face for equal speeds and $k \geq 2f + 1$ (Theorem 3). Read.
- Dobrev et al. (2020b). Search for a line (the shoreline) at known distance d in the plane (they write δ): every algorithm needs time at least $2d$ with three robots (Theorem 1) and $(1 + \sqrt{3} + \pi/6)d \approx 3.2556d$ with two (Theorem 3), matching known upper bounds; the bound for three robots also holds for unknown d. Search, not evacuation. Read.
- Xu and Qin (2013). Groups of evacuees at one point of an unknown convex region of the plane, sharing information at all times, leave when all have reached the boundary: every strategy has ratio at least 3 (Theorem 1); $k \geq 3$ groups in equally spaced directions achieve $3/\cos(\pi/k)$ (Theorem 2); strategies on a grid network, ratio 3 with four groups (Theorem 6). Read.
- Wei et al. (2014). One group, without communication with others, escaping an unknown convex region of the plane: a 19.64-competitive strategy along growing semicircles (Theorem 1) and a 21-competitive strategy on a grid network (Theorem 2). Read.
- Li and Xu (2017). Escaping an unknown convex region once the boundary is known to all after the first evacuee reaches it: a 5-competitive strategy from one start (Theorem 2); from several starts, every strategy has ratio at least 3 (Theorem 1), and the strategy MES achieves 4 with two starts (Theorem 3) and $2 + \sqrt{5}$ in general (Theorem 4), improving $4\sqrt{2}$ of (Liu et al. 2015). Read.
- Liu et al. (2015). Groups that start at several points of an unknown convex region of the plane and share information: a $4\sqrt{2}$-competitive strategy for $n \geq 3$ groups (Theorem 1). Read.
- Jiang et al. (2015). Groups escaping a convex region. Title only.
- Wei et al. (2016). Escaping an unknown convex region: a 19.64-competitive strategy for one group (Theorem 1), a 14.37-competitive strategy for two groups (Theorem 2), and 21- and 17-competitive strategies for one and two groups on a grid network (Theorems 3, 4). Read.
- Conley and Georgiou (2025). Related task: several agents inspect the boundary of a disk.
- Georgiou (2026). Related task: optimal average-case disk inspection.
- Coleman et al. (2023). Related task: two robots deliver a target to safety. Not read.
- Georgiou et al. (2025c). Plane search with agents of several speeds; a preprint, not read.
- Coleman et al. (2026b). Search and rescue in the plane; a preprint, not read.

Claude (Anthropic) was used in cataloguing the problems.

- Georgiou et al. (2017). Fence-jumping: one robot searches the unit circle around its start for a treasure, knowing that an arc of length β at an unknown position (the fence) holds no treasure; it may cut across chords. The optimal trajectory with one jump (Theorem 1), the jumps of an optimal k-jump trajectory (Theorem 2) and a halving heuristic (Theorem 3). Read (arXiv v1, 2016).

- Coleman et al. (2021). Robots with different speeds, anywhere in the plane, relay a message face-to-face from a source to a destination. Offline: optimal for two robots via Apollonius circles and a $\sqrt{2}$-approximation for any number; online: an algorithm with competitive ratio below 2 for any number of robots, tight for that algorithm, and lower bounds of 1.0391 and 1.0405 for every online algorithm. Read (arXiv v1).

- Nakamura et al. (2021). A different model, a metamorphic robotic system (connected oblivious modules on the square grid) that must leave a rectangular field through an exit: two modules with a common compass are necessary and sufficient (Theorem 1); without one, four modules from restricted initial shapes and seven from any (Theorems 2 and 3), and two when the system starts touching a wall (Theorem 4); also when the system can stop after leaving, mazes and convex fields. Read (arXiv v3).

Relations to other problems

- **Generalisation of Linear search (cow path).** Evacuation is a form of search with an extra requirement: "optimality in evacuation is measured by the time it takes for the last robot to reach the exit, whereas in traditional search, optimality is measured by the time it takes the first robot to reach the exit" (Czyzowicz et al. 2017c). The introductory paper notes its similarity with the cow-path problem, and that evacuating uniform agents on a line "is as hard as the cow-path problem" (Czyzowicz et al. 2014).

- **Derived problem: Chauffeuring a crashed robot** (variant of). Evacuation where one robot has crashed inside the disk: the active robot must first find it, then carry it, more slowly, to an exit.

Assumptions specific to this problem

Domain

Czyzowicz et al. posed evacuation in the unit disk with the exit on its boundary (Czyzowicz et al. 2014), which remains the standard domain. Evacuation on the line had been raised just before by Chrobak et al.: robots that communicate only when they meet need time $9d - o(d)$ to evacuate whatever their number, as much as a single robot in the cow-path problem (Chrobak et al. 2015); Czyzowicz et al. cite this as showing that evacuating several uniform agents "is as hard as the cow-path problem" (Czyzowicz et al. 2014). Polygons followed: equilateral triangles and squares, first with wireless robots (Czyzowicz et al. 2015c) and then face-to-face (Chuangpishit et al. 2020b). Czyzowicz et al. let the robots start either at the centroid or at a point of the boundary, and the start changes the answer: from the boundary of a triangle three or more robots do no better than two, while from the centroid three robots beat two (Czyzowicz et al. 2015c). A start on the boundary is therefore a separate domain value. Brandt et al. studied m concurrent rays, citing Hammar et al. that "a problem with paradigmatic status in this framework is searching on m concurrent rays" (Brandt et al. 2020b). Evacuation from a general convex region appears in a separate line of online-strategy papers (Xu and Qin 2013; Wei et al. 2014).

Claude (Anthropic) was used in cataloguing the problems.

SEARCH

Level	Value	Formal definition
0	**any domain**	A bounded convex domain; the robots start together at a given point of it, in its interior or on its boundary.
1	**disk**	The unit disk; robots start at its centre; the exit is a point of the boundary circle.
1	**triangle**	The equilateral triangle of unit side; robots start at its centroid; the exit is a point of the boundary.
1	**square**	The unit square; robots start at its centroid; the exit is a point of the boundary.
1	**triangle, boundary start**	A triangle (the equilateral triangle of unit side unless the setting names another family); the robots start together at a point of its boundary, either given (at distance y from the nearest midpoint of a side) or chosen by the algorithm, as the setting says; the exit is a point of the boundary.
1	**square, boundary start**	The unit square; the robots start together at a given point of its boundary, at distance $x \leq 1/2$ from the nearest corner; the exit is a point of the boundary.
1	**line**	The robots start at the origin of the real line; the exit is at an unknown point at distance at least 1; the measure is the competitive ratio, the evacuation time divided by that distance.
1	**rays**	$m \geq 2$ half-lines share an endpoint, the junction; the exit lies on one ray at distance at least 1 from the start; the measure is the competitive ratio.

SEARCH

Number of robots

Czyzowicz et al. studied the exact optimal evacuation time for $k = 2$ and $k = 3$ robots in the disk, and the asymptotic behaviour as k grows: $3 + 2\pi/k$ face-to-face and $3 + \pi/k$ wireless, up to lower-order terms (Czyzowicz et al. 2014). Most later disk papers fix $k = 2$, and even there the face-to-face case is not closed after several improvements (Czyzowicz et al. 2020d; Brandt et al. 2017; Disser and Schmitt 2019). Specific small team sizes also appear in priority evacuation with one or two servants (Czyzowicz et al. 2020a) and with polygons, where triangles and squares were solved for $k = 2, 3, 4$ (Chuangpishit et al. 2020b).

Level	Value	Formal definition
0	**any k**	$k \geq 1$ robots.
1	**two robots**	Two robots.
1	**three robots**	Three robots.

Goal

Search ends at the first arrival and evacuation at the last (Czyzowicz et al. 2014). Czyzowicz et al. introduced *priority evacuation*, in which only a distinguished robot, the queen, must reach the exit while servants help. They motivate it by a robot that alone "has the capability to address an urgent need at the target, for example, performing an emergency procedure, or closing a breach in the perimeter" (Czyzowicz et al. 2020b). Georgiou, Karakostas and Kranakis introduced *search-and-fetch* (treasure evacuation), where a robot must carry a treasure to the exit. It is motivated by search-and-rescue, where drones must "locate a victim (the treasure) and bring her to safety (the exit)" (Georgiou et al. 2019). They note that "this is the first time an explicit ordering on the tasks to be performed is imposed on two moving robots" (Georgiou et al. 2019). Georgiou and Lucier weighted the arrival times of two robots on the line, a light one and a heavy one, and obtained priority evacuation on the line as the limit of

Claude (Anthropic) was used in cataloguing the problems.

large weight (Georgiou and Lucier 2023). With faulty robots, *search* (the exit is found by a non-faulty robot and known to all non-faulty robots) is the first stage of evacuation: Georgiou et al. solved it on the circle with crash faults and with one Byzantine robot (Georgiou et al. 2023b), and Behrouz et al. note that the evacuation time "is inherently greater than or equal to the time required to find the exit" (Behrouz et al. 2025). Linear search uses the same two goals.

Level	Value	Formal definition
0	**all robots**	The task ends when the last robot reaches the exit.
1	**priority (queen)**	One designated robot (the queen) must reach the exit; the others (servants) only help; the task ends when the queen exits.
1	**search-and-fetch**	A treasure lies at an unknown point of the boundary, and the exit at a known arc distance α from it; the task ends when some robot has carried the treasure to the exit.
1	**weighted**	Two robots, a light one and a heavy one of known weight $w \geq 1$, both must reach the exit and stay there; with arrival times t_1 (light) and t_2 (heavy), the cost is $(t_1 + w\,t_2)/(1 + w)$.
1	**search**	The task ends as soon as a non-faulty robot has reached the exit and every non-faulty robot knows its position for certain; the robots need not go to the exit.

Exits

Evacuation was posed with a single unknown exit (Czyzowicz et al. 2014). Czyzowicz et al. considered two robots on a circle with k exits whose distribution is known, where the robots do not know their own position (Czyzowicz et al. 2018a). Pattanayak, Ramesh, Mandal and Schmid studied two robots starting at the centre of the disk with two exits at unknown points whose arc distance α is known. They worked in both communication models and introduced labelled exits (Pattanayak et al. 2018). More exits help the robots, but only if the algorithm can use what it knows about the exits' relative position.

Level	Value	Formal definition
0	**one exit**	One exit, at an unknown point of the boundary.
1	**two exits**	Two exits at unknown points of the boundary; the task ends when every robot has reached one of them.

Case tree

The root assumes nothing; going down, each edge adds the assumptions written in grey (+) or leaves an axis unassumed. Bold entries are catalogued settings with their status per measure (Evacuation time) and page; italic entries are branch points, named after the axis on which their branches differ. Axis order: Domain → Number of robots → Goal → Exits → Communication → Faults → Speeds → Randomness.

No assumptions
+ disk
Number of robots
+ two robots
Goal
all robots, one exit, + wireless, + no faults
Disk, two wireless robots of different speeds ● p. 516
+ equal speeds
Two wireless robots, disk STANDARD SETTING ● p. 517

Claude (Anthropic) was used in cataloguing the problems.

- all robots, + two exits
  - *Communication*
    - face-to-face, + no faults, + equal speeds
      - **Two robots, two exits, face-to-face** • p. 517
    - + wireless, + no faults, + equal speeds
      - **Two robots, two exits, wireless** • p. 518
- + search-and-fetch
  - *Communication*
    - + wireless, + no faults
      - **Disk, search-and-fetch, two wireless robots** • p. 518
    - face-to-face, + no faults
      - **Disk, search-and-fetch, two face-to-face robots** • p. 519
- all robots, one exit, face-to-face, + no faults, + equal speeds
  - **Two face-to-face robots, disk** • p. 519
- all robots, one exit, + sender/receiver, + no faults, + equal speeds
  - **Two robots with asymmetric communication, disk** • p. 520
- + priority (queen), + wireless, + no faults
  - **Disk, priority evacuation, queen and one servant** • p. 520
- + weighted, + wireless, + no faults, + equal speeds
  - **Disk, weighted evacuation of two wireless robots** • p. 521

+ three robots
- *Goal*
  - all robots, + wireless
    - *Faults*
      - + no faults, + equal speeds
        - **Three wireless robots, disk** • p. 522
      - + crash faults, + equal speeds
        - **Three wireless robots, one crash fault** • p. 522
      - Byzantine, + equal speeds
        - **Three wireless robots, one Byzantine** • p. 523
  - all robots, face-to-face, + no faults, + equal speeds
    - **Three face-to-face robots, disk** • p. 523
  - + priority (queen), + wireless, + no faults
    - **Disk, priority evacuation, queen and two servants** • p. 524

any k, all robots, + wireless
- *Faults*
  - + no faults, + equal speeds
    - **Many wireless robots, disk** • p. 524
  - Byzantine, + equal speeds
    - **Disk, k wireless robots, f Byzantine** • p. 525

any k, + search, + wireless
- *Faults*
  - + crash faults, + equal speeds
    - **Disk search, k wireless robots, f crash faults** • p. 525

Claude (Anthropic) was used in cataloguing the problems.

Byzantine, + equal speeds
Disk search, k wireless robots, one Byzantine ● p. 526

any k, all robots, face-to-face, + no faults, + equal speeds
Many face-to-face robots, disk ● p. 527

any k, + priority (queen), + wireless, + no faults, + equal speeds
Priority evacuation (queen and servants), disk ● p. 527

any k, all robots, face-to-face, Byzantine, + equal speeds
Disk, k face-to-face robots, f Byzantine ● p. 528

any k, all robots, face-to-face, + crash faults, + equal speeds
Disk, k face-to-face robots, f crash faults ● p. 528

+ line
Number of robots

+ two robots
Goal

+ weighted
Communication

+ wireless, + no faults, + equal speeds
Line, weighted evacuation of two wireless robots ● p. 529

face-to-face, + no faults, + equal speeds
Line, weighted evacuation of two face-to-face robots ● p. 530

+ priority (queen)
Communication

+ wireless, + no faults, + equal speeds
Line, priority evacuation, queen and one wireless servant ● p. 530

face-to-face, + no faults, + equal speeds
Line, priority evacuation, queen and one face-to-face servant ● p. 531

all robots, face-to-face, + no faults
Line, two face-to-face robots of different speeds ● p. 531

all robots, + wireless, + no faults
Line, two wireless robots of different speeds ● p. 532

all robots, + sender/receiver, + no faults, + equal speeds
Line, a sender and a receiver ● p. 532

any k, + wireless
Faults

+ no faults, + equal speeds
Line, any number of wireless robots ● p. 533

+ crash faults, + equal speeds
Line, $2f + 1$ wireless robots, f crash faults ● p. 534

any k, face-to-face, + no faults, + equal speeds
Line, any number of face-to-face robots ● p. 534

+ triangle
Number of robots

+ two robots
Communication

SEARCH

Claude (Anthropic) was used in cataloguing the problems.

face-to-face, + no faults, + equal speeds
Two face-to-face robots, equilateral triangle ● p. 535

+ wireless, + no faults, + equal speeds
Triangle, two wireless robots at the centroid ● p. 535

any k, face-to-face, + no faults, + equal speeds
Triangle, any number of face-to-face robots ● p. 536

any k, + wireless, + no faults, + equal speeds
Triangle, any number of wireless robots at the centroid ● p. 536

+ square
Number of robots

+ two robots, + no faults, + equal speeds
Two face-to-face robots, square ● p. 537

any k, + wireless, + no faults, + equal speeds
Square, any number of wireless robots at the centre ● p. 537

+ triangle, boundary start
Number of robots

+ two robots, + wireless, + no faults, + equal speeds
Triangle, two wireless robots starting on the boundary ● p. 538

any k, + wireless, + no faults, + equal speeds
Triangle, any number of wireless robots starting on the boundary ● p. 538

+ rays, + wireless, + no faults, + equal speeds
m rays, $k = m$ wireless robots at the junction ● p. 539

+ square, boundary start, + two robots, + wireless, + no faults, + equal speeds
Square, two wireless robots starting on the boundary ● p. 539

SEARCH

What is known

The best upper bound against the best lower bound in each catalogued setting, in tree order. The settings follow in detail.

Setting	Evacuation time	p.
Disk, two wireless robots of different speeds	● polynomial gap $\frac{1+2\arccos(-2/v)}{v} + \sqrt{1-4/v^2}$ vs matching for $v \geq 2.75$	516
Two wireless robots, disk STANDARD SETTING	● tight $1 + 2\pi/3 + \sqrt{3}$ vs $1 + 2\pi/3 + \sqrt{3}$	517
Two robots, two exits, face-to-face	● tight $1 + \pi/2$ vs $1 + \pi/2$	517
Two robots, two exits, wireless	● polynomial gap $1 + \pi/3$ vs 2	518
Disk, search-and-fetch, two wireless robots	● unknown $1 + \pi - \alpha + 4\sin(\alpha/2)$ vs —	518
Disk, search-and-fetch, two face-to-face robots	● polynomial gap $1 + \pi - \alpha/2 + 3\sin(\alpha/2)$ vs $1 + \pi/3 + 4\sin(\alpha/2)$	519
Two face-to-face robots, disk	● polynomial gap ≈ 5.6234 vs $3 + \pi/6 + \sqrt{3}$	519
Two robots with asymmetric communication, disk	● polynomial gap $\pi + 2$ vs $> 1 + 2\pi/3 + \sqrt{3}$	520

Claude (Anthropic) was used in cataloguing the problems.

Setting	Evacuation time	p.
Disk, priority evacuation, queen and one servant	● polynomial gap ≈ 4.81854 vs ≈ 4.56798	520
Disk, weighted evacuation of two wireless robots	● polynomial gap $1 + d + \frac{2w}{w+1} \sin d$ vs $1 + \pi$	521
Three wireless robots, disk	● polynomial gap < 4.22 vs ≈ 4.159	522
Three wireless robots, one crash fault	● polynomial gap ≈ 6.309 vs $1 + 4\pi/3 \approx 5.188$	522
Three wireless robots, one Byzantine	● polynomial gap $1 + 4\pi/3 + \sqrt{3} \approx 6.921$ vs ≈ 5.948	523
Three face-to-face robots, disk	● polynomial gap $3 + 2\pi/3$ vs ≈ 4.519	523
Disk, priority evacuation, queen and two servants	● polynomial gap ≈ 3.8327 vs ≈ 3.6307	524
Many wireless robots, disk	● near-optimal $3 + \pi/k + O(k^{-4/3})$ vs $3 + \pi/k$	524
Disk, k wireless robots, f Byzantine	● polynomial gap $1 + 2\pi(f+1)/k + \max\{G_\epsilon(k^*), H_\epsilon(k^*)\}$ vs $1 + 2\pi(f+1)/k$	525
Disk search, k wireless robots, f crash faults	● tight $1 + 2\pi(f+1)/k$ vs $1 + 2\pi(f+1)/k$	525
Disk search, k wireless robots, one Byzantine	● tight $1 + 4\pi/k$ vs $1 + 4\pi/k$	526
Many face-to-face robots, disk	● near-optimal $3 + 2\pi/k$ vs $3 + 2\pi/k - O(k^{-2})$	527
Priority evacuation (queen and servants), disk	● polynomial gap $2 + 4(\sqrt{2}-1)\pi/\sigma$ vs $1 + \frac{2}{\sigma}\arccos\left(-\frac{2}{\sigma}\right) + \sqrt{1 - \frac{4}{\sigma^2}}$	527
Disk, k face-to-face robots, f Byzantine	● polynomial gap $3 + 2\pi(f+1)/k + \max_{2\le j\le k} 2(j-1)\sin\left(\frac{f-j+2}{j-1} \cdot \frac{\pi}{k}\right)$ vs $1 + 2\pi(f+1)/k$	528
Disk, k face-to-face robots, f crash faults	● polynomial gap $3 + 2\pi(f+1)/k$ vs $1 + 2\pi(f+1)/k$	528
Line, weighted evacuation of two wireless robots	● tight $1 + \sqrt{2w/(1+w)}$ vs $1 + \sqrt{2w/(1+w)}$	529
Line, weighted evacuation of two face-to-face robots	● polynomial gap 6.60258 to 6.82843 vs $1 + \sqrt{2w/(1+w)}$	530
Line, priority evacuation, queen and one wireless servant	● unknown $1 + \sqrt{2}$ vs —	530
Line, priority evacuation, queen and one face-to-face servant	● unknown $2 + \sqrt{5}$ vs —	531
Line, two face-to-face robots of different speeds	● tight $\frac{1+3v}{1-v}$ $(v \le \frac{1}{3})$, $9v$ $(v \ge \frac{1}{3})$ vs *matching*	531
Line, two wireless robots of different speeds	● tight $\frac{1+3v}{1-v}$ $(v \le \sqrt{17}-4)$, $\frac{2+v+\sqrt{v^2+8v}}{2}$ above vs *matching*	532
Line, a sender and a receiver	● tight $3 + 2\sqrt{2}$ vs $3 + 2\sqrt{2}$	532
Line, any number of wireless robots	● tight 3 vs 3	533
Line, $2f+1$ wireless robots, f crash faults	● polynomial gap ≈ 7.437 vs ≈ 5.233	534

Claude (Anthropic) was used in cataloguing the problems.

SEARCH

Setting	Evacuation time	p.
Line, any number of face-to-face robots	● tight 9 vs 9	534
Two face-to-face robots, equilateral triangle	● polynomial gap 2.3367 vs $1 + 2/\sqrt{3}$	535
Triangle, two wireless robots at the centroid	● tight $3/2 + \sqrt{3}/6$ vs $3/2 + \sqrt{3}/6$	535
Triangle, any number of face-to-face robots	● near-optimal $\sqrt{3} + 3/k$ vs $\sqrt{3}$	536
Triangle, any number of wireless robots at the centroid	● tight $1 + \sqrt{3}/3$ ($k \geq 8$) vs $1 + \sqrt{3}/3$	536
Two face-to-face robots, square	● polynomial gap 3.4645 vs 3.118	537
Square, any number of wireless robots at the centre	● near-optimal $3\sqrt{2}/2 + 4/k$ vs $3\sqrt{2}/2$	537
Triangle, two wireless robots starting on the boundary	● tight $(a + b + c)/2$ vs $(a + b + c)/2$	538
Triangle, any number of wireless robots starting on the boundary	● tight $3/2 + y$ vs $3/2 + y$	538
m rays, $k = m$ wireless robots at the junction	● tight 3 vs 3	539
Square, two wireless robots starting on the boundary	● tight $1 + x + \sqrt{1 + (1 - 2x)^2}$ vs $1 + x + \sqrt{1 + (1 - 2x)^2}$	539

SEARCH

20.2.1 Disk, two wireless robots of different speeds

Adds all robots, one exit, + wireless, + no faults VERIFIED

With speeds 1 *and* $v \geq 1$*, two wireless robots evacuate optimally for* $v \geq 2.75$*; for smaller* v *the bounds are within a factor* 1.22*.*

Assumed: domain: unit disk, start at the centre; number of robots: two robots; communication: wireless: instant communication at any distance; faults: no faults. **Not assumed:** goal: every robot must reach the exit; exits: one exit; speeds: robots may have different maximum speeds; randomness: deterministic.

Measure	Lower bound	Best upper bound	Status
Evacuation time	matching for $v \geq 2.75$ (Lamprou et al. 2020) Tight for $v \geq 2.75$; a constant factor below it.	$\frac{1+2\arccos(-2/v)}{v} + \sqrt{1 - 4/v^2}$ (Lamprou et al. 2020)	● polynomial gap

Results.

- Lamprou et al. (2020) [lower bound]: evacuation time matching for $v \geq 2.75$. *Bottleneck:* unknown exit. *Holds when:* speeds 1 and $v \geq 1$, both known; Section 5.
- Lamprou et al. (2020) [current best]: evacuation time $\frac{1+2\arccos(-2/v)}{v} + \sqrt{1 - 4/v^2}$. Introductory and current best: optimal (Half-Chord) for $v \geq 2.75$; for $1 \leq v < 2.75$ the best of three algorithms, within a factor 1.22 of the lower bound.

Open questions.

Q1. Optimal evacuation for $1 < v < 2.75$, for more than two robots, and face-to-face (Lamprou et al. 2020).

Claude (Anthropic) was used in cataloguing the problems.

Notes. Checked against the full text (arXiv version).

20.2.2 Two wireless robots, disk

Adds + equal speeds **to** Disk, two wireless robots of different speeds (p. 516) STANDARD SETTING VERIFIED

Two wireless robots evacuate the unit disk in exactly $1 + 2\pi/3 + \sqrt{3} \approx 4.826$.

Assumed: domain: unit disk, start at the centre; number of robots: two robots; communication: wireless: instant communication at any distance; faults: no faults; speeds: all robots have the same unit speed. **Not assumed:** goal: every robot must reach the exit; exits: one exit; randomness: deterministic.

Measure	Lower bound	Best upper bound	Status
Evacuation time	$1 + 2\pi/3 + \sqrt{3}$ (Czyzowicz et al. 2014) ≈ 4.826, exact.	$1 + 2\pi/3 + \sqrt{3}$ (Czyzowicz et al. 2014)	• tight

Results.

- Czyzowicz et al. (2014) [lower bound]: evacuation time $1 + 2\pi/3 + \sqrt{3}$. *Bottleneck:* unknown exit. *Holds when:* deterministic; start at the centre.
- Czyzowicz et al. (2014) [current best]: evacuation time $1 + 2\pi/3 + \sqrt{3}$. Introductory and current best.

Notes. With one robot that has crashed and must be carried out, see Chauffeuring. In the unit disk of an ℓ_p norm, two wireless robots evacuate optimally for every $p \geq 1$, in time 5 for $p = 1$ and $p = \infty$ and about 4.7544 at best (Georgiou et al. 2023a). If every robot must visit every one of k treasures, $k = 1$ is this problem (Kaminski et al. 2026), a preprint.

20.2.3 Two robots, two exits, face-to-face

Adds face-to-face, + no faults, + equal speeds VERIFIED

With two exits at unknown positions and communication only on meeting, the optimal evacuation time is $(1 + \pi/2)R \approx 2.57R$.

Assumed: domain: unit disk, start at the centre; number of robots: two robots; exits: two exits at unknown positions; faults: no faults; speeds: all robots have the same unit speed. **Not assumed:** goal: every robot must reach the exit; communication: face-to-face: robots exchange information only when they meet; randomness: deterministic.

Measure	Lower bound	Best upper bound	Status
Evacuation time	$1 + \pi/2$ (Pattanayak et al. 2018) Optimal: $(1 + \pi/2)R$.	$1 + \pi/2$ (Pattanayak et al. 2018)	• tight

Results.

- Pattanayak et al. (2018) [lower bound]: evacuation time $1 + \pi/2$. *Bottleneck:* unknown exits. *Holds when:* two robots start at the centre; radius $R = 1$; oblivious robots.
- Pattanayak et al. (2018) [current best]: evacuation time $1 + \pi/2$. Introductory and current best; optimal.

Notes. Bounds from the full version, supplied by the catalogue owner (a co-author); the robots are oblivious.

Claude (Anthropic) was used in cataloguing the problems.

SEARCH

20.2.4 Two robots, two exits, wireless

Adds + wireless, + no faults, + equal speeds VERIFIED

With two exits and wireless communication, evacuation takes between $2R$ and $(1+\pi/3)R \approx 2.047R$.

Assumed: domain: unit disk, start at the centre; number of robots: two robots; exits: two exits at unknown positions; communication: wireless: instant communication at any distance; faults: no faults; speeds: all robots have the same unit speed. **Not assumed:** goal: every robot must reach the exit; randomness: deterministic.

Measure	Lower bound	Best upper bound	Status
Evacuation time	2 (Pattanayak et al. 2018) $(1+\pi/3)R$ versus $2R$: a constant gap.	$1+\pi/3$ (Pattanayak et al. 2018)	• polynomial gap

Results.

- Pattanayak et al. (2018) [lower bound]: evacuation time 2. *Bottleneck:* unknown exits. *Holds when:* two robots start at the centre; radius $R = 1$.
- Pattanayak et al. (2018) [current best]: evacuation time $1+\pi/3$. Introductory and current best.

Open questions.

Q1. Close the gap between $2R$ and $(1+\pi/3)R$ for two wireless robots and two exits (Pattanayak et al. 2018).

Notes. Bounds supplied by the catalogue owner (a co-author). When two wireless robots move only along a circle of unit length, know the arrangement of the exits but not their own position, and start at arc distance $L \le 1/2$, they evacuate through one exit optimally in $3/4 + L/2$ (Theorem 1) (Czyzowicz et al. 2018a).

20.2.5 Disk, search-and-fetch, two wireless robots

Adds + wireless, + no faults VERIFIED

Two wireless robots find a treasure on the boundary and bring it to an exit at known arc distance α in $1+\pi-\alpha+4\sin(\alpha/2)$.

Assumed: domain: unit disk, start at the centre; number of robots: two robots; goal: search-and-fetch: a treasure must be brought to the exit; communication: wireless: instant communication at any distance; faults: no faults. **Not assumed:** exits: one exit; speeds: robots may have different maximum speeds; randomness: deterministic.

Measure	Lower bound	Best upper bound	Status
Evacuation time	none known No lower bound is given.	$1+\pi-\alpha+4\sin(\alpha/2)$ (Georgiou et al. 2019)	• unknown

Results.

- Georgiou et al. (2019) [current best]: evacuation time $1+\pi-\alpha+4\sin(\alpha/2)$. Introductory and current best (Theorem 2.1).

Open questions.

Q1. Lower bounds that separate wireless from face-to-face search-and-fetch (Georgiou et al. 2019).

Claude (Anthropic) was used in cataloguing the problems.

Notes. With one robot, treasure evacuation is solved optimally when only a lower bound α on the actual treasure–exit arc distance λ is known: time $1 + 2\pi - \alpha + 2\sin(\alpha/2) + 2\sin(\lambda/2)$ (Theorem 2; the paper writes l for λ); when the exact distance is known, the best algorithm is within a factor 1.019 (additively 0.157) of a lower bound (Theorems 7, 8) (Georgiou et al. 2021). Checked against the full texts of (Georgiou et al. 2019) and (Georgiou et al. 2021).

20.2.6 Disk, search-and-fetch, two face-to-face robots

Adds face-to-face, + no faults VERIFIED

Face-to-face, two robots fetch the treasure in $1 + \pi - \alpha/2 + 3\sin(\alpha/2)$*, against* $1 + \pi/3 + 4\sin(\alpha/2)$ *for* $\alpha \le 2\pi/3$.

Assumed: domain: unit disk, start at the centre; number of robots: two robots; goal: search-and-fetch: a treasure must be brought to the exit; faults: no faults. **Not assumed:** exits: one exit; communication: face-to-face: robots exchange information only when they meet; speeds: robots may have different maximum speeds; randomness: deterministic.

Measure	Lower bound	Best upper bound	Status
Evacuation time	$1 + \pi/3 + 4\sin(\alpha/2)$ (Georgiou et al. 2019) Constant gap.	$1 + \pi - \alpha/2 + 3\sin(\alpha/2)$ (Georgiou et al. 2019)	• polynomial gap

Results.

- Georgiou et al. (2019) [lower bound]: evacuation time $1 + \pi/3 + 4\sin(\alpha/2)$. *Bottleneck:* unknown exit. *Holds when:* Theorem 4.1; $\alpha \le 2\pi/3$; $1 + \pi/3 + 2\sin\alpha + 2\sin(\alpha/2)$ for larger α.
- Georgiou et al. (2019) [current best]: evacuation time $1 + \pi - \alpha/2 + 3\sin(\alpha/2)$. Introductory and current best (Theorem 3.1).

Notes. Checked against the full text.

20.2.7 Two face-to-face robots, disk

Adds all robots, one exit, face-to-face, + no faults, + equal speeds VERIFIED

Face-to-face, two robots need between ≈ 5.255 *and* ≈ 5.6234*; the gap has been open since 2019.*

Assumed: domain: unit disk, start at the centre; number of robots: two robots; faults: no faults; speeds: all robots have the same unit speed. **Not assumed:** goal: every robot must reach the exit; exits: one exit; communication: face-to-face: robots exchange information only when they meet; randomness: deterministic.

Measure	Lower bound	Best upper bound	Status
Evacuation time	$3 + \pi/6 + \sqrt{3}$ (Czyzowicz et al. 2020d) ≈ 5.6234 versus ≈ 5.255: a constant gap of about 0.37.	≈ 5.6234 (Disser and Schmitt 2019)	• polynomial gap

Results.

- Czyzowicz et al. (2020d) [lower bound]: evacuation time $3 + \pi/6 + \sqrt{3}$. *Bottleneck:* unknown exit. *Holds when:* face-to-face; improves $3 + \pi/4 + \sqrt{2}$ of (Czyzowicz et al. 2014).
- Czyzowicz et al. (2014) [introductory]: evacuation time ≈ 5.74.
- Czyzowicz et al. (2020d) [also]: evacuation time ≈ 5.628.
- Brandt et al. (2017) [also]: evacuation time ≈ 5.625. Forced meetings exchange no useful information; simpler algorithm.

Claude (Anthropic) was used in cataloguing the problems.

SEARCH

- Disser and Schmitt (2019) [current best]: evacuation time ≈ 5.6234. A second detour through the interior.

Open questions.

Q1. Close the gap between $3 + \pi/6 + \sqrt{3} \approx 5.255$ and 5.6234 for two face-to-face robots.

Notes. The "second cut" of (Disser and Schmitt 2019) is a second detour through the interior: an upper-bound technique. Measured in the average case over the exit position, the trade-off between average and worst-case time is mapped, and $1 + \pi$ is conjectured to be the optimal average for any number of robots, even wireless (Chuangpishit et al. 2018b, 2020a).

20.2.8 Two robots with asymmetric communication, disk

Adds all robots, one exit, + sender/receiver, + no faults, + equal speeds VERIFIED

When one robot can only send and the other only receive, evacuation lies strictly between the wireless and face-to-face optima; $\pi + 2$ suffices.

Assumed: domain: unit disk, start at the centre; number of robots: two robots; communication: asymmetric: senders and receivers; faults: no faults; speeds: all robots have the same unit speed. **Not assumed:** goal: every robot must reach the exit; exits: one exit; randomness: deterministic.

Measure	Lower bound	Best upper bound	Status
Evacuation time	$> 1 + 2\pi/3 + \sqrt{3}$ (Georgiou et al. 2022) No explicit lower bound.	$\pi + 2$ (Georgiou et al. 2022)	• polynomial gap

Results.

- Georgiou et al. (2022) [lower bound]: evacuation time $> 1 + 2\pi/3 + \sqrt{3}$. *Bottleneck:* one-way communication. *Holds when:* strictly above the wireless optimum; no explicit value.
- Georgiou et al. (2022) [current best]: evacuation time $\pi + 2$. Introductory and current best.

Open questions.

Q1. Give an explicit lower bound above ≈ 4.826 for sender/receiver evacuation.

20.2.9 Disk, priority evacuation, queen and one servant

Adds + priority (queen), + wireless, + no faults VERIFIED

A queen and one servant evacuate the queen in about 4.819*, strictly faster than evacuating both; the lower bound is about* 4.568*.*

Assumed: domain: unit disk, start at the centre; number of robots: two robots; goal: priority: only the queen must reach the exit; communication: wireless: instant communication at any distance; faults: no faults. **Not assumed:** exits: one exit; speeds: robots may have different maximum speeds; randomness: deterministic.

Measure	Lower bound	Best upper bound	Status
Evacuation time	≈ 4.56798 (Georgiou and Wang 2025) About 4.819 against 4.568.	≈ 4.81854 (Georgiou and Wang 2025)	• polynomial gap

Claude (Anthropic) was used in cataloguing the problems.

Results.

- Georgiou and Wang (2025) [lower bound]: evacuation time ≈ 4.56798. *Bottleneck:* unknown exit. *Holds when:* Theorem 1; a linear-programming relaxation over polygon instances.
- Czyzowicz et al. (2020a) [introductory]: evacuation time ≈ 4.81854. Theorem 8; also the first lower bound, about 4.3896 (Theorem 17).
- Georgiou and Wang (2025) [current best]: evacuation time ≈ 4.81854. Same algorithm, with the improved lower bound.

Open questions.

Q1. Close the gap between 4.568 and 4.819 (Georgiou and Wang 2025).

Notes. Weighted group search, where the two robots' arrival times are weighted 1 and w, is bounded in the same paper (Georgiou and Wang 2025); see the weighted setting in the disk. Checked against the full texts.

20.2.10 Disk, weighted evacuation of two wireless robots

Adds + weighted, + wireless, + no faults, + equal speeds VERIFIED

For two wireless robots in the disk the weighted cost is exactly $1+\pi$ *when* $w=1$*; for larger* w *it lies between* $\max(1+\pi,\ 3.618+0.782\,w/(w+1))$ *and* $1+d+\frac{2w}{w+1}\sin d$*, tending to priority evacuation as* $w\to\infty$.

Assumed: domain: unit disk, start at the centre; number of robots: two robots; goal: weighted: a weighted average of two arrival times; communication: wireless: instant communication at any distance; faults: no faults; speeds: all robots have the same unit speed. **Not assumed:** exits: one exit; randomness: deterministic.

Statement for this setting. Two wireless robots with known identities start at the centre of the unit disk; the heavy robot's arrival time counts $w \geq 1$ times the light one's, and the cost is $(t_1 + w\,t_2)/(1+w)$ in the worst case over exit positions.

Measure	Lower bound	Best upper bound	Status
Evacuation time	$1+\pi$ (Georgiou and Wang 2025); $1+\pi/7+5\sin(\pi/7)+\frac{w}{w+1}\cos(3\pi/14)$ (Georgiou and Wang 2025)	$1+d+\frac{2w}{w+1}\sin d$ (Georgiou and Wang 2025)	• polynomial gap
	Tight $(1+\pi)$ at $w=1$; a constant gap otherwise, up to 4.568 against 4.819 in the limit of priority evacuation.		

Results.

- Georgiou and Wang (2025) [lower bound]: evacuation time $1+\pi$. *Bottleneck:* unknown exit. *Holds when:* Lemma 7 (arXiv numbering); the two robots together need time $1+\pi$ to search the whole boundary.
- Georgiou and Wang (2025) [lower bound]: evacuation time $1+\pi/7+5\sin(\pi/7)+\frac{w}{w+1}\cos(3\pi/14)$. *Bottleneck:* unknown exit. *Holds when:* Theorem 2 (arXiv numbering), about $3.618+0.782\,w/(w+1)$; above $1+\pi$ for large w; linear-programming relaxation over exits at the vertices of a regular 7-gon, computer-assisted (Lemmas 4, 8).
- Georgiou and Wang (2025) [current best]: evacuation time $1+d+\frac{2w}{w+1}\sin d$. Introductory and current best (Lemmas 5, 6, arXiv numbering): the robots split the boundary in halves, $d=\arccos(-(w+1)/(2w))$, while $w \leq 21.8861$; for larger w the heavy robot leaves the boundary early (a detour, with d the solution of a non-linear system), which as $w\to\infty$ is the priority algorithm of cost ≈ 4.81854.

Claude (Anthropic) was used in cataloguing the problems.

SEARCH

Open questions.

Q1. Close the gap for weighted evacuation of two wireless robots in the disk when $w > 1$ (Georgiou and Wang 2025).

Notes. The paper writes the cost as $w'x + y$ with $w' \in [0, 1]$; its w' is $1/w$ here. For $w \to \infty$ (their $w' = 0$) the cost is priority evacuation, where the same framework gives the lower bound ≈ 4.56798 (Theorem 1); see the setting of a queen and one servant in the disk. Checked against the full text (arXiv 2406.19490v1; theorem numbers may differ in the journal version).

20.2.11 Three wireless robots, disk

Adds + no faults, + equal speeds VERIFIED

Three wireless robots: between ≈ 4.159 and 4.22.

Assumed: domain: unit disk, start at the centre; number of robots: three robots; communication: wireless: instant communication at any distance; faults: no faults; speeds: all robots have the same unit speed. **Not assumed:** goal: every robot must reach the exit; exits: one exit; randomness: deterministic.

Measure	Lower bound	Best upper bound	Status
Evacuation time	≈ 4.159 (Czyzowicz et al. 2014) A small constant gap.	< 4.22 (Czyzowicz et al. 2014)	• polynomial gap

Results.

- Czyzowicz et al. (2014) [lower bound]: evacuation time ≈ 4.159. *Bottleneck:* unknown exit. *Holds when:* three robots, wireless.
- Czyzowicz et al. (2014) [current best]: evacuation time < 4.22. Introductory and current best.

Open questions.

Q1. Find the optimal evacuation time for three wireless robots.

20.2.12 Three wireless robots, one crash fault

Adds + crash faults, + equal speeds VERIFIED

If one of three wireless robots may crash, evacuation takes between $1 + 4\pi/3 \approx 5.188$ and ≈ 6.309.

Assumed: domain: unit disk, start at the centre; number of robots: three robots; communication: wireless: instant communication at any distance; faults: up to f crash-faulty robots (fail to report the target); speeds: all robots have the same unit speed. **Not assumed:** goal: every robot must reach the exit; exits: one exit; randomness: deterministic.

Measure	Lower bound	Best upper bound	Status
Evacuation time	$1 + 4\pi/3 \approx 5.188$ (Czyzowicz et al. 2017c) Constant gap.	≈ 6.309 (Czyzowicz et al. 2017c)	• polynomial gap

Results.

- Czyzowicz et al. (2017c) [lower bound]: evacuation time $1 + 4\pi/3 \approx 5.188$. *Bottleneck:* faulty reports. *Holds when:* Theorem 3(a); three robots, at most one crash; every boundary point must be visited by two robots, so some robot explores an arc of length at least $4\pi/3$.

Claude (Anthropic) was used in cataloguing the problems.

- Czyzowicz et al. (2017c) [current best]: evacuation time ≈ 6.309. Introductory and current best (Theorem 1; the best asymmetric-persistent algorithm, Lemma 3); time until the last non-faulty robot exits.

Open questions.

Q1. Tight bounds for disk evacuation with one crash-faulty robot among three.

Notes. The best symmetric-persistent algorithm, with all three robots exploring in the same direction, takes $1 + 4\pi/3 + \sqrt{3} \approx 6.921$ (Lemma 1) (Czyzowicz et al. 2017c). The survey (Czyzowicz et al. 2019d) (Theorem 17) states a weaker crash lower bound of ≈ 5.082; the SIROCCO paper proves $1 + 4\pi/3$. Checked against the full text of (Czyzowicz et al. 2017c).

20.2.13 Three wireless robots, one Byzantine

Adds Byzantine, + equal speeds VERIFIED

With one Byzantine robot among three: between ≈ 5.948 *and* ≈ 6.921.

Assumed: domain: unit disk, start at the centre; number of robots: three robots; communication: wireless: instant communication at any distance; speeds: all robots have the same unit speed. **Not assumed:** goal: every robot must reach the exit; exits: one exit; faults: up to f Byzantine robots (may lie or stay silent); randomness: deterministic.

Measure	Lower bound	Best upper bound	Status
Evacuation time	≈ 5.948 (Czyzowicz et al. 2017c) Constant gap.	$1 + 4\pi/3 + \sqrt{3} \approx 6.921$ (Czyzowicz et al. 2017c)	• polynomial gap

Results.

- Czyzowicz et al. (2017c) [lower bound]: evacuation time ≈ 5.948. *Bottleneck:* indistinguishable Byzantine agents. *Holds when:* Theorem 3(b); three robots, at most one Byzantine.
- Czyzowicz et al. (2017c) [current best]: evacuation time $1 + 4\pi/3 + \sqrt{3} \approx 6.921$. Introductory and current best (Theorem 2).

Notes. Checked against the full text of (Czyzowicz et al. 2017c). For k wireless robots of which one may be faulty, Leonardos, Pagourtzis and Papaioannou prove that symmetric-persistent algorithms (all robots go to the boundary and explore in the same direction until told of the exit) need time at least $1 + 4\pi/k + 2\sin(\pi/2 - \pi/k)$ (Theorem 1, proved with a crash-faulty robot), and give a symmetric-persistent algorithm with time at most $3 + 4\pi/k$ for one Byzantine robot (Theorem 3), which for $k = 3$ is about 7.19, above 6.921; with two Byzantine robots the time is at most $3 + 6\pi/k$ for $k \geq 9$ (Theorem 4); they also give face-to-face upper bounds (Theorems 5–7) (Leonardos et al. 2023). Behrouz et al. give upper bounds for f Byzantine robots among $k > 2f$ wireless robots (Theorem 1) and among k face-to-face robots (Theorem 2), and no lower bounds (Behrouz et al. 2025).

20.2.14 Three face-to-face robots, disk

Adds all robots, face-to-face, + no faults, + equal speeds VERIFIED

Three face-to-face robots: between ≈ 4.519 *and* $3 + 2\pi/3 \approx 5.09$.

Assumed: domain: unit disk, start at the centre; number of robots: three robots; faults: no faults; speeds: all robots have the same unit speed. **Not assumed:** goal: every robot must reach the exit; exits: one exit; communication: face-to-face: robots exchange information only when they meet; randomness: deterministic.

Claude (Anthropic) was used in cataloguing the problems.

SEARCH

Measure	Lower bound	Best upper bound	Status
Evacuation time	≈ 4.519 (Czyzowicz et al. 2014) Constant gap.	$3 + 2\pi/3$ (Czyzowicz et al. 2014)	• polynomial gap

Results.

- Czyzowicz et al. (2014) [lower bound]: evacuation time ≈ 4.519. *Bottleneck:* unknown exit. *Holds when:* three robots, face-to-face.
- Czyzowicz et al. (2014) [current best]: evacuation time $3 + 2\pi/3$. Introductory and current best.

20.2.15 Disk, priority evacuation, queen and two servants

Adds + priority (queen), + wireless, + no faults VERIFIED

A queen and two servants evacuate the queen in about 3.833 *against a lower bound of about* 3.631.

Assumed: domain: unit disk, start at the centre; number of robots: three robots; goal: priority: only the queen must reach the exit; communication: wireless: instant communication at any distance; faults: no faults. **Not assumed:** exits: one exit; speeds: robots may have different maximum speeds; randomness: deterministic.

Measure	Lower bound	Best upper bound	Status
Evacuation time	≈ 3.6307 (Czyzowicz et al. 2020a) Constant gap.	≈ 3.8327 (Czyzowicz et al. 2020a)	• polynomial gap

Results.

- Czyzowicz et al. (2020a) [lower bound]: evacuation time ≈ 3.6307. *Bottleneck:* unknown exit. *Holds when:* Theorem 19.
- Czyzowicz et al. (2020a) [current best]: evacuation time ≈ 3.8327. Introductory and current best (Theorem 10).

Notes. Checked against the full text.

20.2.16 Many wireless robots, disk

Adds + no faults, + equal speeds VERIFIED

With k wireless robots the time approaches $3 + \pi/k$.

Assumed: domain: unit disk, start at the centre; communication: wireless: instant communication at any distance; faults: no faults; speeds: all robots have the same unit speed. **Not assumed:** number of robots: any number k of robots; goal: every robot must reach the exit; exits: one exit; randomness: deterministic.

Measure	Lower bound	Best upper bound	Status
Evacuation time	$3 + \pi/k$ (Czyzowicz et al. 2014) Matching up to $O(k^{-4/3})$.	$3 + \pi/k + O(k^{-4/3})$ (Czyzowicz et al. 2014)	• near-optimal

Results.

- Czyzowicz et al. (2014) [lower bound]: evacuation time $3 + \pi/k$. *Bottleneck:* unknown exit. *Holds when:* k robots, wireless.
- Czyzowicz et al. (2014) [current best]: evacuation time $3 + \pi/k + O(k^{-4/3})$. Introductory and current best.

Claude (Anthropic) was used in cataloguing the problems.

20.2.17 Disk, k wireless robots, f Byzantine

Adds Byzantine, + equal speeds VERIFIED

With k wireless robots and one Byzantine robot evacuation takes at most $3 + 4\pi/k$; general $f < k/2$ has upper bounds only, and lower bounds come from search and from the fault-free case.

Assumed: domain: unit disk, start at the centre; communication: wireless: instant communication at any distance; speeds: all robots have the same unit speed. **Not assumed:** number of robots: any number k of robots; goal: every robot must reach the exit; exits: one exit; faults: up to f Byzantine robots (may lie or stay silent); randomness: deterministic.

Statement for this setting. k wireless robots start at the centre of the unit disk; up to $f < k/2$ of them are Byzantine. Every non-faulty robot must reach the exit; messages carry the sender's identifier.

Measure	Lower bound	Best upper bound	Status
Evacuation time	$1 + 2\pi(f+1)/k$ (Georgiou et al. 2023b); $3 + \pi/k$ (Czyzowicz et al. 2014)	$1 + 2\pi(f+1)/k + \max\{G_e(k^*), H_e(k^*)\}$ (Behrouz et al. 2025)	• polynomial gap
	For $f = 1$: $3 + 4\pi/k$ against $3 + \pi/k$, an additive gap $3\pi/k$; for general f the gap is not quantified.		

Results.

- Georgiou et al. (2023b) [lower bound]: evacuation time $1 + 2\pi(f+1)/k$. *Bottleneck:* faulty reports. *Holds when:* Theorem 1: search bound with f crash-faulty robots, hence with Byzantine ones; every point of the boundary must be visited by $f + 1$ robots; evacuation takes at least as long as search.
- Czyzowicz et al. (2014) [lower bound]: evacuation time $3 + \pi/k$. *Bottleneck:* unknown exit. *Holds when:* fault-free bound of k wireless robots (setting with many wireless robots in the disk).
- Leonardos et al. (2023) [introductory]: evacuation time $3+4\pi/k$ ($f = 1$). Theorem 3 (search algorithm of Georgiou et al. 2023b plus a diameter); $3 + 6\pi/k$ for $f = 2$ when $k \geq 9$, plus a term $\delta(k) \leq 0.5687$ when $k < 9$ (Theorem 4).
- Behrouz et al. (2025) [current best]: evacuation time $1 + 2\pi(f+1)/k + \max\{G_e(k^*), H_e(k^*)\}$. Theorem 1, for every $f < k/2$; G_e and H_e are the evacuation times of two groups of robots that settle disputed reports, defined in the theorem.

Open questions.

Q1. Lower bounds for evacuation of k robots with f Byzantine faults from the disk (Behrouz et al. 2025).

Notes. Symmetric-persistent algorithms (all robots go to the boundary and explore in the same direction until told of the exit) need at least $1 + 4\pi/k + 2\sin(\pi/2 - \pi/k)$ with one faulty robot (Theorem 1 of Leonardos et al. 2023), so $3 + 4\pi/k$ is within $O(1/k^2)$ of the best such algorithm. With $k = 3$ see the setting with three robots, where ≈ 6.921 is known (Czyzowicz et al. 2017c). Checked against the full texts of (Leonardos et al. 2023), (Behrouz et al. 2025) and (Georgiou et al. 2023b).

20.2.18 Disk search, k wireless robots, f crash faults

Adds + crash faults, + equal speeds VERIFIED

Search of the disk boundary with f crash faults takes exactly $1 + 2\pi(f+1)/k$: every point must be seen by $f + 1$ robots.

Assumed: domain: unit disk, start at the centre; goal: search: the exit is found and known to every non-faulty robot; communication: wireless: instant communication at any distance; faults: up to f crash-faulty robots (fail to report the target); speeds: all robots have the same unit speed. **Not assumed:** number of robots: any number k of robots; exits: one exit; randomness: deterministic.

Claude (Anthropic) was used in cataloguing the problems.

SEARCH

Statement for this setting. $k \geq f + 1$ wireless robots start at the centre of the unit disk; up to f may crash. The task ends when a non-faulty robot has reached the exit and every non-faulty robot knows where it is.

Measure	Lower bound	Best upper bound	Status
Evacuation time	$1 + 2\pi(f+1)/k$ (Georgiou et al. 2023b)	$1 + 2\pi(f+1)/k$ (Georgiou et al. 2023b)	• tight
	Exact.		

Results.

- Georgiou et al. (2023b) [lower bound]: evacuation time $1 + 2\pi(f+1)/k$. *Bottleneck:* faulty reports. *Holds when:* Theorem 1; every point of the boundary must be visited by $f + 1$ robots.
- Georgiou et al. (2023b) [current best]: evacuation time $1 + 2\pi(f+1)/k$. Introductory and current best (Theorem 2): robot i explores $f + 1$ consecutive arcs of length $2\pi/k$ from the point $2\pi i/k$.

Notes. Checked against the full text.

20.2.19 Disk search, k wireless robots, one Byzantine

Adds Byzantine, + equal speeds VERIFIED

With one Byzantine robot, search takes exactly $1 + 4\pi/k$, as with one crash fault.

Assumed: domain: unit disk, start at the centre; goal: search: the exit is found and known to every non-faulty robot; communication: wireless: instant communication at any distance; speeds: all robots have the same unit speed. **Not assumed:** number of robots: any number k of robots; exits: one exit; faults: up to f Byzantine robots (may lie or stay silent); randomness: deterministic.

Statement for this setting. $k \geq 2$ wireless robots start at the centre of the unit disk; one of them may be Byzantine (it may lie about the exit). Messages carry the sender's identifier. The task ends when a non-faulty robot has reached the exit and every non-faulty robot knows its position for certain.

Measure	Lower bound	Best upper bound	Status
Evacuation time	$1 + 4\pi/k$ (Georgiou et al. 2023b)	$1 + 4\pi/k$ (Georgiou et al. 2023b)	• tight
	Exact for one Byzantine robot.		

Results.

- Georgiou et al. (2023b) [lower bound]: evacuation time $1 + 4\pi/k$. *Bottleneck:* faulty reports. *Holds when:* Corollary 1: the crash bound of Theorem 1 with $f = 1$.
- Georgiou et al. (2023b) [current best]: evacuation time $1 + 4\pi/k$. Introductory and current best (Theorem 3, with separate analyses for $k = 3, 4$ in Lemma 1 and Section 4.2).

Open questions.

Q1. Search of the disk with $f \geq 2$ Byzantine robots: only the crash lower bound $1 + 2\pi(f+1)/k$ and evacuation upper bounds are known (Georgiou et al. 2023b; Behrouz et al. 2025).

Notes. With one Byzantine and $f - 1$ crash-faulty robots, search takes at most $1 + 2\pi f/k + 2\sin(2\pi/k)$ (Theorem 4). Checked against the full text of (Georgiou et al. 2023b).

Claude (Anthropic) was used in cataloguing the problems.

20.2.20 Many face-to-face robots, disk

Adds any k, all robots, face-to-face, + no faults, + equal speeds VERIFIED

With k face-to-face robots the time is $3 + 2\pi/k$ up to $O(k^{-2})$; the separation from wireless is a factor 2 in the second term.

Assumed: domain: unit disk, start at the centre; faults: no faults; speeds: all robots have the same unit speed. **Not assumed:** number of robots: any number k of robots; goal: every robot must reach the exit; exits: one exit; communication: face-to-face: robots exchange information only when they meet; randomness: deterministic.

Measure	Lower bound	Best upper bound	Status
Evacuation time	$3 + 2\pi/k - O(k^{-2})$ (Czyzowicz et al. 2014) Matching up to $O(k^{-2})$.	$3 + 2\pi/k$ (Czyzowicz et al. 2014)	• near-optimal

Results.

- Czyzowicz et al. (2014) [lower bound]: evacuation time $3 + 2\pi/k - O(k^{-2})$. *Bottleneck:* unknown exit. *Holds when:* k robots, face-to-face.
- Czyzowicz et al. (2014) [current best]: evacuation time $3 + 2\pi/k$. Introductory and current best.

20.2.21 Priority evacuation (queen and servants), disk

Adds any k, + priority (queen), + wireless, + no faults, + equal speeds VERIFIED

Only the queen must leave; σ servants (the paper writes n; $k = \sigma + 1$ robots) help her find the exit.

Assumed: domain: unit disk, start at the centre; goal: priority: only the queen must reach the exit; communication: wireless: instant communication at any distance; faults: no faults; speeds: all robots have the same unit speed. **Not assumed:** number of robots: any number k of robots; exits: one exit; randomness: deterministic.

Measure	Lower bound	Best upper bound	Status
Evacuation time	$1 + \frac{2}{\sigma}\arccos\left(-\frac{2}{\sigma}\right) + \sqrt{1 - \frac{4}{\sigma^2}}$ (Czyzowicz et al. 2020b) For large σ the gap between the closed-form bounds is about $(4\sqrt{2} - 5)\pi/\sigma \approx 0.657\pi/\sigma$ (Czyzowicz et al. 2020b).	$2 + 4(\sqrt{2} - 1)\pi/\sigma$ (Czyzowicz et al. 2020b)	• polynomial gap

Results.

- Czyzowicz et al. (2020b) [lower bound]: evacuation time $1 + \frac{2}{\sigma}\arccos\left(-\frac{2}{\sigma}\right) + \sqrt{1 - \frac{4}{\sigma^2}}$. *Bottleneck:* unknown exit. *Holds when:* $\sigma \geq 4$ servants, wireless; all robots start at the centre of a unit disk. Theorem 5 of (Czyzowicz et al. 2020b); it exceeds $2 + \pi/\sigma + 2/\sigma^2$, the leading terms of its expansion in $1/\sigma$, which is the form stated in the abstract (e.g. 2.913 against 2.910 for $\sigma = 4$). The naive bound is $2 + \pi/(\sigma + 1)$.
- Czyzowicz et al. (2020b) [current best]: evacuation time $2 + 4(\sqrt{2} - 1)\pi/\sigma$. *Assumes:* $\sigma \geq 4$ servants. Introductory and current best (Theorem 1). The exact times of the algorithm are computed numerically, e.g. 3.113 for $\sigma = 4$ against the closed-form upper bound 3.301.

Open questions.

Q1. Close the gap for priority evacuation; for one servant the bounds differ by about 0.25 (a 2024 preprint, not yet in the bibliography).

Notes. For a queen and three servants the bounds are about 3.374 and 3.202 (Czyzowicz et al. 2020a). The authors conjecture their algorithm for $\sigma \geq 4$ servants is optimal (Czyzowicz et al. 2020b). The lower

Claude (Anthropic) was used in cataloguing the problems.

SEARCH

bound was checked against the full text (arXiv version) of (Czyzowicz et al. 2020b): Section 4 proves the exact expression and the abstract states its asymptotic form $2 + \pi/\sigma + 2/\sigma^2$, so the two agree.

20.2.22 Disk, k face-to-face robots, f Byzantine

Adds any k, all robots, face-to-face, Byzantine, + equal speeds VERIFIED

Face-to-face with f Byzantine robots: $3 + 4\pi/k + 2\sin(\pi/k)$ for $f = 1$, and a general upper bound; lower bounds come from search and the fault-free case.

Assumed: domain: unit disk, start at the centre; speeds: all robots have the same unit speed. **Not assumed:** number of robots: any number k of robots; goal: every robot must reach the exit; exits: one exit; communication: face-to-face: robots exchange information only when they meet; faults: up to f Byzantine robots (may lie or stay silent); randomness: deterministic.

Statement for this setting. k robots start at the centre of the unit disk and exchange information only when they meet; up to f of them are Byzantine. Every non-faulty robot must reach the exit.

Measure	Lower bound	Best upper bound	Status
Evacuation time	$1 + 2\pi(f+1)/k$ (Georgiou et al. 2023b); $3 + 2\pi/k - O(k^{-2})$ (Czyzowicz et al. 2014) Constant gap.	$3 + 2\pi(f+1)/k + \max_{2\le j\le k} 2(j-1)\sin\left(\frac{f-j+2}{j-1}\cdot\frac{\pi}{k}\right)$ (Behrouz et al. 2025)	• polynomial gap

Results.

- Georgiou et al. (2023b) [lower bound]: evacuation time $1 + 2\pi(f+1)/k$. *Bottleneck:* faulty reports. *Holds when:* Theorem 1: search bound with f crash-faulty robots, hence with Byzantine ones; every point of the boundary must be visited by $f+1$ robots; wireless search bound; it holds face-to-face and for evacuation.
- Czyzowicz et al. (2014) [lower bound]: evacuation time $3 + 2\pi/k - O(k^{-2})$. *Bottleneck:* unknown exit. *Holds when:* fault-free bound of k face-to-face robots (setting with many face-to-face robots in the disk).
- Leonardos et al. (2023) [introductory]: evacuation time $3 + 4\pi/k + 2\sin(\pi/k)$ ($f = 1$). Theorem 6; $3 + 6\pi/k + 2\sin(2\pi/k)$ for $f = 2$ (Theorem 7).
- Behrouz et al. (2025) [current best]: evacuation time $3 + 2\pi(f+1)/k + \max_{2\le j\le k} 2(j-1)\sin\left(\frac{f-j+2}{j-1}\cdot\frac{\pi}{k}\right)$. Theorem 2, for every f (their k is j here, the number of conflicting reports); equals the bounds of (Leonardos et al. 2023) for $f = 1, 2$. Against the trivial algorithm (every robot walks the whole boundary, time $1 + 2\pi$): the trivial one is better when $f \ge 0.3842\,k$ (Lemma 6), Theorem 2 is better when $f \le \beta k$ with $\beta < (\pi - 1)/(2\pi) \approx 0.34$ and k large enough (Lemma 7).

Open questions.

Q1. Lower bounds for face-to-face evacuation of the disk with Byzantine robots (Behrouz et al. 2025).

Notes. Checked against the full texts of (Leonardos et al. 2023), (Behrouz et al. 2025) and (Georgiou et al. 2023b).

20.2.23 Disk, k face-to-face robots, f crash faults

Adds any k, all robots, face-to-face, + crash faults, + equal speeds VERIFIED

Face-to-face with f crash faults, k robots evacuate in at most $3 + 2\pi(f+1)/k$.

Assumed: domain: unit disk, start at the centre; faults: up to f crash-faulty robots (fail to report the target); speeds: all robots have the same unit speed. **Not assumed:** number of robots: any number k of robots; goal: every robot must reach the exit; exits: one exit; communication: face-to-face: robots exchange information only when they meet; randomness: deterministic.

Claude (Anthropic) was used in cataloguing the problems.

Statement for this setting. k robots start at the centre of the unit disk and exchange information only when they meet; up to $f < k$ of them may crash. Every non-faulty robot must reach the exit.

Measure	Lower bound	Best upper bound	Status
Evacuation time	$1 + 2\pi(f+1)/k$ (Georgiou et al. 2023b); $3 + 2\pi/k - O(k^{-2})$ (Czyzowicz et al. 2014) Additive gap at most 2.	$3 + 2\pi(f+1)/k$ (Leonardos et al. 2023)	• polynomial gap

Results.

- Georgiou et al. (2023b) [lower bound]: evacuation time $1 + 2\pi(f+1)/k$. *Bottleneck:* faulty reports. *Holds when:* Theorem 1: search bound with f crash-faulty robots, hence with Byzantine ones; every point of the boundary must be visited by $f+1$ robots; wireless search bound; it holds face-to-face and for evacuation.
- Czyzowicz et al. (2014) [lower bound]: evacuation time $3 + 2\pi/k - O(k^{-2})$. *Bottleneck:* unknown exit. *Holds when:* fault-free bound of k face-to-face robots (setting with many face-to-face robots in the disk).
- Leonardos et al. (2023) [current best]: evacuation time $3 + 2\pi(f+1)/k$. Introductory and current best (Theorem 5).

Open questions.

Q1. Tight bounds for face-to-face evacuation of the disk with crash faults.

Notes. Checked against the full texts of (Leonardos et al. 2023) and (Georgiou et al. 2023b).

20.2.24 Line, weighted evacuation of two wireless robots

Adds + wireless, + no faults, + equal speeds VERIFIED

When the heavy robot's arrival counts w times the light one's, two wireless robots achieve exactly $1 + \sqrt{2w/(1+w)}$, between 2 and $1 + \sqrt{2}$.

Assumed: domain: the infinite line; number of robots: two robots; goal: weighted: a weighted average of two arrival times; communication: wireless: instant communication at any distance; faults: no faults; speeds: all robots have the same unit speed. **Not assumed:** exits: one exit; randomness: deterministic.

Statement for this setting. Two unit-speed robots start at the origin of the line; the exit is at unknown distance $d \geq 1$. Both know which robot is heavy and the weight $w \geq 1$. The cost is $(t_1 + w\,t_2)/((1+w)d)$ in the worst case, where t_1, t_2 are the times from which the light and the heavy robot stay at the exit.

Measure	Lower bound	Best upper bound	Status
Evacuation time	$1 + \sqrt{2w/(1+w)}$ (Georgiou and Lucier 2023) Exact for every $w \geq 1$.	$1 + \sqrt{2w/(1+w)}$ (Georgiou and Lucier 2023)	• tight

Results.

- Georgiou and Lucier (2023) [lower bound]: evacuation time $1 + \sqrt{2w/(1+w)}$. *Bottleneck:* unknown distance. *Holds when:* Theorem 3.6; identities and w known.
- Georgiou and Lucier (2023) [current best]: evacuation time $1 + \sqrt{2w/(1+w)}$. Introductory and current best (Theorems 3.1, 3.2). Without the identities, robots walking apart at unit speed achieve $1 + 2w/(1+w)$ (Theorem 5.1).

Claude (Anthropic) was used in cataloguing the problems.

SEARCH

Open questions.

Q1. Weighted evacuation with more than two robots (Georgiou and Lucier 2023).

Notes. For $w = 1$ the cost is the average arrival time, ratio 2; as w grows the ratio tends to $1 + \sqrt{2}$, below the ratio 3 of evacuating both robots. Checked against the full text.

20.2.25 Line, weighted evacuation of two face-to-face robots

Adds face-to-face, + no faults, + equal speeds VERIFIED

Face-to-face, the weighted cost is at most between 6.60258 *and* 6.82843*, decreasing in* w*, well below the ratio* 9 *of evacuating both robots; no face-to-face lower bound is known.*

Assumed: domain: the infinite line; number of robots: two robots; goal: weighted: a weighted average of two arrival times; faults: no faults; speeds: all robots have the same unit speed. **Not assumed:** exits: one exit; communication: face-to-face: robots exchange information only when they meet; randomness: deterministic.

Statement for this setting. As in the wireless setting, but the robots exchange information only when they meet.

Measure	Lower bound	Best upper bound	Status
Evacuation time	$1 + \sqrt{2w/(1+w)}$ (Georgiou and Lucier 2023) Constant gap: only the wireless lower bound is known.	6.60258 to 6.82843 (Georgiou and Lucier 2023)	• polynomial gap

Results.

- Georgiou and Lucier (2023) [lower bound]: evacuation time $1 + \sqrt{2w/(1+w)}$. *Bottleneck:* unknown distance. *Holds when:* Theorem 3.6; wireless bound; it holds face-to-face.
- Georgiou and Lucier (2023) [current best]: evacuation time 6.60258 to 6.82843. Introductory and current best (Theorem 4.1; numerical parameters in Theorems 4.12–4.14): $4 + 2\sqrt{2} \approx 6.82843$ at $w = 1$, decreasing to 6.60258 as w grows. Oblivious to the identities: $(2w + 4\sqrt{w(w+1)})/(w+1) + 3$ with w known, $4w/(w+1) + 5$ with nothing known (Theorem 5.2).

Open questions.

Q1. Lower bounds for weighted evacuation of two face-to-face robots on the line (Georgiou and Lucier 2023).

Notes. The robots meet at a sequence of points away from the origin; a missed meeting tells a robot the direction of the exit. Checked against the full text.

20.2.26 Line, priority evacuation, queen and one wireless servant

Adds + wireless, + no faults, + equal speeds VERIFIED

A queen and one wireless servant bring the queen to the exit with ratio at most $1 + \sqrt{2}$*, against* 3 *for evacuating both; no lower bound is known.*

Assumed: domain: the infinite line; number of robots: two robots; goal: priority: only the queen must reach the exit; communication: wireless: instant communication at any distance; faults: no faults; speeds: all robots have the same unit speed. **Not assumed:** exits: one exit; randomness: deterministic.

Claude (Anthropic) was used in cataloguing the problems.

Measure	Lower bound	Best upper bound	Status
Evacuation time	none known No lower bound beyond the trivial one.	$1+\sqrt{2}$ (Georgiou and Lucier 2023)	• unknown

Results.

- Georgiou and Lucier (2023) [current best]: evacuation time $1+\sqrt{2}$. Introductory and current best (Theorem 6.1, the weighted algorithm as $w \to \infty$); the servant also reaches the exit.

Open questions.

Q1. A lower bound for priority evacuation of a queen and one wireless servant on the line.

Notes. Georgiou and Lucier call this the strict version (the servant must also reach the exit) and note that the problem had been studied only on the circle before. Checked against the full text of (Georgiou and Lucier 2023).

20.2.27 Line, priority evacuation, queen and one face-to-face servant

Adds face-to-face, + no faults, + equal speeds VERIFIED

Face-to-face, the queen reaches the exit with ratio at most $2+\sqrt{5} \approx 4.236$, far below 9; no lower bound is known.

Assumed: domain: the infinite line; number of robots: two robots; goal: priority: only the queen must reach the exit; faults: no faults; speeds: all robots have the same unit speed. **Not assumed:** exits: one exit; communication: face-to-face: robots exchange information only when they meet; randomness: deterministic.

Measure	Lower bound	Best upper bound	Status
Evacuation time	none known No lower bound is known.	$2+\sqrt{5}$ (Georgiou and Lucier 2023)	• unknown

Results.

- Georgiou and Lucier (2023) [current best]: evacuation time $2+\sqrt{5}$. Introductory and current best (Theorem 6.2): the queen walks one way at unit speed, the servant the other way at speed $\sqrt{5}-2$; a servant that finds the exit catches the queen and brings her back. If the servant must also reach the exit, 6.60258 (Theorem 6.1).

Open questions.

Q1. A lower bound for priority evacuation of a queen and one face-to-face servant on the line.

Notes. Checked against the full text of (Georgiou and Lucier 2023).

20.2.28 Line, two face-to-face robots of different speeds

Adds all robots, face-to-face, + no faults VERIFIED

With speeds 1 and $v \le 1$, two face-to-face robots evacuate the line optimally: ratio $(1+3v)/(1-v)$ for $v \le 1/3$ and $9v$ above.

Assumed: domain: the infinite line; number of robots: two robots; faults: no faults. **Not assumed:** goal: every robot must reach the exit; exits: one exit; communication: face-to-face: robots exchange information only when they meet; speeds: robots may have different maximum speeds; randomness: deterministic.

Claude (Anthropic) was used in cataloguing the problems.

SEARCH

Measure	Lower bound	Best upper bound	Status
Evacuation time	*matching* (Bampas et al. 2019b) Exact for every v.	$\frac{1+3v}{1-v}$ $(v \le \frac{1}{3})$, $9v$ $(v \ge \frac{1}{3})$ (Bampas et al. 2019b)	• tight

Results.

- Bampas et al. (2019b) [lower bound]: evacuation time *matching*. *Bottleneck:* unknown distance. *Holds when:* Theorem 3.11.
- Bampas et al. (2019b) [current best]: evacuation time $\frac{1+3v}{1-v}$ $(v \le \frac{1}{3})$, $9v$ $(v \ge \frac{1}{3})$. Introductory and current best (Theorem 3.11); both robots must reach the exit.

Notes. Checked against the full text.

20.2.29 Line, two wireless robots of different speeds

Adds all robots, + wireless, + no faults VERIFIED

With speeds 1 *and* $v \le 1$, *two wireless robots evacuate the line optimally, with a ratio that changes form at* $v = \sqrt{17} - 4$.

Assumed: domain: the infinite line; number of robots: two robots; communication: wireless: instant communication at any distance; faults: no faults. **Not assumed:** goal: every robot must reach the exit; exits: one exit; speeds: robots may have different maximum speeds; randomness: deterministic.

Measure	Lower bound	Best upper bound	Status
Evacuation time	*matching* (Bampas et al. 2019b) Exact for every v.	$\frac{1+3v}{1-v}$ $(v \le \sqrt{17} - 4)$, $\frac{2+v+\sqrt{v^2+8v}}{2}$ above (Bampas et al. 2019b)	• tight

Results.

- Bampas et al. (2019b) [lower bound]: evacuation time *matching*. *Bottleneck:* unknown distance. *Holds when:* Theorem 4.8.
- Bampas et al. (2019b) [current best]: evacuation time $\frac{1+3v}{1-v}$ $(v \le \sqrt{17} - 4)$, $\frac{2+v+\sqrt{v^2+8v}}{2}$ above. Introductory and current best (Theorem 4.8).

Notes. Checked against the full text.

20.2.30 Line, a sender and a receiver

Adds all robots, + sender/receiver, + no faults, + equal speeds VERIFIED

A robot that can only send and one that can only receive wirelessly evacuate the line with ratio exactly $3 + 2\sqrt{2}$.

Assumed: domain: the infinite line; number of robots: two robots; communication: asymmetric: senders and receivers; faults: no faults; speeds: all robots have the same unit speed. **Not assumed:** goal: every robot must reach the exit; exits: one exit; randomness: deterministic.

Measure	Lower bound	Best upper bound	Status
Evacuation time	$3 + 2\sqrt{2}$ (Czyzowicz et al. 2021d) Exactly $3 + 2\sqrt{2}$.	$3 + 2\sqrt{2}$ (Czyzowicz et al. 2021d)	• tight

Claude (Anthropic) was used in cataloguing the problems.

Results.

- Czyzowicz et al. (2021d) [lower bound]: evacuation time $3 + 2\sqrt{2}$. *Bottleneck:* one-way communication. *Holds when:* Theorem 10.
- Czyzowicz et al. (2021d) [current best]: evacuation time $3 + 2\sqrt{2}$. Introductory and current best (Theorem 1).

Open questions.

Q1. Close the gaps for one sender and many receivers ($2 + \sqrt{5}$ against 5) and for many senders and one receiver (3 against about 5.681) (Czyzowicz et al. 2021d).

Notes. If an extra vehicle (a bike) can carry a robot faster, sender/receiver evacuation on the line has piecewise bounds in the bike's speed (Jawhar and Kranakis 2025a). Checked against the full text of (Czyzowicz et al. 2021d).

20.2.31 Line, any number of wireless robots

Adds + no faults, + equal speeds VERIFIED

With wireless communication, two or more robots evacuate the line with competitive ratio exactly 3.

Assumed: domain: the infinite line; communication: wireless: instant communication at any distance; faults: no faults; speeds: all robots have the same unit speed. **Not assumed:** number of robots: any number k of robots; goal: every robot must reach the exit; exits: one exit; randomness: deterministic.

Measure	Lower bound	Best upper bound	Status
Evacuation time	3 (Czyzowicz et al. 2021d); 3 (Bampas et al. 2019b) Exactly 3.	3 (Bampas et al. 2019b)	• tight

Results.

- Czyzowicz et al. (2021d) [lower bound]: evacuation time 3. *Bottleneck:* unknown distance. *Holds when:* Section 4 (a remark); consider the first time any robot reaches d or $-d$ and put the exit at the other point; any number of robots and any communication model.
- Bampas et al. (2019b) [lower bound]: evacuation time 3. *Bottleneck:* unknown distance. *Holds when:* Theorem 4.8 with equal speeds ($v = 1$); two robots.
- Baeza-Yates and Schott (1995) [introductory]: evacuation time 3. Section 2.2: with instantaneous communication, two robots search in opposite directions; "if both robots must reach the goal, the distance increases to $4D$ and the time to 3 units" (their D is d here). Stated without a lower bound.
- Bampas et al. (2019b) [current best]: evacuation time 3. Algorithm B^* with $v = 1$ (Corollary 4.3): two robots move in opposite directions at unit speed, the finder calls the other, which needs $2d$ more. Further robots can move with one of the two (the catalogue's own observation, see notes). (Czyzowicz et al. 2021d) attribute the bound to (Chrobak et al. 2015), which treats only robots that communicate when they meet; it is stated earlier by (Baeza-Yates and Schott 1995).

Notes. Checked against the full texts of (Baeza-Yates and Schott 1995), (Bampas et al. 2019b), (Czyzowicz et al. 2021d) and (Chrobak et al. 2015). Own observation of this catalogue, not taken from a paper: with $k > 2$ robots, let the extra robots move together with one of the two robots that walk apart at unit speed. Whichever of the two finds the exit, at distance d and time d, every robot is then within $2d$ of it and all arrive by time $3d$; so ratio 3 holds for every $k \geq 2$.

Claude (Anthropic) was used in cataloguing the problems.

SEARCH

20.2.32 Line, $2f + 1$ wireless robots, f crash faults

Adds + crash faults, + equal speeds VERIFIED

With a near majority of faulty robots, evacuation on the line takes ratio about 7.437 for $(k, f) = (3, 1)$, tending to $4 + 2\sqrt{2}$; the lower bounds come from search.

Assumed: domain: the infinite line; communication: wireless: instant communication at any distance; faults: up to f crash-faulty robots (fail to report the target); speeds: all robots have the same unit speed. **Not assumed:** number of robots: any number k of robots; goal: every robot must reach the exit; exits: one exit; randomness: deterministic.

Statement for this setting. $k = 2f + 1$ robots, of which up to f may fail to report the exit; every non-faulty robot must reach it.

Measure	Lower bound	Best upper bound	Status
Evacuation time	≈ 5.233 (Kupavskii and Welzl 2021)	≈ 7.437 (Czyzowicz et al. 2021e)	• polynomial gap
	Constant gap against the search lower bounds.		

Results.

- Kupavskii and Welzl (2021) [lower bound]: evacuation time ≈ 5.233. *Bottleneck:* faulty reports. *Holds when:* $(k, f) = (3, 1)$; inherited from search with crash faults.
- Czyzowicz et al. (2021e) [current best]: evacuation time ≈ 7.437. Introductory and current best for $(3, 1)$; 7.254 for $(5, 2)$ and $(7, 3)$, 7.147 for $(9, 4)$, tending to $4 + 2\sqrt{2}$ (Theorems 2.2, 3.1, 3.2).

Open questions.

Q1. Tight bounds for evacuation with $k = 2f + 1$ crash or Byzantine robots (Czyzowicz et al. 2021e).

Notes. When every attempt to turn fails with known probability $\beta < 1/2$ (the paper's p), two wireless robots achieve ratio $3 + 4\sqrt{\beta(1-\beta)} + \varepsilon$ (Georgiou et al. 2024). Checked against the full text of (Czyzowicz et al. 2021e).

20.2.33 Line, any number of face-to-face robots

Adds any k, face-to-face, + no faults, + equal speeds VERIFIED

On the line, face-to-face robots evacuate with competitive ratio exactly 9: extra robots do not help.

Assumed: domain: the infinite line; faults: no faults; speeds: all robots have the same unit speed. **Not assumed:** number of robots: any number k of robots; goal: every robot must reach the exit; exits: one exit; communication: face-to-face: robots exchange information only when they meet; randomness: deterministic.

Measure	Lower bound	Best upper bound	Status
Evacuation time	9 (Bampas et al. 2019b); 9 (Chrobak et al. 2015) Exactly 9.	9 (Chrobak et al. 2015)	• tight

Results.

- Bampas et al. (2019b) [lower bound]: evacuation time 9. *Bottleneck:* unknown distance. *Holds when:* any number of robots of maximum speed at most 1; full proof of $9d - o(d)$.
- Chrobak et al. (2015) [lower bound]: evacuation time 9. *Bottleneck:* unknown distance. *Holds when:* Theorem 3 and Corollary 1; any number $k \geq 2$ of unit-speed robots that communicate only when they meet; the argument

Claude (Anthropic) was used in cataloguing the problems.

covers algorithms whose explored segment does not grow at rate exactly $1/3$; (Bampas et al. 2019b) call it partial and complete it (Corollary 3.4).

- Baeza-Yates and Schott (1995) [introductory]: evacuation time 9. Section 2.2: two robots that cannot communicate wait half the time (equivalently, search at speed $1/3$); the finder goes back, meets the other and both return to the exit, in time $9d$, "the same time as one robot". No lower bound is claimed for two robots.
- Chrobak et al. (2015) [current best]: evacuation time 9. Each robot runs the doubling strategy, the finder waiting at the exit, or two robots search in opposite directions at speed $1/3$ and the finder catches the other (Section 2.2); with the lower bound of Theorem 3 this shows that $9d$ is optimal for any number of robots.

Notes. For two face-to-face robots the energy of group search can be traded against time; with maximum speed v and time at most $9d/v$ the energy is about $8.42588\,v^2d$ (the paper writes b for v; Theorem 16) (Czyzowicz et al. 2019a). Checked against the full texts of (Baeza-Yates and Schott 1995), (Chrobak et al. 2015) and (Bampas et al. 2019b).

20.2.34 Two face-to-face robots, equilateral triangle

Adds face-to-face, + no faults, + equal speeds VERIFIED

From the centroid of a unit equilateral triangle: between $1 + 2/\sqrt{3}$ and 2.3367.

Assumed: domain: unit equilateral triangle, start at the centroid; number of robots: two robots; faults: no faults; speeds: all robots have the same unit speed. **Not assumed:** goal: every robot must reach the exit; exits: one exit; communication: face-to-face: robots exchange information only when they meet; randomness: deterministic.

Measure	Lower bound	Best upper bound	Status
Evacuation time	$1 + 2/\sqrt{3}$ (Chuangpishit et al. 2020b) Constant gap.	2.3367 (Chuangpishit et al. 2020b)	• polynomial gap

Results.

- Chuangpishit et al. (2020b) [lower bound]: evacuation time $1 + 2/\sqrt{3}$. *Bottleneck:* unknown exit. *Holds when:* two robots, face-to-face, start at the centroid.
- Chuangpishit et al. (2020b) [current best]: evacuation time 2.3367. For k robots: $\sqrt{3} + 3/k$ against $\sqrt{3}$.

Notes. Wireless evacuation of triangles and squares was introduced by (Czyzowicz et al. 2015c).

20.2.35 Triangle, two wireless robots at the centroid

Adds + wireless, + no faults, + equal speeds VERIFIED

Two wireless robots at the centroid of the unit equilateral triangle evacuate in exactly $3/2 + \sqrt{3}/6 \approx 1.789$.

Assumed: domain: unit equilateral triangle, start at the centroid; number of robots: two robots; communication: wireless: instant communication at any distance; faults: no faults; speeds: all robots have the same unit speed. **Not assumed:** goal: every robot must reach the exit; exits: one exit; randomness: deterministic.

Measure	Lower bound	Best upper bound	Status
Evacuation time	$3/2 + \sqrt{3}/6$ (Czyzowicz et al. 2015c) Exact.	$3/2 + \sqrt{3}/6$ (Czyzowicz et al. 2015c)	• tight

Claude (Anthropic) was used in cataloguing the problems.

SEARCH

Results.

- Czyzowicz et al. (2015c) [lower bound]: evacuation time $3/2 + \sqrt{3}/6$. *Bottleneck:* unknown exit. *Holds when:* Theorem 2 with the start at the centroid, at distance $\sqrt{3}/6$ from the midpoints of the sides.
- Czyzowicz et al. (2015c) [current best]: evacuation time $3/2 + \sqrt{3}/6$. Introductory and current best (Theorem 2: from an interior point at distance y from the nearest midpoint of a side, $3/2 + y$ is tight).

Notes. Three robots at the centroid do better than two (Theorem 4). Checked against the full text.

20.2.36 Triangle, any number of face-to-face robots

Adds any k, face-to-face, + no faults, + equal speeds VERIFIED

k face-to-face robots at the centroid of an equilateral unit triangle evacuate in at most $\sqrt{3} + 3/k$, against $\sqrt{3}$.

Assumed: domain: unit equilateral triangle, start at the centroid; faults: no faults; speeds: all robots have the same unit speed. **Not assumed:** number of robots: any number k of robots; goal: every robot must reach the exit; exits: one exit; communication: face-to-face: robots exchange information only when they meet; randomness: deterministic.

Measure	Lower bound	Best upper bound	Status
Evacuation time	$\sqrt{3}$ (Chuangpishit et al. 2020b) The gap $3/k$ vanishes as k grows.	$\sqrt{3} + 3/k$ (Chuangpishit et al. 2020b)	• near-optimal

Results.

- Chuangpishit et al. (2020b) [lower bound]: evacuation time $\sqrt{3}$. *Bottleneck:* unknown exit. *Holds when:* Theorem 3.1; every k.
- Chuangpishit et al. (2020b) [current best]: evacuation time $\sqrt{3} + 3/k$. Introductory and current best (Observation 3.2); for $k = 3, 4, 5, 6$ about 2.089, 1.982, 1.876, 1.826.

Notes. Checked against the full text.

20.2.37 Triangle, any number of wireless robots at the centroid

Adds any k, + wireless, + no faults, + equal speeds VERIFIED

k wireless robots at the centroid evacuate in $1 + \sqrt{3}/3 + 3/k$, and no number of robots beats $1 + \sqrt{3}/3$, which 8 robots attain.

Assumed: domain: unit equilateral triangle, start at the centroid; communication: wireless: instant communication at any distance; faults: no faults; speeds: all robots have the same unit speed. **Not assumed:** number of robots: any number k of robots; goal: every robot must reach the exit; exits: one exit; randomness: deterministic.

Measure	Lower bound	Best upper bound	Status
Evacuation time	$1 + \sqrt{3}/3$ (Czyzowicz et al. 2015c) Exact for $k \geq 8$; for $3 \leq k \leq 7$ the optimum is not proved.	$1 + \sqrt{3}/3$ ($k \geq 8$) (Bagheri et al. 2019)	• tight

Results.

- Czyzowicz et al. (2015c) [lower bound]: evacuation time $1 + \sqrt{3}/3$. *Bottleneck:* unknown exit. *Holds when:* Theorem 5; every k.
- Czyzowicz et al. (2015c) [introductory]: evacuation time $1 + \sqrt{3}/3 + 3/k$. Theorem 5.

Claude (Anthropic) was used in cataloguing the problems.

- Bagheri et al. (2019) [current best]: evacuation time $1+\sqrt{3}/3$ ($k \geq 8$). Theorem 5.2: $6+2\lceil 1/r-1\rceil$ robots of communication range $r<1$ suffice, hence 8 wireless robots; they report that 6 suffice with range 1, citing an extended version of (Czyzowicz et al. 2015c).

Open questions.

Q1. The optimal evacuation time of $k=3,\dots,7$ wireless robots from the centroid of the triangle.

Notes. For range-limited robots, $\Omega(1/r)$ robots are needed to reach $1+\sqrt{3}/3$ (Theorem 5.1 of Bagheri et al. 2019); their Table 1 gives about 1.667 and 1.6105 for three and four robots with range 1, citing the extended version. Checked against the full texts of (Czyzowicz et al. 2015c) and (Bagheri et al. 2019).

20.2.38 Two face-to-face robots, square

Adds + two robots, + no faults, + equal speeds VERIFIED

From the centroid of a unit square: between 3.118 *and* 3.4645.

Assumed: domain: unit square, start at the centroid; number of robots: two robots; faults: no faults; speeds: all robots have the same unit speed. **Not assumed:** goal: every robot must reach the exit; exits: one exit; communication: face-to-face: robots exchange information only when they meet; randomness: deterministic.

Measure	Lower bound	Best upper bound	Status
Evacuation time	3.118 (Chuangpishit et al. 2020b) Constant gap.	3.4645 (Chuangpishit et al. 2020b)	• polynomial gap

Results.

- Chuangpishit et al. (2020b) [lower bound]: evacuation time 3.118. *Bottleneck:* unknown exit. *Holds when:* two robots, face-to-face.
- Chuangpishit et al. (2020b) [current best]: evacuation time 3.4645.

Notes. The lower bound is $1+3\sqrt{2}/2\approx 3.121$ in Theorem 4.1 (the abstract states 3.118). Three and four robots evacuate in about 3.178 and 2.664 (Chuangpishit et al. 2020b).

20.2.39 Square, any number of wireless robots at the centre

Adds any k, + wireless, + no faults, + equal speeds VERIFIED

k wireless robots at the centre of the unit square evacuate in $3\sqrt{2}/2+4/k$, against the lower bound $3\sqrt{2}/2$.

Assumed: domain: unit square, start at the centroid; communication: wireless: instant communication at any distance; faults: no faults; speeds: all robots have the same unit speed. **Not assumed:** number of robots: any number k of robots; goal: every robot must reach the exit; exits: one exit; randomness: deterministic.

Measure	Lower bound	Best upper bound	Status
Evacuation time	$3\sqrt{2}/2$ (Czyzowicz et al. 2015c) The gap $4/k$ vanishes as k grows.	$3\sqrt{2}/2+4/k$ (Czyzowicz et al. 2015c)	• near-optimal

Results.

- Czyzowicz et al. (2015c) [lower bound]: evacuation time $3\sqrt{2}/2$. *Bottleneck:* unknown exit. *Holds when:* Theorem 9; every k.

Claude (Anthropic) was used in cataloguing the problems.

SEARCH

- Czyzowicz et al. (2015c) [current best]: evacuation time $3\sqrt{2}/2 + 4/k$. Introductory and current best (Theorem 9): the robots split the boundary into k pieces of length $4/k$ and the finder calls the others.

Open questions.

Q1. The optimal evacuation time of k wireless robots from the centre of the square for fixed k.

Notes. Checked against the full text.

20.2.40 Triangle, two wireless robots starting on the boundary

Adds + two robots, + wireless, + no faults, + equal speeds VERIFIED

Two wireless robots starting on the boundary of a non-obtuse triangle evacuate in half the perimeter if they may choose their start, which is optimal.

Assumed: domain: triangle, start on the boundary; number of robots: two robots; communication: wireless: instant communication at any distance; faults: no faults; speeds: all robots have the same unit speed. **Not assumed:** goal: every robot must reach the exit; exits: one exit; randomness: deterministic.

Measure	Lower bound	Best upper bound	Status
Evacuation time	$(a+b+c)/2$ (Georgiou and Jang 2026) When the algorithm chooses the start.	$(a+b+c)/2$ (Georgiou and Jang 2026)	• tight

Results.

- Georgiou and Jang (2026) [lower bound]: evacuation time $(a+b+c)/2$. *Bottleneck:* unknown exit. *Holds when:* Theorem 1; non-obtuse triangle with sides $a \geq b \geq c$; the algorithm chooses the starting edge and point.
- Czyzowicz et al. (2015c) [introductory]: evacuation time $3/2 + y$. Equilateral unit triangle, start on the boundary at distance y from the nearest midpoint of a side; tight (Theorem 1), and $k \geq 3$ robots starting there do no better (Theorem 3).
- Georgiou and Jang (2026) [current best]: evacuation time $(a+b+c)/2$. Non-obtuse triangles (Theorem 1); separate tight or near-tight bounds when the adversary chooses the edge or the point.

Open questions.

Q1. Face-to-face evacuation of general triangles, and wireless evacuation of obtuse triangles (Georgiou and Jang 2026).

Notes. With $k \geq 3$ robots starting at the same point of the boundary the bound $3/2 + y$ is still tight (Theorem 3 of Czyzowicz et al. 2015c); see the setting with any number of robots. Checked against the full texts of (Czyzowicz et al. 2015c) and (Georgiou and Jang 2026).

20.2.41 Triangle, any number of wireless robots starting on the boundary

Adds any k, + wireless, + no faults, + equal speeds VERIFIED

From a point of the boundary at distance y from the nearest midpoint of a side, any number $k \geq 2$ of wireless robots needs exactly $3/2 + y$: extra robots do not help.

Assumed: domain: triangle, start on the boundary; communication: wireless: instant communication at any distance; faults: no faults; speeds: all robots have the same unit speed. **Not assumed:** number of robots: any number k of robots; goal: every robot must reach the exit; exits: one exit; randomness: deterministic.

Claude (Anthropic) was used in cataloguing the problems.

Measure	Lower bound	Best upper bound	Status
Evacuation time	$3/2 + y$ (Czyzowicz et al. 2015c) Exact for every $k \geq 2$.	$3/2 + y$ (Czyzowicz et al. 2015c)	• tight

Results.

- Czyzowicz et al. (2015c) [lower bound]: evacuation time $3/2+y$. *Bottleneck:* unknown exit. *Holds when:* Theorems 1 and 3; unit equilateral triangle, given start on the boundary; the argument does not depend on the number of robots.
- Czyzowicz et al. (2015c) [current best]: evacuation time $3/2 + y$. Introductory and current best: two robots walk the boundary in opposite directions (Theorem 1); more robots do no better (Theorem 3).

Notes. Checked against the full text.

20.2.42 m rays, $k = m$ wireless robots at the junction

Adds + rays, + wireless, + no faults, + equal speeds VERIFIED

With one wireless robot per ray, starting at the junction, the competitive ratio is exactly 3.

Assumed: domain: m rays from a junction (a star); communication: wireless: instant communication at any distance; faults: no faults; speeds: all robots have the same unit speed. **Not assumed:** number of robots: any number k of robots; goal: every robot must reach the exit; exits: one exit; randomness: deterministic.

Measure	Lower bound	Best upper bound	Status
Evacuation time	3 (Brandt et al. 2020b) Exactly 3.	3 (Brandt et al. 2020b)	• tight

Results.

- Brandt et al. (2020b) [lower bound]: evacuation time 3. *Bottleneck:* unknown exit. *Holds when:* Observation 2; every $2 \leq k \leq m$.
- Brandt et al. (2020b) [current best]: evacuation time 3. Introductory and current best (Observation 1): each robot explores its own ray.

Open questions.

Q1. Starting off the junction with $k = m$: close $2 + \sqrt{7/3} \approx 3.528$ against 4 (Brandt et al. 2020b).

Q2. Fewer robots than rays ($1 < k < m$): the bounds $3 + 2\lfloor (m-1)/k \rfloor$ and $1 + 2\frac{m-1}{k}(1 + \frac{k}{m-1})^{1+(m-1)/k}$ differ (Brandt et al. 2020b).

Notes. Checked against the full text (conference version).

20.2.43 Square, two wireless robots starting on the boundary

Adds + square, boundary start, + two robots, + wireless, + no faults, + equal speeds VERIFIED

From a point of the boundary at distance x from the nearest corner, two wireless robots need exactly $1 + x + \sqrt{1 + (1-2x)^2}$; the best start is $x = 1/2 - 1/\sqrt{12}$.

Claude (Anthropic) was used in cataloguing the problems.

SEARCH

Assumed: domain: unit square, start on the boundary; number of robots: two robots; communication: wireless: instant communication at any distance; faults: no faults; speeds: all robots have the same unit speed. **Not assumed:** goal: every robot must reach the exit; exits: one exit; randomness: deterministic.

Measure	Lower bound	Best upper bound	Status
Evacuation time	$1 + x + \sqrt{1 + (1 - 2x)^2}$ (Czyzowicz et al. 2015c) Exact for every start.	$1 + x + \sqrt{1 + (1 - 2x)^2}$ (Czyzowicz et al. 2015c)	• tight

Results.

- Czyzowicz et al. (2015c) [lower bound]: evacuation time $1 + x + \sqrt{1 + (1 - 2x)^2}$. *Bottleneck:* unknown exit. *Holds when:* Theorem 6; given start on the boundary, $x \leq 1/2$.
- Czyzowicz et al. (2015c) [current best]: evacuation time $1 + x + \sqrt{1 + (1 - 2x)^2}$. Introductory and current best (Theorem 6): the robots walk the boundary in opposite directions and the finder calls the other.

Open questions.

Q1. Optimal evacuation of the square by $k \geq 3$ wireless robots from a point of the boundary other than a corner.

Notes. From a corner, any number of robots needs $1 + \sqrt{2}$, the two-robot time (Theorem 7); from the midpoint of a side three robots beat two (Theorem 8) (Czyzowicz et al. 2015c). Checked against the full text.

20.3 Chauffeuring a crashed robot

An active robot must find a crashed robot inside a disk and carry it, at reduced speed, to one of e exits on the boundary.

First posed in 2019 (Pattanayak et al. 2019a). ALGOSENSORS 2019.

Problem statement and measures

A unit disk has $e \geq 1$ exits at unknown points of its boundary. An active robot starts at the centre and moves at speed v; a crashed robot lies at an unknown point inside the disk. The active robot must find it and carry it, at speed $v' < v$, to an exit; it learns the exits' positions once it reaches the boundary. **Ratio** is the competitive ratio of the **evacuation time** (the time until every robot has reached an exit) against the optimum.

Measures: **Competitive ratio**, evacuation time divided by the optimal time.

A first look

A rescue robot in a round room must carry a broken robot to a door; carrying it slows the rescuer down. In chauffeuring the room is a unit **disk**, the points at distance at most 1 from its centre; the edge of the room is the **boundary**, the circle of radius 1; and the doors are one or more **exits** at unknown points of the boundary. The rescuer is the **active robot**: it starts at the centre and moves at speed v. The broken robot is a **crashed robot**, one that has stopped moving and communicating; it lies somewhere inside the disk. The active robot must **carry** the crashed robot to an exit, that is, move together with it, at the

Claude (Anthropic) was used in cataloguing the problems.



(a) Start: two robots at the centre, exit unknown

(b) Final: carried to the exit at speed v' = v/2

Figure 20.3: Chauffeuring: an active robot must find a crashed robot inside a unit disk and carry it, at a lower speed $v' < v$, to an exit on the boundary (Pattanayak et al. 2019a). The animation runs the simplest algorithm in the setting of the arXiv version: two robots walk together from the centre to the boundary and along it; when one crashes (at time w), the other picks it up on the spot and carries it along the same route at speed $v' = v/2$ (red) until it finds the exit, within time $w + (1 + 2\pi - w)/v'$.

slower speed $v' < v$. Performance is the **competitive ratio**: the **evacuation time** (the time until every robot has reached an exit) divided by the best time possible if everything were known. The catalogue's bounds for this formulation (competitive ratio at most $4 + 2\pi$ for any number of exits) were supplied by the catalogue owner and have not been checked against a published text; the ideas below come from the setting of the arXiv version of (Pattanayak et al. 2019a), where two robots start together at the centre, one exit is hidden on the boundary, and one robot may crash at some time w.

1. Algorithm: Move together, so nobody is far from the crash. (Pattanayak et al. 2019a) (One exit, p. 543) Take speed $v = 1$; carrying a robot one unit of distance takes time $1/v'$.

1. Both robots walk together from the centre to the boundary.
2. They walk together along the boundary until they find the exit, then leave.
3. If one crashes on the way, the other picks it up on the spot and continues the same route, carrying it at speed v'.

Why it works and what it costs. Without a crash the robots reach the boundary in time 1 and walk at most the whole boundary, of length 2π, so they leave by time $1 + 2\pi$. If one crashes at time w, the rest of the route, of length at most $1 + 2\pi - w$, is walked at the slow speed, so the time is at most $w + (1 + 2\pi - w)/v'$ (Pattanayak et al. 2019a). It does not use the second robot to search elsewhere; (Pattanayak et al. 2019a) give better algorithms in which the robots search in opposite directions, as in two-robot evacuation.

Why it matters. It is the baseline every chauffeuring algorithm must beat, and it shows the trade-off: staying together makes a crash cheap but makes the search slow.

2. Lower bound: Explore the whole boundary and carry from the centre. (Pattanayak et al. 2019a) (One exit, p. 543) Let one robot crash immediately, at the centre. The active robot alone must search the boundary, and the **adversary**, an opponent who chooses the exit position, keeps the exit at a point the active robot has not yet visited, so it must walk the whole boundary, of length 2π. In addition, the crashed robot has to be carried at least the radius, from the centre to the boundary, which takes time at least $1/v'$.

Why it holds. Carrying from the centre to the boundary and walking around the boundary are separate stretches of time, so every algorithm needs at least $2\pi + 1/v'$ in this case (Pattanayak et al. 2019a) (Theorem 1 there, which writes the slowdown as a factor). Compared with the fault-free optimum

Claude (Anthropic) was used in cataloguing the problems.

SEARCH

$1 + 2\pi/3 + \sqrt{3} \approx 4.826$ (Czyzowicz et al. 2014), a crash can cost a lot.

Why it matters. It isolates the two unavoidable costs, searching alone and carrying slowly, that every chauffeuring bound balances.

What the community started looking at

Chauffeuring asked how the cost of evacuation changes when a robot **must be carried** rather than merely informed, and how the number of exits affects the competitive ratio (Pattanayak et al. 2019a).

Relations to other problems

- **Variant of Evacuation.** Evacuation where one robot has crashed inside the disk: the active robot must first find it, then carry it, more slowly, to an exit.

Assumptions specific to this problem

Exits

Pattanayak, Ramesh and Mandal introduced chauffeuring at ALGOSENSORS 2019: a crashed robot "stops moving and sending messages altogether", and instead of abandoning it the non-faulty robot carries it to the exit at a reduced speed (Pattanayak et al. 2019a). That paper has two robots, wireless communication and a single exit on the boundary of the disk (Pattanayak et al. 2019a). Two exits at unknown boundary points come from the two-exit evacuation problem of Pattanayak et al. (Pattanayak et al. 2018). The catalogue's statement with $e \geq 1$ exits, and its bounds for one, two and any number of exits, were supplied by the catalogue owner; they have not been checked against a published text.

Level	Value	Formal definition
0	***e* exits**	$e \geq 1$ exits at unknown boundary points.
1	**one exit**	Exactly one exit.
1	**two exits**	Exactly two exits.

Case tree

The root assumes nothing; going down, each edge adds the assumptions written in grey (+) or leaves an axis unassumed. Bold entries are catalogued settings with their status per measure (Competitive ratio) and page; italic entries are branch points, named after the axis on which their branches differ. Axis order: Exits → Communication → Faults → Speeds → Randomness.



What is known

The best upper bound against the best lower bound in each catalogued setting, in tree order. The settings follow in detail.

Claude (Anthropic) was used in cataloguing the problems.

Setting	Competitive ratio	p.
e exits STANDARD SETTING	● unknown $4 + 2\pi$ vs —	543
One exit	● unknown 5 vs —	543
Two exits	● unknown 6.6 vs —	543

20.3.1 *e* exits

the root: no assumptions STANDARD SETTING VERIFIED

A constant competitive ratio, at most $4 + 2\pi \approx 10.28$*, for any number of exits.*

Assumed: nothing. **Not assumed:** exits: $e \geq 1$ exits; communication: face-to-face: robots exchange information only when they meet; faults: up to f Byzantine robots (may lie or stay silent); speeds: robots may have different maximum speeds; randomness: deterministic.

Measure	Lower bound	Best upper bound	Status
Competitive ratio	none known No lower bound recorded.	$4 + 2\pi$ (Pattanayak et al. 2019a)	● unknown

Results.

- Pattanayak et al. (2019a) [current best]: competitive ratio $4 + 2\pi$. Introductory and current best.

Notes. Results supplied by the catalogue owner (a co-author).

20.3.2 One exit

Adds + one exit **to** *e* exits (p. 543) VERIFIED

Competitive ratio at most 5 with one exit.

Assumed: exits: one exit. **Not assumed:** communication: face-to-face: robots exchange information only when they meet; faults: up to f Byzantine robots (may lie or stay silent); speeds: robots may have different maximum speeds; randomness: deterministic.

Measure	Lower bound	Best upper bound	Status
Competitive ratio	none known No lower bound recorded.	5 (Pattanayak et al. 2019a)	● unknown

Results.

- Pattanayak et al. (2019a) [current best]: competitive ratio 5.

20.3.3 Two exits

Adds + two exits **to** *e* exits (p. 543) VERIFIED

Competitive ratio at most 6.6 with two exits.

Claude (Anthropic) was used in cataloguing the problems.

SEARCH

Assumed: exits: two exits. **Not assumed:** communication: face-to-face: robots exchange information only when they meet; faults: up to f Byzantine robots (may lie or stay silent); speeds: robots may have different maximum speeds; randomness: deterministic.

Measure	Lower bound	Best upper bound	Status
Competitive ratio	none known No lower bound recorded.	6.6 (Pattanayak et al. 2019a)	• unknown

Results.

- Pattanayak et al. (2019a) [current best]: competitive ratio 6.6.

20.4 Search for a moving target

A target starts at unknown distance d from the origin of a line (or of m rays) and moves at constant speed u; how fast can robots capture it, depending on what they know about d and u?

First posed in 2003 (Alpern and Gal 2003). An early treatment, as reported by (Coleman et al. 2022): Alpern and Gal give an optimal algorithm for one robot and a target moving away from the origin at a known speed. Coleman et al. write that the case of a moving target "appears to have been first considered by McCabe" (1974), for a target performing a random walk on the integers; that paper is not in the bibliography.

Problem statement and measures

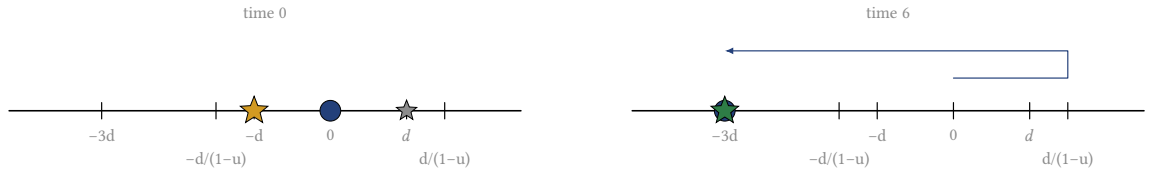
Robots of unit speed start at the origin of the line (or of m rays). The target starts at distance $d \geq 1$ from the origin, on a side (ray) unknown to the robots, and moves along it at constant speed u, either away from the origin ($u < 1$, since a faster target can never be caught) or toward it ($u \geq 0$); the robots know which. The target cannot change its speed or direction. Depending on the setting the robots know d, u, both or neither. The target is **captured** when every robot stands on it. **Time** is the competitive ratio: the supremum, over the target's initial positions and speeds, of the capture time divided by the time the robots would need if they knew the target's initial position.

Measures: **Competitive ratio**, worst-case capture time divided by the capture time of robots that know the target's initial position.

A first look

A dog has run off along a straight road and keeps trotting away at a steady pace, slower than its owner, who does not know which way it went. In this problem the owner is a **robot**, a point that moves at speed 1 and sees the **target** (the dog) only when it reaches it. The robot starts at the **origin**, the point 0 of an infinite line. The target starts at distance $d \geq 1$ from the origin, on a side the robot does not know, and moves at constant speed u along its side. It moves either away from the origin, with $u < 1$ (a faster target could never be caught), or toward it; the robot knows which. On m **rays** (half-lines that share the origin) the target moves along one of them. The target is **captured** when the robot stands on it. We measure the **competitive ratio**: the worst case, over all starting positions and speeds of the target, of the capture time divided by the time a robot that knew where the target started would need. In the standard version the target moves away and the robot knows u but not d. The other versions change what the robot knows, or use two robots, and then both must reach the target.

Claude (Anthropic) was used in cataloguing the problems.



(a) Start: d = 1, u = 1/3; the robot does not know the side (grey: the other possibility)

(b) Final: captured at time 6 = 4 × 1.5; ratio (3−u)/(1−u) = 4

Figure 20.4: Search for a target moving away, by a robot that knows the distance d and the speed u but not the side, Algorithm 1 of (Coleman et al. 2022): walk for time $d/(1-u)$ to one side, where a target on that side would be caught, and if it is not there turn and walk until reaching it. Here $d = 1$ and $u = 1/3$. The target (amber star) starts at $-d$; the grey star shows where it would be on the other side. The robot turns at time 1.5, when it is $2d/(1-u) = 3$ from the target, closes the gap at speed $1-u$ and captures it at time 6, four times the 1.5 that a robot knowing the side would need: the ratio is $(3-u)/(1-u) = 4$, which is optimal. The trace above the line shows the robot's two trips.

1. Algorithm: Go far enough on one side, then turn. (Coleman et al. 2022) (One robot, line, target moving away, d and u known, p. 559) Suppose the target moves away and the robot knows both d and u; only the side is unknown.

1. Walk to one side for time $d/(1-u)$. If the target is on this side, the robot reaches it exactly then, because by that time the target has moved from d to $d + u \cdot d/(1-u) = d/(1-u)$.
2. If the target is not there, it is on the other side: turn and walk until reaching it.

Why it works and what it costs. When the robot turns, it stands at $d/(1-u)$ on one side and the target at $d/(1-u)$ on the other, so they are $2d/(1-u)$ apart. The robot gains on the target at speed $1-u$, which takes another $2d/(1-u)^2$. A robot that knew the side would need $d/(1-u)$, so the ratio is $1 + 2/(1-u) = (3-u)/(1-u)$ (Coleman et al. 2022). No strategy does better. To catch the target the robot must visit $d/(1-u)$ or $-d/(1-u)$, and the **adversary**, an opponent who knows the strategy, puts the target on the side that the robot reaches second. For a target that does not move ($u = 0$) the ratio is 3.

Why it matters. It is the baseline: every other version is measured by how much it costs not to know d, u or both.

2. Algorithm: Zigzag with trips that grow faster. (Alpern and Gal 2003) (One robot, line, target moving away, speed known, p. 558) Now the robot knows u but not d, the standard version. It zigzags: it walks out to distance 1 on one side and back to the origin, then out to distance a on the other side and back, then a^2 on the first side, and so on, with the growth factor $a = 2(1+u)/(1-u)$.

Why it works and what it costs. A target that moves away gains ground during every trip, so the trips must grow faster than for a target that stays put. With this growth factor the ratio is $(3+u)^2/(1-u)^2$, and no strategy does better. This is the result of Alpern and Gal, as reported by (Coleman et al. 2022) (Theorem 6, Algorithm 3); the book was not read for this atlas, and the setting stays a draft. For a target that does not move ($u = 0$) the factor is 2: this is the doubling strategy of linear search, with ratio 9.

Why it matters. It shows how the doubling idea of linear search adapts to a target that runs away, and how quickly the price grows as u approaches 1.

3. Lower bound: Waiting is best when nothing is known. (Coleman et al. 2022) (One robot, line, target moving toward the origin, nothing known, p. 563) Suppose the target moves toward the origin and the robot knows neither d nor u. The robot could simply wait at the origin. Any other strategy leaves the origin at some time, say it waits for time t and then walks a distance d' to the right. The adversary puts the target at $-d$, with speed $u = d/(t+d')$, so that it reaches the origin exactly when the robot is at d'.

Claude (Anthropic) was used in cataloguing the problems.

Why it holds. The two are then d' apart and approach each other at speed $1+u$, so they meet after time $d/u + d'/(1+u)$, which is more than the d/u it takes the target to reach the waiting robot. A robot that knew where the target started would walk to meet it, in time $d/(1+u)$. The ratio is therefore at least $(d/u)/(d/(1+u)) = 1 + 1/u$, and waiting achieves exactly that (Coleman et al. 2022). It grows without bound for slow targets. Knowing d helps a lot: going for time d on one side and then turning gives ratio 3 for every speed.

Why it matters. It shows that a moving target can make searching useless: what the robot knows decides between a constant ratio and an unbounded one.

What the community started looking at

A target that moves away from the origin at a speed the robot knows was treated by Alpern and Gal (Alpern and Gal 2003), and a target performing a random walk on the integers by McCabe (1974), both as reported by (Coleman et al. 2022). Coleman, Kranakis, Krizanc and Morales-Ponce posed the problem for one robot in all eight combinations of the target's direction (away or toward) and of what the robot knows about d and u, with tight bounds whenever the target moves toward the origin or its speed is known (Coleman et al. 2022). With Ivanov they settled the exponent for a target moving away at an unknown speed when d is known: the ratio is $(1-u)^{-4+o(1)}$ (Coleman et al. 2026a). Jawhar and Kranakis extended it to two robots that communicate face-to-face (Jawhar and Kranakis 2026b) or as a sender and a receiver (Jawhar and Kranakis 2025b), and to m rays (Jawhar and Kranakis 2026a).

Further reading

Papers on this problem that are not yet placed in a setting.

- Kranakis (2024). Invited survey (SSS 2024) on how knowledge of the communication model, of faults and of the target's distance and speed changes the competitive ratio of linear group search and evacuation, with open problems. Read; no new results.
- Czyzowicz et al. (2020f). The target, a bus, moves at constant speed u (their s) along a cycle of length 2π, and robots of speed 1 search for it. One robot that knows u but not the direction needs exactly $2\pi/u$ for $u \geq 1$, $4\pi/(u+1)$ for $1/3 \leq u \leq 1$ and $2\pi/(1-u)$ for $u \leq 1/3$ (Theorem 2); knowing neither, exactly $2\pi(1+1/(u+1))$, so some speed forces $4\pi - \varepsilon$ (Theorem 3). For k robots the bounds match when the direction is known (Theorem 4), when u is known and $u \geq 1$ or k is even (Theorem 5), and up to the parity of k when nothing is known (Theorem 6). Read in full.

Relations to other problems

- **Generalisation of Linear search (cow path).** A static target is the case $u = 0$; then capture by every robot is evacuation on the line or on m rays, and the ratio of (Jawhar and Kranakis 2025b) reduces to $3 + 2\sqrt{2}$ (Czyzowicz et al. 2021d). Algorithms carry over from linear search only for $u = 0$; lower bounds for a static target hold for a target of speed u chosen by the adversary.

Assumptions specific to this problem

Domain

Moving targets were studied on the line (Jawhar and Kranakis 2026b, 2025b); Jawhar and Kranakis call the m-ray star "a natural generalization of linear search to multiple directions" (Jawhar and Kranakis 2026a).

Claude (Anthropic) was used in cataloguing the problems.

Level	Value	Formal definition
0	m **rays**	$m \geq 2$ half-lines sharing their endpoint, the origin; the target moves along one of them.
1	**line**	The real line; the target moves along one side of the origin.

Number of robots

One robot: Alpern and Gal for a target moving away at known speed (as reported by Coleman et al. 2022) and Coleman, Kranakis, Krizanc and Morales-Ponce for the other cases (Coleman et al. 2022). Two robots that must both reach the target were studied by Jawhar and Kranakis (Jawhar and Kranakis 2026b, 2025b).

Level	Value	Formal definition
0	**any** k	$k \geq 1$ robots.
1	**one robot**	A single robot.
1	**two robots**	Two robots, both of which must reach the target.

Direction of the target

Coleman, Kranakis, Krizanc and Morales-Ponce treat targets moving away from and toward the origin separately and assume that the robot knows which (Coleman et al. 2022); the later papers read do the same. Tight bounds when the direction is unknown are left open in (Coleman et al. 2022); no catalogued setting leaves it unknown.

Level	Value	Formal definition
0	**direction unknown**	The target moves toward or away from the origin, and the robots do not know which.
1	**away**	The target moves away from the origin at constant speed $u < 1$, and the robots know it.
1	**toward**	The target moves toward the origin at constant speed $u \geq 0$, and the robots know it.

Knowledge of the target

The four cases (their FullKnowledge, NoDistance, NoSpeed and NoKnowledge) are those of Coleman, Kranakis, Krizanc and Morales-Ponce (Coleman et al. 2022), who extend the case of a known speed and an unknown distance treated by Alpern and Gal (as reported). Kranakis surveys how such knowledge changes the competitive ratio (Kranakis 2024).

Level	Value	Formal definition
0	**nothing known**	The robots know neither the initial distance d nor the speed u.
1	**speed known**	The robots know u but not d.
1	**distance known**	The robots know d but not u.
2	**both known**	The robots know d and u; only the side (ray) of the target is unknown.

Claude (Anthropic) was used in cataloguing the problems.

Case tree

The root assumes nothing; going down, each edge adds the assumptions written in grey (+) or leaves an axis unassumed. Bold entries are catalogued settings with their status per measure (Competitive ratio) and page; italic entries are branch points, named after the axis on which their branches differ. Axis order: Domain → Number of robots → Direction of the target → Knowledge of the target → Communication → Faults → Speeds → Randomness.

No assumptions

+ line

Number of robots

+ two robots

Direction of the target

+ away

Knowledge of the target

+ speed known

Communication

face-to-face, + no faults, + equal speeds

Line, two face-to-face robots, away, u known ● p. 551

+ sender/receiver, + no faults, + equal speeds

Line, a sender and a receiver, away, u known ● p. 552

+ distance known

Communication

face-to-face, + no faults, + equal speeds

Line, two face-to-face robots, away, d known ● p. 552

+ sender/receiver, + no faults, + equal speeds

Line, a sender and a receiver, away, d known ● p. 553

+ both known, + no faults, + equal speeds

Line, two face-to-face robots, away, d and u known ● p. 553

nothing known, face-to-face, + no faults, + equal speeds

Line, two face-to-face robots, away, nothing known ● p. 554

nothing known, + sender/receiver, + no faults, + equal speeds

Line, a sender and a receiver, away, nothing known ● p. 554

+ toward

Knowledge of the target

+ speed known

Communication

face-to-face, + no faults, + equal speeds

Line, two face-to-face robots, toward, u known ● p. 555

+ sender/receiver, + no faults, + equal speeds

Line, a sender and a receiver, toward, u known ● p. 555

+ distance known

Communication

face-to-face, + no faults, + equal speeds

Line, two face-to-face robots, toward, d known ● p. 556

Claude (Anthropic) was used in cataloguing the problems.

+ sender/receiver, + no faults, + equal speeds
Line, a sender and a receiver, toward, d known ● p. 556

+ both known, + no faults, + equal speeds
Line, two face-to-face robots, toward, d and u known ● p. 556

nothing known, face-to-face, + no faults, + equal speeds
Line, two face-to-face robots, toward, nothing known ● p. 557

nothing known, + sender/receiver, + no faults, + equal speeds
Line, a sender and a receiver, toward, nothing known ● p. 557

+ one robot
Direction of the target

+ away
Knowledge of the target

+ speed known, + no faults, + equal speeds
One robot, line, target moving away, speed known STANDARD SETTING ● p. 558

+ both known, + no faults, + equal speeds
One robot, line, target moving away, d and u known ● p. 559

+ distance known, + no faults, + equal speeds
One robot, line, target moving away, d known ● p. 559

nothing known, + no faults, + equal speeds
One robot, line, target moving away, nothing known ● p. 560

+ toward
Knowledge of the target

+ both known, + no faults, + equal speeds
One robot, line, target moving toward the origin, d and u known ● p. 561

+ speed known, + no faults, + equal speeds
One robot, line, target moving toward the origin, speed known ● p. 561

+ distance known, + no faults, + equal speeds
One robot, line, target moving toward the origin, d known ● p. 562

nothing known, + no faults, + equal speeds
One robot, line, target moving toward the origin, nothing known ● p. 563

m rays, + two robots, + away
Knowledge of the target

+ speed known
Communication

face-to-face, + no faults, + equal speeds
m rays, two face-to-face robots, away, u known ● p. 563

+ sender/receiver, + no faults, + equal speeds
m rays, a sender and a receiver, away, u known ● p. 563

+ distance known
Communication

face-to-face, + no faults, + equal speeds
m rays, two face-to-face robots, away, d known ● p. 564

+ sender/receiver, + no faults, + equal speeds
m rays, a sender and a receiver, away, d known ● p. 565

SEARCH

Claude (Anthropic) was used in cataloguing the problems.

nothing known, face-to-face, + no faults, + equal speeds
m rays, two face-to-face robots, away, nothing known • p. 565

nothing known, + sender/receiver, + no faults, + equal speeds
m rays, a sender and a receiver, away, nothing known • p. 566

What is known

The best upper bound against the best lower bound in each catalogued setting, in tree order. The settings follow in detail.

Setting	Competitive ratio	p.
Line, two face-to-face robots, away, u known	• polynomial gap $\frac{(3+u)^2}{(1-u)^2}$ vs $\frac{3-u}{1-u}$	551
Line, a sender and a receiver, away, u known	• unknown $\frac{\sqrt{u^2-2u+2}+1}{\sqrt{u^2-2u+2}-1}$ vs $3+2\sqrt{2}$ $(u=0)$	552
Line, two face-to-face robots, away, d known	• polynomial gap $O(M^8 \log^2 M)$, $M=\frac{1}{1-u}$ vs $\frac{3-u}{1-u}$	552
Line, a sender and a receiver, away, d known	• polynomial gap $1+O(M^{10/3}\log M)$, $M=\frac{1}{1-u}$ vs $1+\frac{2}{1-u}$	553
Line, two face-to-face robots, away, d and u known	• tight $\frac{3-u}{1-u}$ vs $\frac{3-u}{1-u}$	553
Line, two face-to-face robots, away, nothing known	• polynomial gap $12M^7+192(\log\log M+3)M^{10}\log^2 M/d$ vs —	554
Line, a sender and a receiver, away, nothing known	• polynomial gap $1+O(M^{16/3}\log M(\log\log M)^{3/2})$ vs —	554
Line, two face-to-face robots, toward, u known	• polynomial gap $\frac{(3-u)^2}{(1+u)^2}$ $(u<\frac{1}{3})$, $1+\frac{1}{u}$ $(u\geq\frac{1}{3})$ vs $\frac{3+u}{1+u}$ $(u\leq 1)$, $1+\frac{1}{u}$ $(u\geq 1)$	555
Line, a sender and a receiver, toward, u known	• unknown $\frac{\sqrt{u^2+2u+2}+1}{\sqrt{u^2+2u+2}-1}$ $(u\leq\frac{1}{3})$, $1+\frac{1}{u}$ $(u\geq\frac{1}{3})$ vs —	555
Line, two face-to-face robots, toward, d known	• tight 3 vs 3	556
Line, a sender and a receiver, toward, d known	• tight 3 vs 3	556
Line, two face-to-face robots, toward, d and u known	• tight $\frac{3+u}{1+u}$ $(u\leq 1)$, $1+\frac{1}{u}$ $(u\geq 1)$ vs $\frac{3+u}{1+u}$ $(u\leq 1)$, $1+\frac{1}{u}$ $(u\geq 1)$	556
Line, two face-to-face robots, toward, nothing known	• tight $1+\frac{1}{u}$ vs $1+\frac{1}{u}$	557
Line, a sender and a receiver, toward, nothing known	• tight $1+\frac{1}{u}$ vs $1+\frac{1}{u}$	557
One robot, line, target moving away, speed known STANDARD SETTING	• tight $\frac{(3+u)^2}{(1-u)^2}$ vs $\frac{(3+u)^2}{(1-u)^2}$	558
One robot, line, target moving away, d and u known	• tight $\frac{3-u}{1-u}$ vs $\frac{3-u}{1-u}$	559
One robot, line, target moving away, d known	• near-optimal 9 $(u\leq\frac{3}{4})$, $56.18\,(1-u)^{-4+(\log_2\log_2\frac{1}{1-u})^{-2}}$ $(u>\frac{3}{4})$ vs not $O\big((1-u)^{-4+\varepsilon}\big)$	559

Claude (Anthropic) was used in cataloguing the problems.

Setting	Competitive ratio	p.
One robot, line, target moving away, nothing known	● polynomial gap $1+\frac{8}{d}$ $(E\le 4)$, $1+\frac{56.18\,E^{4-(\log_2\log_2 E)^{-2}}-1}{d}$ $(E>4)$ vs not $O\big((1-u)^{-4+\varepsilon}\big)$	560
One robot, line, target moving toward the origin, d and u known	● tight $\frac{3+u}{1+u}$ $(u\le 1)$, $1+\frac{1}{u}$ $(u\ge 1)$ vs $\frac{3+u}{1+u}$ $(u\le 1)$, $1+\frac{1}{u}$ $(u\ge 1)$	561
One robot, line, target moving toward the origin, speed known	● tight $1+\frac{1}{u}$ $(u\ge\frac{1}{3})$, $\frac{(3-u)^2}{(1+u)^2}$ $(u<\frac{1}{3})$ vs $1+\frac{1}{u}$ $(u\ge\frac{1}{3})$, $\frac{(3-u)^2}{(1+u)^2}$ $(u<\frac{1}{3})$	561
One robot, line, target moving toward the origin, d known	● tight 3 vs 3	562
One robot, line, target moving toward the origin, nothing known	● tight $1+\frac{1}{u}$ vs $1+\frac{1}{u}$	563
m rays, two face-to-face robots, away, u known	● polynomial gap $1+\frac{(m+2)^{(m+2)/2}(1+u)^{m/2}}{m^{m/2}(1-u)^{(m+2)/2}}$ (m even) vs —	563
m rays, a sender and a receiver, away, u known	● polynomial gap $\frac{2}{1-u}+\frac{(m-1)(1+u^2)}{(1+u)(1-u)}\Big(\frac{(m+1)(1+u)}{(m-1)(1-u)}\Big)^{(m+1)/2}$ (m odd) vs m	563
m rays, two face-to-face robots, away, d known	● polynomial gap $1+O\big(M^{2^{(m+2)/2}}\log^2 M\big)$, $M=\frac{1}{1-u}$ (m even) vs —	564
m rays, a sender and a receiver, away, d known	● polynomial gap $O(M)+O\big(M^{2^{(m+1)/2}+1}\log^2 M\big)$, $M=\frac{1}{1-u}$ (m odd) vs m	565
m rays, two face-to-face robots, away, nothing known	● polynomial gap $1+O\big(M^{2^{(m+4)/2}}\log^2 M\log\log M\big)$ (m even) vs —	565
m rays, a sender and a receiver, away, nothing known	● polynomial gap $O\big(\frac{1}{1-u}\big)+O\Big(\frac{\log\log M}{d(1-u)}\log^2 M\,M^{2^{(m+3)/2}}\Big)$ (m odd) vs —	566

20.4.1 Line, two face-to-face robots, away, u known

Adds face-to-face, + no faults, + equal speeds DRAFT

Knowing only the speed, two face-to-face robots capture a target moving away with ratio $(3+u)^2/(1-u)^2$, the single-robot value, using three turns.

Assumed: domain: the line (two rays); number of robots: two robots; direction of the target: away from the origin, speed $u<1$; knowledge of the target: u known, d unknown; faults: no faults; speeds: all robots have the same unit speed. **Not assumed:** communication: face-to-face: robots exchange information only when they meet; randomness: deterministic.

Measure	Lower bound	Best upper bound	Status
Competitive ratio	$\frac{3-u}{1-u}$ (Jawhar and Kranakis 2026b)	$\frac{(3+u)^2}{(1-u)^2}$ (Jawhar and Kranakis 2026b)	● polynomial gap
	Against the full-knowledge bound, a factor $(3+u)/(1-u)$; at $u=0$ the ratio 9 is optimal (Chrobak et al. 2015).		

Results.

- Jawhar and Kranakis (2026b) [lower bound]: competitive ratio $\frac{3-u}{1-u}$. *Bottleneck:* unknown side. *Holds when:* bound for robots that know d and u (Theorem 2.2 of Jawhar and Kranakis 2026b); the argument does not use the communication model.
- Jawhar and Kranakis (2026b) [current best]: competitive ratio $\frac{(3+u)^2}{(1-u)^2}$. Introductory and best: each robot zigzags on its own side (Algorithm 4, Theorem 3.1, unboundedly many turns), or both walk apart at speed $(3u+1)/(3+u)$ and the finder fetches the other (Algorithm 5, Theorem 3.3, three turns, the fewest possible by Theorem 3.4).

Claude (Anthropic) was used in cataloguing the problems.

Open questions.

Q1. Is $(3+u)^2/(1-u)^2$ optimal for two face-to-face robots for every $u \in (0,1)$, as it is for one robot?

Notes. At $u = 0$ this is evacuation of two face-to-face robots on the line, where 9 is optimal (Chrobak et al. 2015).

20.4.2 Line, a sender and a receiver, away, u known

Adds + sender/receiver, + no faults, + equal speeds DRAFT

A sender and a receiver that know the speed capture a target moving away with ratio $(\sqrt{u^2-2u+2}+1)/(\sqrt{u^2-2u+2}-1)$*, which is* $3+2\sqrt{2}$ *at* $u=0$.

Assumed: domain: the line (two rays); number of robots: two robots; direction of the target: away from the origin, speed $u < 1$; knowledge of the target: u known, d unknown; communication: asymmetric: senders and receivers; faults: no faults; speeds: all robots have the same unit speed. **Not assumed:** randomness: deterministic.

Measure	Lower bound	Best upper bound	Status
Competitive ratio	$3+2\sqrt{2}$ $(u=0)$ (Czyzowicz et al. 2021d)	$\frac{\sqrt{u^2-2u+2}+1}{\sqrt{u^2-2u+2}-1}$ (Jawhar and Kranakis 2025b)	• unknown
	Optimal at $u = 0$; no lower bound for $u > 0$ beyond the full-knowledge argument.		

Results.

- Czyzowicz et al. (2021d) [lower bound]: competitive ratio $3+2\sqrt{2}$ $(u=0)$. *Bottleneck:* one-way communication. *Holds when:* static target; Theorem 10 of (Czyzowicz et al. 2021d).
- Jawhar and Kranakis (2025b) [current best]: competitive ratio $\frac{\sqrt{u^2-2u+2}+1}{\sqrt{u^2-2u+2}-1}$. Introductory and best (Algorithm 3, Theorem 4): the sender walks away at reduced speed, the receiver at unit speed; generalises (Czyzowicz et al. 2021d).

Open questions.

Q1. Is $(\sqrt{u^2-2u+2}+1)/(\sqrt{u^2-2u+2}-1)$ optimal for a sender and a receiver for $u > 0$?

Notes. Draft.

20.4.3 Line, two face-to-face robots, away, d known

Adds face-to-face, + no faults, + equal speeds DRAFT

Knowing only the initial distance, two face-to-face robots capture a target moving away with ratio $O(M^8 \log^2 M)$*,* $M = 1/(1-u)$.

Assumed: domain: the line (two rays); number of robots: two robots; direction of the target: away from the origin, speed $u < 1$; knowledge of the target: d known, u unknown; faults: no faults; speeds: all robots have the same unit speed. **Not assumed:** communication: face-to-face: robots exchange information only when they meet; randomness: deterministic.

Measure	Lower bound	Best upper bound	Status
Competitive ratio	$\frac{3-u}{1-u}$ (Jawhar and Kranakis 2026b)	$O(M^8 \log^2 M)$, $M = \frac{1}{1-u}$ (Jawhar and Kranakis 2026b)	• polynomial gap
	Polynomial in $1/(1-u)$.		

Claude (Anthropic) was used in cataloguing the problems.

Results.

- Jawhar and Kranakis (2026b) [lower bound]: competitive ratio $\frac{3-u}{1-u}$. *Bottleneck:* unknown side. *Holds when:* bound for robots that know d and u (Theorem 2.2 of Jawhar and Kranakis 2026b); the argument does not use the communication model.
- Jawhar and Kranakis (2026b) [current best]: competitive ratio $O(M^8 \log^2 M)$, $M = \frac{1}{1-u}$. Introductory and best (Algorithm 8, Theorem 4.1): the robots walk apart, guessing speeds $1 - 2^{-2^i}$, and speed up when a guess fails; three turns (Theorem 4.2).

Notes. For one robot, (Coleman et al. 2022) give $O(M^4 \log^2 M)$ (Theorem 12), and (Coleman et al. 2026a) $O(M^{4-(\log_2 \log_2 M)^{-2}})$ (Theorem 2) against no $O(M^{4-\varepsilon})$ (Theorem 1).

20.4.4 Line, a sender and a receiver, away, d known

Adds + sender/receiver, + no faults, + equal speeds DRAFT

Knowing only the initial distance, a sender and a receiver capture a target moving away with ratio $1 + O(M^{10/3} \log M)$, $M = 1/(1-u)$.

Assumed: domain: the line (two rays); number of robots: two robots; direction of the target: away from the origin, speed $u < 1$; knowledge of the target: d known, u unknown; communication: asymmetric: senders and receivers; faults: no faults; speeds: all robots have the same unit speed. **Not assumed:** randomness: deterministic.

Measure	Lower bound	Best upper bound	Status
Competitive ratio	$1 + \frac{2}{1-u}$ (Jawhar and Kranakis 2025b) Polynomial in $1/(1-u)$.	$1 + O(M^{10/3} \log M)$, $M = \frac{1}{1-u}$ (Jawhar and Kranakis 2025b)	• polynomial gap

Results.

- Jawhar and Kranakis (2025b) [lower bound]: competitive ratio $1 + \frac{2}{1-u}$. *Bottleneck:* unknown side. *Holds when:* Theorem 7; the first robot to reach $d/(1-u)$ finds the target on the other side.
- Jawhar and Kranakis (2025b) [current best]: competitive ratio $1 + O(M^{10/3} \log M)$, $M = \frac{1}{1-u}$. Introductory and best (Algorithm 5, Theorem 6).

Notes. Two robots with one-way communication beat the single-robot lower bound $\Omega(M^{4-\varepsilon})$ reported by (Jawhar and Kranakis 2025b).

20.4.5 Line, two face-to-face robots, away, d and u known

Adds + both known, + no faults, + equal speeds DRAFT

Robots that know d *and* u *but not the side capture a target moving away with ratio exactly* $(3-u)/(1-u)$.

Assumed: domain: the line (two rays); number of robots: two robots; direction of the target: away from the origin, speed $u < 1$; knowledge of the target: d and u known; faults: no faults; speeds: all robots have the same unit speed. **Not assumed:** communication: face-to-face: robots exchange information only when they meet; randomness: deterministic.

Measure	Lower bound	Best upper bound	Status
Competitive ratio	$\frac{3-u}{1-u}$ (Jawhar and Kranakis 2026b) Exact.	$\frac{3-u}{1-u}$ (Jawhar and Kranakis 2026b)	• tight

Claude (Anthropic) was used in cataloguing the problems.

Results.

- Jawhar and Kranakis (2026b) [lower bound]: competitive ratio $\frac{3-u}{1-u}$. *Bottleneck:* unknown side. *Holds when:* the first robot to reach distance $d/(1-u)$ finds the target on the other side; any communication model. Theorem 2.2.
- Jawhar and Kranakis (2026b) [current best]: competitive ratio $\frac{3-u}{1-u}$. Introductory and best (Theorem 2.1): both robots walk together to one side, then turn.

Notes. The same value as for one robot (Theorem 1 of Coleman et al. 2022).

20.4.6 Line, two face-to-face robots, away, nothing known

Adds nothing known, face-to-face, + no faults, + equal speeds DRAFT

Knowing neither distance nor speed, two face-to-face robots capture a target moving away with a ratio polynomial in $M = \max\{d, 1/(1-u)\}$.

Assumed: domain: the line (two rays); number of robots: two robots; direction of the target: away from the origin, speed $u < 1$; faults: no faults; speeds: all robots have the same unit speed. **Not assumed:** knowledge of the target: neither d nor u known; communication: face-to-face: robots exchange information only when they meet; randomness: deterministic.

Measure	Lower bound	Best upper bound	Status
Competitive ratio	none known	$12M^7 + 192(\log\log M + 3)M^{10}\log^2 M/d$ (Jawhar and Kranakis 2026b)	• polynomial gap
	No lower bound beyond those of the stronger settings.		

Results.

- Jawhar and Kranakis (2026b) [current best]: competitive ratio $12M^7+192(\log\log M+3)M^{10}\log^2 M/d$. Introductory and best (Algorithm 10, Theorem 5.1), with $M = \max\{d, 1/(1-u)\}$; three turns (Theorem 5.2).

Notes. Draft.

20.4.7 Line, a sender and a receiver, away, nothing known

Adds nothing known, + sender/receiver, + no faults, + equal speeds DRAFT

Knowing neither distance nor speed, a sender and a receiver capture a target moving away with ratio $1 + O(M^{16/3}\log M(\log\log M)^{3/2})$, $M = \max\{d, 1/(1-u)\}$.

Assumed: domain: the line (two rays); number of robots: two robots; direction of the target: away from the origin, speed $u < 1$; communication: asymmetric: senders and receivers; faults: no faults; speeds: all robots have the same unit speed. **Not assumed:** knowledge of the target: neither d nor u known; randomness: deterministic.

Measure	Lower bound	Best upper bound	Status
Competitive ratio	none known	$1 + O(M^{16/3}\log M (\log\log M)^{3/2})$ (Jawhar and Kranakis 2025b)	• polynomial gap
	No lower bound beyond those of the stronger settings.		

Results.

- Jawhar and Kranakis (2025b) [current best]: competitive ratio $1 + O(M^{16/3}\log M\,(\log\log M)^{3/2})$. Introductory and best (Algorithm 6, Theorem 9), with $M = \max\{d, 1/(1-u)\}$; the proof ends with a factor $6^{\log\log M}$, which the statement writes as $\log M$.

Claude (Anthropic) was used in cataloguing the problems.

Notes. Draft.

20.4.8 Line, two face-to-face robots, toward, u known

Adds face-to-face, + no faults, + equal speeds DRAFT

Knowing only the speed, two face-to-face robots capture a target moving toward the origin with ratio $(3-u)^2/(1+u)^2$ for $u < 1/3$ and $1 + 1/u$ otherwise.

Assumed: domain: the line (two rays); number of robots: two robots; direction of the target: toward the origin, speed $u \geq 0$; knowledge of the target: u known, d unknown; faults: no faults; speeds: all robots have the same unit speed. **Not assumed:** communication: face-to-face: robots exchange information only when they meet; randomness: deterministic.

Measure	Lower bound	Best upper bound	Status
Competitive ratio	$\frac{3+u}{1+u}$ $(u \leq 1)$, $1 + \frac{1}{u}$ $(u \geq 1)$ (Jawhar and Kranakis 2026b) No lower bound beyond the full-knowledge one.	$\frac{(3-u)^2}{(1+u)^2}$ $(u < \frac{1}{3})$, $1 + \frac{1}{u}$ $(u \geq \frac{1}{3})$ (Jawhar and Kranakis 2026b)	• polynomial gap

Results.

- Jawhar and Kranakis (2026b) [lower bound]: competitive ratio $\frac{3+u}{1+u}$ $(u \leq 1)$, $1 + \frac{1}{u}$ $(u \geq 1)$. *Bottleneck:* unknown side. *Holds when:* full-knowledge bound (Theorem 2.5 of Jawhar and Kranakis 2026b).
- Jawhar and Kranakis (2026b) [current best]: competitive ratio $\frac{(3-u)^2}{(1+u)^2}$ $(u < \frac{1}{3})$, $1 + \frac{1}{u}$ $(u \geq \frac{1}{3})$. Introductory and best: zigzag (Theorem 3.5) or walking apart at speed $(1-3u)/(3-u)$ (Algorithm 7, Theorem 3.6, three turns); waiting at the origin when $u \geq 1/3$.

Notes. Draft.

20.4.9 Line, a sender and a receiver, toward, u known

Adds + sender/receiver, + no faults, + equal speeds DRAFT

A sender and a receiver that know the speed capture a target moving toward the origin with ratio $(\sqrt{u^2+2u+2}+1)/(\sqrt{u^2+2u+2}-1)$ for $u \leq 1/3$ and $1 + 1/u$ above.

Assumed: domain: the line (two rays); number of robots: two robots; direction of the target: toward the origin, speed $u \geq 0$; knowledge of the target: u known, d unknown; communication: asymmetric: senders and receivers; faults: no faults; speeds: all robots have the same unit speed. **Not assumed:** randomness: deterministic.

Measure	Lower bound	Best upper bound	Status
Competitive ratio	none known No lower bound is stated.	$\frac{\sqrt{u^2+2u+2}+1}{\sqrt{u^2+2u+2}-1}$ $(u \leq \frac{1}{3})$, $1 + \frac{1}{u}$ $(u \geq \frac{1}{3})$ (Jawhar and Kranakis 2025b)	• unknown

Results.

- Jawhar and Kranakis (2025b) [current best]: competitive ratio $\frac{\sqrt{u^2+2u+2}+1}{\sqrt{u^2+2u+2}-1}$ $(u \leq \frac{1}{3})$, $1 + \frac{1}{u}$ $(u \geq \frac{1}{3})$. Introductory and best: walking apart at two speeds (Algorithm 1, Theorem 1) or waiting (Theorem 3); a zigzag strategy (Theorem 2) is dominated.

Notes. Draft.

Claude (Anthropic) was used in cataloguing the problems.

20.4.10 Line, two face-to-face robots, toward, d known

Adds face-to-face, + no faults, + equal speeds DRAFT

Knowing only the initial distance, two face-to-face robots capture a target moving toward the origin with ratio exactly 3.

Assumed: domain: the line (two rays); number of robots: two robots; direction of the target: toward the origin, speed $u \geq 0$; knowledge of the target: d known, u unknown; faults: no faults; speeds: all robots have the same unit speed. **Not assumed:** communication: face-to-face: robots exchange information only when they meet; randomness: deterministic.

Measure	Lower bound	Best upper bound	Status
Competitive ratio	3 (Jawhar and Kranakis 2026b) Exactly 3.	3 (Jawhar and Kranakis 2026b)	• tight

Results.

- Jawhar and Kranakis (2026b) [lower bound]: competitive ratio 3. *Bottleneck:* unknown side. *Holds when:* Theorem 4.4; a slow target placed at $-d$ after a robot visits d.
- Jawhar and Kranakis (2026b) [current best]: competitive ratio 3. Introductory and best (Algorithm 9, Theorem 4.3): both walk distance d to one side, then turn.

Notes. Draft.

20.4.11 Line, a sender and a receiver, toward, d known

Adds + sender/receiver, + no faults, + equal speeds DRAFT

Knowing only the initial distance, a sender and a receiver capture a target moving toward the origin with ratio exactly 3.

Assumed: domain: the line (two rays); number of robots: two robots; direction of the target: toward the origin, speed $u \geq 0$; knowledge of the target: d known, u unknown; communication: asymmetric: senders and receivers; faults: no faults; speeds: all robots have the same unit speed. **Not assumed:** randomness: deterministic.

Measure	Lower bound	Best upper bound	Status
Competitive ratio	3 (Jawhar and Kranakis 2025b) Exactly 3.	3 (Jawhar and Kranakis 2025b)	• tight

Results.

- Jawhar and Kranakis (2025b) [lower bound]: competitive ratio 3. *Bottleneck:* unknown side. *Holds when:* Theorem 5.
- Jawhar and Kranakis (2025b) [current best]: competitive ratio 3. Introductory and best (Algorithm 4, Theorem 5): the robots move together, as in the face-to-face case.

Notes. One-way communication does not help here.

20.4.12 Line, two face-to-face robots, toward, d and u known

Adds + both known, + no faults, + equal speeds DRAFT

Robots that know d and u capture a target moving toward the origin with ratio $(3+u)/(1+u)$ when $u \leq 1$

Claude (Anthropic) was used in cataloguing the problems.

and $1 + 1/u$ *when* $u \geq 1$*, both optimal.*

Assumed: domain: the line (two rays); number of robots: two robots; direction of the target: toward the origin, speed $u \geq 0$; knowledge of the target: d and u known; faults: no faults; speeds: all robots have the same unit speed. **Not assumed:** communication: face-to-face: robots exchange information only when they meet; randomness: deterministic.

Measure	Lower bound	Best upper bound	Status
Competitive ratio	$\frac{3+u}{1+u}$ $(u \leq 1)$, $1 + \frac{1}{u}$ $(u \geq 1)$ (Jawhar and Kranakis 2026b) Exact.	$\frac{3+u}{1+u}$ $(u \leq 1)$, $1 + \frac{1}{u}$ $(u \geq 1)$ (Jawhar and Kranakis 2026b)	• tight

Results.

- Jawhar and Kranakis (2026b) [lower bound]: competitive ratio $\frac{3+u}{1+u}$ $(u \leq 1)$, $1 + \frac{1}{u}$ $(u \geq 1)$. *Bottleneck:* unknown side. *Holds when:* Theorem 2.5; any communication model.
- Jawhar and Kranakis (2026b) [current best]: competitive ratio $\frac{3+u}{1+u}$ $(u \leq 1)$, $1 + \frac{1}{u}$ $(u \geq 1)$. Introductory and best: go to distance $d/(1+u)$ and turn if needed (Theorem 2.3), or wait at the origin when $u \geq 1$ (Theorem 2.4).

Notes. The same values as for one robot (Theorems 2–4 of Coleman et al. 2022).

20.4.13 Line, two face-to-face robots, toward, nothing known

Adds nothing known, face-to-face, + no faults, + equal speeds DRAFT

Knowing neither distance nor speed, the best two face-to-face robots can do against a target moving toward the origin is to wait: ratio $1 + 1/u$*.*

Assumed: domain: the line (two rays); number of robots: two robots; direction of the target: toward the origin, speed $u \geq 0$; faults: no faults; speeds: all robots have the same unit speed. **Not assumed:** knowledge of the target: neither d nor u known; communication: face-to-face: robots exchange information only when they meet; randomness: deterministic.

Measure	Lower bound	Best upper bound	Status
Competitive ratio	$1 + \frac{1}{u}$ (Jawhar and Kranakis 2026b) Exact; unbounded as $u \to 0$.	$1 + \frac{1}{u}$ (Jawhar and Kranakis 2026b)	• tight

Results.

- Jawhar and Kranakis (2026b) [lower bound]: competitive ratio $1 + \frac{1}{u}$. *Bottleneck:* unknown side. *Holds when:* Theorem 5.3.
- Jawhar and Kranakis (2026b) [current best]: competitive ratio $1 + \frac{1}{u}$. Introductory and best: both robots wait at the origin (Theorem 5.3).

Notes. Draft.

20.4.14 Line, a sender and a receiver, toward, nothing known

Adds nothing known, + sender/receiver, + no faults, + equal speeds DRAFT

Knowing neither distance nor speed, a sender and a receiver can do no better than waiting: ratio $1 + 1/u$*.*

Assumed: domain: the line (two rays); number of robots: two robots; direction of the target: toward the origin, speed $u \geq 0$; communication: asymmetric: senders and receivers; faults: no faults; speeds: all robots have the same unit speed. **Not assumed:** knowledge of the target: neither d nor u known; randomness: deterministic.

Claude (Anthropic) was used in cataloguing the problems.

Measure	Lower bound	Best upper bound	Status
Competitive ratio	$1+\frac{1}{u}$ (Jawhar and Kranakis 2025b)	$1+\frac{1}{u}$ (Jawhar and Kranakis 2025b)	• tight
	Exact.		

Results.

- Jawhar and Kranakis (2025b) [lower bound]: competitive ratio $1+\frac{1}{u}$. *Bottleneck:* unknown side. *Holds when:* Theorem 8.
- Jawhar and Kranakis (2025b) [current best]: competitive ratio $1+\frac{1}{u}$. Introductory and best: both robots wait at the origin (Theorem 8).

Notes. Draft.

20.4.15 One robot, line, target moving away, speed known

Adds + speed known, + no faults, + equal speeds STANDARD SETTING DRAFT

One robot that knows the speed $u<1$ of a target moving away from the origin, but not its distance, zigzags with turning distances that grow by the factor $2(1+u)/(1-u)$; its ratio $(3+u)^2/(1-u)^2$ is optimal (Alpern and Gal, as reported).

Assumed: domain: the line (two rays); number of robots: a single robot; direction of the target: away from the origin, speed $u<1$; knowledge of the target: u known, d unknown; faults: no faults; speeds: all robots have the same unit speed. **Not assumed:** communication: face-to-face: robots exchange information only when they meet; randomness: deterministic.

Measure	Lower bound	Best upper bound	Status
Competitive ratio	$\frac{(3+u)^2}{(1-u)^2}$ (Alpern and Gal 2003)	$\frac{(3+u)^2}{(1-u)^2}$ (Alpern and Gal 2003)	• tight
	As reported; the book of Alpern and Gal was not read.		

Results.

- Alpern and Gal (2003) [lower bound]: competitive ratio $\frac{(3+u)^2}{(1-u)^2}$. *Bottleneck:* unknown distance. *Holds when:* one robot; as reported by (Coleman et al. 2022) (Theorem 6) and (Jawhar and Kranakis 2025b); the book was not read. At $u=0$ it is the bound 9 of linear search.
- Alpern and Gal (2003) [current best]: competitive ratio $\frac{(3+u)^2}{(1-u)^2}$. Introductory and current best, as reported by (Coleman et al. 2022) (Theorem 6, which writes it $1+8\frac{1+u}{(1-u)^2}$) and (Jawhar and Kranakis 2025b) (who cite p. 134, Equation 8.25): the zigzag of Algorithm 3 of (Coleman et al. 2022), which turns at distances $1, a, a^2, \ldots$ on alternate sides, with $a=2(1+u)/(1-u)$. Not read.

Open questions.

Q1. Tight bounds when the robot does not know whether the target moves toward or away from the origin (Coleman et al. 2022).

Notes. Draft because the source of the bound, the book of Alpern and Gal, was not read; (Coleman et al. 2022) state the result without proof. At $u=0$ the zigzag is the doubling strategy of linear search.

Claude (Anthropic) was used in cataloguing the problems.

20.4.16 One robot, line, target moving away, d and u known

Adds + both known, + no faults, + equal speeds VERIFIED

A robot that knows d and u but not the side goes far enough to catch the target on one side and then turns: ratio exactly $(3-u)/(1-u)$.

Assumed: domain: the line (two rays); number of robots: a single robot; direction of the target: away from the origin, speed $u < 1$; knowledge of the target: d and u known; faults: no faults; speeds: all robots have the same unit speed. **Not assumed:** communication: face-to-face: robots exchange information only when they meet; randomness: deterministic.

Measure	Lower bound	Best upper bound	Status
Competitive ratio	$\frac{3-u}{1-u}$ (Coleman et al. 2022)	$\frac{3-u}{1-u}$ (Coleman et al. 2022)	• tight
	Exactly $(3-u)/(1-u)$; 3 for a static target.		

Results.

- Coleman et al. (2022) [lower bound]: competitive ratio $\frac{3-u}{1-u}$. *Bottleneck:* unknown side. *Holds when:* the robot must visit $d/(1-u)$ or $-d/(1-u)$ to catch the target, and the adversary puts the target on the other side (Theorem 1). Theorem 1, written $1 + 2/(1-u)$ in the paper.
- Coleman et al. (2022) [current best]: competitive ratio $\frac{3-u}{1-u}$. Introductory and current best (Algorithm 1, Theorem 1): go for time $d/(1-u)$ on one side; if the target is not there, turn and go until it is found.

Notes. Checked against the full text of (Coleman et al. 2022) (OPODIS 2022).

20.4.17 One robot, line, target moving away, d known

Adds + distance known, + no faults, + equal speeds VERIFIED

A robot that knows the distance but not the speed of a target moving away guesses ever larger speeds, trip by trip: its ratio is $(1-u)^{-4+o(1)}$, and no strategy achieves $O((1-u)^{-4+\varepsilon})$.

Assumed: domain: the line (two rays); number of robots: a single robot; direction of the target: away from the origin, speed $u < 1$; knowledge of the target: d known, u unknown; faults: no faults; speeds: all robots have the same unit speed. **Not assumed:** communication: face-to-face: robots exchange information only when they meet; randomness: deterministic.

Measure	Lower bound	Best upper bound	Status
Competitive ratio	not $O\big((1-u)^{-4+\varepsilon}\big)$ (Coleman et al. 2026a); $\frac{3-u}{1-u}$ (Coleman et al. 2022)	9 $(u \le \frac{3}{4})$, 56.18 $(1-u)^{-4+(\log_2\log_2 \frac{1}{1-u})^{-2}}$ $(u > \frac{3}{4})$ (Coleman et al. 2026a)	• near-optimal
	Both bounds have exponent 4: with $M = 1/(1-u)$, $O(M^{4-(\log_2\log_2 M)^{-2}})$ against no $O(M^{4-\varepsilon})$. The remaining factor $M^{(\log_2\log_2 M)^{-2}}$ grows more slowly than any power of M but faster than any power of $\log M$; the authors call the result optimal up to lower-order terms in the exponent (Coleman et al. 2026a).		

Results.

- Coleman et al. (2026a) [lower bound]: competitive ratio not $O\big((1-u)^{-4+\varepsilon}\big)$. *Bottleneck:* unknown speed. *Holds when:* for every constant $\varepsilon > 0$ (Theorem 1); every strategy; zigzag strategies that return to the origin after each trip are no worse; adapts the argument of Beck and Newman for linear search.
- Coleman et al. (2022) [lower bound]: competitive ratio $\frac{3-u}{1-u}$. *Bottleneck:* unknown side. *Holds when:* bound for a robot that knows d and u (Theorem 1), which holds a fortiori here (Remark 13).
- Coleman et al. (2022) [introductory]: competitive ratio 5 $(u \le \frac{1}{2})$, $1 + \frac{16\log^2 \frac{1}{1-u}}{(1-u)^4}$ $(u > \frac{1}{2})$. Introductory (Algorithm 5, Theorem 12, with Lemmas 10 and 11; logarithms to base 2): in round i the robot guesses the speed $1 - 2^{-2^i}$,

Claude (Anthropic) was used in cataloguing the problems.

goes on alternate sides far enough to catch such a target, and returns to the origin. Better than (Coleman et al. 2026a) for $u \le 1/2$ (5 against 9).

- Coleman et al. (2026a) [current best]: competitive ratio 9 ($u \le \frac{3}{4}$), $56.18\,(1-u)^{-4+(\log_2 \log_2 \frac{1}{1-u})^{-2}}$ ($u > \frac{3}{4}$). Current best for $u > 1/2$ (Algorithm 1, Theorem 2): a zigzag strategy that guesses a larger speed on each trip, goes far enough to catch a target of that speed, and returns to the origin; the guesses grow faster than in (Coleman et al. 2022). The paper writes v for the speed and u for the evasiveness $1/(1-v)$.

Open questions.

Q1. Determine the lower-order terms in the exponent: the bounds differ by a factor $(1-u)^{-o(1)}$ (Coleman et al. 2026a).

Notes. Without an upper bound on u below 1 the robot cannot use the algorithm for known u; a known bound $\hat{u} < 1$ on the speed would give $1 + 2/(1-\hat{u})$ (Section 4.1 of Coleman et al. 2022). (Coleman et al. 2022) asked for tight bounds for an unknown speed; (Coleman et al. 2026a) settle the exponent. Checked against the full texts of (Coleman et al. 2022) (OPODIS 2022) and (Coleman et al. 2026a) (arXiv v2, the full version; the JCSS version states the same bounds as Theorems 3 and 4).

20.4.18 One robot, line, target moving away, nothing known

Adds nothing known, + no faults, + equal speeds DRAFT

A robot that knows neither the distance nor the speed of a target moving away runs the algorithm for a known distance as if $d = 1$: its ratio grows like $d^3/(1-u)^4$ up to lower-order terms in the exponent, against $(3+u)^2/(1-u)^2$ and no $O((1-u)^{-4+\varepsilon})$.

Assumed: domain: the line (two rays); number of robots: a single robot; direction of the target: away from the origin, speed $u < 1$; faults: no faults; speeds: all robots have the same unit speed. **Not assumed:** knowledge of the target: neither d nor u known; communication: face-to-face: robots exchange information only when they meet; randomness: deterministic.

Measure	Lower bound	Best upper bound	Status
Competitive ratio	not $O\big((1-u)^{-4+\varepsilon}\big)$ (Coleman et al. 2026a); $\frac{(3+u)^2}{(1-u)^2}$ (Alpern and Gal 2003); $\frac{3-u}{1-u}$ (Coleman et al. 2022)	$1 + \frac{8}{d}$ ($E \le 4$), $1 + \frac{56.18\,E^{4-(\log_2 \log_2 E)^{-2}} - 1}{d}$ ($E > 4$) (Coleman et al. 2026a)	• polynomial gap
	The upper bound grows like $d^3(1-u)^{-4}$ up to lower-order terms in the exponent; the lower bounds do not depend on d.		

Results.

- Coleman et al. (2026a) [lower bound]: competitive ratio not $O\big((1-u)^{-4+\varepsilon}\big)$. *Bottleneck:* unknown speed. *Holds when:* for every constant $\varepsilon > 0$; bound for a robot that knows d (Theorem 1), which holds a fortiori here (end of Section 3).
- Alpern and Gal (2003) [lower bound]: competitive ratio $\frac{(3+u)^2}{(1-u)^2}$. *Bottleneck:* unknown distance. *Holds when:* bound for a robot that knows u (Remark 17 of Coleman et al. 2022); as reported; the book was not read.
- Coleman et al. (2022) [lower bound]: competitive ratio $\frac{3-u}{1-u}$. *Bottleneck:* unknown side. *Holds when:* bound for a robot that knows d and u (Theorem 1).
- Coleman et al. (2022) [introductory]: competitive ratio $1 + \frac{16}{d}(\log\log M + 3)\,M^8 \log^2 M$. Introductory (Algorithm 7, Theorem 16, Lemma 15), with $M = \max\{d, 1/(1-u)\}$ and logarithms to base 2: in round i the robot guesses the speed $1 - 2^{-2^i}$ and the distance 2^{2^i}.
- Coleman et al. (2026a) [current best]: competitive ratio $1 + \frac{8}{d}$ ($E \le 4$), $1 + \frac{56.18\,E^{4-(\log_2 \log_2 E)^{-2}} - 1}{d}$ ($E > 4$). Current best (Algorithm 2, Theorem 3), with $E = d/(1-u)$: the robot runs the algorithm for a known distance with $d = 1$; a

Claude (Anthropic) was used in cataloguing the problems.

target at distance d is caught no later than one at distance 1 with evasiveness E (Lemma 11). Since $E \le M^2$, this improves on (Coleman et al. 2022).

Open questions.

Q1. Tight bounds when neither d nor u is known (Coleman et al. 2026a); the upper bound grows with d, the lower bounds do not.

Notes. Draft because the bound $(3+u)^2/(1-u)^2$ rests on the result of Alpern and Gal, which was not read. The other bounds were checked against the full texts of (Coleman et al. 2022) (OPODIS 2022) and (Coleman et al. 2026a) (arXiv v2, Theorem 3; Theorem 5 of the JCSS version). The JCSS version states the Alpern–Gal bound as its Theorem 2 without a proof, so it does not settle this.

20.4.19 One robot, line, target moving toward the origin, d and u known

Adds + both known, + no faults, + equal speeds VERIFIED

A robot that knows d and u goes for time $d/(1+u)$ on one side and turns if needed, with ratio $(3+u)/(1+u)$ for $u \le 1$; for $u \ge 1$ waiting at the origin, with ratio $1 + 1/u$, is optimal.

Assumed: domain: the line (two rays); number of robots: a single robot; direction of the target: toward the origin, speed $u \ge 0$; knowledge of the target: d and u known; faults: no faults; speeds: all robots have the same unit speed. **Not assumed:** communication: face-to-face: robots exchange information only when they meet; randomness: deterministic.

Measure	Lower bound	Best upper bound	Status
Competitive ratio	$\frac{3+u}{1+u}$ $(u \le 1)$, $1 + \frac{1}{u}$ $(u \ge 1)$ (Coleman et al. 2022) Exact for every u.	$\frac{3+u}{1+u}$ $(u \le 1)$, $1 + \frac{1}{u}$ $(u \ge 1)$ (Coleman et al. 2022)	• tight

Results.

- Coleman et al. (2022) [lower bound]: competitive ratio $\frac{3+u}{1+u}$ $(u \le 1)$, $1 + \frac{1}{u}$ $(u \ge 1)$. *Bottleneck:* unknown side. *Holds when:* the adversary puts the target on the side opposite to the first point $\pm a$ that the robot occupies exactly when the target would reach it (Theorem 3, for $u < 1$); for $u \ge 1$ the same bound is smallest when the robot never leaves the origin (Theorem 4). Theorems 3 and 4; the paper writes $1 + 2/(1+u)$.
- Coleman et al. (2022) [current best]: competitive ratio $\frac{3+u}{1+u}$ $(u \le 1)$, $1 + \frac{1}{u}$ $(u \ge 1)$. Introductory and current best: Algorithm 2 (go for time $d/(1+u)$ on one side, then turn; Theorem 2) for $u < 1$, and the waiting algorithm, which stays at the origin until the target arrives (Theorem 4), for $u \ge 1$.

Notes. The waiting algorithm uses neither d nor u, so it applies in every setting in which the target moves toward the origin (Remark 5 of Coleman et al. 2022). Checked against the full text of (Coleman et al. 2022) (OPODIS 2022).

20.4.20 One robot, line, target moving toward the origin, speed known

Adds + speed known, + no faults, + equal speeds VERIFIED

A robot that knows the speed but not the distance of a target moving toward the origin should wait at the origin if $u \ge 1/3$ and otherwise zigzag with turning distances that grow by the factor $2(1-u)/(1+u)$; both are optimal.

Assumed: domain: the line (two rays); number of robots: a single robot; direction of the target: toward the origin, speed $u \ge 0$; knowledge of the target: u known, d unknown; faults: no faults; speeds: all robots have the same unit speed. **Not assumed:** communication: face-to-face: robots exchange information only when they meet; randomness: deterministic.

Claude (Anthropic) was used in cataloguing the problems.

Measure	Lower bound	Best upper bound	Status
Competitive ratio	$1+\frac{1}{u}$ ($u \geq \frac{1}{3}$), $\frac{(3-u)^2}{(1+u)^2}$ ($u < \frac{1}{3}$) (Coleman et al. 2022) Exact for every u.	$1+\frac{1}{u}$ ($u \geq \frac{1}{3}$), $\frac{(3-u)^2}{(1+u)^2}$ ($u < \frac{1}{3}$) (Coleman et al. 2022)	• tight

Results.

- Coleman et al. (2022) [lower bound]: competitive ratio $1+\frac{1}{u}$ ($u \geq \frac{1}{3}$), $\frac{(3-u)^2}{(1+u)^2}$ ($u < \frac{1}{3}$). *Bottleneck:* unknown distance. *Holds when:* any strategy: the adversary takes the limit β of the ratio of time to distance from the origin along the robot's trajectory and places the target on the side the robot is not on, just beyond what the robot can have reached (Theorem 9, with techniques of an unpublished manuscript of Killick); a contracting strategy, which never goes beyond some distance on one side, cannot beat $1+1/u$ (Theorem 7). Theorem 9; the paper writes $1+8\frac{1-u}{(1+u)^2}$ for $u < 1/3$.

- Coleman et al. (2022) [current best]: competitive ratio $1+\frac{1}{u}$ ($u \geq \frac{1}{3}$), $\frac{(3-u)^2}{(1+u)^2}$ ($u < \frac{1}{3}$). Introductory and current best (Algorithm 4, Theorem 8): wait at the origin if $u \geq 1/3$; otherwise zigzag, turning at distances $1, a, a^2, \ldots$ on alternate sides with $a = 2(1-u)/(1+u)$.

Notes. A target moving toward the origin comes to the robot, so waiting is a strategy; it is optimal down to speed $1/3$, and the zigzag below. Checked against the full text of (Coleman et al. 2022) (OPODIS 2022).

SEARCH

20.4.21 One robot, line, target moving toward the origin, d known

Adds + distance known, + no faults, + equal speeds VERIFIED

A robot that knows the distance of a target moving toward the origin goes for time d on one side and turns: ratio exactly 3, for every speed.

Assumed: domain: the line (two rays); number of robots: a single robot; direction of the target: toward the origin, speed $u \geq 0$; knowledge of the target: d known, u unknown; faults: no faults; speeds: all robots have the same unit speed. **Not assumed:** communication: face-to-face: robots exchange information only when they meet; randomness: deterministic.

Measure	Lower bound	Best upper bound	Status
Competitive ratio	3 (Coleman et al. 2022) Exactly 3.	3 (Coleman et al. 2022)	• tight

Results.

- Coleman et al. (2022) [lower bound]: competitive ratio 3. *Bottleneck:* unknown side. *Holds when:* a slow target ($u = \varepsilon/3$) placed on the side opposite to the first of the points $\pm(d-\varepsilon)$ that the robot reaches (Theorem 14); a robot that never visits $\pm d$ has an unbounded ratio for small u.

- Coleman et al. (2022) [current best]: competitive ratio 3. Introductory and current best (Algorithm 6, Theorem 14): go for time d on one side, then turn.

Notes. Two robots do no better, whether they communicate face-to-face (Jawhar and Kranakis 2026b) or as a sender and a receiver (Jawhar and Kranakis 2025b). Checked against the full text of (Coleman et al. 2022) (OPODIS 2022).

Claude (Anthropic) was used in cataloguing the problems.

20.4.22 One robot, line, target moving toward the origin, nothing known

Adds nothing known, + no faults, + equal speeds VERIFIED

A robot that knows neither the distance nor the speed of a target moving toward the origin can do no better than wait for it: ratio exactly $1 + 1/u$.

Assumed: domain: the line (two rays); number of robots: a single robot; direction of the target: toward the origin, speed $u \geq 0$; faults: no faults; speeds: all robots have the same unit speed. **Not assumed:** knowledge of the target: neither d nor u known; communication: face-to-face: robots exchange information only when they meet; randomness: deterministic.

Measure	Lower bound	Best upper bound	Status
Competitive ratio	$1 + \frac{1}{u}$ (Coleman et al. 2022)	$1 + \frac{1}{u}$ (Coleman et al. 2022)	• tight
	Exact; it grows without bound as $u \to 0$.		

Results.

- Coleman et al. (2022) [lower bound]: competitive ratio $1 + \frac{1}{u}$. *Bottleneck:* unknown side. *Holds when:* a robot that waits for time t and then goes distance d' on one side meets a target that starts at distance d on the other side with speed $u = d/(t + d')$ no earlier than time d/u (Theorem 18).
- Coleman et al. (2022) [current best]: competitive ratio $1 + \frac{1}{u}$. Introductory and current best: the waiting algorithm (Theorems 4 and 18).

Notes. Checked against the full text of (Coleman et al. 2022) (OPODIS 2022).

20.4.23 m rays, two face-to-face robots, away, u known

Adds face-to-face, + no faults, + equal speeds DRAFT

On m rays, two face-to-face robots that know the speed capture a target moving away with a ratio that grows exponentially with m.

Assumed: number of robots: two robots; direction of the target: away from the origin, speed $u < 1$; knowledge of the target: u known, d unknown; faults: no faults; speeds: all robots have the same unit speed. **Not assumed:** domain: m rays from a common origin; communication: face-to-face: robots exchange information only when they meet; randomness: deterministic.

Measure	Lower bound	Best upper bound	Status
Competitive ratio	none known	$1 + \frac{(m+2)^{(m+2)/2}(1+u)^{m/2}}{m^{m/2}(1-u)^{(m+2)/2}}$ (m even) (Jawhar and Kranakis 2026a) [preprint]	• polynomial gap
	Only the bound m of Lemma 4.1 (in the sender/receiver setting) applies.		

Results.

- Jawhar and Kranakis (2026a) [preprint] [current best]: competitive ratio $1 + \frac{(m+2)^{(m+2)/2}(1+u)^{m/2}}{m^{m/2}(1-u)^{(m+2)/2}}$ (m even). Introductory and best (Algorithm 1, Theorem 2.1): the robots explore two new rays per round with geometrically growing depths; for odd m replace m by $m + 1$. Preprint.

Notes. At $m = 2$, $u = 0$ the ratio is 9.

20.4.24 m rays, a sender and a receiver, away, u known

Adds + sender/receiver, + no faults, + equal speeds DRAFT

On m rays, a sender and a receiver that know the speed improve on face-to-face robots when m is odd.

Claude (Anthropic) was used in cataloguing the problems.

Assumed: number of robots: two robots; direction of the target: away from the origin, speed $u < 1$; knowledge of the target: u known, d unknown; communication: asymmetric: senders and receivers; faults: no faults; speeds: all robots have the same unit speed. **Not assumed:** domain: m rays from a common origin; randomness: deterministic.

Measure	Lower bound	Best upper bound	Status
Competitive ratio	m (Jawhar and Kranakis 2026a) [preprint]; $(m-1)+2+2\sqrt{2}$ $(u=0)$ (Jawhar and Kranakis 2026a) [preprint] For $m=3$, $u=0$: $4+2\sqrt{2}\approx 6.83$ against 10.	$\frac{2}{1-u}+\frac{(m-1)(1+u^2)}{(1+u)(1-u)}\left(\frac{(m+1)(1+u)}{(m-1)(1-u)}\right)^{(m+1)/2}$ (m odd) (Jawhar and Kranakis 2026a) [preprint]	• polynomial gap

Results.

- Jawhar and Kranakis (2026a) [preprint] [lower bound]: competitive ratio m. *Bottleneck:* unknown ray. *Holds when:* Lemma 4.1; any deterministic algorithm with two unit-speed robots; the target on the last ray explored.
- Jawhar and Kranakis (2026a) [preprint] [lower bound]: competitive ratio $(m-1)+2+2\sqrt{2}$ $(u=0)$. *Bottleneck:* one-way communication. *Holds when:* Theorem 3.3; static target; adapts Lemmas 26 and 29 of (Czyzowicz et al. 2021d); the proof is a sketch.
- Jawhar and Kranakis (2026a) [preprint] [current best]: competitive ratio $\frac{2}{1-u}+\frac{(m-1)(1+u^2)}{(1+u)(1-u)}\left(\frac{(m+1)(1+u)}{(m-1)(1-u)}\right)^{(m+1)/2}$ (m odd). Introductory and best for odd m (Algorithm 4, Theorem 3.2); 10 for $m=3$, $u=0$. For even m the paper uses the face-to-face algorithm. Preprint.

Open questions.

Q1. For even $m \geq 4$, can a sender and a receiver beat two face-to-face robots, as they do on the line?

Notes. Lemma 3.1 of (Jawhar and Kranakis 2026a) claims that for even m a sender and a receiver do no better than face-to-face robots; for $m=2$ and a static target this contradicts $3+2\sqrt{2}$ (Czyzowicz et al. 2021d) against 9 (Chrobak et al. 2015), and for moving targets on the line the sender/receiver bounds of (Jawhar and Kranakis 2025b) are smaller than the face-to-face ones of (Jawhar and Kranakis 2026b). Even m is therefore open for a sender and a receiver.

20.4.25 m rays, two face-to-face robots, away, d known

Adds face-to-face, + no faults, + equal speeds DRAFT

On m rays, knowing only the initial distance, two face-to-face robots capture a target moving away with a ratio whose exponent in $1/(1-u)$ is about $2^{m/2}$.

Assumed: number of robots: two robots; direction of the target: away from the origin, speed $u < 1$; knowledge of the target: d known, u unknown; faults: no faults; speeds: all robots have the same unit speed. **Not assumed:** domain: m rays from a common origin; communication: face-to-face: robots exchange information only when they meet; randomness: deterministic.

Measure	Lower bound	Best upper bound	Status
Competitive ratio	none known No matching lower bound.	$1+O\big(M^{2^{(m+2)/2}}\log^2 M\big)$, $M=\frac{1}{1-u}$ (m even) (Jawhar and Kranakis 2026a) [preprint]	• polynomial gap

Results.

- Jawhar and Kranakis (2026a) [preprint] [current best]: competitive ratio $1+O\big(M^{2^{(m+2)/2}}\log^2 M\big)$, $M=\frac{1}{1-u}$ (m even). Introductory and best (Theorem 2.2); exponent $2^{(m+3)/2}$ for odd m. Preprint.

Claude (Anthropic) was used in cataloguing the problems.

Notes. Draft.

20.4.26 m rays, a sender and a receiver, away, d known

Adds + sender/receiver, + no faults, + equal speeds DRAFT

On m rays, knowing only the initial distance, a sender and a receiver lower the exponent of the face-to-face bound when m is odd.

Assumed: number of robots: two robots; direction of the target: away from the origin, speed $u < 1$; knowledge of the target: d known, u unknown; communication: asymmetric: senders and receivers; faults: no faults; speeds: all robots have the same unit speed. **Not assumed:** domain: m rays from a common origin; randomness: deterministic.

Measure	Lower bound	Best upper bound	Status
Competitive ratio	m (Jawhar and Kranakis 2026a) [preprint] Exponential gap in m.	$O(M) + O\big(M^{2^{(m+1)/2}+1} \log^2 M\big)$, $M = \frac{1}{1-u}$ (m odd) (Jawhar and Kranakis 2026a) [preprint]	• polynomial gap

Results.

- Jawhar and Kranakis (2026a) [preprint] [lower bound]: competitive ratio m. *Bottleneck:* unknown ray. *Holds when:* Lemma 4.1; any deterministic algorithm with two unit-speed robots.
- Jawhar and Kranakis (2026a) [preprint] [current best]: competitive ratio $O(M) + O\big(M^{2^{(m+1)/2}+1} \log^2 M\big)$, $M = \frac{1}{1-u}$ (m odd). Introductory and best for odd m (Algorithm 5, Theorem 3.4). Preprint.

Notes. Draft.

20.4.27 m rays, two face-to-face robots, away, nothing known

Adds nothing known, face-to-face, + no faults, + equal speeds DRAFT

On m rays, knowing neither distance nor speed, two face-to-face robots capture a target moving away with a ratio polynomial in $M = \max\{d, 1/(1-u)\}$, of degree about $2^{m/2}$.

Assumed: number of robots: two robots; direction of the target: away from the origin, speed $u < 1$; faults: no faults; speeds: all robots have the same unit speed. **Not assumed:** domain: m rays from a common origin; knowledge of the target: neither d nor u known; communication: face-to-face: robots exchange information only when they meet; randomness: deterministic.

Measure	Lower bound	Best upper bound	Status
Competitive ratio	none known No matching lower bound.	$1 + O\big(M^{2^{(m+4)/2}} \log^2 M \log\log M\big)$ (m even) (Jawhar and Kranakis 2026a) [preprint]	• polynomial gap

Results.

- Jawhar and Kranakis (2026a) [preprint] [current best]: competitive ratio $1 + O\big(M^{2^{(m+4)/2}} \log^2 M \log\log M\big)$ (m even). Introductory and best (Theorem 2.3), $M = \max\{d, 1/(1-u)\}$; exponent $2^{(m+5)/2}$ for odd m. Preprint.

Notes. Draft.

Claude (Anthropic) was used in cataloguing the problems.

20.4.28 m rays, a sender and a receiver, away, nothing known

Adds nothing known, + sender/receiver, + no faults, + equal speeds DRAFT

On m rays, knowing neither distance nor speed, a sender and a receiver capture a target moving away with a ratio polynomial in $M = \max\{d, 1/(1-u)\}$.

Assumed: number of robots: two robots; direction of the target: away from the origin, speed $u < 1$; communication: asymmetric: senders and receivers; faults: no faults; speeds: all robots have the same unit speed. **Not assumed:** domain: m rays from a common origin; knowledge of the target: neither d nor u known; randomness: deterministic.

Measure	Lower bound	Best upper bound	Status
Competitive ratio	none known No matching lower bound.	$O\left(\frac{1}{1-u}\right) + O\left(\frac{\log\log M}{d(1-u)} \log^2 M\, M^{2^{(m+3)/2}}\right)$ (m odd) (Jawhar and Kranakis 2026a) [preprint]	• polynomial gap

Results.

- Jawhar and Kranakis (2026a) [preprint] [current best]: competitive ratio $O\left(\frac{1}{1-u}\right)+O\left(\frac{\log\log M}{d(1-u)} \log^2 M\, M^{2^{(m+3)/2}}\right)$ (m odd). Introductory and best for odd m (Algorithm 6, Theorem 3.5), $M = \max\{d, 1/(1-u)\}$. Preprint.

Notes. Draft.

SEARCH

20.5 Patrolling

Robots move forever so that no point of a fence (a segment, a circle, or the edges of a graph) stays unvisited for long.

First posed in 2011 (Czyzowicz et al. 2011). Boundary patrolling by agents with distinct speeds (ESA 2011).

Problem statement and measures

k robots with maximum speeds $v_1, \dots, v_k$ move forever along a **fence**: a segment, a circle, or the edges of a graph. The **idle time** is the largest time interval during which some point is not visited. The aim is to minimise it, or, equivalently, to find the longest fence that can be patrolled with idle time 1.

Measures: **Idle time**, worst idle time, or the longest fence patrollable with idle time 1 (stated per variant).

A first look

Security guards with different walking speeds must watch a long straight fence so that no point is left unvisited for too long. In fence patrolling the guards are k **robots** with maximum speeds $v_1, \dots, v_k$ that move forever along the **fence**, here a segment; a point is **visited** when some robot is at it. The **idle time** is the longest stretch of time during which some point of the fence is not visited by any robot. Equivalently, fixing the idle time to 1, we ask for the longest fence, of length ℓ, that the robots can patrol. Robots may pass each other, and the **schedule**, the motion of every robot over time, is planned in advance.

Claude (Anthropic) was used in cataloguing the problems.



(a) Start: piece i has length $v_i/2$

(b) Final: back after time 1; every point visited again

Figure 20.5: Fence patrolling: robots with maximum speeds $v_1, \dots, v_k$ move forever along a segment so as to minimise the idle time, the longest interval during which some point is not visited. The animation runs the partition strategy (Czyzowicz et al. 2011) with speeds $1, 2, 1$: robot i sweeps its own piece, of length $v_i/2$, back and forth at full speed (bars: the part swept on the way out, then on the way back), so every point is visited again within time 1 and a fence of length $(v_1 + \cdots + v_k)/2$ is patrolled; this is not always optimal (Kawamura and Kobayashi 2015).

1. Algorithm: Partition: give each robot its own piece. (Czyzowicz et al. 2011) (Open fence, distinct speeds, p. 572) The **partition** strategy:

1. Cut the fence into k **pieces**, piece i of length $v_i/2$.
2. Robot i walks back and forth over its own piece at full speed, forever.

Why it works and what it costs. A round trip over piece i takes $2 \cdot (v_i/2)/v_i = 1$, so every point of the piece is visited at least once in every unit of time, and the fence has length $\ell = (v_1 + \cdots + v_k)/2$. On the other hand, in any unit of time robot i visits a stretch of at most v_i, so no schedule patrols more than $v_1 + \cdots + v_k$: partition is within a factor 2 of optimal. (Czyzowicz et al. 2011) proved it optimal for $k = 2$ and conjectured it optimal for all k; (Kawamura and Kobayashi 2015) confirmed $k = 3$.

Why it matters. Partition is the natural first answer, and it is used as a building block in most patrolling strategies.

2. Lower bound: Every point needs $f + 1$ visitors: a counting bound. (Czyzowicz et al. 2017e) (Graphs with faulty patrollers, p. 573) Now patrol a **graph**: **nodes** joined by **edges**, where each edge is a segment with a length; the fence is the union of the edges, of total length $|E|$. There are k robots of speed 1, of which up to f are **unreliable**: their visits do not count, and the **adversary**, an opponent who knows the schedule, picks them. Take any **window**, an interval of time of length T equal to the idle time. If some point is visited in that window by only f distinct robots or fewer, the adversary declares those robots unreliable and the point counts as not visited. So every point must be visited by at least $f + 1$ different robots in the window.

Why it holds. Together the robots walk at most kT in the window, and they must cover every point of every edge $f + 1$ times, which needs at least $(f + 1)|E|$. Hence $T \geq (f + 1)|E|/k$. (Czyzowicz et al. 2017e) match it on graphs where a **closed walk** (a route along edges that ends where it starts) uses every edge exactly once: spread the robots evenly along that walk, $|E|/k$ apart, so every point is visited by a new robot every $|E|/k$ time units; in general they use a shortest closed walk through all edges and show that finding the optimum is NP-hard.

Why it matters. Counting total work against what must be covered is the simplest lower-bound technique in patrolling, and it is tight here.

3. Algorithm: Six robots beat the partition. (Kawamura and Kobayashi 2015) (Open fence, distinct speeds, p. 572) Take six robots with speeds $1, 1, 1, 1, 7/3, 1/2$. Partition patrols a fence of length $(1 + 1 + 1 + 1 + 7/3 + 1/2)/2 = 41/12$. (Kawamura and Kobayashi 2015) patrol a longer fence, of length $7/2$:

Claude (Anthropic) was used in cataloguing the problems.

SEARCH

1. The four robots of speed 1 walk end to end, covering most of the fence in a staggered pattern.
2. The fast robot covers most of what they leave uncovered.
3. The slowest robot fills the last small gaps.

Why it works and what it costs. Since $7/2 = 42/12 > 41/12$, the conjecture that partition is always optimal is false; adding further robots shows it fails for every $k \geq 6$. Later schedules patrol fences up to $4/3$ times longer than partition (Kawamura and Soejima 2020), and on a circle simple strategies can be beaten by a factor $21/20$. The optimal schedule for general speeds is unknown.

Why it matters. It shows that sharing pieces of the fence can beat dividing it, which is why even this one-dimensional problem is still open.

What the community started looking at

Patrolling asked first for **optimal strategies** and whether the simple ones, partitioning the fence or cycling around it, are optimal (Czyzowicz et al. 2011). They are not in general (Kawamura and Kobayashi 2015; Kawamura and Soejima 2020): optimal schedules patrol a segment almost twice as long as partition, and a one-way circle unboundedly longer than the cyclic strategy (Haeupler et al. 2019); faults later changed the question to the idle time a graph admits when up to f patrollers are unreliable (Czyzowicz et al. 2017e).

Further reading

Papers on this problem that are not yet placed in a setting.

- Das et al. (2019c). Patrolling 1-interval-connected rings: with visibility limited to the current node the idle time is $n - \alpha k$, only an additive gain over one agent (Section 4); with a global snapshot, upper and lower bounds that differ by small constant factors for 2 agents and for $k > 2$ agents, and a separation between agents that know the schedule of the ring and agents that do not. Read (arXiv v1).
- Chuangpishit et al. (2018a). Points on a path, each with its own required frequency of visits, patrolled by two robots: approximation algorithms, the best with ratio $\sqrt{3}$. Read (arXiv v1).
- Morales-Ponce (2022). A unit segment whose high-priority parts must be revisited as often as possible while the rest is visited infinitely often, by k unit-speed robots: a lower bound on the idle time from lid covers of the high-priority parts, computed in $O(\max(k, n) \log n)$ time for n parts, and a schedule that attains it. Read (arXiv v1).
- Georgiou et al. (2025a). One unit-speed robot patrols the three edges of an acute triangle so that the longest time between two visits of an edge is minimised: the orthic triangle, the classical answer to Fagnano's problem, is optimal, and billiard trajectories solve a related 2-gap problem. Read (arXiv v6).
- Amir et al. (2023). One agent patrols a d-dimensional grid graph with little memory: some grids cannot be patrolled with 0 bits whatever the sensing range, an exact characterisation of the grids patrollable with 0 bits and range V, and a 1-bit algorithm with range 1 for every grid. Read (arXiv v2).
- Afshani et al. (2021). Patrol scheduling with weighted latency: k robots of equal speed visit n sites of a metric space to minimise the largest weighted time between visits of a site. A polynomial $O(k^2 \log \frac{w_{\max}}{w_{\min}})$-approximation; on a line, an exact algorithm for equal weights and a 12-approximation for weights. Read (arXiv v3).
- Chen et al. (2024). The same patrol scheduling on a weighted metric graph: a $5h$-approximation for one robot (Theorem 1) and an $O(kh)$-approximation for k robots (Theorem 2), with $h = \log_2 \frac{w_{\max}}{w_{\min}} + 1$, improving on (Afshani et al. 2021). Read (HAL version).
- Oshrat et al. (2020). Adversarial fence patrolling: non-uniform probabilistic policies along an open polyline raise the minimal probability of detecting a penetration above uniform ones (AAAI). Abstract read.

Claude (Anthropic) was used in cataloguing the problems.

- Coleman and Morales-Ponce (2023). The snow plow problem: snow falls continuously on the segment $[0, 1]$, and k robots of equal speed (their n agents) must plow every point infinitely often and carry what they collect to a dump point. With the dump at an endpoint and uniform snowfall, giving each robot one interval so that all carry the same largest load is optimal and is computed in $O(k)$ time (Theorem 3.3); for costs given by a travel function and a snowfall function, a $(1+\varepsilon)$-approximation in $O(k \log(1/\varepsilon))$ time (Theorems 3.7, 4.1), exact in $O(k)$ time when both are linear or constant (Theorem 3.8). Read in full.

Assumptions specific to this problem

Fence

Patrolling asks agents to visit every point of a terrain often enough; the measure is the idle time, the longest time a point is left unvisited (Kawamura and Kobayashi 2015). Czyzowicz, Gąsieniec, Kosowski and Kranakis proposed the open fence (a segment) and the closed fence (a circle) for agents with distinct speeds at ESA 2011, and conjectured that simple strategies are optimal on both: partition on the segment, and on the circle the cyclic strategy for robots that all move in one direction; on the circle the same paper shows that for three robots moving in both directions neither simple strategy is optimal (Czyzowicz et al. 2011). Kawamura and Kobayashi disproved the segment conjecture (Kawamura and Kobayashi 2015), Dumitrescu, Ghosh and Tóth disproved the one-way circle conjecture and showed that on the two-way circle a new strategy beats both simple ones by a factor approaching $3/2$ (Dumitrescu et al. 2014), and Kawamura and Soejima showed that optimal schedules can beat the simple strategies by factors $4/3$ on the segment and $21/20$ on the circle with every robot moving in one direction (Kawamura and Soejima 2020). They note that "finding an optimal strategy is not at all straightforward, even when the terrain is as simple as it could be" (Kawamura and Soejima 2020). Graphs, the most general domain, were studied by Czyzowicz et al. with unreliable robots: the domain is "represented by all points belonging to the graph edges", and edges are rectifiable curves whose lengths are their weights (Czyzowicz et al. 2017e).

Level	Value	Formal definition
0	**graph**	Every point of every edge of a graph G must be visited; robots move along edges.
1	**segment**	An open fence: the segment $[0, \ell]$; robots may reverse direction anywhere.
1	**circle**	A closed fence: a circle of length ℓ.

Direction of movement

Czyzowicz, Gąsieniec, Kosowski and Kranakis first studied the circle with every robot moving in the same direction, since "in order to profit from the circle topology it may seem natural to try to traverse the cycle always in the same direction", and then asked whether allowing both directions helps: it does not for two robots, but it does for some speeds of three (Czyzowicz et al. 2011). Dumitrescu, Ghosh and Tóth studied both versions, as the unidirectional and the bidirectional unit circle (Dumitrescu et al. 2014), and Kawamura and Soejima call the first the (unidirectional) circle patrolling problem (Kawamura and Soejima 2020). The scale is one-way < two-way: a one-way schedule is also a two-way schedule.

Claude (Anthropic) was used in cataloguing the problems.

SEARCH

Level	Value	Formal definition
0	**one-way**	On a closed fence every robot moves only counterclockwise, at any speed between 0 and its maximum speed v_i; it never reverses. Czyzowicz et al. require strictly positive speeds, while Dumitrescu, Ghosh and Tóth and Kawamura and Soejima allow robots to stop; the counterexample of Dumitrescu et al. holds under both conventions (Remark after Theorem 1 of Dumitrescu et al. 2014). Schedules are a subset of the two-way schedules, so this is the harder level for the robots.
1	**two-way**	Every robot moves along the fence in either direction at any speed between 0 and v_i, and may reverse at any point and any time. On a segment and on a graph this is the only meaningful choice.

Case tree

The root assumes nothing; going down, each edge adds the assumptions written in grey (+) or leaves an axis unassumed. Bold entries are catalogued settings with their status per measure (Idle time) and page; italic entries are branch points, named after the axis on which their branches differ. Axis order: Fence → Direction of movement → Faults → Speeds → Communication → Randomness.

No assumptions

- + circle
  - *Direction of movement*
    - one-way, + no faults
      - **Closed fence, one-way, distinct speeds** ● p. 571
    - + two-way, + no faults
      - **Closed fence, two-way, distinct speeds** ● p. 572
- + segment, + two-way, + no faults
  - **Open fence, distinct speeds** STANDARD SETTING ● p. 572
- graph, + two-way, + crash faults, + equal speeds
  - **Graphs with faulty patrollers** ● p. 573

What is known

The best upper bound against the best lower bound in each catalogued setting, in tree order. The settings follow in detail.

Setting	Idle time	p.
Closed fence, one-way, distinct speeds	● near-optimal $\frac{1}{33 \ln \log_2 k} \sum_i v_i$ vs $L \le \sum_i v_i$	571
Closed fence, two-way, distinct speeds	● unknown → $\frac{3}{2}$ × simple vs —	572
Open fence, distinct speeds STANDARD SETTING	● tight → 2 × partition vs *partition is not optimal*	572
Graphs with faulty patrollers	● tight $(f+1)\,\mathrm{CPT}(G)/k$ vs $(f+1)\lvert E\rvert/k$	573

Claude (Anthropic) was used in cataloguing the problems.

20.5.1 Closed fence, one-way, distinct speeds

Adds one-way, + no faults

VERIFIED

With every robot moving in one direction, the cyclic strategy is optimal for up to four robots but not within any constant factor in general: every set of robots patrols a circle of length $\sum_i v_i/(33 \ln \log_2 k)$.

Assumed: fence: a circle (closed fence); faults: no faults. **Not assumed:** direction of movement: one-way: every robot moves in the same fixed direction; speeds: robots may have different maximum speeds; communication: face-to-face: robots exchange information only when they meet; randomness: deterministic.

Measure	Lower bound	Best upper bound	Status
Idle time	$L \le \sum_i v_i$ (Czyzowicz et al. 2011)	$\frac{1}{33 \ln \log_2 k} \sum_i v_i$ (Haeupler et al. 2019)	• near-optimal
	For every set of speeds, within a factor $O(\log \log k)$ of the trivial bound $\sum_i v_i$; the ratio to the cyclic strategy is unbounded.		

Results.

- Czyzowicz et al. (2011) [lower bound]: idle time $L \le \sum_i v_i$. *Bottleneck:* space-time coverage. *Holds when:* Proposition 2: no schedule has idle time below $1/(v_1 + \cdots + v_k)$ on the unit circle, in either direction.

- Czyzowicz et al. (2011) [introductory]: idle time cyclic optimal for $k \le 4$. The cyclic strategy (the r fastest robots, equally spaced, all at speed v_r, with r maximising iv_i) has idle time $1/\max_i iv_i$ on the unit circle (Theorem 2); it is optimal for $k \le 4$ (Theorem 3) and conjectured optimal for every k (Conjecture 2).

- Dumitrescu et al. (2014) [also]: idle time *cyclic not optimal.* Theorem 1: 32 robots with speeds $v_i = 1/i$ have a periodic schedule with idle time strictly below 1, the idle time of the cyclic strategy; this disproves Conjecture 2 of (Czyzowicz et al. 2011), also when speeds must stay positive (Remark). Theorem 2: for every $\tau \in (0, 1]$ and horizon $t \ge \tau$, $k \le e^{4t/\tau^2}$ robots with speeds $1/i$ keep the idle time at most τ during $[0, t]$.

- Kawamura and Soejima (2020) [also]: idle time $\frac{21}{20}\times$ simple. Theorem 10: speeds and a schedule that patrol a circle of length $1.05 \max_r rv_r$ with idle time 1. Kawamura and Soejima conjecture that the cyclic strategy is not within any constant factor of optimal.

- Haeupler et al. (2019) [current best]: idle time $\frac{1}{33 \ln \log_2 k} \sum_i v_i$. Theorem 2.4: for large k and any speeds, a schedule with idle time 1 on a one-way circle of this length. With speeds $1/i$ the cyclic strategy patrols length 1 and this schedule $\Theta(\log k / \log \log k)$, so the cyclic strategy is not within a constant factor of optimal, as Kawamura and Soejima conjectured.

Open questions.

Q1. Close the factor $O(\log \log k)$ between the schedule of (Haeupler et al. 2019) and the bound $\sum_i v_i$ on the one-way circle.

Notes. Conjecture 2 of (Czyzowicz et al. 2011) was disproved by Dumitrescu, Ghosh and Tóth (Theorem 1 of Dumitrescu et al. 2014); (Kawamura and Soejima 2020) then gave a quantitative gap of 21/20, and (Haeupler et al. 2019) an unbounded one. Theorem 2 of (Dumitrescu et al. 2014) suggests that the ratio to the cyclic strategy is unbounded, but it holds only over a finite horizon. Theorem 3 of (Czyzowicz et al. 2011) (optimality of the cyclic strategy for $k \le 4$) is proved for strictly positive speeds; whether it survives when robots may stop was not checked. A lower bound valid for both directions: no schedule has idle time below $1/(v_1 + \cdots + v_k)$ on the unit circle (Proposition 2 of Czyzowicz et al. 2011). Theorem 2.4 of (Haeupler et al. 2019) was checked against its arXiv v2.

Claude (Anthropic) was used in cataloguing the problems.

SEARCH

20.5.2 Closed fence, two-way, distinct speeds

Adds + two-way, + no faults DRAFT

If robots may reverse, two robots gain nothing over the cyclic strategy, but for suitable speeds a train of slow robots with one fast robot shuttling along it beats both simple strategies by a factor approaching 3/2.

Assumed: fence: a circle (closed fence); direction of movement: two-way: robots may reverse direction at any time; faults: no faults. **Not assumed:** speeds: robots may have different maximum speeds; communication: face-to-face: robots exchange information only when they meet; randomness: deterministic.

Measure	Lower bound	Best upper bound	Status
Idle time	none known	$\to \frac{3}{2} \times$ simple (Dumitrescu et al. 2014)	• unknown
	The largest ratio between optimal schedules and the simple strategies is unknown; it tends to at least 3/2 for large k (Dumitrescu et al. 2014) and is at most 2.		

Results.

- Czyzowicz et al. (2011) [introductory]: idle time $\frac{36}{35} \times$ simple. For $k = 2$ no schedule moving in both directions beats the cyclic strategy (Theorem 4). For three robots with speeds 1, $\frac{1}{2}$, $\frac{1}{3}$ a schedule in which some robots reverse has idle time $\frac{35}{36}$, below the idle time 1 of the cyclic strategy and of the partition (Theorem 5; the conference version omits the analysis and gives the schedule as a figure). Dumitrescu, Ghosh and Tóth restate these idle times (35/36 against 12/11 for the partition and 1 for the cyclic strategy) (Dumitrescu et al. 2014).

- Dumitrescu et al. (2014) [current best]: idle time $\to \frac{3}{2} \times$ simple. Proposition 1 (train algorithm A_3): robots $a_2, \dots, a_k$ equally spaced move one way at speed v_k, and the fastest robot a_1 shuttles back and forth along the arc between the end and the start of the train. With $v_1 = 1$ and $v_2 = \cdots = v_k = 1/k$ its idle time tends to $2/3$, against 1 for the partition and the cyclic strategy; for every $k \geq 4$ some speeds make it beat both. Proposition 2: for every $k \geq 2$ some speeds have an optimal schedule that leaves up to $k - 1$ robots unused (on a segment every robot is needed).

Open questions.

Q1. What is the largest ratio between optimal schedules and the best simple strategy on the two-way circle? It lies between 3/2 (Dumitrescu et al. 2014) and 2, since partition patrols $\sum_i v_i/2$ and no schedule patrols more than $\sum_i v_i$ (Proposition 2 of Czyzowicz et al. 2011).

Notes. The analysis of Theorem 5 of (Czyzowicz et al. 2011) is omitted in the conference version, so its 35/36 is not checked; the best result, Proposition 1 of (Dumitrescu et al. 2014), was checked with its proof. The setting stays a draft only because of Theorem 5.

20.5.3 Open fence, distinct speeds

Adds + segment, + two-way, + no faults STANDARD SETTING VERIFIED

Partitioning the segment among the robots is optimal for 2 and 3 robots but not in general: optimal schedules can patrol a segment almost, but never quite, twice as long.

Assumed: fence: a segment (open fence); direction of movement: two-way: robots may reverse direction at any time; faults: no faults. **Not assumed:** speeds: robots may have different maximum speeds; communication: face-to-face: robots exchange information only when they meet; randomness: deterministic.

Claude (Anthropic) was used in cataloguing the problems.

Measure	Lower bound	Best upper bound	Status
Idle time	*partition is not optimal* (Kawamura and Kobayashi 2015); $L \le \sum_i \frac{v_i}{1+v_i/v_{\max}} < 2\times$partition (Haeupler et al. 2019)	$\to 2\times$ partition (Haeupler et al. 2019)	• tight
	The largest ratio between optimal schedules and partition tends to 2 and is never reached (Haeupler et al. 2019).		

Results.

- Kawamura and Kobayashi (2015) [lower bound]: idle time *partition is not optimal. Bottleneck:* distinct speeds. *Holds when:* counterexample with $k = 6$ agents.
- Haeupler et al. (2019) [lower bound]: idle time $L \le \sum_i \frac{v_i}{1+v_i/v_{\max}} < 2\times$ partition. *Bottleneck:* space-time coverage. *Holds when:* Theorem 2.1: the longest segment patrolled with idle time 1, where $v_{\max}$ is the largest speed; with equal speeds it gives the partition's $\sum_i v_i/2$; for k robots the efficiency $L/\sum_i v_i$ is at most $1 - 1/\sqrt{k+1}$ (Lemma 2.2).
- Czyzowicz et al. (2011) [introductory]: idle time partition optimal for $k = 2$. Introduces patrolling by robots with distinct speeds; shows the partition strategy optimal for $k = 2$ (Theorem 1) and conjectures it optimal for every k (Conjecture 1); (Kawamura and Kobayashi 2015) disproved the conjecture.
- Kawamura and Kobayashi (2015) [also]: idle time partition optimal for $k = 2, 3$.
- Dumitrescu et al. (2014) [also]: idle time $\to \frac{25}{24}\times$ partition. Theorem 3: for every integer $x \ge 2$, $k = 4x + 1$ robots with $v_1 + \cdots + v_k = 16x + 1$ patrol a segment of length $25x/3$ with idle time 1, against $(16x + 1)/2$ for the partition; this improved the counterexample of (Kawamura and Kobayashi 2015) and is superseded by the 4/3 of (Kawamura and Soejima 2020).
- Kawamura and Soejima (2020) [also]: idle time $\frac{4}{3}\times$ partition. Schedules that patrol a segment 4/3 times longer than partition (efficiency approaching 2/3); Kawamura and Soejima conjectured that 2/3 is the best efficiency, which (Haeupler et al. 2019) disproved.
- Haeupler et al. (2019) [current best]: idle time $\to 2\times$ partition. Theorem 2.3: for large k, $k - 1$ robots of speed 1 and one of speed $\Theta(\sqrt{k})$ patrol a segment with efficiency $1 - 3.5/\sqrt{k}$, where partition has efficiency 1/2. By Theorem 2.1 and Lemma 2.2, efficiency $1 - \varepsilon$ needs $\Omega(\varepsilon^{-2})$ robots and a speed ratio $\Omega(\varepsilon^{-1})$, so the construction is asymptotically optimal.

Open questions.

Q1. Is computing an optimal schedule for given speeds on a segment hard?

Notes. Czyzowicz et al. asked how much better than partition an optimal schedule can be (Czyzowicz et al. 2011); Haeupler, Kuhn, Martinsson, Petrova and Pfister answered it (Haeupler et al. 2019). Their bound of Theorem 2.1 is attained by the constructions of (Dumitrescu et al. 2014) and (Kawamura and Soejima 2020). Checked against the full text of (Haeupler et al. 2019) (arXiv v2).

20.5.4 Graphs with faulty patrollers

Adds graph, + two-way, + crash faults, + equal speeds VERIFIED

With at most f unreliable robots among k, the optimal idle time lies between $(f + 1)|E|/k$ and $(f + 1)\,\mathrm{CPT}(G)/k$, which coincide on Eulerian graphs; computing it is NP-hard.

Assumed: direction of movement: two-way: robots may reverse direction at any time; faults: up to f crash-faulty robots (fail to report the target); speeds: all robots have the same unit speed. **Not assumed:** fence: the edges of a graph; communication: face-to-face: robots exchange information only when they meet; randomness: deterministic.

Claude (Anthropic) was used in cataloguing the problems.

SEARCH

Statement for this setting. k unit-speed robots move perpetually along the edges of a connected graph G whose edges are curves of given lengths; $|E|$ is the total length. Knowing the strategy, an adversary chooses a set of f faulty robots, whose visits do not count, a point p and a time t. The idle time of a strategy is the supremum, over these choices, of the length of an interval starting at t during which no reliable robot visits p; the goal is to minimise it.

Measure	Lower bound	Best upper bound	Status
Idle time	$(f+1)\lvert E\rvert/k$ (Czyzowicz et al. 2017e); *NP-hard* (Czyzowicz et al. 2017e)	$(f+1)\,\mathrm{CPT}(G)/k$ (Czyzowicz et al. 2017e)	• tight
	Exact on Eulerian graphs; within a factor $\mathrm{CPT}(G)/\lvert E\rvert \le 2$ in general.		

Results.

- Czyzowicz et al. (2017e) [lower bound]: idle time $(f+1)|E|/k$. *Bottleneck:* faulty reports. *Holds when:* any connected graph, $k \ge 2$, $f \le k-1$; every point must be passed by at least $f+1$ robots within each idle window, else the adversary makes them all faulty. Theorem 3 (lower bound, via Lemma 1).
- Czyzowicz et al. (2017e) [lower bound]: idle time *NP-hard*. *Bottleneck:* combinatorial hardness. *Holds when:* $k = 3$ robots, at most one faulty; graphs with unit-length edges; reduction from 3-edge-colouring of cubic graphs. Theorem 4: deciding whether the idle time equals $\frac{2}{3}|E|$ is NP-hard.
- Czyzowicz et al. (2017e) [current best]: idle time $(f+1)\,\mathrm{CPT}(G)/k$. Introductory and current best: robots equally spaced along a Chinese Postman Tour (Theorem 3); equals $(f+1)|E|/k$ on Eulerian graphs (Corollary 1). For a segment with $f < k/2-1$ the paper gives a three-group strategy that is optimal for odd f and almost optimal for even f (Theorems 1 and 2).

Open questions.

Q1. Is the decision version of fault-tolerant patrolling in NP? With k part of the input and integer edge lengths it is in PSPACE (Czyzowicz et al. 2017e).

Q2. Optimal fault-tolerant patrolling of a segment when $f \ge k/2-1$, and for even f (Czyzowicz et al. 2017e).

Q3. Approximation algorithms for the optimal idle time on general graphs (Czyzowicz et al. 2017e).

Notes. Faulty here means a robot whose visits do not count; the adversary chooses the faulty set knowing the strategy. $\mathrm{CPT}(G)$ is the length of a Chinese Postman Tour of G. Kotzig graphs (cubic graphs whose three perfect matchings pairwise form Hamiltonian cycles) attain the minimum idle time $\frac{2}{3}|E|$, the number of their vertices, for $k = 3$, $f = 1$ (Corollary 2). The problem was posed at a MITACS workshop in 2012. Checked against the full text of (Czyzowicz et al. 2017e).

20.6 Search with segment visibility

Robots search the plane for a point that they see when it lies on the segment joining two of them.

First posed in 2026 (Kranakis 2026). Kranakis introduced segment visibility in a preprint of November 2026 (Kranakis 2026); the earlier uses this note rests on are the author's own account.

Problem statement and measures

$k \ge 2$ robots of unit speed start together at the origin O of the plane, share a coordinate frame and a clock, and move deterministically. A stationary target lies at an unknown point $x \ne O$ at distance $d = \|x\| > 0$.

Claude (Anthropic) was used in cataloguing the problems.

The target is found at the first time t at which x lies on the closed segment joining the positions of two robots at time t; robots learn nothing about the target before. Depending on the setting the robots know d or not; they never know the direction of x. **Time** is the competitive ratio: the supremum, over targets, of the time until the target is found divided by d.

Measures: **Competitive ratio**, worst-case time until the target lies on a segment between two robots, divided by its distance d from the origin.

A first look

Two or more lifeguards scan the sea for a buoy, and a buoy counts as seen as soon as it lies on the line of sight between two of them. Here $k \geq 2$ **robots**, points of unit speed, start together at the **origin** O of the plane. A **target**, a point x at distance d from the origin, sits in an unknown direction. At every moment each pair of robots sees the straight **segment** that joins them, and the target is found the first time it lies on such a segment. The robots share a coordinate frame and a clock, and their routes are fixed in advance. The cost of a route plan is its **competitive ratio**: the largest value, over all targets, of the time until the target is found divided by d. In the basic version two robots do not know d.

1. Algorithm: Walk out in k equally spaced directions. (Kranakis 2026) ($k \geq 3$ robots, unknown distance, p. 579) For $k \geq 3$ robots:

1. Number the robots $0, 1, \dots, k-1$.
2. Robot i walks straight away from the origin in the direction at angle $2\pi i/k$, forever.
3. At time t the robots are the corners of a regular k-gon of circumradius t; its sides are segments between neighbouring robots, and they sweep the plane as the polygon grows.

Why it works and what it costs. A target at distance d whose direction is at angle φ from the middle of its sector lies on a side of the polygon when $t\cos(\pi/k) = d\cos\varphi$, that is at time $d\cos\varphi/\cos(\pi/k)$. The worst targets are in the middle of a sector, so the competitive ratio is $1/\cos(\pi/k) = \sec(\pi/k)$: 2 for three robots, about 1.41 for four, and close to 1 for many robots. (Kranakis 2026) proves that no plan does better, whether or not the robots know d.

Why it matters. It is optimal and needs no knowledge of d; the same walk is optimal for robots looking for a line.

2. Lower bound: Seeing a point is at least as hard as touching its tangent line. (Kranakis 2026) (Two robots, known distance, p. 578) Take a target x at distance d and the straight line through x that is perpendicular to the direction of x (the tangent at x to the circle of radius d about the origin). The robots start on the origin's side of that line. When x lies on a segment between two robots, one of the two must be on the line or beyond it.

Why it holds. So a plan that finds every target point of the circle within time Td also makes some robot touch every tangent line within time Td: it solves **shoreline search**, the search for a line at known distance. The best ratio for two robots that look for a line at known distance is $1 + \sqrt{3} + \pi/6 \approx 3.256$ (Dobrev et al. 2020b), so no two-robot plan for points does better. (Kranakis 2026) gives a two-robot route that reaches exactly this ratio, so the bound is tight. When d is unknown, the best known two-robot plan, two spirals on opposite sides of the origin, needs about 8.93, and the best lower bound is about 3.29.

Why it matters. The reduction lets every lower bound for line search transfer to point search with segment visibility, which settles all cases except two robots that do not know d.

Claude (Anthropic) was used in cataloguing the problems.

What the community started looking at

Search for a target at an unknown place in the plane goes back to Bellman's question of how to find a line at known distance, and to the search for a line or a point by one or several robots of unit speed (Baeza-Yates et al. 1993; Dobrev et al. 2020b). In those models a robot finds the target by touching it. Kranakis lets the robots see along the segments that join them, so a team sees a whole set of chords at each moment, and studies how much this helps; the answers are expressed through shoreline search, whose lower bounds carry over (Kranakis 2026).

Assumptions specific to this problem

Number of robots

Kranakis treats two robots separately because with two robots there is a single visible segment and the expanding-polygon algorithm does not exist; for $k \geq 3$ the optimum is $\sec(\pi/k)$ (Kranakis 2026). The same split appears in shoreline search, where the cases $k = 2$ and $k = 3$ were the last to be settled for a known distance (Dobrev et al. 2020b).

Level	Value	Formal definition
0	**any k**	$k \geq 2$ robots; bounds are stated as functions of k.
1	**two robots**	Two robots; the visible set at each time is the one segment between them.

Knowledge of the distance

Shoreline search was first studied at a known distance (Bellman's problem, solved for one searcher by Isbell, as reported by Dobrev et al. 2020b); Baeza-Yates, Culberson and Rawlins considered the unknown distance (Baeza-Yates et al. 1993) (as reported by Dobrev et al. 2020b). Kranakis distinguishes the same two regimes for segment visibility and shows that for two robots not knowing d costs at least $\sqrt{17}/4 - 1 \approx 0.031$ in the ratio, while for $k \geq 3$ it costs nothing (Kranakis 2026).

Level	Value	Formal definition
0	**d unknown**	The trajectories do not depend on d; the ratio is the supremum over all targets $x \neq O$ of $T(x)/\|x\|$.
1	**d known**	The robots know d (not the direction of the target); the trajectories may depend on d, and the ratio is the supremum over targets with $\|x\| = d$ of $T(x)/d$.

Case tree

The root assumes nothing; going down, each edge adds the assumptions written in grey (+) or leaves an axis unassumed. Bold entries are catalogued settings with their status per measure (Competitive ratio) and page; italic entries are branch points, named after the axis on which their branches differ. Axis order: Number of robots → Knowledge of the distance → Faults → Speeds → Randomness.

No assumptions

+ two robots

Knowledge of the distance

d unknown, + no faults, + equal speeds

Two robots, unknown distance STANDARD SETTING • p. 577

Claude (Anthropic) was used in cataloguing the problems.

+ d known, + no faults, + equal speeds
Two robots, known distance • p. 578

any k, d unknown, + no faults, + equal speeds
$k \geq 3$ **robots, unknown distance** • p. 579

any k, + d known, + no faults, + equal speeds
$k \geq 3$ **robots, known distance** • p. 579

What is known

The best upper bound against the best lower bound in each catalogued setting, in tree order. The settings follow in detail.

Setting	Competitive ratio	p.
Two robots, unknown distance STANDARD SETTING	• polynomial gap $e^{\pi\lambda^*}\sqrt{1+\lambda^{*2}}/\lambda^* \approx 8.9285$ vs $\sqrt{3}+\pi/6+\sqrt{17}/4 \approx 3.2864$	577
Two robots, known distance	• tight $1+\sqrt{3}+\pi/6$ vs $1+\sqrt{3}+\pi/6$	578
$k \geq 3$ robots, unknown distance	• tight $\sec(\pi/k)$ vs $\sec(\pi/k)$	579
$k \geq 3$ robots, known distance	• tight $\sec(\pi/k)$ vs $\sec(\pi/k)$	579

20.6.1 Two robots, unknown distance

Adds d unknown, + no faults, + equal speeds — STANDARD SETTING VERIFIED

Two robots that do not know d need a ratio between $\sqrt{3}+\pi/6+\sqrt{17}/4 \approx 3.2864$ and ≈ 8.9285; the gap is open (a preprint).

Assumed: number of robots: two robots; faults: no faults; speeds: all robots have the same unit speed. **Not assumed:** knowledge of the distance: d unknown; randomness: deterministic.

Statement for this setting. Two robots; the trajectories do not depend on d, and every $d > 0$ is possible.

Measure	Lower bound	Best upper bound	Status
Competitive ratio	$\sqrt{3}+\pi/6+\sqrt{17}/4 \approx 3.2864$ (Kranakis 2026) [preprint] ≈ 8.9285 against ≈ 3.2864: a constant gap, factor about 2.7.	$e^{\pi\lambda^*}\sqrt{1+\lambda^{*2}}/\lambda^* \approx 8.9285$ (Kranakis 2026) [preprint]	• polynomial gap

Results.

- Kranakis (2026) [preprint] [lower bound]: competitive ratio $\sqrt{3}+\pi/6+\sqrt{17}/4 \approx 3.2864$. *Bottleneck:* unknown distance. *Holds when:* deterministic; targets on the three circles of radii $1/8$, $1/4$ and 1 about the origin; holds even if the robots know these three radii; any two 1-Lipschitz trajectories (no symmetry assumed); scale-invariant: also with circles of radii 1, 2, 8, so it holds when $d \geq 1$ is promised. Theorem 3.2 of (Kranakis 2026). Below this ratio, the robot that first reaches the unit circle stays within distance $1/8$ of a line through the origin until then (Lemma 3.7), so the other robot alone must reach the levels $\pm 1/8$, $\pm 1/4$ on the perpendicular line, which forces ratio 4 (Lemma 3.4). It exceeds the known-distance optimum $1+\sqrt{3}+\pi/6$ by $\sqrt{17}/4 - 1 \approx 0.031$.
- Kranakis (2026) [preprint] [current best]: competitive ratio $e^{\pi\lambda^*}\sqrt{1+\lambda^{*2}}/\lambda^* \approx 8.9285$. *Assumes:* common coordinate frame and clock; exact spirals from the origin, with infinitely many turns as $t \to 0$. Introductory and current best (Theorem 3.1): the robots follow two antipodal logarithmic spirals, so the visible segment is a diameter

Claude (Anthropic) was used in cataloguing the problems.

turning about the origin while it grows; ratio $e^{\pi\lambda}\sqrt{1+\lambda^2}/\lambda$, minimised at $\pi\lambda^*(1+\lambda^{*2}) = 1$, $\lambda^* \approx 0.2931$. Optimal only within this family of spirals.

Open questions.

Q1. Determine the optimal competitive ratio of two robots that do not know d: it lies between $\sqrt{3} + \pi/6 + \sqrt{17}/4 \approx 3.2864$ and ≈ 8.9285 (Kranakis 2026).

Q2. Do asymmetric, non-antipodal or non-spiral trajectories beat ≈ 8.9285? The antipodal spirals see only along a diameter through the origin, while a segment between two moving robots can sweep chords that miss it (Kranakis 2026).

Notes. From the preprint (Kranakis 2026), read in full (arXiv v1). The known-distance optimum $1 + \sqrt{3} + \pi/6$ is a lower bound here as well; the point of Theorem 3.2 is that the unknown distance costs strictly more. With a promised lower bound on d the spiral needs a finite start, whose ratio the paper does not analyse. For comparison, two robots looking for a line (shoreline search) at unknown distance reach about 5.2644 with two equally spaced logarithmic spirals (Baeza-Yates and Schott, as reported by Dobrev et al. 2020b); a line is touched no later than its tangency point is seen, so that bound does not carry over to segment search.

20.6.2 Two robots, known distance

Adds + d known, + no faults, + equal speeds VERIFIED

Two robots that know d need exactly $1 + \sqrt{3} + \pi/6 \approx 3.2556$, the same ratio as two robots looking for a line at known distance.

Assumed: number of robots: two robots; knowledge of the distance: d known; faults: no faults; speeds: all robots have the same unit speed. **Not assumed:** randomness: deterministic.

Statement for this setting. Two robots that know d (not the direction of the target); the targets are the points of the circle of radius d about the origin.

Measure	Lower bound	Best upper bound	Status
Competitive ratio	$1 + \sqrt{3} + \pi/6$ (Kranakis 2026) [preprint] Exactly $1 + \sqrt{3} + \pi/6 \approx 3.2556$.	$1 + \sqrt{3} + \pi/6$ (Kranakis 2026) [preprint]	• tight

Results.

- Kranakis (2026) [preprint] [lower bound]: competitive ratio $1 + \sqrt{3} + \pi/6$. *Bottleneck:* unknown direction. *Holds when:* deterministic; targets on the circle of radius d, d known; any two 1-Lipschitz trajectories; from shoreline search at known distance (Theorem 3 of Dobrev et al. 2020b): a robot touches the line tangent at the target no later than the target is seen. Theorem 2.1 with Lemma 2.5 of (Kranakis 2026).
- Kranakis (2026) [preprint] [current best]: competitive ratio $1 + \sqrt{3} + \pi/6$. *Assumes:* common coordinate frame and clock. Introductory and current best (Theorem 2.1, Proposition 2.3): robot $\sigma \in \{\pm 1\}$ walks straight to $(\sigma d/\sqrt{3}, d)$, along the tangent to $(\sigma\sqrt{3}d/2, d/2)$, along the circle to $(\sigma d, 0)$, then straight down to $(\sigma d, -d)$; the last point found is $(0, -d)$.

Notes. From the preprint (Kranakis 2026), read in full (arXiv v1). The upper bound is proved directly; the lower bound is Theorem 3 of (Dobrev et al. 2020b) for shoreline search, transferred by the tangent-line reduction.

Claude (Anthropic) was used in cataloguing the problems.

20.6.3 $k \geq 3$ robots, unknown distance

Adds any k, d unknown, + no faults, + equal speeds VERIFIED

With $k \geq 3$ robots the optimal ratio is exactly $\sec(\pi/k)$: the robots walk straight in k equally spaced directions and the edges of the growing regular k-gon sweep the plane.

Assumed: faults: no faults; speeds: all robots have the same unit speed. **Not assumed:** number of robots: any number $k \geq 2$ of robots; knowledge of the distance: d unknown; randomness: deterministic.

Statement for this setting. $k \geq 3$ robots that do not know d; every segment between two robots is visible (the algorithm uses only segments between neighbours on the k-gon). For $k = 2$ see the two-robot settings.

Measure	Lower bound	Best upper bound	Status
Competitive ratio	$\sec(\pi/k)$ (Kranakis 2026) [preprint] Exactly $\sec(\pi/k)$ for every $k \geq 3$.	$\sec(\pi/k)$ (Kranakis 2026) [preprint]	• tight

Results.

- Kranakis (2026) [preprint] [lower bound]: competitive ratio $\sec(\pi/k)$. *Bottleneck:* unknown direction. *Holds when:* deterministic; $k \geq 3$; holds even when every segment between two robots is visible; from shoreline search at unknown distance: Theorem 2.1 of (Acharjee et al. 2019) for $k \geq 4$, Theorem 1 of (Dobrev et al. 2020b) for $k = 3$. Theorem 4.1 with Lemma 4.4 of (Kranakis 2026).
- Kranakis (2026) [preprint] [current best]: competitive ratio $\sec(\pi/k)$. *Assumes:* common coordinate frame and clock; only the k segments between cyclic neighbours are monitored. Introductory and current best (Theorem 4.1, Lemma 4.2): robot i walks straight in direction $2\pi i/k$; a target at angle φ from the bisector of its sector is found at time $d\cos\varphi/\cos(\pi/k)$, worst on the bisectors. Does not use d.

Notes. From the preprint (Kranakis 2026), read in full (arXiv v1). The ratio is the same as the optimum of shoreline search with $k \geq 3$ robots, whose upper bound $1/\cos(\pi/k)$ is the same walk along k equally spaced rays (Baeza-Yates and Schott, as reported by Dobrev et al. 2020b): with three or more robots, seeing a point on a segment costs no more than touching its tangent line.

20.6.4 $k \geq 3$ robots, known distance

Adds any k, + d known, + no faults, + equal speeds VERIFIED

Knowing d does not help $k \geq 3$ robots: the optimal ratio is still exactly $\sec(\pi/k)$.

Assumed: knowledge of the distance: d known; faults: no faults; speeds: all robots have the same unit speed. **Not assumed:** number of robots: any number $k \geq 2$ of robots; randomness: deterministic.

Statement for this setting. $k \geq 3$ robots that know d (not the direction of the target); the targets are the points of the circle of radius d about the origin.

Measure	Lower bound	Best upper bound	Status
Competitive ratio	$\sec(\pi/k)$ (Kranakis 2026) [preprint] Exactly $\sec(\pi/k)$ for every $k \geq 3$.	$\sec(\pi/k)$ (Kranakis 2026) [preprint]	• tight

Results.

Claude (Anthropic) was used in cataloguing the problems.

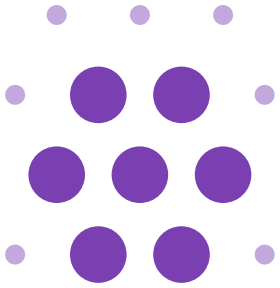
- Kranakis (2026) [preprint] [lower bound]: competitive ratio $\sec(\pi/k)$. *Bottleneck:* unknown direction. *Holds when:* deterministic; $k \geq 3$, targets on the circle of radius d, d known; for $k \geq 4$ a direct argument: at any time $T < d/\cos(\pi/k)$ some open cone of angle $2\pi/k$ at the origin holds no robot, and no robot has touched the line tangent to the circle on its bisector; for $k = 3$ from Theorem 1 of (Dobrev et al. 2020b) (shoreline search at known distance, bound 2). Lemma 4.5 and Theorem 4.1 of (Kranakis 2026); Lemma 4.5 adapts Section 2 of (Acharjee et al. 2019) to a fixed distance.
- Kranakis (2026) [preprint] [current best]: competitive ratio $\sec(\pi/k)$. *Assumes:* common coordinate frame and clock. The expanding k-gon of the unknown-distance setting; it does not use d (Theorem 4.1).

Notes. From the preprint (Kranakis 2026), read in full (arXiv v1).

Claude (Anthropic) was used in cataloguing the problems.

Part VI

Programmable matter



The model and its assumption axes are in Chapter 5.

Claude (Anthropic) was used in cataloguing the problems.

Chapter 21

Self-organising particle systems

Leader election, shape formation, coating, scattering, stochastic tasks (compression, separation, alignment, bridging), energy distribution, hybrid and centralised reconfiguration.

Problems in this family:

- Leader election in programmable matter
- Shape formation in programmable matter
- Coating
- Compression
- Separation of coloured particles — generalisation of Compression
- Alignment — generalisation of Compression
- Bridging — generalisation of Compression
- Convex hull formation
- Scattering of particles
- Energy distribution
- Shape formation by hybrid programmable matter
- Centralised transformations of modular robots



21.1 Leader election in programmable matter

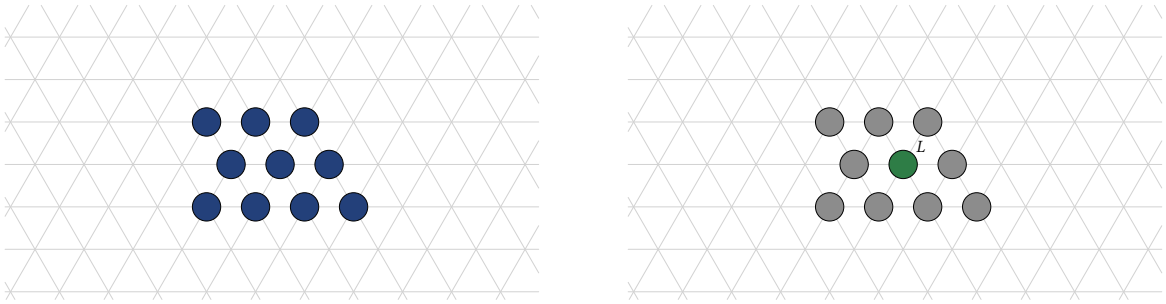
Anonymous constant-memory particles must elect exactly one leader, which most shape-formation algorithms need first.

First posed in 2015 (Derakhshandeh et al. 2015b). Introduced with shape formation at DNA 21 (2015).

Problem statement and measures

A connected **configuration** (the set of nodes occupied by particles) of n anonymous particles (initially contracted) must reach a state in which exactly one particle is the leader and all others know they are not. **Time** is in asynchronous rounds; L_{out} is the length of the outer boundary and D the diameter of the configuration.

Measures: **Time**, asynchronous rounds; **Moves**, total expansions and contractions (work).



(a) Start: every particle is eligible

(b) Final: one candidate left, the leader

Figure 21.1: Leader election by erosion on a configuration without holes: no particle moves. An eligible particle (blue) whose eligible neighbours are few and consecutive (none, one, two that are neighbours of each other, three in a row) is a corner and drops out (grey); corners drop out round after round, peeling the configuration from its boundary, until a single candidate remains and becomes the leader (Di Luna et al. 2020e; D'Angelo et al. 2020). Configurations with holes need other algorithms (Derakhshandeh et al. 2015b).

A first look

Imagine a swarm of identical tiny machines, packed together on a floor tiled with triangles, that must pick one boss without any name tags. Each machine is a **particle**. It occupies one **node** of the **triangular lattice**, the infinite tiling of the plane by equilateral triangles whose corners are the nodes; particles on adjacent nodes are **neighbours**. A particle remembers only a few bits, and can read and write the memory of its neighbours. It moves in two steps: an **expansion** into an empty adjacent node, after which it briefly occupies both nodes, then a **contraction** into one of the two; so one expansion and one contraction move it by one node. In the standard version there are n particles whose nodes form one connected **configuration**, possibly with **holes** (empty regions enclosed by particles); all particles agree on what clockwise means (**chirality**), but not on a common north. An adversary, a scheduler that we assume chooses the worst order, decides when each particle acts; each time a particle acts is an **activation**, and a **round** is a stretch of time in which every particle has been activated at least once. The goal: exactly one particle ends in state **leader**, every other particle knows it is not, and we count the rounds this takes.

1. Algorithm: Erosion: peel the configuration from the outside. (D'Angelo et al. 2020) (Silent oblivious particles (SILBOT), p. 596) This works on configurations **without holes**. It is the erosion of (Di Luna et al. 2020e), which (D'Angelo et al. 2020) showed can be run even by particles with no memory and no messages. No particle changes its node (in the memoryless version, expansions serve as signals).

1. Every particle starts as *eligible*.
2. An eligible particle is a *corner* if its eligible neighbours are few and sit next to each other: none, one, two that are neighbours of each other, or three in a row.
3. A corner drops out (becomes ineligible). Corners keep dropping out round after round, so the configuration is eaten from its outer **boundary** (the particles next to the empty outside; a hole has a boundary too) inwards, like a coastline eroded by the sea.
4. A few special cases stop the peeling: a lone eligible particle, two eligible particles that are each other's only eligible neighbour, or a triangle of three, become **candidates** (for leader) instead of dropping out.
5. At the end, 1, 2 or 3 candidates remain, each a neighbour of the others.

Claude (Anthropic) was used in cataloguing the problems.

Why it works and what it costs. A hole-free configuration always has a corner, and removing corners never splits the eligible particles into parts or creates a hole, so erosion never gets stuck. Each removal leaves fewer eligible particles, and (D'Angelo et al. 2020) show the process ends in $O(n)$ rounds. It cannot always leave a single leader: three particles in a triangle look exactly alike (see the next idea). It also fails with holes: in a ring of particles around a hole, each particle has two eligible neighbours that are not neighbours of each other, so nobody is a corner. The best known result elects exactly one leader, holes allowed, deterministically in $O(L_{out} + D)$ rounds, where L_{out} is the length of the outer boundary and D the diameter of the configuration (Dufoulon et al. 2021).

Why it matters. Erosion is the first step of the deterministic shape-formation algorithm and is reused whenever a few candidate leaders are good enough.

2. Impossibility: Three identical particles in a triangle. (Chalopin et al. 2026b) (Stationary, chirality, weak scheduler, p. 593) Take three particles forming a small triangle, and suppose particles never move and the scheduler may activate all three at the same moment. Each particle sees the same picture: two neighbours, one to its left and one to its right (they have chirality, so 'left' means the same to all). They start in the same state, so in their first activation they compute the same thing and write the same thing. After that activation they are again in identical states with identical views, and so on forever. Whatever a deterministic algorithm (one that flips no coins) does, either all three become leader or none does (Chalopin et al. 2026b).

Why it holds. The argument only uses symmetry: an algorithm cannot tell apart particles whose surroundings are rotated copies of each other. The same reasoning shows that a configuration with six-fold symmetry cannot end with fewer than six leaders, which is why stationary particles with chirality elect up to six leaders (Bazzi and Briones 2019). To get exactly one leader, papers add something that breaks the symmetry: coin flips (Derakhshandeh et al. 2015b), a scheduler that activates one particle at a time (Briones et al. 2023), or a common direction (Chalopin et al. 2026b).

Why it matters. Symmetry is the basic obstacle in all of programmable matter; it explains why results say 'at most three' or 'at most six' leaders.

3. Algorithm: Coin flips along the boundaries. (Derakhshandeh et al. 2015b) (Randomized, with chirality, p. 589) Only particles on a boundary (the outer boundary, or the boundary of a hole) take part. Each boundary closes on itself, and with chirality everyone walks it in the same direction.

1. Every boundary particle starts as a candidate. Between two consecutive candidates lies a *segment* of non-candidates.
2. Compare segments: a candidate whose segment ahead is shorter than the segment behind it gives up (particles compare lengths by passing tokens, one per particle).
3. If all segments are equal, every candidate flips a coin. On heads it passes its candidacy forward to the next candidate's spot; when it does and the next candidate keeps its own, two candidacies land on one spot and merge.
4. A candidate that is alone on its boundary checks whether it is the outer boundary (by adding up the turns along it; the outer boundary turns one full circle one way, the boundary of a hole the other way). Alone on the outer boundary: it is the leader. Alone on the boundary of a hole: it gives up.

Why it works and what it costs. There is always at least one candidate on the outer boundary, and while there are two or more, each phase removes one with probability at least $1/4$, so the number of candidates keeps shrinking. Holes are no problem, because the boundary of a hole eventually loses all its candidates. The expected time is $O(L_{max})$ rounds, L_{max} the length of the longest boundary (Derakhshandeh et al.

Claude (Anthropic) was used in cataloguing the problems.

2015b); this was improved to $O(n)$ rounds with high probability (Daymude et al. 2017), and later matched without coins (Dufoulon et al. 2021).

Why it matters. Working on boundaries instead of the whole configuration is the idea behind the fast algorithms that followed.

What the community started looking at

Leader election was the first task studied, because shape formation needs a leader (Derakhshandeh et al. 2015b). The first question was **feasibility with constant memory**, then **rounds**: randomized algorithms came first (Derakhshandeh et al. 2015b; Daymude et al. 2017), then deterministic ones without chirality (Emek et al. 2019) and in linear time (Dufoulon et al. 2021). Later work weakened the model in three directions: **stationary** particles (Bazzi and Briones 2019; Chalopin et al. 2026b), **asynchronous** adversaries (Briones et al. 2023), and **silent, oblivious** particles (D'Angelo et al. 2020); self-stabilising election with constant memory followed (Chalopin et al. 2025, 2026a). Brief announcements in 2026 added crash-recovery faults (Warner and Scheideler 2026a) and noisy reconfigurable circuits (Warner and Scheideler 2026b).

Further reading

Papers on this problem that are not yet placed in a setting.

- Gastineau et al. (2020). Deterministic leader election for particles on the face-centred cubic grid, in three dimensions: one algorithm assumes a special initial configuration, the other works for every connected structure that is electable in the authors' sense (Theorem 3.6); also identifiers that differ within distance ℓ, with memory independent of n. Read (arXiv v1).

Relations to other problems

- **Related: Leader election in the plane.** Both elect a leader among anonymous entities from geometry alone; particles on the lattice can also move and talk to neighbours, robots in the plane only see (Dufoulon et al. 2021).

Assumptions specific to this problem

Movement

The early leader-election algorithms for amoebots only exchanged messages, and "did not use the movement ability as an aid to the computation" (Dufoulon et al. 2021). Emek et al. showed that movement lets particles elect a unique leader deterministically with no assumption on chirality or holes; the main idea is "the usage of the ability of the amoebots to move, which previous leader election algorithms have not used" (Emek et al. 2019). The stationary setting stayed interesting because particles may be unable to move, "so as to maintain a configuration or due to the presence of obstacles" (Chalopin et al. 2026b). Bazzi and Briones studied stationary deterministic election with chirality under a weak scheduler, where a configuration with k-fold rotational symmetry ($k = 2$, 3 or 6) cannot elect fewer than k leaders (Bazzi and Briones 2019). Chalopin, Das and Kokkou showed that one common direction suffices for a unique stationary leader (Chalopin et al. 2026b).

Claude (Anthropic) was used in cataloguing the problems.

Level	Value	Formal definition
0	**stationary**	No particle ever expands or contracts; the configuration is fixed and only states change.
1	**may move**	Particles may expand and contract during the election.

Case tree

The root assumes nothing; going down, each edge adds the assumptions written in grey (+) or leaves an axis unassumed. Bold entries are catalogued settings with their status per measure (Time, Moves) and page; italic entries are branch points, named after the axis on which their branches differ. Axis order: Memory → Scheduler → Orientation → Movement → Communication → Randomness → Initial shape → Connectivity during execution.

No assumptions

+ constant memory
Scheduler

+ strong scheduler
Orientation

+ chirality
Movement

+ may move
Randomness

deterministic, + may disconnect
Deterministic, with chirality STANDARD SETTING ●● p. 588

+ randomized
Randomized, with chirality ●● p. 589

stationary, + no holes
Stationary, chirality, no holes ●● p. 589

no orientation, + may move
Deterministic, no orientation ●● p. 590

+ common direction
Stationary, common direction, no chirality ●● p. 590

no orientation, stationary, + no holes
Stationary, fair scheduler, no orientation, no holes ●● p. 591

+ sequential
Randomness

deterministic, + no holes
Stationary, sequential, no orientation, no holes ●● p. 591

+ randomized
Stationary, randomized, sequential, holes allowed ●● p. 592

weak scheduler, + chirality
Stationary, chirality, weak scheduler ●● p. 593

weak scheduler, no orientation, local communication, deterministic, + no holes
Stationary, asynchronous, no orientation, no holes ●● p. 594

+ synchronous, + circuits, + randomized
Stationary, reconfigurable circuits, synchronous, randomized ●● p. 594

Claude (Anthropic) was used in cataloguing the problems.

oblivious, weak scheduler, no orientation, + may move
SILBOT with holes ●● p. 595

+ no holes
Silent oblivious particles (SILBOT) ●● p. 596

oblivious, + sequential, + common compass, + may move
Oblivious, common compass, holes allowed ●● p. 597

What is known

The best upper bound against the best lower bound in each catalogued setting, in tree order. The settings follow in detail.

Setting	Time	Moves	p.
Deterministic, with chirality STANDARD SETTING	● unknown $O(L_{out} + D)$ vs —	● unknown — vs —	588
Randomized, with chirality	● unknown $O(n)$ vs —	● unknown — vs —	589
Stationary, chirality, no holes	● unknown $O(n)$ vs —	● tight — vs —	589
Deterministic, no orientation	● polynomial gap $O(L_{out}n^2)$ vs —	● unknown — vs —	590
Stationary, common direction, no chirality	● unknown $O(n^3)$ vs *explicit termination impossible*	● tight — vs —	590
Stationary, fair scheduler, no orientation, no holes	● unknown *finite* vs —	● tight — vs —	591
Stationary, sequential, no orientation, no holes	● unknown $O(n)$ vs —	● tight — vs —	591
Stationary, randomized, sequential, holes allowed	● unknown $O(n \log n)$ vs —	● tight — vs —	592
Stationary, chirality, weak scheduler	● unknown $O(n^2)$ vs *unique leader impossible*	● tight — vs —	593
Stationary, asynchronous, no orientation, no holes	● unknown *finite* vs —	● tight — vs —	594
Stationary, reconfigurable circuits, synchronous, randomized	● unknown $O(\log n)$ vs —	● tight — vs —	594
SILBOT with holes	● impossible	● unknown — vs —	595
Silent oblivious particles (SILBOT)	● unknown $O(n)$ vs —	● unknown — vs —	596
Oblivious, common compass, holes allowed	● unknown *finite* vs —	● unknown — vs —	597

21.1.1 Deterministic, with chirality

Adds deterministic, + may disconnect STANDARD SETTING VERIFIED

Deterministic leader election in $O(L_{out} + D)$ rounds, as fast as the best randomized algorithm, holes allowed.

Assumed: memory: constant memory and local communication; scheduler: strong: atomic activations under a fair adversary; orientation: common chirality; movement: particles may move; connectivity during execution: the system may disconnect

Claude (Anthropic) was used in cataloguing the problems.

temporarily. **Not assumed:** communication: local: only between neighbouring particles; randomness: deterministic; initial shape: connected, holes allowed.

Measure	Lower bound	Best upper bound	Status
Time	none known Only an informal $\Omega(D)$ remark; no proven lower bound.	$O(L_{out} + D)$ (Dufoulon et al. 2021)	• unknown
Moves	none known —	—	• unknown

Results.

- Dufoulon et al. (2021) [current best]: time $O(L_{out} + D)$. Introductory for linear time and current best; $O(D_A)$, where D_A is the diameter of the area enclosed by the particles (holes included), if particles know whether their boundary is the outer one. The system may temporarily disconnect.

Open questions.

Q1. Fast deterministic leader election under weaker (non-atomic) schedulers, and without letting the system disconnect (Dufoulon et al. 2021).

21.1.2 Randomized, with chirality

Adds + randomized VERIFIED

Randomized leader election in a linear number of rounds with high probability.

Assumed: memory: constant memory and local communication; scheduler: strong: atomic activations under a fair adversary; orientation: common chirality; movement: particles may move; randomness: randomized. **Not assumed:** communication: local: only between neighbouring particles; initial shape: connected, holes allowed; connectivity during execution: the system must stay connected.

Measure	Lower bound	Best upper bound	Status
Time	none known No proven lower bound.	$O(n)$ (Daymude et al. 2017)	• unknown
Moves	none known —	—	• unknown

Results.

- Derakhshandeh et al. (2015b) [introductory]: time $O(L_{max})$. Expected rounds; L_{max} is the longest boundary.
- Daymude et al. (2017) [current best]: time $O(n)$. With high probability; restated as $O(L_{out} + D)$ by (Dufoulon et al. 2021).

Notes. In the general amoebot model on arbitrary graphs leader election is impossible, by reduction to anonymous rings (Derakhshandeh et al. 2015b).

21.1.3 Stationary, chirality, no holes

Adds stationary, + no holes VERIFIED

On hole-free configurations with chirality, stationary constant-memory particles elect a leader deterministically in $O(n)$ rounds.

Assumed: memory: constant memory and local communication; scheduler: strong: atomic activations under a fair adversary; orientation: common chirality; initial shape: simply connected (no holes). **Not assumed:** movement: stationary: particles

Claude (Anthropic) was used in cataloguing the problems.

never move; communication: local: only between neighbouring particles; randomness: deterministic; connectivity during execution: the system must stay connected.

Measure	Lower bound	Best upper bound	Status
Time	none known No lower bound recorded.	$O(n)$ (Gastineau et al. 2018)	● unknown
Moves	none known No moves.	—	● tight

Results.

- Gastineau et al. (2018) [current best]: time $O(n)$. Introductory and current best: $O(r + m_{tree})$ rounds, fewer than $2n$ (Proposition 1). Also computes k-local identifiers; works on the square, triangular and king grids.

Notes. r and m_{tree} are parameters of the configuration defined in (Gastineau et al. 2018): r is the radius of the configuration with respect to its border particles (the least, over particles u, of the largest distance from u to a border particle), and m_{tree} is the largest height of an induced subgraph of the configuration that is a tree. The scheduler is strong, as listed by (Dufoulon et al. 2021). Checked against the full text.

21.1.4 Deterministic, no orientation

Adds no orientation, + may move VERIFIED

Without chirality, a unique leader can still be elected deterministically, in $O(L_{out}n^2)$ rounds.

Assumed: memory: constant memory and local communication; scheduler: strong: atomic activations under a fair adversary; movement: particles may move. **Not assumed:** orientation: no common orientation and no chirality; communication: local: only between neighbouring particles; randomness: deterministic; initial shape: connected, holes allowed; connectivity during execution: the system must stay connected.

Measure	Lower bound	Best upper bound	Status
Time	none known Far from the linear bound known with chirality.	$O(L_{out}n^2)$ (Emek et al. 2019)	● polynomial gap
Moves	none known —	—	● unknown

Results.

- Emek et al. (2019) [current best]: time $O(L_{out}n^2)$. Introductory and current best; also yields agreement on chirality.

Open questions.

Q1. Deterministic leader election without chirality in linear time.

21.1.5 Stationary, common direction, no chirality

Adds + common direction VERIFIED

With one common direction and no chirality, a unique leader is elected without moving, with obstacles allowed, but only with implicit termination.

Assumed: memory: constant memory and local communication; scheduler: strong: atomic activations under a fair adversary; orientation: a common direction, no chirality. **Not assumed:** movement: stationary: particles never move; communication: local: only between neighbouring particles; randomness: deterministic; initial shape: connected, holes allowed; connectivity during execution: the system must stay connected.

Claude (Anthropic) was used in cataloguing the problems.

Measure	Lower bound	Best upper bound	Status
Time	*explicit termination impossible* (Chalopin et al. 2026b) No round bound recorded.	$O(n^3)$ (Chalopin et al. 2026b)	● unknown
Moves	none known No moves.	—	● tight

Results.

- Chalopin et al. (2026b) [lower bound]: time *explicit termination impossible*. *Bottleneck:* symmetry breaking. *Holds when:* Theorem 1: deterministic algorithms; connected configuration with obstacles (holes); common direction (one axis), no chirality; holds even under a synchronous scheduler; implicit termination remains possible.
- Chalopin et al. (2026b) [current best]: time $O(n^3)$. Introductory and current best (Section 6); implicit termination, unique leader, deterministic, stationary.

Notes. With chirality but no common direction, explicit termination is always possible but the number of leaders depends on the symmetry of the initial configuration (Chalopin et al. 2026b). Here "common direction" means agreement on one axis only. Checked against the full text.

21.1.6 Stationary, fair scheduler, no orientation, no holes

Adds no orientation, stationary, + no holes VERIFIED

Stationary constant-memory particles with no orientation and no chirality elect a unique leader from any initial memory: silent self-stabilising election on simply connected systems.

Assumed: memory: constant memory and local communication; scheduler: strong: atomic activations under a fair adversary; initial shape: simply connected (no holes). **Not assumed:** orientation: no common orientation and no chirality; movement: stationary: particles never move; communication: local: only between neighbouring particles; randomness: deterministic; connectivity during execution: the system must stay connected.

Measure	Lower bound	Best upper bound	Status
Time	none known —	*finite* (Chalopin et al. 2025)	● unknown
Moves	none known No moves.	—	● tight

Results.

- Chalopin et al. (2025) [current best]: time *finite*. *Assumes:* Gouda-fair scheduler; self-stabilising: any initial memory. Introductory and current best. The algorithm fails under an unfair scheduler (Section 6). Constant memory suffices although silent self-stabilising election in general networks needs $\Omega(\log n)$ bits, even on rings.

Notes. Self-stabilisation is a stronger requirement than the problem statement and has no axis here. Checked against the full text.

21.1.7 Stationary, sequential, no orientation, no holes

Adds deterministic, + no holes VERIFIED

Under an unfair sequential adversary, stationary particles with no common orientation elect a unique leader in $O(n)$ rounds, in 2D or 3D.

Assumed: memory: constant memory and local communication; scheduler: sequential: one particle active at a time; initial shape: simply connected (no holes). **Not assumed:** orientation: no common orientation and no chirality; movement:

Claude (Anthropic) was used in cataloguing the problems.

stationary: particles never move; communication: local: only between neighbouring particles; randomness: deterministic; connectivity during execution: the system must stay connected.

Measure	Lower bound	Best upper bound	Status
Time	none known No lower bound recorded.	$O(n)$ (Briones et al. 2023)	● unknown
Moves	none known No moves.	—	● tight

Results.

- Briones et al. (2023) [current best]: time $O(n)$. *Assumes:* unfair sequential adversary. Introductory and current best (Theorem 3 and Corollary 4).

Notes. Checked against the full text.

21.1.8 Stationary, randomized, sequential, holes allowed

Adds + randomized DRAFT

With coin flips and a sequential scheduler, stationary particles without a common orientation elect a unique leader, holes allowed, in $O(n \log n)$ rounds with high probability, as stated without proof in a brief announcement that also tolerates crash-recovery faults.

Assumed: memory: constant memory and local communication; scheduler: sequential: one particle active at a time; randomness: randomized. **Not assumed:** orientation: no common orientation and no chirality; movement: stationary: particles never move; communication: local: only between neighbouring particles; initial shape: connected, holes allowed; connectivity during execution: the system must stay connected.

Measure	Lower bound	Best upper bound	Status
Time	none known No lower bound; the upper bound is stated without proof.	$O(n \log n)$ (Warner and Scheideler 2026a)	● unknown
Moves	none known No moves.	—	● tight

Results.

- Warner and Scheideler (2026a) [current best]: time $O(n \log n)$. *Assumes:* neighbours know their relative orientation: the rotation or reflection of the lattice that maps one particle's local directions to the other's; with high probability; crash-recovery faults allowed; time counted after the last crash. Introductory and current best, as stated in Section 3.2 without proof (a brief announcement). Every candidate is the root of a tree. In each phase every root flips a coin, and a tree whose root flipped 0 attaches to a neighbouring tree whose root flipped 1, so a constant fraction of the candidates disappears in expectation and $O(\log n)$ phases suffice with high probability. A solitude check certifies that one tree remains: particles compute their coordinates relative to their root bit by bit and compare them across each edge. The paper says "runtime" without naming the unit.

Open questions.

Q1. Prove the $O(n \log n)$ bound with high probability that (Warner and Scheideler 2026a) states without proof, and give a lower bound on rounds.

Q2. Can stationary randomized particles without a common orientation elect a unique leader on configurations with holes under the strong or the weak scheduler, which let several particles act at the same time?

Claude (Anthropic) was used in cataloguing the problems.

Notes. Warner and Scheideler describe the local frame of a particle by a symmetry of the lattice: one of the 12 rotations and reflections of the triangular lattice, or one of the 48 of the face-centred cubic lattice in 3D. A particle does not know its own frame, but two neighbours know how their frames relate. Numbers of $O(\log n)$ bits are streamed or spread over $O(\log n)$ particles, so memory stays constant. Faults follow a crash-recovery model from earlier work of Kostitsyna, Scheideler and Warner (DNA 28, 2022; not in this atlas): a crashed particle loses its memory and enters a crashed state that it can detect, and on recovery it becomes the root of its own tree, so a crash cuts a tree into smaller trees that keep competing; correctness is required whenever there are finitely many crashes. The randomized algorithm, as outlined, never moves particles. The same paper outlines a deterministic algorithm: run the stationary merging once for every possible global frame, which leaves at most 12 candidates on the triangular lattice in $O(n \log^2 n)$ rounds after the last crash, then move the candidates towards the centroid of the configuration until one remains. No bound is given for this movement phase, and its faster option may disconnect the configuration and then loses fault tolerance. Without faults, on the triangular lattice, deterministic election with movement and no orientation is known under the strong scheduler (Emek et al. 2019), a weaker assumption than the sequential one. Checked against the full text, a brief announcement without theorems or proofs; the bounds are the authors' statements.

21.1.9 Stationary, chirality, weak scheduler

Adds weak scheduler, + chirality VERIFIED

Without moving, particles with chirality elect a unique leader unless the configuration is k-symmetric, and then $k \in \{2, 3, 6\}$ leaders, which cannot be avoided; holes allowed, $O(n^2)$ rounds as stated.

Assumed: memory: constant memory and local communication; orientation: common chirality. **Not assumed:** scheduler: weak: non-atomic activations, an action may interleave with others; movement: stationary: particles never move; communication: local: only between neighbouring particles; randomness: deterministic; initial shape: connected, holes allowed; connectivity during execution: the system must stay connected.

Statement for this setting. Elect a unique leader when the configuration allows it; otherwise elect k leaders, where $k \in \{2, 3, 6\}$ is the symmetricity of the configuration. Holes are allowed.

Measure	Lower bound	Best upper bound	Status
Time	*unique leader impossible* (Bazzi and Briones 2019) No round lower bound; the $O(n^2)$ upper bound is stated without proof. A unique leader is impossible on k-symmetric configurations, so the relaxation to k leaders is necessary.	$O(n^2)$ (Bazzi and Briones 2019)	● unknown
Moves	none known No moves.	—	● tight

Results.

- Bazzi and Briones (2019) [lower bound]: time *unique leader impossible. Bottleneck:* symmetry breaking. *Holds when:* Section 5.3: deterministic, stationary algorithms; a configuration with k-fold rotational symmetry, $k \in \{2, 3, 6\}$: fewer than k leaders cannot be elected; weak scheduler: all particles read their neighbours' memory, then all write, and the cycle repeats; holds even with chirality. With a strong scheduler symmetry can be broken without moving, for example between two particles (Section 6).
- Bazzi and Briones (2019) [current best]: time $O(n^2)$. Introductory and current best: a unique leader whenever the configuration is not k-symmetric, otherwise exactly k leaders (Sections 4–6). First 1, 2, 3 or 6 candidates are elected on the outer boundary by merging stretches (runs of consecutive points along the boundary, each carrying the sum of the ±1 labels of its points, at most 6), then candidates grow trees and compare them. The $O(n^2)$ bound is stated in the abstract (as steps) and in Table 1 (as rounds); the paper gives no proof of it.

Open questions.

Claude (Anthropic) was used in cataloguing the problems.

Q1. Prove, or improve, the $O(n^2)$-round bound of (Bazzi and Briones 2019), which the paper states without proof, and give a lower bound on rounds.

Q2. Elect a bounded number of leaders deterministically, without moving, under the weak scheduler and without chirality, on configurations with holes; Bazzi and Briones conjecture that it is possible (Bazzi and Briones 2019).

Notes. Model (Section 2): anonymous particles with finite memory and chirality, all contracted, connected, holes allowed; a particle reads its neighbours' memory and writes its own, but not in one atomic step, and the scheduler may activate a particle any finite number of times before another. The question whether a unique leader is ever impossible here (raised from Dufoulon et al. 2021) is answered by Section 5.3 of (Bazzi and Briones 2019): yes, on every k-symmetric configuration with $k \in \{2, 3, 6\}$. Bazzi and Briones report that (Emek et al. 2019) use their algorithm, announced at SSS 2018, as a subroutine. Checked against the full text.

21.1.10 Stationary, asynchronous, no orientation, no holes

Adds weak scheduler, no orientation, local communication, deterministic, + no holes VERIFIED

Stationary particles with no common orientation elect a leader in connected, contractible 2D or 3D systems under an unfair asynchronous adversary; the running time there is not analysed.

Assumed: memory: constant memory and local communication; initial shape: simply connected (no holes). **Not assumed:** scheduler: weak: non-atomic activations, an action may interleave with others; orientation: no common orientation and no chirality; movement: stationary: particles never move; communication: local: only between neighbouring particles; randomness: deterministic; connectivity during execution: the system must stay connected.

Measure	Lower bound	Best upper bound	Status
Time	none known Runtime not analysed under the asynchronous adversary.	*finite* (Briones et al. 2023)	● unknown
Moves	none known No moves.	—	● tight

Results.

- Briones et al. (2023) [current best]: time *finite*. Introductory and current best (Corollary 6): the sequential algorithm transformed by the concurrency control framework (Daymude et al. 2023); the paper says its runtime "has not yet been analyzed".

Open questions.

Q1. Deterministic leader election in $O(L_{out} + D)$ rounds under a weaker scheduler (Dufoulon et al. 2021).

Notes. Contractible means no holes, in 2D or in 3D (face-centred cubic lattice). Particles have "assorted orientations": no common direction and no chirality. Checked against the full text.

21.1.11 Stationary, reconfigurable circuits, synchronous, randomized

Adds + synchronous, + circuits, + randomized VERIFIED

With reconfigurable circuits, stationary particles acting in synchronous rounds elect a unique leader in $O(\log n)$ rounds with high probability, holes allowed, by coin-flip tournaments on a circuit that joins all particles.

Claude (Anthropic) was used in cataloguing the problems.

Assumed: memory: constant memory and local communication; scheduler: synchronous: every particle acts in every round; communication: reconfigurable circuits; randomness: randomized. **Not assumed:** orientation: no common orientation and no chirality; movement: stationary: particles never move; initial shape: connected, holes allowed; connectivity during execution: the system must stay connected.

Statement for this setting. Particles do not move, act in synchronous rounds and also communicate through reconfigurable circuits: a beep sent on a circuit reaches every particle on it in the next round. The configuration is connected and may have holes; particles share no orientation.

Measure	Lower bound	Best upper bound	Status
Time	none known No lower bound for circuits in the papers read.	$O(\log n)$ (Feldmann et al. 2022)	● unknown
Moves	none known No moves.	—	● tight

Results.

- Feldmann et al. (2022) [current best]: time $O(\log n)$. *Assumes:* with high probability; Monte Carlo: with small probability more than one candidate remains; one pin per edge suffices; no common chirality. Introductory and current best (Theorem 1.1 and Section 1.2.1 of the arXiv version): all particles link their pins into one global circuit. In each step every candidate flips a coin and beeps on the circuit in the first of two rounds if it flipped heads, in the second if tails; if some candidate beeped in the first round, those that beeped in the second drop out. A second tournament run in parallel, repeated a constant number of times, ends the election after $\Theta(\log n)$ rounds. The scheme adapts the beeping-model election of Gilbert and Newport, as the authors report.
- Warner and Scheideler (2026b) [also]: time $O(\log n)$. *Assumes:* with high probability; Las Vegas: never two leaders; stated in Section 1 and Table 1, not as a theorem; the proofs are sketches. Candidates are roots of trees that merge by coin flips, and a candidate stops only after checking, bit by bit, that no edge joins two trees, so the algorithm never elects two leaders. With noisy circuits, on which each particle hears a beep only with probability $1-\varepsilon$ for a constant $\varepsilon \in (0,1)$ while communication between neighbours stays reliable, it takes $O(\sqrt{\log n}\,\log n)$ rounds with high probability (Theorem 1); once a leader exists, any stationary algorithm for noiseless circuits runs with a factor $O(\sqrt{\log n})$ of overhead (Theorem 2). It works on the triangular lattice and on the face-centred cubic lattice in three dimensions. A brief announcement.

Open questions.

Q1. Give a lower bound on the rounds needed to elect a leader with circuits; neither paper read proves one.

Notes. Without circuits information travels one particle at a time, so a leader cannot be elected faster than the diameter of the configuration allows (Feldmann et al. 2022); circuits remove this bound. Warner and Scheideler's check never looks for the boundary of a hole, so holes are allowed (Warner and Scheideler 2026b). Checked against the model section, Section 1.2.1 and Theorem 1.1 of the arXiv version of (Feldmann et al. 2022) (the proof in Appendix B was not checked line by line) and the full text of (Warner and Scheideler 2026b).

21.1.12 SILBOT with holes

Adds oblivious, weak scheduler, no orientation, + may move VERIFIED

With holes, SILBOT particles cannot elect up to three leaders while staying connected; if they can tell a hole from the exterior, election with compaction takes $O(n^2)$ rounds.

Assumed: movement: particles may move. **Not assumed:** memory: oblivious, silent particles (SILBOT); scheduler: weak: non-atomic activations, an action may interleave with others; orientation: no common orientation and no chirality; communication:

Claude (Anthropic) was used in cataloguing the problems.

local: only between neighbouring particles; randomness: deterministic; initial shape: connected, holes allowed; connectivity during execution: the system must stay connected.

Statement for this setting. Particle leader election as in the hole-free SILBOT variant (one, two or three mutually adjacent leaders), from any connected configuration of contracted particles, possibly with holes, keeping the system connected.

Measure	Lower bound	Best upper bound	Status
Time	*impossible* (D'Angelo et al. 2020)	$O(n^2)$ (D'Angelo et al. 2020)	• impossible
	Impossible in plain SILBOT; with exterior awareness $O(n^2)$ rounds and no lower bound.		
Moves	none known	—	• unknown
	Not analysed.		

Results.

- D'Angelo et al. (2020) [lower bound]: time *impossible*. *Bottleneck:* symmetry breaking. *Holds when:* a connected configuration with a hole in which groups of particles have identical 2-hop views; the adversary activates each such set simultaneously, each particle twice in a row; the system must stay connected; holds even with unlimited memory and chirality. Theorem 2: every move some particle could be instructed to make disconnects the system under some schedule.
- D'Angelo et al. (2020) [current best]: time $O(n^2)$. *Assumes:* exterior awareness: a particle tells an empty node in a hole from an exterior empty node within its view; 2-hop view. Introductory and current best: Algorithm CHLE (Theorem 3) fills the holes (compaction: the configuration becomes simply connected) while electing at most three leaders; on hole-free configurations it behaves as LESC.

Open questions.

Q1. Can SILBOT particles elect leaders from configurations with holes if the system may disconnect temporarily, without exterior awareness? The impossibility of (D'Angelo et al. 2020) assumes that connectivity is preserved.

Notes. *Compaction* (of holes) asks to turn any connected configuration into a simply connected one. Exterior awareness is motivated by local sensing (light or pressure). Checked against the full text of (D'Angelo et al. 2020).

21.1.13 Silent oblivious particles (SILBOT)

Adds + no holes **to** SILBOT with holes (p. 595) VERIFIED

Silent oblivious particles (SILBOT) elect at most three mutually adjacent leaders from hole-free configurations in $O(n)$ rounds.

Assumed: movement: particles may move; initial shape: simply connected (no holes). **Not assumed:** memory: oblivious, silent particles (SILBOT); scheduler: weak: non-atomic activations, an action may interleave with others; orientation: no common orientation and no chirality; communication: local: only between neighbouring particles; randomness: deterministic; connectivity during execution: the system must stay connected.

Statement for this setting. Particle leader election (PLE): from a connected configuration of contracted particles, terminate with one, two or three mutually adjacent contracted particles that recognise themselves as LEADER, every other particle expanded (NON-LEADER), and the system connected throughout. A single leader is not required: D'Angelo et al. note that a single particle can be elected only with randomisation or a sequential scheduler, citing (Emek et al. 2019).

Claude (Anthropic) was used in cataloguing the problems.

Measure	Lower bound	Best upper bound	Status
Time	none known No lower bound recorded.	$O(n)$ (D'Angelo et al. 2020)	• unknown
Moves	none known Not analysed.	—	• unknown

Results.

- D'Angelo et al. (2020) [current best]: time $O(n)$. *Assumes:* 2-hop view; initial configuration deeply simply connected (e.g. all particles contracted, no holes); no chirality, deterministic. Introductory and current best: Algorithm LESC (Theorem 1), which only expands particles; an emulation of the erosion of (Di Luna et al. 2020e) with simpler rules.

Notes. SILBOT (D'Angelo et al. 2020) removes from the amoebot model the serialised scheduler (no two neighbouring particles active together), explicit communication, atomicity of actions and randomness. Particles see up to two hops, distinguish empty nodes from contracted and expanded particles, and act asynchronously; a particle that has expanded along an edge is committed and contracts into the target node at its next activation if the node is empty. A round is the shortest period in which every particle has completed an activation. The paper argues that 1-hop visibility is not enough without communication. Holes are handled in the variant without the hole-free assumption. Checked against the full text of (D'Angelo et al. 2020) (IEEE Access version; preliminary version at AAMAS 2020).

21.1.14 Oblivious, common compass, holes allowed

Adds oblivious, + sequential, + common compass, + may move VERIFIED

Oblivious particles that agree on all directions elect a unique leader in connected configurations with holes, self-stabilising, using movement, under an unfair sequential scheduler.

Assumed: scheduler: sequential: one particle active at a time; orientation: full common orientation (compass); movement: particles may move. **Not assumed:** memory: oblivious, silent particles (SILBOT); communication: local: only between neighbouring particles; randomness: deterministic; initial shape: connected, holes allowed; connectivity during execution: the system must stay connected.

Measure	Lower bound	Best upper bound	Status
Time	none known —	*finite* (Chalopin et al. 2026a)	• unknown
Moves	none known —	—	• unknown

Results.

- Chalopin et al. (2026a) [current best]: time *finite*. *Assumes:* unfair sequential scheduler with atomic moves. Introductory and current best; silent self-stabilising.

Notes. The abstract says "common sense of direction"; Section 2 assumes agreement on all directions. Movement together with this orientation overcomes the impossibility of constant-memory self-stabilising election in general networks (Chalopin et al. 2026a). Checked against the full text.

Claude (Anthropic) was used in cataloguing the problems.

21.2 Shape formation in programmable matter

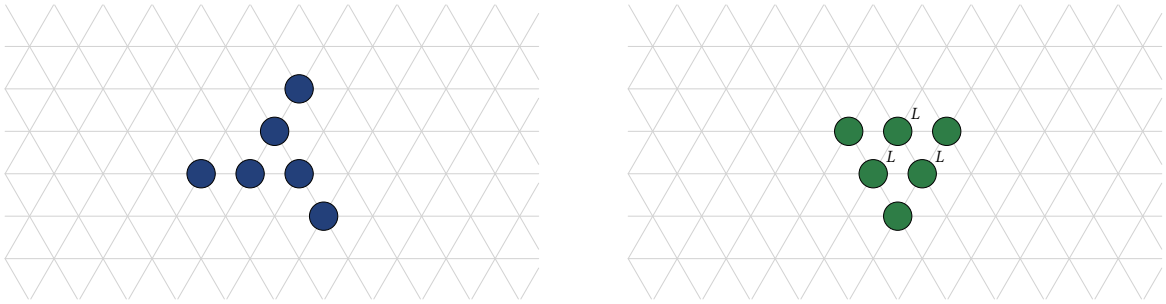
Particles must rearrange into a target shape (a line, a triangle, any shape built from triangles).

First posed in 2015 (Derakhshandeh et al. 2015b). Introduced with leader election at DNA 21 (2015).

Problem statement and measures

From a connected initial **configuration** (the set of nodes occupied by particles) of n particles, the particles must reach a configuration forming a given **target shape** (at a scale depending on n). **Time** is in rounds; **moves** counts expansions and contractions.

Measures: **Time**, asynchronous rounds; **Moves**, total expansions and contractions (work).



(a) Start: a connected configuration

(b) Final: the target triangle is formed

Figure 21.2: Shape formation by the universal algorithm of (Di Luna et al. 2020e) (the problem as in Derakhshandeh et al. 2016), here on a configuration with three-fold symmetry and a triangle as target shape. Erosion (no movement) leaves three candidates, neighbours of each other (green), that cannot break the symmetry, so all three lead. Each leader picks a ray (dashed; the three rays are rotations of each other) and pulls its part into a straight segment along it; then each segment folds into its third of the triangle. A particle moves by one node with an expansion followed by a contraction.

A first look

Think of a pile of clay made of tiny identical machines that should mould itself into a triangle, a hexagon or a line. Each machine is a **particle**. It occupies one **node** of the **triangular lattice**, the infinite tiling of the plane by equilateral triangles whose corners are the nodes; particles on adjacent nodes are **neighbours**. There are n particles, and their nodes form a connected **configuration** with no holes (empty regions enclosed by particles); each particle has a few bits of memory and talks only to its neighbours. A particle moves in two steps: an **expansion** into an empty adjacent node, after which it briefly occupies both nodes, then a **contraction** into one of the two; so one expansion and one contraction move it by one node. In the standard version the particles do not agree on clockwise (they have no **chirality**) or on north, and an adversarial scheduler decides when each particle acts; each time a particle acts is an **activation**. They all know a small description of the **target shape**, and must end up forming it, scaled up so that it uses all n particles. We count **rounds** (stretches of time in which every particle is activated at least once) and **moves** (expansions and contractions).

1. Algorithm: Elect, straighten into a line, rebuild. (Di Luna et al. 2020e) (Deterministic, simply connected start, no chirality, p. 602) The universal algorithm of (Di Luna et al. 2020e) has seven phases; in

Claude (Anthropic) was used in cataloguing the problems.

plain words:

1. Erosion: particles on corners of the configuration drop out one by one (no movement) until 1, 2 or 3 candidates remain, each a neighbour of the others; a candidate may become a **leader**, a particle that directs the others.
2. Each candidate becomes the root of a tree that spans part of the configuration (every particle remembers a parent).
3. The particles agree on chirality, copying it from the candidates.
4. The candidates try to break ties; if they cannot, the starting configuration was symmetric and several leaders carry on together.
5. Each leader pulls its tree into a straight line; all lines get the same length.
6. Walking along its line, each leader works out the scale of the target shape and tells every particle its future role.
7. Each line, guided by its leader, folds into its share of the target shape.

Why it works and what it costs. A line is easy to control: a leader at one end can count along it and send instructions particle by particle. The whole process takes $O(n^2)$ rounds and $O(n^2)$ moves, and the moves are optimal by the next idea. With coin flips, chirality and one particle activated at a time, shapes made of a constant number of triangles are built in $\Theta(\sqrt{n})$ rounds from a well-prepared configuration (Derakhshandeh et al. 2016).

Why it matters. Using a line as a universal halfway shape is the standard trick for shape formation, reused for tiles and modular robots too.

2. Lower bound: A compact configuration must travel far to become a line. (Di Luna et al. 2020e) (Deterministic, simply connected start, no chirality, p. 602) Start with the n particles packed into a hexagon; it fits inside a circle of radius about $\sqrt{n}$ around its centre c. Ask for a straight line. A line has at most $2r+1$ of its particles within distance r of c. Take $r = n/4$: at most about $n/2$ particles of the final line are within distance $n/4$ of c, so at least about $n/2$ particles end farther than $n/4$ from c. Each of them started within about $\sqrt{n}$ of c, so it must travel at least $n/4 - \sqrt{n}$ nodes, one node per expansion and contraction.

Why it holds. Adding up gives at least about $(n/2)(n/4) = n^2/8$ expansion-and-contraction pairs, so $\Omega(n^2)$ moves for any algorithm (Di Luna et al. 2020e). The algorithm above uses $O(n^2)$ moves, so it is optimal in moves. The argument says nothing about rounds, because many particles can move in the same round.

Why it matters. The same distance-counting argument gives the quadratic lower bounds for line formation, coating and centralised transformations.

3. Impossibility: Symmetry that cannot be broken. (Di Luna et al. 2020e) (Deterministic, simply connected start, no chirality, p. 602) Take three particles in a small triangle and ask them to form a straight line of three. The triangle looks the same after a turn by a third of a circle, and its centre is the middle of a triangular face, not a node. Each particle numbers the six directions around it; these local numbers are its **ports**. Let the scheduler always activate the three rotated copies of a particle together, and give them rotated ports. Then they see rotated copies of the same view and make rotated copies of the same move. Because the centre is not a node, two of them never try to enter the same node, so nothing breaks the tie: the configuration stays three-fold symmetric forever. A line of three is not three-fold symmetric, so it is never formed.

Why it holds. In general, a configuration that is symmetric under a turn of a half or a third of a circle around a point that is not a node can only reach configurations with the same symmetry (Di Luna et al.

Claude (Anthropic) was used in cataloguing the problems.

2020e) (Theorem 1). The paper also proves the converse for hole-free configurations: every other target shape can be formed, so this is the only obstacle. With coin flips the obstacle disappears and any shape can be formed once n is large (Di Luna et al. 2020e).

Why it matters. It shows exactly which shapes are out of reach and why, and it explains why many algorithms elect up to three leaders instead of one.

What the community started looking at

The first shapes were lines, triangles and hexagons, built after electing a leader (Derakhshandeh et al. 2015b); then any shape made of triangles (Derakhshandeh et al. 2016). The deterministic question, **which shapes are formable from which starts**, mirrors pattern formation for robots (Di Luna et al. 2020e). Later work formed arbitrary connected shapes with a constant number of particles acting as a mobile RAM (Di Luna et al. 2020d), reconfigured between arbitrary shapes in a linear number of rounds without a canonical intermediate shape (Kostitsyna et al. 2023), and asked which shapes **silent, oblivious** particles can form, starting with the line (Navarra and Piselli 2023b). Most tasks have a natural $\Omega(D)$ lower bound because particles and information move one node at a time; joint movements and reconfigurable circuits are extensions proposed to break it (Padalkin et al. 2025).

Further reading

Papers on this problem that are not yet placed in a setting.

- Artmann et al. (2025). Shape containment, a stationary relative of shape formation: given a shape, the particles must find the largest scale σ (their k_{max}) at which a copy of it, in one of six rotations, fits inside the configuration, and every particle must learn whether it is a valid placement at that scale; with self-disassembly a solution yields shape formation. The particles share a compass and chirality, have constant memory, act in synchronous rounds and use reconfigurable circuits (pins of neighbouring particles linked into circuits on which a beep reaches the whole circuit in one round). Some shapes need $\Omega(\sqrt{n})$ rounds even when σ is known (Theorem 3.1); deterministic algorithms solve star-convex shapes in $O(\log^2 \sigma)$ rounds and the larger class of snowflake shapes in $O(\sqrt{n}\log n)$ rounds (Theorem 6.2), and binary search over the scale works exactly for star-convex shapes (Theorem 5.5). Not placed: the atlas has no shape-formation setting with circuits. Read (arXiv v1).
- Feldmann et al. (2022). The paper that introduced reconfigurable circuits (its leader election is placed under leader election). With circuits, stationary particles reach consensus in $O(1)$ rounds (Theorem 3.1), align their compasses and agree on a chirality in $O(\log n)$ rounds with high probability (Theorem 3.4, Corollary 3.6; chirality needs two pins per bond), and recognise a shape composed of a constant number of triangles in $O(1)$ rounds with a common chirality, $O(\log n)$ otherwise (Theorem 3.8); parallelograms with linear or polynomial side ratio take $\Theta(\log n)$ rounds (Appendix D). Read (arXiv v1).
- Padalkin et al. (2024). Structural problems with reconfigurable circuits, for particles that share a compass and chirality: the particles on an axis through a given particle in $O(\log n)$ rounds (Theorem 9), global maxima in $O(\log^2 n)$ rounds (Theorem 10), a canonical skeleton and a spanning tree in $O(\log^2 n)$ rounds (Theorems 14 and 16) and symmetry detection in $O(\log^5 n)$ rounds (Theorem 18), all with high probability. Read (arXiv v1).
- Padalkin and Scheideler (2026). Shortest path forests with reconfigurable circuits: in a hole-free structure of stationary particles that share a compass and chirality and have a leader, a shortest path tree from one source to ℓ destinations takes $O(\log \ell)$ rounds (Theorem 39) and a forest for a sources $O(\log n \log^2 a)$ rounds (Theorem 56), deterministically (their k is a here); Kostitsyna et al. use such trees to transform shapes. Read (arXiv v1).
- Di Luna et al. (2018). Line recovery: n particles form a line and f of them crash; the $n - f \geq 5$ others, without chirality and under a semi-synchronous scheduler, form one line, or two lines of equal length when the configuration is symmetric (Theorem 4). Read (arXiv v1).
- Derakhshandeh et al. (2015a). A framework for shape formation by particles with constant memory, applied to hexagons and triangles, asymptotically optimal in moves. Abstract read.

Claude (Anthropic) was used in cataloguing the problems.

- Kostitsyna et al. (2022b). Shape formation by particles whose crashes reset their memory and which detect that they crashed and recover. Abstract read.
- Kostitsyna et al. (2022a). Feather trees: shortest-path trees for amoebot structures that remain usable when several of them overlap. A brief announcement; abstract read.
- Porter and Richa (2018). Computation beyond a particle's constant memory, such as counting, distributed over a particle system. Abstract read.

Relations to other problems

- **Related: Pattern formation.** "Shape formation is a basic distributed problem for systems of computational mobile entities. Intensively studied for systems of autonomous mobile robots, it has recently been investigated in the realm of programmable matter" (Di Luna et al. 2020e).
- **Related: Shape formation by hybrid programmable matter.** In the hybrid model the tiles are passive and one active agent moves them; in the amoebot model every particle is active.
- **Related: Centralised transformations of modular robots.** A central controller plans the moves, so the question is which mechanisms are universal and how many moves they need, not how particles coordinate.

Case tree

The root assumes nothing; going down, each edge adds the assumptions written in grey (+) or leaves an axis unassumed. Bold entries are catalogued settings with their status per measure (Time, Moves) and page; italic entries are branch points, named after the axis on which their branches differ. Axis order: Memory → Scheduler → Communication → Orientation → Randomness → Initial shape → Connectivity during execution.

No assumptions

+ constant memory

Scheduler

+ strong scheduler, + no holes

Deterministic, simply connected start, no chirality STANDARD SETTING ● ● p. 602

+ sequential, + chirality, + randomized, + no holes

Universal shapes, sequential, randomized ● ● p. 602

oblivious, + chirality

SILBOT, chirality ● ● p. 603

oblivious, + common compass, + may disconnect

SILBOT, full orientation ● ● p. 603

What is known

The best upper bound against the best lower bound in each catalogued setting, in tree order. The settings follow in detail.

Setting	Time	Moves	p.
Deterministic, simply connected start, no chirality STANDARD SETTING	● unknown $O(n^2)$ vs —	● tight $O(n^2)$ vs $\Omega(n^2)$	602
Universal shapes, sequential, randomized	● tight $O(\sqrt{n})$ vs $\Omega(\sqrt{n})$	● unknown — vs —	602

Claude (Anthropic) was used in cataloguing the problems.

Setting	Time	Moves	p.
SILBOT, chirality	● impossible	● impossible	603
SILBOT, full orientation	● unknown — vs —	● tight $O(n^2)$ vs $\Omega(n^2)$	603

21.2.1 Deterministic, simply connected start, no chirality

Adds + strong scheduler, + no holes STANDARD SETTING VERIFIED

Which shapes are deterministically formable from a simply connected start is characterised; $O(n^2)$ rounds and moves, and the moves are optimal.

Assumed: memory: constant memory and local communication; scheduler: strong: atomic activations under a fair adversary; initial shape: simply connected (no holes). **Not assumed:** communication: local: only between neighbouring particles; orientation: no common orientation and no chirality; randomness: deterministic; connectivity during execution: the system must stay connected.

Measure	Lower bound	Best upper bound	Status
Time	none known No time lower bound recorded.	$O(n^2)$ (Di Luna et al. 2020e)	● unknown
Moves	$\Omega(n^2)$ (Di Luna et al. 2020e) $\Theta(n^2)$ moves.	$O(n^2)$ (Di Luna et al. 2020e)	● tight

Results.

- Di Luna et al. (2020e) [lower bound]: moves $\Omega(n^2)$. *Bottleneck:* distance. *Holds when:* some initial configurations and shapes.
- Di Luna et al. (2020e) [current best]: time $O(n^2)$, moves $O(n^2)$. Introductory for the deterministic, chirality-free setting and current best; "chirality is computationally irrelevant for shape formation"; with randomization any shape is formable for large n.

21.2.2 Universal shapes, sequential, randomized

Adds + sequential, + chirality, + randomized, + no holes VERIFIED

Any shape made of a constant number of triangles, from a well-initialised start, in $\Theta(\sqrt{n})$ rounds.

Assumed: memory: constant memory and local communication; scheduler: sequential: one particle active at a time; orientation: common chirality; randomness: randomized; initial shape: simply connected (no holes). **Not assumed:** communication: local: only between neighbouring particles; connectivity during execution: the system must stay connected.

Measure	Lower bound	Best upper bound	Status
Time	$\Omega(\sqrt{n})$ (Derakhshandeh et al. 2016) $\Theta(\sqrt{n})$.	$O(\sqrt{n})$ (Derakhshandeh et al. 2016)	● tight
Moves	none known —	—	● unknown

Results.

- Derakhshandeh et al. (2016) [lower bound]: time $\Omega(\sqrt{n})$. *Bottleneck:* distance. *Holds when:* when the shape differs from the initial configuration.
- Derakhshandeh et al. (2016) [current best]: time $O(\sqrt{n})$. *Assumes:* well-initialised configuration. Introductory and current best.

Claude (Anthropic) was used in cataloguing the problems.

Notes. The model assumptions (chirality, sequential activation, randomization) are as described by (Di Luna et al. 2020e).

21.2.3 SILBOT, chirality

Adds oblivious, + chirality VERIFIED

Silent oblivious particles cannot form a line if they share only chirality.

Assumed: orientation: common chirality. **Not assumed:** memory: oblivious, silent particles (SILBOT); scheduler: weak: non-atomic activations, an action may interleave with others; communication: local: only between neighbouring particles; randomness: deterministic; initial shape: connected, holes allowed; connectivity during execution: the system must stay connected.

Statement for this setting. The target shape is a line.

Measure	Lower bound	Best upper bound	Status
Time	*impossible* (Navarra and Piselli 2023b) Unsolvable with chirality alone (Navarra and Piselli 2023b).	—	• impossible
Moves	none known —	—	• impossible

Results.

- Navarra and Piselli (2023b) [lower bound]: time *impossible*. *Bottleneck:* symmetry breaking. *Holds when:* a symmetric initial configuration; a synchronous adversary; holds even with common chirality.

Open questions.

Q1. Is line formation solvable in SILBOT when particles share just one direction instead of the full orientation (Navarra and Piselli 2023b)?

21.2.4 SILBOT, full orientation

Adds oblivious, + common compass, + may disconnect VERIFIED

With a common compass, silent oblivious particles form a line with $O(n^2)$ moves, which is optimal.

Assumed: orientation: full common orientation (compass); connectivity during execution: the system may disconnect temporarily. **Not assumed:** memory: oblivious, silent particles (SILBOT); scheduler: weak: non-atomic activations, an action may interleave with others; communication: local: only between neighbouring particles; randomness: deterministic; initial shape: connected, holes allowed.

Statement for this setting. The target shape is a line.

Measure	Lower bound	Best upper bound	Status
Time	none known No round bound recorded.	—	• unknown
Moves	$\Omega(n^2)$ (Derakhshandeh et al. 2015b) $\Theta(n^2)$ moves.	$O(n^2)$ (Navarra and Piselli 2023b)	• tight

Results.

- Derakhshandeh et al. (2015b) [lower bound]: moves $\Omega(n^2)$. *Bottleneck:* distance. *Holds when:* Lemma 7: a compact start of diameter at most $2\sqrt{n}+2$ must become a line; holds for any algorithm.

Claude (Anthropic) was used in cataloguing the problems.

- Navarra and Piselli (2023b) [current best]: moves $O(n^2)$. *Assumes:* 2-hop view. Introductory and current best.
- Navarra et al. (2025) [also]: moves *finite*. Journal version; adds configurations with immovable obstacles, and scattering.

Open questions.

Q1. Does 1-hop visibility suffice for line formation in SILBOT (Navarra and Piselli 2023b)?

Q2. Which other shapes can SILBOT particles form (Navarra and Piselli 2023b)?

Notes. The model is SILBOT (D'Angelo et al. 2020): no memory, no messages, non-atomic asynchronous actions, view of two hops. The algorithm (WRain) disconnects the system temporarily. Checked against the full text of (Navarra and Piselli 2023b).

21.3 Coating

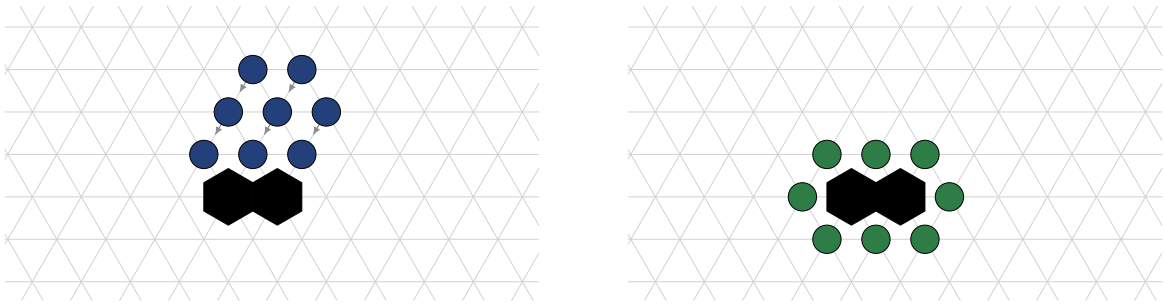
Particles must cover a fixed object as uniformly as possible: in layers, each filled before the next.

First posed in 2014 (Derakhshandeh et al. 2017). Infinite object coating appeared as an arXiv preprint in 2014; universal coating in (Derakhshandeh et al. 2017).

Problem statement and measures

A fixed **object** (a connected set of nodes of the triangular lattice that particles cannot enter) is surrounded by particles. They must arrange themselves into **layers** around the object, each layer filled before the next. **Time** in rounds; **work** is the number of moves (expansions and contractions).

Measures: **Time**, asynchronous rounds; **Moves**, total expansions and contractions (work).



(a) Start: roots touch the object; arrows point to parents

(b) Final: the first layer is filled

Figure 21.3: Coating, following (Derakhshandeh et al. 2017): particles that neighbour the object (black) are roots, the others follow a parent closer to the object (forming trees). Roots walk clockwise along the first layer, one expansion and one contraction per node; when an expanded particle contracts, its child expands into the node it leaves (a handover), so the trees are pulled along without breaking. With as many particles as first-layer nodes, the walk ends when every node that neighbours the object is occupied (green); further particles would start the next layer.

Claude (Anthropic) was used in cataloguing the problems.

A first look

Imagine ants covering a crumb, one even layer at a time. The floor is the **triangular lattice**: the endless graph whose **nodes** are the corners of a tiling of the plane by equal triangles; each node is joined by an edge to six others, its **neighbours**. A **particle** is a tiny machine that occupies one node; it moves in two actions: an **expansion** into an empty neighbouring node, after which it is **expanded** and occupies both, then a **contraction** into one of the two; so one expansion and one contraction move it by one node, and each of them is one **move**. The **object** is a connected set of nodes that particles cannot enter; n particles, connected to each other and to the object, sit around it. They must arrange themselves into **layers** around the object: the first layer (the nodes that neighbour the object) must be full before any particle sits in the second, and so on, with any extra particles in the last layer. In the standard version particles have constant memory, agree on which way is clockwise (**chirality**), and may flip coins; we count **rounds** (intervals of time in which every particle is activated at least once) and work (the total number of moves).

1. Algorithm: Follow the leader around the object, layer by layer. (Derakhshandeh et al. 2017) (Universal coating, p. 607) The particles move in **handovers**: an expanded particle contracts and at the same moment pulls a neighbour into the node it vacates, so the chain of particles never breaks.

1. Particles that neighbour the object become **roots**; every other particle picks a neighbour closer to the object as its *parent*, so the particles form trees hanging off the object.
2. Roots walk clockwise around the object along the first layer; followers trail behind their parents by handovers.
3. A root that bumps into a particle that has already stopped stops too (it *retires*). A follower that still wants a node in the layer sends a 'complaint' to a root with free space ahead, which then moves on and makes room.
4. A leader, elected among the particles of the first layer, marks where the layer begins and ends. When the first layer is full, the next layer starts from that mark, and so on.

Why it works and what it costs. In the worst case the total work is $\Theta(n^2)$, which matches the lower bound below (Derakhshandeh et al. 2017). The running time is $O(n)$ rounds with high probability (Daymude et al. 2018); the coins are needed only for the leader election, and $O(n)$ is the best possible in the worst case. Particles with no memory and no messages can still coat a convex object with one layer, in $O(n^2)$ rounds (Navarra and Piselli 2023a).

Why it matters. Spanning trees plus handovers became the standard way to move all the particles without ever disconnecting them.

2. Lower bound: A long tail of particles. (Daymude et al. 2018) (Universal coating, p. 607) Put all n particles in one straight line that meets the object only at its first particle, and let the first layer have more than n nodes. The particle at distance i from the object must end as a neighbour of it, so it must travel about i nodes, which takes about $2i$ moves. The last particle is at distance n, and an unkind scheduler may let it act only once per round.

Why it holds. The last particle needs about n moves at one move per round, so every algorithm needs $\Omega(n)$ rounds in the worst case (Daymude et al. 2018). Adding the distances, $2(1 + 2 + \cdots + (n-1)) \approx n^2$, so every algorithm needs $\Omega(n^2)$ work (Derakhshandeh et al. 2017). Both bounds are matched by the algorithm above.

Why it matters. It is the simplest example of the distance argument that underlies most lower bounds for programmable matter.

Claude (Anthropic) was used in cataloguing the problems.

3. Lower bound: Local particles cannot see where the gap is. (Daymude et al. 2018) (Universal coating, p. 607) Let the object be a long straight bar. The first layer is completely full except for one empty node exactly in the middle below the bar, and one extra particle sits in the second layer exactly in the middle above it. Someone who sees everything solves it in a constant number of rounds: every particle in the first layer shifts by one node around the bar, closing the gap, and the extra particle drops in. Local particles near the two ends of the bar cannot tell, within about $n/4$ rounds, which way the shift should go: information moves at most one node per round, and the situation looks the same from both ends.

Why it holds. So for about $n/4$ rounds the flows at the two ends are decided independently and by symmetry are equally likely to go either way; in the cases where they do not fit together, the layer is not yet correct. Hence any local algorithm is, on such inputs, about n times slower than the best algorithm with global knowledge: a competitive ratio of $\Omega(n)$ (Daymude et al. 2018) (Theorem 1).

Why it matters. It shows that 'worst case over all inputs' is the right measure here, since comparing to the best possible run on each input is hopeless for local algorithms.

What the community started looking at

Coating was posed as a basic task for self-organising matter with **work** (moves) as the first measure (Derakhshandeh et al. 2017); rounds followed, with matching bounds (Daymude et al. 2018). Coating was later solved by silent, oblivious particles (Navarra and Piselli 2023a) and in 3D hybrid programmable matter (Kostitsyna et al. 2024).

Case tree

The root assumes nothing; going down, each edge adds the assumptions written in grey (+) or leaves an axis unassumed. Bold entries are catalogued settings with their status per measure (Time, Moves) and page; italic entries are branch points, named after the axis on which their branches differ. Axis order: Memory → Scheduler → Communication → Orientation → Randomness → Initial shape → Connectivity during execution.

No assumptions

+ constant memory, + strong scheduler, + chirality, + randomized
Universal coating STANDARD SETTING ●● p. 607

oblivious, weak scheduler, + chirality
SILBOT, chirality, deterministic ●● p. 607

What is known

The best upper bound against the best lower bound in each catalogued setting, in tree order. The settings follow in detail.

Setting	Time	Moves	p.
Universal coating STANDARD SETTING	● tight $O(n)$ vs $\Omega(n)$	● tight $O(n^2)$ vs $\Omega(n^2)$	607
SILBOT, chirality, deterministic	● tight $O(n^2)$ vs $\Omega(n^2)$	● unknown — vs —	607

Claude (Anthropic) was used in cataloguing the problems.

21.3.1 Universal coating

Adds + constant memory, + strong scheduler, + chirality, + randomized STANDARD SETTING VERIFIED

Any object admitting a uniform coating is coated with optimal work, and in $\Theta(n)$ rounds with high probability.

Assumed: memory: constant memory and local communication; scheduler: strong: atomic activations under a fair adversary; orientation: common chirality; randomness: randomized. **Not assumed:** communication: local: only between neighbouring particles; initial shape: connected, holes allowed; connectivity during execution: the system must stay connected.

Measure	Lower bound	Best upper bound	Status
Time	$\Omega(n)$ (Daymude et al. 2018) $\Theta(n)$ rounds.	$O(n)$ (Daymude et al. 2018)	• tight
Moves	$\Omega(n^2)$ (Derakhshandeh et al. 2017) $\Theta(n^2)$ moves.	$O(n^2)$ (Derakhshandeh et al. 2017)	• tight

Results.

- Daymude et al. (2018) [lower bound]: time $\Omega(n)$. *Bottleneck:* locality. *Holds when:* Lemma 1: deterministic worst case; a line of n particles attached to the object by one particle, the object's boundary longer than n.
- Derakhshandeh et al. (2017) [lower bound]: moves $\Omega(n^2)$. *Bottleneck:* distance. *Holds when:* Lemma 15.
- Derakhshandeh et al. (2017) [introductory]: moves $O(n^2)$. Worst-case work-optimal: $\Theta(n^2)$ moves (Theorem 2).
- Daymude et al. (2018) [current best]: time $O(n)$. With high probability; randomisation enters only through the leader election.

Notes. Chirality is part of the model of both papers. In 3D hybrid programmable matter (one finite-automaton agent moving passive tiles), universal coating takes $O(n^2)$ steps, worst-case optimal against an agent with global knowledge (Kostitsyna et al. 2024). Checked against the full texts.

21.3.2 SILBOT, chirality, deterministic

Adds oblivious, weak scheduler, + chirality VERIFIED

Silent oblivious particles with chirality coat a convex object with one layer, deterministically, in $O(n^2)$ rounds.

Assumed: orientation: common chirality. **Not assumed:** memory: oblivious, silent particles (SILBOT); scheduler: weak: non-atomic activations, an action may interleave with others; communication: local: only between neighbouring particles; randomness: deterministic; initial shape: connected, holes allowed; connectivity during execution: the system must stay connected.

Statement for this setting. Single-layer coating of a **convex** object, together with compaction of holes.

Measure	Lower bound	Best upper bound	Status
Time	$\Omega(n^2)$ (Navarra and Piselli 2023a) $\Theta(n^2)$ rounds; the lower bound is argued informally.	$O(n^2)$ (Navarra and Piselli 2023a)	• tight
Moves	none known —	—	• unknown

Results.

- Navarra and Piselli (2023a) [lower bound]: time $\Omega(n^2)$. *Bottleneck:* oblivious silent particles. *Holds when:* informal argument before Theorem 4, not a theorem.

Claude (Anthropic) was used in cataloguing the problems.

- Navarra and Piselli (2023a) [current best]: time $O(n^2)$. *Assumes:* 2-hop view; exterior awareness: a particle can tell a hole from the outside. Introductory and current best.

Open questions.

Q1. Coating in SILBOT without chirality (Navarra and Piselli 2023a).

Q2. Multi-layer coating, which "seems unfeasible" in SILBOT (Navarra and Piselli 2023a).

Notes. Checked against the full text.

21.4 Compression

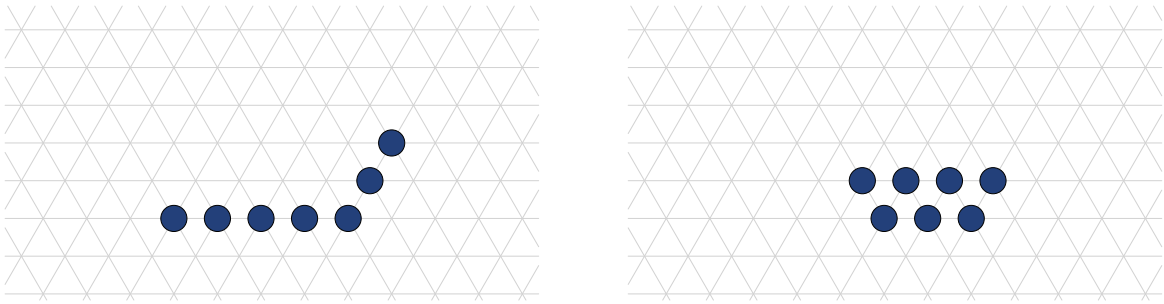
Particles following a local stochastic rule must gather into a configuration with small perimeter.

First posed in 2016 (Cannon et al. 2016). PODC 2016.

Problem statement and measures

Starting connected, the n particles perform a **Markov chain** (at each step one particle, chosen at random, proposes a move and accepts it with a probability that depends only on its neighbourhood) with **bias** λ towards having more neighbours. The system is α-**compressed** if its **perimeter** is at most α times the minimum possible.

Measures: **Time**, asynchronous rounds; **Moves**, total expansions and contractions (work).



(a) Start: a long chain, long perimeter

(b) Final: compressed, short perimeter

Figure 21.4: Compression by the Markov chain of (Cannon et al. 2016): one particle at a time (outlined) expands into a random free neighbouring node and then either contracts forward, with probability $\lambda^{e'-e}$ (e and e' its neighbours before and after; certainly if $e' \geq e$), or contracts back; moves that would disconnect the system or create a hole are refused. For $\lambda > 2 + \sqrt{2}$ the system becomes compressed (perimeter at most α times the minimum) with high probability; for $\lambda < 2.17$ it does not.

A first look

Picture a crowd of penguins huddling together for warmth, each one just shuffling towards places with more neighbours. The floor is the **triangular lattice**: the endless graph whose **nodes** are the corners of a tiling of the plane by equal triangles; each node is joined by an edge to six others, its **neighbours**. A **particle** occupies one node and moves in two actions: an **expansion** into an empty neighbouring

Claude (Anthropic) was used in cataloguing the problems.

node, after which it occupies both, then a **contraction** into one of the two; so one expansion and one contraction move it by one node. The **system** of n particles starts connected. One particle at a time, chosen at random, is activated (an **activation**) and makes a random move; particles have no names, a few bits of memory, and no compass. The goal is to become **compressed**: the **perimeter** (the length of the walk around the outside of the system) should be at most α times the smallest possible perimeter for n particles, for a constant $\alpha > 1$. The question is not how fast this happens, but whether, after running long enough, the system is compressed with very high probability.

1. Algorithm: Move at random, but prefer more neighbours. (Cannon et al. 2016) (Markov chain, sequential, p. 610) Fix a number $\lambda > 0$, the **bias**. A randomly chosen particle does the following.

1. Pick one of its six directions at random. If that node is empty, expand into it.
2. Count e, its neighbours at the old node, and e', the neighbours it would have at the new node.
3. Check, by looking only at nearby nodes, that the move would neither split the system nor enclose a **hole** (a region of empty nodes surrounded by particles). If it would, contract back.
4. Otherwise contract into the new node with probability $\lambda^{e'-e}$ (or with certainty if this number exceeds 1); else contract back.

So moves that gain neighbours are always accepted when $\lambda > 1$, and moves that lose neighbours are accepted only sometimes.

Why it works and what it costs. In the long run, each **configuration** (set of occupied nodes) appears with probability proportional to λ raised to its number of pairs of neighbours, and fewer pairs of neighbours means a longer perimeter. For a large bias, $\lambda > 2 + \sqrt{2} \approx 3.41$, the system is compressed with all but exponentially small probability (Cannon et al. 2016). For a small bias, $\lambda < 2.17$, it is the opposite: the system stays stretched out, with perimeter a constant fraction of the largest possible, even though each particle still prefers neighbours. Between 2.17 and 3.41 the paper's theorems say nothing.

Why it matters. This 'local random rule with a bias' recipe became the template for separation, alignment and bridging.

2. Lower bound: Too many stretched configurations: why a weak bias fails. (Cannon et al. 2016) (Markov chain, sequential, p. 610) Two forces compete. The bias favours each single compact configuration over each single stretched one: a configuration with perimeter p gets weight roughly λ^{-p}. But stretched configurations are far more numerous. The number of configurations with perimeter p grows roughly like $(2 + \sqrt{2})^p$, because the walk around the outside never crosses itself on the honeycomb lattice (the hexagonal lattice of the triangles' centres), and such walks grow at exactly that rate. Multiply: all configurations of perimeter p together weigh about $\big((2 + \sqrt{2})/\lambda\big)^p$.

Why it holds. If $\lambda > 2 + \sqrt{2}$, this total shrinks exponentially as p grows, so long perimeters are very unlikely and the system compresses. If λ is small, the sheer number of long-perimeter configurations outweighs the bias, and (Cannon et al. 2016) prove that for $\lambda < 2.17$ the perimeter stays at least a constant fraction of its maximum. This is the same balance of energy against counting that physicists use for phase transitions.

Why it matters. It explains why 'prefer neighbours' is not enough on its own and how strong the preference must be.

What the community started looking at

Compression introduced the **stochastic** approach to programmable matter, where the question is which bias produces which emergent behaviour (Cannon et al. 2016).

Claude (Anthropic) was used in cataloguing the problems.

Further reading

Papers on this problem that are not yet placed in a setting.

- Oh et al. (2022). Foraging: particles must gather and compress around a food particle while it stays, and disperse to search when it disappears or moves, over and over. A stochastic algorithm built on the compression chain of Cannon et al. reaches an α-compressed configuration around the food at stationarity with high probability (Theorem 1), and a structured one builds a spiral around the food (Theorem 2); a token wave that disperses the cluster outpaces the wave that compresses it. Read (arXiv v1, the full version).

Relations to other problems

- **Derived problem:** **Separation of coloured particles** (generalisation of). Separation adds two colours to compression; the same local chain compresses and, depending on a second bias, separates or integrates the colours (Cannon et al. 2019).
- **Derived problem:** **Alignment** (generalisation of). Alignment adds an orientation to each particle; the same chains also give compression and expansion (Kedia et al. 2022).
- **Derived problem:** **Bridging** (generalisation of). Bridging adds terrain: the compression chain is biased to stay on land and to shorten a path across a gap (Arroyo et al. 2018).

Case tree

The root assumes nothing; going down, each edge adds the assumptions written in grey (+) or leaves an axis unassumed. Bold entries are catalogued settings with their status per measure (Time, Moves) and page; italic entries are branch points, named after the axis on which their branches differ. Axis order: Memory → Scheduler → Communication → Orientation → Randomness → Initial shape → Connectivity during execution.

No assumptions

+ constant memory, + sequential, + randomized

Markov chain, sequential STANDARD SETTING ●● p. 610

What is known

The best upper bound against the best lower bound in each catalogued setting, in tree order. The settings follow in detail.

Setting	Time	Moves	p.
Markov chain, sequential STANDARD SETTING	● unknown *eventually* vs —	● unknown — vs —	610

21.4.1 Markov chain, sequential

Adds + constant memory, + sequential, + randomized STANDARD SETTING VERIFIED

For $\lambda > 2 + \sqrt{2}$ the system eventually compresses; for $\lambda < 2.17$ it expands.

Assumed: memory: constant memory and local communication; scheduler: sequential: one particle active at a time; randomness: randomized. **Not assumed:** communication: local: only between neighbouring particles; orientation: no common orientation and no chirality; initial shape: connected, holes allowed; connectivity during execution: the system must stay connected.

Claude (Anthropic) was used in cataloguing the problems.

Measure	Lower bound	Best upper bound	Status
Time	none known Mixing time not bounded.	*eventually* (Cannon et al. 2016)	• unknown
Moves	none known —	—	• unknown

Results.

- Cannon et al. (2016) [current best]: time *eventually*. Introductory and current best; with all but exponentially small probability.

Open questions.

Q1. Behaviour for $2.17 \leq \lambda \leq 2 + \sqrt{2}$, and bounds on the convergence time.

21.5 Separation of coloured particles

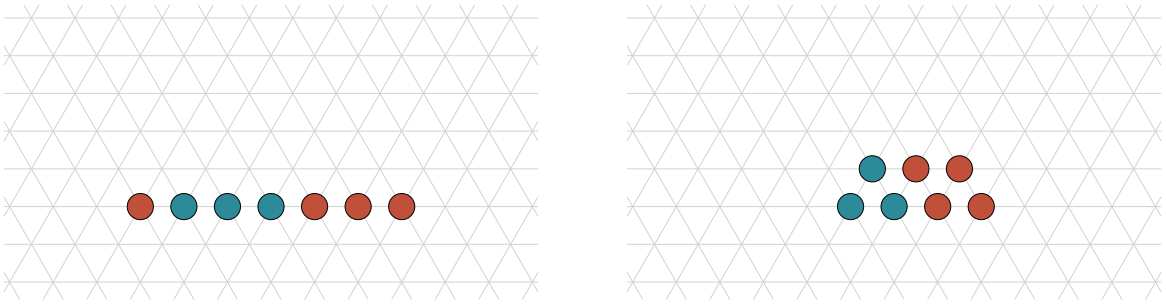
Particles of two colours, following a local stochastic rule, must cluster by colour (or mix).

First posed in 2018 (Cannon et al. 2019). arXiv 2018; APPROX/RANDOM 2019.

Problem statement and measures

Each particle has one of two colours. Particles run a local **Markov chain** with a **bias** λ towards more neighbours and a bias γ towards neighbours of the same colour. The system should be **compressed** and **separated** (few pairs of neighbours of different colours), or **integrated** (colours mixed), with high probability in the stationary distribution.

Measures: **Time**, asynchronous rounds; **Moves**, total expansions and contractions (work).



(a) Start: a chain, colours mixed

(b) Final: compressed and separated

Figure 21.5: Separation by the Markov chain of (Cannon et al. 2019): one particle at a time (outlined) expands into a free neighbouring node and contracts forward with probability $\lambda^{\Delta e}\gamma^{\Delta h}$ (Δe, Δh: the change in its neighbours and in its neighbours of its own colour; certainly if the product is at least 1), and otherwise contracts back; moves that would disconnect the system or create a hole are refused. The chain also lets neighbours of different colours swap places (not shown). For large λ and γ the system becomes compressed and separated with high probability; for γ close to 1 it stays integrated.

MATTER

Claude (Anthropic) was used in cataloguing the problems.

A first look

Think of a mixed crowd of red and blue marbles in a tray that, when shaken, sorts itself into a red patch and a blue patch, or stays mixed, depending on how 'sticky' same-coloured marbles are. The floor is the **triangular lattice**: the endless graph whose **nodes** are the corners of a tiling of the plane by equal triangles; each node is joined by an edge to six others, its **neighbours**. A **particle** occupies one node and moves in two actions: an **expansion** into an empty neighbouring node, after which it occupies both, then a **contraction** into one of the two; so one expansion and one contraction move it by one node. The **system** of n particles starts connected; each particle carries a fixed colour, one of two, has constant memory and no compass, and one randomly chosen particle acts at a time. The system should become **compressed** (its **perimeter**, the length of the walk around its outside, is at most a constant times the smallest possible, as in compression) and either **separated** (each colour mostly in one region, with few pairs of neighbours of different colours) or **integrated** (colours mixed), whichever is asked for. As for compression, the question is what the system looks like after running for a long time, with high probability.

1. Algorithm: The compression rule with a second bias for colour. (Cannon et al. 2019) (Markov chain, sequential, p. 613) There are two **biases**: λ rewards neighbours of any colour, and γ rewards neighbours of one's own colour. A randomly chosen particle does one of two things.

1. **Step.** As in compression: pick a random empty neighbouring node; check locally that moving there neither splits the system nor encloses a **hole** (a region of empty nodes surrounded by particles); then move with probability $\lambda^{(\text{gain in neighbours})} \cdot \gamma^{(\text{gain in same-coloured neighbours})}$ (certainly, if this number exceeds 1).
2. **Swap.** Pick a neighbour of the other colour and exchange nodes with it, with probability $\gamma^{(\text{gain in same-coloured neighbours of the two})}$ (certainly, if above 1). Swaps let colours move through the middle of the system.

Why it works and what it costs. With only one colour the rule is exactly the compression rule (Cannon et al. 2016). For a strong colour bias, $\gamma > 4^{5/4} \approx 5.66$ together with $\lambda\gamma > 6.83$ (and $\lambda > 1$), the system is compressed and separated with high probability (Cannon et al. 2019). For γ very close to 1, between $79/81$ and $81/79$, and λ large enough, it is compressed but integrated. So one rule does both jobs, and a single number set by the environment decides which.

Why it matters. It showed that the biased-rule recipe extends from shape to internal organisation, here the arrangement of types.

2. Lower bound: A mild colour preference does not separate the colours. (Cannon et al. 2019) (Markov chain, sequential, p. 613) One might expect any preference $\gamma > 1$ for same-coloured neighbours to separate the colours eventually. It does not. There are vastly more mixed arrangements of the colours inside a compressed system than separated ones: separation means fixing a short border between the colours, while mixing leaves almost every particle free to be either colour. A slight preference gives each separated arrangement only a slightly larger weight, which cannot make up for the huge number of mixed ones.

Why it holds. (Cannon et al. 2019) prove that for γ between $79/81$ and $81/79$, which includes values slightly above 1, the system fails to separate with high probability: the number of pairs of neighbours of different colours stays large. Separation needs the colour bias to beat this counting, which is why the positive result asks for $\gamma > 5.66$. What happens for γ between these ranges is not settled by the paper.

Why it matters. It is the colour version of the compression lesson: a preference must be strong enough to beat the number of disordered states.

Claude (Anthropic) was used in cataloguing the problems.

What the community started looking at

Separation extended the stochastic approach of compression to **heterogeneous** systems; the question is which bias parameters give which emergent behaviour (Cannon et al. 2019).

Relations to other problems

- **Generalisation of Compression.** Separation adds two colours to compression; the same local chain compresses and, depending on a second bias, separates or integrates the colours (Cannon et al. 2019).
- **Related: Separation.** Robots in the plane separate by hidden colours with deterministic algorithms; here a stochastic local rule separates two visible colours.

Case tree

The root assumes nothing; going down, each edge adds the assumptions written in grey (+) or leaves an axis unassumed. Bold entries are catalogued settings with their status per measure (Time, Moves) and page; italic entries are branch points, named after the axis on which their branches differ. Axis order: Memory → Scheduler → Communication → Orientation → Randomness → Initial shape → Connectivity during execution.

No assumptions

+ constant memory, + sequential, + randomized

Markov chain, sequential STANDARD SETTING ●● p. 613

What is known

The best upper bound against the best lower bound in each catalogued setting, in tree order. The settings follow in detail.

Setting	Time	Moves	p.
Markov chain, sequential STANDARD SETTING	● unknown *eventually* vs —	● unknown — vs —	613

21.5.1 Markov chain, sequential

Adds + constant memory, + sequential, + randomized STANDARD SETTING VERIFIED

One local algorithm separates or integrates the colours depending on a single global parameter.

Assumed: memory: constant memory and local communication; scheduler: sequential: one particle active at a time; randomness: randomized. **Not assumed:** communication: local: only between neighbouring particles; orientation: no common orientation and no chirality; initial shape: connected, holes allowed; connectivity during execution: the system must stay connected.

Measure	Lower bound	Best upper bound	Status
Time	none known Mixing time not bounded.	*eventually* (Cannon et al. 2019)	● unknown
Moves	none known —	—	● unknown

Claude (Anthropic) was used in cataloguing the problems.

Results.

- Cannon et al. (2019) [current best]: time *eventually*. Introductory and current best; proofs use cluster expansions and the high-temperature expansion of the Ising model.

Open questions.

Q1. Bounds on the convergence (mixing) time.

21.6 Alignment

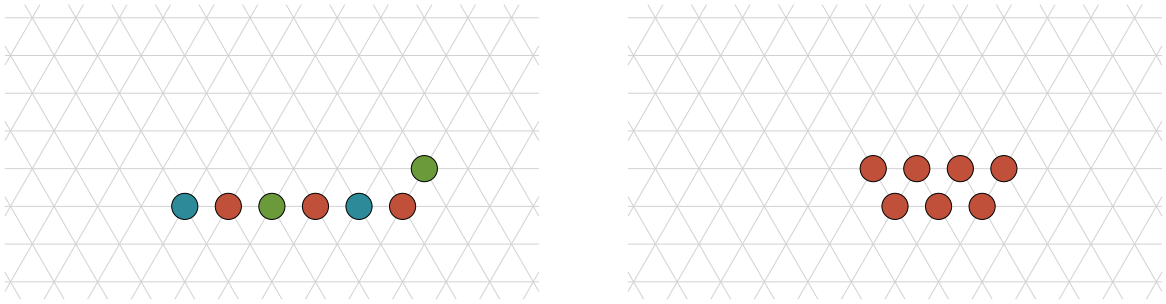
Oriented particles, following a local stochastic rule, must align their orientations.

First posed in 2022 (Kedia et al. 2022). APPROX/RANDOM 2022.

Problem statement and measures

Each particle carries one of $q \geq 2$ **orientations**, a spin state, not a sense of direction. Particles run a local **Markov chain**; the system should be **aligned** (most particles share an orientation) or not, depending on the **bias**, with high probability in the stationary distribution, both when connectivity is kept and when particles may disconnect.

Measures: **Time**, asynchronous rounds; **Moves**, total expansions and contractions (work).



(a) Start: orientations mixed

(b) Final: compressed, one orientation shared

Figure 21.6: Alignment by the Markov chain of (Kedia et al. 2022), connected setting: each particle carries one of q orientations (here $q = 3$, shown as three colours). One particle at a time (outlined) either steps, expanding into a free neighbouring node and contracting forward with a probability that grows with its gain in neighbours (λ) and in agreeing neighbours (γ), or turns to a new orientation, accepted with a probability that grows with its gain in agreeing neighbours; moves that would disconnect the system or create a hole are refused. Shown is a run in which every proposed move is accepted. For large biases most particles end up sharing one orientation, with high probability; for small biases they do not.

A first look

Picture a school of fish or a flock of birds in which every animal turns to face the same way as its neighbours. The floor is the **triangular lattice**: the endless graph whose **nodes** are the corners of a tiling of the plane by equal triangles; each node is joined by an edge to six others, its **neighbours**. A **particle** occupies one node and moves in two actions: an **expansion** into an empty neighbouring node, after which it occupies both, then a **contraction** into one of the two; so one expansion and one

Claude (Anthropic) was used in cataloguing the problems.

contraction move it by one node. The **system** of n particles starts connected, and each particle carries an **orientation**, one of $q \geq 2$ values drawn as an arrow; the orientation is just a state, not a compass, and a particle can only compare its orientation with its neighbours' orientations. One randomly chosen particle acts at a time; it either steps (moves to a neighbouring node) or turns (changes its orientation). The system should become **aligned** (one orientation is held by most particles) or stay *non-aligned* (all orientations about equally common), depending on a bias; this is studied both when the system must stay connected and when it may fall apart.

1. Algorithm: Step or turn, with a bias towards agreeing neighbours. (Kedia et al. 2022) (Markov chain, sequential, p. 616) In the connected setting there are two **biases**: λ rewards having neighbours, as in compression, and a second bias, written γ in (Kedia et al. 2022), rewards **agreeing neighbours** (neighbours with the same orientation). A randomly chosen particle does one of two things, with equal chance.

1. **Step.** Pick a random empty neighbouring node; check locally that the system stays connected without **holes** (regions of empty nodes surrounded by particles); move with a probability that grows with the gain in neighbours and in agreeing neighbours.

2. **Turn.** Pick a new orientation; accept it with a probability that grows with the gain in agreeing neighbours (certainly, if it gains).

Why it works and what it costs. With both biases large (for instance $\gamma > 29.3(q-1)$ and $\lambda\gamma$ large enough), the system is **compressed** (its perimeter, the length of the walk around its outside, is at most a constant times the minimum) with high probability, and under further conditions on γ one orientation dominates (Kedia et al. 2022). With λ large but γ small, the system is compressed but its orientations stay evenly split among the q values; with λ small, the system stays stretched out. Compactness matters: a long thin bottleneck lets two regions with different orientations coexist cheaply, so alignment is proved only for well compressed systems.

Why it matters. It brings the Potts and clock models of magnetism into programmable matter, with particles that move as well as turn.

2. Algorithm: When particles may drift apart: one knob. (Kedia et al. 2022) (Markov chain, sequential, p. 616) If the system is allowed to disconnect, the connectivity check is dropped: a particle may step to any empty neighbouring node. A single bias then rewards, at the same time, having neighbours and having agreeing neighbours.

1. Pick step or turn with equal chance.

2. Step: move to a random empty neighbouring node, accepted with a probability that grows with the gain in (agreeing) neighbours.

3. Turn: as before.

Why it works and what it costs. When the single bias is large, the particles gather into compact **clusters** (sets of particles connected through neighbours) and align; when it is small, they disperse and the orientations stay balanced (Kedia et al. 2022). The proof is harder than in the connected case because a spread-out system can contain many small clusters scattered over the lattice.

Why it matters. It shows that the stochastic approach does not need the connectivity rule, which is costly to check and not always realistic.

Claude (Anthropic) was used in cataloguing the problems.

What the community started looking at

Alignment brought Potts and clock models from statistical physics to programmable matter (Kedia et al. 2022).

Relations to other problems

- **Generalisation of Compression.** Alignment adds an orientation to each particle; the same chains also give compression and expansion (Kedia et al. 2022).

Case tree

The root assumes nothing; going down, each edge adds the assumptions written in grey (+) or leaves an axis unassumed. Bold entries are catalogued settings with their status per measure (Time, Moves) and page; italic entries are branch points, named after the axis on which their branches differ. Axis order: Memory → Scheduler → Communication → Orientation → Randomness → Initial shape → Connectivity during execution.

No assumptions

+ constant memory, + sequential, + randomized

Markov chain, sequential STANDARD SETTING ●● p. 616

What is known

The best upper bound against the best lower bound in each catalogued setting, in tree order. The settings follow in detail.

Setting	Time	Moves	p.
Markov chain, sequential STANDARD SETTING	● unknown *eventually* vs —	● unknown — vs —	616

21.6.1 Markov chain, sequential

Adds + constant memory, + sequential, + randomized STANDARD SETTING VERIFIED

For every $q \geq 2$, alignment and non-alignment are both achieved by the bias.

Assumed: memory: constant memory and local communication; scheduler: sequential: one particle active at a time; randomness: randomized. **Not assumed:** communication: local: only between neighbouring particles; orientation: no common orientation and no chirality; initial shape: connected, holes allowed; connectivity during execution: the system must stay connected.

Measure	Lower bound	Best upper bound	Status
Time	none known Mixing time not bounded.	*eventually* (Kedia et al. 2022)	● unknown
Moves	none known —	—	● unknown

Results.

- Kedia et al. (2022) [current best]: time *eventually*. Introductory and current best; simply connected and unconstrained settings.

Claude (Anthropic) was used in cataloguing the problems.

21.7 Bridging

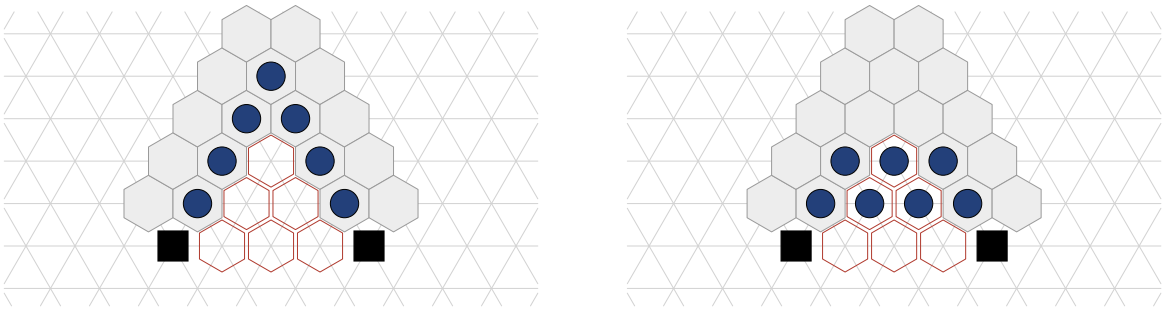
Particles moving between two sites must build a bridge across a gap that balances path length against bridge cost.

First posed in 2017 (Arroyo et al. 2018). arXiv 2017; Natural Computing 2018.

Problem statement and measures

Particles occupy a path between two fixed **sites** across a **gap** in the terrain. They run a local **Markov chain** that rewards neighbours and penalises particles over the gap. The system should form a shortcut **bridge** whose length and cost are balanced, with high probability in the stationary distribution.

Measures: **Time**, asynchronous rounds; **Moves**, total expansions and contractions (work).



(a) Start: particles follow the land

(b) Final: a shortcut bridge over the gap

Figure 21.7: Shortcut bridging by the Markov chain of (Arroyo et al. 2018): particles join two fixed sites (black squares) along the land (shaded); the nodes under the land are over the gap (red outline). One particle at a time (outlined) expands into a free neighbouring node and contracts forward with a probability that rewards a short perimeter (λ) and penalises perimeter over the gap (γ), otherwise it contracts back; moves that would disconnect the particles from the sites, or create a hole, are refused. Shown is a run of accepted moves: the bridge shortcuts the corner but keeps few particles over the gap.

A first look

Army ants build living bridges out of their own bodies to cut a corner across a gap: a longer bridge saves walking but takes more ants out of work. The floor is the **triangular lattice**: the endless graph whose **nodes** are the corners of a tiling of the plane by equal triangles; each node is joined by an edge to six others, its **neighbours**. A **particle** occupies one node and moves in two actions: an **expansion** into an empty neighbouring node, after which it occupies both, then a **contraction** into one of the two; so one expansion and one contraction move it by one node. The particles form a connected **system** joining two fixed **sites** (fixed locations at the two ends of the path); some nodes are marked as **gap** (over empty space) and the others as **land**, and a particle can tell which kind of node it is on. One randomly chosen particle acts at a time, with constant memory and no compass. The system should settle into a **bridge** that balances a short path between the two sites against few particles hanging over the gap, measured by a weighted **perimeter**: the length of the walk around the outside of the system, in which each edge of the walk over the gap costs more than one over land.

Claude (Anthropic) was used in cataloguing the problems.

1. Algorithm: The compression rule, with the gap made expensive. (Arroyo et al. 2018) (Markov chain, sequential, p. 619) There are two **biases**: $\lambda > 1$ rewards a short perimeter (many neighbours), and a second bias, written γ in (Arroyo et al. 2018), rewards a short perimeter over the gap.

1. A randomly chosen particle picks a random empty neighbouring node.
2. It checks locally that the move neither splits the system nor encloses a **hole** (a region of empty nodes surrounded by particles).
3. It works out, from its neighbourhood, how the move changes the perimeter and the part of the perimeter that lies over the gap.
4. It moves with a probability that is larger when both get shorter (certainly, when the move improves both).

Why it works and what it costs. For $\lambda > 2 + \sqrt{2} \approx 3.41$ and $\gamma > 1$, the weighted perimeter is within a constant factor of its minimum with all but exponentially small probability (Arroyo et al. 2018). For a V-shaped gap the bridge settles at a position that depends on the angle of the V, as observed in real ant bridges. The threshold $2 + \sqrt{2}$ is the same as for compression, and for the same counting reason; for small biases the paper gives no guarantee.

Why it matters. It showed that the compression rule is a flexible tool: changing the weights of the edges of the perimeter changes the shape the system settles into.

2. Algorithm: Why only one bridge forms. (Oh et al. 2024) (Markov chain, sequential, p. 619) Fire ants searching for food across water start several bridges but end with only one. (Oh et al. 2024) model this with particles that prefer to move in one direction (towards the food) and prefer having neighbours, with one bias for each preference.

1. Particles move at random, but a step towards the food is more likely than a step away.
2. A step that gains neighbours is more likely than one that loses them.
3. Nothing else: no particle knows how many bridges exist.

Why it works and what it costs. When both biases are large, a single bridge forms almost certainly (Oh et al. 2024). Intuitively, two bridges have a longer perimeter than one bridge made of the same particles, and the neighbour bias makes long perimeters unlikely. The proof follows which nodes are occupied rather than which particle is where, which simplifies the analysis.

Why it matters. It shows that a striking collective decision, choosing one bridge, can arise from two simple preferences without coordination.

What the community started looking at

Bridging was inspired by army ants, whose bridges depend on the gap angle (Arroyo et al. 2018); the later question was whether a **single** bridge forms (Oh et al. 2024).

Relations to other problems

- **Generalisation of Compression.** Bridging adds terrain: the compression chain is biased to stay on land and to shorten a path across a gap (Arroyo et al. 2018).

Claude (Anthropic) was used in cataloguing the problems.

Case tree

The root assumes nothing; going down, each edge adds the assumptions written in grey (+) or leaves an axis unassumed. Bold entries are catalogued settings with their status per measure (Time, Moves) and page; italic entries are branch points, named after the axis on which their branches differ. Axis order: Memory → Scheduler → Communication → Orientation → Randomness → Initial shape → Connectivity during execution.

No assumptions

+ constant memory, + sequential, + randomized

Markov chain, sequential STANDARD SETTING ●● p. 619

What is known

The best upper bound against the best lower bound in each catalogued setting, in tree order. The settings follow in detail.

Setting	Time	Moves	p.
Markov chain, sequential STANDARD SETTING	● unknown *eventually* vs —	● unknown — vs —	619

21.7.1 Markov chain, sequential

Adds + constant memory, + sequential, + randomized STANDARD SETTING VERIFIED

A local stochastic rule builds near-optimal shortcut bridges; for large biases a single bridge forms.

Assumed: memory: constant memory and local communication; scheduler: sequential: one particle active at a time; randomness: randomized. **Not assumed:** communication: local: only between neighbouring particles; orientation: no common orientation and no chirality; initial shape: connected, holes allowed; connectivity during execution: the system must stay connected.

Measure	Lower bound	Best upper bound	Status
Time	none known Mixing time not bounded.	*eventually* (Arroyo et al. 2018)	● unknown
Moves	none known —	—	● unknown

Results.

- Arroyo et al. (2018) [current best]: time *eventually*. Introductory and current best for shortcut bridging; the trade-off depends on the gap angle.
- Oh et al. (2024) [also]: time *eventually*. A single bridge almost certainly forms for large biases; occupancy-chain technique.

21.8 Convex hull formation

Particles must enclose an object in its convex hull, sensing only locally.

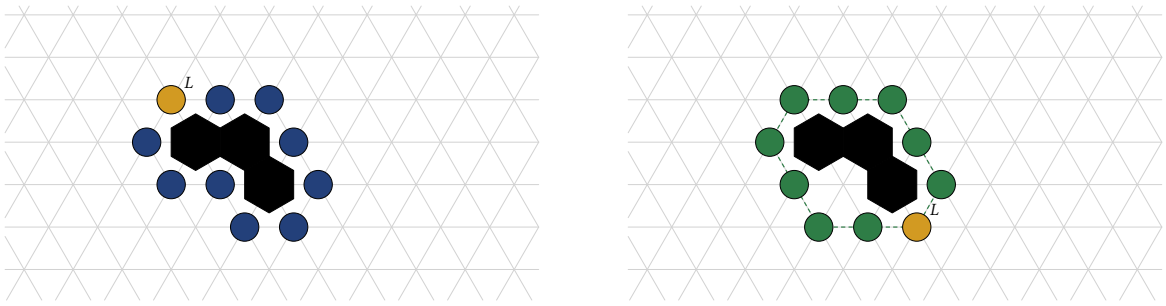
First posed in 2018 (Daymude et al. 2020). arXiv 2018; ICDCN 2020.

Claude (Anthropic) was used in cataloguing the problems.

Problem statement and measures

Particles must form the **convex hull** (and the weak $\mathcal{O}_\Delta$-hull) of a fixed **object**, a connected set of nodes of the triangular lattice that particles cannot enter. B is the length of the object's **boundary**.

Measures: **Time**, asynchronous rounds; **Moves**, total expansions and contractions (work).



(a) Start: leader L, all six walls through it

(b) Final: every node next to the hull occupied

Figure 21.8: Convex hull formation, following (Daymude et al. 2020): a leader (L) walks clockwise around the object (black) and keeps its distance to six walls, one per side of a hexagon with lattice-parallel sides; a step beyond a wall pushes that wall outwards (dashed: the current estimate). Here the particles already surround the object, so the leader advances by role swaps, handing its role and counters to the particle ahead, and nobody moves. Once the leader has touched all six walls since its last push, the hull is known, and the particle in the bay steps out (expansion, contraction), so that the particles occupy exactly the nodes that neighbour the hull.

A first look

Imagine sealing an oddly shaped object inside the tightest convex 'fence', like a rubber band stretched around it. The floor is the **triangular lattice**: the endless graph whose **nodes** are the corners of a tiling of the plane by equal triangles; each node is joined by an edge to six others, its **neighbours**. A **particle** occupies one node and moves in two actions: an **expansion** into an empty neighbouring node, after which it occupies both, then a **contraction** into one of the two; so one expansion and one contraction move it by one node. A fixed **object** occupies a connected set of nodes that particles cannot enter, and particles sit next to it; B is the length of the object's **boundary** (the walk around its outside). The particles must occupy exactly the nodes of the object's **convex hull**, taken one layer outside the object: on the triangular lattice, the smallest region bounded by six straight sides in the lattice's three directions that contains the object. Particles have constant memory, no **compass** (no agreement on the six directions of the lattice), and an adversarial scheduler; one particle starts as the **leader** on the object's boundary. Time is counted in **rounds** (intervals of time in which every particle is activated at least once).

1. Algorithm: Walk around the object and push six walls outwards. (Daymude et al. 2020) (No orientation, p. 622)

1. Using its own local names for the six directions, the leader imagines six **walls** (half-planes), one per side of the future hull. It keeps six **counters**: its distance to each wall, all starting at 0.
2. The leader walks clockwise around the object's boundary. Each step brings it closer to two walls and farther from two others, and it updates the counters.
3. If a step would take it beyond a wall (a counter already at 0 must go down), that wall is *pushed* one step outwards: the estimate of the hull grows.

Claude (Anthropic) was used in cataloguing the problems.

4. The leader also remembers, for each wall, whether it has touched that wall since its last push. Once it has touched all six without pushing any, the walls are in their final places.
5. The counters can be as large as B, too big for one particle, so the followers trailing the leader each store one bit of a binary counter.
6. Finally the particles fill the nodes along the six walls.

Why it works and what it costs. The walls only move outwards and stop at the true hull, so the walk ends with the correct hull. The paper proves that the whole process, including the distributed counters, takes $O(B)$ rounds (Daymude et al. 2020). The same algorithm can be extended to a tighter hull (the weak $\mathcal{O}_\Delta$-hull) in the same time; if there are too few particles, it forms as much of the hull as it can.

Why it matters. Distributed binary counters let constant-memory particles count up to large numbers, a tool other algorithms reuse.

2. Lower bound: Someone must reach the far end. (No orientation, p. 622) This simple argument is ours; the catalogue records no lower bound for this problem. Let the object be a long straight bar, so B is about twice its length, and let all the particles start bunched together near one end, enough of them to fill the hull. The hull has nodes next to the far end of the bar, at distance about $B/2$ from where the particles start, so some particle must travel about $B/2$ nodes.

Why it holds. Each node of travel takes one expansion and one contraction, and an unkind scheduler may let each particle act only once per round, so this takes at least about $B/2$ rounds. So $\Omega(B)$ rounds are needed in the worst case, and the $O(B)$ algorithm above is best possible on such inputs.

Why it matters. It shows that the boundary length, not the number of particles, is the right measure for this problem.

What the community started looking at

Convex hull formation asked whether particles **without shared orientation** can compute a global geometric property, and how fast (Daymude et al. 2020).

Case tree

The root assumes nothing; going down, each edge adds the assumptions written in grey (+) or leaves an axis unassumed. Bold entries are catalogued settings with their status per measure (Time, Moves) and page; italic entries are branch points, named after the axis on which their branches differ. Axis order: Memory → Scheduler → Communication → Orientation → Randomness → Initial shape → Connectivity during execution.

No assumptions

+ constant memory, + strong scheduler

No orientation STANDARD SETTING ●● p. 622

What is known

The best upper bound against the best lower bound in each catalogued setting, in tree order. The settings follow in detail.

Setting	Time	Moves	p.
No orientation STANDARD SETTING	● unknown $O(B)$ vs —	● unknown — vs —	622

Claude (Anthropic) was used in cataloguing the problems.

Setting	Time	Moves	p.

21.8.1 No orientation

Adds + constant memory, + strong scheduler STANDARD SETTING VERIFIED

$O(B)$ rounds without shared orientation.

Assumed: memory: constant memory and local communication; scheduler: strong: atomic activations under a fair adversary. **Not assumed:** communication: local: only between neighbouring particles; orientation: no common orientation and no chirality; randomness: deterministic; initial shape: connected, holes allowed; connectivity during execution: the system must stay connected.

Measure	Lower bound	Best upper bound	Status
Time	none known No lower bound found.	$O(B)$ (Daymude et al. 2020)	• unknown
Moves	none known —	—	• unknown

Results.

- Daymude et al. (2020) [current best]: time $O(B)$. Introductory and current best; uses distributed binary counters.

21.9 Scattering of particles

Particles must reach a configuration in which no two particles occupy neighbouring nodes.

First posed in 2023 (Navarra et al. 2025). Scattering in SILBOT first appeared at AINA 2023 (Navarra, Prencipe, Bonini, Tracolli, "Scattering with programmable matter", pp. 236–247), as the journal version states; that paper was not read and its DOI was not verified, so the journal version (Navarra, Piselli, Prencipe) is cited.

Problem statement and measures

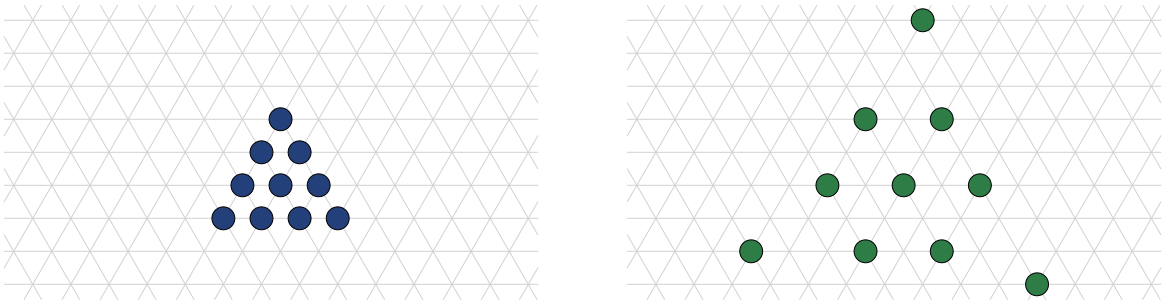
From any initial **configuration** of n **contracted** particles (each occupying one node of the triangular lattice), the particles must reach a configuration in which no two occupy neighbouring nodes (a **scattered** configuration), and stop.

Measures: **Time**, asynchronous rounds; **Moves**, total expansions and contractions (work).

A first look

Picture people standing in a crowded square who must spread out until nobody is standing right next to anyone else. The floor is the **triangular lattice**: the endless graph whose **nodes** are the corners of a tiling of the plane by equal triangles; each node is joined by an edge to six others, its **neighbours**. A **particle** occupies one node and moves in two actions: an **expansion** into an empty neighbouring node, after which it occupies both, then a **contraction** into one of the two; so one expansion and one contraction move it by one node. There are n particles, each occupying one node at the start; the set of occupied nodes is the **configuration**. The particles are **oblivious** (they remember nothing from one action to the next) and **silent** (they send no messages); each decides only from what it sees around it.

Claude (Anthropic) was used in cataloguing the problems.



(a) Start: a crowded triangle; every particle with a neighbour is activated

(b) Final: no two particles are neighbours; none is activated, all stop

Figure 21.9: Scattering under event-driven activation (Navarra et al. 2025): from a configuration of contracted particles, here a triangle of ten, the particles must reach a configuration in which no two occupy neighbouring nodes, and stop. A particle is activated only while it has a neighbour (blue). An activated particle finds its longest run of consecutive empty neighbouring nodes; if the run has 3, 4 or 5 nodes, it makes an expansion into the middle node (outlined: the activated particles that move) and then a contraction there, otherwise it stays. In this run every activated particle acts in each step and all particles break ties the same way; isolated particles (green) are never activated again.

The goal is to reach a **scattered** configuration, where no two particles occupy neighbouring nodes, and then stop moving. The basic question is whether this is possible at all, from every starting configuration.

1. Algorithm: Activated only when crowded. (Navarra et al. 2025) (SILBOT with event-driven activation, p. 626) Navarra, Piselli and Prencipe (Navarra et al. 2025) let each particle see only its six neighbouring nodes, give the particles no common sense of direction, and use **event-driven activation**: a contracted particle is activated (it gets a turn to act, an **activation**) as soon as it has a neighbour, and only then.

1. A particle with no neighbour is never activated, so it stays put.
2. An activated particle looks around itself and finds the longest run I of consecutive empty neighbouring nodes.
3. If I has 3, 4 or 5 nodes, the particle makes an expansion into the most central node of I (in a tie, the one it sees as rightmost), and a contraction into that node once it is free; otherwise it stays.
4. When no particle has a neighbour, no particle is activated again: the task is done and the particles have stopped.

Why it works and what it costs. Event-driven activation gives stopping for free: a scattered configuration is exactly one in which nobody is ever activated again. The difficulty is choosing the local rule so that crowded particles really do spread out instead of moving back and forth forever. The proof looks at the smallest regular hexagon around a fixed centre that contains all particles: a particle on its border never moves inwards, each particle moves away from the centre at most $n - 1$ times (Lemma 4 of Navarra et al. 2025), so the particles on the outermost hexagon end with no neighbour, and induction on n handles the rest (Theorem 6). The paper gives no bound on rounds or moves.

Why it matters. It shows how much the choice of scheduler can replace memory: with a 1-hop view and the usual asynchronous scheduler scattering is impossible (Theorem 4 of Navarra et al. 2025), but with event-driven activation particles without memory scatter and stop.

2. Lower bound: A crowded hexagon must spread over three times its area. (SILBOT with event-driven activation, p. 626) This simple counting argument is ours; the catalogue records no lower bound on

Claude (Anthropic) was used in cataloguing the problems.

time or moves. In the triangular lattice every small triangle of three mutually neighbouring nodes can hold at most one particle of a scattered configuration. Counting triangles, a region with N nodes can hold at most about $N/3$ scattered particles. Now start with the n particles packed into a full hexagon of radius r (it has about $3r^2 = n$ nodes). The hexagon of radius $1.5r$ around the same centre has about $6.75r^2$ nodes, so it can hold only about $2.25r^2 = 0.75n$ scattered particles.

Why it holds. So about $n/4$ particles must end more than $1.5r$ from the centre, having started within r of it: each travels at least $r/2$, which is about $\sqrt{n}$ nodes. That is at least about $n\sqrt{n}$ moves in total, and, if the scheduler lets each particle act once per **round** (an interval of time in which every particle is activated at least once), about $\sqrt{n}$ rounds. (Navarra et al. 2025) give no bound on rounds or moves, so how close their algorithm comes to these numbers is unknown.

Why it matters. It shows that scattering, unlike gathering, forces the occupied region to grow in area, which costs real movement.

What the community started looking at

Scattering was posed for **silent, oblivious** particles, where the question is solvability without memory or messages. Navarra, Piselli and Prencipe restrict the 2-hop view of SILBOT to 1 hop; then scattering is unsolvable under the asynchronous scheduler, solvable without obstacles when a particle is activated as soon as it has a neighbour, and unsolvable with obstacles even under a synchronous scheduler (Navarra et al. 2025).

Relations to other problems

- **Related: Scattering.** Robots in the plane scatter to distinct points; particles must scatter so that no two are neighbours.

Assumptions specific to this problem

Activation

Navarra, Piselli and Prencipe introduced event-driven activation for scattering after showing that with a 1-hop view the problem is unsolvable when the adversary controls activations (Navarra et al. 2025). Under it "the adversary does not control anymore the activation of contracted particles which becomes an event driven (ED) process"; they read it as sensing that reacts to the stimuli of neighbouring particles. In their words, "this kind of schedule, that we call ED-Async, is less general than the pure asynchronous one, but it is clearly more general than the synchronous case where all particles are always active" (Navarra et al. 2025). It is kept as an axis of this problem rather than a value of the model's scheduler axis, because it neither implies nor follows from a sequential scheduler.

Level	Value	Formal definition
0	**adversarial activation**	The scheduler of the setting decides, fairly, when each particle is activated; an isolated particle may be activated, and a particle with a neighbour may wait any finite time.
1	**event-driven**	A contracted particle is activated as soon as it has, or gains, a neighbour, and a contracted particle with no neighbour is not activated. The adversary still chooses the duration of each activation, the time an expansion or a contraction takes, and the activation of expanded particles; operations stay non-atomic.

Claude (Anthropic) was used in cataloguing the problems.

Case tree

The root assumes nothing; going down, each edge adds the assumptions written in grey (+) or leaves an axis unassumed. Bold entries are catalogued settings with their status per measure (Time, Moves) and page; italic entries are branch points, named after the axis on which their branches differ. Axis order: Memory → Scheduler → Communication → Activation → Orientation → Randomness → Initial shape → Connectivity during execution.

SILBOT ●● p. 625

+ event-driven

SILBOT with event-driven activation STANDARD SETTING ●● p. 626

What is known

The best upper bound against the best lower bound in each catalogued setting, in tree order. The settings follow in detail.

Setting	Time	Moves	p.
SILBOT	● impossible	● unknown — vs —	625
SILBOT with event-driven activation STANDARD SETTING	● unknown *finite* vs —	● unknown — vs —	626

21.9.1 SILBOT

the root: no assumptions VERIFIED

With a 1-hop view and adversarial activation, silent oblivious particles cannot scatter; with the 2-hop view of SILBOT the question is open.

Assumed: nothing. **Not assumed:** memory: oblivious, silent particles (SILBOT); scheduler: weak: non-atomic activations, an action may interleave with others; communication: local: only between neighbouring particles; activation: adversarial activation; orientation: no common orientation and no chirality; randomness: deterministic; initial shape: connected, holes allowed; connectivity during execution: the system must stay connected.

Measure	Lower bound	Best upper bound	Status
Time	*impossible* (Navarra et al. 2025) Impossible with a 1-hop view (Theorem 4); open with the 2-hop view of SILBOT. Event-driven activation makes it solvable.	—	● impossible
Moves	none known No algorithm in this setting.	—	● unknown

Results.

- Navarra et al. (2025) [lower bound]: time *impossible*. *Bottleneck:* symmetry breaking. *Holds when:* Theorem 4: deterministic, 1-hop view, no common direction and no chirality; a particle with no neighbour must never move; three particles in a row: whatever a particle with one neighbour does, a fair asynchronous adversary makes the middle particle move back and forth forever; open with the 2-hop view of SILBOT. Lifted by event-driven activation (Theorem 6).

Open questions.

Claude (Anthropic) was used in cataloguing the problems.

Q1. Is scattering solvable in SILBOT with its original 2-hop view under adversarial activation? (Navarra et al. 2025) leave it open.

Notes. Model of (Navarra et al. 2025) (Section 2): SILBOT particles are silent, oblivious and disoriented, all operations are non-atomic, and a fair adversary decides activations; for scattering the view is restricted to 1 hop. The impossibility uses three particles in a row (Figures 3 and 4): a particle with one neighbour must move, either straight away from it or to one of two symmetric nodes, and in both cases the adversary makes the middle particle move back and forth forever. Theorem 7 (obstacles, even synchronous) is recorded on the event-driven setting. Checked against the full text.

21.9.2 SILBOT with event-driven activation

Adds + event-driven **to** SILBOT (p. 625) STANDARD SETTING VERIFIED

Under event-driven activation, silent oblivious particles with a 1-hop view and no common direction scatter from any configuration without obstacles.

Assumed: activation: event-driven activation (ED-Async). **Not assumed:** memory: oblivious, silent particles (SILBOT); scheduler: weak: non-atomic activations, an action may interleave with others; communication: local: only between neighbouring particles; orientation: no common orientation and no chirality; randomness: deterministic; initial shape: connected, holes allowed; connectivity during execution: the system must stay connected.

Measure	Lower bound	Best upper bound	Status
Time	none known	*finite* (Navarra et al. 2025)	• unknown
	Finite; no round bound and no lower bound.		
Moves	none known	—	• unknown
	No move bound is given; each particle moves away from the centre of the enclosing hexagon at most $n-1$ times (Lemma 4).		

Results.

- Navarra et al. (2025) [current best]: time *finite*. *Assumes:* 1-hop view; no obstacles. Introductory and current best (Theorem 6, Corollary 2): the rule of the primer, from any configuration of contracted particles, connected or not; particles share no direction and no chirality. No bound on rounds or moves.

Open questions.

Q1. Bound the rounds and moves of scattering under event-driven activation (Navarra et al. 2025).

Q2. Scattering with obstacles: unsolvable deterministically with a 1-hop view even under a synchronous scheduler (Theorem 7 of Navarra et al. 2025); which extra capability (chirality, a 2-hop view, randomised choices) makes it solvable?

Notes. Scattering is Simple Distancing (Definition 4 of Navarra et al. 2025): eventually every particle is contracted and has no neighbour; under event-driven activation the particles then stop. The proof takes the smallest regular hexagon around a fixed centre containing all particles: no particle on its border moves inwards (Lemma 3), each particle moves outwards at most $n-1$ times (Lemma 4), and induction on n finishes (Theorem 6). With obstacles, where Strong Distancing also asks particles to keep away from obstacles, the problem is unsolvable even under a synchronous scheduler, in which only particles with a neighbour may move (Theorem 7); the paper then tests heuristic rules with randomised tie-breaking in simulation only. Checked against the full text.

Claude (Anthropic) was used in cataloguing the problems.

21.10 Energy distribution

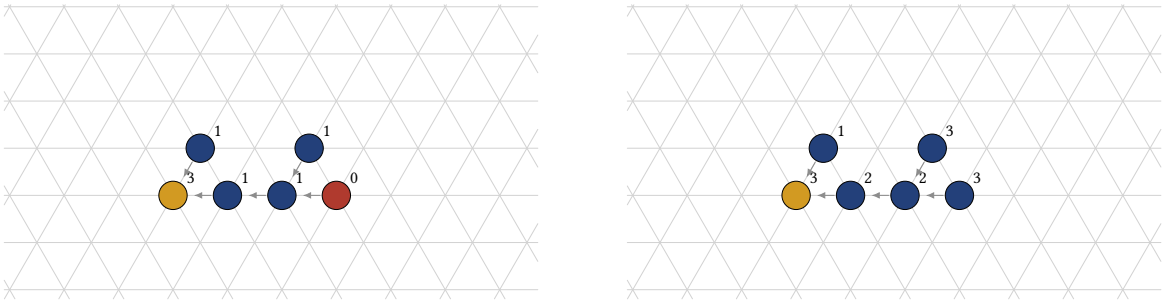
Only some particles reach the energy source; particles must share energy so that every action is paid for.

First posed in 2020 (Daymude et al. 2021). arXiv 2020; ICDCN 2021.

Problem statement and measures

Each action costs energy, and only s particles, the **sources**, can harvest energy from outside. Particles must distribute energy locally so that every particle can act, or so that a given energy-agnostic algorithm runs correctly. **Time** is in rounds.

Measures: **Time**, asynchronous rounds.



(a) Start: one particle has an empty battery

(b) Final: no particle stressed, all can act

Figure 21.10: Energy distribution, following (Daymude et al. 2021): numbers are battery levels (capacity 3) and every action costs 1; only the source (amber) harvests energy, 1 per round. The particles form a tree hanging from the source (arrows point to parents); a particle with energy to spare passes 1 to a child whose battery is not full. A particle with an empty battery is stressed (red): it raises a flag that climbs to the root (outlined), the root then inhibits its tree (grey: inhibited particles stop spending energy on their own actions), and energy flows down until the stress is gone; then the flags clear and everyone acts again. One frame per round.

A first look

Think of a colony of bacteria where only cells at the edge of the colony get food and must pass nutrients inward so that every cell can keep working. The floor is the **triangular lattice**: the endless graph whose **nodes** are the corners of a tiling of the plane by equal triangles; each node is joined by an edge to six others, its **neighbours**. A **particle** occupies one node and moves in two actions: an **expansion** into an empty neighbouring node, after which it occupies both, then a **contraction** into one of the two; so one expansion and one contraction move it by one node. The **system** of n particles is connected; only s of them, the **sources**, can take energy from outside. Every action (a move, or any step of some other algorithm) costs energy; each particle has a **battery** of fixed capacity and can pass a limited amount of energy to a neighbour each time it is activated (an **activation**). The goal is that no particle stays **stressed** (short of energy for its next action) for long, and that the system as a whole keeps acting; we count **rounds** (intervals of time in which every particle is activated at least once). Particles have constant memory and no compass.

Claude (Anthropic) was used in cataloguing the problems.

1. Algorithm: Share energy down a forest, and hold back when a particle is stressed. (Daymude et al. 2021) (Unfair adversary, p. 629)

1. **Set-up.** The sources are **roots**. Every other particle picks a neighbour that is already in a tree as its *parent*, so the system becomes a forest of trees hanging from the sources.
2. **Signal.** A stressed particle raises a 'stress' flag. A particle with a stressed child raises its flag too, so the signal climbs to the root.
3. **Inhibit.** A root that hears of stress raises an 'inhibit' flag, which spreads down its whole tree: **inhibited** particles stop spending energy on their own actions.
4. **Share.** Roots harvest energy; any particle with enough energy passes some to a child whose battery is not full.
5. **Use.** A particle that is not inhibited and has enough energy performs its next action. Flags are cleared when the stress is gone.

Why it works and what it costs. Energy flows down the trees like water through pipes, and the inhibit signal stops particles near the source from using everything before it reaches the far ends. (Daymude et al. 2021) prove that no particle stays stressed for more than $O(n)$ rounds; without the inhibit step, their simulations show particles staying stressed forever. With a constant number of sources this is optimal, by the next idea. (Weber et al. 2023) show how to run any algorithm designed without energy limits on top of this: one that needs T rounds then needs $O(n^2\,T)$ rounds.

Why it matters. It was the first model of energy as a limited resource in programmable matter, and the stress/inhibit signals are modelled on how bacterial biofilms share nutrients.

2. Lower bound: Energy enters only through the sources. (Daymude et al. 2021) (Unfair adversary, p. 629) Start with every battery empty and every particle needing a full battery. Then the system needs n full batteries' worth of energy in total. Energy enters only at the s sources, and each source harvests at most a fixed amount per activation. An unkind scheduler activates each source only once per round.

Why it holds. So at most a constant times s units of energy enter per round, and filling n batteries takes at least a constant times n/s rounds: $\Omega(n/s)$ (Daymude et al. 2021) (Theorem 9). With a constant number of sources this matches the $O(n)$ algorithm; with many sources there is a gap of a factor s between n/s and n that has not been closed.

Why it matters. It is a supply argument rather than a distance argument: the bottleneck is how fast energy can enter, not how far it must travel.

What the community started looking at

Energy distribution was inspired by biofilms (Daymude et al. 2021); the later question was a **general transformation** of energy-agnostic algorithms and its overhead under unfair adversaries (Weber et al. 2023).

Claude (Anthropic) was used in cataloguing the problems.

Case tree

The root assumes nothing; going down, each edge adds the assumptions written in grey (+) or leaves an axis unassumed. Bold entries are catalogued settings with their status per measure (Time) and page; italic entries are branch points, named after the axis on which their branches differ. Axis order: Memory → Scheduler → Communication → Orientation → Randomness → Initial shape → Connectivity during execution.

No assumptions

+ constant memory, + sequential

Unfair adversary STANDARD SETTING ● p. 629

What is known

The best upper bound against the best lower bound in each catalogued setting, in tree order. The settings follow in detail.

Setting	Time	p.
Unfair adversary STANDARD SETTING	● polynomial gap $O(n)$ vs $\Omega(n/s)$	629

21.10.1 Unfair adversary

Adds + constant memory, + sequential STANDARD SETTING VERIFIED

Energy can be distributed in $O(n)$ rounds, against an $\Omega(n/s)$ lower bound; any conforming energy-agnostic algorithm becomes energy-constrained with $O(n^2)$ rounds of overhead per action, even under an unfair adversary.

Assumed: memory: constant memory and local communication; scheduler: sequential: one particle active at a time. **Not assumed:** communication: local: only between neighbouring particles; orientation: no common orientation and no chirality; randomness: deterministic; initial shape: connected, holes allowed; connectivity during execution: the system must stay connected.

Measure	Lower bound	Best upper bound	Status
Time	$\Omega(n/s)$ (Daymude et al. 2021) $O(n)$ against $\Omega(n/s)$: a factor s.	$O(n)$ (Daymude et al. 2021)	● polynomial gap

Results.

- Daymude et al. (2021) [lower bound]: time $\Omega(n/s)$. *Bottleneck:* distance. *Holds when:* Theorem 9; s energy sources.
- Daymude et al. (2021) [current best]: time $O(n)$. Introductory and current best (Theorem 8); a self-stabilising spanning forest tolerates crash failures.
- Weber et al. (2023) [also]: time $O(n^2\, T_A(n))$. *Assumes:* unfair adversary. Theorem 4: $T_A(n)$ is the running time of the energy-agnostic algorithm. Composes with the concurrency control framework.

Notes. No orientation is assumed. Checked against the full texts.

Claude (Anthropic) was used in cataloguing the problems.

21.11 Shape formation by hybrid programmable matter

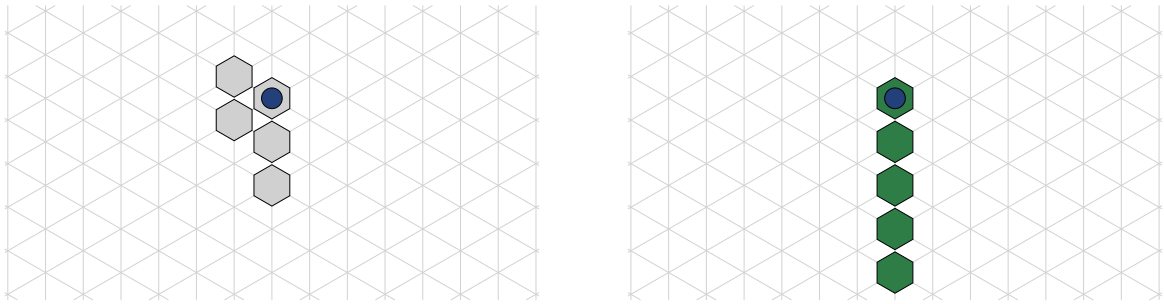
One finite-automaton agent rearranges passive tiles into a target shape.

First posed in 2017 (Gmyr et al. 2020). First presented at EuroCG 2017 (non-archival); DNA 24 (2018); Natural Computing 2020. The work began at a Dagstuhl seminar in 2016.

Problem statement and measures

n passive hexagonal tiles (rhombic dodecahedra in 3D) occupy nodes of a lattice and form a connected structure. A single active agent, a deterministic finite automaton, starts on a tile, must always stand on or next to a tile, sees its node and the adjacent nodes, and in each **round** lifts a tile, places the tile it carries, or moves to an adjacent node; it carries at most one tile, and the tiles (with the carried tile counted at the agent's node) must stay connected. It must rearrange the tiles into a target shape (a line, a triangle, or an intermediate structure such as a block or a tree). **Time** counts the agent's rounds.

Measures: **Time**, rounds; in each round the agent performs one Look-Compute-Move step.



(a) Start: connected tiles, agent on a tile

(b) Final: the tiles form a line

Figure 21.11: Shape formation by hybrid programmable matter, the line algorithm of (Gmyr et al. 2020): a single agent (blue ring), a finite automaton, walks on passive hexagonal tiles and carries at most one tile (amber). It first walks south as far as there are tiles; then it repeatedly searches, walking N, NW or SW (in that order) while there is a tile there, until it stands on a top-left tile (no tile to its N, NW or SW); it lifts that tile, moves once SE, walks S to the first empty node and places the tile there. One frame per round (move, lift or place); the tiles, with the carried tile counted at the agent, stay connected. It stops when the tiles form a single north-south line.

A first look

Imagine a single small machine on a beach of hexagonal tiles, which can pick up one tile at a time, carry it, and put it down elsewhere, and must rebuild the whole pile into a given shape. Here the machine is an **agent** and the beach is the **triangular lattice**: the infinite tiling of the plane by equilateral triangles, whose corners are its **nodes**. n passive hexagonal **tiles** sit on nodes, one tile per node; together they form a **structure**, which must be **connected**: from every tile one can reach every other tile by moving between tiles on adjacent nodes. The agent has only a fixed finite memory (it is a **finite automaton**) and starts on a tile. Unlike in the amoebot model, the tiles cannot expand or contract; only the agent moves. In each **round** it does one thing: it moves to an adjacent node, always staying on or next to a tile, or it lifts the tile under it, or it places the tile it carries. The tiles (counting the carried one) must stay connected. The goal is a **target shape** such as a straight **line** of tiles (tiles on consecutive nodes in

Claude (Anthropic) was used in cataloguing the problems.

one direction) or a triangle; time is the number of rounds, and D is the **diameter** of the initial structure, the largest distance between two of its tiles, counted in moves between adjacent nodes.

1. Algorithm: Build a line by moving top-left tiles down. (Gmyr et al. 2020) (Planar, p. 633) Here the line runs north-south.

1. The agent walks south (down) as long as there is a tile below it.
2. **Search.** It walks north, north-west or south-west (in that order of preference) as long as one of these has a tile. It stops on a **top-left tile**: a tile with no tile to its north, north-west or south-west.
3. **Move.** It lifts that tile, moves once to the south-east, walks south until it reaches an empty node, and places the tile there.
4. It repeats the search. When a search finds no tile to the east or west, the structure is a line and the agent stops.

Why it works and what it costs. The tile it moves is never needed to hold the structure together, because it is placed back where it reconnects any tiles it might have separated. Tiles only move south-east and south, and each tile is carried across at most about $2n$ nodes, so there are $O(n^2)$ rounds in which the agent carries a tile; a bookkeeping argument bounds the searching rounds by the same amount. Total: $O(n^2)$ rounds (Gmyr et al. 2020). The same paper builds a more compact intermediate shape (a **block**) in $O(nD)$ rounds, and from it a triangle in $O(nD)$ rounds, which is $O(n^{3/2})$ when D is about $\sqrt{n}$.

Why it matters. It shows an agent with constant memory can reshape a large structure, using simple intermediate shapes it can navigate.

2. Lower bound: Carrying one tile at a time is slow. (Gmyr et al. 2020) (Planar, p. 633) Start from a compact structure of n tiles, of diameter about $\sqrt{n}$, and ask for a line. As in shape formation, a constant fraction of the tiles must end at distance proportional to n from where they started. The agent carries one tile at a time and moves to one adjacent node per round, so each such tile costs it about n rounds of carrying.

Why it holds. So a constant fraction of n tiles times about n rounds each gives $\Omega(n^2)$ rounds (Gmyr et al. 2020), and the line algorithm above is optimal. A similar argument gives $\Omega(nD)$ for the block, so the block algorithm is optimal as well.

Why it matters. With a single agent, the total distance tiles must travel directly bounds the time.

3. Impossibility: An agent with finite memory cannot find a safely removable tile. (Gmyr et al. 2020) (Planar, p. 633) The naive approach would be: find a tile whose removal keeps the structure connected (a **safely removable** tile), carry it to its place in the target shape, repeat. Consider a large hollow hexagon of tiles, a closed ring. An agent with s memory states walking along a long side of the ring sees the same thing in every round, so after at most s rounds it repeats its behaviour; it cannot count the length of a side. So it cannot tell this ring apart from a structure that looks the same nearby but is not closed far away, where the same tile is no longer safely removable. (Gmyr et al. 2020) (Theorem 1) make this precise with hollow hexagons and spirals; the full proof is longer.

Why it holds. No finite automaton can, without lifting tiles, always stop on a safely removable tile. With a single **pebble**, a marker it can drop and pick up, it can find one in $O(n^2)$ rounds (Gmyr et al. 2020) (Theorem 2). This is why the algorithms above move tiles only in ways that are always locally safe.

Why it matters. It shows how far a finite memory falls short of seeing the whole picture, and why a single marker helps so much.

Claude (Anthropic) was used in cataloguing the problems.

What the community started looking at

The hybrid model asked what **one weak active agent** can build from passive material; the first answer was that it cannot always find a tile that is safe to remove, so it builds intermediate structures by local moves instead (Gmyr et al. 2020), then whether the same holds in 3D (Hinnenthal et al. 2025), and how fast an agent that senses the target nodes can repair a structure that is close to its target (Friemel et al. 2026).

Further reading

Papers on this problem that are not yet placed in a setting.

- Nokhanji et al. (2023). Several agents (finite automata) keep a line of tiles while tiles fail at any time and must be removed and replaced, under a semi-synchronous scheduler. No deterministic algorithm works for every number of agents and every placement, even fully synchronously with one failure (Theorem 1); algorithms for one agent, for one agent per tile, and for any number of agents placed at one end of the line (Theorems 3, 6 and 7). Read (journal version).
- Gmyr et al. (2018). A finite-automaton robot decides whether hexagonal tiles form a parallelogram with sides h and $f(h)$: possible for f linear without pebbles, impossible for superlinear f; with one pebble, any polynomial of constant degree. Abstract read.
- Czyzowicz et al. (2020c). A finite-automaton robot in the infinite grid moves bricks, one at a time, to build the most compact structure (a nest) from a connected field. Abstract read.
- Kant et al. (2021). A finite-automaton robot builds a fort (a hollow rectangle of bricks) in $\Theta(z^2)$ time, z the number of bricks. Abstract read.
- Fekete et al. (2019). Finite-automaton robots assemble and reconfigure connected tile structures. A preprint; abstract read.

Relations to other problems

- **Related: Shape formation in programmable matter.** In the hybrid model the tiles are passive and one active agent moves them; in the amoebot model every particle is active.

Assumptions specific to this problem

Space

The hybrid model has a few active agents with the power of finite automata and many passive tiles that the agents pick up, carry and place (Hinnenthal et al. 2025). It is motivated by the observation that "while fabricating computing DNA robots remains challenging, producing simple passive tiles from folded DNA strands is efficient and scalable", and an agent can be reused after a task (Hinnenthal et al. 2025). The earliest paper of the model in this atlas, by Gmyr et al. (DNA 24, 2018), works in the plane (Gmyr et al. 2020). Hinnenthal, Liedtke and Scheideler moved to three dimensions, with rhombic-dodecahedral tiles on the face-centred cubic lattice. There "complexity increases significantly, with instances where any tile movement can locally disconnect the structure" (Hinnenthal et al. 2025). Coating in 3D followed (Kostitsyna et al. 2024).

Level	Value	Formal definition
0	**3D**	Tiles occupy nodes of the face-centred cubic lattice; planar structures are a special case.
1	**2D**	Tiles occupy nodes of the triangular lattice.

Claude (Anthropic) was used in cataloguing the problems.

Case tree

The root assumes nothing; going down, each edge adds the assumptions written in grey (+) or leaves an axis unassumed. Bold entries are catalogued settings with their status per measure (Time) and page; italic entries are branch points, named after the axis on which their branches differ. Axis order: Space.

Three-dimensional ● p. 633

+ 2D

Planar STANDARD SETTING ● p. 633

What is known

The best upper bound against the best lower bound in each catalogued setting, in tree order. The settings follow in detail.

Setting	Time	p.
Three-dimensional	● unknown $O(n^3)$ vs —	633
Planar STANDARD SETTING	● tight $O(n^2)$ vs $\Omega(n^2)$	633

21.11.1 Three-dimensional

the root: no assumptions VERIFIED

In 3D, one agent turns any connected tile structure into a low-diameter icicle in $O(n^3)$ rounds.

Assumed: nothing. **Not assumed:** space: three-dimensional lattice.

Measure	Lower bound	Best upper bound	Status
Time	none known No lower bound recorded.	$O(n^3)$ (Hinnenthal et al. 2025)	● unknown

Results.

- Hinnenthal et al. (2025) [current best]: time $O(n^3)$. *Assumes:* the agent starts carrying a tile; a consistent compass; the agent may pass through tiles. Introductory and current best (Theorem 14); matches the prior runtime of 3D line formation.

Notes. Checked against the full text.

21.11.2 Planar

Adds + 2D **to** Three-dimensional (p. 633) STANDARD SETTING VERIFIED

One agent builds a line or a tree in $O(n^2)$ rounds and a block in $O(nD)$ rounds, which is asymptotically optimal, and then a triangle in $O(nD)$ rounds.

Assumed: space: the triangular lattice (planar). **Not assumed:** —.

Claude (Anthropic) was used in cataloguing the problems.

Measure	Lower bound	Best upper bound	Status
Time	$\Omega(n^2)$ (Gmyr et al. 2020); *impossible* (Gmyr et al. 2020)	$O(n^2)$ (Gmyr et al. 2020)	• tight
	$\Theta(n^2)$ for the line and $\Theta(nD)$ for the block in the worst case; the lower bounds are sketched.		

Results.

- Gmyr et al. (2020) [lower bound]: time $\Omega(n^2)$. *Bottleneck:* one active agent. *Holds when:* line formation from an initial configuration of diameter $O(\sqrt{n})$: a constant fraction of the tiles must move a distance linear in n; a single agent carrying one tile at a time. Stated after Theorem 3 with a one-line argument ("not hard to see"); a similar argument gives $\Omega(nD)$ for a block.
- Gmyr et al. (2020) [lower bound]: time *impossible*. *Bottleneck:* memory. *Holds when:* the agent must stop on a safely removable tile (one whose removal keeps the structure connected) without lifting any tile; no pebble; instances: hollow hexagons and a spiral of long segments. Theorem 1: no finite automaton finds a safely removable tile in general; with one pebble it takes $O(n^2)$ rounds (Theorem 2). This rules out the naive approach, not shape formation.
- Gmyr et al. (2020) [current best]: time $O(n^2)$. *Assumes:* no compass needed: the agent keeps its orientation relative to its initial one. Introductory and current best: line in $O(n^2)$ (Theorem 3), block of diameter $O(D)$ in $O(nD)$ (Theorem 4), tree inside the initial convex hull in $O(n^2)$ (Theorem 5), triangle in $O(nD)$ (Theorem 6), hence $O(n^{3/2})$ when $D = O(\sqrt{n})$. With several agents, line formation was sped up almost linearly for a small number of agents in simulations only.
- Friemel et al. (2026) [also]: time $O(mn)$. *Assumes:* shape reconfiguration: the target shape is given in fixed position, and the agent senses which nodes belong to it; simply connected target shape; the tiles that start on target nodes form a non-empty connected set. m is the number of tiles that start outside the target shape. $O(mn)$ rounds (Theorem 5), and $\Omega(mn)$ rounds are needed in general: a line of n tiles shifted by m nodes along itself (Theorem 6). This is faster than building a line from scratch when $m = o(n)$, and it forms every simply connected target shape, but the algorithm does not detect that it is done; the authors state that termination can be added within $O(n^2)$ rounds (Section 5). Target shapes with holes: $O(n^4)$ rounds if no target node has disconnected non-target neighbours (Theorem 16); any target shape in $O(mn^2)$ rounds with a counter of $O(\log n)$ bits or $O(mn^3)$ with two pebbles (Theorem 8).

Notes. D is the diameter of the initial tile set. Checked against the full text of (Gmyr et al. 2020) (Natural Computing version).

MATTER



21.12 Centralised transformations of modular robots

A centralised algorithm must transform one connected shape of n modules on the square grid into another using local moves.

First posed in 2017 (Michail et al. 2019). arXiv 2017; JCSS 2019.

Problem statement and measures

n **modules** (unit squares, the particles of this model) occupy **cells** of the square grid and form a connected **shape**; modules in cells that share a side are **neighbours**, and the shape is connected when every module can be reached from every other through neighbours. A **centralised algorithm** must transform it into a given target shape with the same number of modules, using only the allowed local moves, keeping the shape connected where required. **Moves** counts sequential moves; **time** is parallel time when several moves may happen at once.

Measures: **Moves**, sequential moves; **Time**, parallel time.

Claude (Anthropic) was used in cataloguing the problems.



(a) Start: a connected shape of 5 modules

(b) Final: the modules form a line

Figure 21.12: Centralised transformation with rotations and slidings, the universal strategy of (Michail et al. 2019): repeatedly pick a module on the outside whose removal keeps the rest connected and walk it around the outside, rotating at corners (a quarter turn around a neighbour, the swept cells being empty) and sliding along straight stretches (one cell along two neighbours), to the end of a line growing to the right along the bottom row (dashed); a target shape is reached by also playing its own plan to the line backwards (not shown). One frame per move; a rotation is drawn along its arc.

A first look

Picture a set of square magnets on a chessboard, all touching one another, that you are allowed to rearrange only by small local moves, and ask: can one arrangement be turned into another, and how many moves does it take? Here the magnets are n square **modules** and the squares of the board are **cells** of the infinite **square grid**. Modules in cells that share a side are **neighbours**, and the modules form a **connected shape**: every module can be reached from every other through neighbours. A **centralised algorithm** sees everything and decides the moves. Unlike amoebot particles, which move by expanding and contracting, a module moves by **rotation** (turning a quarter circle around a neighbour, ending on a diagonal cell) or by **sliding** (moving one cell along two neighbouring modules); a **move** is allowed only if the cells it sweeps through are empty. The goal: transform the shape into a given **target shape** with the same number of modules, and count moves (one at a time) or **parallel time** (the number of steps when several moves may happen at once).

1. Algorithm: Everything turns into a line, and moves can be undone. (Michail et al. 2019) (Rotation and sliding, p. 639) With rotations and slidings both allowed:

1. Find a module on the outside of the shape whose removal keeps the rest connected; every connected shape with at least two modules has at least two.
2. That module can walk around the outside of the rest, rotating at corners and sliding along straight stretches, always staying a neighbour of some module.
3. Walk it to the end of a horizontal **line** (modules in consecutive cells of one row) growing from the shape's lowest right-most module; repeat until all n modules form the line.
4. To reach a target shape B, also plan $B \to$ line, then play that plan backwards: every rotation and sliding can be undone by the opposite move.

Why it works and what it costs. So any two connected shapes with the same number of modules can be transformed into each other without ever disconnecting (Michail et al. 2019). Each module walks at most about the perimeter of the shape, so the plan uses $O(n^2)$ moves, and that is the best possible in the worst case (the third idea). Allowing many moves at once, the same paper reaches $O(n)$ parallel time, also best possible.

Claude (Anthropic) was used in cataloguing the problems.

Why it matters. Going through a canonical shape and using reversibility is the standard way to prove that a set of moves is universal.

2. Impossibility: Rotations never change a module's chessboard colour. (Michail et al. 2019) (Rotation only, p. 638) Colour the grid like a chessboard. A rotation moves a module to a cell diagonal to its old one, which has the same colour. So with rotations alone, the number of modules on black cells never changes, and neither does the number on white cells. Example: a horizontal line of 3 modules on black, white, black cells has two on black; a shape of 3 modules with two on white cells can never be reached from it.

Why it holds. Two shapes with different colour counts can never be transformed into each other by rotations (Michail et al. 2019). Equal colour counts are not enough either: some shapes have no rotation available at all and are stuck. If disconnecting is allowed, the paper shows that deciding whether a transformation exists takes polynomial time; if connectivity must be kept, the decision is in PSPACE.

Why it matters. An **invariant**, a quantity that no move changes, is the simplest tool for proving that a transformation is impossible.

3. Lower bound: A staircase is far from a line. (Michail et al. 2019) (Rotation and sliding, p. 639) Take a *ladder*: a diagonal staircase two modules wide, climbing one row every two modules. Ask for a horizontal line. Wherever the line is placed, at least half of the ladder's modules lie on one side of it, at heights $1, 1, 2, 2, \ldots$ up to about $n/4$ from it. Each must cover at least that height, so the total distance is at least $2(1 + 2 + \cdots + n/4)$, which is about $n^2/16$.

Why it holds. One rotation or sliding moves one module by a constant distance, so it reduces this total by at most a constant. Hence $\Omega(n^2)$ moves are needed, even if the shape may disconnect (Michail et al. 2019) (Theorem 18), and the line-based plan is optimal. The same measure, total distance under the best matching of modules to target cells, can be computed in polynomial time, so such lower bounds are easy to evaluate.

Why it matters. The total-distance argument gives quadratic lower bounds for every model where one entity moves one step at a time.

MATTER

What the community started looking at

The first question was **which mechanisms are universal** and how hard it is to decide transformability (Michail et al. 2019); then how far stronger mechanisms reduce the number of moves (Almethen et al. 2020; Akitaya et al. 2021b).

Further reading

Papers on this problem that are not yet placed in a setting.

- Kumar et al. (2026b). Centralised reconfiguration of amoebot structures on the triangular lattice with joint movements: in each synchronous round every particle may expand, contract or hand over an edge to a neighbour, and the particles bonded to it move along with it, a linear-strength operation like line pushing (Almethen et al. 2020); no metamodules. Any connected structure of n particles becomes a straight line in $O(\sqrt{n}\log n)$ rounds of parallel time, hence universal reconfiguration in the same time (Theorem 14); monotone structures become a line in 16 rounds (Theorem 8), star-convex ones in $O(1)$ rounds (Theorems 8 and 10), spirals in $O(1)$ rounds (Theorem 18). This answers an open question of (Padalkin et al. 2025), as reported by the authors; polylogarithmic universal reconfiguration is open (Section 7). Not placed: the settings of this problem are on the square grid. Read (arXiv v1).
- Akitaya et al. (2021a). Pivoting hexagonal modules: with restricted and 'monkey' pivots together, any two connected configurations are reconfigurable with $O(n^3)$ moves; hardness results for weaker move sets. Abstract

Claude (Anthropic) was used in cataloguing the problems.

read.

- Akitaya and Stock (2023). Rhombic dodecahedral modules in 3D: reconfiguration is PSPACE-hard under restricted pivoting, and configurations are not universally reconfigurable even with more general pivots. A preprint; abstract read.
- Parada et al. (2021). Compact 3D meta-modules for edge-hinged and central-point-hinged modular robots, which expand and contract to support efficient reconfiguration. Abstract read.
- Gupta et al. (2023b). Deciding whether expansions and contractions of square modules may cause collisions: $O(n^2 \log^2 n)$ to $O(n \log^2 n)$ algorithms depending on coupled operations; avoiding collisions is hard. A preprint; abstract read.
- Schmidt et al. (2018). Staged self-assembly of particles moved by uniform external forces, with sublinear construction time for large classes of shapes. Abstract read.
- Naz et al. (2016). A decentralised, asynchronous self-reconfiguration algorithm for rolling cylindrical modules in a vertical hexagonal lattice. Abstract read.
- Bassil et al. (2022). Distributed self-reconfiguration of modular robots through a porous 3D structure built from meta-modules. Abstract read.
- Bassil et al. (2024). Self-reconfiguration of porous modular structures with a centralised max-flow planner for module paths and asynchronous execution. Abstract read.

Relations to other problems

- **Related: Shape formation in programmable matter.** A central controller plans the moves, so the question is which mechanisms are universal and how many moves they need, not how particles coordinate.

Assumptions specific to this problem

Mechanism

Michail, Skretas and Spirakis chose two movements that are "easily implementable by existing technology": a module may *rotate* 90° around a neighbour, or *slide* one cell over two neighbours (Michail et al. 2019). Rotation and sliding together had been studied before for metamorphic robotic systems (Dumitrescu et al. 2004) (as cited by Michail et al. 2019). Rotation alone is the weaker mechanism. Michail et al. showed that deciding transformability with rotations alone is in P, and in PSPACE when connectivity must be kept; adding sliding makes any two connected shapes of the same size transformable into each other with $\Theta(n^2)$ moves (Michail et al. 2019). Almethen, Michail and Potapov later added a line-pushing mechanism that contains both as special cases and is faster (Almethen et al. 2020).

Level	Value	Formal definition
0	**rotation**	A module may rotate 90° around an adjacent module, if the cells it sweeps are empty.
1	**rotation + sliding**	In addition, a module may slide one cell along a line of modules.

Claude (Anthropic) was used in cataloguing the problems.

Case tree

The root assumes nothing; going down, each edge adds the assumptions written in grey (+) or leaves an axis unassumed. Bold entries are catalogued settings with their status per measure (Moves, Time) and page; italic entries are branch points, named after the axis on which their branches differ. Axis order: Mechanism.

Rotation only STANDARD SETTING ●● p. 638

+ rotation + sliding

Rotation and sliding ●● p. 639

What is known

The best upper bound against the best lower bound in each catalogued setting, in tree order. The settings follow in detail.

Setting	Moves	Time	p.
Rotation only STANDARD SETTING	● unknown *decidable in P* vs —	● unknown — vs —	638
Rotation and sliding	● tight $O(n^2)$ vs $\Omega(n^2)$	● tight $O(n)$ vs $\Omega(n)$	639

21.12.1 Rotation only

the root: no assumptions STANDARD SETTING VERIFIED

With rotations alone, deciding whether one shape can be transformed into another is in P; with connectivity preserved it is in PSPACE.

Assumed: nothing. **Not assumed:** mechanism: rotation only.

Measure	Lower bound	Best upper bound	Status
Moves	none known —	*decidable in P* (Michail et al. 2019)	● unknown
Time	none known —	—	● unknown

Results.

- Michail et al. (2019) [current best]: moves *decidable in P*. Introductory and current best: transformability with rotations, allowing disconnection, is decidable in polynomial time (Theorem 9); preserving connectivity it is in PSPACE (Theorem 10).

Open questions.

Q1. What is the complexity of connectivity-preserving transformability with rotations alone? The paper leaves it open (Section 7) (Michail et al. 2019).

Notes. Checked against the full text.

Claude (Anthropic) was used in cataloguing the problems.

21.12.2 Rotation and sliding

Adds + rotation + sliding **to** Rotation only (p. 638) VERIFIED

Rotation and sliding are universal for connected shapes: $\Theta(n^2)$ moves sequentially and $O(n)$ parallel time, both optimal.

Assumed: mechanism: rotation and sliding. **Not assumed:** —.

Measure	Lower bound	Best upper bound	Status
Moves	$\Omega(n^2)$ (Michail et al. 2019) $\Theta(n^2)$ moves.	$O(n^2)$ (Michail et al. 2019)	• tight
Time	$\Omega(n)$ (Michail et al. 2019) $\Theta(n)$ parallel time.	$O(n)$ (Michail et al. 2019)	• tight

Results.

- Michail et al. (2019) [lower bound]: moves $\Omega(n^2)$. *Bottleneck:* distance. *Holds when:* Theorem 18: transforming a ladder into a line; holds even without preserving connectivity.
- Michail et al. (2019) [lower bound]: time $\Omega(n)$. *Bottleneck:* distance. *Holds when:* Proposition 15 (arXiv version); connectivity is preserved.
- Michail et al. (2019) [current best]: moves $O(n^2)$, time $O(n)$. Introductory and current best; connectivity preserved.
- Almethen et al. (2020) [also]: moves $O(n \log n)$. *Assumes:* line-pushing mechanism: one move shifts a whole line of modules; the shape may disconnect. A stronger mechanism; a suspected $\Omega(n \log n)$ lower bound is left open.

Open questions.

Q1. Prove the suspected $\Omega(n \log n)$ lower bound for universal transformations with line pushing (Almethen et al. 2020).

Q2. Distributed universal transformations with line pushing (Almethen et al. 2020).

Notes. With pivoting square modules (facet-connected), universal reconfiguration between any two configurations needs only five extra helper modules and $O(n^2)$ pivot moves, worst-case optimal (Akitaya et al. 2021b). Checked against the full texts.

Claude (Anthropic) was used in cataloguing the problems.

Appendices

Appendix A

All open questions

The most consequential

A ranking by downstream impact: how many problems and settings a solution would feed, through derived problems, related problems and the carry-over of algorithms and lower bounds; weighed with how long a question has been open and how far apart its bounds are. The ranking is the maintainer's judgement; the counts in the table below are computed from the catalogue.

How the ranking works

A solution propagates along three kinds of links: a derived problem inherits lower bounds from its parent and gives algorithms back to it; problems that use another as a first step inherit its running time; and inside one problem, an algorithm for weaker assumptions carries over to stronger settings while a lower bound carries over to weaker ones. The ranking weighs this reach with how long a question has been open and how far apart its bounds are. It is a judgement; the open questions page shows the underlying counts.

1. **Collective exploration of trees.** Is the competitive ratio of collective tree exploration $O(\log k)$? The bounds are $\Omega(\log k / \log\log k)$ (Disser et al. 2020) against $O(\log^2 k)$ (Cosson and Massoulié 2026), a preprint. In its sharpest form: can n agents explore an anonymous tree of height D in $O(D^2)$ rounds with local communication, and is $\Omega(D^2)$ necessary for every D (Disser et al. 2020)?

   Why it matters. Dispersion is derived from it: with $k = n$ every dispersion algorithm explores, and dispersion's $\Omega(D^2)$ tree bound is borrowed from it (Augustine and Moses Jr. 2018). Leader election, independent and dominating sets, spanning and BFS trees by agents all start by dispersing, so their running times inherit any change. Black hole search and dynamic exploration are team exploration under faults.

   Settings: Collaborative exploration, Collaborative exploration, Collaborative exploration, Dispersion.

2. **Optimal-time rendezvous of labelled agents.** Find an optimal-time rendezvous algorithm for two labelled agents in arbitrary graphs: the best bound is $\tilde{O}(n^5 \log \lambda)$ (Ta-Shma and Zwick 2014) against $\Omega(n^2 + \log \lambda)$ (Dessmark et al. 2006), where λ is the smaller label; the question has been open since 2006 (Pelc 2019).

   Why it matters. Rendezvous is the two-agent core of gathering in graphs, and its algorithms rest on the same universal exploration sequences as treasure hunt; a tight bound would sharpen gathering, treasure hunt and leader election by agents.

   Settings: Rendezvous.

3. **How much memory exploration needs.** Can an agent explore a tree and stop with asymptotically less memory than exploration with return? The bounds are $\Omega(\log\log\log n)$ (Diks et al. 2004) against $O(\log n)$ (Ambühl et al. 2011). More broadly: what is the time of memory-optimal exploration?

   Why it matters. Memory lower bounds for agent problems, dispersion among them, come from exploration arguments; exploration is also the oldest question in the area.

Settings: Graph exploration by a single agent, Graph exploration by a single agent.

4. **Dispersion with optimal memory and termination.** Can k agents disperse in $O(k)$ rounds with $O(\log k)$ bits, independent of Δ, and terminate without knowing k, m or Δ? The best algorithm uses $O(\log(k+\Delta))$ bits (Kshemkalyani et al. 2025b); the lower bound is $\Omega(\log k)$ (Augustine and Moses Jr. 2018).

   Why it matters. Dispersion is the first step of most agent algorithms and has the most open questions in the atlas; a $\log \Delta$ memory lower bound, or its removal, would settle a question shared by all of them.

   Settings: Dispersion, Dispersion.

5. **The asynchronous landscape of robots in the plane.** Can robots with internal lights meet in ASYNC (Flocchini et al. 2016a; Wada et al. 2026)? Can oblivious robots scatter in ASYNC (Bramas and Tixeuil 2016)? Is point convergence with limited visibility solvable in ASYNC without measurement errors (Kirkpatrick et al. 2024)?

   Why it matters. These questions decide where models separate under asynchrony (see the model grid on the plane model page); their answers carry over to every plane problem, and the first would settle five separations that rest on an unproved claim (Flocchini et al. 2023).

   Settings: Gathering, Scattering, Convergence.

6. **The time of arbitrary pattern formation.** Bound the number of epochs an arbitrary pattern formation algorithm needs in ASYNC; only solvability is known (Cicerone et al. 2019).

   Why it matters. Pattern formation has the largest family of special cases in the atlas (circle and uniform circle formation, mutual visibility, k-grouping, partitioning and more), and a time bound would carry over to them.

   Settings: Pattern formation.

7. **How many faults can be tolerated.** "The maximum number of faults under which a solution is still feasible, for gathering and other tasks, has yet to be established" (Agmon and Peleg 2006), in particular in ASYNC; and polynomial-time Byzantine gathering in graphs with f Byzantine agents and teams of close to $f+1$ good agents (Bouchard et al. 2016).

   Why it matters. Fault thresholds recur across gathering, black hole search, dispersion, evacuation and search; the arguments that bound one usually transfer.

   Settings: Gathering, Gathering in graphs.

8. **Leader election in programmable matter.** Deterministic leader election without chirality in linear time (the best is $O(L_{out}n^2)$ Emek et al. 2019), and in $O(L_{out}+D)$ rounds under weaker schedulers and without disconnection (Dufoulon et al. 2021).

   Why it matters. Shape formation, coating and convex hull formation start by electing a leader, so their deterministic bounds depend on it.

   Settings: Leader election in programmable matter, Leader election in programmable matter.

9. **Exact constants for evacuating a disk.** Close the gap for two face-to-face robots: $3+\pi/6+\sqrt{3} \approx 5.255$ (Czyzowicz et al. 2020d) against ≈ 5.6234 (Disser and Schmitt 2019).

   Why it matters. The question is old and well studied; its techniques shape the whole search family, though its reach stays inside it.

   Settings: Evacuation.

Claude (Anthropic) was used in cataloguing the problems.

10. **Compression and its mixing time.** What happens for $2.17 \leq \lambda \leq 2 + \sqrt{2}$, and how fast does the chain compress (Cannon et al. 2016)?

    Why it matters. Separation, alignment and bridging generalise compression, so an answer, and especially a mixing-time technique, would transfer to all three.

    Settings: Compression.

Who works on what

Counted by distinct authors of the papers the atlas cites, gathering in the plane (64 people), exploration by one agent (57) and dispersion (42) lead, followed by convergence, collaborative exploration and evacuation. The counts measure the atlas's coverage rather than the whole literature: gathering and exploration have far larger literatures than the atlas cites, and the youngest problems (flocking, alignment, subgraph counting) have a single paper each. Counts as of version 2026-10-10.

Every open question

Every open question recorded in the atlas, grouped by problem and ordered along each problem's assumption tree.

Graph exploration by a single agent

Arbitrary graphs, perpetual exploration (p. 93)

Q1. What is the time of memory-optimal perpetual exploration, and is there a trade-off between memory and time?

Trees, exploration with stop (p. 94)

Q1. Can exploration with stop on trees use asymptotically less memory than exploration with return? (Open problem of Diks et al. 2004.) Equivalently: close $\Omega(\log \log \log n)$ versus $O(\log n)$.

Exploration with return and pebbles (p. 95)

Q1. How much memory is needed to encode the agent's transition function? The naive table needs $(\log n)^{O(1)}$ bits (Remark 1 of Disser et al. 2019).

Exploration with stop when n is known (p. 96)

Q1. Is $\Omega(\log n)$ bits necessary when n is known? The trap arguments (Fraigniaud et al. 2005a) are stated for agents that know nothing.

Randomized exploration when n is known (p. 97)

Q1. What is the memory needed by a randomized agent that must stop correctly with high probability, as a function of n?

Exploration with stop and one pebble (p. 98)

Q1. Is $\Omega(D \log \Delta)$ memory needed for exploration with stop by an agent with one pebble (Fraigniaud et al. 2005b)?

Claude (Anthropic) was used in cataloguing the problems.

Port numbers chosen by the designer (periodic exploration) (p. 99)

Q1. Close the constant gaps: $2n-2$ versus $3.5n-2$ for constant memory, $2.8n-O(1)$ versus $4n-2$ for oblivious agents (Czyzowicz et al. 2012c; Kosowski and Navarra 2012).

Q2. Trade-offs between period and memory: the only lower bound for agents with memory does not depend on memory.

Directed graphs, one pebble (p. 100)

Q1. Close $\Omega(n \log \Delta_{out})$ versus $O(n\Delta_{out} \log n)$ bits for exploring digraphs with one pebble.

Q2. Determine the trade-off between time and memory: polynomial-time exploration with stop is known only with $O(n^2\Delta_{out} \log n)$ bits and $O(\log \log n)$ pebbles, far from the $\Omega(n \log \Delta_{out})$ lower bound (Section 5 of Fraigniaud and Ilcinkas 2004).

Directed graphs, one pebble, bound on n known (p. 101)

Q1. Improve the $O(N^2n^6\Delta_{out}^2)$ running time (Section 5 of Bender et al. 2002).

Directed graphs with labelled nodes (p. 102)

Q1. Close $\Omega(\phi^2 m)$ versus $O(\phi^8 m)$ for online exploration of digraphs with deficiency ϕ.

Temporal graphs, schedule known (p. 103)

Q1. Tight bounds when the underlying graph is a grid, has bounded degree or is planar (Erlebach et al. 2021). For bounded degree and for planar underlying graphs the best bounds known are $\Omega(n \log n)$ and $O(n^{3/2}\sqrt{\log n})$ (Bastide et al. 2025); the upper bound holds for grids too.

Q2. For underlying maximum degree Δ, close the gap between $\Omega(\Delta n(1+\log(n/\Delta)))$ (Carnevale 2026) and $O(\min\{n^2, n^{3/2}\sqrt{\Delta \log n}\})$ (Bastide et al. 2025); for constant Δ the bounds are $\Omega(n \log n)$ and $O(n^{3/2}\sqrt{\log n})$. The upper bound needs only that every snapshot has maximum degree at most Δ.

Q3. When at most μ edges are missing in each snapshot, remove the factor μ between $O(\mu n \log \mu)$ (Lahtin and Zamaraev 2027) and $\Omega(n \log \mu)$ (as reported by Lahtin and Zamaraev 2027).

T-interval-connected graphs, labelled nodes (p. 105)

Q1. Close the gap on the window, $\Omega(m)$ against $O(\zeta m + n \log^2 n)$; it is at most a $\log^2 n$ factor and remains when $m = n^{1+o(1)}$ (Sudo et al. 2026c).

Q2. How much smaller can the window be, and how much faster the exploration, for a randomized agent (Sudo et al. 2026c)?

Q3. Close the polylogarithmic gap on the time with 1-hop view, $\Omega(m)$ against $O(\zeta m + n \log^2 n)$, and determine exactly what the 1-hop view buys (Sudo et al. 2026c).

Collaborative exploration

Connected team, labelled graphs, shared knowledge (p. 250)

Q1. Is there an algorithm with sublinear overhead for $n > k$ agents? The paper gives experiments only and calls a bound an important milestone (Mutzari et al. 2026).

Arbitrary graphs, complete communication (p. 251)

Claude (Anthropic) was used in cataloguing the problems.

Q1. Is there an $o(k)$-competitive collaborative exploration algorithm for arbitrary graphs? (Stated as open in Cosson and Massoulié 2026.)

Trees, complete communication (p. 252)

Q1. Determine the competitive ratio of synchronous collective tree exploration: between $\Omega(\log k / \log\log k)$ and $O(\log^2 k)$.

Q2. Do randomized algorithms help? A randomized lower bound of $\Omega(\sqrt{\log k}/\log\log k)$ is attributed to Ortolf and Schindelhauer (SPAA 2012), not checked here. Cosson and Massoulié conjecture they do not (Cosson and Massoulié 2025).

Trees with $k = n$ agents (p. 253)

Q1. For $k = n$ and trees of depth D well above $\sqrt{\log n}$ (e.g. $D = \sqrt{n}$), is $\Omega(D^2)$ still necessary?

Large teams, complete communication (p. 253)

Q1. Which team size is necessary and sufficient for $O(D)$-round exploration of graphs of larger diameter? (Stated in Dereniowski et al. 2015a.)

Trees, communication by whiteboards only (p. 255)

Q1. Is there a separation between write-read and complete communication? (Cosson and Massoulié 2025) conjecture that centralised communication helps.

Trees, asynchronous, whiteboards only (p. 255)

Q1. Is there a distributed asynchronous algorithm with polylogarithmic competitive ratio?

Anonymous graphs with $k = n$ agents (p. 257)

Q1. Can n agents explore an anonymous tree in $o(n)$ rounds, e.g. $O(D^2)$, with local communication? With labelled nodes and global communication D^2 suffices (Disser et al. 2020).

Dispersion

Asynchronous agents (p. 265)

Q1. Can an asynchronous algorithm let every agent **terminate** in polynomial time without knowledge of the graph? The only terminating asynchronous algorithm known uses $O(\Delta^{\min(D,k)+1})$ time (Kshemkalyani 2025); the linear-time algorithms keep settled agents active.

Q2. Semi-synchronous schedulers sit between the two models; no separation or dedicated bound is known.

Arbitrary graph, arbitrary initial configuration (p. 266)

Q1. Can the memory be reduced to $O(\log k)$ bits, independent of Δ, while keeping $O(k)$ time? Equivalently, is there an $\Omega(\log \Delta)$ memory lower bound for deterministic agents with identifiers? The bound $\Omega(\log(k+\Delta))$ is quoted in several papers but only the $\log k$ term is argued.

Q2. Can all agents **terminate explicitly** in $O(k)$ rounds without any knowledge of k, m or Δ? (Sudo et al. 2026a) terminates the rooted case in $O(k)$ extra rounds but needs global upper bounds in the general case; (Kshemkalyani et al. 2025b) does not address termination detection for arbitrary initial configurations.

Claude (Anthropic) was used in cataloguing the problems.

Q3. **Ongoing dispersion** (Kshemkalyani and Ali 2019): agents that have settled must periodically leave (for example to recharge) and the configuration must be re-dispersed; no bounds beyond the original proposal are known.

Rooted initial configuration (p. 268)

Q1. Is $O(k)$ time achievable with $O(\log k)$ bits? Memory $O(\log(k+\Delta))$ is needed by every known algorithm to remember a port number.

Global communication (p. 268)

Q1. Does global communication buy anything for dispersion? Since $O(k)$ time with $O(\log(k+\Delta))$ bits holds with local communication, a separation could only come from memory ($O(\log k)$ bits), from termination detection, or from the BFS-style question of achieving time close to $O(D)$-dependent bounds for small D.

Q2. (Kshemkalyani et al. 2022) asks for $O(\min(m, k\Delta))$ time with only $O(\log(k+\Delta))$ bits in the global model; this is implied by the later local-model result, so the live version of the question is the memory question above.

Silent agents (no communication) (p. 269)

Q1. Close the time gap on general graphs: is $O(k + \log L)$ achievable, as on rings?

Q2. Arbitrary (multi-source) initial configurations on general graphs; only rings are solved (Das et al. 2026a).

Q3. Weaker sensing: agents that only learn whether they are alone and whether the count decreased, and agents that wake up at different times (a 2026 ICDCIT paper by Gorain, Mandal and Sharma treats this; its bounds are not recorded here).

Oriented grids (p. 270)

Q1. Rectangular $W \times H$ oriented grids with $W \le H$: from a corner at most $W(t+1)$ nodes are reachable in t rounds, so $\lceil k/W \rceil - 1$ rounds are needed; an algorithm matching a bound that depends on W and H, with local communication and no knowledge of W and H, is open (Kumar et al. 2026a).

Q2. $O(\sqrt{k})$ rounds on oriented grids with asynchronous agents, or on the oriented torus (Kumar et al. 2026a).

Q3. Dispersion in $O(\sqrt{k})$ rounds on oriented grids with explicit termination: every agent knows when it may stop (see notes).

Crash faults on oriented grids (p. 272)

Q1. Crash-tolerant dispersion in $O(\sqrt{k})$ rounds on oriented grids for $k = o(n)$; the fault-free case is solved by (Kumar et al. 2026a) (preprint), which names fault tolerance as future work.

Weak Byzantine agents on oriented grids (p. 272)

Q1. Reduce the memory below $O(n \log n)$ bits on grids with weak Byzantine agents.

Q2. Strong Byzantine agents (able to fake identifiers) on grids (Banerjee et al. 2025).

Q3. Weak Byzantine agents on unoriented grids, and on oriented grids with $k = o(n)$ agents in $O(\sqrt{k})$ rounds; fault tolerance is named as future work in (Kumar et al. 2026a).

Unoriented grids (p. 273)

Claude (Anthropic) was used in cataloguing the problems.

Q1. Can $k \leq n$ agents disperse on an **unoriented** grid in $O(\sqrt{k})$ rounds with **local** communication? On oriented grids $O(\sqrt{k})$ rounds are reached with local communication (Kumar et al. 2026a) (preprint), and with global communication by (Kshemkalyani et al. 2020c) (as reported); asked in (Banerjee et al. 2025), and for unoriented grids in (Kumar et al. 2026a).

Q2. A lower bound in terms of k for $k = o(\sqrt{n})$ with $O(\log n)$ memory on unoriented grids (Banerjee et al. 2024).

Q3. Unoriented grids with no knowledge of n and with faults, matching the fault-free bounds (Banerjee et al. 2024).

Crash faults on unoriented grids (p. 274)

Q1. Unoriented grids with crashes: close the gap between $O(\sqrt{n}\log n)$ rounds with $O(\sqrt{n}\log n)$ bits and the fault-free $O(\sqrt{n})$ rounds with $O(\log n)$ bits (Banerjee et al. 2024).

Rings and paths (p. 275)

Q1. Anonymous randomized agents disperse from a **rooted** ring configuration with $O(1)$ bits in $O(n)$ rounds (Molla and Moses Jr. 2019). From an **arbitrary** configuration the known algorithm never terminates; is $O(1)$-bit dispersion with termination possible on rings?

Silent agents on rings (p. 276)

Q1. Unoriented rings, and rings where agents cannot detect count changes.

Weak Byzantine agents on rings (p. 276)

Q1. Keep $O(n)$ time and $O(\log n)$ bits while dropping any of: identifiers in $[1, n]$, knowledge of f, the bound $f \leq (n-4)/17$, the follow primitive (Molla et al. 2021a).

Q2. Asynchronous Byzantine dispersion on rings (Molla et al. 2021a).

Trees (p. 277)

Q1. (Augustine and Moses Jr. 2018) asked for $o(n)$-round dispersion on arbitrary trees. Precisely: for $k = n$ and an arbitrary initial configuration, is there an algorithm whose time is bounded by a function of D alone, ideally $O(D^2)$ as in the rooted case? No tree-specific algorithm has appeared since 2018; the linear-time general algorithms give only $O(n)$.

Q2. With global communication, trees admit $O(D \cdot \max(D, k))$ rounds with $O(\max(D, \Delta \log k))$ bits (arXiv version of Kshemkalyani et al. 2020b); whether the $\max(D, k)$ factor can be reduced to D, and whether local communication suffices, is open.

Rooted trees (p. 278)

Q1. For $k < n$ the lower bound $\Omega(D^2)$ no longer applies (only k nodes must be reached). Whether $O(\min(k, D^2))$ time is achievable, and whether it is necessary, has not been addressed.

Q2. For $k = n$ agents rooted on a tree of height D much larger than $\sqrt{\log n}$, is $\Omega(D^2)$ time necessary? The known bound is existential in D (Disser et al. 2020).

Crash faults (p. 279)

Q1. Is $O(k)$ time achievable with crash faults, or does fault tolerance cost a factor of Δ? No lower bound beyond the fault-free $\Omega(k)$ is known, even with $O(\log(k + \Delta))$ memory (Chand et al. 2025a).

Q2. Simultaneously $O(\min(m, k\Delta))$ (or better) time and $O(\log(k+\Delta))$ memory for arbitrary configurations, without knowledge of f, ℓ, Δ or m (Chand et al. 2025a; Pattanayak et al. 2023).

Claude (Anthropic) was used in cataloguing the problems.

Q3. Crash faults under asynchrony: no result exists.

Weak Byzantine agents (p. 280)

Q1. Faster Byzantine dispersion on general graphs: the time is dominated by Byzantine gathering, and the authors ask whether building a map before settling is necessary (Molla et al. 2021b).

Q2. Tolerating more than $n/2$ weak Byzantine agents on general graphs (possible only on the restricted class so far) (Molla et al. 2021b).

Q3. Byzantine dispersion with bounded memory per agent, on trees, with $k < n$, or under asynchrony: nothing is known (Molla et al. 2021a; Chand et al. 2025a).

Strong Byzantine agents (p. 281)

Q1. Polynomial-time strong-Byzantine dispersion from arbitrary configurations without a trusted agent, and the largest tolerable f.

Q2. Asynchronous versions: a slow agent cannot be distinguished from a silent Byzantine one (Moses Jr. and Redlich 2024).

Anonymous agents with randomness (p. 282)

Q1. Close the gap between $\Omega(k)$ and $\Theta(k^2)$ rounds for anonymous agents from a rooted configuration with $O(\log \Delta)$ bits (Das et al. 2023).

Q2. Can anonymous randomized agents disperse from an **arbitrary** configuration and then stop, rather than staying active forever? (Molla and Moses Jr. 2019; Das et al. 2023)

Q3. On rings the memory bound is $O(1)$; is $O(1)$-bit dispersion with termination possible on rings from arbitrary configurations?

More agents than nodes (p. 283)

Q1. **Deterministic** dispersion with $k > n$ agents with identifiers on static graphs has not been studied: no algorithm and no lower bound beyond $\Omega(k)$ is recorded in the literature surveyed here. (Related work exists only on dynamic rings and dynamic graphs under the name balanced dispersion.)

Q2. Is $O(k)$ time possible for rooted configurations when $k > n$, matching the $k \leq n$ case?

Q3. Terminating dispersion with $k > n$ on general rooted graphs with $O(\log \Delta)$ bits per agent (Open Problem 2 of Molla and Moses Jr. 2019).

Directed graphs (p. 284)

Q1. Close the time gaps; in particular $O((k + f)k\Delta_{out})$ with f crashes (Italiano et al. 2025).

Q2. BFS-based dispersion in the local model, without 1-hop communication (Italiano et al. 2025).

Q3. Dispersion on dynamic directed graphs, and in semi-synchronous and asynchronous settings (Italiano et al. 2025).

Directed graphs, 1-hop communication (p. 285)

Q1. A BFS-based algorithm with $O(\log(k + \Delta_{out}))$ bits per agent, as with local communication (Italiano et al. 2025).

Q2. Does $O(kD)$ improve to $O(k + D^2)$ or $O(k)$ with 1-hop communication?

Dynamic graphs (1-interval connected) (p. 286)

Claude (Anthropic) was used in cataloguing the problems.

Q1. Which combination of communication range and visibility is **necessary** for dispersion on T-interval connected graphs with $T > 1$, where the impossibility proof for $T = 1$ does not apply? (Saxena and Mondal 2025)

Q2. Dynamic graphs with a bounded number of missing edges per round: (Saxena et al. 2026a) gives impossibilities on 1-bounded 1-interval connected graphs for several weaker communication and visibility combinations and an $O(k)$ algorithm with global communication and 1-hop visibility; the local-communication case with few missing edges is not settled.

Dynamic graphs with global communication (p. 287)

Q1. Byzantine agents on dynamic graphs (Kshemkalyani et al. 2020a).

Q2. Memory $O(\log k)$ instead of $O(\log \max(k, T))$ on T-path connected graphs when T need not be known (Saxena and Mondal 2025).

Dynamic rings (p. 287)

Q1. Visibility strictly between none and full: how much must an agent see of a dynamic ring for dispersion to be possible? (Agarwalla et al. 2018)

Q2. Achiral agents on even rings of size at least 6 under vertex permutation and 1-interval connectivity: is $O(n)$ time possible from arbitrary configurations? (Agarwalla et al. 2018); $O(\ell n)$ is known without vertex permutation (Mondal et al. 2025b).

Uniform deployment on graphs

Dynamic rings, all agents but one (p. 315)

Q1. Is $O(n)$ time possible on a 1-interval connected ring? (Shibata et al. 2022a) reach $O(n \log k)$ against $\Omega(n)$ and leave the lower bound on time to future work.

Q2. Can almost uniform deployment be reached without chirality or without strong multiplicity detection (Shibata et al. 2022a)?

Treasure hunt

Anonymous finite graphs (p. 112)

Q1. Treasure hunt when the treasure may appear after the agent starts and the agent does **not** see its entry port: "We do not know if an efficient solution is possible when backtracking is not allowed" (Ta-Shma and Zwick 2014).

Q2. Do strongly universal traversal sequences exist? They exist if and only if treasure hunt has a solution independent of the start time (Ta-Shma and Zwick 2014).

Q3. Is there a strongly universal exploration sequence with cover time $O(n^\theta)$, or $O(n^\theta \log n)$, for fixed-degree graphs? (Xin 2026) (preprint) reaches $O(n^\theta \log n (\log \log n)^{1+\varepsilon})$ and shows that its recurrence cannot keep linear length with periods $\Theta(2^j j)$ (Corollary 10.2), which is "not a lower bound for arbitrary SUESs".

Labelled nodes, locally finite graphs (p. 115)

Q1. Remove the $O(\log d)$ factor between $O(e(d) \log d)$ and $\Omega(e(d))$ (Bouchard et al. 2023a).

Claude (Anthropic) was used in cataloguing the problems.

Dynamic graphs, connected at every step (p. 115)

Q1. With flags in VP(ξ), the lower bound of $\xi(n-3)$ steps for one deterministic agent is proved for $\xi \geq 2n$; does $\Omega(\xi n)$ hold for every $\xi \geq \lceil (n-1)/2 \rceil$, the smallest window for which VP(ξ) is non-empty? (Ayoubi et al. 2026)

Q2. What is the competitive ratio of randomized agents in VP(ξ) against an adaptive online adversary? (Ayoubi et al. 2026)

Q3. Which of these results extend to vertex-permuted graphs other than rings? (Ayoubi et al. 2026)

Rendezvous

Asynchronous labelled agents (p. 123)

Q1. Find an optimal-cost asynchronous rendezvous algorithm in arbitrary graphs (open problem of Pelc 2019).

Labelled agents, arbitrary finite graphs (p. 124)

Q1. Find an optimal-time synchronous rendezvous algorithm for labelled agents in arbitrary graphs (open problem of Pelc 2019).

Anonymous agents: memory (p. 126)

Q1. Is there a universal algorithm that meets for every feasible start in time polynomial in n and τ? Posed in (Pelc and Yadav 2019b).

Labelled agents in infinite trees (p. 127)

Q1. Close the gap between $O(\text{ball}(d)\log L)$ and $\Omega(\text{ball}(d) + d\log L)$ in unoriented regular trees. Posed in (Bhagat and Pelc 2024).

Q2. Find efficient rendezvous algorithms for arbitrary infinite graphs without extra knowledge such as the agents' positions. Posed in (Bhagat and Pelc 2024).

Labelled agents in infinite trees, simultaneous start (p. 128)

Q1. In oriented trees with simultaneous start, can agents meet in time $O(d\log L)$ without knowing anything about d or L? Posed in (Bhagat and Pelc 2024).

Labelled agents, graphs with infinite degrees (p. 130)

Q1. Find an optimal rendezvous algorithm, "even for finite graphs" (Pattanayak and Pelc 2024b).

Pattern formation

ASYNC, no agreement on coordinates (p. 386)

Q1. Bound the number of epochs needed by an arbitrary pattern formation algorithm in ASYNC; no time analysis is recorded.

Randomized, ASYNC, chirality (p. 387)

Claude (Anthropic) was used in cataloguing the problems.

Q1. A proof that does not rely on the disputed algorithm of (Fujinaga et al. 2015), for example by combining the symmetry-breaking phase of (Yamauchi and Yamashita 2014) with the algorithm of (Cicerone et al. 2019).

Opaque robots with lights, ASYNC, one axis, rigid (p. 390)

Q1. The minimum number of colours, and formation with chirality only or with no agreement (Bose et al. 2021a).

Limited visibility (p. 391)

Q1. Arbitrary pattern formation by oblivious robots with limited visibility from a general (connected) initial configuration (Hahn et al. 2024).

Gathering

Rendezvous, internal light (FSTA), SSYNC, non-rigid (p. 439)

Q1. Rendezvous with internal lights in non-rigid SSYNC when δ is unknown.

Gathering, internal light (FSTA), SSYNC, non-rigid (p. 440)

Q1. Gathering with internal lights in SSYNC from every start, with more than 2 colours, and in ASYNC (Otaka et al. 2026).

Gathering, external light (FCOM), SSYNC, rigid (p. 441)

Q1. Gathering with external lights in ASYNC (Otaka et al. 2026).

Gathering, full lights, SSYNC, no multiplicity detection (p. 442)

Q1. Gathering with lights without chirality (Terai et al. 2023).

Self-stabilizing gathering, full lights, SSYNC, strong multiplicity detection (p. 443)

Q1. Does self-stabilizing gathering with 2 colours extend to ASYNC (Bonnet et al. 2026a)?

Q2. Is self-stabilizing SSYNC gathering with lights solvable with weak or no multiplicity detection, or with non-rigid moves? The algorithm of (Bonnet et al. 2026a) uses strong multiplicity detection and rigid moves.

Homonymous robots, SSYNC (p. 445)

Q1. What is the minimum number of identifiers for Byzantine-tolerant gathering? (Asahiro and Yamashita 2025)

Gathering, full lights, ASYNC, no multiplicity detection (p. 446)

Q1. Is 3 the minimum number of colours for ASYNC gathering with lights, and can chirality be dropped (Nakai et al. 2021)?

Q2. Self-stabilizing gathering of n robots with lights (Heriban et al. 2018).

Rendezvous, external light (FCOM), ASYNC, non-rigid (p. 447)

Q1. Rendezvous with external lights in non-rigid ASYNC when δ is unknown (Flocchini et al. 2016a; Wada et al. 2026).

Claude (Anthropic) was used in cataloguing the problems.

Rendezvous, external light (FCOM), ASYNC, rigid (p. 448)

Q1. Close the gap between 3 and 6 colours (Wada et al. 2026).

ASYNC with weak multiplicity detection (p. 448)

Q1. How many epochs does oblivious ASYNC gathering with weak multiplicity detection need?

Rendezvous, internal light (FSTA), ASYNC (p. 450)

Q1. Can robots with internal lights solve rendezvous in ASYNC (Flocchini et al. 2016a; Wada et al. 2026)?

Crash faults, SSYNC (p. 451)

Q1. "The maximum number of faults under which a solution is still feasible, for gathering and other tasks, has yet to be established" (Agmon and Peleg 2006), in particular in ASYNC.

Crash faults, ASYNC, restricted start (p. 452)

Q1. Gather despite crashes in ASYNC from every solvable initial configuration, symmetric ones included, with weak multiplicity detection.

Limited visibility with a compass, rigid, ASYNC (p. 455)

Q1. Does $O(\Delta_0)$-epoch gathering need rigid moves, or a visibility radius larger than the connectivity range (Poudel and Sharma 2021)?

Limited visibility, one common axis, rigid, ASYNC (p. 456)

Q1. Can asynchronous robots with one common axis and limited visibility gather at a point, and in $O(\Delta_0)$ epochs? (Poudel and Sharma 2021) reach only a horizontal segment of length 1, because two robots on a horizontal line, disagreeing on east and west, may swap places forever.

Limited visibility, FSYNC, disoriented (p. 457)

Q1. Is $\Omega(\Delta_0^2)$ necessary for every protocol, not only local convex-hull ones?

Fat robots, few robots, no common orientation (p. 457)

Q1. Gathering any number of fat robots without chirality (Agathangelou et al. 2013).

Fat robots, chirality, any number (p. 458)

Q1. Remove the chirality assumption (Agathangelou et al. 2013).

Transparent fat robots, no chirality (p. 459)

Q1. Gathering transparent fat robots from symmetric starts; the paper proves no general impossibility (Chaudhuri and Mukhopadhyaya 2015).

Uniform circle formation

ASYNC, oblivious, disoriented (p. 401)

Q1. Can oblivious robots without lights form a uniform circle in $O(1)$ epochs, with rigid or non-rigid moves? (Feletti et al. 2026a)

Claude (Anthropic) was used in cataloguing the problems.

Opaque robots with lights, ASYNC (p. 402)

Q1. What is the exact minimum number of colours for $O(1)$-epoch uniform circle formation in ASYNC? (Feletti et al. 2026a)

Q2. Uniform circle formation with non-rigid moves in the luminous opaque model (Feletti et al. 2026a).

Convergence

Limited visibility, asynchronous (p. 471)

Q1. Is point convergence with limited visibility solvable in full ASYNC, without perception or motion errors? (Kirkpatrick et al. 2024)

Q2. Is there a problem that separates k-ASYNC from SSYNC? (Kirkpatrick et al. 2024)

Linear search (cow path)

k robots on a line, Byzantine faults (p. 503)

Q1. Determine the optimal competitive ratio $B(k, f)$ for Byzantine search on the line; already $B(3, 1)$ is open.

Evacuation

Disk, two wireless robots of different speeds (p. 516)

Q1. Optimal evacuation for $1 < v < 2.75$, for more than two robots, and face-to-face (Lamprou et al. 2020).

Two robots, two exits, wireless (p. 518)

Q1. Close the gap between $2R$ and $(1 + \pi/3)R$ for two wireless robots and two exits (Pattanayak et al. 2018).

Disk, search-and-fetch, two wireless robots (p. 518)

Q1. Lower bounds that separate wireless from face-to-face search-and-fetch (Georgiou et al. 2019).

Two face-to-face robots, disk (p. 519)

Q1. Close the gap between $3 + \pi/6 + \sqrt{3} \approx 5.255$ and 5.6234 for two face-to-face robots.

Two robots with asymmetric communication, disk (p. 520)

Q1. Give an explicit lower bound above ≈ 4.826 for sender/receiver evacuation.

Disk, priority evacuation, queen and one servant (p. 520)

Q1. Close the gap between 4.568 and 4.819 (Georgiou and Wang 2025).

Disk, weighted evacuation of two wireless robots (p. 521)

Claude (Anthropic) was used in cataloguing the problems.

Q1. Close the gap for weighted evacuation of two wireless robots in the disk when $w > 1$ (Georgiou and Wang 2025).

Three wireless robots, disk (p. 522)

Q1. Find the optimal evacuation time for three wireless robots.

Three wireless robots, one crash fault (p. 522)

Q1. Tight bounds for disk evacuation with one crash-faulty robot among three.

Disk, k wireless robots, f Byzantine (p. 525)

Q1. Lower bounds for evacuation of k robots with f Byzantine faults from the disk (Behrouz et al. 2025).

Disk search, k wireless robots, one Byzantine (p. 526)

Q1. Search of the disk with $f \geq 2$ Byzantine robots: only the crash lower bound $1 + 2\pi(f + 1)/k$ and evacuation upper bounds are known (Georgiou et al. 2023b; Behrouz et al. 2025).

Priority evacuation (queen and servants), disk (p. 527)

Q1. Close the gap for priority evacuation; for one servant the bounds differ by about 0.25 (a 2024 preprint, not yet in the bibliography).

Disk, k face-to-face robots, f Byzantine (p. 528)

Q1. Lower bounds for face-to-face evacuation of the disk with Byzantine robots (Behrouz et al. 2025).

Disk, k face-to-face robots, f crash faults (p. 528)

Q1. Tight bounds for face-to-face evacuation of the disk with crash faults.

Line, weighted evacuation of two wireless robots (p. 529)

Q1. Weighted evacuation with more than two robots (Georgiou and Lucier 2023).

Line, weighted evacuation of two face-to-face robots (p. 530)

Q1. Lower bounds for weighted evacuation of two face-to-face robots on the line (Georgiou and Lucier 2023).

Line, priority evacuation, queen and one wireless servant (p. 530)

Q1. A lower bound for priority evacuation of a queen and one wireless servant on the line.

Line, priority evacuation, queen and one face-to-face servant (p. 531)

Q1. A lower bound for priority evacuation of a queen and one face-to-face servant on the line.

Line, a sender and a receiver (p. 532)

Q1. Close the gaps for one sender and many receivers ($2 + \sqrt{5}$ against 5) and for many senders and one receiver (3 against about 5.681) (Czyzowicz et al. 2021d).

Line, $2f + 1$ wireless robots, f crash faults (p. 534)

Q1. Tight bounds for evacuation with $k = 2f + 1$ crash or Byzantine robots (Czyzowicz et al. 2021e).

Triangle, any number of wireless robots at the centroid (p. 536)

Claude (Anthropic) was used in cataloguing the problems.

Q1. The optimal evacuation time of $k = 3, \ldots, 7$ wireless robots from the centroid of the triangle.

Square, any number of wireless robots at the centre (p. 537)

Q1. The optimal evacuation time of k wireless robots from the centre of the square for fixed k.

Triangle, two wireless robots starting on the boundary (p. 538)

Q1. Face-to-face evacuation of general triangles, and wireless evacuation of obtuse triangles (Georgiou and Jang 2026).

m rays, $k = m$ wireless robots at the junction (p. 539)

Q1. Starting off the junction with $k = m$: close $2 + \sqrt{7/3} \approx 3.528$ against 4 (Brandt et al. 2020b).

Q2. Fewer robots than rays ($1 < k < m$): the bounds $3 + 2\lfloor (m-1)/k \rfloor$ and $1 + 2\frac{m-1}{k}(1 + \frac{k}{m-1})^{1+(m-1)/k}$ differ (Brandt et al. 2020b).

Square, two wireless robots starting on the boundary (p. 539)

Q1. Optimal evacuation of the square by $k \geq 3$ wireless robots from a point of the boundary other than a corner.

Black hole search

Eventually emerging black hole, synchronous agents, arbitrary graphs (p. 299)

Q1. Do three synchronous agents suffice on arbitrary graphs (Kaur et al. 2026a)?

Eventually emerging black hole, synchronous agents, oriented ring (p. 300)

Q1. Is one algorithm optimal in every respect at once: three agents, no knowledge of n, no pebble and $O(n)$ rounds? Left open by (Bonnet et al. 2026b).

Eventually emerging black hole, asynchronous agents, oriented ring (p. 301)

Q1. How many asynchronous agents without pebbles are needed with implicit termination: between 3 and n? With two pebbles four suffice (Bonnet et al. 2026b).

Dynamic graphs with whiteboards (p. 303)

Q1. With at most one missing edge per round, do four or five co-located agents suffice? Three do not (Kaur et al. 2026b).

Q2. With scattered agents and at most one missing edge per round, do $2\deg(v_{BH}) + 1$ or $2\deg(v_{BH}) + 2$ agents suffice? $2\deg(v_{BH})$ do not (Kaur et al. 2026b) and $2\deg(v_{BH}) + 3$ do (Bileski and Miller 2026).

Q3. With up to μ missing edges per round, can $O(\mu)$ co-located agents locate the black hole in polynomial time? The known algorithm with 6μ agents takes exponential time (Kaur et al. 2026b).

Gathering in graphs

Labelled agents (p. 136)

Q1. Gathering without communication when no bound on n is known.

Claude (Anthropic) was used in cataloguing the problems.

Weakly Byzantine agents (p. 137)

Q1. Can weakly Byzantine agents solve self-stabilizing perpetual gathering (from arbitrary memory states, all good agents eventually always at one node) with no global knowledge? (Hirose et al. 2025) solve it with upper bounds on n, k and f and the length of the largest label of a good agent, and leave this open.

Byzantine agents (p. 138)

Q1. Polynomial-time Byzantine gathering with teams of size close to $f + 1$.

Exploration of dynamic graphs by teams

Connected at every step, any graph (p. 308)

Q1. Does the algorithm with $2m - n + 1$ agents explore in polynomial time? Posed in (Dobrev et al. 2020a).

1-interval-connected rings (p. 309)

Q1. With a landmark but no chirality, is $O(n \log n)$ time necessary? (Di Luna et al. 2020b)

Scattering

Randomized, ASYNC (p. 477)

Q1. Solve scattering in ASYNC without multiplicity detection: Bramas and Tixeuil call ASYNC scattering "the open problem of ASYNC scattering" (Bramas and Tixeuil 2016, 2017); Léchine and Tixeuil solve it with detection of multiplicity at the robot's own point and leave the plane without detection open (Léchine and Tixeuil 2019).

Randomized, ASYNC, weak multiplicity detection (p. 478)

Q1. Bound the expected time of ASYNC scattering with weak multiplicity detection.

Randomized, ASYNC, rigid, weak multiplicity detection, no vision needed (p. 479)

Q1. Is there a polynomial bound on the expected time of blind rigid ASYNC scattering in the plane?

Pattern sequences

Oblivious, SSYNC, chirality (p. 410)

Q1. Form a sequence of patterns in ASYNC with oblivious robots (Das et al. 2015b).

Luminous robots, ASYNC, non-rigid, chirality (p. 410)

Q1. Close the gap between the colours used and the lower bound of Theorem 18 (Das et al. 2020).

Q2. Pattern sequences with lights without chirality, with faults, or self-stabilising (Das et al. 2020).

Claude (Anthropic) was used in cataloguing the problems.

Near-gathering

ASYNC, limited visibility, one common axis (p. 463)

Q1. Pagli, Prencipe and Viglietta conjecture that near-gathering is impossible without agreement on one axis.

SSYNC, limited visibility, disoriented (p. 463)

Q1. Near-gathering in $O(\Delta_0)$ epochs?

Mutual visibility

Finite state, SSYNC, non-rigid (p. 417)

Q1. Solve mutual visibility with finite-state robots under ASYNC (Bhagat and Mukhopadhyaya 2019).

Lights, ASYNC, rigid (p. 418)

Q1. Fewer colours with $O(1)$ epochs (Sharma et al. 2021a).

Q2. Time bounds with non-rigid moves.

Lights, ASYNC, non-rigid, one axis (p. 419)

Q1. Mutual visibility in non-rigid ASYNC without axis agreement, ideally with 2 colours (Di Luna et al. 2017).

Lights, FSYNC, rigid, chirality (p. 419)

Q1. What is the lower bound on the number of rounds, and can chirality and crossing paths be dropped (Vaidyanathan et al. 2015)?

Lights, mobility faults, SSYNC, non-rigid (p. 420)

Q1. Can mobility-fault-tolerant mutual visibility under a movement budget be solved without the common reference point and without any axis agreement (Shukla et al. 2026a)?

Q2. Does the budget-constrained algorithm extend to ASYNC (Shukla et al. 2026a)?

Lights, one crash, SSYNC, rigid, compass (p. 421)

Q1. Tolerating three or more crashes, and using 2 colours (Aljohani and Sharma 2018).

Lights, crash faults, ASYNC, one axis (p. 422)

Q1. Mutual visibility with several crashes when a crashed robot's light stops changing too, which (Poudel et al. 2021) call "a major open problem".

Q2. Two colours in ASYNC with crashes (Poudel et al. 2021).

Fat robots, ASYNC, rigid, one axis (p. 423)

Q1. Non-rigid moves, and SSYNC and ASYNC extensions of the 2-colour FSYNC algorithm (Alsaedi et al. 2023a).

Q2. Mutual visibility of fat robots in ASYNC with non-rigid moves and no axis agreement (Bhagat et al. 2024).

Claude (Anthropic) was used in cataloguing the problems.

k-grouping

k equal groups, randomized, SSYNC (p. 428)

Q1. Partitioning in ASYNC without a common orientation, by a randomized algorithm with no extra memory.

Minimum spanning tree by agents

Arbitrary start (p. 362)

Q1. An MST from an arbitrary start in $O(m + n \log n)$ rounds, or fewer, with $O(\log n)$ bits per agent and no knowledge of graph parameters.

Leader election in programmable matter

Deterministic, with chirality (p. 588)

Q1. Fast deterministic leader election under weaker (non-atomic) schedulers, and without letting the system disconnect (Dufoulon et al. 2021).

Deterministic, no orientation (p. 590)

Q1. Deterministic leader election without chirality in linear time.

Stationary, randomized, sequential, holes allowed (p. 592)

Q1. Prove the $O(n \log n)$ bound with high probability that (Warner and Scheideler 2026a) states without proof, and give a lower bound on rounds.

Q2. Can stationary randomized particles without a common orientation elect a unique leader on configurations with holes under the strong or the weak scheduler, which let several particles act at the same time?

Stationary, chirality, weak scheduler (p. 593)

Q1. Prove, or improve, the $O(n^2)$-round bound of (Bazzi and Briones 2019), which the paper states without proof, and give a lower bound on rounds.

Q2. Elect a bounded number of leaders deterministically, without moving, under the weak scheduler and without chirality, on configurations with holes; Bazzi and Briones conjecture that it is possible (Bazzi and Briones 2019).

Stationary, asynchronous, no orientation, no holes (p. 594)

Q1. Deterministic leader election in $O(L_{out} + D)$ rounds under a weaker scheduler (Dufoulon et al. 2021).

Stationary, reconfigurable circuits, synchronous, randomized (p. 594)

Q1. Give a lower bound on the rounds needed to elect a leader with circuits; neither paper read proves one.

SILBOT with holes (p. 595)

Claude (Anthropic) was used in cataloguing the problems.

Q1. Can SILBOT particles elect leaders from configurations with holes if the system may disconnect temporarily, without exterior awareness? The impossibility of (D'Angelo et al. 2020) assumes that connectivity is preserved.

Shape formation in programmable matter

SILBOT, chirality (p. 603)

Q1. Is line formation solvable in SILBOT when particles share just one direction instead of the full orientation (Navarra and Piselli 2023b)?

SILBOT, full orientation (p. 603)

Q1. Does 1-hop visibility suffice for line formation in SILBOT (Navarra and Piselli 2023b)?

Q2. Which other shapes can SILBOT particles form (Navarra and Piselli 2023b)?

Coating

SILBOT, chirality, deterministic (p. 607)

Q1. Coating in SILBOT without chirality (Navarra and Piselli 2023a).

Q2. Multi-layer coating, which "seems unfeasible" in SILBOT (Navarra and Piselli 2023a).

Compression

Markov chain, sequential (p. 610)

Q1. Behaviour for $2.17 \leq \lambda \leq 2 + \sqrt{2}$, and bounds on the convergence time.

Patrolling

Closed fence, one-way, distinct speeds (p. 571)

Q1. Close the factor $O(\log \log k)$ between the schedule of (Haeupler et al. 2019) and the bound $\sum_i v_i$ on the one-way circle.

Closed fence, two-way, distinct speeds (p. 572)

Q1. What is the largest ratio between optimal schedules and the best simple strategy on the two-way circle? It lies between 3/2 (Dumitrescu et al. 2014) and 2, since partition patrols $\sum_i v_i/2$ and no schedule patrols more than $\sum_i v_i$ (Proposition 2 of Czyzowicz et al. 2011).

Open fence, distinct speeds (p. 572)

Q1. Is computing an optimal schedule for given speeds on a segment hard?

Graphs with faulty patrollers (p. 573)

Q1. Is the decision version of fault-tolerant patrolling in NP? With k part of the input and integer edge lengths it is in PSPACE (Czyzowicz et al. 2017e).

Claude (Anthropic) was used in cataloguing the problems.

Q2. Optimal fault-tolerant patrolling of a segment when $f \geq k/2 - 1$, and for even f (Czyzowicz et al. 2017e).

Q3. Approximation algorithms for the optimal idle time on general graphs (Czyzowicz et al. 2017e).

Network decontamination

Dynamic graphs, monotone strategies (p. 322)

Q1. How many co-located agents are necessary and sufficient to clean every 1-interval connected dynamic graph monotonically, as a function of n, m and D? $D + 2(m - n + 1)$ suffice, and $n - 1$ (on $K_{n/2,n/2}$) and n (on wheels) can be necessary (Bar et al. 2026).

Exploration by oblivious robots

Rings, ASYNC, terminating (p. 168)

Q1. Can four robots explore and stop on every ring of odd size n under ASYNC? Four suffice under SSYNC for odd $n > 7$; the question is posed in (Lamani et al. 2010).

Q2. For which pairs (n, k) can k robots explore an n-node ring (Flocchini et al. 2013)? In particular, is exploration impossible whenever $\gcd(n, k) > 1$ and $k < n$, as conjectured in (Lamani et al. 2010)? Impossibility is proved only when k divides n (Flocchini et al. 2013), and for $k = 4$ with even n (Lamani et al. 2010).

Q3. What is the minimum number of robots when n is a multiple of 5? From model checking on rings of fewer than 18 nodes, Bérard et al. conjecture that the algorithm of (Flocchini et al. 2013) is correct for every odd k with $5 \leq k < 17$ from initial configurations with at most one axis of symmetry (Bérard et al. 2016); the lower bounds use more symmetric starts.

Rings, SSYNC, terminating (p. 170)

Q1. What is the minimum number of robots under SSYNC when n is a multiple of 10 (four and five robots are impossible)? This is the SSYNC case of the characterisation of solvable pairs (n, k) asked for in (Flocchini et al. 2013).

Rings, probabilistic, SSYNC, terminating (p. 171)

Q1. How many moves (or rounds) do four probabilistic robots need to explore a ring of n nodes and stop, and how does the time decrease when more robots are used? Posed in (Devismes et al. 2013), where termination is only shown to be finite.

Rings, ASYNC, exclusive perpetual (p. 172)

Q1. What is the minimum number of robots for exclusive perpetual exploration of a ring whose size is a multiple of 3, from every configuration without towers?

Q2. Would coin flips help in the cases where deterministic exclusive perpetual exploration is impossible? Posed in (Blin et al. 2010).

Q3. What is the largest team that can explore exclusively and perpetually from every configuration without towers? (Blin et al. 2010) claim $n - 5$ robots for odd $k \geq 7$ with $k \nmid n$, but an SMT check found collisions in their algorithm for 7 and 9 robots on rings of $k + 5$ nodes under FSYNC (Sangnier et al. 2020).

Claude (Anthropic) was used in cataloguing the problems.

Exploration by luminous robots

Grids, FSYNC, terminating, no chirality, $\varphi = 1$ (p. 187)

Q1. How many robots are needed for terminating grid exploration under FSYNC with $\varphi = 1$ and no chirality: three or four?

Grids, ASYNC, terminating, no chirality, $\varphi = 1$ (p. 189)

Q1. Close the gap between three and six robots for terminating grid exploration under ASYNC with $\varphi = 1$ and no chirality (Nagahama et al. 2022).

Grids, ASYNC, perpetual, chirality, $\varphi = 2$ (p. 192)

Q1. Can two luminous robots with common chirality and $\varphi = 2$ explore a finite grid perpetually under ASYNC (or SSYNC)?

Grids, ASYNC, perpetual, no chirality, $\varphi = 2$ (p. 192)

Q1. Is there an optimal ASYNC algorithm (three robots, $\varphi = 2$, no chirality) for opaque robots? Posed in (Bramas et al. 2026a).

Q2. Can two robots without chirality explore a grid perpetually with $\varphi \geq 2$ under FSYNC? (Rauch et al. 2021) state that they cannot for every finite φ, but prove it for $\varphi = 1$.

Grids, ASYNC, perpetual, chirality, $\varphi = 1$ (exclusive) (p. 193)

Q1. Is perpetual grid exploration with $\varphi = 1$ solvable under ASYNC when robots may share nodes (form towers)? Terminating exploration is, with three robots and three colours (Nagahama et al. 2022).

Grids, ASYNC, perpetual, no chirality, $\varphi = 3$ (p. 194)

Q1. Can two robots without chirality explore a grid perpetually with $\varphi \geq 2$ under FSYNC? (Rauch et al. 2021) state that they cannot for every finite φ, but prove it for $\varphi = 1$.

Rings, FSYNC, perpetual, up to f crashes (p. 195)

Q1. With $f + 2$ robots under FSYNC, how many colours are necessary for crash-tolerant perpetual ring exploration? The best algorithm uses $3f + 2$ (Ooshita et al. 2024).

Rings, ASYNC, perpetual, $\varphi = 1$ (p. 197)

Q1. Can three oblivious robots ($c = 1$) with $\varphi = 1$ explore a ring perpetually under SSYNC or ASYNC, or are two colours necessary?

Rings, FSYNC, terminating, $\varphi = 1$ (p. 198)

Q1. Characterise the initial configurations of three robots with $c \geq 3$ colours from which terminating exploration of a ring is solvable under FSYNC with $\varphi = 1$ (with two colours no universal algorithm exists Ooshita and Tixeuil 2022).

Rings, ASYNC, terminating, $\varphi = 1$ (p. 199)

Q1. Does any task separate SSYNC from ASYNC for myopic luminous robots on rings? The known exploration bounds coincide under both (Ooshita and Tixeuil 2022).

Tori, FSYNC, perpetual, no chirality, $\varphi = 2$ (p. 202)

Claude (Anthropic) was used in cataloguing the problems.

Q1. Can four oblivious robots with $\varphi = 2$ explore a torus perpetually without chirality?

Tori, FSYNC, perpetual, no chirality, $\varphi = 1$ (p. 203)

Q1. Can three robots with two colours and $\varphi = 1$ explore a torus perpetually without common chirality?

Triangular grids, FSYNC, perpetual, no chirality, $\varphi = 1$ (p. 203)

Q1. Is there a proof of perpetual triangular-grid exploration by two robots with $\varphi = 1$ for self-inconsistent robots, or with two colours?

Exploration of recolourable graphs

Recolourable nodes, oblivious agent, any graph (p. 214)

Q1. Is the colour complexity of recolourable exploration of arbitrary graphs 4, 5 or 6 (Takahashi et al. 2025; Hiraoka et al. 2026)?

Q2. Give a proof of the four-colour lower bound without computer assistance, or for a single graph (Hiraoka et al. 2026).

Q3. How many moves does a constant-colour algorithm need on general graphs? No bound is known; for graphs whose blocks are cliques or triangle-free, five colours and $O(n\Delta)$ moves suffice (Hiraoka et al. 2026).

Q4. How many colours suffice if the agent has a constant number of bits, or even one bit, of memory, or may move at random (Böckenhauer et al. 2023; Takahashi et al. 2025)?

Q5. Does forbidding recolouring back to c_0 change the colour complexity, or only the algorithms (Takahashi et al. 2025)?

Recolourable nodes on cacti (p. 215)

Q1. Is four colours enough for φ-free graphs, or for graphs of maximum degree three? Both lie between 4 and 5 (Hiraoka et al. 2026).

Once-colourable nodes, any graph (p. 216)

Q1. For graphs of circumference $\mathrm{circ}(G) \leq g$, close the gap between $\min(2g-2, n)$ and $\min(2g, n)$ colours (Böckenhauer et al. 2023).

Once-colourable nodes, n known (p. 217)

Q1. Is the colour complexity with known n equal to $n-2$ or $n-1$ (Böckenhauer et al. 2023)?

Rotor-router: a pointer in every node (p. 218)

Q1. For which graphs (other than paths) is the rotor-router exactly time-optimal among oblivious agents without the entry port (Menc et al. 2017)?

Rotor-router with pointers set by a designer (p. 220)

Q1. How few moves does a rotor-router with designer pointers need to visit every node, as opposed to locking in? In Hamiltonian graphs the pointers can follow a Hamiltonian cycle (Bampas et al. 2017).

Label-guided: colours fixed by a designer (p. 220)

Claude (Anthropic) was used in cataloguing the problems.

Q1. Is there a 1-bit (two-colour) labeling with which a finite automaton explores every graph (Cohen et al. 2008)?

Label-guided, oblivious agent (p. 221)

Q1. How many designer colours does an oblivious agent need to explore all graphs of maximum degree Δ (Cohen et al. 2008)?

Uniform deployment on a ring

Oriented ring, asynchronous, oblivious (p. 492)

Q1. Is knowledge of d necessary for exact deployment on an oriented ring? Which capabilities allow asynchronous exact deployment on unoriented rings? (Flocchini et al. 2008b)

Search with segment visibility

Two robots, unknown distance (p. 577)

Q1. Determine the optimal competitive ratio of two robots that do not know d: it lies between $\sqrt{3} + \pi/6 + \sqrt{17}/4 \approx 3.2864$ and ≈ 8.9285 (Kranakis 2026).

Q2. Do asymmetric, non-antipodal or non-spiral trajectories beat ≈ 8.9285? The antipodal spirals see only along a diameter through the origin, while a segment between two moving robots can sweep chords that miss it (Kranakis 2026).

Partitioning

Exactly n/k robots at each fixed point (p. 467)

Q1. Partitioning into k groups of size n/k at points chosen by the robots (no fixed points given).

Scattering of particles

SILBOT (p. 625)

Q1. Is scattering solvable in SILBOT with its original 2-hop view under adversarial activation? (Navarra et al. 2025) leave it open.

SILBOT with event-driven activation (p. 626)

Q1. Bound the rounds and moves of scattering under event-driven activation (Navarra et al. 2025).

Q2. Scattering with obstacles: unsolvable deterministically with a 1-hop view even under a synchronous scheduler (Theorem 7 of Navarra et al. 2025); which extra capability (chirality, a 2-hop view, randomised choices) makes it solvable?

Claude (Anthropic) was used in cataloguing the problems.

Separation of coloured particles

Markov chain, sequential (p. 613)

Q1. Bounds on the convergence (mixing) time.

Centralised transformations of modular robots

Rotation only (p. 638)

Q1. What is the complexity of connectivity-preserving transformability with rotations alone? The paper leaves it open (Section 7) (Michail et al. 2019).

Rotation and sliding (p. 639)

Q1. Prove the suspected $\Omega(n \log n)$ lower bound for universal transformations with line pushing (Almethen et al. 2020).

Q2. Distributed universal transformations with line pushing (Almethen et al. 2020).

Online exploration of weighted graphs

Arbitrary weighted graphs (p. 227)

Q1. Is there a deterministic algorithm with a constant competitive ratio (Kalyanasundaram and Pruhs Kalyanasundaram and Pruhs 1994, as reported by Baligács 2026)?

Q2. Is there an $o(\log n)$-competitive algorithm? Blocking with parameter χ (their δ) is $\Omega(\log n)$-competitive for $\chi = o(\log n / \log\log n)$ and for $\chi = \Omega(\log n)$, and $\Omega(\log n / \log\log n)$-competitive for every χ; the range in between is open (Baligács et al. 2026a).

Q3. Does randomization lower the competitive ratio on general graphs? It does on cycles (Baligács et al. 2026b).

Known graph, weight estimates (p. 228)

Q1. Does every graph admit a $\frac{1+\vartheta}{2}$-competitive algorithm when all weights are announced in the same interval $[1, \vartheta]$ with $\vartheta \leq 2$? Conjectured by (Gehnen et al. 2025).

Graphs excluding a fixed minor (p. 229)

Q1. How does the best constant depend on the excluded minor H, and on the genus for graphs of bounded genus (Baligács et al. 2026a)?

Planar graphs (p. 230)

Q1. Close 4 against 16 on planar graphs.

Weighted cycles, randomized agent (p. 231)

Q1. Determine the randomized competitive ratio on cycles: 1.1155 against 1.3144 (Baligács et al. 2026b).

Q2. Must an optimal randomized algorithm on cycles be forward-greedy (Baligács et al. 2026b)?

Claude (Anthropic) was used in cataloguing the problems.

Gathering by oblivious robots on graphs

Rings, round robin, no multiplicity detection (p. 147)

Q1. Which sequential schedulers between round robin and an arbitrary sequential scheduler still allow gathering on rings without multiplicity detection? (Navarra and Piselli 2025) ask about other sequential schedulers; on graphs that are not vertex-transitive, a new order in every epoch suffices (Cicerone et al. 2025d).

Rings, ASYNC, weak multiplicity detection (p. 149)

Q1. Even k: which symmetric starts whose axis passes through a node can be gathered? Left open by (Klasing et al. 2008); four robots on rings of seven and nine nodes cannot gather from some of them (D'Emidio et al. 2018), and the proofs do not extend to other ring sizes. Later ring papers listed in (Cicerone et al. 2025d) were not read here.

Complete graphs, FSYNC, strong multiplicity detection (p. 151)

Q1. Does every algorithm need $\Omega(\log k)$ epochs on complete graphs for infinitely many k, without assuming infinitely many Fibonacci primes? The lower bound of (Cicerone et al. 2021b) uses $k = F_\ell$ robots and $F_{\ell-1}$ empty nodes, solvable only when F_ℓ is prime.

Graphs that are not vertex-transitive, round robin (p. 152)

Q1. Can robots gather in $O(d)$ epochs on every graph that is not vertex-transitive and has no terminal orbit? Asked in (Cicerone et al. 2025d).

Finite grids, ASYNC, no multiplicity detection (p. 154)

Q1. How many epochs does gathering on finite grids without multiplicity detection take?

Broadcast by agents in dynamic graphs

Dynamic graph, any topology (p. 326)

Q1. Which graph property determines the least number of agents that solve broadcast on a graph G? Edge density, edge connectivity and the number of redundant edges do not (Moses Jr. et al. 2025); is there an analogue of dismantlability, which characterises cop-win graphs? (Moses Jr. et al. 2026)

Q2. How many rounds do the agents need when they win, as a function of n and D? Only lower bounds of order D are known (Moses Jr. et al. 2026).

Q3. Can broadcast be solved with weaker agents than global visibility, for example agents that see only the source agent? (Das et al. 2026b)

Dynamic hypercubes (p. 329)

Q1. How many agents are needed on Q_j for $j \geq 4$: $n/2$, $n-3$, or in between? (Das et al. 2026b)

Dynamic grids (p. 330)

Q1. Close the gap between $(\sqrt{n}-1)^2+1$ and $n-3$ agents on the $\sqrt{n} \times \sqrt{n}$ grid (Das et al. 2026b).

Claude (Anthropic) was used in cataloguing the problems.

Search for a moving target

Line, two face-to-face robots, away, u known (p. 551)

Q1. Is $(3+u)^2/(1-u)^2$ optimal for two face-to-face robots for every $u \in (0,1)$, as it is for one robot?

Line, a sender and a receiver, away, u known (p. 552)

Q1. Is $(\sqrt{u^2-2u+2}+1)/(\sqrt{u^2-2u+2}-1)$ optimal for a sender and a receiver for $u > 0$?

One robot, line, target moving away, speed known (p. 558)

Q1. Tight bounds when the robot does not know whether the target moves toward or away from the origin (Coleman et al. 2022).

One robot, line, target moving away, d known (p. 559)

Q1. Determine the lower-order terms in the exponent: the bounds differ by a factor $(1-u)^{-o(1)}$ (Coleman et al. 2026a).

One robot, line, target moving away, nothing known (p. 560)

Q1. Tight bounds when neither d nor u is known (Coleman et al. 2026a); the upper bound grows with d, the lower bounds do not.

m rays, a sender and a receiver, away, u known (p. 563)

Q1. For even $m \geq 4$, can a sender and a receiver beat two face-to-face robots, as they do on the line?

Collision-free exploration

Two agents sensing presence, with coloured pebbles (p. 238)

Q1. Do two colours suffice on every graph (Das et al. 2025a)?

Q2. How fast can two agents explore with three colours (Das et al. 2025a)?

Two agents, anonymous graphs, vision of radius 2 (p. 240)

Q1. Can two agents explore without collisions as fast as one agent explores alone? Asked by (Bhagat and Pelc 2025); (Papin 2026) claims a constant factor.

Q2. For which number of agents below n is collision-free exploration of anonymous graphs feasible, and with which vision radius (Bhagat and Pelc 2025)?

Labelled graphs, no knowledge, any number of agents (p. 243)

Q1. Close $O(n^5 \log n)$ against $\Omega(n^2)$ for agents without a map on graphs with labelled nodes.

Deciding graph properties by mobile agents

Any number of agents (p. 371)

Q1. What is the least information about the number of agents that lets several agents reconstruct the whole configuration? Posed in (Fraigniaud and Pelc 2017).

Claude (Anthropic) was used in cataloguing the problems.

One agent (p. 372)

Q1. Give a combinatorial characterisation of the problems that a lone agent decides. Posed in (Fraigniaud and Pelc 2017).

Q2. Is it decidable whether a lone agent can decide a given problem: is there a Turing machine that, given a Turing machine deciding membership in Π, decides whether a lone agent can decide Π? Posed in (Fraigniaud and Pelc 2017).

One agent that knows n (p. 373)

Q1. Is it decidable whether a lone agent that knows n can decide a given problem? Posed in (Fraigniaud and Pelc 2017).

Any number of agents, with certificates (p. 374)

Q1. Do certificates that may depend on the agents' identifiers verify more problems? Raised in (Bampas and Ilcinkas 2018).

Q2. Which problems become verifiable under the promise that there are at most two agents? Raised in (Bampas and Ilcinkas 2018).

Delivery by energy-constrained agents

Arbitrary graphs (p. 335)

Q1. What is the complexity of delivering several messages at once, for instance on paths? Raised in (Bärtschi et al. 2020).

Paths (p. 336)

Q1. Is there a pseudo-polynomial algorithm for delivery on a path? Posed in (Chalopin et al. 2014).

Q2. Find good resource-augmented planners for delivery on a path. Posed in (Chalopin et al. 2014).

Paths, equal budgets (p. 337)

Q1. Is delivery with equal budgets between two nodes of a tree NP-hard or polynomial? Posed in (Anaya et al. 2016) and again in (Chalopin et al. 2013); (Chalopin et al. 2021) states that on trees delivery is known to be weakly NP-hard, without a reference, in a paper about equal budgets.

Arbitrary graphs, equal budgets (p. 338)

Q1. Is there a constant $c > 1$ such that the smallest equal budget cannot be approximated within c in polynomial time unless P = NP? (Chalopin et al. 2021) notes that no lower bound is known.

Planar graphs, equal budgets (p. 339)

Q1. Is there an inapproximability bound for the smallest equal budget, already on planar graphs? No lower bound above 1 is known for any graph class (Chalopin et al. 2021).

Claude (Anthropic) was used in cataloguing the problems.

Appendix B

Index of assumptions

Every assumption the atlas uses, alphabetically, with its formal definition. Each entry names its axis and the model or problem that defines it, and the page where the axis is introduced with its origin. Level 0 assumes nothing; higher levels are stronger assumptions.

2D
Space, level 1; Shape formation by hybrid programmable matter, p. 632
Tiles occupy nodes of the triangular lattice.

3D
Space, level 0; Shape formation by hybrid programmable matter, p. 632
Tiles occupy nodes of the face-centred cubic lattice; planar structures are a special case.

adversarial activation
Activation, level 0; Scattering of particles, p. 624
The scheduler of the setting decides, fairly, when each particle is activated; an isolated particle may be activated, and a particle with a neighbour may wait any finite time.

adversarial ports
Preprocessing, level 0; Graph exploration by a single agent, p. 89
Port numbers (and any node labels) are chosen by an adversary.

adversarial wake-up
Start, level 0; Collision-free exploration, p. 236
Each agent starts in a round chosen by the adversary; until then it stays at its start node, where no other agent may enter.

advice
External help, level 1; Treasure hunt, p. 110
An oracle that knows the whole instance gives the agent a binary string before it starts; its length (in bits) is the measure.

all agents
Goal, level 0; Uniform deployment on graphs, p. 313
Every agent ends evenly spread and stops.

all but one
Goal, level 1; Uniform deployment on graphs, p. 313
All agents but one end evenly spread and stop; the remaining agent may stop on any node.

all robots
Goal, level 0; Evacuation, p. 510
The task ends when the last robot reaches the exit.

anonymous
Identifiers, level 0; Mobile agents on graphs, p. 11
All agents run the same deterministic algorithm on the same input; no agent has a label. Formally the label function is constant: $\mathrm{id}(a) = \mathrm{id}(b)$ for all agents a, b.

anonymous
Identifiers, level 0; Mobile robots in the plane, p. 26
Robots are indistinguishable and run the same algorithm.

anonymous nodes
Node identifiers, level 0; Mobile agents on graphs, p. 15

Nodes have no identifiers; an agent at v sees only $\deg(v)$ and the port numbers $1, \dots, \deg(v)$, which are assigned locally and independently at each endpoint of an edge.

any budgets
Budgets, level 0; Delivery by energy-constrained agents, p. 334
Each agent i has its own budget B_i.

any delay
Start, level 0; Rendezvous, p. 121
An adversary wakes the agents at times t_1 and t_2; the delay $\tau = |t_1 - t_2|$ is unknown.

any domain
Domain, level 0; Evacuation, p. 509
A bounded convex domain; the robots start together at a given point of it, in its interior or on its boundary.

any dynamics
Graph dynamics, level 0; Mobile agents on graphs, p. 10
A **temporal graph** is a sequence $\mathcal{G} = (G_t)_{t \geq 0}$ with $G_t = (V, E_t)$ on a fixed node set. It is *temporally connected* if for all $u, v \in V$ and every time t there is a journey (a path whose edges appear at non-decreasing times $\geq t$) from u to v. No G_t need be connected.

any faults
Faults, level 0; Mobile agents on graphs, p. 12
Up to f agents behave arbitrarily (adversarially) and may **forge** identifiers, i.e. report any label in $\mathcal{L}$.

any graph
Excluded minors, level 0; Online exploration of weighted graphs, p. 226
G is any connected graph.

any graph
Graph class, level 0; Delivery by energy-constrained agents, p. 334
G is any connected undirected graph with positive edge weights.

any graph
Network, level 0; Exploration by luminous robots, p. 181
A finite, connected, anonymous graph $G = (V, E)$ with n nodes and no port numbers visible to the robots.

any graph
Network, level 0; Exploration of recolourable graphs, p. 212
A finite, simple, connected, undirected, anonymous graph $G = (V, E)$ with n nodes.

any graph
Network, level 0; Gathering by oblivious robots on graphs, p. 145
Any connected anonymous graph, finite unless stated; robots see it whole but share no orientation.

any k
Number of agents, level 0; Mobile agents on graphs, p. 11
The number of agents $k \geq 1$ is arbitrary; $k > n$ is allowed.

any k
Number of agents, level 0; Collision-free exploration, p. 235
Any number k of agents, $2 \leq k \leq n$, which they do not know.

any k
Number of agents, level 0; Deciding graph properties by mobile agents, p. 369
Any number $k \geq 1$ of agents; no agent knows k, so an agent cannot tell whether it is alone.

any k
Number of robots, level 0; Linear search (cow path), p. 499
$k \geq 1$ robots.

any k
Number of robots, level 0; Evacuation, p. 510
$k \geq 1$ robots.

Claude (Anthropic) was used in cataloguing the problems.

any k
Number of robots, level 0; Search with segment visibility, p. 576
$k \geq 2$ robots; bounds are stated as functions of k.

any k
Number of robots, level 0; Search for a moving target, p. 547
$k \geq 1$ robots.

any k
Team size, level 0; Collaborative exploration, p. 248
Any number $k \geq 2$ of agents.

any n
Number of robots, level 0; Gathering, p. 434
Any number $n \geq 2$ of robots.

any order
Activation order, level 0; Gathering by oblivious robots on graphs, p. 144
The adversary chooses which robots to activate, subject only to the scheduler's fairness.

any start
Initial configuration, level 0; Mobile agents on graphs, p. 16
Agents start at arbitrary nodes; $\ell \leq k$ nodes are initially occupied, possibly with several agents each.

any start
Initial configurations, level 0; Gathering, p. 435
The algorithm must gather from every initial configuration from which gathering is possible at all in the setting; configurations proved unsolvable there (for example bivalent ones, or several multiplicities with weak detection and crashes) are excluded by the problem, not by the algorithm.

any topology
Topology, level 0; Mobile agents on graphs, p. 15
G is any connected graph in the class allowed by the other axes.

arbitrary colours
Initial colouring, level 0; Exploration of recolourable graphs, p. 211
The initial colouring $\psi_0 : V \to \{1, \dots, c\}$ is chosen by the adversary; an algorithm must work from every ψ_0 (as with a rotor-router whose pointers are set arbitrarily).

ASYNC
Scheduler, level 0; Exploration by oblivious robots, p. 166
ASYNC: each robot's Look, Compute and Move phases take finite but unbounded, adversarially chosen time; a robot may Look while others are between phases, so it may move on an outdated snapshot. A robot is always seen on a node (moves are instantaneous). The scheduler is fair: every robot is activated infinitely often. Called *CORDA* (with instantaneous moves) in (Lamani et al. 2010) and (Blin et al. 2010).

ASYNC
Scheduler, level 0; Exploration by luminous robots, p. 178
ASYNC: each robot's Look, Compute and Move phases take finite but unbounded, adversarially chosen time; a robot may Look while others are between phases, so it may move on an outdated snapshot. A robot is always seen on a node (an edge traversal is instantaneous), and its new colour is visible from the end of its Compute phase. The scheduler is fair: every robot is activated infinitely often. Called *CORDA* in (Datta et al. 2013).

ASYNC
Scheduler, level 0; Gathering by oblivious robots on graphs, p. 143
ASYNC: each robot's Look, Compute and Move phases take finite but unbounded, adversarially chosen time; a robot may Look while others are between phases, so it may move on an outdated snapshot. A robot is always seen on a node (moves are instantaneous). The scheduler is fair: every robot is activated infinitely often.

ASYNC
Scheduler, level 0; Mobile robots in the plane, p. 20
ASYNC: each robot's Look, Compute and Move phases take finite but unbounded, adversarially chosen time; a robot may look while others move, and compute on an outdated snapshot. Time is counted in epochs: an

Claude (Anthropic) was used in cataloguing the problems.

epoch is an interval of time in which every robot completes at least one full Look–Compute–Move cycle; it need not be the shortest such interval. The execution is divided into consecutive epochs, and a bound of X epochs holds for every such division. In FSYNC and SSYNC time is counted in rounds. Called *Corda* by Prencipe (Prencipe 2007). Intermediate variants restrict interleavings: *LC-atomic* (no Look by another robot between a robot's Look and its Compute), *Move-atomic* (no Look during another robot's Move) and *CM-atomic* (Flocchini et al. 2023).

asynchronous

Timing, level 0; Mobile agents on graphs, p. 14

Each agent's Communicate–Compute–Move steps take finite but unbounded, adversarially chosen time. Time is measured in epochs. An **epoch** is an interval of time in which every agent completes at least one full Communicate–Compute–Move step; it need not be the shortest such interval. Time is counted by dividing the execution into consecutive epochs, and a bound of X epochs holds for every such division.

at most crash faults

Faults, level 2; Mobile agents on graphs, p. 12

Up to f agents may stop forever at an adversarially chosen time; a crashed agent stays at its node (or vanishes, as stated per result) and never acts again.

at most weak Byzantine faults

Faults, level 1; Mobile agents on graphs, p. 12

Up to f agents behave arbitrarily but always report their own true identifier.

away

Direction of the target, level 1; Search for a moving target, p. 547

The target moves away from the origin at constant speed $u < 1$, and the robots know it.

bipartite

Topology, level 1; Mobile agents on graphs, p. 15

$V = V_1 \cup V_2$ with $V_1 \cap V_2 = \emptyset$ and every edge joining V_1 to V_2; trees are a special case.

both goals

Exploration goal, level 0; Exploration by oblivious robots, p. 167

Solve both terminating and exclusive perpetual exploration with the stated resources (possibly with two algorithms). A lower bound for either goal holds here, and a result here gives both. No variant is catalogued here; the value exists because the two goals do not imply each other.

both goals

Exploration goal, level 0; Exploration by luminous robots, p. 180

Solve both terminating and perpetual exploration with the stated resources (possibly with two algorithms). A lower bound for either goal holds here, and a result here gives both. No variant is catalogued here; the value exists because the two goals do not imply each other.

both known

Knowledge of the target, level 2; Search for a moving target, p. 547

The robots know d and u; only the side (ray) of the target is unknown.

bound on n

Knowledge, level 1; Map construction, p. 159

An upper bound $N \geq n$ is known.

bound on n

Knowledge of the graph, level 1; Collision-free exploration, p. 235

The agents know an upper bound on n (Das et al. 2025a give them n itself).

Byzantine

Faults, level 0; Mobile robots in the plane, p. 25

Up to f robots move arbitrarily (adversarially); with lights, they may also set arbitrary colours.

Byzantine

Faults, level 0; Search in geometric domains, p. 40

Up to f robots may lie: stay silent when they find the target, or claim to have found it where it is not.

Claude (Anthropic) was used in cataloguing the problems.

cactus
Network, level 1; Exploration of recolourable graphs, p. 212
G is a cactus: every edge lies on at most one cycle (equivalently, every block is a cycle or a single edge).

chirality
Agreement on coordinates, level 1; Mobile robots in the plane, p. 22
All local coordinate systems have the same handedness (clockwise orientation): no reflection. Also called *handedness*. Prencipe uses *chirality* for agreement on North and East, i.e. a compass (Prencipe 2007).

chirality
Chirality, level 1; Exploration by luminous robots, p. 180
All robots agree on the clockwise direction: a snapshot is known up to rotation only (no reflection), so a robot on an axis of symmetry of its view can still tell its two sides apart. Robots still share no North.

chirality
Orientation, level 1; Programmable matter, p. 44
All particles agree on clockwise order, not on a common direction.

circle
Fence, level 1; Patrolling, p. 569
A closed fence: a circle of length ℓ.

circuits
Communication, level 1; Programmable matter, p. 45
Each particle has a constant number of **pins** on every incident edge, shared with the neighbour across that edge, and in every round it links its pins into groups. A **circuit** is a connected component of the resulting graph on pins. A particle may send a **beep** on any of its circuits; every particle on the circuit receives it in the next round, without learning who sent it or how many did (Feldmann et al. 2022). Local communication remains available.

colour once
Recolouring, level 1; Exploration of recolourable graphs, p. 210
Every node starts with the initial colour c_0. The agent at v may change $\psi(v)$ only while $\psi(v) = c_0$; once v has a colour other than c_0 it keeps it forever. This is the main model of (Böckenhauer et al. 2023).

common compass
Orientation, level 2; Programmable matter, p. 44
All particles agree on all six directions.

common direction
Orientation, level 1; Programmable matter, p. 44
All particles agree on one lattice direction, not on handedness. Chalopin, Das and Kokkou call this a *common direction* (Chalopin et al. 2026b), but the same words mean agreement on all six directions in other papers (Chalopin et al. 2026a; Navarra and Piselli 2023b).

compass
Agreement on coordinates, level 2; Mobile robots in the plane, p. 22
All robots agree on the directions and orientations of both axes (a common compass); they may still disagree on origin and unit. Called *full compass* by Efrima and Peleg, who also use *half compass* (both axis directions, the positive sense of one axis, no chirality), *direction-only* and *axes-only* (Efrima and Peleg 2009). Agreement on one axis together with common chirality is equivalent: the orientation of the second axis then follows from handedness.

complete graph
Network, level 1; Gathering by oblivious robots on graphs, p. 145
K_n: every two nodes are neighbours.

complete graph
Topology, level 1; Mobile agents on graphs, p. 15
$G = K_n$: every pair of nodes is adjacent.

connected every round
Graph dynamics, level 1; Mobile agents on graphs, p. 10

Claude (Anthropic) was used in cataloguing the problems.

$\mathcal{G} = (G_t)_{t\geq 0}$ is **1-interval connected** if every snapshot $G_t = (V, E_t)$ is connected. More generally it is T-interval connected if for every t the graph $\left(V, \bigcap_{i=t}^{t+T-1} E_i\right)$ is connected.

connected team

Connectivity of the team, level 0; Collaborative exploration, p. 248

In every round the nodes occupied by the agents induce a connected subgraph. An algorithm is then compared with the best traversal of the known graph by a team that obeys the same rule (Mutzari et al. 2026).

constant memory

Memory, level 1; Programmable matter, p. 46

Each particle has $O(1)$ bits of persistent memory and reads or writes neighbours' memory.

crash faults

Faults, level 1; Mobile robots in the plane, p. 25

Up to f robots stop forever at adversarial times and remain visible at their last position.

crash faults

Faults, level 1; Search in geometric domains, p. 40

Up to f robots may fail to detect or to report the target when they pass over it.

d known

Knowledge of the distance, level 1; Search with segment visibility, p. 576

The robots know d (not the direction of the target); the trajectories may depend on d, and the ratio is the supremum over targets with $\|x\| = d$ of $T(x)/d$.

d unknown

Knowledge of the distance, level 0; Search with segment visibility, p. 576

The trajectories do not depend on d; the ratio is the supremum over all targets $x \neq O$ of $T(x)/\|x\|$.

decide

Certificates, level 0; Deciding graph properties by mobile agents, p. 370

The agents receive only their identifiers and inputs.

designer labels

Preprocessing, level 1; Graph exploration by a single agent, p. 89

Before the agent starts, the designer may write a label of b bits (a constant) at every node, which the agent reads on arrival.

designer ports

Preprocessing, level 1; Graph exploration by a single agent, p. 89

Before the agent starts, the designer may choose the port numbering $\mathrm{port}_v : \{\text{edges at } v\} \to \{1, \dots, \deg(v)\}$ at every node. Ilcinkas calls a designer's port numbering a *local orientation* (Ilcinkas 2008).

designer's colours

Initial colouring, level 2; Exploration of recolourable graphs, p. 211

Before the agent starts, a designer who knows G (and the start node) chooses ψ_0. Used with fixed colours this is label-guided exploration (Cohen et al. 2008).

deterministic

Randomness, level 0; Mobile agents on graphs, p. 12

The next state and move of an agent are a function of its current state and what it observes.

deterministic

Randomness, level 0; Mobile robots in the plane, p. 25

The destination computed in a cycle is a function of the snapshot (and the light, if any).

deterministic

Randomness, level 0; Search in geometric domains, p. 41

Trajectories are fixed functions of the information received.

deterministic

Randomness, level 0; Programmable matter, p. 46

No random choices.

directed

Edge direction, level 0; Mobile agents on graphs, p. 11

$G = (V, A)$ with $A \subseteq V \times V$; an agent at u may move along $(u, v) \in A$ only, never back along it unless $(v, u) \in A$. G is strongly connected.

direction unknown

Direction of the target, level 0; Search for a moving target, p. 547

The target moves toward or away from the origin, and the robots do not know which.

disk

Domain, level 1; Evacuation, p. 509

The unit disk; robots start at its centre; the exit is a point of the boundary circle.

disoriented

Agreement on coordinates, level 0; Mobile robots in the plane, p. 22

Each robot r observes in its own coordinate system, obtained from the global one by an unknown similarity (translation, rotation, uniform scaling, and possibly reflection); systems may differ between robots and, in some models, between cycles. Papers that call robots *disoriented* do not always mean this: Défago and Souissi use the word while assuming chirality (Défago and Souissi 2008). Some papers also let the adversary change a robot's rotation, reflection and unit at every Look (*variable disorientation*) (Flocchini et al. 2023; Wada et al. 2026).

dispersed start

Initial configuration, level 1; Mobile agents on graphs, p. 16

Every initially occupied node holds exactly one agent ($\ell = k$). Also called a *scattered* start, notably in black hole search.

distance known

Knowledge of the target, level 1; Search for a moving target, p. 547

The robots know d but not u.

distinct IDs (eponymous)

Identifiers, level 2; Mobile agents on graphs, p. 11

Distinct identifiers (eponymous agents): the label function $\mathrm{id} : \{1, \dots, k\} \to \{1, \dots, L\}$ is injective; usually $L = k^{O(1)}$, so a label takes $O(\log k)$ bits.

distinct IDs (eponymous)

Identifiers, level 2; Mobile robots in the plane, p. 26

Distinct identifiers (eponymous robots): labels are pairwise distinct ($\ell = n$); whether others can see a robot's label is stated per result.

distinct speeds

Speeds, level 0; Search in geometric domains, p. 41

Each robot i has its own maximum speed v_i, known to all.

***e* exits**

Exits, level 0; Chauffeuring a crashed robot, p. 542

$e \geq 1$ exits at unknown boundary points.

equal budgets

Budgets, level 1; Delivery by energy-constrained agents, p. 334

All agents have the same budget B.

equal speeds

Speeds, level 1; Search in geometric domains, p. 41

All robots move at most at unit speed.

evacuation

Goal, level 0; Linear search (cow path), p. 500

The task ends when every robot has reached the target.

event-driven

Activation, level 1; Scattering of particles, p. 624

A contracted particle is activated as soon as it has, or gains, a neighbour, and a contracted particle with no

Claude (Anthropic) was used in cataloguing the problems.

neighbour is not activated. The adversary still chooses the duration of each activation, the time an expansion or a contraction takes, and the activation of expanded particles; operations stay non-atomic.

eventually emerging
Black hole appearance, level 0; Black hole search, p. 294
There is a time $t \geq 0$ and a node u such that from time t on u destroys every agent on it or entering it, and stays a black hole. If $t = 0$ then $u \neq s$; the home base may become the black hole after the first round (after one epoch for asynchronous agents). Time is counted from t (Bonnet et al. 2026b).

every agent
What must be visited, level 0; Collision-free exploration, p. 235
Every agent must visit every node and stop (in Czyzowicz et al. 2017a also return to its start node).

exclusive perpetual
Exploration goal, level 1; Exploration by oblivious robots, p. 167
Every robot visits every node infinitely often, no two robots are ever on the same node, and no two robots ever cross the same edge in opposite directions.

external light (FCOM)
Lights (persistent memory), level 1; Mobile robots in the plane, p. 21
FCOM: each robot has a light with c colours that only the others can see (constant communication, no persistent memory of its own). Also called an *external* light.

face-to-face
Communication, level 0; Search in geometric domains, p. 40
Face-to-face: two robots exchange information only when they are at the same point at the same time.

fat
Body, level 0; Mobile robots in the plane, p. 19
Each robot is a closed disc of radius 1 (unit disc). Discs may not overlap, so robots cannot share a point and must avoid collisions; a disc hides what lies behind it from other robots. Gathering then asks for a configuration whose union of discs is connected and in which every robot sees all others (Czyzowicz et al. 2009b).

FCOM
Lights (memory and messages), level 1; Exploration by luminous robots, p. 179
FCOM (*finite-communication*): each robot has a light with c colours that the other robots within distance φ can see but the robot itself cannot; the robot stays oblivious, but every robot that looks reads its last colour: $\log_2 c$ bits broadcast to the neighbourhood, no memory (Flocchini et al. 2016a).

finite automaton
Agent memory, level 1; Exploration of recolourable graphs, p. 210
The agent is a deterministic automaton with K states, K independent of the graph; its action and next state depend on its state and its observation.

finite graph
Graph size, level 2; Treasure hunt, p. 110
G has finitely many nodes.

finite graph
Graph size, level 2; Rendezvous, p. 121
G has finitely many nodes.

finite grid
Network, level 1; Gathering by oblivious robots on graphs, p. 145
An anonymous $a \times b$ grid without orientation: nodes are the points (i, j) with $1 \leq i \leq a$ and $1 \leq j \leq b$, neighbours at distance one.

fixed colours
Recolouring, level 0; Exploration of recolourable graphs, p. 210
The colouring $\psi : V \to \{1, \ldots, c\}$ never changes; the agent reads $\psi(v)$ (and, with neighbour view, its neighbours' colours) but cannot write.

Claude (Anthropic) was used in cataloguing the problems.

free team

Connectivity of the team, level 1; Collaborative exploration, p. 248

Agents may be anywhere in the graph.

FSTA

Lights (memory and messages), level 1; Exploration by luminous robots, p. 179

FSTA (*finite-state*): each robot has a light with c colours that only it can see. The colour is set at the end of Compute and persists until the robot changes it: $\log_2 c$ bits of persistent memory, no communication (Flocchini et al. 2016a).

FSYNC

Scheduler, level 2; Exploration by oblivious robots, p. 166

FSYNC: in every round every robot executes one atomic Look–Compute–Move cycle.

FSYNC

Scheduler, level 2; Exploration by luminous robots, p. 178

FSYNC: in every round every robot executes one atomic Look–Compute–Move cycle.

FSYNC

Scheduler, level 2; Gathering by oblivious robots on graphs, p. 143

FSYNC: in every round every robot executes one atomic Look–Compute–Move cycle.

FSYNC

Scheduler, level 2; Mobile robots in the plane, p. 20

FSYNC: every robot executes one atomic cycle in every round.

global communication

Communication, level 3; Mobile agents on graphs, p. 13

At every round every agent can read the memory of every other agent, wherever it is.

graph

Fence, level 0; Patrolling, p. 569

Every point of every edge of a graph G must be visited; robots move along edges.

grid

Network, level 1; Exploration by luminous robots, p. 181

An $a \times b$ grid ($n = ab$ nodes; nodes (i, j), edges between nodes at Manhattan distance one); anonymous, with a and b unknown. A robot sees the border of the grid when it is within range, since nodes beyond the border are missing from its snapshot.

grid

Topology, level 1; Mobile agents on graphs, p. 15

G is a $\sqrt{n} \times \sqrt{n}$ grid graph; whether ports are globally consistent (oriented) is stated per result.

h-hop communication

Communication, level 2; Mobile agents on graphs, p. 13

For a fixed $h \geq 1$, stated with each result: at every round an agent at v can read the memory of every agent at a node u with $\mathrm{dist}(v, u) \leq h$ (in the current snapshot G_t for a dynamic graph), in addition to the co-located agents. $h = 1$ is **1-hop communication** (agents at neighbouring nodes communicate); local communication is the case $h = 0$, and global communication the case $h \geq D$. Reading another agent's memory does not tell an agent which port leads to it.

H-minor-free

Excluded minors, level 1; Online exploration of weighted graphs, p. 226

For a fixed graph H, no graph obtained from G by deleting nodes and edges and contracting edges is isomorphic to H. Graphs of bounded genus and of bounded treewidth are such classes; bounds may depend on H.

holes allowed

Initial shape, level 0; Programmable matter, p. 47

The initial set of occupied nodes induces a connected subgraph; it may enclose empty regions.

homonymous

Identifiers, level 1; Mobile agents on graphs, p. 11

Each agent a has a label $\mathrm{id}(a) \in \mathscr{L}$ known to itself (and revealed on meeting), but the labelling need not be

Claude (Anthropic) was used in cataloguing the problems.

injective: $1 \le |\{\mathrm{id}(a)\}| \le k$. With $|\mathscr{L}|$ distinct labels, $|\mathscr{L}| = 1$ is the anonymous case and $|\mathscr{L}| = k$ the eponymous one (Delporte-Gallet et al. 2013).

homonymous
Identifiers, level 1; Mobile robots in the plane, p. 26
Robots carry labels from a set of size ℓ, $1 \le \ell \le n$, which may repeat; equivalently, robots may run one of ℓ algorithms. The minimum ℓ that solves a problem is its *minimum algorithm size* (Asahiro and Yamashita 2025).

hypercube
Network, level 1; Gathering by oblivious robots on graphs, p. 145
The hypercube of dimension $\log n$: nodes are the n binary strings of length $\log n$, neighbours when they differ in one bit. Its diameter is $D = \log n$.

hypercube
Topology, level 1; Mobile agents on graphs, p. 15
$G = Q_j$: nodes are the $n = 2^j$ binary strings of length j, adjacent when they differ in one bit.

infinite degrees allowed
Graph size, level 0; Treasure hunt, p. 110
G is finite or countably infinite; a node may have countably infinite degree (ports $1, 2, 3, \ldots$).

infinite degrees allowed
Graph size, level 0; Rendezvous, p. 121
G is finite or countably infinite; degrees may be infinite.

infinite grid
Network, level 1; Gathering by oblivious robots on graphs, p. 145
The infinite square grid: nodes are the points of $\mathbb{Z}^2$, neighbours at Manhattan distance one; finitely many robots.

internal light (FSTA)
Lights (persistent memory), level 1; Mobile robots in the plane, p. 21
FSTA: each robot has a light with c colours that only it can see; it persists between cycles (constant persistent memory, no communication). Also called an *internal* light.

$k = n$
Team size, level 1; Collaborative exploration, p. 248
Exactly $k = n$ agents.

$k \ge Dn^{1+\varepsilon}$
Team size, level 1; Collaborative exploration, p. 248
$k \ge D\,n^{1+\varepsilon}$ agents for a constant $\varepsilon > 0$ (written $k = Dn^{\gamma}$, $\gamma > 1$).

$k \le n$
Number of agents, level 1; Mobile agents on graphs, p. 11
The number of agents satisfies $1 \le k \le n$.

known graph
Knowledge, level 1; Network decontamination, p. 319
The topology is known to the agents (or to a coordinator).

known $k \ge 2$
Number of agents, level 1; Deciding graph properties by mobile agents, p. 369
$k \ge 2$ agents, and every agent knows k.

knows a bound on n
Knowledge, level 1; Graph exploration by a single agent, p. 88
The agent is given an integer $N \ge n$ (an upper bound on the number of nodes) as input.

knows n
Knowledge of the graph, level 1; Deciding graph properties by mobile agents, p. 370
The agents know n (an oracle for the number of nodes in Fraigniaud and Pelc 2017).

Claude (Anthropic) was used in cataloguing the problems.

knows n, Δ
Knowledge, level 1; Leader election by agents, p. 343
Every agent knows n and Δ (upper bounds suffice where stated).

knows n, Δ
Knowledge, level 1; Maximal independent set by agents, p. 348
Every agent knows n and Δ (upper bounds suffice where stated).

knows n, Δ
Knowledge, level 1; Minimal dominating set by agents, p. 352
Every agent knows n and Δ (upper bounds suffice where stated).

knows n, Δ
Knowledge, level 1; Triangle and butterfly counting by agents, p. 356
Every agent knows n and Δ (upper bounds suffice where stated).

knows n, Δ
Knowledge, level 1; Minimum spanning tree by agents, p. 361
Every agent knows n and Δ (upper bounds suffice where stated).

knows n, Δ
Knowledge, level 1; BFS tree by agents, p. 365
Every agent knows n and Δ (upper bounds suffice where stated).

knows the map
Knowledge, level 2; Graph exploration by a single agent, p. 88
The agent is given a graph isomorphic to G, with port numbers, and (where stated) its own position on it; for a dynamic graph, the whole sequence (G_t).

labelled nodes
Node identifiers, level 1; Mobile agents on graphs, p. 15
Each node v carries a distinct identifier that agents at v can read; hence an agent can recognise a node it has visited.

lights (LUMI)
Lights (persistent memory), level 2; Mobile robots in the plane, p. 21
LUMI: each robot has a light with c colours, set at the end of Compute, visible to all robots including itself; it is both memory and communication. Also called *luminous* robots, or a *full* light.

limited visibility
Visibility, level 0; Mobile robots in the plane, p. 24
A robot sees exactly the robots within Euclidean distance V of its position, for a fixed radius $V > 0$.

line
Domain, level 1; Linear search (cow path), p. 499
The real line, robots start at 0; the target is at an unknown x with $|x| \geq 1$.

line
Domain, level 1; Evacuation, p. 509
The robots start at the origin of the real line; the exit is at an unknown point at distance at least 1; the measure is the competitive ratio, the evacuation time divided by that distance.

line
Domain, level 1; Search for a moving target, p. 546
The real line; the target moves along one side of the origin.

local communication
Communication, level 1; Mobile agents on graphs, p. 13
Agents at the same node at the same time can read each other's entire memory (face-to-face communication). Agents crossing on an edge do not meet.

local communication
Communication, level 0; Programmable matter, p. 45
A particle exchanges constant-size information only with the particles on adjacent nodes, by messages or by reading and writing their memory, so information travels at most one particle per activation.

Claude (Anthropic) was used in cataloguing the problems.

locally finite
Graph size, level 1; Treasure hunt, p. 110
G is finite or countably infinite, and every node has finite degree.

locally finite
Graph size, level 1; Rendezvous, p. 121
G is finite or countably infinite with finite degrees.

luminous
Lights (memory and messages), level 2; Exploration by luminous robots, p. 179
LUMI (*luminous* robots): each robot has a light with c colours, set at the end of Compute, persistent across cycles, and visible to the robot itself and to every robot within distance φ (Das et al. 2016). It is both memory and message. A Look returns, for every node within distance φ, the multiset of colours of the robots on it.

m rays
Domain, level 0; Linear search (cow path), p. 499
$m \geq 2$ half-lines sharing their endpoint, the origin, where the robots start.

m rays
Domain, level 0; Search for a moving target, p. 546
$m \geq 2$ half-lines sharing their endpoint, the origin; the target moves along one of them.

map
Knowledge of the graph, level 2; Black hole search, p. 293
Agents have a map of G and know their starting position on it.

map
Knowledge of the graph, level 2; Collision-free exploration, p. 235
Every agent knows G and its own start node in it, but not the number or the start nodes of the other agents.

map
Knowledge of the graph, level 2; Deciding graph properties by mobile agents, p. 370
The agents have a copy of G with its ports, without their start nodes marked on it.

markers
Markers, level 1; Map construction, p. 159
The agent carries markers it can drop and pick up.

may disconnect
Connectivity during execution, level 1; Programmable matter, p. 47
Connectivity is required only initially and at the end.

may move
Movement, level 1; Leader election in programmable matter, p. 586
Particles may expand and contract during the election.

$n = 2$
Number of robots, level 1; Gathering, p. 434
Exactly two robots (rendezvous).

$n \geq 3$
Number of robots, level 1; Gathering, p. 434
$n \geq 3$ robots.

n known
Knowledge, level 1; Exploration of recolourable graphs, p. 211
The algorithm may depend on n: it must explore every graph with exactly n nodes (a *non-uniform* algorithm).

neighbour colours
View, level 1; Exploration of recolourable graphs, p. 209
At node v the agent observes $\psi(v)$ and the multiset $\{\!\{\psi(u) : u \in N(v)\}\!\}$ of its neighbours' colours (hence also $\deg(v)$), without knowing which edge leads to which colour unless ports say so.

no chirality
Chirality, level 0; Exploration by luminous robots, p. 180

Claude (Anthropic) was used in cataloguing the problems.

Robots share no orientation: on a ring a robot cannot tell its two directions apart; on a grid or torus its snapshot is known only up to rotation and reflection, and the adversary may choose a different transformation in every cycle (*self-inconsistent* robots, as in Darwich et al. 2023 and Bramas et al. 2026a).

no communication
Communication, level 0; Mobile agents on graphs, p. 13
Agents exchange no messages; an agent at v observes only the number of agents at v (and, where stated, whether that number changed).

no faults
Faults, level 3; Mobile agents on graphs, p. 12
All agents follow the algorithm forever.

no faults
Faults, level 2; Mobile robots in the plane, p. 25
All robots follow the algorithm forever.

no faults
Faults, level 2; Search in geometric domains, p. 40
Every robot detects and reports the target.

no help
External help, level 0; Treasure hunt, p. 110
No information beyond the agent's own observations.

no help
Help from an oracle, level 0; Collision-free exploration, p. 236
Nothing is placed on the nodes.

no holes
Initial shape, level 1; Programmable matter, p. 47
The initial configuration is connected and encloses no empty node. Also called *contractible* (Briones et al. 2023) or *hole-free* (Gastineau et al. 2018).

no knowledge
Knowledge, level 0; Leader election by agents, p. 343
Agents know no graph parameter (n, m, Δ, D).

no knowledge
Knowledge, level 0; Maximal independent set by agents, p. 348
Agents know no graph parameter (n, m, Δ, D).

no knowledge
Knowledge, level 0; Minimal dominating set by agents, p. 352
Agents know no graph parameter (n, m, Δ, D).

no knowledge
Knowledge, level 0; Triangle and butterfly counting by agents, p. 356
Agents know no graph parameter (n, m, Δ, D).

no knowledge
Knowledge, level 0; Minimum spanning tree by agents, p. 361
Agents know no graph parameter (n, m, Δ, D).

no knowledge
Knowledge, level 0; BFS tree by agents, p. 365
Agents know no graph parameter (n, m, Δ, D).

no knowledge
Knowledge of the graph, level 0; Deciding graph properties by mobile agents, p. 370
The agents know nothing about G.

no markers
Markers, level 0; Graph exploration by a single agent, p. 89
The agent cannot mark nodes.

Claude (Anthropic) was used in cataloguing the problems.

no markers
Markers, level 0; Map construction, p. 159
No markers.

no marking
Marking nodes, level 0; Black hole search, p. 293
Agents cannot leave information at nodes.

no multiplicity detection
Multiplicity detection, level 0; Gathering by oblivious robots on graphs, p. 144
A snapshot shows which nodes are occupied, not how many robots each holds.

no multiplicity detection
Multiplicity detection, level 0; Mobile robots in the plane, p. 24
A snapshot is a set of points: a point occupied by several robots looks like a single robot.

no orientation
Orientation, level 0; Programmable matter, p. 44
Each particle labels its six incident directions in its own clockwise or counter-clockwise order, starting anywhere.

no ports
Port numbers, level 0; Exploration of recolourable graphs, p. 209
Edges carry no port numbers. To move, the agent at v names a colour x; the adversary moves it to any neighbour u of v with $\psi(u) = x$ (if the agent sees no neighbour colours, to any neighbour). The agent never learns by which edge it arrived.

no vision
Vision of the other agents, level 0; Collision-free exploration, p. 236
An agent never perceives another agent.

no whiteboards
Node memory, level 0; Mobile agents on graphs, p. 14
Nodes have no storage: no information can be left at a node except by an agent staying there.

no whiteboards
Whiteboards, level 0; Map construction, p. 159
No whiteboards.

non-rigid
Movement, level 0; Mobile robots in the plane, p. 23
In a Move phase the adversary may stop a robot before its destination, but only after it has travelled at least $\delta > 0$ (a constant unknown to the robots) or reached the destination.

not vertex-transitive
Network, level 1; Gathering by oblivious robots on graphs, p. 145
A finite connected graph whose nodes fall into at least two **orbits**, an orbit being a class of nodes that automorphisms of G map onto each other.

nothing known
Knowledge, level 0; Graph exploration by a single agent, p. 88
The agent knows nothing about G: not n, m, Δ or D.

nothing known
Knowledge, level 0; Map construction, p. 159
Nothing is known about G.

nothing known
Knowledge, level 0; Exploration of recolourable graphs, p. 211
The algorithm is the same for all graphs: it knows neither n nor Δ nor any other parameter (a *uniform* algorithm in Böckenhauer et al. 2023).

nothing known
Knowledge of the graph, level 0; Online exploration of weighted graphs, p. 226

Claude (Anthropic) was used in cataloguing the problems.

The agent knows nothing about G before it starts; the edges at a node and their weights are revealed when the agent first visits the node.

nothing known
Knowledge of the graph, level 0; Collision-free exploration, p. 235
The agents know nothing about G, not even n.

nothing known
Knowledge of the target, level 0; Search for a moving target, p. 547
The robots know neither the initial distance d nor the speed u.

oblivious
Agent memory, level 0; Exploration of recolourable graphs, p. 210
The agent has no internal state: its action (new colour of v, then a move, *stay* or *stop*) is a function of its current observation only. Called *zero-memory*, *memoryless* or a *1-state* robot.

oblivious
Lights (memory and messages), level 0; Exploration by luminous robots, p. 179
OBLOT: a robot's Compute phase depends only on the current snapshot; nothing persists across cycles and robots communicate only by moving. Equivalent to a light with $c = 1$ colour.

oblivious
Lights (persistent memory), level 0; Mobile robots in the plane, p. 21
OBLOT: a robot's computation depends only on the current snapshot; nothing persists across cycles.

oblivious
Memory, level 0; Programmable matter, p. 46
No persistent state and no messages; a particle decides from its view only.

$O(\log n)$ bits
Agent memory, level 2; Exploration of recolourable graphs, p. 210
The agent has $b = O(\log n)$ bits of memory, enough to hold a counter up to a polynomial in n. No variant is catalogued here yet.

one agent
Agents, level 0; Map construction, p. 158
A single agent.

one agent
Number of agents, level 1; Deciding graph properties by mobile agents, p. 369
Exactly one agent, $k = 1$: configurations with more agents are excluded.

one axis
Agreement on coordinates, level 1; Mobile robots in the plane, p. 22
All robots agree on the direction and orientation of one axis (e.g. North), but not necessarily on handedness: two robots may disagree on which side is East. This is how the papers use *one axis agreement* (e.g. robots "agree on only 'left' and 'right'" Bose et al. 2021a; robots "do not necessarily have the same handedness" Pagli et al. 2015). One axis and chirality are alternatives on this axis; together they amount to a common compass.

one exit
Exits, level 0; Evacuation, p. 511
One exit, at an unknown point of the boundary.

one exit
Exits, level 1; Chauffeuring a crashed robot, p. 542
Exactly one exit.

one pebble
Markers, level 1; Graph exploration by a single agent, p. 89
The agent carries one pebble that it can drop at its current node and later pick up; it sees whether a pebble is at its node.

one robot
Number of robots, level 1; Linear search (cow path), p. 499
A single robot.

Claude (Anthropic) was used in cataloguing the problems.

one robot
Number of robots, level 1; Search for a moving target, p. 547
A single robot.

one-way
Direction of movement, level 0; Patrolling, p. 569
On a closed fence every robot moves only counterclockwise, at any speed between 0 and its maximum speed v_i; it never reverses. Czyzowicz et al. require strictly positive speeds, while Dumitrescu, Ghosh and Tóth and Kawamura and Soejima allow robots to stop; the counterexample of Dumitrescu et al. holds under both conventions (Remark after Theorem 1 of Dumitrescu et al. 2014). Schedules are a subset of the two-way schedules, so this is the harder level for the robots.

opaque
Opacity, level 0; Mobile robots in the plane, p. 25
Robots are points that obstruct sight: r sees s iff no third robot lies on the open segment $\overline{rs}$.

oracle pebbles
External help, level 1; Treasure hunt, p. 110
An oracle that knows the instance places pebbles on nodes (at most one per node) before the agent starts; their number is the measure.

oriented
Grid orientation, level 1; Dispersion, p. 261
The graph is a grid with nodes (x, y), and every node labels its ports North, East, South and West consistently: the port North of (x, y) leads to $(x, y+1)$, East to $(x+1, y)$, South to $(x, y-1)$ and West to $(x-1, y)$, whenever that neighbour exists. The same direction carries the same label at every node, so an agent can keep moving along a row or column; a boundary node is recognised by its missing ports.

own node
View, level 0; Exploration of recolourable graphs, p. 209
At node v the agent observes $\psi(v)$, $\deg(v)$ and nothing about its neighbours (and, where ports exist, what the port axis grants).

path
Graph class, level 2; Delivery by energy-constrained agents, p. 334
G is a path, equivalently a segment of the line with the agents, v_s and v_t on it.

pebbles
Help from an oracle, level 1; Collision-free exploration, p. 236
Before the start an oracle that knows G, the start nodes and the wake-up rounds places at most one pebble on each node; pebbles have colours, and an agent sees the pebble of the node it is at (Das et al. 2025a).

pebbles
Markers, level 2; Graph exploration by a single agent, p. 89
The agent carries p pebbles (identical or distinguishable, as stated); p is a resource reported with the memory.

perpetual
Exploration goal, level 1; Exploration by luminous robots, p. 180
Every node is visited by at least one robot infinitely often; robots never stop.

perpetual
Termination requirement, level 2; Graph exploration by a single agent, p. 90
Every node is visited at least once (and, for periodic exploration, infinitely often); the agent never needs to halt.

perpetual
Termination requirement, level 2; Exploration of recolourable graphs, p. 211
Every node is visited at least once (for the rotor-router: infinitely often); the agent never needs to halt.

planar
Excluded minors, level 2; Online exploration of weighted graphs, p. 226
G can be drawn in the plane without crossing edges; equivalently it has neither K_5 nor $K_{3,3}$ as a minor.

Claude (Anthropic) was used in cataloguing the problems.

planar
Graph class, level 1; Delivery by energy-constrained agents, p. 334
G is planar.

point
Body, level 1; Mobile robots in the plane, p. 19
Each robot is a point of the plane: several robots may share a point (multiplicity), and a robot hides only what lies exactly behind it on a line of sight.

ports + entry port
Port numbers, level 2; Exploration of recolourable graphs, p. 209
As for port numbers, and on arrival at v the agent also reads the port $\mathrm{port}_v(e)$ of the edge e by which it entered. This is the standard convention of single-agent exploration.

ports, no entry port
Port numbers, level 1; Exploration of recolourable graphs, p. 209
The edges at v carry port numbers $1, \dots, \deg(v)$, assigned by an adversary independently at each endpoint (port_v). The agent chooses the port by which it leaves, but on arrival it is not told the port by which it entered (*unknown inport*).

presence
Vision of the other agents, level 1; Collision-free exploration, p. 236
An agent notices that the other agent is on a neighbouring node, but not on which one (Das et al. 2025a).

present from the start
Black hole appearance, level 1; Black hole search, p. 294
The black hole is present at time 0, and the home base s is safe throughout.

priority (queen)
Goal, level 1; Evacuation, p. 510
One designated robot (the queen) must reach the exit; the others (servants) only help; the task ends when the queen exits.

private coins
Randomness, level 1; Mobile agents on graphs, p. 12
Each agent additionally has access to independent fair coin flips; guarantees hold with probability 1, in expectation, or with high probability, as stated per result.

radius 1
Vision of the other agents, level 2; Collision-free exploration, p. 236
An agent sees the subgraph induced by the nodes at distance at most 1, with its port numbers and the agents on these nodes.

radius 2
Vision of the other agents, level 3; Collision-free exploration, p. 236
An agent sees the subgraph induced by the nodes at distance at most 2, with its port numbers and the agents on these nodes (Bhagat and Pelc 2025).

randomized
Randomness, level 1; Mobile robots in the plane, p. 25
Robots may use random bits in Compute; guarantees hold with probability 1 or in expectation, as stated per result.

randomized
Randomness, level 1; Search in geometric domains, p. 41
Trajectories may depend on random choices; the measure is the expected competitive ratio against an oblivious adversary.

randomized
Randomness, level 1; Programmable matter, p. 46
Particles flip fair coins; guarantees hold with high probability or in expectation.

rays
Domain, level 1; Evacuation, p. 509

Claude (Anthropic) was used in cataloguing the problems.

$m \geq 2$ half-lines share an endpoint, the junction; the exit lies on one ray at distance at least 1 from the start; the measure is the competitive ratio.

recolourable
Recolouring, level 2; Exploration of recolourable graphs, p. 210
At every step the agent at v may set $\psi(v)$ to any colour of the palette, including back to c_0, any number of times (the convention of Takahashi et al. 2025 and Hiraoka et al. 2026).

restricted start
Initial configurations, level 1; Gathering, p. 435
The algorithm gathers only from a stated family of initial configurations that leaves out some solvable ones, typically symmetric configurations; the family is named on each result.

return
Termination requirement, level 0; Exploration of recolourable graphs, p. 211
The agent visits every node and then halts at its start node s (*exploration with return*).

rigid
Movement, level 1; Mobile robots in the plane, p. 23
Every Move phase ends at the computed destination.

ring
Network, level 1; Exploration by luminous robots, p. 181
An anonymous, unoriented ring of n nodes; n is unknown to the robots.

ring
Network, level 1; Gathering by oblivious robots on graphs, p. 145
An anonymous, unoriented ring of n nodes.

ring
Topology, level 1; Mobile agents on graphs, p. 15
G is a cycle C_n (every node of degree 2), or a path when stated.

rooted start
Initial configuration, level 1; Mobile agents on graphs, p. 16
All k agents start at the same node ($\ell = 1$). Also called *co-located* (black hole search: all agents at the *home base*) or a *rooted graph* (Molla and Moses Jr. 2019).

rotation
Mechanism, level 0; Centralised transformations of modular robots, p. 637
A module may rotate 90° around an adjacent module, if the cells it sweeps are empty.

rotation + sliding
Mechanism, level 1; Centralised transformations of modular robots, p. 637
In addition, a module may slide one cell along a line of modules.

round robin
Activation order, level 1; Gathering by oblivious robots on graphs, p. 144
With the sequential scheduler: the robots are activated one per round in a cyclic order that the adversary fixes at the start, that the robots do not know, and that repeats forever; an epoch is k consecutive rounds, one activation per robot.

search
Goal, level 1; Linear search (cow path), p. 500
The task ends when the first robot reaches the target (and, with faults, the others can be sure of it).

search
Goal, level 1; Evacuation, p. 510
The task ends as soon as a non-faulty robot has reached the exit and every non-faulty robot knows its position for certain; the robots need not go to the exit.

search-and-fetch
Goal, level 1; Evacuation, p. 510
A treasure lies at an unknown point of the boundary, and the exit at a known arc distance α from it; the task ends when some robot has carried the treasure to the exit.

Claude (Anthropic) was used in cataloguing the problems.

segment
Fence, level 1; Patrolling, p. 569
An open fence: the segment $[0, \ell]$; robots may reverse direction anywhere.

sender/receiver
Communication, level 1; Search in geometric domains, p. 40
Sender/receiver: one robot can only transmit and the other only receive, instantly at any distance.

sense of direction
Knowledge of the graph, level 1; Black hole search, p. 293
Agents have a consistent sense of direction: a globally consistent labelling of edges from which they can tell whether two walks lead to the same node.

sequential
Scheduler, level 2; Gathering by oblivious robots on graphs, p. 143
Sequential: an SSYNC scheduler that activates exactly one robot per round, fairly; which robot is the adversary's choice unless the activation order is fixed (see the order axis).

sequential
Scheduler, level 2; Mobile robots in the plane, p. 20
Sequential: an SSYNC scheduler that activates exactly one robot per round (fairly). Also called a *centralized* scheduler (Terai et al. 2023); a *round-robin* scheduler is a fair special case.

sequential
Scheduler, level 2; Programmable matter, p. 45
At most one particle is active at any time.

several agents
Agents, level 1; Map construction, p. 158
$k \geq 2$ identical agents.

simultaneous start
Start, level 1; Rendezvous, p. 121
Both agents start in the same round ($\tau = 0$).

simultaneous start
Start, level 1; Collision-free exploration, p. 236
All agents start in the same round.

speed known
Knowledge of the target, level 1; Search for a moving target, p. 547
The robots know u but not d.

square
Domain, level 1; Evacuation, p. 509
The unit square; robots start at its centroid; the exit is a point of the boundary.

square, boundary start
Domain, level 1; Evacuation, p. 509
The unit square; the robots start together at a given point of its boundary, at distance $x \leq 1/2$ from the nearest corner; the exit is a point of the boundary.

SSYNC
Scheduler, level 1; Exploration by oblivious robots, p. 166
SSYNC: time is divided into rounds; in each round a fair adversary chooses a non-empty subset of robots, which execute one atomic Look–Compute–Move cycle together. Called *ATOM* in (Lamani et al. 2010).

SSYNC
Scheduler, level 1; Exploration by luminous robots, p. 178
SSYNC: time is divided into rounds; in each round a fair adversary chooses a non-empty subset of robots, which execute one atomic Look–Compute–Move cycle together. Called *ATOM* in (Datta et al. 2013).

SSYNC
Scheduler, level 1; Gathering by oblivious robots on graphs, p. 143

Claude (Anthropic) was used in cataloguing the problems.

SSYNC: time is divided into rounds; in each round a fair adversary chooses a non-empty subset of robots, which execute one atomic Look–Compute–Move cycle together.

SSYNC

Scheduler, level 1; Mobile robots in the plane, p. 20

SSYNC: time is divided into rounds; in each round a non-empty subset of robots, chosen by a fair adversary, executes one atomic Look–Compute–Move cycle. Called *Atom* by Prencipe (Prencipe 2007).

static graph

Graph dynamics, level 2; Mobile agents on graphs, p. 10

$G_t = G$ for all t: the edge set never changes.

stationary

Movement, level 0; Leader election in programmable matter, p. 586

No particle ever expands or contracts; the configuration is fixed and only states change.

stays connected

Connectivity during execution, level 0; Programmable matter, p. 47

At every time the occupied nodes induce a connected subgraph.

stop

Termination requirement, level 1; Exploration of recolourable graphs, p. 211

The agent visits every node and then halts, at any node.

strong detection

Multiplicity detection, level 2; Gathering by oblivious robots on graphs, p. 144

A snapshot gives the number of robots on every node.

strong multiplicity

Multiplicity detection, level 2; Mobile robots in the plane, p. 24

A snapshot is a multiset: each occupied point comes with the exact number of robots on it.

strong scheduler

Scheduler, level 1; Programmable matter, p. 45

Each activation is executed atomically; the adversary chooses the order, fairly.

synchronous

Scheduler, level 2; Programmable matter, p. 45

Time proceeds in rounds; in every round every particle executes one look–compute–act cycle, and what a particle sends in a round is received in the next (Feldmann et al. 2022). It is an alternative to the sequential scheduler: one activates every particle at once, the other one at a time.

synchronous

Timing, level 1; Mobile agents on graphs, p. 14

Time proceeds in rounds $t = 1, 2, \ldots$; in every round every agent communicates, computes and moves (at most one edge) simultaneously.

terminating

Exploration goal, level 1; Exploration by oblivious robots, p. 167

From every allowed initial configuration, every node is visited by at least one robot and the robots reach a configuration in which no robot ever moves again. With coin flips, this must happen with probability 1.

terminating

Exploration goal, level 1; Exploration by luminous robots, p. 180

Every node is visited by at least one robot, and from some time on no robot is enabled (no robot changes its position or colour again).

the team

What must be visited, level 1; Collision-free exploration, p. 235

Every node must be visited by at least one agent, and all agents stop.

three robots

Number of robots, level 1; Evacuation, p. 510

Three robots.

Claude (Anthropic) was used in cataloguing the problems.

tokens
Marking nodes, level 1; Black hole search, p. 293
Agents carry tokens (pebbles) they can place at nodes or on ports; tokens are identical and carry no other information.

topology unknown
Knowledge of the graph, level 0; Black hole search, p. 293
Agents know n but not the topology of G.

torus
Network, level 1; Exploration by luminous robots, p. 181
An $a \times b$ torus: the grid with wrap-around edges in both dimensions, so every node has degree four and there is no border.

toward
Direction of the target, level 1; Search for a moving target, p. 547
The target moves toward the origin at constant speed $u \geq 0$, and the robots know it.

transparent
Opacity, level 1; Mobile robots in the plane, p. 25
Robots do not obstruct each other's view.

tree
Network, level 2; Exploration of recolourable graphs, p. 212
G is a tree (a cactus without cycles).

tree
Network, level 1; Gathering by oblivious robots on graphs, p. 145
An anonymous tree of n nodes without orientation.

tree
Topology, level 1; Mobile agents on graphs, p. 15
G is connected and acyclic: $|E| = n - 1$.

triangle
Domain, level 1; Evacuation, p. 509
The equilateral triangle of unit side; robots start at its centroid; the exit is a point of the boundary.

triangle, boundary start
Domain, level 1; Evacuation, p. 509
A triangle (the equilateral triangle of unit side unless the setting names another family); the robots start together at a point of its boundary, either given (at distance y from the nearest midpoint of a side) or chosen by the algorithm, as the setting says; the exit is a point of the boundary.

triangular grid
Network, level 1; Exploration by luminous robots, p. 181
The subgraph of the infinite triangular grid (every node of degree six, every face an equilateral triangle of unit side) induced by the nodes on or inside a rectangle, one pair of whose sides is parallel to a family of grid lines; the lines parallel to those sides are its *columns*. It is anonymous and its dimensions are unknown. Corner nodes have degree two or three and other border nodes degree three, four or five, so the border is seen from within range as on a square grid, but with more cases.

two agents
Number of agents, level 1; Collision-free exploration, p. 235
Exactly two agents.

two exits
Exits, level 1; Evacuation, p. 511
Two exits at unknown points of the boundary; the task ends when every robot has reached one of them.

two exits
Exits, level 1; Chauffeuring a crashed robot, p. 542
Exactly two exits.

Claude (Anthropic) was used in cataloguing the problems.

two robots
Number of robots, level 1; Linear search (cow path), p. 499
Two robots.

two robots
Number of robots, level 1; Evacuation, p. 510
Two robots.

two robots
Number of robots, level 1; Search with segment visibility, p. 576
Two robots; the visible set at each time is the one segment between them.

two robots
Number of robots, level 1; Search for a moving target, p. 547
Two robots, both of which must reach the target.

two-way
Direction of movement, level 1; Patrolling, p. 569
Every robot moves along the fence in either direction at any speed between 0 and v_i, and may reverse at any point and any time. On a segment and on a graph this is the only meaningful choice.

undirected
Edge direction, level 1; Mobile agents on graphs, p. 11
$G = (V, E)$ with $E \subseteq \binom{V}{2}$; every edge can be traversed in both directions.

uniform c_0
Initial colouring, level 1; Exploration of recolourable graphs, p. 211
Initially $\psi(v) = c_0$ for every node v (*clean* node memory). The colour c_0 counts as one of the c colours.

unknown graph
Knowledge, level 0; Network decontamination, p. 319
Agents know nothing about G.

unlimited visibility
Visibility, level 1; Mobile robots in the plane, p. 24
A robot sees all robots (subject to opacity).

unoriented
Grid orientation, level 0; Dispersion, p. 261
Port labels carry no global meaning: at every node v the δ_v incident edges get the labels $1, \dots, \delta_v$ in an arbitrary order, chosen independently at each node, so an agent that enters v cannot tell from the labels which port continues in the same row or column. This is the default on every graph; on a grid it is called an *unoriented* grid.

$\varphi = 1$
Visibility range, level 0; Exploration by luminous robots, p. 179
In a Look phase a robot at node v sees every node at distance at most $\varphi = 1$ from v (in the graph metric), with the robots on it, their colours, and how many robots each node holds. It sees no node identifiers and, without chirality, no orientation.

$\varphi = 2$
Visibility range, level 1; Exploration by luminous robots, p. 179
As for $\varphi = 1$, but the robot sees every node at distance at most $\varphi = 2$. An algorithm for $\varphi = 2$ works for every larger φ, since a robot can ignore what lies further away.

$\varphi = 3$
Visibility range, level 2; Exploration by luminous robots, p. 179
As for $\varphi = 1$, but the robot sees every node at distance at most $\varphi = 3$.

verify
Certificates, level 1; Deciding graph properties by mobile agents, p. 370
Each agent also receives a certificate string. For every configuration in Π some certificate makes all agents output yes; for every other configuration, whatever the certificate, some agent outputs no.

Claude (Anthropic) was used in cataloguing the problems.

weak detection
Multiplicity detection, level 1; Gathering by oblivious robots on graphs, p. 144
A snapshot marks every occupied node as holding one robot or more than one.

weak multiplicity
Multiplicity detection, level 1; Mobile robots in the plane, p. 24
A snapshot marks each occupied point as holding one robot or more than one.

weak scheduler
Scheduler, level 0; Programmable matter, p. 45
An activation's look, compute and move need not be atomic; concurrent activations may interfere. The scheduler of Bazzi and Briones is described as weak sequential, as non-atomic, and as fair asynchronous by different later papers (Briones et al. 2023; Dufoulon et al. 2021; Chalopin et al. 2026b).

weight estimates
Knowledge of the graph, level 1; Online exploration of weighted graphs, p. 226
The agent is given G and, for every edge e, an interval $[\mathrm{wt}^-(e), \mathrm{wt}^+(e)]$ that contains $\mathrm{wt}(e)$, with $\mathrm{wt}^+(e) \leq \vartheta\, \mathrm{wt}^-(e)$ for a factor $\vartheta \geq 1$ (their α); the exact weight of e is revealed when the agent visits an end of e (Gehnen et al. 2025).

weighted
Goal, level 1; Evacuation, p. 510
Two robots, a light one and a heavy one of known weight $w \geq 1$, both must reach the exit and stay there; with arrival times t_1 (light) and t_2 (heavy), the cost is $(t_1 + w\, t_2)/(1 + w)$.

whiteboards
Marking nodes, level 2; Black hole search, p. 293
Every node has a whiteboard that agents read and write under mutual exclusion.

whiteboards
Node memory, level 1; Mobile agents on graphs, p. 14
Each node v has a whiteboard of bounded size that agents at v can read and write, with mutual exclusion.

whiteboards
Whiteboards, level 1; Map construction, p. 159
Each node has a whiteboard.

wireless
Communication, level 2; Search in geometric domains, p. 40
Wireless: every robot can send messages to every other robot instantly at any distance.

with return
Termination requirement, level 0; Graph exploration by a single agent, p. 90
The agent visits every node and then halts at its start node s.

with stop
Termination requirement, level 1; Graph exploration by a single agent, p. 90
The agent visits every node and then halts, at any node. Also called *terminating exploration*.

Claude (Anthropic) was used in cataloguing the problems.

Appendix C

Timeline

Each year lists the problems first posed then and the papers that introduced a setting, gave the current best result, or proved a lower bound, dated by their first appearance in any form (conference, journal or arXiv preprint).

1951

Graph exploration by a single agent is posed in Shannon (1951). Shannon's maze-solving mouse Theseus (first prototype 1950) explored a 5×5 grid with constant memory per cell (Disser et al. 2019); the first impossibility theorem for finite automata is Budach's (Budach 1978).

1964

Linear search (cow path) is posed in Beck (1964). Bellman posed the problem in 1963 and Beck analysed it in 1964; computer scientists rediscovered it in the late 1980s as the cow-path problem (Kupavskii and Welzl 2021).

1979

- Aleliunas et al. (1979) — Graph exploration by a single agent: Randomized exploration when n is known [current best]

1990

- Deng and Papadimitriou (1999) — Graph exploration by a single agent: Directed graphs with labelled nodes [introductory], Directed graphs with labelled nodes [lower bound]

1991

Map construction is posed in Dudek et al. (1991). Dudek, Jenkin, Milios and Wilkes framed robotic exploration as graph construction. In the logical view: strengthening of Graph exploration by a single agent, which was posed earlier (1951).

- Dudek et al. (1991) — Map construction: One agent with a marker [current best]

1993

Online exploration of weighted graphs is posed in Kalyanasundaram and Pruhs (1994). Posed by Kalyanasundaram and Pruhs at ICALP 1993 (journal version 1994), who asked whether a constant competitive ratio is possible and gave one for planar graphs, as reported by (Baligács 2026) and (Baligács et al. 2026a); the paper itself was not read. In the logical view: variant of Graph exploration by a single agent, which was posed earlier (1951).

- Baeza-Yates et al. (1993) — Linear search (cow path): One robot on a line [current best], One robot on a line [lower bound], One robot on m rays [current best], One robot on m rays [lower bound]
- Kao et al. (1996) — Linear search (cow path): One randomized robot on a line [current best], One randomized robot on a line [lower bound], One randomized robot on m rays [current best]
- Kalyanasundaram and Pruhs (1994) — Online exploration of weighted graphs: Planar graphs [current best]

1995

- Baeza-Yates and Schott (1995) — Evacuation: Line, any number of face-to-face robots [introductory], Line, any number of wireless robots [introductory]

1998

- Bender et al. (2002) — Graph exploration by a single agent: Directed graphs, one pebble, bound on n known [current best]

1999

Pattern formation is posed in Suzuki and Yamashita (1999). Suzuki and Yamashita characterised the patterns that anonymous robots can form in terms of their initial configuration.

Gathering is posed in Suzuki and Yamashita (1999). Suzuki and Yamashita gave an algorithm "moving the robots to a single point in finite steps" and showed that two oblivious robots cannot meet in SSYNC. In the logical view: special case of k-grouping, but was posed before it (2007).

Convergence is posed in Suzuki and Yamashita (1999). Suzuki and Yamashita gave algorithms for converging to a point and for reaching a point; Ando et al. treated limited visibility the same year (Ando et al. 1999).

- Suzuki and Yamashita (1999) — Gathering: Self-stabilizing gathering, full lights, SSYNC, strong multiplicity detection [lower bound]
- Ando et al. (1999) — Convergence: Limited visibility, SSYNC, rigid moves [current best]

2001

Black hole search is posed in Dobrev et al. (2007). Dobrev, Flocchini, Prencipe and Santoro introduced black hole search in rings (DISC 2001) and in arbitrary networks (PODC 2002) (Dobrev et al. 2006). In the logical view: variant of Collaborative exploration, but was posed before it (2004).

- Flocchini et al. (2005) — Gathering: Limited visibility with a compass, ASYNC [current best], Limited visibility with a compass, ASYNC [lower bound]
- Dobrev et al. (2007) — Black hole search: Asynchronous agents with whiteboards in a ring [current best], Asynchronous agents with whiteboards in a ring [lower bound]

2002

Circle formation is posed in Défago and Konagaya (2002). In the logical view: special case of Pattern formation, which was posed earlier (1999).

Claude (Anthropic) was used in cataloguing the problems.

Network decontamination is posed in Barrière et al. (2002). Capture of an intruder by mobile agents (SPAA 2002).

- Diks et al. (2004) — Graph exploration by a single agent: Trees, perpetual exploration [current best], Trees, perpetual exploration [lower bound], Trees, exploration with stop [lower bound], Trees, exploration with return [introductory], Trees, exploration with return [lower bound]
- Défago and Konagaya (2002) — Circle formation: Circle formation, SSYNC [introductory]
- Dobrev et al. (2006) — Black hole search: Asynchronous agents with whiteboards [current best], Asynchronous agents with whiteboards [lower bound], Whiteboards and sense of direction [current best], Whiteboards and sense of direction [lower bound], Whiteboards and a map [current best], Whiteboards and a map [lower bound], Tokens and a map [lower bound]
- Barrière et al. (2002) — Network decontamination: Trees [current best]

2003

Rendezvous is posed in Dessmark et al. (2006). Deterministic rendezvous of labelled agents in arbitrary graphs was posed by Dessmark, Fraigniaud and Pelc at ESA 2003. Rendezvous itself is older (randomized search games); the catalogue covers the deterministic line (Pelc 2019). In the logical view: subproblem of Graph exploration by a single agent, which was posed earlier (1951).

Search for a moving target is posed in Alpern and Gal (2003). An early treatment, as reported by (Coleman et al. 2022): Alpern and Gal give an optimal algorithm for one robot and a target moving away from the origin at a known speed. Coleman et al. write that the case of a moving target "appears to have been first considered by McCabe" (1974), for a target performing a random walk on the integers; that paper is not in the bibliography. In the logical view: generalisation of Linear search (cow path), which was posed earlier (1964).

- Dessmark et al. (2006) — Rendezvous: Labelled agents, arbitrary finite graphs [introductory], Labelled agents, arbitrary finite graphs [lower bound], Labelled agents in trees [current best], Labelled agents in trees [lower bound], Labelled agents in rings, simultaneous start [current best], Labelled agents in rings, simultaneous start [lower bound]
- Cieliebak et al. (2012) — Gathering: ASYNC with weak multiplicity detection [current best]
- Yanovski et al. (2003) — Exploration of recolourable graphs: Rotor-router: a pointer in every node [current best]
- Alpern and Gal (2003) — Search for a moving target: One robot, line, target moving away, speed known [current best], One robot, line, target moving away, speed known [lower bound], One robot, line, target moving away, nothing known [lower bound]

2004

Collaborative exploration is posed in Fraigniaud et al. (2006a). Collective tree exploration was posed at LATIN 2004 (journal version 2006).

Flocking is posed in Gervasi and Prencipe (2004). Coordination without communication: the flocking problem.

- Fraigniaud et al. (2005a) — Graph exploration by a single agent: Arbitrary graphs, perpetual exploration [current best], Arbitrary graphs, perpetual exploration [lower bound]

Claude (Anthropic) was used in cataloguing the problems.

- Fraigniaud and Ilcinkas (2004) — Graph exploration by a single agent: Directed graphs, one pebble [current best], Directed graphs, one pebble [lower bound]
- Fraigniaud et al. (2006a) — Collaborative exploration: Trees, complete communication [introductory], Trees, communication by whiteboards only [current best], Trees, no communication [current best], Trees, no communication [lower bound]
- Cohen and Peleg (2005) — Convergence: ASYNC, unlimited visibility [current best]
- Gervasi and Prencipe (2004) — Flocking: Oblivious robots following a leader [current best]

2005

- Reingold (2008) — Graph exploration by a single agent: Exploration with stop when n is known [current best]
- Fraigniaud et al. (2005b) — Graph exploration by a single agent: Exploration with stop and one pebble [current best]
- Fraigniaud et al. (2006c) — Graph exploration by a single agent: Exploration with stop and one pebble [lower bound]
- Cohen et al. (2008) — Graph exploration by a single agent: Node labels chosen by the designer [current best]; Exploration of recolourable graphs: Label-guided: colours fixed by a designer [current best], Label-guided: colours fixed by a designer [lower bound], Label-guided, oblivious agent [lower bound]
- Fleischer and Trippen (2005) — Graph exploration by a single agent: Directed graphs with labelled nodes [current best]
- De Marco et al. (2006) — Rendezvous: Asynchronous labelled agents [introductory]
- Prencipe (2007) — Gathering: No multiplicity detection, SSYNC [lower bound], Gathering, full lights, SSYNC, no multiplicity detection [lower bound], Gathering, full lights, ASYNC, no multiplicity detection [lower bound]
- Klasing et al. (2007) — Black hole search: Synchronous agents with a map [current best], Synchronous agents with a map [lower bound]
- Flocchini et al. (2008c) — Network decontamination: Hypercubes [current best], Hypercubes [lower bound]

2006

Uniform circle formation is posed in Dieudonné et al. (2008). The first deterministic solution (SSS 2006); circle formation with convergence towards uniformity predates it (Défago and Konagaya 2002). In the logical view: special case of Pattern formation, which was posed earlier (1999).

Uniform deployment on a ring is posed in Flocchini et al. (2008b). Self-deployment of mobile sensors on a ring (ALGOSENSORS 2006).

Gathering by oblivious robots on graphs is posed in Klasing et al. (2008). Gathering of asynchronous oblivious robots on a ring (ISAAC 2006). In the logical view: variant of Gathering in graphs, but was posed before it (2012).

- Ilcinkas (2008) — Graph exploration by a single agent: Port numbers chosen by the designer (periodic exploration) [introductory]
- Agmon and Peleg (2006) — Gathering: Crash faults, SSYNC [introductory], Byzantine faults, SSYNC [lower bound], Byzantine faults, FSYNC [current best]
- Czyzowicz et al. (2009b) — Gathering: Fat robots, few robots, no common orientation [current best]
- Dieudonné et al. (2008) — Uniform circle formation: SSYNC, oblivious [introductory]

Claude (Anthropic) was used in cataloguing the problems.

- Dobrev et al. (2013) — Black hole search: Tokens, no map [introductory]
- Czyzowicz et al. (2006) — Black hole search: Synchronous agents with a map [introductory]
- Flocchini et al. (2008b) — Uniform deployment on a ring: Oriented ring, asynchronous, oblivious [current best], Oriented ring, asynchronous, oblivious [lower bound]
- Klasing et al. (2008) — Gathering by oblivious robots on graphs: Rings, SSYNC, no multiplicity detection [lower bound], Rings, ASYNC, weak multiplicity detection [introductory], Rings, ASYNC, weak multiplicity detection [lower bound]

2007

Treasure hunt is posed in Ta-Shma and Zwick (2014). Ta-Shma and Zwick introduced the deterministic treasure hunt problem at SODA 2007, as a variant of rendezvous in which one agent never moves. In the logical view: subproblem of Graph exploration by a single agent, which was posed earlier (1951).

Leader election in the plane is posed in Dieudonné et al. (2013). Posed for disoriented robots at OPODIS 2007 (Dieudonné and Petit); characterised in (Dieudonné et al. 2013).

k-grouping is posed in Efrima and Peleg (2009). Splitting a swarm into a given number of groups was introduced at SIROCCO 2007, under the name partitioning. No paper titled "k-grouping" was found. In the logical view: special case of Pattern formation, which was posed earlier (1999).

Exploration by oblivious robots is posed in Flocchini et al. (2013). Ring exploration by asynchronous oblivious robots (OPODIS 2007). In the logical view: variant of Graph exploration by a single agent, which was posed earlier (1951).

- Ambühl et al. (2011) — Graph exploration by a single agent: Trees, exploration with stop [current best], Trees, exploration with return [current best]
- Ta-Shma and Zwick (2014) — Treasure hunt: Anonymous finite graphs [introductory]; Rendezvous: Labelled agents, arbitrary finite graphs [current best]
- Das et al. (2007) — Map construction: Identical agents with whiteboards [introductory], Identical agents with whiteboards [current best], Identical agents with whiteboards [lower bound]
- Dieudonné and Petit (2009) — Scattering: Randomized, SSYNC, strong multiplicity detection [introductory], Randomized, SSYNC, strong multiplicity detection [lower bound]
- Efrima and Peleg (2009) — k-grouping: k groups of equal size [current best], k equal groups, no common orientation, deterministic [lower bound], k equal groups, randomized, SSYNC [current best]
- Ilcinkas et al. (2009) — Network decontamination: Unknown graphs, monotone strategies [current best], Unknown graphs, monotone strategies [lower bound]
- Flocchini et al. (2013) — Exploration by oblivious robots: Rings, ASYNC, terminating [introductory], Rings, ASYNC, terminating [lower bound], Rings, SSYNC, terminating [lower bound]

2008

- Fraigniaud and Pelc (2013) — Rendezvous: Anonymous agents in trees, simultaneous start [current best], Anonymous agents in trees, simultaneous start [lower bound]
- Flocchini et al. (2008a) — Pattern formation: ASYNC, no agreement on coordinates [introductory], ASYNC, no agreement on coordinates [lower bound], ASYNC with one common axis [current best], ASYNC with one common axis [lower bound], ASYNC with a common compass [current best]

Claude (Anthropic) was used in cataloguing the problems.

- Défago and Souissi (2008) — Circle formation: Circle formation, SSYNC [current best]
- Flocchini et al. (2012b) — Black hole search: Tokens and a map [current best]
- Clement et al. (2010) — Scattering: Randomized, SSYNC, strong multiplicity detection [current best]

2009

Scattering is posed in Dieudonné and Petit (2009). Dieudonné and Petit showed scattering is deterministically impossible and gave a randomized solution (preprint 2007).

- Czyzowicz et al. (2012c) — Graph exploration by a single agent: Port numbers chosen by the designer (periodic exploration) [current best], Port numbers chosen by the designer (periodic exploration) [lower bound]
- Lamani et al. (2010) — Exploration by oblivious robots: Rings, ASYNC, terminating [current best], Rings, ASYNC, terminating [lower bound], Rings, SSYNC, terminating [current best], Rings, SSYNC, terminating [lower bound]
- Devismes et al. (2013) — Exploration by oblivious robots: Rings, ASYNC, terminating [lower bound], Rings, SSYNC, terminating [lower bound], Rings, probabilistic, SSYNC, terminating [current best], Rings, probabilistic, SSYNC, terminating [lower bound]
- Bampas et al. (2017) — Exploration of recolourable graphs: Rotor-router with pointers set by a designer [current best]
- Miyazaki et al. (2009) — Online exploration of weighted graphs: Weighted cycles [current best], Weighted cycles [lower bound]

2010

Pattern sequences is posed in Das et al. (2015b). Introduced at PODC 2010. In the logical view: generalisation of Pattern formation, which was posed earlier (1999).

Deciding graph properties by mobile agents is posed in Fraigniaud and Pelc (2017). Fraigniaud and Pelc introduced decision problems for mobile agents and the classes MAD and MAV, as (Bampas and Ilcinkas 2018) reports (arXiv 2010; LATIN 2012; Journal of Parallel and Distributed Computing 2017).

- Czyzowicz et al. (2012b) — Rendezvous: Anonymous agents: memory [current best], Anonymous agents: memory [lower bound]
- Chaudhuri and Mukhopadhyaya (2010) — Gathering: Transparent fat robots, no chirality [introductory]
- Chalopin et al. (2010a) — Map construction: One agent, no marking [current best]
- Izumi et al. (2018) — Scattering: Limited visibility, connectivity preserving [current best], Limited visibility, connectivity preserving [lower bound]
- Das et al. (2015b) — Pattern sequences: Oblivious, SSYNC, chirality [current best]
- Blin et al. (2010) — Exploration by oblivious robots: Rings, ASYNC, exclusive perpetual [introductory], Rings, ASYNC, exclusive perpetual [lower bound]
- Fraigniaud and Pelc (2017) — Deciding graph properties by mobile agents: Any number of agents [lower bound], Two or more agents that know their number [current best], One agent [lower bound], One agent that knows n [lower bound], One agent with a map [current best]

2011

Patrolling is posed in Czyzowicz et al. (2011). Boundary patrolling by agents with distinct speeds (ESA 2011).

- Guilbault and Pelc (2011) — Rendezvous: Asynchronous anonymous agents [current best]
- Degener et al. (2011) — Gathering: Limited visibility, FSYNC, disoriented [introductory]
- Katreniak (2011) — Convergence: Limited visibility, asynchronous [introductory]
- Dieudonné and Pelc (2016) — Gathering in graphs: Anonymous agents [current best], Anonymous agents [lower bound]
- Czyzowicz et al. (2011) — Patrolling: Open fence, distinct speeds [introductory], Closed fence, one-way, distinct speeds [introductory], Closed fence, one-way, distinct speeds [lower bound], Closed fence, two-way, distinct speeds [introductory]
- Devismes et al. (2021) — Exploration by oblivious robots: Grids, ASYNC, terminating [current best], Grids, ASYNC, terminating [lower bound]

2012

Gathering in graphs is posed in Dieudonné et al. (2014). Gathering many agents in arbitrary graphs grew out of two-agent rendezvous; the Byzantine version was posed at SODA 2012. In the logical view: generalisation of Rendezvous, which was posed earlier (2003).

Near-gathering is posed in Pagli et al. (2015). Introduced at SIROCCO 2012. In the logical view: relaxation of Gathering, which was posed earlier (1999).

Collision-free exploration is posed in Czyzowicz et al. (2017a). Formalised by Czyzowicz, Dereniowski, Gąsieniec, Klasing, Kosowski and Pająk for agents on graphs with labelled nodes (a 2012 report; LATIN 2014; Journal of Computer and System Sciences 2017); (Papin 2026) calls this the first formalisation. Bhagat and Pelc posed it for anonymous graphs (Bhagat and Pelc 2025). In the logical view: variant of Graph exploration by a single agent, which was posed earlier (1951).

- Dieudonné and Pelc (2012) — Graph exploration by a single agent: Exploration with stop, nothing known [lower bound]
- Das et al. (2016) — Gathering: Rendezvous, full lights (LUMI), ASYNC, non-rigid [introductory]
- Viglietta (2013) — Gathering: Rendezvous, full lights (LUMI), ASYNC, non-rigid [lower bound]
- Agathangelou et al. (2013) — Gathering: Fat robots, chirality, any number [current best]
- Chaudhuri and Mukhopadhyaya (2015) — Gathering: Transparent fat robots, no chirality [current best]; Leader election in the plane: Transparent fat robots, no chirality [current best], Transparent fat robots, no chirality [lower bound]
- Balamohan et al. (2016) — Black hole search: Tokens, no map [current best], Tokens, no map [lower bound]
- Dieudonné et al. (2014) — Gathering in graphs: Weakly Byzantine agents [current best], Weakly Byzantine agents [lower bound], Byzantine agents [introductory], Byzantine agents [lower bound]
- Dieudonné et al. (2013) — Leader election in the plane: Disoriented robots, with or without chirality [current best], Disoriented robots, with or without chirality [lower bound]
- Pagli et al. (2015) — Near-gathering: ASYNC, limited visibility, one common axis [current best]
- Kawamura and Kobayashi (2015) — Patrolling: Open fence, distinct speeds [lower bound]

Claude (Anthropic) was used in cataloguing the problems.

- Datta et al. (2013) — Exploration by luminous robots: Rings, FSYNC, terminating, $\varphi = 1$ [lower bound], Rings, oblivious, FSYNC, terminating, $\varphi = 1$ [current best], Rings, oblivious, FSYNC, terminating, $\varphi = 1$ [lower bound], Rings, oblivious, ASYNC, terminating, $\varphi = 1$ [lower bound]
- D'Angelo et al. (2016) — Gathering by oblivious robots on graphs: Finite grids, ASYNC, no multiplicity detection [current best], Finite grids, ASYNC, no multiplicity detection [lower bound], Trees, ASYNC, no multiplicity detection [current best], Trees, ASYNC, no multiplicity detection [lower bound]
- Czyzowicz et al. (2017a) — Collision-free exploration: Labelled trees, with a map, any number of agents [current best], Labelled trees, with a map, any number of agents [lower bound], Labelled graphs, with a map, any number of agents [current best], Labelled graphs, with a map, any number of agents [lower bound], Labelled trees, no knowledge, any number of agents [current best], Labelled trees, no knowledge, any number of agents [lower bound], Labelled graphs, no knowledge, any number of agents [current best], Labelled graphs, no knowledge, any number of agents [lower bound]
- Anaya et al. (2016) — Delivery by energy-constrained agents: Paths, equal budgets [current best]

2013

Delivery by energy-constrained agents is posed in Chalopin et al. (2013). Chalopin, Das, Mihalák, Penna and Widmayer first studied collaborative delivery, as (Chalopin et al. 2021) reports (ALGOSENSORS 2013). Anaya et al. had treated the transfer between two points of a path as part of convergecast with equal budgets (DISC 2012) (Anaya et al. 2016).

- Ilcinkas and Wade (2018) — Graph exploration by a single agent: Dynamic rings [current best], Dynamic rings [lower bound]
- Dereniowski et al. (2015a) — Collaborative exploration: Large teams, complete communication [current best], Large teams, complete communication [lower bound], Large teams, local communication [current best], Large teams, local communication [lower bound]
- Dieudonné et al. (2015) — Rendezvous: Asynchronous labelled agents [current best]
- Flocchini et al. (2016a) — Gathering: Two robots, SSYNC [lower bound], Rendezvous, external light (FCOM), SSYNC [current best], Rendezvous, external light (FCOM), ASYNC, non-rigid [current best], Rendezvous, external light (FCOM), ASYNC, rigid [introductory], Rendezvous, internal light (FSTA), SSYNC, non-rigid [current best], Rendezvous, internal light (FSTA), SSYNC, rigid [introductory]
- Bramas and Tixeuil (2017) — Scattering: Randomized, SSYNC, weak multiplicity detection [current best], Randomized, SSYNC, weak multiplicity detection [lower bound]
- Chalopin et al. (2013) — Delivery by energy-constrained agents: Arbitrary graphs [current best], Arbitrary graphs [lower bound], Arbitrary graphs, equal budgets [current best], Arbitrary graphs, equal budgets [lower bound]

2014

Evacuation is posed in Czyzowicz et al. (2014). Czyzowicz et al. posed evacuation of k robots from a disk through an unknown exit on its boundary at DISC 2014. In the logical view: generalisation of Linear search (cow path), which was posed earlier (1964).

Mutual visibility is posed in Di Luna et al. (2017). Introduced at CCCG 2014 for oblivious robots and at SSS 2014 for robots with lights (Di Luna et al. 2014); journal version (Di Luna et al. 2017). In the logical view: special case of Pattern formation, which was posed earlier (1999).

Coating is posed in Derakhshandeh et al. (2017). Infinite object coating appeared as an arXiv preprint in 2014; universal coating in (Derakhshandeh et al. 2017).

Claude (Anthropic) was used in cataloguing the problems.

- Miller and Pelc (2015b) — Treasure hunt: Treasure hunt with advice [current best], Treasure hunt with advice [lower bound]
- Yamauchi and Yamashita (2014) — Pattern formation: Randomized, ASYNC, chirality [introductory]
- Flocchini et al. (2017) — Circle formation: Circle formation [current best]; Uniform circle formation: ASYNC, oblivious, disoriented [current best], ASYNC, oblivious, disoriented [lower bound]
- Czyzowicz et al. (2014) — Evacuation: Two wireless robots, disk [current best], Two wireless robots, disk [lower bound], Two face-to-face robots, disk [introductory], Three wireless robots, disk [current best], Three wireless robots, disk [lower bound], Three face-to-face robots, disk [current best], Three face-to-face robots, disk [lower bound], Many wireless robots, disk [current best], Many wireless robots, disk [lower bound], Many face-to-face robots, disk [current best], Many face-to-face robots, disk [lower bound], Disk, k wireless robots, f Byzantine [lower bound], Disk, k face-to-face robots, f Byzantine [lower bound], Disk, k face-to-face robots, f crash faults [lower bound]
- Das et al. (2020) — Pattern sequences: Luminous robots, ASYNC, non-rigid, chirality [current best], Luminous robots, ASYNC, non-rigid, chirality [lower bound]
- Dumitrescu et al. (2014) — Patrolling: Closed fence, two-way, distinct speeds [current best]
- Chalopin et al. (2014) — Delivery by energy-constrained agents: Paths [current best], Paths [lower bound]

2015

Leader election in programmable matter is posed in Derakhshandeh et al. (2015b). Introduced with shape formation at DNA 21 (2015).

Shape formation in programmable matter is posed in Derakhshandeh et al. (2015b). Introduced with leader election at DNA 21 (2015).

- Erlebach et al. (2021) — Graph exploration by a single agent: Temporal graphs, schedule known [current best], Temporal graphs, schedule known [lower bound]
- Bhagat et al. (2016a) — Gathering: Crash faults, one common axis, opaque robots [current best]
- Chrobak et al. (2015) — Linear search (cow path): Two robots of different speeds, both must reach the target [introductory]; Evacuation: Line, any number of face-to-face robots [current best], Line, any number of face-to-face robots [lower bound]
- Czyzowicz et al. (2020d) — Evacuation: Two face-to-face robots, disk [lower bound]
- Czyzowicz et al. (2015c) — Evacuation: Triangle, two wireless robots starting on the boundary [introductory], Triangle, two wireless robots at the centroid [current best], Triangle, two wireless robots at the centroid [lower bound], Triangle, any number of wireless robots at the centroid [introductory], Triangle, any number of wireless robots at the centroid [lower bound], Triangle, any number of wireless robots starting on the boundary [current best], Triangle, any number of wireless robots starting on the boundary [lower bound], Square, any number of wireless robots at the centre [current best], Square, any number of wireless robots at the centre [lower bound], Square, two wireless robots starting on the boundary [current best], Square, two wireless robots starting on the boundary [lower bound]
- Bouchard et al. (2016) — Gathering in graphs: Byzantine agents [current best]
- Di Luna et al. (2020b) — Exploration of dynamic graphs by teams: 1-interval-connected rings [introductory], 1-interval-connected rings [lower bound]
- Di Luna et al. (2017) — Mutual visibility: Lights, SSYNC, rigid [current best], Lights, SSYNC, rigid [lower bound], Lights, SSYNC, non-rigid [introductory], Lights, SSYNC, non-rigid [lower bound], Finite state, SSYNC, non-rigid [lower bound], Lights, ASYNC, rigid [introductory], Lights, ASYNC, rigid [lower bound], Lights, ASYNC, non-rigid, one axis [introductory], Lights, ASYNC, non-rigid, one axis [lower bound], Lights, crash faults, ASYNC, one axis [lower bound], Lights, mobility faults, SSYNC, non-rigid [lower bound]

Claude (Anthropic) was used in cataloguing the problems.

- Sharma et al. (2015b) — Mutual visibility: Lights, SSYNC, non-rigid [current best], Lights, ASYNC, non-rigid, one axis [current best]
- Vaidyanathan et al. (2015) — Mutual visibility: Lights, FSYNC, rigid, chirality [current best]
- Derakhshandeh et al. (2015b) — Leader election in programmable matter: Randomized, with chirality [introductory]; Shape formation in programmable matter: SILBOT, full orientation [lower bound]
- Czyzowicz et al. (2017e) — Patrolling: Graphs with faulty patrollers [current best], Graphs with faulty patrollers [lower bound]
- Menc et al. (2017) — Exploration of recolourable graphs: Rotor-router: a pointer in every node [lower bound]
- D'Emidio et al. (2018) — Gathering by oblivious robots on graphs: Rings, ASYNC, weak multiplicity detection [lower bound]
- Bampas and Ilcinkas (2018) — Deciding graph properties by mobile agents: Any number of agents, with certificates [lower bound]

2016

Uniform deployment on graphs is posed in Shibata et al. (2018). Agents that have memory but see only the node they are on, in asynchronous unidirectional rings (PODC 2016), as reported by (Shibata et al. 2022a); not read. Look–Compute–Move robots that see several nodes had been deployed on rings and grids before, as listed in (Shibata et al. 2022a).

Exploration of dynamic graphs by teams is posed in Di Luna et al. (2020b). Live exploration of dynamic rings was posed at ICDCS 2016. In the logical view: variant of Collaborative exploration, which was posed earlier (2004).

Compression is posed in Cannon et al. (2016). PODC 2016.

- Disser et al. (2019) — Graph exploration by a single agent: Exploration with stop when n is known [lower bound], Exploration with return and pebbles [current best], Exploration with return and pebbles [lower bound]; Collaborative exploration: Anonymous graphs, a team of constant-memory agents [current best], Anonymous graphs, a team of constant-memory agents [lower bound]
- Disser et al. (2020) — Collaborative exploration: Trees, complete communication [lower bound], Trees with $k = n$ agents [current best], Trees with $k = n$ agents [lower bound]; Dispersion: Trees [lower bound], Rooted trees [lower bound]
- Shibata et al. (2018) — Uniform deployment on graphs: Asynchronous unidirectional rings [current best], Asynchronous unidirectional rings [lower bound]
- Pattanayak et al. (2019b) — Gathering: Crash faults, SSYNC [current best], Crash faults, SSYNC [lower bound], Crash faults, ASYNC, restricted start [current best]
- Balabonski et al. (2018b) — Gathering: FSYNC, no multiplicity detection [current best]
- Czyzowicz et al. (2019c) — Linear search (cow path): k robots on a line, f crash faults [current best]
- Czyzowicz et al. (2021a) — Linear search (cow path): k robots on a line, Byzantine faults [current best]
- Bampas et al. (2019b) — Linear search (cow path): Two robots of different speeds, both must reach the target [current best], Two robots of different speeds, both must reach the target [lower bound]; Evacuation: Line, any number of face-to-face robots [lower bound], Line, any number of wireless robots [current best], Line, any number of wireless robots [lower bound], Line, two face-to-face robots of different speeds [current best], Line, two face-to-face robots of different speeds [lower bound], Line, two wireless robots of different speeds [current best], Line, two wireless robots of different speeds [lower bound]

Claude (Anthropic) was used in cataloguing the problems.

- Lamprou et al. (2020) — Evacuation: Disk, two wireless robots of different speeds [current best], Disk, two wireless robots of different speeds [lower bound]
- Georgiou et al. (2019) — Evacuation: Disk, search-and-fetch, two wireless robots [current best], Disk, search-and-fetch, two face-to-face robots [current best], Disk, search-and-fetch, two face-to-face robots [lower bound]
- Derakhshandeh et al. (2016) — Shape formation in programmable matter: Universal shapes, sequential, randomized [current best], Universal shapes, sequential, randomized [lower bound]
- Derakhshandeh et al. (2017) — Coating: Universal coating [introductory], Universal coating [lower bound]
- Daymude et al. (2018) — Coating: Universal coating [current best], Universal coating [lower bound]
- Cannon et al. (2016) — Compression: Markov chain, sequential [current best]
- Bérard et al. (2016) — Exploration by oblivious robots: Rings, ASYNC, exclusive perpetual [current best]
- Bärtschi et al. (2020) — Delivery by energy-constrained agents: Arbitrary graphs [current best], Arbitrary graphs [lower bound], Planar graphs, equal budgets [lower bound]

2017

Bridging is posed in Arroyo et al. (2018). arXiv 2017; Natural Computing 2018. In the logical view: generalisation of Compression, which was posed earlier (2016).

Shape formation by hybrid programmable matter is posed in Gmyr et al. (2020). First presented at EuroCG 2017 (non-archival); DNA 24 (2018); Natural Computing 2020. The work began at a Dagstuhl seminar in 2016.

Centralised transformations of modular robots is posed in Michail et al. (2019). arXiv 2017; JCSS 2019.

- Augustine and Moses Jr. (2018) — Collaborative exploration: Anonymous graphs with $k = n$ agents [introductory]; Dispersion: Arbitrary graph, arbitrary initial configuration [introductory], Arbitrary graph, arbitrary initial configuration [lower bound], Rooted initial configuration [introductory], Rooted initial configuration [lower bound], Asynchronous agents [lower bound], Rings and paths [introductory], Rings and paths [lower bound], Trees [introductory], Trees [lower bound], Rooted trees [current best], Rooted trees [lower bound], Crash faults [lower bound], Weak Byzantine agents [lower bound], Weak Byzantine agents on rings [lower bound], Weak Byzantine agents on oriented grids [lower bound]
- Agarwalla et al. (2018) — Dispersion: Dynamic rings [current best]
- Cicerone et al. (2019) — Pattern formation: ASYNC, no agreement on coordinates [current best]
- Poudel and Sharma (2021) — Gathering: Limited visibility with a compass, rigid, ASYNC [current best], Limited visibility with a compass, rigid, ASYNC [lower bound], Limited visibility, one common axis, rigid, ASYNC [current best], Limited visibility, one common axis, rigid, ASYNC [lower bound]
- Heriban et al. (2018) — Gathering: Rendezvous, full lights (LUMI), ASYNC, non-rigid [current best]
- Mondal and Chaudhuri (2017) — Uniform circle formation: Transparent fat robots, SSYNC [current best]
- Kupavskii and Welzl (2021) — Linear search (cow path): k robots on a line, f crash faults [lower bound], k robots on a line, Byzantine faults [lower bound]; Evacuation: Line, $2f + 1$ wireless robots, f crash faults [lower bound]
- Czyzowicz et al. (2017c) — Evacuation: Three wireless robots, one crash fault [current best], Three wireless robots, one crash fault [lower bound], Three wireless robots, one Byzantine [current best], Three wireless robots, one Byzantine [lower bound]
- Chuangpishit et al. (2020b) — Evacuation: Two face-to-face robots, equilateral triangle [current best], Two face-to-face robots, equilateral triangle [lower bound], Two face-to-face robots, square [current best], Two face-to-face

Claude (Anthropic) was used in cataloguing the problems.

robots, square [lower bound], Triangle, any number of face-to-face robots [current best], Triangle, any number of face-to-face robots [lower bound]

- Pattanayak et al. (2018) — Evacuation: Two robots, two exits, face-to-face [current best], Two robots, two exits, face-to-face [lower bound], Two robots, two exits, wireless [current best], Two robots, two exits, wireless [lower bound]
- Brandt et al. (2020b) — Evacuation: m rays, $k = m$ wireless robots at the junction [current best], m rays, $k = m$ wireless robots at the junction [lower bound]
- Bhagat and Mukhopadhyaya (2019) — Mutual visibility: Finite state, SSYNC, non-rigid [current best]
- Sharma et al. (2021a) — Mutual visibility: Lights, ASYNC, rigid [current best], Lights, ASYNC, rigid [lower bound]
- Aljohani and Sharma (2018) — Mutual visibility: Lights, one crash, SSYNC, rigid, compass [current best], Lights, one Byzantine robot [lower bound]
- Daymude et al. (2017) — Leader election in programmable matter: Randomized, with chirality [current best]
- Di Luna et al. (2020e) — Shape formation in programmable matter: Deterministic, simply connected start, no chirality [current best], Deterministic, simply connected start, no chirality [lower bound]
- Arroyo et al. (2018) — Bridging: Markov chain, sequential [current best]
- Gmyr et al. (2020) — Shape formation by hybrid programmable matter: Planar [current best], Planar [lower bound]
- Michail et al. (2019) — Centralised transformations of modular robots: Rotation only [current best], Rotation and sliding [current best], Rotation and sliding [lower bound]

2018

Dispersion is posed in Augustine and Moses Jr. (2018). Posed at ICDCN 2018, fourteen years after collaborative exploration; the introductory paper already relates dispersion to n-robot collaborative exploration and takes its tree lower bound from it. In the logical view: derived from Collaborative exploration, which was posed earlier (2004).

Convex hull formation is posed in Daymude et al. (2020). arXiv 2018; ICDCN 2020.

Exploration by luminous robots is posed in Ooshita and Tixeuil (2022). Ring exploration by myopic robots with a two-colour light (SSS 2018). In the logical view: variant of Exploration by oblivious robots, which was posed earlier (2007).

Separation of coloured particles is posed in Cannon et al. (2019). arXiv 2018; APPROX/RANDOM 2019. In the logical view: generalisation of Compression, which was posed earlier (2016).

- Kshemkalyani and Ali (2019) — Collaborative exploration: Anonymous graphs with $k = n$ agents [lower bound]; Dispersion: Arbitrary graph, arbitrary initial configuration [lower bound], Rooted initial configuration [lower bound], Asynchronous agents [introductory], Asynchronous agents [lower bound], Rings and paths [lower bound], Anonymous agents with randomness [lower bound], Crash faults [lower bound], Weak Byzantine agents [lower bound], Strong Byzantine agents [lower bound]
- Kshemkalyani et al. (2020c) — Dispersion: Unoriented grids [current best], Unoriented grids [lower bound], Oriented grids [lower bound], Crash faults on unoriented grids [lower bound], Crash faults on oriented grids [lower bound], Weak Byzantine agents on oriented grids [lower bound]
- Pelc and Yadav (2019b) — Rendezvous: Anonymous agents: memory [current best], Anonymous agents: memory [lower bound]
- Okumura et al. (2023) — Gathering: Rendezvous, external light (FCOM), SSYNC [lower bound]

Claude (Anthropic) was used in cataloguing the problems.

- Terai et al. (2023) — Gathering: Gathering, full lights, SSYNC, no multiplicity detection [current best], Gathering, full lights, ASYNC, no multiplicity detection [introductory], Gathering, external light (FCOM), SSYNC, rigid [current best], Gathering, internal light (FSTA), SSYNC, non-rigid [introductory], Gathering, external light, sequential scheduler [current best], Gathering, external light, sequential scheduler [lower bound]
- Feletti et al. (2018) — Uniform circle formation: Opaque robots with lights, FSYNC [introductory]
- Czyzowicz et al. (2020b) — Evacuation: Priority evacuation (queen and servants), disk [current best], Priority evacuation (queen and servants), disk [lower bound]
- Czyzowicz et al. (2020a) — Evacuation: Disk, priority evacuation, queen and one servant [introductory], Disk, priority evacuation, queen and two servants [current best], Disk, priority evacuation, queen and two servants [lower bound]
- Sharma et al. (2018) — Mutual visibility: Fat robots, FSYNC, rigid [introductory]
- Poudel et al. (2021) — Mutual visibility: Lights, crash faults, ASYNC, one axis [current best]
- Gastineau et al. (2018) — Leader election in programmable matter: Stationary, chirality, no holes [current best]
- Bazzi and Briones (2019) — Leader election in programmable matter: Stationary, chirality, weak scheduler [current best], Stationary, chirality, weak scheduler [lower bound]
- Daymude et al. (2020) — Convex hull formation: No orientation [current best]
- Haeupler et al. (2019) — Patrolling: Open fence, distinct speeds [current best], Open fence, distinct speeds [lower bound], Closed fence, one-way, distinct speeds [current best]
- Ooshita and Tixeuil (2022) — Exploration by luminous robots: Rings, FSYNC, perpetual, $\varphi = 1$ [current best], Rings, FSYNC, perpetual, $\varphi = 1$ [lower bound], Rings, FSYNC, terminating, $\varphi = 1$ [current best], Rings, FSYNC, terminating, $\varphi = 1$ [lower bound], Rings, ASYNC, perpetual, $\varphi = 1$ [current best], Rings, ASYNC, perpetual, $\varphi = 1$ [lower bound], Rings, ASYNC, terminating, $\varphi = 1$ [current best], Rings, ASYNC, terminating, $\varphi = 1$ [lower bound]
- Cannon et al. (2019) — Separation of coloured particles: Markov chain, sequential [current best]

2019

Chauffeuring a crashed robot is posed in Pattanayak et al. (2019a). ALGOSENSORS 2019. In the logical view: variant of Evacuation, which was posed earlier (2014).

- Molla and Moses Jr. (2019) — Dispersion: More agents than nodes [current best], More agents than nodes [lower bound], Anonymous agents with randomness [introductory], Anonymous agents with randomness [lower bound]
- Bose et al. (2021a) — Pattern formation: Opaque robots with lights, ASYNC, one axis, rigid [current best], Opaque robots with lights, ASYNC, compass, rigid [current best], Opaque oblivious robots, ASYNC, compass [lower bound]
- Bose et al. (2020b) — Pattern formation: Opaque fat robots with lights, ASYNC, one axis [current best]
- Disser and Schmitt (2019) — Evacuation: Two face-to-face robots, disk [current best]
- Georgiou et al. (2023b) — Evacuation: Disk, k wireless robots, f Byzantine [lower bound], Disk, k face-to-face robots, f Byzantine [lower bound], Disk, k face-to-face robots, f crash faults [lower bound], Disk search, k wireless robots, f crash faults [current best], Disk search, k wireless robots, f crash faults [lower bound], Disk search, k wireless robots, one Byzantine [current best], Disk search, k wireless robots, one Byzantine [lower bound]
- Bagheri et al. (2019) — Evacuation: Triangle, any number of wireless robots at the centroid [current best]
- Gotoh et al. (2021) — Exploration of dynamic graphs by teams: Temporally connected graphs, arbitrary topology [current best], Temporally connected graphs, arbitrary topology [lower bound]
- Léchine and Tixeuil (2019) — Scattering: Randomized, ASYNC, weak multiplicity detection [current best], Randomized, ASYNC, rigid, weak multiplicity detection, no vision needed [current best]
- Emek et al. (2019) — Leader election in programmable matter: Deterministic, no orientation [current best]

- Ilcinkas (2019) — Exploration by oblivious robots: Rings, ASYNC, exclusive perpetual [lower bound]
- Nagahama et al. (2023) — Exploration by luminous robots: Rings, ASYNC, perpetual, $\varphi \geq 2$ [current best], Rings, ASYNC, perpetual, $\varphi \geq 2$ [lower bound], Rings, ASYNC, terminating, $\varphi \geq 2$ [current best], Rings, ASYNC, terminating, $\varphi \geq 2$ [lower bound]
- Pattanayak et al. (2019a) — Chauffeuring a crashed robot: e exits [current best], One exit [current best], Two exits [current best]
- Cicerone et al. (2021b) — Gathering by oblivious robots on graphs: Complete graphs, FSYNC, strong multiplicity detection [current best], Complete graphs, FSYNC, strong multiplicity detection [lower bound]

2020

Energy distribution is posed in Daymude et al. (2021). arXiv 2020; ICDCN 2021.

Broadcast by agents in dynamic graphs is posed in Das et al. (2026b). Posed at OPODIS 2020; the journal version appeared in Discrete Applied Mathematics (2026).

- Kshemkalyani et al. (2020a) — Dispersion: Dynamic graphs (1-interval connected) [lower bound], Dynamic graphs with global communication [current best], Dynamic graphs with global communication [lower bound]
- Das et al. (2023) — Dispersion: Anonymous agents with randomness [current best]
- Molla et al. (2021a) — Dispersion: Weak Byzantine agents on rings [current best], Weak Byzantine agents on rings [lower bound]
- Shibata et al. (2022a) — Uniform deployment on graphs: Dynamic rings, all agents [lower bound], Dynamic rings, all agents but one [current best], Dynamic rings, all agents but one [lower bound]
- Bouchard et al. (2023a) — Treasure hunt: Labelled nodes, locally finite graphs [current best], Labelled nodes, locally finite graphs [lower bound]
- Mondal et al. (2020) — Uniform circle formation: Opaque fat robots, ASYNC, limited visibility, compass [current best]
- Georgiou and Lucier (2023) — Evacuation: Line, weighted evacuation of two wireless robots [current best], Line, weighted evacuation of two wireless robots [lower bound], Line, weighted evacuation of two face-to-face robots [current best], Line, weighted evacuation of two face-to-face robots [lower bound], Line, priority evacuation, queen and one wireless servant [current best], Line, priority evacuation, queen and one face-to-face servant [current best]
- Di Luna et al. (2025b) — Black hole search: Dynamic rings [current best], Dynamic rings [lower bound]
- Dobrev et al. (2020a) — Exploration of dynamic graphs by teams: Connected at every step, any graph [current best], Connected at every step, any graph [lower bound]
- D'Angelo et al. (2020) — Leader election in programmable matter: Silent oblivious particles (SILBOT) [current best], SILBOT with holes [current best], SILBOT with holes [lower bound]
- Daymude et al. (2021) — Energy distribution: Unfair adversary [current best], Unfair adversary [lower bound]
- Das et al. (2026b) — Broadcast by agents in dynamic graphs: Static graph [current best], Dynamic graph, any topology [current best], Dynamic graph, any topology [lower bound], Dynamic rings [current best], Dynamic rings [lower bound], Dynamic complete graphs [current best], Dynamic complete graphs [lower bound], Dynamic bipartite graphs [current best], Dynamic bipartite graphs [lower bound], Dynamic hypercubes [current best], Dynamic hypercubes [lower bound], Dynamic grids [current best], Dynamic grids [lower bound]

2021

Claude (Anthropic) was used in cataloguing the problems.

Partitioning is posed in Bhagat et al. (2021). The fixed-point version (k-EPF) was posed with k-circle formation in 2021; the earlier partitioning of Efrima and Peleg fixes the number of groups instead and is recorded under k-grouping. In the logical view: special case of Pattern formation, which was posed earlier (1999).

- Pattanayak et al. (2021b) — Dispersion: Crash faults [introductory]
- Molla et al. (2021b) — Dispersion: Weak Byzantine agents [current best], Weak Byzantine agents [lower bound], Strong Byzantine agents [introductory]
- Gorain et al. (2022) — Treasure hunt: Treasure hunt guided by pebbles [current best], Treasure hunt guided by pebbles [lower bound]
- Cicerone et al. (2021c) — Pattern formation: ASYNC with common chirality [current best]
- Bramas et al. (2023b) — Gathering: Rendezvous, full lights, a crash, SSYNC [lower bound]
- Nakai et al. (2021) — Gathering: Gathering, full lights, ASYNC, no multiplicity detection [current best]
- Kirkpatrick et al. (2024) — Convergence: Limited visibility, asynchronous [current best]
- Czyzowicz et al. (2021d) — Evacuation: Line, any number of wireless robots [lower bound], Line, a sender and a receiver [current best], Line, a sender and a receiver [lower bound]; Search for a moving target: Line, a sender and a receiver, away, u known [lower bound]
- Czyzowicz et al. (2021e) — Evacuation: Line, $2f + 1$ wireless robots, f crash faults [current best]
- Leonardos et al. (2023) — Evacuation: Disk, k wireless robots, f Byzantine [introductory], Disk, k face-to-face robots, f Byzantine [introductory], Disk, k face-to-face robots, f crash faults [current best]
- Dufoulon et al. (2021) — Leader election in programmable matter: Deterministic, with chirality [current best]
- Feldmann et al. (2022) — Leader election in programmable matter: Stationary, reconfigurable circuits, synchronous, randomized [current best]
- Bramas et al. (2023a) — Exploration by luminous robots: Grids, FSYNC, perpetual, chirality, $\varphi = 1$ [current best], Grids, FSYNC, perpetual, chirality, $\varphi = 1$ [lower bound], Grids, FSYNC, perpetual, chirality, $\varphi = 2$ [current best], Grids, FSYNC, perpetual, chirality, $\varphi = 2$ [lower bound], Grids, ASYNC, perpetual, chirality, $\varphi = 2$ [current best], Grids, ASYNC, perpetual, chirality, $\varphi = 2$ [lower bound], Grids, FSYNC, terminating, chirality, $\varphi = 1$ [lower bound], Grids, FSYNC, terminating, no chirality, $\varphi = 1$ [lower bound], Grids, FSYNC, terminating, chirality, $\varphi = 2$ [lower bound], Grids, FSYNC, terminating, no chirality, $\varphi = 2$ [lower bound], Grids, ASYNC, terminating, chirality, $\varphi = 2$ [lower bound], Grids, ASYNC, terminating, no chirality, $\varphi = 2$ [lower bound]
- Rauch et al. (2021) — Exploration by luminous robots: Grids, FSYNC, perpetual, no chirality, $\varphi = 1$ [current best], Grids, FSYNC, perpetual, no chirality, $\varphi = 1$ [lower bound], Grids, ASYNC, perpetual, no chirality, $\varphi = 2$ [lower bound], Grids, ASYNC, perpetual, no chirality, $\varphi = 3$ [lower bound]
- Nagahama et al. (2022) — Exploration by luminous robots: Grids, FSYNC, terminating, chirality, $\varphi = 1$ [current best], Grids, FSYNC, terminating, no chirality, $\varphi = 1$ [current best], Grids, FSYNC, terminating, chirality, $\varphi = 2$ [current best], Grids, FSYNC, terminating, no chirality, $\varphi = 2$ [current best], Grids, ASYNC, terminating, chirality, $\varphi = 1$ [current best], Grids, ASYNC, terminating, chirality, $\varphi = 1$ [lower bound], Grids, ASYNC, terminating, no chirality, $\varphi = 1$ [current best], Grids, ASYNC, terminating, no chirality, $\varphi = 1$ [lower bound], Grids, ASYNC, terminating, chirality, $\varphi = 2$ [current best], Grids, ASYNC, terminating, no chirality, $\varphi = 2$ [current best]
- Bojko et al. (2022) — Exploration of recolourable graphs: Recolourable nodes, oblivious agent, any graph [lower bound], Recolourable nodes on cacti [lower bound], Recolourable nodes on trees [lower bound], Label-guided: colours fixed by a designer [lower bound]
- Bhagat et al. (2021) — Partitioning: Exactly n/k robots at each fixed point [current best]

2022

Claude (Anthropic) was used in cataloguing the problems.

Alignment is posed in Kedia et al. (2022). APPROX/RANDOM 2022. In the logical view: generalisation of Compression, which was posed earlier (2016).

- Kshemkalyani et al. (2022) — Dispersion: Global communication [introductory], Global communication [lower bound]
- Italiano et al. (2025) — Dispersion: Directed graphs [current best], Directed graphs [lower bound], Directed graphs, 1-hop communication [current best], Directed graphs, 1-hop communication [lower bound]
- Gorain et al. (2024) — Dispersion: Silent agents (no communication) [current best], Silent agents (no communication) [lower bound], Silent agents on rings [lower bound]
- Bhagat and Pelc (2024) — Rendezvous: Labelled agents in infinite trees [current best], Labelled agents in infinite trees [lower bound], Labelled agents in infinite trees, simultaneous start [current best], Labelled agents in infinite trees, simultaneous start [lower bound]
- Castenow et al. (2022) — Gathering: Limited visibility, FSYNC, disoriented [current best], Limited visibility, FSYNC, disoriented [lower bound]; Near-gathering: SSYNC, limited visibility, disoriented [current best], SSYNC, limited visibility, disoriented [lower bound]
- Feletti et al. (2023b) — Uniform circle formation: Opaque robots with lights, ASYNC [introductory], Opaque robots with lights, FSYNC [current best]
- Georgiou et al. (2022) — Evacuation: Two robots with asymmetric communication, disk [current best], Two robots with asymmetric communication, disk [lower bound]
- Georgiou and Jang (2026) — Evacuation: Triangle, two wireless robots starting on the boundary [current best], Triangle, two wireless robots starting on the boundary [lower bound]
- Alsaedi et al. (2023a) — Mutual visibility: Fat robots, FSYNC, rigid [current best], Fat robots, FSYNC, rigid [lower bound]
- Briones et al. (2023) — Leader election in programmable matter: Stationary, asynchronous, no orientation, no holes [current best], Stationary, sequential, no orientation, no holes [current best]
- Darwich et al. (2023) — Exploration by luminous robots: Tori, FSYNC, perpetual, chirality, $\varphi = 1$ [current best], Tori, FSYNC, perpetual, chirality, $\varphi = 1$ [lower bound], Tori, FSYNC, perpetual, no chirality, $\varphi = 1$ [current best], Tori, FSYNC, perpetual, no chirality, $\varphi = 1$ [lower bound], Tori, FSYNC, perpetual, chirality, $\varphi = 2$ [current best], Tori, FSYNC, perpetual, chirality, $\varphi = 2$ [lower bound], Tori, FSYNC, perpetual, no chirality, $\varphi = 2$ [current best], Tori, FSYNC, perpetual, no chirality, $\varphi = 2$ [lower bound]
- Kedia et al. (2022) — Alignment: Markov chain, sequential [current best]
- Coleman et al. (2022) — Search for a moving target: One robot, line, target moving away, d and u known [current best], One robot, line, target moving away, d and u known [lower bound], One robot, line, target moving toward the origin, d and u known [current best], One robot, line, target moving toward the origin, d and u known [lower bound], One robot, line, target moving toward the origin, speed known [current best], One robot, line, target moving toward the origin, speed known [lower bound], One robot, line, target moving away, d known [introductory], One robot, line, target moving away, d known [lower bound], One robot, line, target moving toward the origin, d known [current best], One robot, line, target moving toward the origin, d known [lower bound], One robot, line, target moving away, nothing known [introductory], One robot, line, target moving away, nothing known [lower bound], One robot, line, target moving toward the origin, nothing known [current best], One robot, line, target moving toward the origin, nothing known [lower bound]
- Izumi et al. (2022) — Deciding graph properties by mobile agents: One agent with whiteboards and little memory [current best], One agent with whiteboards and little memory [lower bound]

2023

Minimal dominating set by agents is posed in Chand et al. (2023). Introduced at ALGOSENSORS 2023.

Claude (Anthropic) was used in cataloguing the problems.

Separation is posed in Seike and Yamauchi (2023). Introduced at SSS 2023.

Exploration of recolourable graphs is posed in Böckenhauer et al. (2023). Böckenhauer, Frei, Unger and Wehner posed exploration by an oblivious agent without port numbers that colours nodes and sees its neighbours' colours (SIROCCO 2023; arXiv January 2023), first with each node coloured once and, in a last section, with recolouring. Fixed colours chosen by a designer (2005) and the rotor-router pointer (1996, 2003) are earlier node-side precursors. In the logical view: variant of Graph exploration by a single agent, which was posed earlier (1951).

Scattering of particles is posed in Navarra et al. (2025). Scattering in SILBOT first appeared at AINA 2023 (Navarra, Prencipe, Bonini, Tracolli, "Scattering with programmable matter", pp. 236–247), as the journal version states; that paper was not read and its DOI was not verified, so the journal version (Navarra, Piselli, Prencipe) is cited.

- Cosson et al. (2023) — Collaborative exploration: Trees, asynchronous, complete communication [introductory], Arbitrary graphs, complete communication [current best]
- Pattanayak et al. (2023) — Dispersion: Crash faults [current best]
- Moses Jr. and Redlich (2024) — Dispersion: Strong Byzantine agents [current best]
- Pattanayak and Pelc (2024b) — Treasure hunt: Anonymous graphs with possibly infinite degrees [current best], Anonymous graphs with possibly infinite degrees [lower bound]; Rendezvous: Labelled agents, graphs with infinite degrees [current best], Labelled agents, graphs with infinite degrees [lower bound]
- Miller and Pelc (2025) — Rendezvous: Anonymous agents on paths and rings with labelled nodes [introductory], Anonymous agents on paths and rings with labelled nodes [lower bound]
- Alsaedi et al. (2025b) — Pattern formation: Fat robots with lights, FSYNC, randomized, no agreement [current best]
- Asahiro and Yamashita (2025) — Gathering: Homonymous robots, SSYNC [current best], Homonymous robots, SSYNC [lower bound]
- Behrouz et al. (2025) — Evacuation: Disk, k wireless robots, f Byzantine [current best], Disk, k face-to-face robots, f Byzantine [current best]
- Molla et al. (2023) — Gathering in graphs: Labelled agents [current best]
- Chand et al. (2023) — Minimal dominating set by agents: No knowledge [introductory]
- Seike and Yamauchi (2023) — Separation: ASYNC, one common axis [introductory]
- Navarra and Piselli (2023b) — Shape formation in programmable matter: SILBOT, chirality [lower bound], SILBOT, full orientation [current best]
- Navarra and Piselli (2023a) — Coating: SILBOT, chirality, deterministic [current best], SILBOT, chirality, deterministic [lower bound]
- Böckenhauer et al. (2023) — Exploration of recolourable graphs: Recolourable nodes, oblivious agent, any graph [introductory], Once-colourable nodes, any graph [current best], Once-colourable nodes, any graph [lower bound], Once-colourable nodes, n known [current best], Once-colourable nodes, n known [lower bound], Once-colourable nodes on trees [current best], Once-colourable nodes on trees [lower bound]
- Navarra et al. (2025) — Scattering of particles: SILBOT [lower bound], SILBOT with event-driven activation [current best]
- Baligács et al. (2026a) — Online exploration of weighted graphs: Arbitrary weighted graphs [current best], Graphs excluding a fixed minor [current best]

2024

Leader election by agents is posed in Kshemkalyani et al. (2024). Leader election in the agent-based model, with MST as an application (DISC 2024).

Maximal independent set by agents is posed in Pattanayak et al. (2024). Introduced at ICDCN 2024.

Triangle and butterfly counting by agents is posed in Chand et al. (2026). Agent-based triangle counting (arXiv 2024; IEEE Transactions on Networking 2026).

Minimum spanning tree by agents is posed in Kshemkalyani et al. (2024). MST in the agent-based model appeared together with leader election (DISC 2024).

- Pattanayak and Pelc (2024a) — Graph exploration by a single agent: Memoryless automaton with pebbles [current best], Memoryless automaton with pebbles [lower bound]
- Mondal et al. (2025a) — Dispersion: Dynamic rings [lower bound]
- Banerjee et al. (2024) — Dispersion: Crash faults on unoriented grids [current best], Crash faults on oriented grids [introductory]
- Banerjee et al. (2025) — Dispersion: Crash faults on oriented grids [current best], Weak Byzantine agents on oriented grids [current best]
- Das et al. (2026a) — Dispersion: Silent agents (no communication) [lower bound], Silent agents on rings [current best], Silent agents on rings [lower bound]
- Hahn et al. (2024) — Pattern formation: Limited visibility [current best]
- Otaka et al. (2026) — Gathering: Rendezvous, external light (FCOM), SSYNC [lower bound], Rendezvous, external light (FCOM), ASYNC, non-rigid [lower bound], Rendezvous, external light (FCOM), ASYNC, rigid [lower bound], Rendezvous, internal light (FSTA), SSYNC, non-rigid [lower bound], Rendezvous, internal light (FSTA), SSYNC, rigid [lower bound], Rendezvous, internal light (FSTA), ASYNC [lower bound], Gathering, external light (FCOM), SSYNC, rigid [lower bound], Gathering, internal light (FSTA), SSYNC, non-rigid [current best], Gathering, internal light (FSTA), SSYNC, non-rigid [lower bound]
- Feletti et al. (2026a) — Uniform circle formation: Opaque robots with lights, ASYNC [current best], Opaque robots with lights, ASYNC [lower bound]
- Georgiou and Wang (2025) — Evacuation: Disk, priority evacuation, queen and one servant [current best], Disk, priority evacuation, queen and one servant [lower bound], Disk, weighted evacuation of two wireless robots [current best], Disk, weighted evacuation of two wireless robots [lower bound]
- Kaur et al. (2026b) — Black hole search: Dynamic graphs with whiteboards [current best], Dynamic graphs with whiteboards [lower bound]
- Kshemkalyani et al. (2024) — Leader election by agents: No knowledge [introductory]; Maximal independent set by agents: No knowledge [introductory]; Minimum spanning tree by agents: Dispersed start [introductory], Arbitrary start [introductory]
- Pattanayak et al. (2024) — Maximal independent set by agents: Knowing n and Δ [introductory], Rooted start, no knowledge [current best]
- Kaur et al. (2026c) — Maximal independent set by agents: No knowledge [current best]
- Flocchini et al. (2026) — Universal pattern formation: Sequential schedulers [current best], Sequential schedulers [lower bound]
- Bhagat et al. (2024) — Mutual visibility: Fat robots, ASYNC, rigid, one axis [current best]
- Flocchini et al. (2025) — Separation: ASYNC, one common axis [current best], ASYNC, one common axis [lower bound]

Claude (Anthropic) was used in cataloguing the problems.

- Chand et al. (2026) — Triangle and butterfly counting by agents: Triangle counting [current best]
- Chalopin et al. (2026b) — Leader election in programmable matter: Stationary, common direction, no chirality [current best], Stationary, common direction, no chirality [lower bound]
- Chalopin et al. (2025) — Leader election in programmable matter: Stationary, fair scheduler, no orientation, no holes [current best]
- Ooshita et al. (2024) — Exploration by luminous robots: Rings, FSYNC, perpetual, up to f crashes [current best], Rings, FSYNC, perpetual, up to f crashes [lower bound], Rings, ASYNC, perpetual, up to f crashes, $\varphi = 1$ [current best], Rings, ASYNC, perpetual, up to f crashes, $\varphi = 1$ [lower bound], Rings, ASYNC, perpetual, up to f crashes, $\varphi \geq 2$ [current best], Rings, ASYNC, perpetual, up to f crashes, $\varphi \geq 2$ [lower bound]
- Bramas et al. (2026a) — Exploration by luminous robots: Grids, ASYNC, perpetual, chirality, $\varphi = 1$ (exclusive) [lower bound], Grids, ASYNC, perpetual, chirality, $\varphi = 2$ [lower bound], Grids, ASYNC, perpetual, no chirality, $\varphi = 2$ [current best], Grids, ASYNC, perpetual, no chirality, $\varphi = 2$ [lower bound], Grids, ASYNC, perpetual, no chirality, $\varphi = 3$ [current best]
- Hinnenthal et al. (2025) — Shape formation by hybrid programmable matter: Three-dimensional [current best]
- Coleman et al. (2026a) — Search for a moving target: One robot, line, target moving away, d known [current best], One robot, line, target moving away, d known [lower bound], One robot, line, target moving away, nothing known [current best], One robot, line, target moving away, nothing known [lower bound]
- Bhagat and Pelc (2025) — Collision-free exploration: Anonymous graphs, nothing known [lower bound], Two agents, anonymous graphs, vision of radius 2 [introductory], Two agents, anonymous graphs, vision of radius 1 [lower bound]
- Das et al. (2025a) — Collision-free exploration: Two agents sensing presence, no pebbles [lower bound], Two agents sensing presence, with coloured pebbles [current best], Two agents sensing presence, with coloured pebbles [lower bound], Two agents sensing presence, with coloured pebbles, bipartite graphs [current best], Two agents sensing presence, with coloured pebbles, bipartite graphs [lower bound]

2025

Universal pattern formation is posed in Flocchini et al. (2026). Introduced at SIROCCO 2025. In the logical view: strengthening of Pattern formation, which was posed earlier (1999).

BFS tree by agents is posed in Chand et al. (2025c). BFS without root knowledge in the agent-based model (SSS 2025).

- Cosson and Massoulié (2025) — Collaborative exploration: Trees, asynchronous, complete communication [lower bound], Trees, asynchronous, whiteboards only [current best]
- Kshemkalyani et al. (2025b) — Collaborative exploration: Anonymous graphs with $k = n$ agents [current best]; Dispersion: Arbitrary graph, arbitrary initial configuration [current best], Rooted initial configuration [current best], Global communication [current best], Rings and paths [current best], Trees [current best]
- Pattanayak et al. (2026) — Dispersion: Asynchronous agents [current best]
- Bourreau et al. (2026) — Rendezvous: Anonymous agents on paths and rings with labelled nodes [current best]
- Bonnet et al. (2026a) — Gathering: Self-stabilizing gathering, full lights, SSYNC, strong multiplicity detection [current best]
- Di Luna et al. (2025a) — Black hole search: Byzantine companions, with a map [current best], Byzantine companions, whiteboards, no map [current best]
- Bonnet et al. (2026b) — Black hole search: Eventually emerging black hole, synchronous agents, oriented ring [current best], Eventually emerging black hole, synchronous agents, oriented ring [lower bound], Eventually emerging black hole, asynchronous agents, oriented ring [current best], Eventually emerging black hole, asynchronous agents, oriented ring [lower bound], Eventually emerging black hole, synchronous agents, arbitrary graphs [lower bound]

Claude (Anthropic) was used in cataloguing the problems.

- Kaur et al. (2026a) — Black hole search: Eventually emerging black hole, synchronous agents, arbitrary graphs [current best]
- Devismes et al. (2026b) — Gathering in graphs: Homonymous agents [current best]
- Hirose et al. (2025) — Gathering in graphs: Weakly Byzantine agents [lower bound]
- Chand and Molla (2026) — Leader election by agents: No knowledge [current best]; Minimal dominating set by agents: No knowledge [current best]
- Kshemkalyani et al. (2026) — Leader election by agents: Knowing n and Δ [current best]; Maximal independent set by agents: Knowing n and Δ [current best]
- Feletti et al. (2026c) — Pattern sequences: Luminous robots, sequential scheduler (universal dancing) [current best], Luminous robots, sequential scheduler (universal dancing) [lower bound]
- Pyati et al. (2026) — Separation: Opaque robots, SSYNC [current best]
- Chand et al. (2025b) — Triangle and butterfly counting by agents: Butterfly counting in bipartite graphs [current best]
- Chand et al. (2025c) — Minimum spanning tree by agents: Dispersed start [current best]; BFS tree by agents: Dispersed start, root unknown [current best]
- Kshemkalyani et al. (2025c) — Minimum spanning tree by agents: Arbitrary start [current best]
- Chalopin et al. (2026a) — Leader election in programmable matter: Oblivious, common compass, holes allowed [current best]
- Bar et al. (2026) — Network decontamination: Dynamic graphs, monotone strategies [current best], Dynamic graphs, monotone strategies [lower bound]
- Das et al. (2025c) — Exploration by luminous robots: Triangular grids, FSYNC, perpetual, no chirality, $\varphi = 1$ [current best], Triangular grids, FSYNC, perpetual, no chirality, $\varphi = 1$ [lower bound], Triangular grids, FSYNC, perpetual, chirality, $\varphi = 2$ [current best], Triangular grids, FSYNC, perpetual, chirality, $\varphi = 2$ [lower bound]
- Takahashi et al. (2025) — Exploration of recolourable graphs: Recolourable nodes, oblivious agent, any graph [current best]
- Gehnen et al. (2025) — Online exploration of weighted graphs: Known graph, weight estimates [current best], Known graph, weight estimates [lower bound], Complete graphs, uniform weight estimates [current best], Complete graphs, uniform weight estimates [lower bound]
- Navarra and Piselli (2025) — Gathering by oblivious robots on graphs: Rings, round robin, no multiplicity detection [current best], Rings, round robin, no multiplicity detection [lower bound], Rings, round robin, one robot per node at the start [current best], Rings, round robin, one robot per node at the start [lower bound], Rings, sequential scheduler, no multiplicity detection [lower bound]
- Cicerone et al. (2025d) — Gathering by oblivious robots on graphs: Graphs that are not vertex-transitive, round robin [current best], Graphs that are not vertex-transitive, round robin [lower bound]
- Cicerone et al. (2025b) — Gathering by oblivious robots on graphs: Hypercubes, round robin [current best], Hypercubes, round robin [lower bound], Infinite grid, round robin [current best], Infinite grid, round robin [lower bound], Complete graphs, round robin [lower bound]
- Moses Jr. et al. (2025) — Broadcast by agents in dynamic graphs: Dynamic graph, any topology [lower bound]
- Jawhar and Kranakis (2026b) — Search for a moving target: Line, two face-to-face robots, away, d and u known [current best], Line, two face-to-face robots, away, d and u known [lower bound], Line, two face-to-face robots, toward, d and u known [current best], Line, two face-to-face robots, toward, d and u known [lower bound], Line, two face-to-face robots, away, u known [current best], Line, two face-to-face robots, away, u known [lower bound], Line, two face-to-face robots, toward, u known [current best], Line, two face-to-face robots, toward, u known [lower bound], Line, two face-to-face robots, away, d known [current best], Line, two face-to-face robots, away, d known [lower bound], Line, two face-to-face robots, toward, d known [current best], Line, two face-to-face robots, toward, d known [lower bound], Line, two face-to-face robots, away, nothing known [current best], Line, two face-to-face robots, toward, nothing known [current best], Line, two face-to-face robots, toward, nothing known [lower bound]

Claude (Anthropic) was used in cataloguing the problems.

- Jawhar and Kranakis (2025b) — Search for a moving target: Line, a sender and a receiver, away, u known [current best], Line, a sender and a receiver, toward, u known [current best], Line, a sender and a receiver, away, d known [current best], Line, a sender and a receiver, away, d known [lower bound], Line, a sender and a receiver, toward, d known [current best], Line, a sender and a receiver, toward, d known [lower bound], Line, a sender and a receiver, away, nothing known [current best], Line, a sender and a receiver, toward, nothing known [current best], Line, a sender and a receiver, toward, nothing known [lower bound]

2026

Search with segment visibility is posed in Kranakis (2026). Kranakis introduced segment visibility in a preprint of November 2026 (Kranakis 2026); the earlier uses this note rests on are the author's own account.

- Carnevale (2026) — Graph exploration by a single agent: Temporal graphs, schedule known [lower bound]
- Sudo et al. (2026c) — Graph exploration by a single agent: T-interval-connected graphs, labelled nodes [current best], T-interval-connected graphs, labelled nodes [lower bound]
- Cosson and Massoulié (2026) — Collaborative exploration: Trees, complete communication [current best], Trees, asynchronous, complete communication [current best]
- Mutzari et al. (2026) — Collaborative exploration: Connected team, labelled graphs, shared knowledge [current best]
- Kumar et al. (2026a) — Dispersion: Oriented grids [current best]
- Xin (2026) — Treasure hunt: Anonymous finite graphs [current best]
- Ayoubi et al. (2026) — Treasure hunt: Dynamic graphs, connected at every step [lower bound]
- Wada et al. (2026) — Gathering: Rendezvous, external light (FCOM), ASYNC, rigid [current best], Rendezvous, internal light (FSTA), SSYNC, rigid [current best]
- Ma and Yin (2026) — Linear search (cow path): One randomized robot on m rays [lower bound]
- Shukla et al. (2026a) — Mutual visibility: Lights, mobility faults, SSYNC, non-rigid [current best]
- Warner and Scheideler (2026a) — Leader election in programmable matter: Stationary, randomized, sequential, holes allowed [current best]
- Hiraoka et al. (2026) — Exploration of recolourable graphs: Recolourable nodes, oblivious agent, any graph [lower bound], Recolourable nodes on cacti [current best], Recolourable nodes on cacti [lower bound], Recolourable nodes on trees [current best], Recolourable nodes on trees [lower bound]
- Kranakis (2026) — Search with segment visibility: Two robots, unknown distance [current best], Two robots, unknown distance [lower bound], Two robots, known distance [current best], Two robots, known distance [lower bound], $k \geq 3$ robots, unknown distance [current best], $k \geq 3$ robots, unknown distance [lower bound], $k \geq 3$ robots, known distance [current best], $k \geq 3$ robots, known distance [lower bound]
- Baligács (2026) — Online exploration of weighted graphs: Arbitrary weighted graphs [lower bound], Graphs excluding a fixed minor [lower bound], Planar graphs [lower bound]
- Baligács et al. (2026b) — Online exploration of weighted graphs: Weighted cycles, randomized agent [current best], Weighted cycles, randomized agent [lower bound]
- Moses Jr. et al. (2026) — Broadcast by agents in dynamic graphs: Static graph [lower bound], Dynamic graph, any topology [lower bound]
- Jawhar and Kranakis (2026a) — Search for a moving target: m rays, two face-to-face robots, away, u known [current best], m rays, two face-to-face robots, away, d known [current best], m rays, two face-to-face robots, away, nothing known [current best], m rays, a sender and a receiver, away, u known [current best], m rays, a sender and a receiver, away, u known [lower bound], m rays, a sender and a receiver, away, d known [current best], m rays, a sender and a receiver, away, d known [lower bound], m rays, a sender and a receiver, away, nothing known [current best]

Claude (Anthropic) was used in cataloguing the problems.

- Papin (2026) — Collision-free exploration: Two agents, anonymous graphs, vision of radius 2 [current best]

Claude (Anthropic) was used in cataloguing the problems.

Bibliography

Sebastian Abshoff, Andreas Cord-Landwehr, Matthias Fischer, Daniel Jung, Friedhelm Meyer auf der Heide. Gathering a Closed Chain of Robots on a Grid. *IPDPS 2016, IEEE, pp. 689–699*, 2016. doi:10.1109/ipdps.2016.51, arXiv:1510.05454 (pdf). Other versions: *arXiv preprint 1501.04877*, 2015, as "Towards Gathering Robots with Limited View in Linear Time: The Closed Chain Case" (link). First appeared in 2015. Cited on p. 142.

Sumi Acharjee, Konstantinos Georgiou, Somnath Kundu, Akshaya Srinivasan. Lower Bounds for Shoreline Searching With 2 or More Robots. *OPODIS 2019, LIPIcs 153, 26:1–26:11*, 2019. doi:10.4230/LIPIcs.OPODIS.2019.26, pdf, arXiv:2001.04311 (pdf). Cited on pp. 507, 579, and 580.

Duncan Adamson. Exploring Temporal Graphs with Frequent and Regular Edges. *arXiv preprint*, 2025. arXiv:2505.14046 (pdf). Other versions: *SSRN preprint, September 2025* (doi:10.2139/ssrn.5450373); *SSRN preprint, November 2025* (doi:10.2139/ssrn.5783511). Preprint, not peer reviewed. Cited on p. 85.

Duncan Adamson. Exploring Word-Representable Temporal Graphs. *SAND 2025 (brief announcement), LIPIcs 330, 22:1–22:6*, 2025. doi:10.4230/LIPIcs.SAND.2025.22, pdf, arXiv:2502.07496 (pdf). Read: arXiv v1 (2025), the full version; theorem numbers refer to it. Cited on p. 85.

Duncan Adamson, Paul G. Spirakis. Brief Announcement: Exploration of Always S-Connected Temporal Graphs. *SAND 2026 (brief announcement), LIPIcs 373, 20:1–20:6*, 2026. doi:10.4230/LIPIcs.SAND.2026.20, pdf, link, arXiv:2602.19657 (pdf). Read: arXiv v2 (2026), the full version, titled "Exploration of Always *S*-Connected Temporal Graphs". Cited on pp. 7, 104, and 306.

Peyman Afshani, Mark de Berg, Kevin Buchin, Jie Gao, Maarten Löffler, Amir Nayyeri, Benjamin Raichel, Rik Sarkar, Haotian Wang, Hao-Tsung Yang. Approximation Algorithms for Multi-Robot Patrol-Scheduling with Min-Max Latency. *WAFR 2020 (Algorithmic Foundations of Robotics XIV), Springer Proceedings in Advanced Robotics, pp. 107–123*, 2021. doi:10.1007/978-3-030-66723-8_7, arXiv:2005.02530 (pdf). First appeared in 2020. Read: arXiv v3. Cited on p. 568.

Ankush Agarwalla, John Augustine, William K. Moses Jr., Madhav Sankar K., Arvind Krishna Sridhar. Deterministic Dispersion of Mobile Robots in Dynamic Rings. *ICDCN 2018*, 2018. doi:10.1145/3154273.3154294, arXiv:1707.06391 (pdf). First appeared in 2017. Read: arXiv v3 (2017). Cited on pp. 7, 10, 288, 649, and 701.

Chrysovalandis Agathangelou, Chryssis Georgiou, Marios Mavronicolas. A distributed algorithm for gathering many fat mobile robots in the plane. *PODC 2013, pp. 250–259*, 2013. doi:10.1145/2484239.2484266, arXiv:1209.3904 (pdf). First appeared in 2012. Read: arXiv v1 (2012). Cited on pp. 18, 20, 25, 458, 459, 652, and 697.

Noa Agmon, David Peleg. Fault-Tolerant Gathering Algorithms for Autonomous Mobile Robots. *SIAM Journal on Computing 36(1):56–82*, 2006. doi:10.1137/050645221. Cited on pp. 18, 25, 64, 74, 75, 78, 432, 444, 451, 453, 454, 642, 652, and 694.

Hugo A. Akitaya, Erik D. Demaine, Andrei Gonczi, Dylan H. Hendrickson, Adam Hesterberg, Matias Korman, Oliver Korten, Jayson Lynch, Irene Parada, Vera Sacristán. Characterizing Universal Reconfigurability of Modular Pivoting Robots. *SoCG 2021, LIPIcs 189, 10:1–10:20*, 2021. doi:10.4230/LIPIcs.SOCG.2021.10, arXiv:2012.07556 (pdf). First appeared in 2020. Cited on p. 636.

Hugo A. Akitaya, Esther M. Arkin, Mirela Damian, Erik D. Demaine, Vida Dujmović, Robin Flatland, Matias Korman, Belen Palop, Irene Parada, André van Renssen, Vera Sacristán. Universal Reconfiguration of Facet-Connected Modular Robots by Pivots: The O(1) Musketeers. *Algorithmica 83:1316–1351*, 2021. doi:10.1007/s00453-020-00784-6. Read: arXiv v2 (2019). Cited on pp. 636 and 639.

Hugo A. Akitaya, Frederick Stock. Reconfiguration of 3D Pivoting Modular Robots. *arXiv preprint*, 2023. arXiv:2304.09990 (pdf). Preprint, not peer reviewed. Cited on p. 637.

Susanne Albers, Monika R. Henzinger. Exploring unknown environments. *SIAM Journal on Computing 29(4):1164–1188*, 2000. doi:10.1137/S009753979732428X. Other versions: *STOC 1997, pp. 416–425* (doi:10.1145/258533.258630). First appeared in 1997. Cited on pp. 11, 15, 102, and 103.

Romas Aleliunas, Richard M. Karp, Richard J. Lipton, László Lovász, Charles Rackoff. Random walks, universal traversal sequences, and the complexity of maze problems. *FOCS 1979, pp. 218–223*, 1979. doi:10.1109/SFCS.1979.34. Cited on pp. 12, 57, 84, 85, 88, 97, 98, 211, and 691.

Aisha Aljohani, Gokarna Sharma. Complete Visibility for Mobile Robots with Lights Tolerating Faults. *International Journal of Networking and Computing 8(1):32–52*, 2018. doi:10.15803/ijnc.8.1_32. Other versions: *IPDPS Workshops 2017 (APDCM), pp. 834–841*, as "Complete Visibility for Mobile Agents with Lights Tolerating a Faulty Agent" (doi:10.1109/ipdpsw.2017.145). First appeared in 2017. Cited on pp. 75, 421, 422, 423, 657, and 702.

Abdullah Almethen, Othon Michail, Igor Potapov. Pushing lines helps: Efficient universal centralised transformations for programmable matter. *Theoretical Computer Science 830–831:43–59*, 2020. doi:10.1016/j.tcs.2020.04.026, arXiv:1904.12777 (pdf). Other versions: *ALGOSENSORS 2019, LNCS, pp. 41–59* (doi:10.1007/978-3-030-34405-4_3). First appeared in 2019. Read: arXiv v1 (2019). Cited on pp. 636, 637, 639, and 664.

Steve Alpern, Shmuel Gal. The Theory of Search Games and Rendezvous. *International Series in Operations Research & Management Science 55, Kluwer Academic Publishers*, 2003. doi:10.1007/b100809. Cited on pp. 544, 545, 546, 558, 560, and 693.

Rusul J. Alsaedi, Joachim Gudmundsson, André van Renssen. The Mutual Visibility Problem for Fat Robots. *WADS 2023, LNCS 14079, pp. 15–28*, 2023. doi:10.1007/978-3-031-38906-1_2, arXiv:2206.14423 (pdf). First appeared in 2022. Read: arXiv v3 (2026). Cited on pp. 424, 657, and 706.

Rusul J. Alsaedi, Joachim Gudmundsson, André van Renssen. Shortest Paths of Mutually Visible Robots. *arXiv preprint*, 2023. arXiv:2309.16901 (pdf). Read: arXiv v2 (2025). Preprint, not peer reviewed. Cited on p. 414.

Rusul J. Alsaedi, Joachim Gudmundsson, André van Renssen. Pattern formation for fat robots with memory. *Computational Geometry 129:102189*, 2025. doi:10.1016/j.comgeo.2025.102189, arXiv:2309.14649 (pdf). First appeared in 2023. Cited on p. 383.

Rusul J. Alsaedi, Joachim Gudmundsson, André van Renssen. Pattern formation for fat robots with lights. *Computational Geometry 128:102161*, 2025. doi:10.1016/j.comgeo.2024.102161, arXiv:2306.14440 (pdf). First appeared in 2023. Read: arXiv v2 (2025). Cited on pp. 392 and 707.

Karine Altisen, Anaïs Durand, Pascal Lafourcade, Oussama Nahnah. Synthesizing Algorithms to Avoid an Obstacle with a Swarm of Robots. *ICDCS 2026, IEEE, pp. 556–566*, 2026. doi:10.1109/2575-8411.2026.00059, link. Read: HAL hal-05680038v1. Cited on p. 178.

Christoph Ambühl, Leszek Gąsieniec, Andrzej Pelc, Tomasz Radzik, Xiaohui Zhang. Tree exploration with logarithmic memory. *ACM Transactions on Algorithms 7(2), article 17*, 2011. doi:10.1145/1921659.1921663. Other versions: *SODA 2007*. First appeared in 2007. Cited on pp. 57, 79, 85, 95, 98, 99, 641, and 695.

Michael Amir, Alfred M. Bruckstein. Fast Uniform Dispersion of a Crash-prone Swarm. *Robotics: Science and Systems XVI (RSS 2020)*, 2020. doi:10.15607/RSS.2020.XVI.017, arXiv:1907.00956 (pdf). First appeared in 2019. Read: arXiv v3 (2020). Cited on p. 261.

Michael Amir, Dmitry Rabinovich, Alfred M. Bruckstein. Patrolling Grids with a Bit of Memory. *arXiv preprint*, 2023. arXiv:2307.09214 (pdf). Read: arXiv v2. Preprint, not peer reviewed. Cited on p. 568.

Julian Anaya, Jérémie Chalopin, Jurek Czyzowicz, Arnaud Labourel, Andrzej Pelc, Yann Vaxès. Convergecast and Broadcast by Power-Aware Mobile Agents. *Algorithmica 74(1):117–155*, 2016. doi:10.1007/s00453-014-9939-8, arXiv:1603.04234 (pdf). Other versions: *DISC 2012, LNCS, pp. 46–60*, as "Collecting Information by Power-Aware Mobile Agents" (doi:10.1007/978-3-642-33651-5_4). First appeared in 2012. Read: arXiv v1 (2016). Cited on pp. 10, 331, 332, 333, 334, 337, 338, 339, 667, and 698.

Hideki Ando, Yoshinobu Oasa, Ichiro Suzuki, Masafumi Yamashita. Distributed memoryless point convergence algorithm for mobile robots with limited visibility. *IEEE Transactions on Robotics and Automation 15(5):818–828*, 1999. doi:10.1109/70.795787. Cited on pp. 18, 24, 63, 467, 469, 471, 472, and 692.

Spyros Angelopoulos, Konstantinos Panagiotou. Optimal strategies for weighted ray search. *arXiv preprint*, 2017. arXiv:1704.03777 (pdf). Read: arXiv v2 (2021). Preprint, not peer reviewed. Cited on p. 498.

Maria-Virginia Aponte, Mathis Bouverot-Dupuis, Quentin Bramas, Pierre Courtieu, Lionel Rieg, Xavier Urbain. Formal Certification of async Protocols: The Case of Gathering in $\mathbb{R}^2$ Using Weber Points. *SIROCCO 2026, LNCS, pp. 34–52*, 2026. doi:10.1007/978-3-032-26465-7_3, pdf, link. Cited on p. 433.

Emmanuel Arrighi, Fedor V. Fomin, Petr Golovach, Petra Wolf. Kernelizing Temporal Exploration Problems. *arXiv preprint*, 2023. arXiv:2302.10110 (pdf). Preprint, not peer reviewed. Cited on p. 87.

Marta Andrés Arroyo, Sarah Cannon, Joshua J. Daymude, Dana Randall, Andréa W. Richa. A stochastic approach to shortcut bridging in programmable matter. *Natural Computing 17:723–741*, 2018. doi:10.1007/s11047-018-9714-x, arXiv:1709.02425 (pdf). Other versions: *DNA 2017, LNCS, pp. 122–138* (doi:10.1007/978-3-319-66799-7_9). First appeared in 2017. Read: arXiv v2 (2018). Cited on pp. 46, 70, 610, 617, 618, 619, 701, and 702.

Matthias Artmann, Andreas Padalkin, Christian Scheideler. On the Shape Containment Problem Within the Amoebot Model with Reconfigurable Circuits. *DISC 2025, LIPIcs 356, 7:1–7:22*, 2025. doi:10.4230/LIPIcs.DISC.2025.7, pdf, link, arXiv:2501.16892 (pdf). Read: arXiv v1 (2025). Cited on p. 600.

Yuichi Asahiro, Ichiro Suzuki, Masafumi Yamashita. Monotonic self-stabilization and its application to robust and adaptive pattern formation. *Theoretical Computer Science 934:21–46*, 2022. doi:10.1016/j.tcs.2022.08.011. Cited on p. 383.

Yuichi Asahiro, Masafumi Yamashita. Compatibility of Convergence Algorithms for Autonomous Mobile Robots (Extended Abstract). *SIROCCO 2023, LNCS, pp. 149–164*, 2023. doi:10.1007/978-3-031-32733-9_8, arXiv:2301.10949 (pdf). Read: arXiv v7 (2023). Cited on p. 469.

Yuichi Asahiro, Masafumi Yamashita. Minimum algorithm sizes for the gathering and related problems of autonomous mobile robots. *Theoretical Computer Science 1051:115397*, 2025. doi:10.1016/j.tcs.2025.115397, arXiv:2304.02212 (pdf). Other versions: *SSS 2023*, as "Minimum Algorithm Sizes for Self-stabilizing Gathering and Related Problems of Autonomous Mobile Robots (Extended Abstract)" (doi:10.1007/978-3-031-44274-2_23). First appeared in 2023. Read: arXiv v4 (2023). Cited on pp. 18, 26, 54, 443, 445, 651, 677, and 707.

John Augustine, William K. Moses Jr. Dispersion of Mobile Robots: A Study of Memory-Time Trade-offs. *ICDCN 2018*, 2018. doi:10.1145/3154273.3154293, arXiv:1707.05629 (pdf). First appeared in 2017. Read: arXiv v4 (2018). Cited on pp. 6, 7, 11, 13, 15, 16, 50, 53, 59, 60, 78, 247, 248, 253, 257, 258, 259, 260, 261, 265, 266, 267, 268, 273, 275, 276, 277, 278, 279, 280, 641, 642, 647, 701, and 702.

Kamran Ayoubi, Bernard Mans, Lata Narayanan. Online Treasure Hunt in Vertex-Permuted Dynamic Rings. *arXiv preprint*, 2026. arXiv:2609.11013 (pdf). Preprint, not peer reviewed. Cited on pp. 109, 115, 116, 307, 650, and 711.

Mohamed Anouar Baaziz, Andrzej Pelc. Asynchronous Rendezvous of Anonymous Deterministic Mobile Automata in the Plane. *SAND 2026, LIPIcs 373, 8:1–8:14*, 2026. doi:10.4230/LIPIcs.SAND.2026.8, pdf, link. Cited on p. 119.

Mohamed Anouar Baaziz, Andrzej Pelc. Exploration of convex terrains by a deterministic automaton with pebbles. *Theoretical Computer Science 1070:115844*, 2026. doi:10.1016/j.tcs.2026.115844. Other versions: *SIROCCO 2025, LNCS, pp. 74–91*, as "Exploration of Convex Terrains by a Deterministic Automaton with Pebbles" (doi:10.1007/978-3-031-91736-3_5); *SSRN preprint*, 2025 (doi:10.2139/ssrn.5380470). First appeared in 2025. Cited on p. 87.

Sahar Badri, Serafino Cicerone, Alessia Di Fonso, Gabriele Di Stefano. An optimal algorithm for geodesic mutual visibility on hexagonal grids. *arXiv preprint*, 2024. arXiv:2405.13615 (pdf). Read: arXiv v2. Preprint, not peer reviewed. Cited on p. 414.

Ricardo A. Baeza-Yates, Joseph C. Culberson, Gregory J. E. Rawlins. Searching in the Plane. *Information and Computation 106(2):234–252*, 1993. doi:10.1006/inco.1993.1054. Cited on pp. 38, 41, 51, 72, 78, 497, 498, 499, 501, 504, 576, and 692.

Ricardo A. Baeza-Yates, René Schott. Parallel searching in the plane. *Computational Geometry 5(3):143–154*, 1995. doi:10.1016/0925-7721(95)00003-R. Cited on pp. 533, 535, and 692.

Iman Bagheri, Lata Narayanan, Jaroslav Opatrny. Evacuation of Equilateral Triangles by Mobile Agents of Limited Communication Range. *ALGOSENSORS 2019, LNCS, pp. 3–22*, 2019. doi:10.1007/978-3-030-34405-4_1, arXiv:1908.07315 (pdf). Read: arXiv v1 (2019). Cited on pp. 536, 537, and 703.

Samuel Baguley, Andreas Göbel, Nicolas Klodt, George Skretas, John Sylvester, Viktor Zamaraev. Temporal Exploration of Random Spanning Tree Models. *SODA 2026, pp. 2876–2887*, 2026. doi:10.1137/1.9781611978971.106, arXiv:2508.03361 (pdf). First appeared in 2025. Read: arXiv v1 (2025). Cited on pp. 86 and 105.

Thibaut Balabonski, Robin Pelle, Lionel Rieg, Sébastien Tixeuil. A foundational framework for certified impossibility results with mobile robots on graphs. *ICDCN 2018, 5:1–5:10*, 2018. doi:10.1145/3154273.3154321. Cited on p. 166.

Thibaut Balabonski, Amélie Delga, Lionel Rieg, Sébastien Tixeuil, Xavier Urbain. Synchronous Gathering without Multiplicity Detection: a Certified Algorithm. *Theory of Computing Systems 63(2):200–218*, 2018. doi:10.1007/s00224-017-9828-z. First appeared in 2016. Read: the HAL version. Cited on pp. 449 and 700.

Thibaut Balabonski, Pierre Courtieu, Robin Pelle, Lionel Rieg, Sébastien Tixeuil, Xavier Urbain. Continuous vs. Discrete Asynchronous Moves: A Certified Approach for Mobile Robots. *NETYS 2019, LNCS, pp. 93–109*, 2019.

doi:10.1007/978-3-030-31277-0_7, pdf, link. Other versions: *SSS 2018 (brief announcement), LNCS, pp. 404–408*, as "Brief Announcement Continuous vs. Discrete Asynchronous Moves: A Certified Approach for Mobile Robots" (doi:10.1007/978-3-030-03232-6_29). First appeared in 2018. Cited on p. 67.

Thibaut Balabonski, Pierre Courtieu, Robin Pelle, Lionel Rieg, Sébastien Tixeuil, Xavier Urbain. Computer Aided Formal Design of Swarm Robotics Algorithms. *SSS 2021 (short paper), LNCS, pp. 469–473*, 2021. doi:10.1007/978-3-030-91081-5_31, arXiv:2101.06966 (pdf). Read: arXiv v1 (2021), the full version. Cited on p. 67.

B. Balamohan, S. Dobrev, P. Flocchini, N. Santoro. Exploring an unknown dangerous graph with a constant number of tokens. *Theoretical Computer Science 610:169–181*, 2016. doi:10.1016/j.tcs.2014.07.013. Other versions: *SIROCCO 2012*. First appeared in 2012. Cited on pp. 74, 79, 291, 293, 299, and 697.

Stefan Balev, Éric Sanlaville, Antoine Toullalan. The Shortest Temporal Exploration Problem. *arXiv preprint*, 2025. arXiv:2504.06652 (pdf). Other versions: *SSRN preprint*, 2026 (doi:10.2139/ssrn.6184193). Preprint, not peer reviewed. Cited on p. 86.

Júlia Baligács, Yann Disser, Irene Heinrich, Pascal Schweitzer. Exploration of graphs with excluded minors. *Journal of Computer and System Sciences 156:103725*, 2026. doi:10.1016/j.jcss.2025.103725, link, arXiv:2308.06823 (pdf). Other versions: *ESA 2023, LIPIcs 274, 11:1–11:15*, as "Exploration of Graphs with Excluded Minors" (doi:10.4230/LIPIcs.ESA.2023.11, pdf). First appeared in 2023. Read: arXiv v1 (2023). Cited on pp. 222, 224, 225, 226, 227, 228, 230, 231, 664, 691, and 707.

Júlia Baligács. A lower bound of 4 for online graph exploration. *arXiv preprint*, 2026. arXiv:2607.15113 (pdf). Preprint, not peer reviewed. Cited on pp. 222, 224, 225, 227, 228, 230, 664, 691, and 711.

Júlia Baligács, Jan Hązła, Lena Volk. Randomization Helps in Online Graph Exploration: Breaking the Deterministic Lower Bound on Cycles. *arXiv preprint*, 2026. arXiv:2607.11203 (pdf). Preprint, not peer reviewed. Cited on pp. 223, 225, 228, 229, 231, 232, 664, and 711.

Evangelos Bampas, Leszek Gąsieniec, Nicolas Hanusse, David Ilcinkas, Ralf Klasing, Adrian Kosowski, Tomasz Radzik. Robustness of the Rotor–Router Mechanism. *Algorithmica 78(3):869–895*, 2017. doi:10.1007/s00453-016-0179-y, link. Other versions: *DISC 2009, LNCS, pp. 423–435*, as "Euler Tour Lock-In Problem in the Rotor-Router Model" (doi:10.1007/978-3-642-04355-0_44); *OPODIS 2009, LNCS 5923, pp. 345–358*, as "Robustness of the Rotor-router Mechanism". First appeared in 2009. Read: DISC 2009 version (section and theorem numbers in notes refer to it), and the journal version (accepted manuscript, King's College London repository): there the cases P-all and $A()P(\pi)$ open Section 3, and Theorems 3 and 4 of DISC are Theorems 4 and 5. Cited on pp. 211, 219, 220, 662, and 696.

Evangelos Bampas, Jérémie Chalopin, Shantanu Das, Jan Hackfeld, Christina Karousatou. Maximal Exploration of Trees with Energy-Constrained Agents. *arXiv preprint*, 2018. arXiv:1802.06636 (pdf). Preprint, not peer reviewed. Cited on pp. 247, 248, and 334.

Evangelos Bampas, David Ilcinkas. On mobile agent verifiable problems. *Information and Computation 260:51–71*, 2018. doi:10.1016/j.ic.2018.03.003, pdf, link. Other versions: *LATIN 2016, LNCS 9644, pp. 123–137* (doi:10.1007/978-3-662-49529-2_10); *AlgoTel 2015 (in French)*, as "Problèmes vérifiables par agents mobiles" (link). First appeared in 2015. Read: author version on HAL (hal-01820407, 2018). Cited on pp. 366, 369, 370, 372, 373, 374, 375, 667, 696, and 700.

Evangelos Bampas, Lélia Blin, Jurek Czyzowicz, David Ilcinkas, Arnaud Labourel, Maria Potop-Butucaru, Sébastien Tixeuil. On asynchronous rendezvous in general graphs. *Theoretical Computer Science 753:80–90*, 2019. doi:10.1016/j.tcs.2018.06.045. First appeared in 2018. Cited on p. 119.

Evangelos Bampas, Jurek Czyzowicz, Leszek Gąsieniec, David Ilcinkas, Ralf Klasing, Tomasz Kociumaka, Dominik Pająk. Linear Search by a Pair of Distinct-Speed Robots. *Algorithmica 81(1):317–342*, 2019. doi:10.1007/s00453-018-0447-0, pdf. Other versions: *SIROCCO 2016* (doi:10.1007/978-3-319-48314-6_13, pdf). First appeared in 2016. Cited on pp. 39, 40, 41, 52, 63, 72, 498, 499, 500, 503, 504, 532, 533, 534, 535, and 700.

Alberto Bandettini, Fabio Luporini, Giovanni Viglietta. A Survey on Open Problems for Mobile Robots. *arXiv preprint*, 2011. link, arXiv:1111.2259 (pdf). Preprint, not peer reviewed. Cited on pp. 18, 20, 23, 24, 65, and 434.

Rik Banerjee, Manish Kumar, Anisur Rahaman Molla. Optimizing Robot Dispersion on Unoriented Grids: With and Without Fault Tolerance. *ALGOWIN 2024 (LNCS 15026)*, 2024. doi:10.1007/978-3-031-74580-5_3, arXiv:2405.02002 (pdf). Read: arXiv v1 (2024). Cited on pp. 261, 271, 272, 273, 274, 275, 647, and 708.

Rik Banerjee, Manish Kumar, Anisur Rahaman Molla. Optimal Fault-Tolerant Dispersion on Oriented Grids. *ICDCN 2025*, 2025. doi:10.1145/3700838.3700842, arXiv:2410.17813 (pdf). First appeared in 2024. Read: arXiv v2 (2024). Cited on pp. 75, 261, 271, 272, 273, 274, 646, 647, and 708.

Nikhil Bansal, John Kuszmaul, William Kuszmaul. A nearly tight lower bound for the d-dimensional cow-path

problem. *Information Processing Letters 182:106389*, 2023. doi:10.1016/j.ipl.2023.106389, arXiv:2209.08427 (pdf). First appeared in 2022. Read: arXiv v1 (2022). Cited on p. 498.

Rajashree Bar, Daibik Barik, Adri Bhattacharya, Partha Sarathi Mandal. Monotone Decontamination of Arbitrary Dynamic Graphs with Mobile Agents. *CALDAM 2026, LNCS, pp. 29–42*, 2026. doi:10.1007/978-3-032-17156-6_3, arXiv:2511.18315 (pdf). First appeared in 2025. Read: arXiv v1 (2025). Cited on pp. 318, 322, 323, 660, and 710.

Ariel Barel, Thomas Dagès, Rotem Manor, Alfred M. Bruckstein. Probabilistic Gathering of Agents with Simple Sensors. *SIAM Journal on Applied Mathematics 81(2):620–640*, 2021. doi:10.1137/20M133333X, arXiv:1902.00294 (pdf). First appeared in 2019. Read: arXiv v2 (2020). Cited on p. 433.

Lali Barrière, Paola Flocchini, Pierre Fraigniaud, Nicola Santoro. Capture of an intruder by mobile agents. *SPAA 2002, pp. 200–209*, 2002. doi:10.1145/564870.564906. Cited on pp. 316, 317, 318, 319, 321, and 693.

Jad Bassil, Benoît Piranda, Abdallah Makhoul, Julien Bourgeois. RePoSt: Distributed Self-Reconfiguration Algorithm for Modular Robots Based on Porous Structure. *IROS 2022, IEEE, pp. 12651–12658*, 2022. doi:10.1109/iros47612.2022.9981212. Cited on p. 637.

Jad Bassil, Benoît Piranda, Abdallah Makhoul, Julien Bourgeois. ASAPs: asynchronous hybrid self-reconfiguration algorithm for porous modular robotic structures. *Autonomous Robots 48(7), article 16*, 2024. doi:10.1007/s10514-024-10171-7. First appeared in 2023. Cited on p. 637.

Paul Bastide, Carla Groenland, Lukas Michel, Clément Rambaud, Paulina Smolarova. Improved exploration of temporal graphs. *arXiv preprint*, 2025. arXiv:2511.22604 (pdf). Read: arXiv v2 (19 October 2026). Preprint, not peer reviewed. Cited on pp. 86, 104, 105, and 644.

Rida A. Bazzi, Joseph L. Briones. Stationary and Deterministic Leader Election in Self-organizing Particle Systems. *SSS 2019, LNCS, pp. 22–37*, 2019. doi:10.1007/978-3-030-34992-9_3. Other versions: *SSS 2018 brief announcement, LNCS, pp. 381–386*, as "Brief Announcement: Deterministic Leader Election in Self-organizing Particle Systems" (doi:10.1007/978-3-030-03232-6_25). First appeared in 2018. Cited on pp. 45, 70, 585, 586, 593, 594, 658, and 703.

Anatole Beck. On the linear search problem. *Israel Journal of Mathematics 2(4):221–228*, 1964. doi:10.1007/BF02759737. Cited on pp. 38, 41, 51, 72, 496, 497, 498, 499, 501, and 691.

Pourandokht Behrouz, Orestis Konstantinidis, Nikos Leonardos, Aris Pagourtzis, Ioannis Papaioannou, Marianna Spyrakou. Byzantine fault-tolerant protocols for (n,f)-evacuation from a circle. *Theoretical Computer Science 1038:115185*, 2025. doi:10.1016/j.tcs.2025.115185. Other versions: *ALGOWIN 2023, LNCS, pp. 87–100* (doi:10.1007/978-3-031-48882-5_7). First appeared in 2023. Cited on pp. 511, 523, 525, 526, 528, 654, and 707.

Thomas Bellitto, Cyril Conchon-Kerjan, Bruno Escoffier. Restless Exploration of Periodic Temporal Graphs. *SAND 2023, LIPIcs 257, 13:1–13:15*, 2023. doi:10.4230/LIPIcs.SAND.2023.13. Cited on p. 87.

Richard Bellman. An Optimal Search. *SIAM Review 5(3):274*, 1963. doi:10.1137/1005070. Cited on pp. 38, 41, and 499.

Michael A. Bender, Donna K. Slonim. The power of team exploration: two robots can learn unlabeled directed graphs. *FOCS 1994, pp. 75–85*, 1994. doi:10.1109/SFCS.1994.365703, link. Cited on pp. 87 and 101.

Michael A. Bender, Antonio Fernández, Dana Ron, Amit Sahai, Salil Vadhan. The power of a pebble: exploring and mapping directed graphs. *Information and Computation 176(1):1–21*, 2002. doi:10.1006/inco.2001.3081. Other versions: *STOC 1998* (doi:10.1145/276698.276759). First appeared in 1998. Cited on pp. 6, 11, 15, 58, 89, 101, 102, 159, 162, 644, and 692.

Subhash Bhagat, Sruti Gan Chaudhuri, Krishnendu Mukhopadhyaya. Fault-tolerant gathering of asynchronous oblivious mobile robots under one-axis agreement. *Journal of Discrete Algorithms 36:50–62*, 2016. doi:10.1016/j.jda.2015.10.005. Other versions: *WALCOM 2015, LNCS 8973, pp. 149–160*, as "Fault-Tolerant Gathering of Asynchronous Oblivious Mobile Robots under One-Axis Agreement" (doi:10.1007/978-3-319-15612-5_14). First appeared in 2015. Cited on pp. 454 and 699.

Subhash Bhagat, Sruti Gan Chaudhuri, Krishnendu Mukhopadhyaya. Formation of General Position by Asynchronous Mobile Robots Under One-Axis Agreement. *ISAAC 2016, LNCS, pp. 80–91*, 2016. doi:10.1007/978-3-319-30139-6_7, arXiv:1408.2072 (pdf). First appeared in 2014. Cited on p. 414.

Subhash Bhagat, Krishnendu Mukhopadhyaya. Mutual Visibility by Robots with Persistent Memory. *FAW 2019, LNCS, pp. 144–155*, 2019. doi:10.1007/978-3-030-18126-0_13, arXiv:1708.02825 (pdf). First appeared in 2017. Read: arXiv v1 (2017). Cited on pp. 418, 657, and 702.

Subhash Bhagat, Bibhuti Das, Abhinav Chakraborty, Krishnendu Mukhopadhyaya. k-Circle Formation and k-epf by Asynchronous Robots. *Algorithms 14(2):62*, 2021. doi:10.3390/a14020062. Cited on pp. 19, 464, 465, 466, 467, and 705.

Subhash Bhagat, Andrzej Pelc. How to Meet at a Node of Any Connected Graph. *DISC 2022, LIPIcs 246, 11:1–11:16*, 2022. doi:10.4230/LIPIcs.DISC.2022.11, pdf, link. Cited on p. 121.

Subhash Bhagat, Abhinav Chakraborty, Bibhuti Das, Krishnendu Mukhopadhyaya. Gathering over Meeting Nodes in Infinite Grid. *Fundamenta Informaticae 187(1):1–30*, 2022. doi:10.3233/FI-222128, link, arXiv:2112.06506 (pdf). Other versions: *CALDAM 2020, LNCS, pp. 318–330* (doi:10.1007/978-3-030-39219-2_26). First appeared in 2020. Cited on p. 142.

Subhash Bhagat, Andrzej Pelc. Deterministic rendezvous in infinite trees. *Theoretical Computer Science 984:114313*, 2024. doi:10.1016/j.tcs.2023.114313, arXiv:2203.05160 (pdf). First appeared in 2022. Cited on pp. 128, 129, 650, and 706.

Subhash Bhagat, Paulomi Dey, Rajarshi Ray. Collision-Free Linear Time Mutual Visibility for Asynchronous Fat Robots. *ICDCN 2024, pp. 84–93*, 2024. doi:10.1145/3631461.3631544. Cited on pp. 413, 423, 424, 657, and 708.

Subhash Bhagat, Andrzej Pelc. Deterministic Collision-Free Exploration of Unknown Anonymous Graphs. *ICDCN 2025, ACM, pp. 115–124*, 2025. doi:10.1145/3700838.3700858, arXiv:2401.13044 (pdf). First appeared in 2024. Read: arXiv v1 (2024). Cited on pp. 9, 88, 232, 234, 235, 236, 240, 241, 666, 684, 697, and 709.

Subhash Bhagat, Andrzej Pelc. Exploring wedges of an oriented grid by an automaton with pebbles. *Journal of Computer and System Sciences 155:103696*, 2026. doi:10.1016/j.jcss.2025.103696, arXiv:2302.00052 (pdf). First appeared in 2023. Read: arXiv v1 (2023). Cited on p. 85.

Adri Bhattacharya, Barun Gorain, Partha Sarathi Mandal. Treasure Hunt in Graph Using Pebbles. *SSS 2022, LNCS, pp. 99–113*, 2022. doi:10.1007/978-3-031-21017-4_7, arXiv:2209.00857 (pdf). Read: arXiv v1 (2022). Cited on p. 109.

Adri Bhattacharya, Barun Gorain, Partha Sarathi Mandal. Pebble Guided Treasure Hunt in Plane. *NETYS 2023, LNCS, pp. 141–156*, 2023. doi:10.1007/978-3-031-37765-5_11, arXiv:2305.06067 (pdf). Read: arXiv v1 (2023). Cited on p. 110.

Adri Bhattacharya, Giuseppe F. Italiano, Partha Sarathi Mandal. Black Hole Search in Dynamic Cactus Graph. *WALCOM 2024, LNCS, pp. 288–303*, 2024. doi:10.1007/978-981-97-0566-5_21, arXiv:2311.10984 (pdf). First appeared in 2023. Read: arXiv v1 (2023). Cited on p. 292.

Adri Bhattacharya, Giuseppe F. Italiano, Partha Sarathi Mandal. Black Hole Search in Dynamic Tori. *SAND 2024, LIPIcs 292, 6:1–6:16*, 2024. doi:10.4230/LIPIcs.SAND.2024.6, pdf, arXiv:2402.04746 (pdf). Read: arXiv v1 (2024). Cited on p. 292.

Adri Bhattacharya, Pritam Goswami, Evangelos Bampas, Partha Sarathi Mandal. Perpetual Exploration in Anonymous Synchronous Networks with a Byzantine Black Hole. *DISC 2025, LIPIcs 356, 16:1–16:17*, 2025. doi:10.4230/LIPIcs.DISC.2025.16, pdf, link, arXiv:2508.07703 (pdf). Read: arXiv v1 (2025), with the appendices. Cited on pp. 291 and 292.

Kass Bileski, Avery Miller. Don't Be Afraid to Die: Black Hole Search in Dynamic Graphs with Fewer Agents. *arXiv preprint*, 2026. arXiv:2609.06850 (pdf). Preprint, not peer reviewed. Cited on pp. 304, 305, and 655.

Alexander Birx, Yann Disser, Alexander V. Hopp, Christina Karousatou. An improved lower bound for competitive graph exploration. *Theoretical Computer Science 868:65–86*, 2021. doi:10.1016/j.tcs.2021.04.003, arXiv:2002.10958 (pdf). First appeared in 2020. Cited on p. 225.

Mainak Biswas, Saif Rahaman, Moumita Mondal, Sruti Gan Chaudhuri. Multiple Uniform Circle Formation by Fat Robots Under Limited Visibility. *ICDCN 2023 (short paper), ACM, pp. 311–317*, 2023. doi:10.1145/3571306.3571427, pdf. Cited on p. 399.

Lélia Blin, Pierre Fraigniaud, Nicolas Nisse, Sandrine Vial. Distributed chasing of network intruders. *Theoretical Computer Science 399(1–2):12–37*, 2008. doi:10.1016/j.tcs.2008.02.004, pdf. Other versions: *SIROCCO 2006* (doi:10.1007/11780823_7, pdf). First appeared in 2006. Cited on p. 319.

Lélia Blin, Alessia Milani, Maria Potop-Butucaru, Sébastien Tixeuil. Exclusive perpetual ring exploration without chirality. *DISC 2010, LNCS 6343, pp. 312–327*, 2010. doi:10.1007/978-3-642-15763-9_29, link. Cited on pp. 165, 166, 167, 172, 173, 660, 670, and 696.

Manuel Blum, Dexter Kozen. On the power of the compass (or, why mazes are easier to search than graphs). *FOCS 1978, pp. 132–142*, 1978. doi:10.1109/SFCS.1978.30. Cited on pp. 56, 85, 89, 93, 207, 210, and 293.

Hans L. Bodlaender, Tom C. van der Zanden. On Exploring Temporal Graphs of Small Pathwidth. *arXiv preprint*, 2018. arXiv:1807.11869 (pdf). Preprint, not peer reviewed. Cited on p. 87.

Dominik Bojko, Karol Gotfryd, Dariusz R. Kowalski, Dominik Pająk. Tree Exploration in Dual-Memory Model.

MFCS 2022 (LIPIcs 241), 22:1–22:16, 2022. doi:10.4230/LIPIcs.MFCS.2022.22, pdf, arXiv:2112.13449 (pdf). First appeared in 2021. Cited on pp. 208, 211, 214, 215, 216, 221, and 705.

François Bonnet, Alessia Milani, Maria Potop-Butucaru, Sébastien Tixeuil. Asynchronous exclusive perpetual grid exploration without sense of direction. *OPODIS 2011, LNCS 7109, pp. 251–265*, 2011. doi:10.1007/978-3-642-25873-2_18, link. Cited on p. 165.

François Bonnet, Xavier Défago, Franck Petit, Maria Potop-Butucaru, Sébastien Tixeuil. Discovering and assessing fine-grained metrics in robot networks protocols. *SRDS Workshops 2014, pp. 50–59*, 2014. doi:10.1109/SRDSW.2014.34. Other versions: *SSS 2012 brief announcement, LNCS, pp. 282–284*, as "Brief Announcement: Discovering and Assessing Fine-Grained Metrics in Robot Networks Protocols" (doi:10.1007/978-3-642-33536-5_28). First appeared in 2012. Cited on pp. 165 and 174.

François Bonnet, Quentin Bramas, Pierre Courtieu, Xavier Défago, Lionel Rieg, Sébastien Tixeuil, Xavier Urbain. Deterministic color-optimal self-stabilizing semi-synchronous gathering: Two certified algorithms. *Theoretical Computer Science 1078:116013*, 2026. doi:10.1016/j.tcs.2026.116013. Other versions: *SIROCCO 2025, LNCS, pp. 127–143*, as "Deterministic Color-Optimal Self-stabilizing Semi-synchronous Gathering: A Certified Algorithm" (doi:10.1007/978-3-031-91736-3_8, pdf, link). First appeared in 2025. Read: SIROCCO 2025 version (HAL hal-04988128v1). Cited on pp. 443, 444, 447, 651, and 709.

François Bonnet, Quentin Bramas, Anissa Lamani. Searching for an Eventually-Emerging Black Hole in Rings. *SAND 2026, LIPIcs 373, 18:1–18:18*, 2026. doi:10.4230/LIPIcs.SAND.2026.18, pdf, link. Other versions: *SSS 2025 (brief announcement), LNCS 16350, pp. 82–87*, as "Brief Announcement: Searching for an Eventually-Emerging Black Hole in Rings" (doi:10.1007/978-3-032-11127-2_8). First appeared in 2025. Cited on pp. 291, 294, 300, 301, 302, 655, 675, and 709.

Piotr Borowiecki, Shantanu Das, Dariusz Dereniowski, Łukasz Kuszner. Discrete evacuation in graphs with multiple exits. *Theoretical Computer Science 1035, pp. 115141*, 2025. doi:10.1016/j.tcs.2025.115141, pdf. Other versions: *SIROCCO 2016, LNCS, pp. 228–241*, as "Distributed Evacuation in Graphs with Multiple Exits" (doi:10.1007/978-3-319-48314-6_15). First appeared in 2016. Cited on p. 508.

Kaustav Bose, Ranendu Adhikary, Sruti Gan Chaudhuri, Buddhadeb Sau. Crash tolerant gathering on grid by asynchronous oblivious robots. *arXiv preprint*, 2017. arXiv:1709.00877 (pdf). Preprint, not peer reviewed. Cited on p. 142.

Kaustav Bose, Ranendu Adhikary, Manash Kumar Kundu, Buddhadeb Sau. Positional Encoding by Robots with Non-rigid Movements. *SIROCCO 2019, LNCS, pp. 94–108*, 2019. doi:10.1007/978-3-030-24922-9_7, arXiv:1905.09786 (pdf). Read: arXiv v1 (2019). Cited on p. 67.

Kaustav Bose, Ranendu Adhikary, Manash Kumar Kundu, Buddhadeb Sau. Arbitrary pattern formation on infinite grid by asynchronous oblivious robots. *Theoretical Computer Science 815:213–227*, 2020. doi:10.1016/j.tcs.2020.02.016, arXiv:1811.00834 (pdf). Other versions: *ISAAC 2018, LNCS, pp. 354–366* (doi:10.1007/978-3-030-10564-8_28). First appeared in 2018. Cited on p. 382.

Kaustav Bose, Ranendu Adhikary, Manash Kumar Kundu, Buddhadeb Sau. Arbitrary Pattern Formation by Opaque Fat Robots with Lights. *CALDAM 2020, LNCS 12016, pp. 347–359*, 2020. doi:10.1007/978-3-030-39219-2_28, arXiv:1910.02706 (pdf). First appeared in 2019. Read: arXiv v1 (2019). Cited on pp. 392 and 703.

Kaustav Bose, Manash Kumar Kundu, Ranendu Adhikary, Buddhadeb Sau. Arbitrary pattern formation by asynchronous opaque robots with lights. *Theoretical Computer Science 849:138–158*, 2021. doi:10.1016/j.tcs.2020.10.015, arXiv:1902.04950 (pdf). Other versions: *SIROCCO 2019* (doi:10.1007/978-3-030-24922-9_8). First appeared in 2019. Read: arXiv v4 (2020). Cited on pp. 23, 25, 390, 391, 651, 682, and 703.

Kaustav Bose, Archak Das, Buddhadeb Sau. Pattern Formation by Robots with Inaccurate Movements. *OPODIS 2021, LIPIcs 217, 10:1–10:20*, 2021. doi:10.4230/LIPIcs.OPODIS.2021.10, arXiv:2010.09667 (pdf). First appeared in 2020. Cited on p. 383.

Sébastien Bouchard, Yoann Dieudonné, Bertrand Ducourthial. Byzantine gathering in networks. *Distributed Computing 29(6):435–457*, 2016. doi:10.1007/s00446-016-0276-9, arXiv:1504.01623 (pdf). Other versions: *SIROCCO 2015* (doi:10.1007/978-3-319-25258-2_13). First appeared in 2015. Cited on pp. 6, 13, 64, 75, 78, 133, 138, 642, and 699.

Sébastien Bouchard, Yoann Dieudonné, Andrzej Pelc, Franck Petit. On deterministic rendezvous at a node of agents with arbitrary velocities. *Information Processing Letters 133:39–43*, 2018. doi:10.1016/j.ipl.2018.01.003, arXiv:1710.08291 (pdf). First appeared in 2017. Cited on p. 119.

Sébastien Bouchard. On the Deterministic Gathering of Mobile Agents. *PhD thesis, Sorbonne Université*, 2019. doi:10.70675/0920326ez9dd5z4d4bz983dz84d910d3fdb7, link. Cited on p. 133.

Claude (Anthropic) was used in cataloguing the problems.

Sébastien Bouchard, Yoann Dieudonné, Andrzej Pelc, Franck Petit. Deterministic Treasure Hunt in the Plane with Angular Hints. *Algorithmica 82(11):3250–3281*, 2020. doi:10.1007/s00453-020-00724-4, arXiv:2005.00875 (pdf). Other versions: *ISAAC 2018, LIPIcs 123, 48:1–48:13* (doi:10.4230/LIPIcs.ISAAC.2018.48). First appeared in 2018. Read: arXiv v1 (2020). Cited on p. 109.

Sébastien Bouchard, Arnaud Labourel, Andrzej Pelc. Impact of knowledge on the cost of treasure hunt in trees. *Networks 80(1):51–62*, 2021. doi:10.1002/net.22075, arXiv:2503.13100 (pdf). Read: arXiv v1 (2025). Cited on p. 109.

Sébastien Bouchard, Yoann Dieudonné, Anissa Lamani. Byzantine gathering in polynomial time. *Distributed Computing 35(3):235–263*, 2022. doi:10.1007/s00446-022-00419-9, arXiv:1801.07656 (pdf). Other versions: *ICALP 2018*. First appeared in 2018. Read: arXiv v1 (2018). Cited on pp. 62, 74, 75, 133, and 138.

Sébastien Bouchard, Yoann Dieudonné, Arnaud Labourel, Andrzej Pelc. Almost-optimal deterministic treasure hunt in unweighted graphs. *ACM Transactions on Algorithms 19(3)*, 2023. doi:10.1145/3588437, arXiv:2010.14916 (pdf). Other versions: *ICALP 2021, LIPIcs 198, 36:1–36:20*, as "Almost-Optimal Deterministic Treasure Hunt in Arbitrary Graphs" (doi:10.4230/LIPIcs.ICALP.2021.36, pdf). First appeared in 2020. Read: arXiv v5 (2021). Cited on pp. 58, 108, 109, 110, 115, 649, and 704.

Sébastien Bouchard, Yoann Dieudonné, Andrzej Pelc. Want to Gather? No Need to Chatter!. *SIAM Journal on Computing 52(2):358–411*, 2023. doi:10.1137/20M1362899, arXiv:1908.11402 (pdf). Other versions: *PODC 2020* (doi:10.1145/3382734.3405693). First appeared in 2019. Read: arXiv v1 (2019). Cited on pp. 60, 62, and 136.

Marjorie Bournat, Swan Dubois, Franck Petit. Computability of perpetual exploration in highly dynamic rings. *ICDCS 2017, pp. 794–804*, 2017. doi:10.1109/ICDCS.2017.80, link, arXiv:1612.05767 (pdf). Other versions: *AlgoTel 2017 (in French)*, as "Quel est le nombre optimal de robots pour explorer un anneau hautement dynamique ?" (link). First appeared in 2016. Cited on p. 307.

Marjorie Bournat, Swan Dubois, Franck Petit. Gracefully Degrading Gathering in Dynamic Rings. *SSS 2018, LNCS, pp. 349–364*, 2018. doi:10.1007/978-3-030-03232-6_23, arXiv:1805.05137 (pdf). Cited on p. 134.

Marjorie Bournat, Ajoy K. Datta, Swan Dubois. Self-stabilizing robots in highly dynamic environments. *Theoretical Computer Science 772:88–110*, 2019. doi:10.1016/j.tcs.2018.11.026, pdf, link, arXiv:1609.06161 (pdf). Other versions: *SSS 2016, LNCS 10083, pp. 54–69* (doi:10.1007/978-3-319-49259-9_5, pdf). First appeared in 2016. Cited on p. 307.

Yann Bourreau, Ananth Narayanan, Alexandre Nolin. Optimal Deterministic Rendezvous in Labeled Lines. *STACS 2026, LIPIcs 364, 18:1–18:20*, 2026. doi:10.4230/LIPIcs.STACS.2026.18, pdf, link, arXiv:2505.04564 (pdf). Other versions: *PODC 2025 (brief announcement), pp. 436–439*, as "Brief Announcement: Optimal Deterministic Rendezvous in Labeled Lines" (doi:10.1145/3732772.3733537). First appeared in 2025. Read: arXiv v2 (2026). Cited on pp. 119, 129, 130, and 709.

Zohir Bouzid, Anissa Lamani. Robot Networks with Homonyms: The Case of Patterns Formation. *SSS 2011, pp. 92–107*, 2011. doi:10.1007/978-3-642-24550-3_9, arXiv:1105.5817 (pdf). Read: arXiv v1 (2011). Cited on pp. 12, 18, 26, 54, and 445.

Zohir Bouzid, Shantanu Das, Sebastien Tixeuil. Gathering of Mobile Robots Tolerating Multiple Crash Faults. *ICDCS 2013, pp. 337–346*, 2013. doi:10.1109/ICDCS.2013.27, arXiv:1207.0226 (pdf). First appeared in 2012. Read: arXiv v1 (2012). Cited on pp. 24, 25, 74, 435, and 451.

Quentin Bramas, Sébastien Tixeuil. Wait-Free Gathering Without Chirality. *SIROCCO 2015, pp. 313–327*, 2015. doi:10.1007/978-3-319-25258-2_22. Cited on pp. 22 and 451.

Quentin Bramas, Sébastien Tixeuil. Probabilistic asynchronous arbitrary pattern formation. *SSS 2016, pp. 88–93*, 2016. doi:10.1007/978-3-319-49259-9_7, arXiv:1508.03714 (pdf). Other versions: *PODC 2016 brief announcement*, as "Brief Announcement: Probabilistic Asynchronous Arbitrary Pattern Formation" (doi:10.1145/2933057.2933074). First appeared in 2015. Cited on pp. 18, 26, 386, 389, 390, 427, 434, 475, 478, 642, and 656.

Quentin Bramas, Sébastien Tixeuil. The Random Bit Complexity of Mobile Robots Scattering. *International Journal of Foundations of Computer Science 28(2):111–133*, 2017. doi:10.1142/S0129054117500083, arXiv:1309.6603 (pdf). Other versions: *ADHOC-NOW 2015* (doi:10.1007/978-3-319-19662-6_15). First appeared in 2013. Cited on pp. 26, 54, 474, 475, 477, 478, 656, and 698.

Quentin Bramas, Sébastien Tixeuil. Arbitrary pattern formation with four robots. *SSS 2018, pp. 333–348*, 2018. doi:10.1007/978-3-030-03232-6_22. Cited on p. 386.

Quentin Bramas, Anissa Lamani, Sébastien Tixeuil. Stand Up Indulgent Rendezvous. *SSS 2020, LNCS, pp. 45–59*, 2020. doi:10.1007/978-3-030-64348-5_4, arXiv:2010.04400 (pdf). Cited on p. 433.

Quentin Bramas, Stéphane Devismes, Pascal Lafourcade. Infinite grid exploration by disoriented robots. *NETYS*

2020, LNCS, pp. 129–145, 2020. doi:10.1007/978-3-030-67087-0_9, pdf, arXiv:1905.09271 (pdf). Other versions: *SIROCCO 2019 (brief announcement), LNCS, pp. 340–344* (doi:10.1007/978-3-030-24922-9_25, pdf). First appeared in 2019. Cited on pp. 177 and 180.

Quentin Bramas, Stéphane Devismes, Anaïs Durand, Pascal Lafourcade, Anissa Lamani. Beedroids: how luminous autonomous swarms of UAVs can save the world?. *FUN 2022 (LIPIcs 226), 7:1–7:21*, 2022. doi:10.4230/LIPIcs.FUN.2022.7, pdf. Cited on p. 177.

Quentin Bramas, Pascal Lafourcade, Stéphane Devismes. Optimal exclusive perpetual grid exploration by luminous myopic opaque robots with common chirality. *Theoretical Computer Science 977:114162*, 2023. doi:10.1016/j.tcs.2023.114162, pdf, link. Other versions: *ICDCN 2021, pp. 76–85* (doi:10.1145/3427796.3427834, pdf). First appeared in 2021. Cited on pp. 176, 178, 180, 181, 186, 187, 188, 189, 190, 191, 192, 193, 194, and 705.

Quentin Bramas, Anissa Lamani, Sébastien Tixeuil. Stand up indulgent gathering. *Theoretical Computer Science 939:63–77*, 2023. doi:10.1016/j.tcs.2022.10.015, pdf, arXiv:2302.03466 (pdf). Other versions: *ALGOSENSORS 2021, LNCS, pp. 17–28* (doi:10.1007/978-3-030-89240-1_2). First appeared in 2021. Read: arXiv v1 (2023). Cited on pp. 51, 62, 74, 75, 452, and 705.

Quentin Bramas, Sayaka Kamei, Anissa Lamani, Sébastien Tixeuil. Stand-up indulgent gathering on lines. *Theoretical Computer Science 1016:114796*, 2024. doi:10.1016/j.tcs.2024.114796, arXiv:2304.05722 (pdf). Other versions: *SSS 2023, LNCS, pp. 451–465*, as "Stand-Up Indulgent Gathering on Lines" (doi:10.1007/978-3-031-44274-2_34); *SIROCCO 2024, LNCS, pp. 119–137*, as "Stand-Up Indulgent Gathering on Rings" (doi:10.1007/978-3-031-60603-8_7). First appeared in 2023. Cited on p. 142.

Quentin Bramas, Toshimitsu Masuzawa, Sébastien Tixeuil. Crash-Tolerant Exploration of Trees by Energy-Sharing Mobile Agents. *OPODIS 2024, LIPIcs 324, 9:1–9:16*, 2024. doi:10.4230/LIPIcs.OPODIS.2024.9. Other versions: *SAND 2024 (brief announcement), LIPIcs 292, 25:1–25:5*, as "Brief Announcement: Crash-Tolerant Exploration of Trees by Energy Sharing Mobile Agents" (doi:10.4230/LIPIcs.SAND.2024.25); *SSS 2023 (brief announcement), LNCS, pp. 380–384*, as "Brief Announcement: Crash-Tolerant Exploration by Energy Sharing Mobile Agents" (doi:10.1007/978-3-031-44274-2_28). First appeared in 2023. Cited on p. 247.

Quentin Bramas, Pascal Lafourcade, Stéphane Devismes. Infinite grid exploration with synchronous myopic robots without chirality. *Discrete Applied Mathematics 375:193–214*, 2025. doi:10.1016/j.dam.2025.05.031. Other versions: *FUN 2021 (LIPIcs 157), 6:1–6:19*, as "Finding water on Poleless using melomaniac myopic chameleon robots" (doi:10.4230/LIPIcs.FUN.2021.6, pdf). First appeared in 2021. Cited on p. 177.

Quentin Bramas, Hirotsugu Kakugawa, Sayaka Kamei, Anissa Lamani, Fukuhito Ooshita, Masahiro Shibata, Sébastien Tixeuil. A visibility vs. memory trade-off for stand-up indulgent gathering on lines. *SIROCCO 2025, LNCS, pp. 159–175*, 2025. doi:10.1007/978-3-031-91736-3_10. Cited on p. 177.

Quentin Bramas, Stéphane Devismes, Anaïs Durand, Pascal Lafourcade, Anissa Lamani. Optimal asynchronous perpetual finite grid exploration. *Theoretical Computer Science 1082:116072*, 2026. doi:10.1016/j.tcs.2026.116072, link. Other versions: *SSS 2024, LNCS, pp. 89–105*, as "Optimal Asynchronous Perpetual Grid Exploration" (doi:10.1007/978-3-031-74498-3_6). First appeared in 2024. Cited on pp. 177, 178, 180, 192, 193, 194, 661, 680, and 709.

Quentin Bramas, Stéphane Devismes, Anaïs Durand, Pascal Lafourcade, Anissa Lamani. 77 Shades of Grey. *FUN 2026, LIPIcs 366, 10:1–10:19*, 2026. doi:10.4230/LIPIcs.FUN.2026.10, pdf. Other versions: *Preprint, HAL hal-05504522*, 2026, as "77 shades of grey" (link). Cited on p. 177.

Quentin Bramas, Hirotsugu Kakugawa, Sayaka Kamei, Anissa Lamani, Fukuhito Ooshita, Masahiro Shibata, Sébastien Tixeuil. Stand-up indulgent gathering on lines for myopic luminous robots. *The Computer Journal 69(7):1122–1132*, 2026. doi:10.1093/comjnl/bxag015, arXiv:2312.12698 (pdf). Other versions: *Advanced Information Networking and Applications 2024, pp. 110–121* (doi:10.1007/978-3-031-57853-3_10). First appeared in 2023. Cited on p. 177.

Sebastian Brandt, Felix Laufenberg, Yuezhou Lv, David Stolz, Roger Wattenhofer. Collaboration Without Communication: Evacuating Two Robots from a Disk. *CIAC 2017, pp. 104–115*, 2017. doi:10.1007/978-3-319-57586-5_10. Cited on pp. 73, 507, 510, and 519.

Sebastian Brandt, Klaus-Tycho Foerster, Jonathan Maurer, Roger Wattenhofer. Online graph exploration on a restricted graph class: Optimal solutions for tadpole graphs. *Theoretical Computer Science 839:176–185*, 2020. doi:10.1016/j.tcs.2020.06.007, arXiv:1903.00581 (pdf). First appeared in 2019. Cited on p. 225.

Sebastian Brandt, Klaus-Tycho Foerster, Benjamin Richner, Roger Wattenhofer. Wireless evacuation on m rays with k searchers. *Theoretical Computer Science 811:56–69*, 2020. doi:10.1016/j.tcs.2018.10.032. Other versions: *SIROCCO 2017, LNCS, pp. 140–157* (doi:10.1007/978-3-319-72050-0_9). First appeared in 2017. Cited on pp. 51, 73, 499, 507, 509, 539, 655, and 702.

Sebastian Brandt, Jara Uitto, Roger Wattenhofer. A Tight Lower Bound for Semi-Synchronous Collaborative Grid Exploration. *Distributed Computing 33(6):471–484*, 2020. doi:10.1007/s00446-020-00369-0, arXiv:1705.03834 (pdf). Other versions: *DISC 2018, LIPIcs 121, 13:1–13:17* (doi:10.4230/LIPIcs.DISC.2018.13). First appeared in 2017. Cited on p. 87.

Michael Braun, Jannik Castenow, Friedhelm Meyer auf der Heide. Local Gathering of Mobile Robots in Three Dimensions. *SIROCCO 2020, LNCS, pp. 63–79*, 2020. doi:10.1007/978-3-030-54921-3_4, arXiv:2005.07495 (pdf). Read: arXiv v1 (2020). Cited on p. 433.

Joseph L. Briones, Tishya Chhabra, Joshua J. Daymude, Andréa W. Richa. Invited Paper: Asynchronous Deterministic Leader Election in Three-Dimensional Programmable Matter. *ICDCN 2023, pp. 38–47*, 2023. doi:10.1145/3571306.3571389, arXiv:2205.15412 (pdf). First appeared in 2022. Read: arXiv v2 (2022). Cited on pp. 45, 47, 70, 585, 586, 592, 594, 680, 690, and 706.

Kevin Buchin, Paola Flocchini, Irina Kostitsyna, Tom Peters, Nicola Santoro, Koichi Wada. On the computational power of energy-constrained mobile robots. *Information and Computation 303:105280*, 2025. doi:10.1016/j.ic.2025.105280, arXiv:2203.06546 (pdf). Other versions: *SIROCCO 2022, LNCS 13298, pp. 42–61*, as "On the Computational Power of Energy-Constrained Mobile Robots: Algorithms and Cross-Model Analysis" (doi:10.1007/978-3-031-09993-9_3). First appeared in 2022. Read: arXiv v1 (2022). Cited on pp. 20, 27, 28, 30, 34, and 37.

Lothar Budach. Automata and labyrinths. *Mathematische Nachrichten 86:195–282*, 1978. doi:10.1002/mana.19780860120. Cited on pp. 56, 83, 85, 89, 93, 210, and 691.

Andreas Bärtschi, Jérémie Chalopin, Shantanu Das, Yann Disser, Daniel Graf, Jan Hackfeld, Paolo Penna. Energy-Efficient Delivery by Heterogeneous Mobile Agents. *STACS 2017, LIPIcs 66, 10:1–10:14*, 2017. doi:10.4230/LIPIcs.STACS.2017.10, arXiv:1610.02361 (pdf). First appeared in 2016. Read: arXiv v1 (2016), abstract, model and contributions. Cited on p. 333.

Andreas Bärtschi, Jérémie Chalopin, Shantanu Das, Yann Disser, Barbara Geissmann, Daniel Graf, Arnaud Labourel, Matúš Mihalák. Collaborative delivery with energy-constrained mobile robots. *Theoretical Computer Science 810:2–14*, 2020. doi:10.1016/j.tcs.2017.04.018, arXiv:1608.08500 (pdf). Other versions: *SIROCCO 2016, LNCS, pp. 258–274* (doi:10.1007/978-3-319-48314-6_17). First appeared in 2016. Read: arXiv v1 (2016). Cited on pp. 333, 334, 336, 337, 338, 339, 667, and 701.

Andreas Bärtschi, Evangelos Bampas, Jérémie Chalopin, Shantanu Das, Christina Karousatou, Matúš Mihalák. Near-gathering of energy-constrained mobile agents. *Theoretical Computer Science 849:35–46*, 2021. doi:10.1016/j.tcs.2020.10.008, pdf, link. Other versions: *SIROCCO 2019, LNCS, pp. 52–65* (doi:10.1007/978-3-030-24922-9_4). First appeared in 2019. Cited on p. 133.

Béatrice Bérard, Pascal Lafourcade, Laure Millet, Maria Potop-Butucaru, Yann Thierry-Mieg, Sébastien Tixeuil. Formal verification of mobile robot protocols. *Distributed Computing 29(6):459–487*, 2016. doi:10.1007/s00446-016-0271-1, pdf, link. Cited on pp. 165, 166, 169, 170, 173, 660, and 701.

Hans-Joachim Böckenhauer, Janosch Fuchs, Ulla Karhumäki, Walter Unger. Online Exploration of Rectangular Grids. *arXiv preprint*, 2016. arXiv:1611.00934 (pdf). Preprint, not peer reviewed. Cited on p. 225.

Hans-Joachim Böckenhauer, Janosch Fuchs, Walter Unger. The Graph Exploration Problem with Advice. *arXiv preprint*, 2018. arXiv:1804.06675 (pdf). Preprint, not peer reviewed. Cited on p. 225.

Hans-Joachim Böckenhauer, Fabian Frei, Walter Unger, David Wehner. Zero-Memory Graph Exploration with Unknown Inports. *SIROCCO 2023, LNCS 13892, pp. 246–261*, 2023. doi:10.1007/978-3-031-32733-9_11, arXiv:2301.13860 (pdf). Cited on pp. 205, 206, 207, 208, 209, 210, 211, 212, 214, 215, 216, 217, 218, 662, 672, 681, and 707.

David Caissy, Andrzej Pelc. Exploration of Faulty Hamiltonian Graphs. *International Journal of Foundations of Computer Science 27(07):809–827*, 2016. doi:10.1142/s0129054116500313, arXiv:1602.05136 (pdf). Cited on p. 86.

Sarah Cannon, Joshua J. Daymude, Dana Randall, Andréa W. Richa. A Markov Chain Algorithm for Compression in Self-Organizing Particle Systems. *PODC 2016, pp. 279–288*, 2016. doi:10.1145/2933057.2933107, arXiv:1603.07991 (pdf). Read: arXiv v4 (2019). Cited on pp. 46, 70, 608, 609, 611, 612, 643, 700, and 701.

Sarah Cannon, Joshua J. Daymude, Cem Gökmen, Dana Randall, Andréa W. Richa. A Local Stochastic Algorithm for Separation in Heterogeneous Self-Organizing Particle Systems. *APPROX/RANDOM 2019, LIPIcs 145, 54:1–54:22*, 2019. doi:10.4230/LIPIcs.APPROX-RANDOM.2019.54, pdf, arXiv:1805.04599 (pdf). Other versions: *PODC 2018 brief announcement, pp. 483–485*, as "Brief Announcement: A Local Stochastic Algorithm for Separation in Heterogeneous Self-Organizing Particle Systems" (doi:10.1145/3212734.3212792). First appeared in 2018. Read: arXiv v2 (2019). Cited on pp. 46, 70, 610, 611, 612, 613, 614, 702, and 703.

Daniele Carnevale. Hiding a Vertex from the Temporal Explorer: A Lower Bound for Degree-Bounded Temporal

Graphs. *arXiv preprint*, 2026. arXiv:2610.06228 (pdf). Preprint, not peer reviewed. Cited on pp. 7, 104, 105, 307, 644, and 711.

Arnaud Casteigts, Paola Flocchini, Walter Quattrociocchi, Nicola Santoro. Time-varying graphs and dynamic networks. *International Journal of Parallel, Emergent and Distributed Systems 27(5):387–408*, 2012. doi:10.1080/17445760.2012.668546, pdf, arXiv:1012.0009 (pdf). Other versions: *ADHOC-NOW 2011* (doi:10.1007/978-3-642-22450-8_27). First appeared in 2010. Cited on pp. 6 and 10.

Jannik Castenow, Matthias Fischer, Jonas Harbig, Daniel Jung, Friedhelm Meyer auf der Heide. Gathering Anonymous, Oblivious Robots on a Grid. *Theoretical Computer Science 815:289–309*, 2020. doi:10.1016/j.tcs.2020.02.018, arXiv:1702.03400 (pdf). Other versions: *ALGOSENSORS 2017, LNCS, pp. 168–181* (doi:10.1007/978-3-319-72751-6_13). First appeared in 2017. Cited on p. 142.

Jannik Castenow, Jonas Harbig, Daniel Jung, Till Knollmann, Friedhelm Meyer auf der Heide. Gathering a Euclidean Closed Chain of Robots in Linear Time. *ALGOSENSORS 2021, LNCS, pp. 29–44*, 2021. doi:10.1007/978-3-030-89240-1_3, arXiv:2010.04424 (pdf). First appeared in 2020. Read: arXiv v2. Cited on p. 433.

Jannik Castenow, Jonas Harbig, Daniel Jung, Peter Kling, Till Knollmann, Friedhelm Meyer auf der Heide. A unifying approach to efficient (near)-gathering of disoriented robots with limited visibility. *OPODIS 2022 (LIPIcs 253), 15:1–15:25*, 2022. doi:10.4230/LIPIcs.OPODIS.2022.15, pdf, arXiv:2206.07567 (pdf). Read: arXiv v4 (2022). Cited on pp. 22, 63, 432, 457, 460, 461, 462, 464, 469, and 706.

Abhinav Chakraborty, Pritam Goswami, Satakshi Ghosh. Min-Max Gathering on Infinite Grid. *arXiv preprint*, 2024. arXiv:2410.11966 (pdf). Preprint, not peer reviewed. Cited on p. 142.

Abhinav Chakraborty, Krishnendu Mukhopadhyaya. Parking problem by oblivious mobile robots in infinite grids. *Theoretical Computer Science 1023:114953*, 2025. doi:10.1016/j.tcs.2024.114953, arXiv:2310.09813 (pdf). First appeared in 2023. Read: arXiv v1 (2023). Cited on p. 466.

Jérémie Chalopin, Shantanu Das, Adrian Kosowski. Constructing a Map of an Anonymous Graph: Applications of Universal Sequences. *OPODIS 2010, pp. 119–134*, 2010. doi:10.1007/978-3-642-17653-1_10. Cited on pp. 156, 157, 158, 159, 160, 161, and 696.

Jérémie Chalopin, Paola Flocchini, Bernard Mans, Nicola Santoro. Network exploration by silent and oblivious robots. *WG 2010, LNCS 6410, pp. 208–219*, 2010. doi:10.1007/978-3-642-16926-7_20. Cited on p. 165.

Jérémie Chalopin, Shantanu Das, Matúš Mihalák, Paolo Penna, Peter Widmayer. Data Delivery by Energy-Constrained Mobile Agents. *ALGOSENSORS 2013, LNCS, pp. 111–122*, 2013. doi:10.1007/978-3-642-45346-5_9, pdf, link. Read: the authors' copy. Cited on pp. 10, 331, 332, 333, 334, 336, 337, 338, 339, 667, and 698.

Jérémie Chalopin, Riko Jacob, Matúš Mihalák, Peter Widmayer. Data Delivery by Energy-Constrained Mobile Agents on a Line. *ICALP 2014, LNCS, pp. 423–434*, 2014. doi:10.1007/978-3-662-43951-7_36, pdf, link. Read: the authors' copy. Cited on pp. 10, 332, 333, 334, 337, 667, and 699.

Jérémie Chalopin, Emmanuel Godard, Antoine Naudin. Anonymous Graph Exploration with Binoculars. *DISC 2015, LNCS, pp. 107–122*, 2015. doi:10.1007/978-3-662-48653-5_8, link, arXiv:1505.00599 (pdf). Other versions: *AlgoTel 2015 (in French)*, as "Exploration de graphes anonymes avec des jumelles" (link). Read: arXiv v1 (2015). Cited on p. 86.

Jérémie Chalopin, Shantanu Das, Paweł Gawrychowski, Adrian Kosowski, Arnaud Labourel, Przemysław Uznański. Limit Behavior of the Multi-agent Rotor-Router System. *DISC 2015, LNCS, pp. 123–139*, 2015. doi:10.1007/978-3-662-48653-5_9. Read: the HAL version. Cited on p. 208.

Jérémie Chalopin, Yoann Dieudonné, Arnaud Labourel, Andrzej Pelc. Rendezvous in networks in spite of delay faults. *Distributed Computing 29(3):187–205*, 2016. doi:10.1007/s00446-015-0259-2, arXiv:1402.2760 (pdf). Other versions: *ICALP 2014, LNCS, pp. 411–422*, as "Fault-Tolerant Rendezvous in Networks" (doi:10.1007/978-3-662-43951-7_35). First appeared in 2014. Cited on p. 119.

Jérémie Chalopin, Emmanuel Godard, Antoine Naudin. Using Binoculars for Fast Exploration and Map Construction in Chordal Graphs and Extensions. *arXiv preprint*, 2016. arXiv:1604.05915 (pdf). Other versions: *AlgoTel 2016 (in French)*, as "Utiliser des jumelles pour explorer rapidement les graphes triangulés" (link). Preprint, not peer reviewed. Cited on p. 158.

Jérémie Chalopin, Shantanu Das, Yann Disser, Arnaud Labourel, Matúš Mihalák. Collaborative delivery on a fixed path with homogeneous energy-constrained agents. *Theoretical Computer Science 868:87–96*, 2021. doi:10.1016/j.tcs.2021.04.004, pdf, link. Other versions: *SIROCCO 2019, LNCS, pp. 139–153* (doi:10.1007/978-3-030-24922-

9_10). First appeared in 2019. Read: author version on HAL (hal-03287588). Cited on pp. 331, 333, 334, 338, 339, 667, and 698.

Jérémie Chalopin, Shantanu Das, Maria Kokkou. Deterministic self-stabilising leader election for programmable matter with constant memory. *Distributed Computing*, 2025. doi:10.1007/s00446-025-00493-9, arXiv:2408.08775 (pdf). Other versions: *DISC 2024, LIPIcs 319, 13:1–13:17* (doi:10.4230/LIPIcs.DISC.2024.13, pdf). First appeared in 2024. Read: arXiv v2 (2025). Cited on pp. 70, 586, 591, and 709.

Jérémie Chalopin, Shantanu Das, Maria Kokkou. Silent Self-stabilising Leader Election in Programmable Matter Systems with Holes. *SIROCCO 2026, LNCS, pp. 151–171*, 2026. doi:10.1007/978-3-032-26465-7_9, arXiv:2507.20201 (pdf). First appeared in 2025. Read: arXiv v2 (2026). Cited on pp. 45, 46, 47, 586, 597, 672, and 710.

Jérémie Chalopin, Shantanu Das, Maria Kokkou. Deterministic leader election for stationary programmable matter with common direction. *Theoretical Computer Science 1080:116045*, 2026. doi:10.1016/j.tcs.2026.116045, arXiv:2402.10582 (pdf). Other versions: *SIROCCO 2024, LNCS, pp. 174–191* (doi:10.1007/978-3-031-60603-8_10). First appeared in 2024. Read: arXiv v1 (2024). Cited on pp. 43, 44, 45, 585, 586, 591, 672, 690, and 709.

Prabhat Kumar Chand, Anisur Rahaman Molla, Sumathi Sivasubramaniam. Run for Cover: Dominating Set via Mobile Agents. *ALGOSENSORS 2023, pp. 133–150*, 2023. doi:10.1007/978-3-031-48882-5_10, arXiv:2309.02200 (pdf). Read: arXiv v1 (2023). Cited on pp. 60, 343, 346, 347, 348, 351, 352, 353, 354, 356, 361, 365, 706, and 707.

Prabhat Kumar Chand, Manish Kumar, Anisur Rahaman Molla, Sumathi Sivasubramaniam. Fault-Tolerant Dispersion of Mobile Robots. *Discrete Applied Mathematics 377:299–313*, 2025. doi:10.1016/j.dam.2025.06.068, arXiv:2211.12451 (pdf). Other versions: *CALDAM 2023 (LNCS 13779)* (doi:10.1007/978-3-031-25211-2_3). First appeared in 2022. Read: arXiv v1 (2022). Cited on pp. 13, 75, 279, 280, 281, 647, and 648.

Prabhat Kumar Chand, Apurba Das, Anisur Rahaman Molla. Brief Announcement: Distributed Butterfly Analysis using Mobile Agents. *SPAA 2025 brief announcement, pp. 623–627*, 2025. doi:10.1145/3694906.3743334, arXiv:2506.17721 (pdf). Read: arXiv v1 (2025). Cited on pp. 16, 61, 356, 357, 358, and 710.

Prabhat Kumar Chand, Manish Kumar, Anisur Rahaman Molla. Computing Tree Structures in Anonymous Graphs via Mobile Agents. *SSS 2025, pp. 111–127*, 2025. doi:10.1007/978-3-032-11127-2_11, arXiv:2506.19365 (pdf). Read: arXiv v1 (2025). Cited on pp. 61, 343, 344, 348, 353, 357, 359, 360, 361, 362, 363, 364, 365, 366, 709, and 710.

Prabhat Kumar Chand, Anisur Rahaman Molla. Linear-time dominating set construction and leader election using mobile agents. *Theoretical Computer Science 1086:116245*, 2026. doi:10.1016/j.tcs.2026.116245, arXiv:2511.19880 (pdf). Other versions: *ICDCIT 2026, pp. 35–51*, as "Improved Linear-Time Construction of Minimal Dominating Set via Mobile Agents" (doi:10.1007/978-3-032-16632-6_3). First appeared in 2025. Read: arXiv v1 (2025). Cited on pp. 60, 61, 261, 340, 342, 343, 344, 345, 352, 353, 354, and 710.

Prabhat Kumar Chand, Apurba Das, Anisur Rahaman Molla. Agent-Based Triangle Counting: Unlocking Truss Decomposition, Triangle Centrality, and Local Clustering Coefficient. *IEEE Transactions on Networking (2026)*, 2026. doi:10.1109/TON.2026.3734259, arXiv:2402.03653 (pdf). First appeared in 2024. Read: arXiv v3 (2026). Cited on pp. 60, 61, 343, 348, 353, 354, 355, 356, 357, 358, 361, 365, 708, and 709.

Sruti Gan Chaudhuri, Krishnendu Mukhopadhyaya. Gathering Asynchronous Transparent Fat Robots. *ICDCIT 2010, LNCS 5966, pp. 170–175*, 2010. doi:10.1007/978-3-642-11659-9_17. Cited on pp. 459 and 696.

Sruti Gan Chaudhuri, Krishnendu Mukhopadhyaya. Leader election and gathering for asynchronous fat robots without common chirality. *Journal of Discrete Algorithms 33:171–192*, 2015. doi:10.1016/j.jda.2015.04.001, arXiv:1208.4484 (pdf). First appeared in 2012. Cited on pp. 53, 459, 483, 652, and 697.

Li-Hsuan Chen, Ling-Ju Hung, Ralf Klasing. Improved Approximation Algorithms for Patrol-Scheduling with Min-Max Latency Using Multiclass Minimum Spanning Forests. *AAIM 2024, LNCS, pp. 99–110*, 2024. doi:10.1007/978-981-97-7801-0_9. Read: the HAL version. Cited on p. 568.

Marek Chrobak, Leszek Gąsieniec, Thomas Gorry, Russell Martin. Group Search on the Line. *SOFSEM 2015, LNCS, pp. 164–176*, 2015. doi:10.1007/978-3-662-46078-8_14. Cited on pp. 39, 41, 63, 72, 499, 500, 503, 504, 507, 509, 533, 534, 535, 551, 552, 564, and 699.

Huda Chuangpishit, Jurek Czyzowicz, Evangelos Kranakis, Danny Krizanc. Rendezvous on a Line by Location-Aware Robots Despite the Presence of Byzantine Faults. *ALGOSENSORS 2017, LNCS, pp. 70–83*, 2017. doi:10.1007/978-3-319-72751-6_6, arXiv:1707.06776 (pdf). Cited on p. 120.

Huda Chuangpishit, Jurek Czyzowicz, Leszek Gąsieniec, Konstantinos Georgiou, Tomasz Jurdziński, Evangelos Kranakis. Patrolling a Path Connecting a Set of Points with Unbalanced Frequencies of Visits. *SOFSEM 2018,*

LNCS, pp. 367–380, 2018. doi:10.1007/978-3-319-73117-9_26, arXiv:1710.00466 (pdf). First appeared in 2017. Read: arXiv v1 (2017). Cited on p. 568.

Huda Chuangpishit, Konstantinos Georgiou, Preeti Sharma. Average Case - Worst Case Tradeoffs for Evacuating 2 Robots from the Disk in the Face-to-Face Model. *ALGOSENSORS 2018, LNCS*, 2018. doi:10.1007/978-3-030-14094-6_5, arXiv:1807.08640 (pdf). Other versions: Chuangpishit et al. (2020a). Read: arXiv v1 (2018). Cited on pp. 508 and 520.

Huda Chuangpishit, Konstantinos Georgiou, Preeti Sharma. A Multi-Objective Optimization Problem on Evacuating 2 Robots from the Disk in the Face-to-Face Model; Trade-Offs between Worst-Case and Average-Case Analysis. *Information 11, 506*, 2020. doi:10.3390/info11110506. Other versions: Chuangpishit et al. (2018b). First appeared in 2018. Cited on pp. 508 and 520.

Huda Chuangpishit, Saeed Mehrabi, Lata Narayanan, Jaroslav Opatrny. Evacuating equilateral triangles and squares in the face-to-face model. *Computational Geometry 89:101624*, 2020. doi:10.1016/j.comgeo.2020.101624, arXiv:1812.10162 (pdf). Other versions: *OPODIS 2017, LIPIcs 95, 11:1–11:16*, as "Evacuating an Equilateral Triangle in the Face-to-Face Model" (doi:10.4230/LIPIcs.OPODIS.2017.11). First appeared in 2017. Read: arXiv v1 (2018). Cited on pp. 73, 507, 509, 510, 535, 536, 537, and 701.

Serafino Cicerone, Gabriele Di Stefano, Alfredo Navarra. Asynchronous arbitrary pattern formation: the effects of a rigorous approach. *Distributed Computing 32(2):91–132*, 2019. doi:10.1007/s00446-018-0325-7, arXiv:1706.02474 (pdf). First appeared in 2017. Read: arXiv v2 (2018). Cited on pp. 22, 54, 66, 381, 382, 384, 386, 387, 388, 389, 390, 481, 482, 642, 651, and 701.

Serafino Cicerone, Gabriele Di Stefano, Alfredo Navarra. A structured methodology for designing distributed algorithms for mobile entities. *Information Sciences 574:111–132*, 2021. doi:10.1016/j.ins.2021.05.043, arXiv:2010.12463 (pdf). First appeared in 2020. Read: arXiv version (2020), titled 'A methodology to design distributed algorithms for mobile entities: the pattern formation problem as case study'. Cited on pp. 387, 435, and 452.

Serafino Cicerone. Breaking Symmetries on Tessellation Graphs via Asynchronous Robots: The Line Formation Problem as a Case Study. *IEEE Access 9:147855–147873*, 2021. doi:10.1109/access.2021.3123510, pdf. Other versions: *ICTCS 2020, CEUR-WS 2756, pp. 122–136*, as "Breaking symmetries on tessellation graphs via asynchronous robots" (link). First appeared in 2020. Cited on p. 383.

Serafino Cicerone, Gabriele Di Stefano, Alfredo Navarra. Gathering robots in graphs: The central role of synchronicity. *Theoretical Computer Science 849:99–120*, 2021. doi:10.1016/j.tcs.2020.10.011. Other versions: *SIROCCO 2019, LNCS 11639, pp. 170–184*, as "Gathering synchronous robots in graphs: from general properties to dense and symmetric topologies". First appeared in 2019. Read: the journal pre-proof (repository of the University of L'Aquila). Cited on pp. 151, 152, 665, and 704.

Serafino Cicerone, Gabriele Di Stefano, Alfredo Navarra. Solving the Pattern Formation by Mobile Robots With Chirality. *IEEE Access 9:88177–88204*, 2021. doi:10.1109/access.2021.3089081, pdf. Cited on pp. 387 and 705.

Serafino Cicerone, Alessia Di Fonso, Gabriele Di Stefano, Alfredo Navarra. Arbitrary Pattern Formation on Infinite Regular Tessellation Graphs. *Theoretical Computer Science 942:1–20*, 2023. doi:10.1016/j.tcs.2022.11.021, arXiv:2010.14152 (pdf). Other versions: *ICDCN 2021, pp. 56–65* (doi:10.1145/3427796.3427833). First appeared in 2020. Cited on p. 382.

Serafino Cicerone, Alessia Di Fonso, Gabriele Di Stefano, Alfredo Navarra. The geodesic mutual visibility problem: Oblivious robots on grids and trees. *Pervasive and Mobile Computing 95:101842*, 2023. doi:10.1016/j.pmcj.2023.101842, pdf. Other versions: *ICDCN 2023, ACM, pp. 150–159*, as "The Geodesic Mutual Visibility Problem for Oblivious Robots: the case of Trees" (doi:10.1145/3571306.3571401); Cicerone et al. (2023c). Cited on p. 414.

Serafino Cicerone, Alessia Di Fonso, Gabriele Di Stefano, Alfredo Navarra. Time-Optimal Geodesic Mutual Visibility of Robots on Grids Within Minimum Area. *SSS 2023, LNCS, pp. 385–399*, 2023. doi:10.1007/978-3-031-44274-2_29, arXiv:2308.01855 (pdf). Other versions: Cicerone et al. (2023b). Read: arXiv v1 (2023). Cited on p. 413.

Serafino Cicerone, Alessia Di Fonso, Gabriele Di Stefano, Alfredo Navarra, Francesco Piselli. Mutual and total mutual visibility in hypercube-like graphs. *Applied Mathematics and Computation 491:129216*, 2025. doi:10.1016/j.amc.2024.129216, pdf, arXiv:2308.14443 (pdf). Other versions: *SIROCCO 2024, LNCS, pp. 192–207*, as "Mutual Visibility in Hypercube-Like Graphs" (doi:10.1007/978-3-031-60603-8_11). First appeared in 2023. Read: arXiv v1 (2023), and the journal version (2025). Cited on p. 414.

Serafino Cicerone, Alessia Di Fonso, Gabriele Di Stefano, Alfredo Navarra. Gathering in Vertex- and Edge-

Transitive Graphs without Multiplicity Detection under Round Robin. *arXiv preprint*, 2025. arXiv:2511.08222 (pdf). Preprint, not peer reviewed. Cited on pp. 8, 141, 144, 145, 151, 153, 154, and 710.

Serafino Cicerone, Alessia Di Fonso, Gabriele Di Stefano, Alfredo Navarra. Brief Announcement: On the Impact of Unlimited Computational Power in OBLOT: Consequences for Synchronous Robots on Graphs. *SSS 2025 (brief announcement), LNCS 16350, pp. 146–152*, 2025. doi:10.1007/978-3-032-11127-2_13, arXiv:2509.04383 (pdf). Read: arXiv v1 (2025), the full version (18 pages). Cited on p. 36.

Serafino Cicerone, Alessia Di Fonso, Gabriele Di Stefano, Alfredo Navarra. Gathering in Non-Vertex-Transitive Graphs Under Round Robin. *SSS 2025, LNCS 16350, pp. 153–170*, 2025. doi:10.1007/978-3-032-11127-2_14, arXiv:2509.06064 (pdf). Read: arXiv v1 (September 2025). Cited on pp. 8, 141, 144, 145, 148, 150, 152, 153, 665, and 710.

Serafino Cicerone, Alessia Di Fonso, Gabriele Di Stefano, Alfredo Navarra. Optimal gathering of robots in anonymous butterfly networks via leader election. *Theoretical Computer Science 1057:115553*, 2025. doi:10.1016/j.tcs.2025.115553, pdf. Cited on p. 143.

Mark Cieliebak, Paola Flocchini, Giuseppe Prencipe, Nicola Santoro. Distributed Computing by Mobile Robots: Gathering. *SIAM Journal on Computing 41(4):829–879*, 2012. doi:10.1137/100796534. Other versions: *ICALP 2003*. First appeared in 2003. Cited on pp. 18, 24, 63, 65, 66, 430, 431, 432, 434, 449, and 693.

Julien Clement, Xavier Défago, Maria Gradinariu Potop-Butucaru, Taisuke Izumi, Stephane Messika. The cost of probabilistic agreement in oblivious robot networks. *Information Processing Letters 110(11):431–438*, 2010. doi:10.1016/j.ipl.2010.04.006, arXiv:0807.1753 (pdf). First appeared in 2008. Read: arXiv v1 (2008). Cited on pp. 21, 25, 26, 51, 53, 54, 62, 474, 475, 476, 477, and 696.

Stefano Clemente, Caterina Feletti. Fault detection and identification by swarms of autonomous mobile robots. *Theoretical Computer Science 1074:115933*, 2026. doi:10.1016/j.tcs.2026.115933. Other versions: *SAND 2025, LIPIcs 330, 10:1–10:20*, as "Fault detection and identification by autonomous mobile robots" (doi:10.4230/LIPIcs.SAND.2025.10). First appeared in 2025. Cited on p. 67.

Wyatt Clements, Costas Busch, Limeng Pu, Daniel Smith, Hsiao-Chun Wu. Balanced Parallel Exploration of Orthogonal Regions. *Algorithms 12(5):104*, 2019. doi:10.3390/a12050104, pdf. Cited on p. 247.

Reuven Cohen, David Peleg. Convergence properties of the gravitational algorithm in asynchronous robot systems. *SIAM Journal on Computing 34(6):1516–1528*, 2005. doi:10.1137/S0097539704446475. Other versions: *ESA 2004* (doi:10.1007/978-3-540-30140-0_22). First appeared in 2004. Cited on pp. 63, 434, 468, 469, 470, 472, and 694.

Reuven Cohen, Pierre Fraigniaud, David Ilcinkas, Amos Korman, David Peleg. Label-guided graph exploration by a finite automaton. *ACM Transactions on Algorithms 4(4)*, 2008. doi:10.1145/1383369.1383373, pdf. Other versions: *ICALP 2005* (doi:10.1007/11523468_28, pdf). First appeared in 2005. Cited on pp. 57, 88, 89, 93, 99, 206, 207, 208, 209, 210, 211, 212, 218, 221, 222, 369, 376, 663, 673, and 694.

Reuven Cohen, David Peleg. Local spreading algorithms for autonomous robot systems. *Theoretical Computer Science 399(1–2):71–82*, 2008. doi:10.1016/j.tcs.2008.02.007. Other versions: *SIROCCO 2006*, as "Local Algorithms for Autonomous Robot Systems" (doi:10.1007/11780823_4). First appeared in 2006. Cited on pp. 491 and 493.

Jared Coleman, Evangelos Kranakis, Oscar Morales-Ponce, Jaroslav Opatrný, Jorge Urrutia, Birgit Vogtenhuber. Minimizing The Maximum Distance Traveled To Form Patterns With Systems of Mobile Robots. *arXiv preprint*, 2020. arXiv:2006.15664 (pdf). Preprint, not peer reviewed. Cited on p. 383.

Jared Coleman, Evangelos Kranakis, Danny Krizanc, Oscar Morales-Ponce. Message Delivery in the Plane by Robots with Different Speeds. *SSS 2021, LNCS, pp. 305–319*, 2021. doi:10.1007/978-3-030-91081-5_20, arXiv:2109.12185 (pdf). Read: arXiv v1 (2021). Cited on p. 509.

Jared Coleman, Evangelos Kranakis, Danny Krizanc, Oscar Morales-Ponce. Line Search for an Oblivious Moving Target. *OPODIS 2022, LIPIcs 253, 12:1–12:19*, 2022. doi:10.4230/LIPIcs.OPODIS.2022.12, pdf, arXiv:2211.03686 (pdf). Cited on pp. 544, 545, 546, 547, 553, 554, 557, 558, 559, 560, 561, 562, 563, 666, 693, and 706.

Jared Coleman, Evangelos Kranakis, Danny Krizanc, Oscar Morales-Ponce. Delivery to Safety with Two Cooperating Robots. *Lecture Notes in Computer Science SOFSEM 2023: Theory and Practice of Computer Science, 359-371*, 2023. doi:10.1007/978-3-031-23101-8_24, arXiv:2210.04080 (pdf). First appeared in 2022. Read: arXiv v2 (2022). Cited on p. 508.

Jared Coleman, Oscar Morales-Ponce. The Snow Plow Problem: Perpetual Maintenance by Mobile Agents on the Line. *ICDCN 2023 (short paper), ACM, pp. 110–114*, 2023. doi:10.1145/3571306.3571396, pdf. Cited on p. 569.

Jared Coleman, Oscar Morales-Ponce. Linear Search with Probabilistic Detection and Variable Speeds. *IWOCA*

2025, LNCS, pp. 433–445, 2025. doi:10.1007/978-3-031-98740-3_31, arXiv:2505.09429 (pdf). Read: arXiv v1 (2025), the full version. Cited on p. 498.

Jared Coleman, Dmitry Ivanov, Evangelos Kranakis, Danny Krizanc, Oscar Morales-Ponce. The power of knowledge in linear search for an escaping target. *Journal of Computer and System Sciences 157:103737*, 2026. doi:10.1016/j.jcss.2025.103737, arXiv:2404.14300 (pdf). Other versions: *IWOCA 2024, LNCS, pp. 396–407*, as "Linear Search for an Escaping Target with Unknown Speed" (doi:10.1007/978-3-031-63021-7_30). First appeared in 2024. Read: arXiv v2 (2024), the full version of the IWOCA paper; theorem numbers refer to it. The JCSS version was also read: its Theorems 3, 4 and 5 are Theorems 1, 2 and 3 of arXiv v2. Cited on pp. 40, 546, 553, 559, 560, 561, 666, and 709.

Jared Coleman, Evangelos Kranakis, Danny Krizanc, Oscar Morales-Ponce. Search and Rescue on the Plane. *arXiv preprint*, 2026. arXiv:2608.12039 (pdf). Preprint, not peer reviewed. Cited on p. 508.

James Conley, Konstantinos Georgiou. Multi-agent Disk Inspection. *Lecture Notes in Computer Science Structural Information and Communication Complexity, 262-280*, 2025. doi:10.1007/978-3-031-91736-3_16, arXiv:2411.15391 (pdf). First appeared in 2024. Read: arXiv v4 (2025). Cited on p. 508.

Mario Coppola, Jian Guo, Eberhard K. A. Gill, Guido C. H. E. de Croon. Provable Emergent Pattern Formation by a Swarm of Anonymous, Homogeneous, Non-Communicating, Reactive Robots with Limited Relative Sensing and no Global Knowledge or Positioning. *arXiv preprint*, 2018. arXiv:1804.06827 (pdf). Preprint, not peer reviewed. Cited on p. 383.

Mario Coppola, Jian Guo, Eberhard Gill, Guido C. H. E. de Croon. Provable self-organizing pattern formation by a swarm of robots with limited knowledge. *Swarm Intelligence 13(1):59–94*, 2019. doi:10.1007/s11721-019-00163-0. Cited on p. 383.

Andreas Cord-Landwehr, Bastian Degener, Matthias Fischer, Martina Hüllmann, Barbara Kempkes, Alexander Klaas, Peter Kling, Sven Kurras, Marcus Märtens, Friedhelm Meyer auf der Heide, Christoph Raupach, Kamil Swierkot, Daniel Warner, Christoph Weddemann, Daniel Wonisch. A new approach for analyzing convergence algorithms for mobile robots. *ICALP 2011, pp. 650–661*, 2011. doi:10.1007/978-3-642-22012-8_52. Cited on pp. 468, 469, and 472.

Andreas Cord-Landwehr, Matthias Fischer, Daniel Jung, Friedhelm Meyer auf der Heide. Asymptotically Optimal Gathering on a Grid. *SPAA 2016, ACM, pp. 301–312*, 2016. doi:10.1145/2935764.2935789, arXiv:1602.03303 (pdf). Cited on p. 142.

Romain Cosson, Laurent Massoulié, Laurent Viennot. Efficient collaborative tree exploration with breadth-first depth-next. *DISC 2023 (LIPIcs 281), paper 14*, 2023. doi:10.4230/LIPIcs.DISC.2023.14, pdf, arXiv:2301.13307 (pdf). Other versions: *PODC 2023 brief announcement, pp. 24–27*, as "Brief Announcement: Efficient Collaborative Tree Exploration with Breadth-First Depth-Next" (doi:10.1145/3583668.3594568, pdf). Read: arXiv v1 (2023). Cited on pp. 59, 245, 246, 251, 252, 254, and 707.

Romain Cosson. Ariadne and Theseus: Exploration and Rendezvous with Two Mobile Agents in an Unknown Graph. *arXiv preprint*, 2024. arXiv:2403.07748 (pdf). Preprint, not peer reviewed. Cited on p. 120.

Romain Cosson. Breaking the k/log k barrier in collective tree exploration via tree-mining. *SODA 2024, pp. 4264–4282*, 2024. doi:10.1137/1.9781611977912.148, pdf, arXiv:2309.07011 (pdf). First appeared in 2023. Read: arXiv v2 (2023). Cited on pp. 59, 252, 253, and 255.

Romain Cosson, Laurent Massoulié. Collective tree exploration via potential function method. *ITCS 2024 (LIPIcs), paper 35*, 2024. doi:10.4230/LIPIcs.ITCS.2024.35, pdf, arXiv:2311.01354 (pdf). First appeared in 2023. Read: arXiv v1 (2023). Cited on pp. 59, 246, 252, 253, and 254.

Romain Cosson, Laurent Massoulié. Asynchronous collective tree exploration: a distributed algorithm, and a new lower bound. *arXiv preprint*, 2025. arXiv:2507.15658 (pdf). Preprint, not peer reviewed. Cited on pp. 78, 252, 254, 255, 645, and 709.

Romain Cosson, Laurent Massoulié. Polylogarithmic collective tree exploration. *arXiv preprint (September 2026)*, 2026. arXiv:2609.26789 (pdf). Preprint, not peer reviewed. Cited on pp. 59, 245, 246, 252, 254, 641, 645, and 711.

Pierre Courtieu, Lionel Rieg, Sébastien Tixeuil, Xavier Urbain. Certified Universal Gathering in $\mathbb{R}^2$ for Oblivious Mobile Robots. *DISC 2016, LNCS, pp. 187–200*, 2016. doi:10.1007/978-3-662-53426-7_14, arXiv:1602.08361 (pdf). First appeared in 2015. Read: arXiv 1506.01603 v1 (2015). Cited on p. 433.

Jurek Czyzowicz, Dariusz Kowalski, Euripides Markou, Andrzej Pelc. Complexity of Searching for a Black Hole. *Fundamenta Informaticae 71(2–3):229–242*, 2006. doi:10.3233/FUN-2006-712-305. Cited on pp. 291, 299, and 695.

Jurek Czyzowicz, Stefan Dobrev, Rastislav Královič, Stanislav Miklík, Dana Pardubská. Black hole search in directed graphs. *SIROCCO 2009, LNCS 5869, pp. 182–194*, 2009. doi:10.1007/978-3-642-11476-2_15. Cited on p. 292.

Jurek Czyzowicz, Leszek Gąsieniec, Andrzej Pelc. Gathering few fat mobile robots in the plane. *Theoretical Computer Science 410(6–7):481–499*, 2009. doi:10.1016/j.tcs.2008.10.005. Other versions: *OPODIS 2006* (doi:10.1007/11945529_25). First appeared in 2006. Cited on pp. 18, 19, 20, 25, 458, 675, and 694.

Jurek Czyzowicz, Leszek Gąsieniec, Adrian Kosowski, Evangelos Kranakis. Boundary Patrolling by Mobile Agents with Distinct Maximal Speeds. *ESA 2011, pp. 701–712*, 2011. doi:10.1007/978-3-642-23719-5_59. Cited on pp. 39, 41, 73, 566, 567, 568, 569, 571, 572, 573, 659, and 697.

Jurek Czyzowicz, Andrzej Pelc, Arnaud Labourel. How to meet asynchronously (almost) everywhere. *ACM Transactions on Algorithms 8(4), article 37*, 2012. doi:10.1145/2344422.2344427, pdf, arXiv:1001.0890 (pdf). Other versions: *SODA 2010* (doi:10.1137/1.9781611973075.3). First appeared in 2010. Cited on pp. 119, 121, and 124.

Jurek Czyzowicz, Adrian Kosowski, Andrzej Pelc. How to meet when you forget: log-space rendezvous in arbitrary graphs. *Distributed Computing 25(2):165–178*, 2012. doi:10.1007/s00446-011-0141-9. Other versions: *PODC 2010* (doi:10.1145/1835698.1835801). First appeared in 2010. Cited on pp. 53, 78, 118, 126, 127, 161, and 696.

Jurek Czyzowicz, Stefan Dobrev, Leszek Gąsieniec, David Ilcinkas, Jesper Jansson, Ralf Klasing, Ioannis Lignos, Russell Martin, Kunihiko Sadakane, Wing-Kin Sung. More efficient periodic traversal in anonymous undirected graphs. *Theoretical Computer Science 444:60–76*, 2012. doi:10.1016/j.tcs.2012.01.035, pdf, arXiv:0905.1737 (pdf). Other versions: *SIROCCO 2009* (doi:10.1007/978-3-642-11476-2_14, pdf). First appeared in 2009. Read: arXiv v1 (2009). Cited on pp. 57, 90, 100, 644, and 696.

Jurek Czyzowicz, Leszek Gąsieniec, Thomas Gorry, Evangelos Kranakis, Russell Martin, Dominik Pajak. Evacuating Robots via Unknown Exit in a Disk. *DISC 2014, pp. 122–136*, 2014. doi:10.1007/978-3-662-45174-8_9. Cited on pp. 38, 40, 41, 51, 63, 72, 78, 499, 505, 506, 507, 509, 510, 511, 517, 519, 522, 524, 525, 527, 528, 529, 542, 698, and 699.

Jurek Czyzowicz, Leszek Gąsieniec, Konstantinos Georgiou, Evangelos Kranakis, Fraser MacQuarrie. The Beachcombers' Problem: Walking and searching with mobile robots. *Theoretical Computer Science 608:201–218*, 2015. doi:10.1016/j.tcs.2015.09.011, arXiv:1304.7693 (pdf). Other versions: *SIROCCO 2014, LNCS, pp. 23–36* (doi:10.1007/978-3-319-09620-9_4). First appeared in 2013. Read: arXiv v1 (2013). Cited on p. 498.

Jurek Czyzowicz, Leszek Gąsieniec, Adrian Kosowski, Evangelos Kranakis, Oscar Morales-Ponce, Eduardo Pacheco. Position discovery for a system of bouncing robots. *Information and Computation 244:122–133*, 2015. doi:10.1016/j.ic.2015.07.005. First appeared in 2012. Read: the HAL version. Cited on p. 67.

J. Czyzowicz, E. Kranakis, D. Krizanc, L. Narayanan, J. Opatrny, S. Shende. Wireless Autonomous Robot Evacuation from Equilateral Triangles and Squares. *ADHOC-NOW 2015, LNCS, pp. 181–194*, 2015. doi:10.1007/978-3-319-19662-6_13. Cited on pp. 509, 535, 536, 537, 538, 539, 540, and 699.

Jurek Czyzowicz, Krzysztof Diks, Jean Moussi, Wojciech Rytter. Communication Problems for Mobile Agents Exchanging Energy. *SIROCCO 2016, LNCS, pp. 275–288*, 2016. doi:10.1007/978-3-319-48314-6_18, arXiv:1511.05987 (pdf). First appeared in 2015. Read: arXiv v1 (2015). Cited on pp. 333, 336, and 337.

Jurek Czyzowicz, Dariusz Dereniowski, Leszek Gąsieniec, Ralf Klasing, Adrian Kosowski, Dominik Pająk. Collision-free network exploration. *Journal of Computer and System Sciences 86:70–81*, 2017. doi:10.1016/j.jcss.2016.11.008, link. Other versions: *LATIN 2014, LNCS, pp. 342–354* (doi:10.1007/978-3-642-54423-1_30); *HAL report hal-00736276*, 2012 (link). First appeared in 2012. Read: HAL hal-00736276v1 (2012); theorem numbers refer to it. Cited on pp. 232, 233, 234, 235, 236, 241, 242, 243, 244, 675, 697, and 698.

Jurek Czyzowicz, Maxime Godon, Evangelos Kranakis, Arnaud Labourel, Euripides Markou. Exploring Graphs with Time Constraints by Unreliable Collections of Mobile Robots. *SOFSEM 2017, LNCS, pp. 381–395*, 2017. doi:10.1007/978-3-319-73117-9_27, arXiv:1710.00775 (pdf). Cited on p. 247.

Jurek Czyzowicz, Konstantinos Georgiou, Maxime Godon, Evangelos Kranakis, Danny Krizanc, Wojciech Rytter, Michał Włodarczyk. Evacuation from a Disc in the Presence of a Faulty Robot. *SIROCCO 2017, pp. 158–173*, 2017. doi:10.1007/978-3-319-72050-0_10. Cited on pp. 41, 64, 73, 75, 499, 507, 509, 522, 523, 525, and 701.

Jurek Czyzowicz, Evangelos Kranakis, Danny Krizanc, Lata Narayanan, Jaroslav Opatrny, Sunil Shende. Linear Search with Terrain-Dependent Speeds. *CIAC 2017, LNCS, pp. 430–441*, 2017. doi:10.1007/978-3-319-57586-5_36, arXiv:1701.03047 (pdf). Read: arXiv v1 (2017). Cited on p. 498.

Jurek Czyzowicz, Leszek Gasieniec, Adrian Kosowski, Evangelos Kranakis, Danny Krizanc, Najmeh Taleb. When Patrolmen Become Corrupted: Monitoring a Graph Using Faulty Mobile Robots. *Algorithmica 79(3):925–940*,

2017. doi:10.1007/s00453-016-0233-9. Other versions: *ISAAC 2015* (doi:10.1007/978-3-662-48971-0_30, pdf). First appeared in 2015. Cited on pp. 40, 73, 567, 568, 569, 574, 659, 660, and 700.

Jurek Czyzowicz, Stefan Dobrev, Konstantinos Georgiou, Evangelos Kranakis, Fraser MacQuarrie. Evacuating two robots from multiple unknown exits in a circle. *Theoretical Computer Science 709:20–30*, 2018. doi:10.1016/j.tcs.2016.11.019. Other versions: *ICDCN 2016* (doi:10.1145/2833312.2833318). First appeared in 2016. Read: the ICDCN 2016 version. Cited on pp. 511 and 518.

Jurek Czyzowicz, Ryan Killick, Evangelos Kranakis. Linear Rendezvous with Asymmetric Clocks. *OPODIS 2018, LIPIcs 125, 25:1–25:16*, 2018. doi:10.4230/LIPIcs.OPODIS.2018.25. Cited on p. 120.

Jurek Czyzowicz, Konstantinos Georgiou, Ryan Killick, Evangelos Kranakis, Danny Krizanc, Manuel Lafond, Lata Narayanan, Jaroslav Opatrny, Sunil Shende. Energy Consumption of Group Search on a Line. *ICALP 2019, LIPIcs 132, 137:1–137:15*, 2019. doi:10.4230/LIPIcs.ICALP.2019.137, pdf, arXiv:1904.09714 (pdf). Cited on pp. 73 and 535.

Jurek Czyzowicz, Krzysztof Diks, Jean Moussi, Wojciech Rytter. Energy-optimal broadcast and exploration in a tree using mobile agents. *Theoretical Computer Science 795:362–374*, 2019. doi:10.1016/j.tcs.2019.07.018. Other versions: *ALGOSENSORS 2017*. First appeared in 2017. Cited on p. 334.

Jurek Czyzowicz, Evangelos Kranakis, Danny Krizanc, Lata Narayanan, Jaroslav Opatrny. Search on a Line with Faulty Robots. *Distributed Computing 32(6):493–504*, 2019. doi:10.1007/s00446-017-0296-0. Other versions: *PODC 2016, pp. 405–414* (doi:10.1145/2933057.2933102). First appeared in 2016. Read: the PODC 2016 version. Cited on pp. 39, 40, 72, 75, 498, 499, 502, and 700.

Jurek Czyzowicz, Kostantinos Georgiou, Evangelos Kranakis. Group Search and Evacuation. *Lecture Notes in Computer Science Distributed Computing by Mobile Entities, 335-370*, 2019. doi:10.1007/978-3-030-11072-7_14. Cited on pp. 38, 507, and 523.

Jurek Czyzowicz, Maxime Godon, Evangelos Kranakis, Arnaud Labourel. Group search of the plane with faulty robots. *Theoretical Computer Science 792, pp. 69-84*, 2019. doi:10.1016/j.tcs.2018.09.029. Cited on p. 508.

Jurek Czyzowicz, Konstantinos Georgiou, Ryan Killick, Evangelos Kranakis, Danny Krizanc, Lata Narayanan, Jaroslav Opatrny, Sunil Shende. Priority evacuation from a disk: the case of $n = 1, 2, 3$. *Theoretical Computer Science 806:595–616*, 2020. doi:10.1016/j.tcs.2019.09.026, arXiv:1804.06011 (pdf). Other versions: *FUN 2018, LIPIcs 100, 16:1–16:20*, as "God Save the Queen" (doi:10.4230/LIPIcs.FUN.2018.16, pdf). First appeared in 2018. Read: the FUN 2018 version. Cited on pp. 73, 510, 521, 524, 527, and 703.

Jurek Czyzowicz, Konstantinos Georgiou, Ryan Killick, Evangelos Kranakis, Danny Krizanc, Lata Narayanan, Jaroslav Opatrny, Sunil Shende. Priority evacuation from a disk: the case of $n \geq 4$. *Theoretical Computer Science 846:91–102*, 2020. doi:10.1016/j.tcs.2020.09.023, arXiv:1805.03568 (pdf). Other versions: *SIROCCO 2018, pp. 392–407*, as "Priority Evacuation from a Disk Using Mobile Robots" (doi:10.1007/978-3-030-01325-7_32). First appeared in 2018. Read: arXiv v1 (2018). Cited on pp. 41, 73, 507, 510, 527, 528, and 703.

Jurek Czyzowicz, Dariusz Dereniowski, Andrzej Pelc. Building a Nest by an Automaton. *Algorithmica 83(1):144–176*, 2020. doi:10.1007/s00453-020-00752-0, arXiv:1904.10850 (pdf). Other versions: *ESA 2019, LIPIcs 144, 35:1–35:14* (doi:10.4230/LIPIcs.ESA.2019.35). First appeared in 2019. Cited on p. 632.

Jurek Czyzowicz, Konstantinos Georgiou, Evangelos Kranakis, Lata Narayanan, Jaroslav Opatrny, Birgit Vogtenhuber. Evacuating Robots from a Disk Using Face-to-Face Communication. *Discrete Mathematics & Theoretical Computer Science 22(4)*, 2020. doi:10.23638/DMTCS-22-4-4, arXiv:1501.04985 (pdf). Other versions: *CIAC 2015*, as "Evacuating Robots from a Disk Using Face-to-Face Communication (Extended Abstract)" (doi:10.1007/978-3-319-18173-8_10). First appeared in 2015. Read: arXiv v5 (2020). Cited on pp. 63, 73, 79, 507, 510, 519, 642, and 699.

Jurek Czyzowicz, Ryan Killick, Evangelos Kranakis, Danny Krizanc, Oscar Morales-Ponce. Gathering in the plane of location-aware robots in the presence of spies. *Theoretical Computer Science 836:94–109*, 2020. doi:10.1016/j.tcs.2020.06.034, arXiv:1712.02474 (pdf). First appeared in 2017. Read: arXiv v1 (2017). Cited on p. 433.

Jurek Czyzowicz, Stefan Dobrev, Maxime Godon, Evangelos Kranakis, Toshinori Sakai, Jorge Urrutia. Searching for a non-adversarial, uncooperative agent on a cycle. *Theoretical Computer Science 806:531–542*, 2020. doi:10.1016/j.tcs.2019.08.031. Other versions: *ALGOSENSORS 2017, LNCS, pp. 114–126*, as "Searching for a Non-adversarial, Uncooperative Agent on a Cycle" (doi:10.1007/978-3-319-72751-6_9). First appeared in 2017. Cited on p. 546.

Jurek Czyzowicz, Konstantinos Georgiou, Evangelos Kranakis, Danny Krizanc, Lata Narayanan, Jaroslav Opatrny, Sunil Shende. Search on a line by Byzantine robots. *International Journal of Foundations of Computer Science 32(4):369–387*, 2021. doi:10.1142/S0129054121500209, arXiv:1611.08209 (pdf). Other versions: *ISAAC 2016, LIPIcs 64,*

27:1–27:12 (doi:10.4230/LIPIcs.ISAAC.2016.27). First appeared in 2016. Read: arXiv v1 (2016). Cited on pp. 38, 39, 40, 41, 72, 75, 498, 499, 503, and 700.

Jurek Czyzowicz, Dariusz Dereniowski, Robert Ostrowski, Wojciech Rytter. Gossiping by energy-constrained mobile agents in tree networks. *Theoretical Computer Science 861:45–65*, 2021. doi:10.1016/j.tcs.2021.02.009. Cited on p. 334.

Jurek Czyzowicz, Stefan Dobrev, Ryan Killick, Evangelos Kranakis, Danny Krizanc, Lata Narayanan, Jaroslav Opatrny, Denis Pankratov, Sunil Shende. Graph Exploration by Energy-Sharing Mobile Agents. *SIROCCO 2021, LNCS, pp. 185–203*, 2021. doi:10.1007/978-3-030-79527-6_11, arXiv:2102.13062 (pdf). Cited on pp. 247, 248, and 334.

Jurek Czyzowicz, Ryan Killick, Evangelos Kranakis, Danny Krizanc, Lata Narayanan, Jaroslav Opatrny, Denis Pankratov, Sunil Shende. Group Evacuation on a Line by Agents with Different Communication Abilities. *ISAAC 2021, LIPIcs 212, 57:1–57:24*, 2021. doi:10.4230/LIPIcs.ISAAC.2021.57, pdf, arXiv:2109.12676 (pdf). Cited on pp. 39, 40, 73, 78, 499, 500, 532, 533, 546, 552, 564, 654, and 705.

Jurek Czyzowicz, Ryan Killick, Evangelos Kranakis, Grzegorz Stachowiak. Search and evacuation with a near majority of faulty agents. *SIAM Conference on Applied and Computational Discrete Algorithms (ACDA 2021)*, 2021. doi:10.1137/1.9781611976830.20, arXiv:2605.08355 (pdf). Read: arXiv v1 (2026). Cited on pp. 73, 500, 503, 534, 654, and 705.

Jurek Czyzowicz, Konstantinos Georgiou, Ryan Killick, Evangelos Kranakis, Danny Krizanc, Manuel Lafond, Lata Narayanan, Jaroslav Opatrny, Sunil Shende. Time-energy tradeoffs for evacuation by two robots in the wireless model. *Theoretical Computer Science 852, 61-72*, 2021. doi:10.1016/j.tcs.2020.11.014, arXiv:1905.06783 (pdf). Other versions: *SIROCCO 2019, LNCS, pp. 185–199* (doi:10.1007/978-3-030-24922-9_13). First appeared in 2019. Read: arXiv v1 (2019). Cited on p. 507.

Gianlorenzo D'Angelo, Gabriele Di Stefano, Alfredo Navarra, Nicolas Nisse, Karol Suchan. Computing on rings by oblivious robots: a unified approach for different tasks. *Algorithmica 72(4):1055–1096*, 2015. doi:10.1007/s00453-014-9892-6, link. Other versions: *IPDPS Workshops 2013, APDCM*. First appeared in 2013. Cited on pp. 173 and 174.

Gianlorenzo D'Angelo, Gabriele Di Stefano, Ralf Klasing, Alfredo Navarra. Gathering of robots on anonymous grids and trees without multiplicity detection. *Theoretical Computer Science 610:158–168*, 2016. doi:10.1016/j.tcs.2014.06.045. Other versions: *SIROCCO 2012, LNCS 7355, pp. 327–338*, as "Gathering of robots on anonymous grids without multiplicity detection". First appeared in 2012. Cited on pp. 140, 145, 154, 155, and 698.

Gianlorenzo D'Angelo, Mattia D'Emidio, Shantanu Das, Alfredo Navarra, Giuseppe Prencipe. Asynchronous Silent Programmable Matter Achieves Leader Election and Compaction. *IEEE Access 8:207619–207634*, 2020. doi:10.1109/ACCESS.2020.3038174, pdf. Other versions: *AAMAS 2020, pp. 276–284*, as "Leader Election and Compaction for Asynchronous Silent Programmable Matter" (pdf). Cited on pp. 43, 45, 46, 70, 584, 585, 586, 596, 597, 604, 659, and 704.

Mattia D'Emidio, Daniele Frigioni, Alfredo Navarra. Explore and repair graphs with black holes using mobile entities. *Theoretical Computer Science 605:129–145*, 2015. doi:10.1016/j.tcs.2015.09.002. Other versions: *ADHOC-NOW 2013, LNCS 7960, pp. 136–147*, as "Exploring and making safe dangerous networks using mobile entities". First appeared in 2013. Cited on p. 292.

Mattia D'Emidio, Daniele Frigioni, Alfredo Navarra. Synchronous Robots vs Asynchronous Lights-Enhanced Robots on Graphs. *ICTCS 2015, Electronic Notes in Theoretical Computer Science 322:169–180*, 2016. doi:10.1016/j.entcs.2016.03.012. First appeared in 2015. Cited on p. 67.

Mattia D'Emidio, Gabriele Di Stefano, Daniele Frigioni, Alfredo Navarra. Characterizing the computational power of mobile robots on graphs and implications for the Euclidean plane. *Information and Computation 263:57–74*, 2018. doi:10.1016/j.ic.2018.09.010. Other versions: *ICDCS 2016, IEEE, pp. 293–302*, as "Characterizing the computational power of anonymous mobile robots"; *IWOCA 2015, LNCS 9538, pp. 136–147*, as "About ungatherability of oblivious and asynchronous robots on anonymous rings". First appeared in 2015. Cited on pp. 67, 150, 665, and 700.

Yassine Daadaa, Asif Jamshed, Mudassir Shabbir. Network Decontamination with a Single Agent. *Graphs and Combinatorics 32(2):559–581*, 2016. doi:10.1007/s00373-015-1579-5. First appeared in 2013. Cited on p. 319.

Omar Darwich, Ahmet-Sefa Ulucan, Quentin Bramas, Anissa Lamani, Anaïs Durand, Pascal Lafourcade. Perpetual torus exploration by myopic luminous robots. *Theoretical Computer Science 976:114143*, 2023. doi:10.1016/j.tcs.2023.114143, pdf, link. Other versions: *SSS 2022, LNCS, pp. 164–177* (doi:10.1007/978-3-031-21017-4_11, pdf). First appeared in 2022. Cited on pp. 177, 180, 181, 201, 202, 203, 680, and 706.

Shantanu Das, Paola Flocchini, Shay Kutten, Amiya Nayak, Nicola Santoro. Map construction of unknown graphs by multiple agents. *Theoretical Computer Science 385(1–3):34–48*, 2007. doi:10.1016/j.tcs.2007.05.011. Cited on pp. 15, 158, 159, 162, and 695.

Shantanu Das, Flaminia L. Luccio, Euripides Markou. Mobile Agents Rendezvous in Spite of a Malicious Agent. *ALGOSENSORS 2015, LNCS, pp. 211–224*, 2015. doi:10.1007/978-3-319-28472-9_16, arXiv:1410.4772 (pdf). First appeared in 2014. Cited on p. 120.

Shantanu Das, Paola Flocchini, Nicola Santoro, Masafumi Yamashita. Forming sequences of geometric patterns with oblivious mobile robots. *Distributed Computing 28(2):131–145*, 2015. doi:10.1007/s00446-014-0220-9. Other versions: *PODC 2010*, as "On the computational power of oblivious robots: forming a series of geometric patterns" (doi:10.1145/1835698.1835761). First appeared in 2010. Cited on pp. 26, 54, 66, 407, 408, 409, 410, 656, and 696.

Shantanu Das, Paola Flocchini, Giuseppe Prencipe, Nicola Santoro, Masafumi Yamashita. Autonomous mobile robots with lights. *Theoretical Computer Science 609:171–184*, 2016. doi:10.1016/j.tcs.2015.09.018, pdf. Other versions: *ICDCS 2012*. First appeared in 2012. Cited on pp. 18, 20, 21, 27, 28, 34, 51, 63, 66, 67, 177, 178, 179, 207, 432, 446, 679, and 697.

Deepanwita Das, Srabani Mukhopadhyaya. Distributed algorithm for painting by a swarm of randomly deployed robots under limited visibility model. *International Journal of Advanced Robotic Systems 15(5)*, 2018. doi:10.1177/1729881418804508. Cited on p. 67.

Shantanu Das, Giuseppe Antonio Di Luna, Linda Pagli, Giuseppe Prencipe. Compacting and Grouping Mobile Agents on Dynamic Rings. *TAMC 2019, LNCS, pp. 114–133*, 2019. doi:10.1007/978-3-030-14812-6_8, pdf. Cited on p. 134.

Shantanu Das, Riccardo Focardi, Flaminia L. Luccio, Euripides Markou, Marco Squarcina. Gathering of robots in a ring with mobile faults. *Theoretical Computer Science 764:42–60*, 2019. doi:10.1016/j.tcs.2018.05.002. Other versions: *ICTCS 2016, CEUR Workshop Proceedings 1720* (pdf). First appeared in 2016. Read: ICTCS 2016 version (abstract). Cited on p. 133.

Shantanu Das, Giuseppe A. Di Luna, Leszek A. Gąsieniec. Patrolling on Dynamic Ring Networks. *SOFSEM 2019, LNCS, pp. 150–163*, 2019. doi:10.1007/978-3-030-10801-4_13, arXiv:1808.04349 (pdf). First appeared in 2018. Read: arXiv v1 (2018). Cited on p. 568.

Shantanu Das. Graph Explorations with Mobile Agents. *Distributed Computing by Mobile Entities, LNCS 11340, pp. 403–422*, 2019. doi:10.1007/978-3-030-11072-7_16, pdf. Cited on pp. 88, 158, and 161.

Shantanu Das, Giuseppe A. Di Luna, Paola Flocchini, Nicola Santoro, Giovanni Viglietta, Masafumi Yamashita. Oblivious Permutations on the Plane. *OPODIS 2019, LIPIcs 153, 24:1–24:16*, 2019. doi:10.4230/LIPIcs.OPODIS.2019.24, link, arXiv:1911.05239 (pdf). Read: arXiv v1 (2019). Cited on p. 409.

Shantanu Das, Paola Flocchini, Giuseppe Prencipe, Nicola Santoro. Forming Sequences of Patterns With Luminous Robots. *IEEE Access 8:90577–90597*, 2020. doi:10.1109/ACCESS.2020.2994052, pdf. First appeared in 2014. Cited on pp. 408, 409, 411, 656, and 699.

Shantanu Das, Giuseppe Antonio Di Luna, Daniele Mazzei, Giuseppe Prencipe. Compacting oblivious agents on dynamic rings. *PeerJ Computer Science 7:e466*, 2021. doi:10.7717/peerj-cs.466. Cited on p. 134.

Archak Das, Avisek Sharma, Buddhadeb Sau. Power of Finite Memory and Finite Communication Robots under Asynchronous Scheduler. *arXiv preprint*, 2021. arXiv:2104.12202 (pdf). Preprint, not peer reviewed. Cited on pp. 27, 29, 35, and 37.

Deepanwita Das, Srabani Mukhopadhyaya, Mrityunjay Ghosh. Synchronizing asynchronous mobile robots with limited visibility using lights. *ICMAI 2021, Journal of Physics: Conference Series 1950:012015*, 2021. doi:10.1088/1742-6596/1950/1/012015. Cited on p. 67.

Archak Das, Kaustav Bose, Buddhadeb Sau. Exploring a dynamic ring without landmark. *Theoretical Computer Science 929:191–205*, 2022. doi:10.1016/j.tcs.2022.07.005, arXiv:2107.02769 (pdf). Other versions: *SSS 2021* (doi:10.1007/978-3-030-91081-5_21). First appeared in 2021. Read: arXiv v1 (2021). Cited on p. 309.

Archak Das, Kaustav Bose, Buddhadeb Sau. Memory Optimal Dispersion by Anonymous Mobile Robots. *Discrete Applied Mathematics 340:171–182*, 2023. doi:10.1016/j.dam.2023.07.005, arXiv:2008.00701 (pdf). Other versions: *CALDAM 2021* (doi:10.1007/978-3-030-67899-9_34). First appeared in 2020. Read: arXiv v2 (2020). Cited on pp. 53, 54, 55, 282, 648, and 704.

Archak Das, Satakshi Ghosh, Avisek Sharma, Pritam Goswami, Buddhadeb Sau. The Computational Landscape of Autonomous Mobile Robots: The Visibility Perspective. *ICDCIT 2024, LNCS, pp. 85–100*, 2024. doi:10.1007/978-3-031-50583-6_6, arXiv:2303.03031 (pdf). First appeared in 2023. Read: arXiv v3 (2023). Cited on pp. 21, 23, 24, and 36.

Sajal K. Das, Amit Kumar Dhar, Barun Gorain, Madhuri Mahawar. Collision-free exploration by mobile agents

using pebbles. *Information and Computation 307:105380*, 2025. doi:10.1016/j.ic.2025.105380, arXiv:2410.17542 (pdf). Other versions: *ICDCN 2025, ACM, pp. 161–170* (doi:10.1145/3700838.3700863). First appeared in 2024. Read: arXiv v1 (2024); theorem numbers refer to it. Cited on pp. 235, 236, 238, 239, 240, 247, 666, 671, 683, 684, and 709.

Raja Das, Avisek Sharma, Buddhadeb Sau. Maximum independent set formation on a finite grid by myopic robots. *Theoretical Computer Science 1031:115077*, 2025. doi:10.1016/j.tcs.2025.115077, arXiv:2207.13403 (pdf). Other versions: *Intelligent Computing 2023, LNNS, pp. 1247–1267* (doi:10.1007/978-3-031-37963-5_86). First appeared in 2022. Cited on p. 177.

Raja Das, Pritam Goswami, Brati Mondal, Buddhadeb Sau. Rectangle enclosed triangular grid exploration with myopic luminous robots. *ICDCN 2025, pp. 249–253*, 2025. doi:10.1145/3700838.3700841. Cited on pp. 181, 204, 205, and 710.

Bibhuti Das, Barun Gorain, Kaushik Mondal, Krishnendu Mukhopadhyaya, Supantha Pandit. Optimal Dispersion of Silent Robots in a Ring. *Theoretical Computer Science 1084:116171*, 2026. doi:10.1016/j.tcs.2026.116171, arXiv:2408.05491 (pdf). Other versions: *SSS 2025 (LNCS)* (doi:10.1007/978-3-032-11127-2_15). First appeared in 2024. Read: arXiv v1 (2024). Cited on pp. 78, 270, 276, 646, and 708.

Shantanu Das, Nikos Giachoudis, Flaminia L. Luccio, Euripides Markou. On the Broadcast problem for mobile agents in dynamic networks. *Discrete Applied Mathematics 379:194–208*, 2026. doi:10.1016/j.dam.2025.08.031, link. Other versions: *OPODIS 2020, LIPIcs 184, 24:1–24:16*, as "Broadcasting with Mobile Agents in Dynamic Networks" (doi:10.4230/LIPIcs.OPODIS.2020.24, pdf). First appeared in 2020. Read: HAL hal-05380085v1 (the journal version): model, theorem and lemma statements, and the proofs of Observation 3.4, Theorem 4.1 and Lemmas 5.4, 5.5 and 6.2. Cited on pp. 7, 323, 324, 325, 326, 327, 328, 329, 330, 331, 665, and 704.

Bibhuti Das, Andrzej Pelc. Rendezvous of Mobile Deterministic Automata in Graphs. *arXiv preprint*, 2026. arXiv:2608.06482 (pdf). Preprint, not peer reviewed. Cited on p. 119.

Bibhuti Das. Universal Rendezvous of Anonymous Agents with Footprints. *arXiv preprint*, 2026. arXiv:2608.08981 (pdf). Preprint, not peer reviewed. Cited on p. 119.

Ajoy K. Datta, Anissa Lamani, Lawrence L. Larmore, Franck Petit. Ring exploration by oblivious agents with local vision. *ICDCS 2013, pp. 347–356*, 2013. doi:10.1109/ICDCS.2013.55, arXiv:1205.5003 (pdf). First appeared in 2012. Cited on pp. 176, 178, 179, 180, 181, 198, 199, 200, 201, 670, 686, and 698.

Ajoy K. Datta, Anissa Lamani, Lawrence L. Larmore, Franck Petit. Enabling ring exploration with myopic oblivious robots. *IPDPS Workshops 2015 (APDCM), pp. 490–499*, 2015. doi:10.1109/IPDPSW.2015.137. Cited on p. 178.

Joshua J. Daymude, Robert Gmyr, Andréa W. Richa, Christian Scheideler, Thim Strothmann. Improved Leader Election for Self-organizing Programmable Matter. *ALGOSENSORS 2017, pp. 127–140*, 2017. doi:10.1007/978-3-319-72751-6_10, arXiv:1701.03616 (pdf). Read: arXiv v3 (2017). Cited on pp. 70, 586, 589, and 702.

Joshua J. Daymude, Zahra Derakhshandeh, Robert Gmyr, Alexandra Porter, Andréa W. Richa, Christian Scheideler, Thim Strothmann. On the runtime of universal coating for programmable matter. *Natural Computing 17:81–96*, 2018. doi:10.1007/s11047-017-9658-6, arXiv:1606.03642 (pdf). Other versions: *DNA 22, LNCS, pp. 148–164*, 2016 (doi:10.1007/978-3-319-43994-5_10). First appeared in 2016. Read: arXiv v4 (2017). Cited on pp. 605, 606, 607, and 701.

Joshua J. Daymude, Robert Gmyr, Kristian Hinnenthal, Irina Kostitsyna, Christian Scheideler, Andréa W. Richa. Convex Hull Formation for Programmable Matter. *ICDCN 2020, 2:1–2:10*, 2020. doi:10.1145/3369740.3372916, arXiv:1805.06149 (pdf). First appeared in 2018. Read: arXiv v2 (2019). Cited on pp. 619, 620, 621, 622, 702, and 703.

Joshua J. Daymude, Andréa W. Richa, Jamison W. Weber. Bio-Inspired Energy Distribution for Programmable Matter. *ICDCN 2021, pp. 86–95*, 2021. doi:10.1145/3427796.3427835, arXiv:2007.04377 (pdf). First appeared in 2020. Read: arXiv v2 (2020). Cited on pp. 627, 628, 629, and 704.

Joshua J. Daymude, Andréa W. Richa, Christian Scheideler. The canonical amoebot model: algorithms and concurrency control. *Distributed Computing 36:159–192*, 2023. doi:10.1007/s00446-023-00443-3, arXiv:2105.02420 (pdf). Other versions: *DISC 2021, LIPIcs 209, 20:1–20:19* (doi:10.4230/LIPIcs.DISC.2021.20, pdf). First appeared in 2021. Read: arXiv v2 (2022). Cited on pp. 42, 43, 44, 45, 46, 51, 69, and 594.

Jean-Lou De Carufel, Paola Flocchini. Fault-Induced Dynamics of Oblivious Robots on a Line. *SSS 2017, LNCS, pp. 126–141*, 2017. doi:10.1007/978-3-319-69084-1_9, arXiv:1707.03492 (pdf). Cited on p. 470.

Gianluca De Marco, Luisa Gargano, Evangelos Kranakis, Danny Krizanc, Andrzej Pelc, Ugo Vaccaro. Asynchronous deterministic rendezvous in graphs. *Theoretical Computer Science 355(3):315–326*, 2006. doi:10.1016/j.tcs.2005.12.016. Other versions: *MFCS 2005* (doi:10.1007/11549345_24). First appeared in 2005. Cited on pp. 7, 14, 63, 119, 124, and 694.

Bastian Degener, Barbara Kempkes, Tobias Langner, Friedhelm Meyer auf der Heide, Peter Pietrzyk, Roger Wattenhofer. A tight runtime bound for synchronous gathering of autonomous robots with limited visibility. *SPAA 2011, pp. 139–148*, 2011. doi:10.1145/1989493.1989515. Cited on pp. 432, 457, and 697.

Carole Delporte-Gallet, Hugues Fauconnier, Rachid Guerraoui, Anne-Marie Kermarrec, Eric Ruppert, Hung Tran-The. Byzantine agreement with homonyms. *Distributed Computing 26(5–6):321–340*, 2013. doi:10.1007/s00446-013-0190-3. Other versions: *PODC 2011* (doi:10.1145/1993806.1993810, pdf); *SPAA 2010 brief announcement, pp. 74–75*, as "Brief announcement: byzantine agreement with homonyms" (doi:10.1145/1810479.1810494). First appeared in 2010. Cited on pp. 12, 26, 54, 445, and 677.

Xiaotie Deng, Christos H. Papadimitriou. Exploring an unknown graph. *Journal of Graph Theory 32(3):265–297*, 1999. doi:10.1002/(SICI)1097-0118(199911)32:3<265::AID-JGT6>3.0.CO;2-8. Other versions: *FOCS 1990* (doi:10.1109/fscs.1990.89554). First appeared in 1990. Cited on pp. 11, 15, 88, 102, 103, 225, and 691.

Zahra Derakhshandeh, Shlomi Dolev, Robert Gmyr, Andréa W. Richa, Christian Scheideler, Thim Strothmann. Amoebot - a new model for programmable matter. *SPAA 2014 brief announcement, pp. 220–222*, 2014. doi:10.1145/2612669.2612712. Cited on pp. 42, 44, 45, 46, 51, and 69.

Zahra Derakhshandeh, Robert Gmyr, Andréa W. Richa, Christian Scheideler, Thim Strothmann. An Algorithmic Framework for Shape Formation Problems in Self-Organizing Particle Systems. *NANOCOM 2015, ACM, pp. 1–2*, 2015. doi:10.1145/2800795.2800829, arXiv:1504.00744 (pdf). Cited on p. 600.

Zahra Derakhshandeh, Robert Gmyr, Thim Strothmann, Rida Bazzi, Andréa W. Richa, Christian Scheideler. Leader Election and Shape Formation with Self-organizing Programmable Matter. *DNA 21 (2015), pp. 117–132*, 2015. doi:10.1007/978-3-319-21999-8_8, arXiv:1503.07991 (pdf). Other versions: *PODC 2015 brief announcement, pp. 67–69*, as "Brief Announcement: On the Feasibility of Leader Election and Shape Formation with Self-Organizing Programmable Matter" (doi:10.1145/2767386.2767451). Read: arXiv v2 (2016). Cited on pp. 42, 44, 45, 46, 47, 51, 54, 69, 77, 583, 584, 585, 586, 589, 598, 600, 603, 699, and 700.

Zahra Derakhshandeh, Robert Gmyr, Andrea W. Richa, Christian Scheideler, Thim Strothmann. Universal Shape Formation for Programmable Matter. *SPAA 2016, pp. 289–299*, 2016. doi:10.1145/2935764.2935784. Cited on pp. 69, 77, 598, 599, 600, 602, and 701.

Zahra Derakhshandeh, Robert Gmyr, Andréa W. Richa, Christian Scheideler, Thim Strothmann. Universal coating for programmable matter. *Theoretical Computer Science 671:56–68*, 2017. doi:10.1016/j.tcs.2016.02.039, arXiv:1601.01008 (pdf). First appeared in 2016. Read: arXiv v1 (2016). Cited on pp. 77, 604, 605, 606, 607, 698, and 701.

Dariusz Dereniowski, Yann Disser, Adrian Kosowski, Dominik Pająk, Przemysław Uznański. Fast collaborative graph exploration. *Information and Computation 243:37–49*, 2015. doi:10.1016/j.ic.2014.12.005, link. Other versions: *ICALP 2013* (doi:10.1007/978-3-642-39212-2_46, pdf). First appeared in 2013. Cited on pp. 246, 248, 252, 254, 256, 645, and 698.

Dariusz Dereniowski, Ralf Klasing, Adrian Kosowski, Łukasz Kuszner. Rendezvous of heterogeneous mobile agents in edge-weighted networks. *Theoretical Computer Science 608:219–230*, 2015. doi:10.1016/j.tcs.2015.05.055, arXiv:1406.2008 (pdf). Other versions: *SIROCCO 2014, LNCS, pp. 311–326* (doi:10.1007/978-3-319-09620-9_24). First appeared in 2014. Cited on p. 120.

Dariusz Dereniowski, Adrian Kosowski, Dominik Pająk, Przemysław Uznański. Bounds on the cover time of parallel rotor walks. *Journal of Computer and System Sciences 82(5):802–816*, 2016. doi:10.1016/j.jcss.2016.01.004. Read: the HAL version. Cited on p. 208.

Anders Dessmark, Pierre Fraigniaud, Dariusz R. Kowalski, Andrzej Pelc. Deterministic rendezvous in graphs. *Algorithmica 46(1):69–96*, 2006. doi:10.1007/s00453-006-0074-2, link. Other versions: *ESA 2003* (doi:10.1007/978-3-540-39658-1_19). First appeared in 2003. Cited on pp. 7, 12, 15, 54, 62, 78, 116, 117, 118, 121, 124, 125, 126, 129, 130, 131, 132, 641, and 693.

Henri Devillez, Béni Egressy, Robin Fritsch, Roger Wattenhofer. Two-Agent Tree Evacuation. *SIROCCO 2021, LNCS, pp. 204–221*, 2021. doi:10.1007/978-3-030-79527-6_12. Cited on p. 507.

Stéphane Devismes. Optimal exploration of small rings. *WRAS 2010 (Third International Workshop on Reliability, Availability, and Security), 9:1–9:6*, 2010. doi:10.1145/1953563.1953571, link. Cited on p. 172.

Stéphane Devismes, Franck Petit, Sébastien Tixeuil. Optimal probabilistic ring exploration by semi-synchronous oblivious robots. *Theoretical Computer Science 498:10–27*, 2013. doi:10.1016/j.tcs.2013.05.031, link, arXiv:0902.1834 (pdf). Other versions: *SIROCCO 2009, LNCS 5869, pp. 195–208* (doi:10.1007/978-3-642-11476-2_16). First appeared in 2009. Cited on pp. 164, 165, 166, 169, 170, 171, 172, 660, and 696.

Stéphane Devismes, Anissa Lamani, Franck Petit, Sébastien Tixeuil. Optimal torus exploration by oblivious

robots. *Computing 101(9):1241–1264*, 2019. doi:10.1007/s00607-018-0595-8, pdf, link. Other versions: *NETYS 2015* (doi:10.1007/978-3-319-26850-7_13, pdf). First appeared in 2015. Cited on p. 165.

Stéphane Devismes, Anissa Lamani, Franck Petit, Pascal Raymond, Sébastien Tixeuil. Terminating Exploration Of A Grid By An Optimal Number Of Asynchronous Oblivious Robots. *The Computer Journal 64(1):132–154*, 2021. doi:10.1093/comjnl/bxz166, pdf, arXiv:1105.2461 (pdf). Other versions: *SSS 2012, pp. 64–76*, as "Optimal Grid Exploration by Asynchronous Oblivious Robots" (doi:10.1007/978-3-642-33536-5_7). First appeared in 2011. Read: arXiv v3 (2012). Cited on pp. 164, 165, 170, 176, 180, and 697.

Stéphane Devismes, Yoann Dieudonné, Arnaud Labourel. Graph Exploration: The Impact of a Distance Constraint. *Algorithmica 88(2):24*, 2026. doi:10.1007/s00453-026-01378-4, arXiv:2410.13386 (pdf). First appeared in 2024. Cited on p. 86.

Stéphane Devismes, Yoann Dieudonné, Arnaud Labourel. Can Like Attract Like? A Study of Homonymous Gathering in Networks. *STOC 2026, pp. 933–942*, 2026. doi:10.1145/3798129.3800807, arXiv:2510.25451 (pdf). First appeared in 2025. Read: arXiv v2 (2025). Cited on pp. 7, 12, 54, 62, 63, 137, and 710.

Giuseppe Antonio Di Luna, Paola Flocchini, Sruti Gan Chaudhuri, Nicola Santoro, Giovanni Viglietta. Robots with Lights: Overcoming Obstructed Visibility Without Colliding. *SSS 2014, LNCS 8756, pp. 150–164*, 2014. doi:10.1007/978-3-319-11764-5_11, arXiv:1405.2430 (pdf). Read: arXiv v1 (2014). Cited on pp. 18, 21, 25, 65, 66, 412, 417, 420, and 698.

Giuseppe A. Di Luna, Paola Flocchini, Sruti Gan Chaudhuri, Federico Poloni, Nicola Santoro, Giovanni Viglietta. Mutual visibility by luminous robots without collisions. *Information and Computation 254:392–418*, 2017. doi:10.1016/j.ic.2016.09.005, arXiv:1503.04347 (pdf). First appeared in 2015. Read: arXiv v2 (2015). Cited on pp. 25, 66, 67, 412, 413, 416, 417, 418, 419, 420, 421, 422, 657, 698, and 699.

Giuseppe Antonio Di Luna, Paola Flocchini, Giuseppe Prencipe, Nicola Santoro, Giovanni Viglietta. Line Recovery by Programmable Particles. *ICDCN 2018, ACM, pp. 1–10*, 2018. doi:10.1145/3154273.3154309, arXiv:1707.05041 (pdf). First appeared in 2017. Read: arXiv v1 (2017). Cited on p. 600.

Giuseppe Antonio Di Luna, Paola Flocchini, Nicola Santoro, Giovanni Viglietta, Masafumi Yamashita. Meeting in a polygon by anonymous oblivious robots. *Distributed Computing 33(5):445–469*, 2020. doi:10.1007/s00446-019-00362-2, arXiv:1705.00324 (pdf). Other versions: *DISC 2017, LIPIcs 91, 14:1–14:15* (doi:10.4230/LIPIcs.DISC.2017.14). First appeared in 2017. Cited on p. 433.

Giuseppe A. Di Luna, Stefan Dobrev, Paola Flocchini, Nicola Santoro. Distributed exploration of dynamic rings. *Distributed Computing 33(1):41–67*, 2020. doi:10.1007/s00446-018-0339-1, arXiv:1512.05306 (pdf). Other versions: *ICDCS 2016*, as "Live Exploration of Dynamic Rings" (doi:10.1109/icdcs.2016.59). First appeared in 2015. Read: arXiv v4 (2018). Cited on pp. 7, 10, 13, 14, 15, 305, 306, 309, 656, 699, and 700.

Giuseppe Antonio Di Luna, Paola Flocchini, Linda Pagli, Giuseppe Prencipe, Nicola Santoro, Giovanni Viglietta. Gathering in dynamic rings. *Theoretical Computer Science 811:79–98*, 2020. doi:10.1016/j.tcs.2018.10.018, arXiv:1704.02427 (pdf). Other versions: *SIROCCO 2017, LNCS, pp. 339–355* (doi:10.1007/978-3-319-72050-0_20). First appeared in 2017. Cited on p. 133.

Giuseppe Antonio Di Luna, Paola Flocchini, Nicola Santoro, Giovanni Viglietta, Yukiko Yamauchi. Mobile RAM and Shape Formation by Programmable Particles. *Euro-Par 2020, LNCS, pp. 343–358*, 2020. doi:10.1007/978-3-030-57675-2_22, arXiv:2002.03384 (pdf). Read: arXiv v2 (2020). Cited on p. 600.

Giuseppe A. Di Luna, Paola Flocchini, Nicola Santoro, Giovanni Viglietta, Yukiko Yamauchi. Shape formation by programmable particles. *Distributed Computing 33:69–101*, 2020. doi:10.1007/s00446-019-00350-6, arXiv:1705.03538 (pdf). Other versions: *OPODIS 2017, LIPIcs 95, 31:1–31:16* (doi:10.4230/LIPIcs.OPODIS.2017.31); *DISC 2017 (brief announcement), LIPIcs 91, 48:1–48:3*, as "Brief Announcement: Shape Formation by Programmable Particles" (doi:10.4230/LIPIcs.DISC.2017.48). First appeared in 2017. Read: arXiv v3 (2017). Cited on pp. 42, 43, 44, 45, 46, 47, 51, 69, 77, 384, 584, 597, 598, 599, 600, 601, 602, 603, and 702.

Giuseppe A. Di Luna, Paola Flocchini, Nicola Santoro, Giovanni Viglietta. TuringMobile: a turing machine of oblivious mobile robots with limited visibility and its applications. *Distributed Computing 35(2):105–122*, 2021. doi:10.1007/s00446-021-00406-6, arXiv:1709.08800 (pdf). Other versions: *DISC 2018, LIPIcs 121, 19:1–19:18* (doi:10.4230/LIPIcs.DISC.2018.19). First appeared in 2017. Read: arXiv v2 (2018). Cited on pp. 67 and 462.

Giuseppe A. Di Luna, Paola Flocchini, Giuseppe Prencipe, Nicola Santoro. Black Hole Search in Dynamic Rings: The Scattered Case. *OPODIS 2023, LIPIcs 286, 33:1–33:18*, 2023. doi:10.4230/LIPIcs.OPODIS.2023.33, pdf, arXiv:2404.15132 (pdf). Read: the OPODIS version and arXiv v4 (July 2026), titled "Black Hole Search by Scattered Agents in Dynamic Rings"; theorem numbers refer to arXiv v4. Cited on p. 303.

Giuseppe Antonio Di Luna, Paola Flocchini, Debasish Pattanayak, Giuseppe Prencipe, Francesco Piselli, Nicola Santoro. Exploring Dangerous Graphs with Byzantine Companions. *ICDCS 2025, pp. 352–361*, 2025. doi:10.1109/ICDCS63083.2025.00042. Cited on pp. 75, 291, 302, 303, and 709.

Giuseppe Antonio Di Luna, Paola Flocchini, Giuseppe Prencipe, Nicola Santoro. Locating a black hole in a dynamic ring. *Journal of Parallel and Distributed Computing 196:104998*, 2025. doi:10.1016/j.jpdc.2024.104998, arXiv:2005.07453 (pdf). Other versions: *ICDCS 2021*, as "Black Hole Search in Dynamic Rings" (doi:10.1109/icdcs51616.2021.00098). First appeared in 2020. Read: arXiv v2 (2020). Cited on pp. 6, 13, 14, 16, 50, 74, 78, 291, 303, and 704.

Giuseppe Antonio Di Luna, Ryuhei Uehara, Giovanni Viglietta, Yukiko Yamauchi. Gathering on a circle with limited visibility by anonymous oblivious robots. *Theoretical Computer Science 1025:114974*, 2025. doi:10.1016/j.tcs.2024.114974, arXiv:2005.07917 (pdf). Other versions: *DISC 2020, LIPIcs 179, 12:1–12:17* (doi:10.4230/LIPIcs.DISC.2020.12). First appeared in 2020. Read: arXiv v3 (2024); theorem numbers refer to it. Cited on p. 433.

Yoann Dieudonné, Ouiddad Labbani-Igbida, Franck Petit. Circle formation of weak mobile robots. *ACM Transactions on Autonomous and Adaptive Systems 3(4), article 16*, 2008. doi:10.1145/1452001.1452006, arXiv:cs/0607060 (pdf). Other versions: *SSS 2006* (doi:10.1007/978-3-540-49823-0_18). First appeared in 2006. Read: arXiv v1 (2006). Cited on pp. 65, 397, 399, 401, 402, and 694.

Yoann Dieudonné, Franck Petit. Scatter of robots. *Parallel Processing Letters 19(1):175–184*, 2009. doi:10.1142/S0129626409000146, arXiv:cs/0701179 (pdf). First appeared in 2007. Read: arXiv v1 (2007). Cited on pp. 18, 25, 53, 54, 70, 473, 474, 475, 476, 477, 695, and 696.

Yoann Dieudonné, Franck Petit, Vincent Villain. Leader election problem versus pattern formation problem. *DISC 2010, pp. 267–281*, 2010. doi:10.1007/978-3-642-15763-9_26, arXiv:0902.2851 (pdf). Other versions: *PODC 2010 brief announcement, pp. 404–405*, as "Brief announcement: leader election vs pattern formation" (doi:10.1145/1835698.1835793). First appeared in 2009. Read: arXiv v3 (2012). Cited on pp. 18, 22, 26, 54, 65, 380, 381, 384, 386, 387, 481, and 482.

Yoann Dieudonné, Andrzej Pelc. Deterministic network exploration by a single agent with Byzantine tokens. *Information Processing Letters 112(12):467–470*, 2012. doi:10.1016/j.ipl.2012.03.017, pdf, link. Cited on pp. 57, 84, 94, and 697.

Yoann Dieudonné, Florence Levé, Franck Petit, Vincent Villain. Deterministic geoleader election in disoriented anonymous systems. *Theoretical Computer Science 506:43–54*, 2013. doi:10.1016/j.tcs.2013.07.033, arXiv:1202.4486 (pdf). First appeared in 2012. Read: arXiv v1 (2012). Cited on pp. 53, 54, 66, 70, 77, 480, 481, 483, 695, and 697.

Yoann Dieudonné, Andrzej Pelc, David Peleg. Gathering Despite Mischief. *ACM Transactions on Algorithms 11(1)*, 2014. doi:10.1145/2629656. Other versions: *SODA 2012* (doi:10.1137/1.9781611973099.45). First appeared in 2012. Cited on pp. 6, 7, 13, 75, 131, 132, 133, 137, 138, 139, and 697.

Yoann Dieudonné, Andrzej Pelc, Vincent Villain. How to meet asynchronously at polynomial cost. *SIAM Journal on Computing 44(3):844–867*, 2015. doi:10.1137/130931990, arXiv:1301.7119 (pdf). Other versions: *PODC 2013* (doi:10.1145/2484239.2484245). First appeared in 2013. Cited on pp. 63, 119, 124, and 698.

Yoann Dieudonné, Andrzej Pelc. Anonymous Meeting in Networks. *Algorithmica 74(2):908–946*, 2016. doi:10.1007/s00453-015-9982-0, arXiv:1111.0321 (pdf). Other versions: *SODA 2013* (doi:10.1137/1.9781611973105.53). First appeared in 2011. Cited on pp. 11, 12, 16, 53, 62, 77, 132, 133, 136, and 697.

Yoann Dieudonné, Andrzej Pelc, Franck Petit. Almost Universal Anonymous Rendezvous in the Plane. *Algorithmica 85(10):3110–3143*, 2023. doi:10.1007/s00453-023-01122-2, arXiv:2005.00880 (pdf). Other versions: *SPAA 2020, ACM, pp. 117–127* (doi:10.1145/3350755.3400283). First appeared in 2020. Cited on p. 120.

Krzysztof Diks, Pierre Fraigniaud, Evangelos Kranakis, Andrzej Pelc. Tree exploration with little memory. *Journal of Algorithms 51(1):38–63*, 2004. doi:10.1016/j.jalgor.2003.10.002, link. Other versions: *SODA 2002, pp. 588–597*. First appeared in 2002. Cited on pp. 57, 78, 79, 83, 84, 85, 90, 94, 95, 98, 99, 641, 643, and 693.

Yann Disser, Jan Hackfeld, Max Klimm. Tight bounds for undirected graph exploration with pebbles and multiple agents. *Journal of the ACM 66(6)*, 2019. doi:10.1145/3356883, arXiv:1805.03476 (pdf). Other versions: *SODA 2016, pp. 25–39*, as "Undirected Graph Exploration with ⊠(log log n) Pebbles" (doi:10.1137/1.9781611974331.ch3). First appeared in 2016. Read: arXiv v1 (2018). Cited on pp. 50, 56, 57, 59, 61, 78, 83, 85, 88, 89, 90, 95, 96, 97, 161, 207, 209, 210, 257, 258, 643, 691, and 700.

Yann Disser, Sören Schmitt. Evacuating Two Robots from a Disk: A Second Cut. *SIROCCO 2019, pp. 200–214*, 2019. doi:10.1007/978-3-030-24922-9_14, arXiv:1905.10592 (pdf). Read: arXiv v1 (2019). Cited on pp. 63, 73, 79, 507, 510, 519, 520, 642, and 703.

Yann Disser, Frank Mousset, Andreas Noever, Nemanja Škorić, Angelika Steger. A general lower bound for collaborative tree exploration. *Theoretical Computer Science 811:70–78*, 2020. doi:10.1016/j.tcs.2018.03.006, arXiv:1610.01753 (pdf). Other versions: *SIROCCO 2017* (doi:10.1007/978-3-319-72050-0_8). First appeared in 2016. Read: arXiv v1 (2016). Cited on pp. 59, 60, 61, 77, 78, 79, 244, 246, 247, 248, 252, 253, 254, 257, 261, 277, 278, 279, 284, 285, 641, 645, 647, and 700.

Yann Disser, Max Klimm. The Space Complexity of Undirected Graph Exploration. *Lecture Notes in Computer Science , pp. 152–166*, 2022. doi:10.1007/978-3-031-21534-6_8. Cited on p. 86.

Ha Thi Thu Doan, François Bonnet, Kazuhiro Ogata. Model checking of a mobile robots perpetual exploration algorithm. *SOFL+MSVL 2016, LNCS 10189, pp. 201–219*, 2016. doi:10.1007/978-3-319-57708-1_12. Cited on pp. 166 and 173.

Ha Thi Thu Doan, François Bonnet, Kazuhiro Ogata. Model Checking of Robot Gathering. *OPODIS 2017, LIPIcs 95, 12:1–12:16*, 2017. doi:10.4230/LIPIcs.OPODIS.2017.12. Cited on pp. 142 and 150.

Stefan Dobrev, Paola Flocchini, Giuseppe Prencipe, Nicola Santoro. Searching for a black hole in arbitrary networks: optimal mobile agents protocols. *Distributed Computing 19(1)*, 2006. doi:10.1007/s00446-006-0154-y. Other versions: *PODC 2002*, as "Searching for a black hole in arbitrary networks: optimal mobile agent protocols" (doi:10.1145/571825.571853). First appeared in 2002. Cited on pp. 11, 13, 14, 16, 74, 78, 159, 207, 288, 289, 290, 291, 293, 296, 297, 298, 692, and 693.

Stefan Dobrev, Paola Flocchini, Giuseppe Prencipe, Nicola Santoro. Mobile Search for a Black Hole in an Anonymous Ring. *Algorithmica 48(1):67–90*, 2007. doi:10.1007/s00453-006-1232-z. Other versions: *DISC 2001* (doi:10.1007/3-540-45414-4_12). First appeared in 2001. Cited on pp. 13, 15, 288, 289, 290, 292, 293, 296, 297, and 692.

Stefan Dobrev, Paola Flocchini, Rastislav Královič, Nicola Santoro. Exploring an unknown dangerous graph using tokens. *Theoretical Computer Science 472:28–45*, 2013. doi:10.1016/j.tcs.2012.11.022. Other versions: *IFIP TCS 2006, pp. 131–150*, as "Exploring an Unknown Graph to Locate a Black Hole Using Tokens" (doi:10.1007/978-0-387-34735-6_14). First appeared in 2006. Cited on pp. 15, 207, 291, 293, 299, and 695.

Stefan Dobrev, Rastislav Královič, Dana Pardubská. Treasure Hunt with Barely Communicating Agents. *OPODIS 2017, LIPIcs 95, 14:1–14:16*, 2017. doi:10.4230/LIPIcs.OPODIS.2017.14, pdf. Cited on p. 110.

Stefan Dobrev, Rastislav Královič, Dana Pardubská. Exploration of Time-Varying Connected Graphs with Silent Agents. *SIROCCO 2020, LNCS, pp. 146–162*, 2020. doi:10.1007/978-3-030-54921-3_9. Cited on pp. 308, 309, 656, and 704.

Stefan Dobrev, Rastislav Královič, Dana Pardubská. Improved Lower Bounds for Shoreline Search. *Structural Information and Communication Complexity, pp. 80-90*, 2020. doi:10.1007/978-3-030-54921-3_5. Cited on pp. 508, 575, 576, 578, 579, and 580.

Stefan Dobrev, Paola Flocchini, Giuseppe Prencipe, Nicola Santoro. Asynchronous Gathering in a Dangerous Ring. *Algorithms 16(5):222*, 2023. doi:10.3390/a16050222, pdf. Other versions: *OPODIS 2003, pp. 34–46*, as "Multiple agents rendezvous in a ring in spite of a black hole". First appeared in 2003. Cited on p. 292.

Stefan Dobrev, Lata Narayanan, Jaroslav Opatrny, Denis Pankratov. Exploration of High-Dimensional Grids by Finite State Machines. *Algorithmica 86(5):1700–1729*, 2024. doi:10.1007/s00453-024-01207-6, arXiv:1902.03693 (pdf). Other versions: *ICALP 2019, LIPIcs 132, 139:1–139:16*, as "Exploration of High-Dimensional Grids by Finite Automata" (doi:10.4230/LIPIcs.ICALP.2019.139). First appeared in 2019. Cited on p. 86.

Konstantinos Dogeas, Thomas Erlebach, Frank Kammer, Johannes Meintrup, William K. Moses Jr. Exploiting Automorphisms of Temporal Graphs for Fast Exploration and Rendezvous. *ICALP 2024, LIPIcs 297, 55:1–55:18*, 2024. doi:10.4230/LIPIcs.ICALP.2024.55, arXiv:2312.07140 (pdf). First appeared in 2023. Cited on p. 120.

Keisuke Doi, Yukiko Yamauchi, Shuji Kijima, Masafumi Yamashita. Search by a metamorphic robotic system in a finite 2D square Grid. *Information and Computation 285:104695*, 2022. doi:10.1016/j.ic.2021.104695, arXiv:1808.00214 (pdf). Other versions: *SSS 2018, LNCS, pp. 96–110*, as "Exploration of Finite 2D Square Grid by a Metamorphic Robotic System" (doi:10.1007/978-3-030-03232-6_7). First appeared in 2018. Read: arXiv v2 (2019), which carries the journal title. Cited on p. 86.

Shlomi Dolev, Sayaka Kamei, Yoshiaki Katayama, Fukuhito Ooshita, Koichi Wada. Brief Announcement: Neighborhood Mutual Remainder and Its Self-Stabilizing Implementation of Look-Compute-Move Robots. *DISC 2019 (brief announcement), LIPIcs 146, 43:1–43:3*, 2019. doi:10.4230/LIPIcs.DISC.2019.43, arXiv:1903.02843 (pdf). Read: arXiv v1 (2019), the full version. Cited on p. 414.

Claude (Anthropic) was used in cataloguing the problems.

G. Dudek, M. Jenkin, E. Milios, D. Wilkes. Robotic exploration as graph construction. *IEEE Transactions on Robotics and Automation 7(6):859–865*, 1991. doi:10.1109/70.105395. Cited on pp. 155, 156, 157, 158, 159, 161, 162, and 691.

Fabien Dufoulon, Shay Kutten, William K. Moses Jr. Efficient Deterministic Leader Election for Programmable Matter. *PODC 2021, pp. 103–113*, 2021. doi:10.1145/3465084.3467900, arXiv:2106.01108 (pdf). Read: arXiv v1 (2021). Cited on pp. 43, 45, 47, 51, 70, 71, 79, 482, 585, 586, 589, 590, 594, 642, 658, 690, and 705.

Adrian Dumitrescu, Ichiro Suzuki, Masafumi Yamashita. Motion Planning for Metamorphic Systems: Feasibility, Decidability, and Distributed Reconfiguration. *IEEE Transactions on Robotics and Automation 20(3):409–418*, 2004. doi:10.1109/TRA.2004.824936. Cited on p. 637.

Adrian Dumitrescu, Anirban Ghosh, Csaba D. Tóth. On Fence Patrolling by Mobile Agents. *The Electronic Journal of Combinatorics 21(3):P3.4*, 2014. doi:10.37236/4063, arXiv:1401.6070 (pdf). Cited on pp. 569, 570, 571, 572, 573, 659, 683, and 699.

Mirosław Dynia, Jakub Łopuszański, Christian Schindelhauer. Why robots need maps. *SIROCCO 2007, pp. 41–50*, 2007. doi:10.1007/978-3-540-72951-8_5. Cited on pp. 246, 252, and 253.

Xavier Défago, Akihiko Konagaya. Circle formation for oblivious anonymous mobile robots with no common sense of orientation. *POMC 2002, pp. 97–104*, 2002. doi:10.1145/584490.584509. Cited on pp. 393, 395, 396, 397, 692, 693, and 694.

Xavier Défago, Samia Souissi. Non-uniform circle formation algorithm for oblivious mobile robots with convergence toward uniformity. *Theoretical Computer Science 396(1–3):97–112*, 2008. doi:10.1016/j.tcs.2008.01.050. Cited on pp. 23, 394, 395, 396, 397, 674, and 696.

Xavier Défago, Maria Potop-Butucaru, Philippe Raipin-Parvédy. Self-stabilizing gathering of mobile robots under crash or Byzantine faults. *Distributed Computing 33(5):393–421*, 2020. doi:10.1007/s00446-019-00359-x. Other versions: *DISC 2006, LNCS, pp. 46–60*, as "Fault-Tolerant and Self-stabilizing Mobile Robots Gathering" (doi:10.1007/11864219_4); *Research report PI 1802, HAL inria-00069600*, 2006, as "Fault-tolerant and Self-stabilizing Mobile Robots Gathering - Feasibility Study" (pdf, link); *Preprint, HAL hal-01275265 (the 2014 revision of the journal submission)*, as "Fault and Byzantine Tolerant Self-stabilizing Mobile Robots Gathering — Feasibility Study —" (pdf, link). First appeared in 2006. Cited on pp. 21, 25, 26, and 443.

Xavier Défago, Adam Heriban, Sébastien Tixeuil, Koichi Wada. Using model checking to formally verify rendezvous algorithms for robots with lights in Euclidean space. *Robotics and Autonomous Systems 163:104378*, 2023. doi:10.1016/j.robot.2023.104378, pdf, arXiv:1907.09871 (pdf). Other versions: *2020 International Symposium on Reliable Distributed Systems (SRDS) 2020, pp. 113–122* (doi:10.1109/srds51746.2020.00019). First appeared in 2019. Read: arXiv v1 (2019). Cited on pp. 382 and 446.

Gianlorenzo D'Angelo, Alfredo Navarra, Nicolas Nisse. A unified approach for gathering and exclusive searching on rings under weak assumptions. *Distributed Computing 30(1):17–48*, 2016. doi:10.1007/s00446-016-0274-y. Cited on p. 142.

Asaf Efrima, David Peleg. Distributed Models and Algorithms for Mobile Robot Systems. *SOFSEM 2007, LNCS 4362, pp. 70–87*, 2007. doi:10.1007/978-3-540-69507-3_5. Cited on p. 21.

Asaf Efrima, David Peleg. Distributed algorithms for partitioning a swarm of autonomous mobile robots. *Theoretical Computer Science 410(14):1355–1368*, 2009. doi:10.1016/j.tcs.2008.04.042. Other versions: *SIROCCO 2007* (doi:10.1007/978-3-540-72951-8_15). First appeared in 2007. Cited on pp. 18, 22, 23, 25, 26, 382, 384, 424, 425, 426, 428, 429, 466, 469, 672, and 695.

Ryota Eguchi, Naoki Kitamura, Taisuke Izumi. Fast Neighborhood Rendezvous. *IEICE Transactions on Information and Systems E105.D(3):597–610*, 2022. doi:10.1587/transinf.2021edp7104, arXiv:2105.03638 (pdf). Other versions: *ICDCS 2020, IEEE, pp. 168–178* (doi:10.1109/icdcs47774.2020.00030). First appeared in 2020. Cited on p. 120.

Anne-Laure Ehresmann, Manuel Lafond, Lata Narayanan, Jaroslav Opatrny. Distributed Pattern Formation in a Ring. *SIROCCO 2019, LNCS, pp. 215–229*, 2019. doi:10.1007/978-3-030-24922-9_15, arXiv:1905.08856 (pdf). Cited on p. 383.

Samir Elouasbi, Andrzej Pelc. Deterministic Meeting of Sniffing Agents in the Plane. *Fundamenta Informaticae 160(3):281–301*, 2018. doi:10.3233/fi-2018-1684, arXiv:1709.06217 (pdf). Other versions: *SIROCCO 2016, LNCS, pp. 212–227* (doi:10.1007/978-3-319-48314-6_14). First appeared in 2016. Cited on p. 120.

Yuval Emek, Shay Kutten, Ron Lavi, William K. Moses Jr. Deterministic Leader Election in Programmable Matter. *ICALP 2019 (LIPIcs 132), 140*, 2019. doi:10.4230/LIPIcs.ICALP.2019.140, pdf, arXiv:1905.00580 (pdf). Read: arXiv v1 (2019). Cited on pp. 42, 43, 44, 46, 47, 54, 69, 70, 586, 590, 593, 594, 596, 642, and 703.

Thomas Erlebach, Michael Hoffmann, Frank Kammer. On temporal graph exploration. *Journal of Computer and System Sciences 119:1–18*, 2021. doi:10.1016/j.jcss.2021.01.005, arXiv:1504.07976 (pdf). Other versions: *ICALP 2015* (doi:10.1007/978-3-662-47672-7_36). First appeared in 2015. Read: arXiv v3 (2021). Cited on pp. 6, 10, 11, 78, 89, 104, 105, 644, and 699.

Sándor Fekete, Chris Gray, Alexander Kröller. Evacuation of Rectilinear Polygons. *Combinatorial Optimization and Applications, pp. 21-30*, 2010. doi:10.1007/978-3-642-17458-2_3, arXiv:1008.4420 (pdf). Cited on p. 508.

Sándor P. Fekete, Eike Niehs, Christian Scheffer, Arne Schmidt. Connected Assembly and Reconfiguration by Finite Automata. *arXiv preprint*, 2019. arXiv:1909.03880 (pdf). Preprint, not peer reviewed. Cited on p. 632.

Michael Feldmann, Andreas Padalkin, Christian Scheideler, Shlomi Dolev. Coordinating Amoebots via Reconfigurable Circuits. *Journal of Computational Biology 29(4):317–343*, 2022. doi:10.1089/cmb.2021.0363, arXiv:2105.05071 (pdf). Other versions: *SSS 2021, LNCS, pp. 484–488* (doi:10.1007/978-3-030-91081-5_34); *arXiv preprint*, 2021, as "Accelerating Amoebots via Reconfigurable Circuits". First appeared in 2021. Read: arXiv v1 (2021), titled "Accelerating Amoebots via Reconfigurable Circuits"; theorem numbers refer to it. Cited on pp. 42, 43, 45, 46, 595, 600, 672, 687, and 705.

Caterina Feletti, Carlo Mereghetti, Beatrice Palano. Uniform circle formation for swarms of opaque robots with lights. *SSS 2018, pp. 317–332*, 2018. doi:10.1007/978-3-030-03232-6_21. Cited on pp. 403 and 703.

Caterina Feletti, Carlo Mereghetti, Beatrice Palano. $O(\log n)$-time uniform circle formation for asynchronous opaque luminous robots. *OPODIS 2023 (LIPIcs 286), 5:1–5:21*, 2023. doi:10.4230/LIPIcs.OPODIS.2023.5, pdf. Cited on pp. 65, 67, and 402.

Caterina Feletti, Carlo Mereghetti, Beatrice Palano. Uniform circle formation for fully, semi-, and asynchronous opaque robots with lights. *Applied Sciences 13(13):7991*, 2023. doi:10.3390/app13137991. First appeared in 2022. Cited on pp. 67, 399, 402, 403, and 706.

Caterina Feletti, Lucia Mambretti, Carlo Mereghetti, Beatrice Palano. Computational power of autonomous robots: Transparency vs. opaqueness. *Theoretical Computer Science 1036:115153*, 2025. doi:10.1016/j.tcs.2025.115153, arXiv:2401.16893 (pdf). Other versions: *SAND 2024, LIPIcs 292, 13:1–13:19*, as "Computational Power of Opaque Robots" (doi:10.4230/LIPIcs.SAND.2024.13, pdf). First appeared in 2024. Cited on pp. 25, 36, and 37.

Caterina Feletti, Debasish Pattanayak, Gokarna Sharma. Optimal luminous UCF under asynchrony. *Journal of Parallel and Distributed Computing 218:105338*, 2026. doi:10.1016/j.jpdc.2026.105338, arXiv:2405.06617 (pdf). Other versions: *DISC 2024 (LIPIcs 319), 46:1–46:7*, as "Brief announcement: optimal uniform circle formation by asynchronous luminous robots" (doi:10.4230/LIPIcs.DISC.2024.46, pdf). First appeared in 2024. Read: arXiv v1 (2024). Cited on pp. 67, 383, 395, 398, 399, 401, 402, 652, 653, and 708.

Caterina Feletti, Paola Flocchini, Nicola Santoro. Computational Power of Energy-Constrained Autonomous Robots Under Sequential Schedulers. *MFCS 2026, LIPIcs 386, 22:1–22:17*, 2026. doi:10.4230/LIPIcs.MFCS.2026.22, pdf, link. Cited on pp. 27, 30, 31, 35, 36, and 37.

Caterina Feletti, Paola Flocchini, Debasish Pattanayak, Giuseppe Prencipe, Nicola Santoro. Universal Dancing by Luminous Robots Under Sequential Schedulers. *SIROCCO 2026, pp. 292–312*, 2026. doi:10.1007/978-3-032-26465-7_16, arXiv:2508.15484 (pdf). Other versions: *DISC 2025 (brief announcement), LIPIcs 356, 56:1–56:7*, as "Brief Announcement: Universal Dancing by Luminous Robots Under Sequential Schedulers" (doi:10.4230/LIPIcs.DISC.2025.56, pdf). First appeared in 2025. Read: arXiv v1 (2025). Cited on pp. 67, 406, 411, and 710.

Caterina Feletti, Paola Flocchini, Giuseppe Prencipe, Nicola Santoro. Safe Exploration of Arbitrary Dynamic Dangerous Networks. *arXiv preprint*, 2026. arXiv:2609.19845 (pdf). Preprint, not peer reviewed. Cited on pp. 291 and 307.

Rudolf Fleischer, Gerhard Trippen. Exploring an unknown graph efficiently. *ESA 2005, pp. 11–22*, 2005. doi:10.1007/11561071_4. Cited on pp. 102, 103, and 694.

Paola Flocchini, Bernard Mans, Nicola Santoro. Sense of direction: Definitions, properties, and classes. *Networks 32(3):165–180*, 1998. doi:10.1002/(SICI)1097-0037(199810)32:3<165::AID-NET1>3.0.CO;2-I. Other versions: *SIROCCO 1995*, as "Sense of Direction: Formal Definitions and Properties" (doi:10.1515/9780773591158-002). First appeared in 1995. Cited on p. 293.

Paola Flocchini, Giuseppe Prencipe, Nicola Santoro, Peter Widmayer. Hard Tasks for Weak Robots: The Role of Common Knowledge in Pattern Formation by Autonomous Mobile Robots. *ISAAC 1999, LNCS 1741, pp. 93–102*, 1999. doi:10.1007/3-540-46632-0_10. Cited on pp. 20, 21, 22, and 382.

Paola Flocchini, Giuseppe Prencipe, Nicola Santoro, Peter Widmayer. Gathering of asynchronous robots with

limited visibility. *Theoretical Computer Science 337(1–3):147–168*, 2005. doi:10.1016/j.tcs.2005.01.001. Other versions: *STACS 2001*, as "Gathering of Asynchronous Oblivious Robots with Limited Visibility" (doi:10.1007/3-540-44693-1_22). First appeared in 2001. Cited on pp. 18, 24, 431, 432, 455, 456, 462, and 692.

Paola Flocchini, Giuseppe Prencipe, Nicola Santoro, Peter Widmayer. Arbitrary pattern formation by asynchronous, anonymous, oblivious robots. *Theoretical Computer Science 407(1–3):412–447*, 2008. doi:10.1016/j.tcs.2008.07.026. Cited on pp. 18, 22, 65, 77, 381, 382, 386, 388, 389, and 695.

Paola Flocchini, Giuseppe Prencipe, Nicola Santoro. Self-deployment of mobile sensors on a ring. *Theoretical Computer Science 402(1):67–80*, 2008. doi:10.1016/j.tcs.2008.03.006. Other versions: *ALGOSENSORS 2006*, as "Self-deployment Algorithms for Mobile Sensors on a Ring" (doi:10.1007/11963271_6). First appeared in 2006. Cited on pp. 313, 489, 490, 491, 492, 663, 694, and 695.

Paola Flocchini, Miao Jun Huang, Flaminia L. Luccio. Decontamination of hypercubes by mobile agents. *Networks 52(3):167–178*, 2008. doi:10.1002/net.20240. Other versions: *IPDPS 2005*. First appeared in 2005. Cited on pp. 15, 318, 319, 321, and 694.

Paola Flocchini, David Ilcinkas, Andrzej Pelc, Nicola Santoro. Remembering without memory: tree exploration by asynchronous oblivious robots. *Theoretical Computer Science 411(14–15):1583–1598*, 2010. doi:10.1016/j.tcs.2010.01.007, pdf, link. Other versions: *SIROCCO 2008* (doi:10.1007/978-3-540-69355-0_5, pdf). First appeared in 2008. Cited on p. 165.

Paola Flocchini, David Ilcinkas, Andrzej Pelc, Nicola Santoro. How many oblivious robots can explore a line. *Information Processing Letters 111(20):1027–1031*, 2011. doi:10.1016/j.ipl.2011.07.018, pdf. Cited on p. 165.

Paola Flocchini, Giuseppe Prencipe, Nicola Santoro. Distributed Computing by Oblivious Mobile Robots. *Synthesis Lectures on Distributed Computing Theory, Morgan & Claypool*, 2012. doi:10.2200/S00440ED1V01Y201208DCT010. Cited on p. 17.

Paola Flocchini, David Ilcinkas, Nicola Santoro. Ping Pong in Dangerous Graphs: Optimal Black Hole Search with Pebbles. *Algorithmica 62(3–4):1006–1033*, 2012. doi:10.1007/s00453-011-9496-3, pdf. Other versions: *DISC 2008*, as "Ping Pong in Dangerous Graphs: Optimal Black Hole Search with Pure Tokens" (doi:10.1007/978-3-540-87779-0_16, pdf). First appeared in 2008. Cited on pp. 15, 74, 159, 291, 293, 298, and 696.

Paola Flocchini, David Ilcinkas, Andrzej Pelc, Nicola Santoro. Computing Without Communicating: Ring Exploration by Asynchronous Oblivious Robots. *Algorithmica 65(3):562–583*, 2013. doi:10.1007/s00453-011-9611-5, pdf, link. Other versions: *OPODIS 2007* (doi:10.1007/978-3-540-77096-1_8, pdf). First appeared in 2007. Cited on pp. 163, 164, 165, 166, 167, 168, 169, 170, 171, 172, 173, 176, 178, 180, 181, 660, and 695.

Paola Flocchini, Nicola Santoro, Giovanni Viglietta, Masafumi Yamashita. Rendezvous with constant memory. *Theoretical Computer Science 621:57–72*, 2016. doi:10.1016/j.tcs.2016.01.025, arXiv:1306.1956 (pdf). Other versions: *SIROCCO 2013*, as "Rendezvous of Two Robots with Constant Memory" (doi:10.1007/978-3-319-03578-9_16). First appeared in 2013. Read: arXiv v1 (2013). Cited on pp. 18, 21, 23, 27, 29, 34, 36, 37, 53, 63, 66, 77, 177, 179, 432, 434, 439, 440, 441, 444, 445, 447, 448, 450, 642, 651, 652, 675, 676, and 698.

Paola Flocchini, Fabrizio Luccio, Linda Pagli, Nicola Santoro. Network decontamination under *m*-immunity. *Discrete Applied Mathematics 201:114–129*, 2016. doi:10.1016/j.dam.2015.07.020. Cited on p. 319.

Paola Flocchini, Nicola Santoro, Giovanni Viglietta, Masafumi Yamashita. Universal Systems of Oblivious Mobile Robots. *SIROCCO 2016, LNCS, pp. 242–257*, 2016. doi:10.1007/978-3-319-48314-6_16, arXiv:1602.04881 (pdf). Read: arXiv v4. Cited on p. 67.

Paola Flocchini, Giuseppe Prencipe, Nicola Santoro, Giovanni Viglietta. Distributed computing by mobile robots: uniform circle formation. *Distributed Computing 30(6):413–457*, 2017. doi:10.1007/s00446-016-0291-x, arXiv:1407.5917 (pdf). Other versions: *OPODIS 2014*, as "Distributed Computing by Mobile Robots: Solving the Uniform Circle Formation Problem" (doi:10.1007/978-3-319-14472-6_15). First appeared in 2014. Cited on pp. 65, 66, 69, 381, 386, 393, 394, 396, 397, 398, 399, 401, 427, 434, 449, and 699.

Paola Flocchini, Ryan Killick, Evangelos Kranakis, Nicola Santoro, Masafumi Yamashita. Gathering and Election by Mobile Robots in a Continuous Cycle. *ISAAC 2019, LIPIcs 149, 8:1–8:19*, 2019. doi:10.4230/LIPIcs.ISAAC.2019.8, pdf. Cited on p. 433.

Paola Flocchini, Nicola Santoro, Koichi Wada. On Memory, Communication, and Synchronous Schedulers When Moving and Computing. *OPODIS 2019, LIPIcs 153, 25:1–25:17*, 2019. doi:10.4230/LIPIcs.OPODIS.2019.25, pdf. Cited on pp. 18, 21, 27, 28, 30, 34, 37, 51, 65, 66, and 67.

Paola Flocchini, Nicola Santoro, Masafumi Yamashita, Yukiko Yamauchi. Synchronization by Asynchronous

Mobile Robots with Limited Visibility. *arXiv preprint*, 2020. arXiv:2006.03249 (pdf). Read: arXiv v2. Preprint, not peer reviewed. Cited on p. 67.

Paola Flocchini, Nicola Santoro, Yuichi Sudo, Koichi Wada. On Asynchrony, Memory, and Communication: Separations and Landscapes. *OPODIS 2023, LIPIcs 286, 28:1–28:23*, 2023. doi:10.4230/LIPIcs.OPODIS.2023.28, pdf, arXiv:2311.03328 (pdf). Read: arXiv v2 (2023). Cited on pp. 17, 18, 20, 21, 23, 24, 27, 28, 29, 33, 34, 35, 37, 65, 66, 434, 642, 671, and 674.

Paola Flocchini, Debasish Pattanayak, Francesco Piselli, Nicola Santoro, Yukiko Yamauchi. Asynchronous Separation of Unconscious Colored Robots. *International Journal of Networking and Computing 15(2):199–219*, 2025. doi:10.15803/ijnc.15.2_199. Other versions: *CANDAR Workshops 2024, pp. 183–189* (doi:10.1109/CANDARW64572.2024.00037). First appeared in 2024. Read: the CANDAR Workshops 2024 version. Cited on pp. 484, 485, 486, and 708.

Paola Flocchini, Alfredo Navarra, Debasish Pattanayak, Francesco Piselli, Nicola Santoro. Universal pattern formation by oblivious robots under sequential schedulers. *Distributed Computing (2026)*, 2026. doi:10.1007/s00446-026-00504-3, arXiv:2412.10733 (pdf). Other versions: *SIROCCO 2025*, as "Oblivious Robots Under Sequential Schedulers: Universal Pattern Formation" (doi:10.1007/978-3-031-91736-3_18). First appeared in 2024. Read: arXiv v3 (2026). Cited on pp. 18, 20, 24, 27, 30, 36, 55, 66, 383, 404, 405, 406, 407, 427, 431, 432, 434, 449, 708, and 709.

Klaus-Tycho Foerster, Roger Wattenhofer. Lower and upper competitive bounds for online directed graph exploration. *Theoretical Computer Science 655:15–29*, 2016. doi:10.1016/j.tcs.2015.11.017, link. Other versions: *OPODIS 2012*, as "Directed Graph Exploration" (link). First appeared in 2012. Cited on p. 103.

Pierre Fraigniaud, David Ilcinkas. Digraphs exploration with little memory. *STACS 2004, pp. 246–257*, 2004. doi:10.1007/978-3-540-24749-4_22, pdf. Cited on pp. 6, 11, 15, 58, 101, 644, and 694.

Pierre Fraigniaud, David Ilcinkas, Guy Peer, Andrzej Pelc, David Peleg. Graph exploration by a finite automaton. *Theoretical Computer Science 345(2–3):331–344*, 2005. pdf, link. Other versions: *MFCS 2004* (doi:10.1007/978-3-540-28629-5_34, pdf). First appeared in 2004. Cited on pp. 15, 56, 78, 84, 85, 90, 93, 97, 207, 209, 210, 211, 221, 643, and 693.

Pierre Fraigniaud, David Ilcinkas, Sergio Rajsbaum, Sébastien Tixeuil. Space Lower Bounds for Graph Exploration via Reduced Automata. *SIROCCO 2005, LNCS 3499, pp. 140–154*, 2005. doi:10.1007/11429647_13, pdf. Other versions: Fraigniaud et al. (2006c). Cited on pp. 57, 98, 643, and 694.

Pierre Fraigniaud, Leszek Gąsieniec, Dariusz R. Kowalski, Andrzej Pelc. Collective tree exploration. *Networks 48(3):166–177*, 2006. doi:10.1002/net.20127. Other versions: *LATIN 2004* (doi:10.1007/978-3-540-24698-5_18). First appeared in 2004. Cited on pp. 11, 13, 15, 16, 59, 244, 245, 246, 248, 252, 255, 256, 693, and 694.

Pierre Fraigniaud, David Ilcinkas, Andrzej Pelc. Oracle size: a new measure of difficulty for communication tasks. *PODC 2006, pp. 179–187*, 2006. doi:10.1145/1146381.1146410, pdf. Cited on p. 89.

Pierre Fraigniaud, David Ilcinkas, Sergio Rajsbaum, Sébastien Tixeuil. The reduced automata technique for graph exploration space lower bounds. *Theoretical Computer Science: Essays in Memory of Shimon Even, LNCS, pp. 1–26*, 2006. doi:10.1007/11685654_1, pdf, link. Other versions: Fraigniaud et al. (2005b). First appeared in 2005. Cited on pp. 50, 56, 57, 78, 85, 89, 90, 94, 98, and 694.

Pierre Fraigniaud, David Ilcinkas, Andrzej Pelc. Tree exploration with advice. *Information and Computation 206(11):1276–1287*, 2008. doi:10.1016/j.ic.2008.07.005, pdf. Other versions: *MFCS 2006*, as "Tree Exploration with an Oracle" (doi:10.1007/11821069_2). First appeared in 2006. Cited on p. 111.

Pierre Fraigniaud, Andrzej Pelc. Delays induce an exponential memory gap for rendezvous in trees. *ACM Transactions on Algorithms 9(2), article 17*, 2013. doi:10.1145/2438645.2438649, arXiv:1102.0467 (pdf). Other versions: *DISC 2008*; *SPAA 2010, pp. 224–232* (doi:10.1145/1810479.1810524). First appeared in 2008. Read: arXiv v1 (2011). Cited on pp. 118, 119, 121, 127, and 695.

Pierre Fraigniaud, Andrzej Pelc. Decidability classes for mobile agents computing. *Journal of Parallel and Distributed Computing 109:117–128*, 2017. doi:10.1016/j.jpdc.2017.04.003, arXiv:1011.2719 (pdf). Other versions: *LATIN 2012, LNCS 7256, pp. 362–374* (doi:10.1007/978-3-642-29344-3_31). First appeared in 2010. Read: arXiv v1 (2010). Cited on pp. 134, 158, 366, 367, 368, 369, 370, 371, 372, 373, 374, 375, 666, 667, 677, and 696.

Jonas Friemel, David Liedtke, Christian Scheffer. Tile Reconfiguration by a Finite Automaton. *WALCOM 2026, LNCS, pp. 512–526*, 2026. doi:10.1007/978-981-95-7127-7_34, arXiv:2501.08663 (pdf). First appeared in 2025. Read: arXiv v2 (2025). Cited on pp. 632 and 634.

Nao Fujinaga, Yukiko Yamauchi, Hirotaka Ono, Shuji Kijima, Masafumi Yamashita. Pattern formation by oblivious

asynchronous mobile robots. *SIAM Journal on Computing 44(3):740–785*, 2015. doi:10.1137/140958682. Other versions: *DISC 2012*, as "Asynchronous Pattern Formation by Anonymous Oblivious Mobile Robots" (doi:10.1007/978-3-642-33651-5_22). First appeared in 2012. Cited on pp. 382, 386, 387, 388, 389, and 651.

Younan Gao, Andrzej Pelc. Sniffing helps to meet: Deterministic rendezvous of anonymous agents in the grid. *Theoretical Computer Science 1050:115395*, 2025. doi:10.1016/j.tcs.2025.115395, arXiv:2407.14969 (pdf). First appeared in 2024. Cited on p. 120.

Younan Gao, Andrzej Pelc. Gathering teams of bounded memory agents on a line. *Distributed Computing 39(2):10*, 2026. doi:10.1007/s00446-025-00495-7, arXiv:2507.04172 (pdf). Other versions: *OPODIS 2024, LIPIcs 324, 11:1–11:17*, as "Gathering Teams of Deterministic Finite Automata on a Line" (doi:10.4230/LIPIcs.OPODIS.2024.11, pdf). First appeared in 2024. Read: arXiv v1 (July 2025). Cited on pp. 119 and 133.

Nicolas Gastineau, Wahabou Abdou, Nader Mbarek, Olivier Togni. Distributed Leader Election and Computation of Local Identifiers for Programmable Matter. *ALGOSENSORS 2018, LNCS*, 2018. doi:10.1007/978-3-030-14094-6_11, arXiv:1807.10461 (pdf). Read: arXiv v1 (2018). Cited on pp. 47, 70, 590, 680, and 703.

Nicolas Gastineau, Wahabou Abdou, Nader Mbarek, Olivier Togni. Leader election and local identifiers for three-dimensional programmable matter. *Concurrency and Computation: Practice and Experience 34(7):e6067*, 2020. doi:10.1002/cpe.6067, link, arXiv:2010.09286 (pdf). Read: arXiv v1 (2020). Cited on p. 586.

Gaurav Gaur, Barun Gorain, Rishi Ranjan Singh, Daya Gaur. Robust Treasure Hunt in Anonymous Graphs with Quantum Pebbles by Oblivious Agents. *arXiv preprint*, 2025. arXiv:2509.02909 (pdf). Preprint, not peer reviewed. Cited on p. 109.

Matthias Gehnen, Ralf Klasing, Émile Naquin. Graph Exploration with Edge Weight Estimates. *arXiv preprint*, 2025. arXiv:2501.18496 (pdf). Preprint, not peer reviewed. Cited on pp. 10, 225, 226, 228, 229, 230, 664, 690, and 710.

Konstantinos Georgiou, Evangelos Kranakis, Alexandra Steau. Searching with Advice: Robot Fence-Jumping. *Journal of Information Processing 25:559–571*, 2017. doi:10.2197/ipsjjip.25.559, pdf, arXiv:1606.08023 (pdf). First appeared in 2016. Read: arXiv v1 (2016). Cited on p. 509.

Konstantinos Georgiou, Jay Griffiths, Yuval Yakubov. Symmetric Rendezvous with Advice: How to Rendezvous in a Disk. *SIROCCO 2018, LNCS, pp. 121–133*, 2018. doi:10.1007/978-3-030-01325-7_14, arXiv:1805.03351 (pdf). Cited on p. 120.

Konstantinos Georgiou, George Karakostas, Evangelos Kranakis. Search-and-Fetch with 2 Robots on a Disk: Wireless and Face-to-Face Communication Models. *Discrete Mathematics & Theoretical Computer Science 21(3)*, 2019. doi:10.23638/DMTCS-21-3-20, arXiv:1611.10208 (pdf). Other versions: *ICORES 2017, pp. 15–26* (doi:10.5220/0006091600150026). First appeared in 2016. Read: arXiv v6 (2019). Cited on pp. 510, 518, 519, 653, and 701.

Konstantinos Georgiou, George Karakostas, Evangelos Kranakis. Treasure evacuation with one robot on a disk. *Theoretical Computer Science 852:18–28*, 2021. doi:10.1016/j.tcs.2020.11.008. Other versions: *ALGOSENSORS 2017, LNCS, pp. 80–94*, as "Search-and-Fetch with One Robot on a Disk: (Track: Wireless and Geometry)" (doi:10.1007/978-3-319-53058-1_6). First appeared in 2016. Cited on p. 519.

Konstantinos Georgiou, Nikos Giachoudis, Evangelos Kranakis. Evacuation from a Disk for Robots with Asymmetric Communication. *ISAAC 2022 (LIPIcs 248), 19:1–19:16*, 2022. doi:10.4230/LIPIcs.ISAAC.2022.19, pdf. Cited on pp. 39, 40, 41, 73, 507, 520, and 706.

Konstantinos Georgiou, Sean Leizerovich, Jesse Lucier, Somnath Kundu. Evacuating from ℓ_p unit disks in the wireless model. *Theoretical Computer Science 944:113675*, 2023. doi:10.1016/j.tcs.2022.12.025, arXiv:2108.02367 (pdf). Other versions: *ALGOSENSORS 2021, LNCS, pp. 76–93* (doi:10.1007/978-3-030-89240-1_6). First appeared in 2021. Read: arXiv v1 (2021). Cited on p. 517.

Konstantinos Georgiou, Evangelos Kranakis, Nikos Leonardos, Aris Pagourtzis, Ioannis Papaioannou. Optimal circle search despite the presence of faulty robots. *Information Processing Letters 182, 106391*, 2023. doi:10.1016/j.ipl.2023.106391. Other versions: *ALGOSENSORS 2019, LNCS, pp. 192–205* (doi:10.1007/978-3-030-34405-4_11). First appeared in 2019. Cited on pp. 511, 525, 526, 528, 529, 654, and 703.

Konstantinos Georgiou, Jesse Lucier. Weighted group search on a line & implications to the priority evacuation problem. *Theoretical Computer Science 939, 1-17*, 2023. doi:10.1016/j.tcs.2022.10.013. Other versions: *ALGOSENSORS 2020, LNCS, pp. 124–139*, as "Weighted group search on a line" (doi:10.1007/978-3-030-62401-9_9). First appeared in 2020. Cited on pp. 511, 529, 530, 531, 654, and 704.

Konstantinos Georgiou, Nikos Giachoudis, Evangelos Kranakis. Overcoming probabilistic faults in disoriented linear search. *Theoretical Computer Science 1014:114761*, 2024. doi:10.1016/j.tcs.2024.114761, arXiv:2303.15608

(pdf). Other versions: *SIROCCO 2023, LNCS, pp. 520–535* (doi:10.1007/978-3-031-32733-9_23). First appeared in 2023. Cited on p. 534.

Konstantinos Georgiou, Somnath Kundu, Paweł Prałat. The Fagnano Triangle Patrolling Problem. *Discrete Mathematics & Theoretical Computer Science 27(3), article 13421*, 2025. doi:10.46298/dmtcs.13421, pdf, arXiv:2307.13153 (pdf). Other versions: *SSS 2023, LNCS, pp. 157–171*, as "The Fagnano Triangle Patrolling Problem (extended abstract)". First appeared in 2023. Read: arXiv v6 (2025), the accepted journal version. Cited on p. 568.

Konstantinos Georgiou, Caleb Jones, Jesse Lucier. Multi-agent Search-Type Problems on Polygons. *SOFSEM 2025, LNCS, pp. 314–332*, 2025. doi:10.1007/978-3-031-82670-2_23, arXiv:2406.19495 (pdf). First appeared in 2024. Cited on p. 507.

Konstantinos Georgiou, Caleb Jones, Matthew Madej. Spirals and Beyond: Competitive Plane Search with Multi-Speed Agents. *arXiv preprint*, 2025. arXiv:2508.10793 (pdf). Preprint, not peer reviewed. Cited on p. 508.

Konstantinos Georgiou, Xin Wang. Weighted group search on the disk & improved lower bounds for priority evacuation. *Journal of Computer and System Sciences 154:103678*, 2025. doi:10.1016/j.jcss.2025.103678, arXiv:2406.19490 (pdf). Other versions: *IWOCA 2024, LNCS, pp. 28–42*, as "Weighted Group Search on the Disk and Improved LP-Based Lower Bounds for Priority Evacuation" (doi:10.1007/978-3-031-63021-7_3). First appeared in 2024. Read: arXiv v1 (2024). Cited on pp. 73, 520, 521, 522, 653, 654, and 708.

Konstantinos Georgiou. Optimal Average Disk-Inspection via Fermat's Principle. *LIPIcs, Volume 364, STACS 2026 44:1*, 2026. doi:10.4230/LIPIcs.STACS.2026.44, pdf, arXiv:2509.06334 (pdf). First appeared in 2025. Read: arXiv v3 (2026). Cited on p. 508.

Konstantinos Georgiou, Woojin Jang. Triangle evacuation of 2 agents in the wireless model & the power of choosing a starting point. *Journal of Computer and System Sciences 155:103695*, 2026. doi:10.1016/j.jcss.2025.103695, arXiv:2209.08544 (pdf). Other versions: *ALGOWIN 2022, LNCS, pp. 77–90*, as "Triangle evacuation of 2 agents in the wireless model" (doi:10.1007/978-3-031-22050-0_6). First appeared in 2022. Read: arXiv v1 (2022). Cited on pp. 73, 538, 655, and 706.

Raphael Gerlach, Sören von der Gracht, Christopher Hahn, Jonas Harbig, Peter Kling. Symmetry Preservation in Swarms of Oblivious Robots with Limited Visibility. *OPODIS 2024, LIPIcs 324, 13:1–13:28*, 2024. doi:10.4230/LIPIcs.OPODIS.2024.13, arXiv:2409.19277 (pdf). Read: arXiv v1 (2024). Cited on p. 462.

Vincenzo Gervasi, Giuseppe Prencipe. Coordination without communication: the case of the flocking problem. *Discrete Applied Mathematics 144(3):324–344*, 2004. doi:10.1016/j.dam.2003.11.010. Cited on pp. 487, 488, 489, 693, and 694.

Satakshi Ghosh, Pritam Goswami, Avisek Sharma, Buddhadeb Sau. Move optimal and time optimal arbitrary pattern formations by asynchronous robots on infinite grid. *International Journal of Parallel, Emergent and Distributed Systems 38(1):35–57*, 2023. doi:10.1080/17445760.2022.2124411, arXiv:2205.13870 (pdf). First appeared in 2022. Cited on p. 382.

Satakshi Ghosh, Avisek Sharma, Pritam Goswami, Buddhadeb Sau. Oblivious Robots Performing Different Tasks on Grid Without Knowing Their Team Members. *ICARA 2023, pp. 180–186*, 2023. doi:10.1109/icara56516.2023.10125816, arXiv:2210.00567 (pdf). First appeared in 2022. Cited on p. 383.

Satakshi Ghosh, Avisek Sharma, Pritam Goswami, Buddhadeb Sau. Asynchronous Gathering of Finite Memory Robots on a Circle under Limited Visibility. *International Journal of Foundations of Computer Science 37(7):941–965*, 2025. doi:10.1142/S0129054125500212, arXiv:2302.07600 (pdf). First appeared in 2023. Read: arXiv v1 (2023). Cited on p. 433.

Nikos Giachoudis, Maria Kokkou, Euripides Markou. Black Virus Decontamination of synchronous ring networks by initially scattered mobile agents. *Discrete Applied Mathematics 392:20–46*, 2026. doi:10.1016/j.dam.2026.05.031. Other versions: *SIROCCO 2020, LNCS, pp. 220–236* (doi:10.1007/978-3-030-54921-3_13). First appeared in 2020. Cited on p. 319.

Robert Gmyr, Kristian Hinnenthal, Irina Kostitsyna, Fabian Kuhn, Dorian Rudolph, Christian Scheideler. Shape Recognition by a Finite Automaton Robot. *MFCS 2018, LIPIcs 117, 52:1–52:15*, 2018. doi:10.4230/LIPIcs.MFCS.2018.52. Cited on p. 632.

Robert Gmyr, Kristian Hinnenthal, Irina Kostitsyna, Fabian Kuhn, Dorian Rudolph, Christian Scheideler, Thim Strothmann. Forming tile shapes with simple robots. *Natural Computing 19:375–390*, 2020. doi:10.1007/s11047-019-09774-2. Other versions: *DNA 2018, LNCS, pp. 122–138* (doi:10.1007/978-3-030-00030-1_8). First appeared in 2017. Cited on pp. 43, 630, 631, 632, 634, 701, and 702.

Barun Gorain, Andrzej Pelc. Deterministic Graph Exploration with Advice. *ACM Transactions on Algorithms 15(1):1–17*, 2018. doi:10.1145/3280823, arXiv:1607.01657 (pdf). First appeared in 2016. Cited on p. 86.

Barun Gorain, Kaushik Mondal, Himadri Nayak, Supantha Pandit. Pebble guided optimal treasure hunt in anonymous graphs. *Theoretical Computer Science 922:61–80*, 2022. doi:10.1016/j.tcs.2022.04.011, arXiv:2103.05933 (pdf). Other versions: *SIROCCO 2021*, as "Pebble Guided Near Optimal Treasure Hunt in Anonymous Graphs" (doi:10.1007/978-3-030-79527-6_13). First appeared in 2021. Read: arXiv v1 (2021). Cited on pp. 58, 78, 109, 111, 114, and 705.

Barun Gorain, Partha Sarathi Mandal, Kaushik Mondal, Supantha Pandit. Collaborative Dispersion by Silent Robots. *Journal of Parallel and Distributed Computing 188:104852*, 2024. doi:10.1016/j.jpdc.2024.104852, arXiv:2202.05710 (pdf). Other versions: *SSS 2022 (LNCS 13751)* (doi:10.1007/978-3-031-21017-4_17). First appeared in 2022. Read: arXiv v1 (2022). Cited on pp. 7, 13, 60, 78, 270, 276, and 706.

Pritam Goswami, Avisek Sharma, Satakshi Ghosh, Buddhadeb Sau. Brief Announcement: Rendezvous on a Known Dynamic Point in a Finite Unoriented Grid. *SSS 2023 (brief announcement), LNCS, pp. 374–379*, 2023. doi:10.1007/978-3-031-44274-2_27, arXiv:2301.08519 (pdf). Cited on p. 142.

Pritam Goswami, Manash Kumar Kundu, Satakshi Ghosh, Buddhadeb Sau. Circle formation by asynchronous opaque fat robots on an infinite grid. *International Journal of Parallel, Emergent and Distributed Systems 39(2):214–247*, 2024. doi:10.1080/17445760.2024.2316017, arXiv:2308.10031 (pdf). First appeared in 2023. Read: arXiv v1 (2023). Cited on p. 395.

Pritam Goswami, Adri Bhattacharya, Raja Das, Partha Sarathi Mandal. Perpetual Exploration of a Ring in Presence of Byzantine Black Hole. *OPODIS 2024, LIPIcs 324, 17:1–17:17*, 2024. doi:10.4230/LIPIcs.OPODIS.2024.17, pdf, arXiv:2407.05280 (pdf). Read: arXiv v2 (2024). Cited on p. 292.

Pritam Goswami, Avisek Sharma, Satakshi Ghosh, Buddhadeb Sau. Time optimal gathering of myopic robots on an infinite triangular grid. *Theoretical Computer Science 1023:114930*, 2025. doi:10.1016/j.tcs.2024.114930, arXiv:2204.14042 (pdf). First appeared in 2022. Read: arXiv v4 (2022). Cited on p. 142.

Tsuyoshi Gotoh, Yuichi Sudo, Fukuhito Ooshita, Toshimitsu Masuzawa. Dynamic Ring Exploration with (H,S) View. *Algorithms 13(6):141*, 2020. doi:10.3390/a13060141, pdf. Other versions: *SSS 2019, LNCS, pp. 165–177*, as "Exploration of Dynamic Ring Networks by a Single Agent with the H-hops and S-time Steps View". First appeared in 2019. Cited on p. 307.

Tsuyoshi Gotoh, Paola Flocchini, Toshimitsu Masuzawa, Nicola Santoro. Exploration of dynamic networks: Tight bounds on the number of agents. *Journal of Computer and System Sciences 122:1–18*, 2021. doi:10.1016/j.jcss.2021.04.003. Other versions: *OPODIS 2019, LIPIcs 153, 22:1–22:16*, as "Tight Bounds on Distributed Exploration of Temporal Graphs" (doi:10.4230/LIPIcs.OPODIS.2019.22). First appeared in 2019. Cited on pp. 306, 307, 310, and 703.

Samuel Guilbault, Andrzej Pelc. Asynchronous rendezvous of anonymous agents in arbitrary graphs. *OPODIS 2011, pp. 421–434*, 2011. doi:10.1007/978-3-642-25873-2_29. Cited on pp. 62, 123, and 697.

Siddharth Gupta, Guy Sa'ar, Meirav Zehavi. Collective Graph Exploration Parameterized by Vertex Cover. *IPEC 2023, LIPIcs 285, 22:1–22:18*, 2023. doi:10.4230/LIPIcs.IPEC.2023.22. Cited on p. 247.

Siddharth Gupta, Marc van Kreveld, Othon Michail, Andreas Padalkin. Collision Detection for Modular Robots – it is easy to cause collisions and hard to avoid them. *arXiv preprint*, 2023. arXiv:2305.01015 (pdf). Preprint, not peer reviewed. Cited on p. 637.

Arya Tanmay Gupta, Sandeep S. Kulkarni. Tolerance to Asynchrony of an Algorithm for Gathering Myopic Robots on an Infinite Triangular Grid. *EDCC 2024, IEEE, pp. 61–68*, 2024. doi:10.1109/EDCC61798.2024.00023, arXiv:2307.13080 (pdf). First appeared in 2023. Read: arXiv v2. Cited on p. 142.

Leszek Gąsieniec, Ralf Klasing, Russell Martin, Alfredo Navarra, Xiaohui Zhang. Fast periodic graph exploration with constant memory. *Journal of Computer and System Sciences 74(5)*, 2008. doi:10.1016/j.jcss.2007.09.004. Other versions: *SIROCCO 2007, LNCS, pp. 26–40* (doi:10.1007/978-3-540-72951-8_4). First appeared in 2007. Cited on p. 100.

Bernhard Haeupler, Fabian Kuhn, Anders Martinsson, Kalina Petrova, Pascal Pfister. Optimal Strategies for Patrolling Fences. *ICALP 2019, LIPIcs 132, 144:1–144:13*, 2019. doi:10.4230/LIPIcs.ICALP.2019.144, pdf, arXiv:1809.06727 (pdf). First appeared in 2018. Read: arXiv v2 (2019); theorem numbers refer to it. Cited on pp. 568, 571, 573, 659, and 703.

Christopher Hahn, Jonas Harbig, Peter Kling. Forming large patterns with local robots in the OBLOT model. *SAND 2024 (LIPIcs 292), 14:1–14:20*, 2024. doi:10.4230/LIPIcs.SAND.2024.14, pdf, arXiv:2404.02771 (pdf). Read: arXiv v2 (2024). Cited on pp. 391, 651, and 708.

Adam Heriban, Xavier Défago, Sébastien Tixeuil. Optimally Gathering Two Robots. *ICDCN 2018, pp. 1–10*, 2018. doi:10.1145/3154273.3154323, arXiv:1708.06183 (pdf). First appeared in 2017. Read: arXiv v1 (2017). Cited on pp. 18, 23, 63, 66, 431, 432, 434, 446, 447, 651, and 701.

Adam Heriban, Sébastien Tixeuil. Unreliable Sensors for Reliable Efficient Robots. *arXiv preprint*, 2021. arXiv:2105.09667 (pdf). Preprint, not peer reviewed. Cited on p. 433.

Attila Hideg, Tamás Lukovszki. Asynchronous filling by myopic luminous robots. *ALGOSENSORS 2020, LNCS, pp. 108–123*, 2020. doi:10.1007/978-3-030-62401-9_8, arXiv:1909.06895 (pdf). First appeared in 2019. Cited on p. 177.

Yuya Higashikawa, Naoki Katoh, Stefan Langerman, Shin-ichi Tanigawa. Online graph exploration algorithms for cycles and trees by multiple searchers. *Journal of Combinatorial Optimization 28(2):480–495*, 2014. doi:10.1007/s10878-012-9571-y. Cited on pp. 252 and 253.

Himani, Supantha Pandit. Efficient dispersion in triangular grids without prior knowledge. *Theoretical Computer Science 1070:115810*, 2026. doi:10.1016/j.tcs.2026.115810. Other versions: *ICDCIT 2025 (LNCS 15507)*, as "Optimal Dispersion in Triangular Grids: Achieving Efficiency Without Prior Knowledge" (doi:10.1007/978-3-031-81404-4_7). First appeared in 2025. Read: the ICDCIT 2025 version. Cited on p. 274.

Himani, Barun Gorain, Supantha Pandit. Efficient Dispersion of Mobile robots in Torus Networks. *ICDCN 2026, ACM, pp. 41–50*, 2026. doi:10.1145/3772290.3772306, pdf. Cited on p. 261.

Himani, Supantha Pandit, Gokarna Sharma. Location-Aware Dispersion on Anonymous Graphs. *arXiv preprint*, 2026. arXiv:2602.05948 (pdf). Preprint, not peer reviewed. Cited on p. 260.

Himani, Supantha Pandit. Optimal Maximal Independent Set Computation by Robots in Unoriented Grid and Torus. *ICDCN 2026, ACM, pp. 31–40*, 2026. doi:10.1145/3772290.3772305, pdf. Cited on p. 348.

Kristian Hinnenthal, David Liedtke, Christian Scheideler. Efficient Shape Formation by 3D Hybrid Programmable Matter: An Algorithm for Low Diameter Intermediate Structures. *Theoretical Computer Science 1057:115552*, 2025. doi:10.1016/j.tcs.2025.115552, arXiv:2401.17734 (pdf). Other versions: *SAND 2024, LIPIcs 292, 15* (doi:10.4230/LIPIcs.SAND.2024.15, pdf). First appeared in 2024. Read: arXiv v3 (2024). Cited on pp. 43, 632, 633, and 709.

Shoma Hiraoka, Shunsuke Imori, Shota Takahashi, Yuichi Sudo. Color Complexity of Recolorable Graph Exploration: Upper and Lower Bounds via Block Structure. *arXiv preprint 2609.14356*, 2026. link, arXiv:2609.14356 (pdf). Preprint, not peer reviewed. Cited on pp. 206, 207, 208, 209, 211, 212, 214, 215, 216, 662, 685, and 711.

Jion Hirose, Junya Nakamura, Fukuhito Ooshita, Michiko Inoue. Weakly Byzantine Gathering with a Strong Team. *IEICE Transactions on Information and Systems E105.D(3):541–555*, 2022. doi:10.1587/transinf.2021fcp0011, arXiv:2007.08217 (pdf). Other versions: *ICDCN 2021, ACM, pp. 26–35*, as "Gathering with a strong team in weakly Byzantine environments" (doi:10.1145/3427796.3427799). First appeared in 2020. Cited on p. 134.

Jion Hirose, Junya Nakamura, Fukuhito Ooshita, Michiko Inoue. Fast gathering despite a linear number of weakly Byzantine agents†. *Concurrency and Computation: Practice and Experience 36(14):e8055*, 2024. doi:10.1002/cpe.8055, arXiv:2205.14937 (pdf). Other versions: *CANDARW 2022, pp. 161–167*, as "Gathering despite a linear number of weakly Byzantine agents" (doi:10.1109/candarw57323.2022.00067); *PODC 2022 (brief announcement), ACM, pp. 375–377*, as "Brief Announcement: Gathering Despite a Linear Number of Weakly Byzantine Agents" (doi:10.1145/3519270.3538470). First appeared in 2022. Cited on p. 133.

Jion Hirose, Ryota Eguchi, Yuichi Sudo. Self-Stabilizing Weakly Byzantine Perpetual Gathering of Mobile Agents. *SAND 2025, LIPIcs 330, 13:1–13:18*, 2025. doi:10.4230/LIPIcs.SAND.2025.13, pdf, link, arXiv:2504.08271 (pdf). Read: arXiv v1 (April 2025). Cited on pp. 137, 138, 656, and 710.

Liang Hu, Meng Zhang, Yi Zhang, Jijun Tang. Label-Guided Graph Exploration with Adjustable Ratio of Labels. *International Journal of Foundations of Computer Science 23(04):903–929*, 2012. doi:10.1142/s0129054112500104, link, arXiv:1204.2955 (pdf). Other versions: *arXiv preprint 1204.2955*, 2012. Cited on pp. 208 and 221.

David Ilcinkas. Setting port numbers for fast graph exploration. *Theoretical Computer Science 401:236–242*, 2008. doi:10.1016/j.tcs.2008.03.035, pdf. Other versions: *SIROCCO 2006* (doi:10.1007/11780823_6, pdf). First appeared in 2006. Cited on pp. 57, 90, 100, 673, and 694.

David Ilcinkas, Nicolas Nisse, David Soguet. The cost of monotonicity in distributed graph searching. *Distributed Computing 22(2):117–127*, 2009. doi:10.1007/s00446-009-0089-1, pdf. Other versions: *OPODIS 2007* (doi:10.1007/978-3-540-77096-1_30, pdf). First appeared in 2007. Cited on pp. 318, 320, 321, and 695.

David Ilcinkas, Ahmed Mouhamadou Wade. Exploration of the T-interval-connected dynamic graphs: the case of the ring. *Theory of Computing Systems 62(5):1144–1160*, 2018. doi:10.1007/s00224-017-9796-3, pdf. Other versions: *SIROCCO 2013* (doi:10.1007/978-3-319-03578-9_2). First appeared in 2013. Cited on pp. 103, 107, and 698.

David Ilcinkas. Oblivious robots on graphs: exploration. *Distributed Computing by Mobile Entities, LNCS 11340, pp. 218–233*, 2019. doi:10.1007/978-3-030-11072-7_9, pdf, link. Cited on pp. 165, 166, 167, 170, 171, 172, 173, 174, and 704.

David Ilcinkas, Ahmed M. Wade. Exploration of carrier-based time-varying networks: The power of waiting. *Theoretical Computer Science 841:50–61*, 2020. doi:10.1016/j.tcs.2020.07.003. Cited on p. 87.

David Ilcinkas, Ahmed M. Wade. Exploration of Dynamic Cactuses with Sub-logarithmic Overhead. *Theory of Computing Systems 65(2):257–273*, 2021. doi:10.1007/s00224-020-10001-0. First appeared in 2020. Cited on p. 87.

Taichi Inoue, Naoki Kitamura, Taisuke Izumi, Toshimitsu Masuzawa. Computational Power of a Single Oblivious Mobile Agent in Two-Edge-Connected Graphs. *OPODIS 2022 (LIPIcs 253), 11:1–11:18*, 2022. doi:10.4230/LIPIcs.OPODIS.2022.11, pdf, arXiv:2211.00332 (pdf). Cited on pp. 208 and 210.

Giuseppe F. Italiano, Debasish Pattanayak, Gokarna Sharma. Dispersion of Mobile Robots on Directed Anonymous Graphs. *Journal of Parallel and Distributed Computing 204:105139*, 2025. doi:10.1016/j.jpdc.2025.105139. Other versions: *SIROCCO 2022 (LNCS 13298)* (doi:10.1007/978-3-031-09993-9_11). First appeared in 2022. Cited on pp. 6, 7, 8, 11, 13, 280, 284, 285, 286, 648, and 706.

Taisuke Izumi, Samia Souissi, Yoshiaki Katayama, Nobuhiro Inuzuka, Xavier Défago, Koichi Wada, Masafumi Yamashita. The gathering problem for two oblivious robots with unreliable compasses. *SIAM Journal on Computing 41(1):26–46*, 2012. doi:10.1137/100797916, arXiv:1111.1492 (pdf). First appeared in 2011. Read: arXiv v1 (2011). Cited on pp. 18, 21, 22, 63, 434, 445, and 470.

Taisuke Izumi, Daichi Kaino, Maria Gradinariu Potop-Butucaru, Sébastien Tixeuil. On time complexity for connectivity-preserving scattering of mobile robots. *Theoretical Computer Science 738:42–52*, 2018. doi:10.1016/j.tcs.2018.04.047. Other versions: *SSS 2010*, as "Connectivity-Preserving Scattering of Mobile Robots with Limited Visibility" (doi:10.1007/978-3-642-16023-3_27). First appeared in 2010. Cited on pp. 475, 479, 480, and 696.

Taisuke Izumi, Kazuki Kakizawa, Yuya Kawabata, Naoki Kitamura, Toshimitsu Masuzawa. Deciding a Graph Property by a Single Mobile Agent: One-Bit Memory Suffices. *arXiv preprint*, 2022. arXiv:2209.01906 (pdf). Preprint, not peer reviewed. Cited on pp. 88, 368, 369, 376, and 706.

Saswata Jana, Giuseppe F. Italiano, Partha Sarathi Mandal. Graph Traversal via Connected Mobile Agents. *ALGOWIN 2025, LNCS, pp. 119–134*, 2025. doi:10.1007/978-3-032-09120-8_9, arXiv:2508.18683 (pdf). Read: arXiv v1 (2025). Cited on pp. 246, 248, and 251.

Saswata Jana, Subhajit Pramanick, Adri Bhattacharya, Partha Sarathi Mandal. Time-Optimal Asynchronous Minimal Vertex Covering by Myopic Robots on Graph. *ALGOWIN 2025, LNCS, pp. 135–150*, 2025. doi:10.1007/978-3-032-09120-8_10, arXiv:2508.14247 (pdf). Read: arXiv v1 (2025). Cited on pp. 178 and 348.

Khaled Jawhar. Bike Assisted Linear Search and Evacuation. *Master's thesis, Carleton University*, 2020. doi:10.22215/etd/2020-14107, pdf. Cited on p. 499.

Khaled Jawhar, Evangelos Kranakis. Robot Evacuation on a Line Assisted by a Bike. *Information 12(1):28*, 2021. doi:10.3390/info12010028. Other versions: *SOFSEM 2021, LNCS, pp. 104–118*, as "Bike Assisted Evacuation on a Line" (doi:10.1007/978-3-030-67731-2_8). Read: SOFSEM 2021. Cited on pp. 499 and 507.

Khaled Jawhar, Evangelos Kranakis. Bike-assisted evacuation of robots on a line with asymmetric S/R communication. *Theoretical Computer Science 1053:115438*, 2025. doi:10.1016/j.tcs.2025.115438, arXiv:2307.15808 (pdf). Other versions: *ALGOWIN 2024, LNCS, pp. 91–105*, as "Bike Assisted Evacuation on a Line of Robots with S/R Communication Faults" (doi:10.1007/978-3-031-74580-5_7). First appeared in 2023. Cited on p. 533.

Khaled Jawhar, Evangelos Kranakis. Linear Search for Capturing an Oblivious Mobile Target in the Sender/Receiver Model. *ALGOWIN 2025, LNCS, pp. 151–165*, 2025. doi:10.1007/978-3-032-09120-8_11, arXiv:2508.04870 (pdf). Read: arXiv v1 (2025), the full version. Cited on pp. 499, 546, 547, 552, 553, 554, 555, 556, 558, 562, 564, and 711.

Khaled Jawhar, Evangelos Kranakis. Capturing a Moving Target on Star Graphs by Two Communicating Mobile Agents. *arXiv preprint*, 2026. arXiv:2610.11955 (pdf). Preprint, not peer reviewed. Cited on pp. 546, 563, 564, 565, 566, and 711.

Khaled Jawhar, Evangelos Kranakis. Linear Search for Capturing a Moving Target by Two Robots in the F2F Model. *ICDCN 2026, ACM, pp. 1–10*, 2026. doi:10.1145/3772290.3772317, arXiv:2503.15688 (pdf). First appeared in 2025. Read: arXiv v2 (2025), titled "Capturing a Moving Target by Two Robots in the F2F Model". Cited on pp. 546, 547, 551, 552, 553, 554, 555, 556, 557, 562, 564, and 710.

Bo Jiang, Yuan Liu, Hao Zhang, Xuehou Tan. An On-Line Strategy of Groups Evacuation from a Convex Region in the Plane. *Mathematical Problems in Data Science, pp. 189-199*, 2015. doi:10.1007/978-3-319-25127-1_11. Cited on p. 508.

Bala Kalyanasundaram, Kirk R. Pruhs. Constructing competitive tours from local information. *Theoretical Computer Science 130(1):125–138*, 1994. doi:10.1016/0304-3975(94)90155-4. Other versions: *ICALP 1993, LNCS, pp. 102–113* (doi:10.1007/3-540-56939-1_65). First appeared in 1993. Cited on pp. 222, 224, 225, 226, 228, 230, 664, 691, and 692.

Tsunehiko Kameda, Ichiro Suzuki, Masafumi Yamashita. An alternative proof for the equivalence of ∞-searcher and 2-searcher. *Theoretical Computer Science 634:108–119*, 2016. doi:10.1016/j.tcs.2016.04.016. Cited on p. 319.

Sayaka Kamei, Anissa Lamani, Fukuhito Ooshita, Sébastien Tixeuil, Koichi Wada. Asynchronous Gathering in a Torus. *OPODIS 2021, LIPIcs 217, 9:1–9:17*, 2021. doi:10.4230/LIPIcs.OPODIS.2021.9, arXiv:2101.05421 (pdf). Cited on p. 142.

Sayaka Kamei, Sébastien Tixeuil. An asynchronous maximum independent set algorithm by myopic luminous robots on grids. *The Computer Journal 67(1):57–77 (advance access 2022)*, 2024. doi:10.1093/comjnl/bxac158, arXiv:2012.03399 (pdf). First appeared in 2020. Cited on p. 177.

Sayaka Kamei, Anissa Lamani, Fukuhito Ooshita, Sébastien Tixeuil, Koichi Wada. Gathering on rings for myopic asynchronous robots with lights. *Theory of Computing Systems 69(1):8*, 2025. doi:10.1007/s00224-024-10199-3, arXiv:1911.04757 (pdf). Other versions: *OPODIS 2019, LIPIcs 153, 27:1–27:17* (doi:10.4230/LIPIcs.OPODIS.2019.27, pdf). First appeared in 2019. Cited on p. 142.

Bogumil Kaminski, Pawel Pralat, Maria Sadza. Going in Circles: Collaborative Multi-Robot Treasure Hunting. *arXiv preprint*, 2026. arXiv:2608.12115 (pdf). Preprint, not peer reviewed. Cited on p. 517.

Frank Kammer, Sandra Kiefer, Johannes Meintrup. Temporal Exploration and Rendezvous via Weisfeiler-Leman Colorings. *arXiv preprint*, 2026. arXiv:2610.30123 (pdf). Preprint, not peer reviewed. Cited on pp. 104 and 119.

Kartikey Kant, Debasish Pattanayak, Partha Sarathi Mandal. Fort Formation by an Automaton. *COMSNETS 2021, pp. 540–547*, 2021. doi:10.1109/comsnets51098.2021.9352839, arXiv:2011.15074 (pdf). First appeared in 2020. Cited on p. 632.

Ming-Yang Kao, John H. Reif, Stephen R. Tate. Searching in an Unknown Environment: An Optimal Randomized Algorithm for the Cow-Path Problem. *Information and Computation 131(1):63–79*, 1996. doi:10.1006/inco.1996.0092. Other versions: *SODA 1993.* First appeared in 1993. Cited on pp. 38, 39, 41, 72, 78, 498, 502, 504, 505, and 692.

Branislav Katreniak. Convergence with limited visibility by asynchronous mobile robots. *SIROCCO 2011, pp. 125–137*, 2011. doi:10.1007/978-3-642-22212-2_12. Cited on pp. 24, 471, and 697.

Tanvir Kaur, Kaushik Mondal. Distance-2-Dispersion: Dispersion with Further Constraints. *NETYS 2023, LNCS, pp. 157–173*, 2023. doi:10.1007/978-3-031-37765-5_12, arXiv:2301.04938 (pdf). Read: arXiv v1 (2023). Cited on p. 260.

Tanvir Kaur, Ashish Saxena, Partha Sarathi Mandal, Kaushik Mondal. Black Hole Search by Scattered Agents on Time-Varying Dynamic Graphs. *SSS 2025, LNCS 16350, pp. 309–324*, 2025. doi:10.1007/978-3-032-11127-2_25. Other versions: Kaur et al. (2026a). Cited on p. 304.

Tanvir Kaur, Ashish Saxena, Partha Sarathi Mandal, Kaushik Mondal. Black Hole Search: Dynamics, Distribution, and Emergence. *arXiv preprint*, 2026. arXiv:2603.00766 (pdf). Other versions: Kaur et al. (2025). First appeared in 2025. Preprint, not peer reviewed. Cited on pp. 291, 294, 300, 304, 655, and 710.

Tanvir Kaur, Ashish Saxena, Partha Sarathi Mandal, Kaushik Mondal. Black hole search in dynamic graphs. *Theoretical Computer Science 1086:116237*, 2026. doi:10.1016/j.tcs.2026.116237, arXiv:2405.18367 (pdf). Other versions: *ICDCN 2025* (doi:10.1145/3700838.3700869). First appeared in 2024. Read: arXiv v2 (2025). Cited on pp. 8, 50, 74, 291, 303, 304, 655, and 708.

Tanvir Kaur, Barun Gorain, Kaushik Mondal. Maximal Independent Set via Leader Election by Mobile Agents from Arbitrary Initial Configuration Without Global Knowledge. *SN Computer Science 7, article 125*, 2026. doi:10.1007/s42979-026-04726-x, arXiv:2410.19468 (pdf). First appeared in 2024. Read: arXiv v2 (2024). Cited on pp. 60, 61, 261, 347, 348, 349, 350, and 708.

Tanvir Kaur, Ashish Saxena. When agents are powerful: Black hole search with verification in time-varying graphs. *Theoretical Computer Science 1088:116285*, 2026. doi:10.1016/j.tcs.2026.116285, arXiv:2510.22309 (pdf). Other versions: *ICDCIT 2026, pp. 3–18*, as "When Agents are Powerful: Black Hole Search in Time-Varying Graphs" (doi:10.1007/978-3-032-16632-6_1). First appeared in 2025. Read: arXiv v2 (2026). Cited on p. 291.

Akitoshi Kawamura, Yusuke Kobayashi. Fence patrolling by mobile agents with distinct speeds. *Distributed Computing 28:147–154*, 2015. doi:10.1007/s00446-014-0226-3, arXiv:1407.8194 (pdf). Other versions: *ISAAC 2012* (doi:10.1007/978-3-642-35261-4_62). First appeared in 2012. Read: arXiv v1 (2014). Cited on pp. 41, 73, 567, 568, 569, 573, and 697.

Akitoshi Kawamura, Makoto Soejima. Simple strategies versus optimal schedules in multi-agent patrolling. *Theoretical Computer Science 839:195–206*, 2020. doi:10.1016/j.tcs.2020.07.037, arXiv:1411.6853 (pdf). Other versions: *CIAC 2015, LNCS, pp. 261–273* (doi:10.1007/978-3-319-18173-8_19). First appeared in 2014. Read: arXiv v3 (2021). Cited on pp. 41, 73, 568, 569, 571, and 573.

Hridesh Kedia, Shunhao Oh, Dana Randall. Local Stochastic Algorithms for Alignment in Self-Organizing Particle Systems. *APPROX/RANDOM 2022, LIPIcs 245, 14*, 2022. doi:10.4230/LIPIcs.APPROX/RANDOM.2022.14, arXiv:2207.07956 (pdf). Read: arXiv v2 (2022). Cited on pp. 46, 70, 610, 614, 615, 616, and 706.

Yonghwan Kim, Yoshiaki Katayama, Koichi Wada. Pairbot: A Novel Model for Autonomous Mobile Robot Systems Consisting of Paired Robots. *arXiv preprint*, 2020. arXiv:2009.14426 (pdf). Read: arXiv v2 (2024). Preprint, not peer reviewed. Cited on p. 67.

Yonghwan Kim, Masahiro Shibata, Yuichi Sudo, Junya Nakamura, Yoshiaki Katayama, Toshimitsu Masuzawa. Gathering of Mobile Robots with Defected Views. *OPODIS 2022, LIPIcs 253, 14:1–14:18*, 2022. doi:10.4230/LIPIcs.OPODIS.2022.14, arXiv:2208.08159 (pdf). Other versions: *DISC 2022 (brief announcement), LIPIcs 246, 46:1–46:3*, as "Brief Announcement: Gathering Despite Defected View" (doi:10.4230/LIPIcs.DISC.2022.46). Cited on p. 433.

Yonghwan Kim, Yoshiaki Katayama, Koichi Wada. Complete Visibility Algorithm for Autonomous Mobile Luminous Robots under an Asynchronous Scheduler on Grid Plane. *arXiv preprint*, 2023. arXiv:2306.08354 (pdf). Other versions: Kim et al. (2025). Read: arXiv v1. Preprint, not peer reviewed. Cited on p. 414.

Yonghwan Kim, Yoshiaki Katayama, Koichi Wada. Complete Visibility Algorithms of Luminous Robots With Two-Color Lights on Grid. *Concurrency and Computation: Practice and Experience 37(2):e8323*, 2025. doi:10.1002/cpe.8323, pdf. Other versions: *PDAA 2023 (15th International Workshop on Parallel and Distributed Algorithms and Applications)*, as "Asynchronous Complete Visibility Algorithm for Luminous Robots on Grid"; Kim et al. (2023). First appeared in 2023. Cited on p. 414.

David Kirkpatrick, Irina Kostitsyna, Alfredo Navarra, Giuseppe Prencipe, Nicola Santoro. On the power of bounded asynchrony: convergence by autonomous robots with limited visibility. *Distributed Computing 37(3):279–308*, 2024. doi:10.1007/s00446-024-00463-7, pdf, arXiv:2105.13042 (pdf). Other versions: *PODC 2021*, as "Separating Bounded and Unbounded Asynchrony for Autonomous Robots: Point Convergence with Limited Visibility" (doi:10.1145/3465084.3467910). First appeared in 2021. Read: arXiv v1 (2021). Cited on pp. 20, 24, 27, 28, 36, 37, 469, 471, 642, 653, and 705.

Naoki Kitamura, Yuichi Sudo, Koichi Wada. Two-Robot Computational Landscape: A Complete Characterization of Model Power in Minimal Mobile Robot Systems. *arXiv preprint*, 2025. arXiv:2512.22770 (pdf). Preprint, not peer reviewed. Cited on pp. 31, 32, 33, 34, 35, 37, and 66.

Ralf Klasing, Euripides Markou, Tomasz Radzik, Fabiano Sarracco. Hardness and approximation results for Black Hole Search in arbitrary networks. *Theoretical Computer Science 384(2–3):201–221*, 2007. doi:10.1016/j.tcs.2007.04.024. Other versions: *SIROCCO 2005*, as "Hardness and Approximation Results for Black Hole Search in Arbitrary Graphs" (doi:10.1007/11429647_17). First appeared in 2005. Cited on pp. 291, 299, and 694.

Ralf Klasing, Euripides Markou, Andrzej Pelc. Gathering asynchronous oblivious mobile robots in a ring. *Theoretical Computer Science 390(1):27–39*, 2008. doi:10.1016/j.tcs.2007.09.032. Other versions: *ISAAC 2006* (doi:10.1007/11940128_74). First appeared in 2006. Cited on pp. 136, 139, 140, 141, 143, 144, 145, 149, 150, 155, 665, 694, and 695.

Ralf Klasing, Adrian Kosowski, Dominik Pająk, Thomas Sauerwald. The multi-agent rotor-router on the ring: a deterministic alternative to parallel random walks. *Distributed Computing 30(2):127–148*, 2016. doi:10.1007/s00446-016-0282-y. First appeared in 2013. Read: the HAL version. Cited on p. 208.

Yuta Kojima, Sébastien Tixeuil, Yukiko Yamauchi. Hardwired Pattern Formation by Mobile Robots with Common Unit Distance. *arXiv preprint*, 2026. arXiv:2606.22386 (pdf). Preprint, not peer reviewed. Cited on p. 382.

Amos Korman, Yoav Rodeh. Parallel Linear Search with no Coordination for a Randomly Placed Treasure. *arXiv preprint*, 2016. arXiv:1602.04952 (pdf). Read: arXiv v2 (2016). Preprint, not peer reviewed. Cited on p. 110.

Adrian Kosowski, Alfredo Navarra, Cristina M. Pinotti. Synchronous black hole search in directed graphs. *Theoretical Computer Science 412(41):5752–5759*, 2011. doi:10.1016/j.tcs.2011.05.054. Other versions: *OPODIS 2009, LNCS 5923, pp. 86–98*, as "Synchronization helps robots to detect black holes in directed graphs" (doi:10.1007/978-3-642-10877-8_9). First appeared in 2009. Cited on p. 292.

Adrian Kosowski, Alfredo Navarra. Graph decomposition for memoryless periodic exploration. *Algorithmica 63(1–2):26–38*, 2012. doi:10.1007/s00453-011-9518-1. Other versions: *MFCS 2009*, as "Graph Decomposition for

Improving Memoryless Periodic Exploration" (doi:10.1007/978-3-642-03816-7_43). First appeared in 2009. Cited on pp. 100 and 644.

Adrian Kosowski, Dominik Pająk. Does adding more agents make a difference? A case study of cover time for the rotor-router. *Journal of Computer and System Sciences 106:80–93*, 2019. doi:10.1016/j.jcss.2019.07.001. First appeared in 2014. Read: the HAL version. Cited on p. 208.

Irina Kostitsyna, Tom Peters, Bettina Speckmann. Brief Announcement: An Effective Geometric Communication Structure for Programmable Matter. *DISC 2022 (brief announcement), LIPIcs 246, 47:1–47:3*, 2022. doi:10.4230/LIPIcs.DISC.2022.47. Cited on p. 601.

Irina Kostitsyna, Christian Scheideler, Daniel Warner. Fault-Tolerant Shape Formation in the Amoebot Model. *DNA 28 (2022), LIPIcs 238, 9:1–9:22*, 2022. doi:10.4230/LIPIcs.DNA.28.9. Other versions: *SAND 2022 (brief announcement), LIPIcs 221, 23:1–23:3*, as "Brief Announcement: Fault-Tolerant Shape Formation in the Amoebot Model" (doi:10.4230/LIPIcs.SAND.2022.23). Cited on p. 601.

Irina Kostitsyna, Tom Peters, Bettina Speckmann. Fast Reconfiguration for Programmable Matter. *DISC 2023, LIPIcs 281, 27*, 2023. doi:10.4230/LIPIcs.DISC.2023.27, pdf, arXiv:2202.11663 (pdf). First appeared in 2022. Read: arXiv v4 (2026). Cited on p. 600.

Irina Kostitsyna, David Liedtke, Christian Scheideler. Universal Coating by 3D Hybrid Programmable Matter. *SIROCCO 2024, LNCS, pp. 384–401*, 2024. doi:10.1007/978-3-031-60603-8_21, arXiv:2303.16180 (pdf). First appeared in 2023. Read: arXiv v4 (2024). Cited on pp. 606, 607, and 632.

Michal Koucký. Universal traversal sequences with backtracking. *Journal of Computer and System Sciences 65(4):717–726*, 2002. doi:10.1016/S0022-0000(02)00023-5. Other versions: *CCC 2001*. First appeared in 2001. Cited on pp. 94 and 97.

Dariusz R. Kowalski, Adam Malinowski. How to meet in anonymous network. *Theoretical Computer Science 399(1–2):141–156*, 2008. doi:10.1016/j.tcs.2008.02.010. Other versions: *SIROCCO 2006* (doi:10.1007/11780823_5). First appeared in 2006. Cited on pp. 118, 121, 125, and 126.

Evangelos Kranakis. Invited Paper: A Survey of the Impact of Knowledge on the Competitive Ratio in Linear Search. *Stabilization, Safety, and Security of Distributed Systems, pp. 23-38*, 2024. doi:10.1007/978-3-031-74498-3_2. Cited on pp. 507, 546, and 547.

Evangelos Kranakis. Searching the Plane in the Segment Visibility Model. *arXiv preprint*, 2026. arXiv:2611.03161 (pdf). Preprint, not peer reviewed. Cited on pp. 574, 575, 576, 577, 578, 579, 580, 663, and 711.

Dominik Krupke, Michael Hemmer, James McLurkin, Yu Zhou, Sándor P. Fekete. A Parallel Distributed Strategy for Arraying a Scattered Robot Swarm. *IROS 2015, IEEE, pp. 2795–2802*, 2015. doi:10.1109/IROS.2015.7353761, arXiv:1505.03128 (pdf). Read: arXiv v1 (2015). Cited on p. 382.

Ajay D. Kshemkalyani, Faizan Ali. Efficient Dispersion of Mobile Robots on Graphs. *ICDCN 2019*, 2019. doi:10.1145/3288599.3288610, arXiv:1805.12242 (pdf). First appeared in 2018. Read: arXiv v1 (2018). Cited on pp. 6, 7, 11, 14, 15, 57, 60, 77, 257, 259, 260, 265, 266, 267, 268, 275, 278, 279, 280, 281, 282, 283, 646, and 702.

Ajay D. Kshemkalyani, Anisur Rahaman Molla, Gokarna Sharma. Fast Dispersion of Mobile Robots on Arbitrary Graphs. *ALGOSENSORS 2019 (LNCS 11931)*, 2019. doi:10.1007/978-3-030-34405-4_2, arXiv:1812.05352 (pdf). First appeared in 2018. Read: arXiv v3 (2019). Cited on p. 267.

Ajay D. Kshemkalyani, Anisur Rahaman Molla, Gokarna Sharma. Efficient Dispersion of Mobile Robots on Dynamic Graphs. *ICDCS 2020*, 2020. doi:10.1109/ICDCS47774.2020.00100, link. Cited on pp. 78, 280, 286, 287, 649, and 704.

Ajay D. Kshemkalyani, Anisur Rahaman Molla, Gokarna Sharma. Dispersion of Mobile Robots in the Global Communication Model. *ICDCN 2020*, 2020. doi:10.1145/3369740.3369775, arXiv:1909.01957 (pdf). First appeared in 2019. Read: arXiv v1 (2019). Cited on pp. 6, 13, 269, 278, and 647.

Ajay D. Kshemkalyani, Anisur Rahaman Molla, Gokarna Sharma. Dispersion of Mobile Robots on Grids. *WALCOM 2020 (LNCS 12049)*, 2020. doi:10.1007/978-3-030-39881-1_16, arXiv:1812.05352 (pdf). First appeared in 2018. Read: arXiv v3 (2019). Cited on pp. 15, 77, 261, 271, 272, 273, 274, 275, 647, and 702.

Ajay D. Kshemkalyani, Anisur Rahaman Molla, Gokarna Sharma. Dispersion of mobile robots using global communication. *Journal of Parallel and Distributed Computing 161:100–117*, 2022. doi:10.1016/j.jpdc.2021.11.007. Cited on pp. 253, 269, 277, 278, 646, and 706.

Ajay D. Kshemkalyani, Manish Kumar, Anisur Rahaman Molla, Gokarna Sharma. Brief Announcement: Agent-Based Leader Election, MST, and Beyond. *DISC 2024 brief announcement (LIPIcs 319), 50:1–50:7*, 2024.

doi:10.4230/LIPIcs.DISC.2024.50, pdf, arXiv:2403.13716 (pdf). Other versions: Kshemkalyani et al. (2025a). Read: arXiv v2 (2024). Cited on pp. 54, 60, 61, 340, 341, 342, 343, 344, 347, 348, 349, 351, 353, 356, 358, 360, 361, 362, 363, 365, and 708.

Ajay D. Kshemkalyani, Gokarna Sharma. Near-Optimal Dispersion on Arbitrary Anonymous Graphs. *Journal of Computer and System Sciences 152:103656*, 2025. doi:10.1016/j.jcss.2025.103656, arXiv:2106.03943 (pdf). Other versions: *OPODIS 2021 (LIPIcs 217)* (doi:10.4230/LIPIcs.OPODIS.2021.8, pdf). First appeared in 2021. Read: arXiv v1 (2021). Cited on pp. 266 and 267.

Ajay D. Kshemkalyani, Manish Kumar, Anisur Rahaman Molla, Gokarna Sharma. Agentic Distributed Computing. *arXiv preprint*, 2025. arXiv:2507.04459 (pdf). Other versions: Kshemkalyani et al. (2024). First appeared in 2024. Preprint, not peer reviewed. Cited on pp. 343 and 360.

Ajay D. Kshemkalyani, Manish Kumar, Anisur Rahaman Molla, Debasish Pattanayak, Gokarna Sharma. Dispersion is (Almost) Optimal under (A)synchrony. *SPAA 2025*, 2025. doi:10.1145/3694906.3743317, arXiv:2503.16216 (pdf). Read: arXiv v1 (2025). Cited on pp. 54, 55, 60, 75, 77, 78, 257, 259, 260, 266, 267, 268, 269, 275, 276, 277, 278, 279, 342, 344, 352, 642, 645, and 709.

Ajay D. Kshemkalyani. Dispersion of mobile robots on graphs in the asynchronous model. *Theoretical Computer Science 1044:115272*, 2025. doi:10.1016/j.tcs.2025.115272. Cited on pp. 265, 266, and 645.

Ajay D. Kshemkalyani, Manish Kumar, Anisur Rahaman Molla, Gokarna Sharma. Near-Linear Time Leader Election in Multiagent Networks. *AAMAS 2025, pp. 1218–1226*, 2025. doi:10.65109/BKZJ3357. Cited on pp. 61, 344, 345, 362, and 710.

Ajay D. Kshemkalyani, Manish Kumar, Anisur Rahaman Molla, Debasish Pattanayak, Gokarna Sharma. Faster leader election via mobile agents and its applications. *Theoretical Computer Science 1059:115578*, 2026. doi:10.1016/j.tcs.2025.115578. Other versions: *ICDCIT 2025, LNCS 15507, pp. 108–123*, as "Faster Leader Election and Its Applications for Mobile Agents with Parameter Advice" (doi:10.1007/978-3-031-81404-4_9). First appeared in 2025. Cited on pp. 16, 261, 340, 342, 343, 345, 347, 350, 362, and 710.

Fabian Kuhn, Nancy Lynch, Rotem Oshman. Distributed computation in dynamic networks. *STOC 2010, pp. 513–522*, 2010. doi:10.1145/1806689.1806760, link. Cited on pp. 6 and 10.

Manish Kumar, Anisur Rahaman Molla, Debasish Pattanayak, Subhajit Pramanick. Beyond the Boundary: Optimal Grid Dispersion by Spread, Reverse, and Collect. *arXiv preprint*, 2026. link, arXiv:2610.16935 (pdf). Preprint, not peer reviewed. Cited on pp. 261, 271, 272, 273, 274, 646, 647, and 711.

Manish Kumar, Othon Michail, Andreas Padalkin, Christian Scheideler. Sublinear-Time Reconfiguration of Programmable Matter with Joint Movements. *SAND 2026, LIPIcs 373, 11:1–11:17*, 2026. doi:10.4230/LIPIcs.SAND.2026.11, pdf, link, arXiv:2603.10720 (pdf). Read: arXiv v1 (2026). Cited on p. 636.

Manash Kumar Kundu, Pritam Goswami, Satakshi Ghosh, Buddhadeb Sau. Arbitrary pattern formation by asynchronous opaque robots on infinite grid. *arXiv preprint*, 2022. arXiv:2205.03053 (pdf). Preprint, not peer reviewed. Cited on p. 382.

Andrey Kupavskii, Emo Welzl. Lower bounds for searching robots, some faulty. *Distributed Computing 34(4):229–237*, 2021. doi:10.1007/s00446-019-00358-y, arXiv:1707.05077 (pdf). Other versions: *PODC 2018*. First appeared in 2017. Read: arXiv v2 (2018). Cited on pp. 39, 40, 51, 72, 75, 78, 496, 498, 499, 502, 503, 504, 534, 691, and 701.

Shay Kutten. Stepwise construction of an efficient distributed traversing algorithm for general strongly connected directed networks, or: Traversing one way streets with no map. *ICCC 1988 (Computer Communication Technologies for the 90's), pp. 446–452*, 1988. Cited on p. 103.

Ivan Lahtin, Viktor Zamaraev. Exploration of k-edge-deficient temporal graphs in linear time. *Discrete Mathematics 350(2):115451*, 2027. doi:10.1016/j.disc.2026.115451, arXiv:2605.15833 (pdf). First appeared in 2026. Read: arXiv v1 (2026). Cited on pp. 8, 104, 105, and 644.

Anissa Lamani, Maria Gradinariu Potop-Butucaru, Sébastien Tixeuil. Optimal Deterministic Ring Exploration with Oblivious Asynchronous Robots. *SIROCCO 2010, LNCS 6058, pp. 183–196*, 2010. doi:10.1007/978-3-642-13284-1_15, arXiv:0910.0832 (pdf). First appeared in 2009. Read: arXiv v1 (2009). Cited on pp. 163, 164, 165, 166, 167, 169, 170, 171, 176, 660, 670, 686, and 696.

Ioannis Lamprou, Russell Martin, Sven Schewe. Fast two-robot disk evacuation with wireless communication. *Theoretical Computer Science 846:38–60*, 2020. doi:10.1016/j.tcs.2020.09.019, arXiv:1604.04217 (pdf). Other versions: *DISC 2016, LNCS, pp. 1–15* (doi:10.1007/978-3-662-53426-7_1). First appeared in 2016. Read: arXiv v1 (2016). Cited on pp. 41, 73, 516, 653, and 701.

Nikos Leonardos, Aris Pagourtzis, Ioannis Papaioannou. Byzantine fault tolerant symmetric-persistent circle evacuation. *Theoretical Computer Science 956:113839*, 2023. doi:10.1016/j.tcs.2023.113839. Other versions: *ALGOSENSORS 2021, LNCS, pp. 111–123* (doi:10.1007/978-3-030-89240-1_8). First appeared in 2021. Cited on pp. 523, 525, 528, 529, and 705.

Songhua Li, Yinfeng Xu. Online Strategies for Evacuating from a Convex Region in the Plane. *Frontiers in Algorithmics, pp. 151-162*, 2017. doi:10.1007/978-3-319-59605-1_14. Cited on p. 508.

Shouwei Li, Christine Markarian, Friedhelm Meyer auf der Heide, Pavel Podlipyan. A continuous strategy for collisionless gathering. *Theoretical Computer Science 852:41–60*, 2021. doi:10.1016/j.tcs.2020.10.037. Other versions: *ALGOSENSORS 2017*. First appeared in 2017. Cited on pp. 434 and 462.

Alberto Quattrini Li, Francesco Amigoni, Nicola Basilico. Searching for Optimal Off-Line Exploration Paths in Grid Environments for a Robot with Limited Visibility. *Proceedings of the AAAI Conference on Artificial Intelligence 26(1):2060–2066*, 2021. doi:10.1609/aaai.v26i1.8381. Cited on p. 87.

Yuan Liu, Bo Jiang, Hao Zhang. Online Strategies for Evacuating from a Convex Region by Groups in the Plane. *2015 Ninth International Conference on Frontier of Computer Science and Technology, pp. 178-183*, 2015. doi:10.1109/fcst.2015.70. Cited on p. 508.

Zhiqiang Liu, Yukiko Yamauchi, Shuji Kijima, Masafumi Yamashita. Team assembling problem for asynchronous heterogeneous mobile robots. *Theoretical Computer Science 721:27–41*, 2018. doi:10.1016/j.tcs.2018.01.009. Cited on p. 426.

Luan Liu, Qi Wei, Yan Zhang, Jie Sun, Wenxin Zhang. Online Algorithm for Evacuating Crashed Faulty Robots from A Square. *2022 IEEE 8th International Conference on Computer and Communications (ICCC), 2276-2280*, 2022. doi:10.1109/iccc56324.2022.10065611. Cited on p. 508.

Fabrizio Luccio, Linda Pagli. More agents may decrease global work: A case in butterfly decontamination. *Theoretical Computer Science 655:41–57*, 2016. doi:10.1016/j.tcs.2016.09.027. Cited on p. 319.

Ulysse Léchine, Sébastien Tixeuil. Asynchronous Scattering. *arXiv preprint*, 2019. arXiv:1905.09177 (pdf). Preprint, not peer reviewed. Cited on pp. 475, 478, 479, 656, and 703.

Yuan Ma, Yiqun Lisa Yin. On the Cyclic Assumption of the Cow-Path Search Algorithm. *arXiv preprint*, 2026. arXiv:2610.10253 (pdf). Preprint, not peer reviewed. Cited on pp. 502, 504, 505, and 711.

Animesh Maiti, Prakhar Shukla, Subhash Bhagat. Conflicting Pattern Formation by Teams of Anonymous, Fully Disoriented Robots. *arXiv preprint*, 2026. arXiv:2609.23454 (pdf). Other versions: *SSRN preprint*, 2026 (doi:10.2139/ssrn.7572020). Preprint, not peer reviewed. Cited on p. 395.

Animesh Maiti, Prakhar Shukla, Subhash Bhagat. Min-Max Uniform Circle Formation by Asynchronous Mobile Robots. *arXiv preprint*, 2026. arXiv:2610.05008 (pdf). Preprint, not peer reviewed. Cited on p. 399.

Animesh Maiti, Abhinav Chakraborty, Bibhuti Das, Subhash Bhagat, Krishnendu Mukhopadhyaya. Min-Sum Uniform Coverage Problem by Autonomous Mobile Robots. *arXiv preprint*, 2026. arXiv:2602.11125 (pdf). Preprint, not peer reviewed. Cited on p. 491.

Animesh Maiti, Abhinav Chakraborty, Subhash Bhagat. The Surplus Parking Gathering Problem in Infinite Grids. *arXiv preprint*, 2026. arXiv:2607.05983 (pdf). Preprint, not peer reviewed. Cited on pp. 7 and 142.

Animesh Maiti, Prakhar Shukla, Abhinav Chakraborty, Subhash Bhagat. When Robots Crash: Optimal Asynchronous Gathering at Weber Meeting Nodes. *arXiv preprint*, 2026. arXiv:2610.06939 (pdf). Preprint, not peer reviewed. Cited on pp. 7, 142, and 144.

Marcello Mamino, Giovanni Viglietta. Square formation by asynchronous oblivious robots. *CCCG 2016*, 2016. arXiv:1605.06093 (pdf). Read: arXiv v2 (2016). Cited on pp. 65, 398, 399, and 401.

Subhrangsu Mandal, Anisur Rahaman Molla, William K. Moses Jr. Efficient live exploration of a dynamic ring with mobile robots. *Theoretical Computer Science 980:114201*, 2023. doi:10.1016/j.tcs.2023.114201, arXiv:2001.04525 (pdf). Other versions: *ALGOSENSORS 2020, LNCS, pp. 92–107*, as "Live Exploration with Mobile Robots in a Dynamic Ring, Revisited" (doi:10.1007/978-3-030-62401-9_7). First appeared in 2020. Read: arXiv v1 (2020). Cited on p. 309.

Artur Menc, Dominik Pająk, Przemysław Uznański. Brief announcement: On the power of one bit in graph exploration without backtracking. *DISC 2015 (brief announcement)*, 2015. pdf, link. Cited on p. 86.

Artur Menc, Dominik Pająk, Przemysław Uznański. Time and space optimality of rotor-router graph exploration. *Information Processing Letters 127:17–20*, 2017. doi:10.1016/j.ipl.2017.06.010, arXiv:1502.05545 (pdf). First appeared in 2015. Cited on pp. 206, 207, 208, 209, 210, 211, 219, 662, and 700.

Othon Michail, Paul G. Spirakis. Traveling salesman problems in temporal graphs. *Theoretical Computer Science 634:1–23*, 2016. doi:10.1016/j.tcs.2016.04.006. Other versions: *MFCS 2014, LNCS, pp. 553–564* (doi:10.1007/978-3-662-44465-8_47). First appeared in 2014. Cited on p. 89.

Othon Michail, George Skretas, Paul G. Spirakis. On the transformation capability of feasible mechanisms for programmable matter. *Journal of Computer and System Sciences 102:18–39*, 2019. doi:10.1016/j.jcss.2018.12.001, arXiv:1703.04381 (pdf). First appeared in 2017. Read: arXiv v1 (2017). Cited on pp. 43, 47, 52, 77, 634, 635, 636, 637, 638, 639, 664, 701, and 702.

Avery Miller, Andrzej Pelc. Fast rendezvous with advice. *Theoretical Computer Science 608:190–198*, 2015. doi:10.1016/j.tcs.2015.09.025, arXiv:1407.1428 (pdf). Other versions: *ALGOSENSORS 2015, LNCS, pp. 75–87* (doi:10.1007/978-3-662-46018-4_5). First appeared in 2014. Cited on p. 120.

Avery Miller, Andrzej Pelc. Tradeoffs between cost and information for rendezvous and treasure hunt. *Journal of Parallel and Distributed Computing 83:159–167*, 2015. doi:10.1016/j.jpdc.2015.06.004, arXiv:1506.07952 (pdf). Other versions: *OPODIS 2014* (doi:10.1007/978-3-319-14472-6_18). First appeared in 2014. Read: arXiv v1 (2015). Cited on pp. 58, 109, 111, 114, and 699.

Avery Miller, Andrzej Pelc. Time versus cost tradeoffs for deterministic rendezvous in networks. *Distributed Computing 29(1):51–64*, 2016. doi:10.1007/s00446-015-0253-8, arXiv:1508.02471 (pdf). Other versions: *PODC 2014, ACM, pp. 282–290* (doi:10.1145/2611462.2611473). First appeared in 2014. Cited on p. 120.

Avery Miller, Ullash Saha. Fast Byzantine Gathering with Visibility in Graphs. *ALGOSENSORS 2020, LNCS, pp. 140–153*, 2020. doi:10.1007/978-3-030-62401-9_10, arXiv:2009.01544 (pdf). Cited on p. 133.

Avery Miller, Andrzej Pelc. Fast deterministic rendezvous in labeled lines. *Distributed Computing 38(4):435–489*, 2025. doi:10.1007/s00446-025-00489-5, arXiv:2311.12976 (pdf). Other versions: *DISC 2023, LIPIcs 281, 29:1–29:22*, as "Fast Deterministic Rendezvous in Labeled Lines" (doi:10.4230/LIPIcs.DISC.2023.29, pdf). First appeared in 2023. Read: arXiv v1 (2023), the full version; theorem numbers refer to it. Cited on pp. 129, 130, and 707.

P. Mironowicz. Entangled rendezvous: a possible application of Bell non-locality for mobile agents on networks. *New Journal of Physics 25(1):013023*, 2023. doi:10.1088/1367-2630/acb22d. Cited on p. 120.

Shuichi Miyazaki, Naoyuki Morimoto, Yasuo Okabe. The Online Graph Exploration Problem on Restricted Graphs. *IEICE Transactions on Information and Systems E92-D(9):1620–1627*, 2009. doi:10.1587/transinf.e92.d.1620, pdf. Cited on pp. 223, 224, 225, 231, 232, and 696.

Anisur Rahaman Molla, William K. Moses Jr. Dispersion of Mobile Robots: The Power of Randomness. *TAMC 2019 (LNCS 11436)*, 2019. doi:10.1007/978-3-030-14812-6_30, arXiv:1902.10489 (pdf). Read: arXiv v1 (2019). Cited on pp. 6, 11, 12, 16, 54, 55, 60, 78, 276, 278, 282, 283, 647, 648, 685, and 703.

Anisur Rahaman Molla, Kaushik Mondal, William K. Moses Jr. Optimal dispersion on an anonymous ring in the presence of weak Byzantine robots. *Theoretical Computer Science 887:111–121*, 2021. doi:10.1016/j.tcs.2021.07.008, arXiv:2004.11439 (pdf). Other versions: *ALGOSENSORS 2020 (LNCS 12503)*, as "Efficient Dispersion on an Anonymous Ring in the Presence of Weak Byzantine Robots" (doi:10.1007/978-3-030-62401-9_11). First appeared in 2020. Read: arXiv v2 (2020). Cited on pp. 74, 75, 277, 281, 647, 648, and 704.

Anisur Rahaman Molla, Kaushik Mondal, William K. Moses Jr. Byzantine Dispersion on Graphs. *IPDPS 2021*, 2021. doi:10.1109/IPDPS49936.2021.00103, arXiv:2102.07528 (pdf). Read: arXiv v2 (2021). Cited on pp. 7, 11, 13, 15, 55, 74, 75, 78, 79, 280, 281, 648, and 705.

Anisur Rahaman Molla, Kaushik Mondal, William K. Moses Jr. Fast Deterministic Gathering with Detection on Arbitrary Graphs: The Power of Many Robots. *IPDPS 2023, pp. 47–57*, 2023. doi:10.1109/IPDPS54959.2023.00015, arXiv:2305.01753 (pdf). Cited on pp. 132, 133, 136, and 707.

Moumita Mondal, Sruti Gan Chaudhuri. Uniform Circle Formation by Transparent Fat Robots. *arXiv preprint*, 2017. arXiv:1710.09423 (pdf). Preprint, not peer reviewed. Cited on pp. 403 and 701.

Moumita Mondal, Sruti Gan Chaudhuri, Ayan Dutta, Krishnendu Mukhopadhyaya, Punyasha Chatterjee. Uniform Circle Formation By Oblivious Swarm Robots. *arXiv preprint*, 2020. arXiv:2012.07113 (pdf). Preprint, not peer reviewed. Cited on pp. 404 and 704.

Moumita Mondal, Sruti Gan Chaudhuri. Uniform Scattering of Robots on Alternate Nodes of a Grid. *ICDCN 2022, ACM, pp. 254–259*, 2022. doi:10.1145/3491003.3493231, arXiv:2012.07112 (pdf). First appeared in 2020. Read: arXiv v1 (2020). Cited on p. 475.

Brati Mondal, Pritam Goswami, Avisek Sharma, Buddhadeb Sau. Arbitrary pattern formation on a continuous

circle by oblivious robot swarm. *Theoretical Computer Science 1021:114882*, 2024. doi:10.1016/j.tcs.2024.114882, arXiv:2303.10366 (pdf). First appeared in 2023. Read: arXiv v2 (2023). Cited on p. 382.

Moumita Mondal, Subodh Singh, Sruti Gan Chaudhuri. K Circle Formation by Swarm Robots on a Grid. *ICDCN 2024 (short paper), ACM, pp. 334–339*, 2024. doi:10.1145/3631461.3632304, pdf. Cited on p. 466.

Brati Mondal, Pritam Goswami, Buddhadeb Sau. Time Optimal Distance-k-Dispersion on Dynamic Ring. *ICDCN 2025*, 2025. doi:10.1145/3700838.3700843, arXiv:2408.12220 (pdf). First appeared in 2024. Read: arXiv v1 (2024). Cited on pp. 288 and 708.

Brati Mondal, Pritam Goswami, Buddhadeb Sau. Simulating Chirality: Solving Distance-k-Dispersion on an 1-Interval Connected Ring. *arXiv preprint*, 2025. arXiv:2507.14723 (pdf). Preprint, not peer reviewed. Cited on pp. 288 and 649.

Akihiro Monde, Yukiko Yamauchi, Shuji Kijima, Masafumi Yamashita. Self-stabilizing Localization of the Middle Point of a Line Segment by an Oblivious Robot with Limited Visibility. *SSS 2017, LNCS, pp. 172–186*, 2017. doi:10.1007/978-3-319-69084-1_12, arXiv:1707.06398 (pdf). Read: arXiv v2. Cited on p. 67.

Oscar Morales-Ponce. Optimal patrolling of high priority segments while visiting the unit interval with a set of mobile robots. *Theoretical Computer Science 911:1–18*, 2022. doi:10.1016/j.tcs.2022.01.046, arXiv:1910.01250 (pdf). First appeared in 2019. Read: arXiv v1 (2019). Cited on p. 568.

William K. Moses Jr., Amanda Redlich. Dispersion, Capacitated Nodes, and the Power of a Trusted Shepherd. *ICDCN 2024*, 2024. doi:10.1145/3631461.3632310, arXiv:2311.01511 (pdf). First appeared in 2023. Read: arXiv v1 (2023). Cited on pp. 75, 281, 282, 648, and 707.

William K. Moses Jr., Amanda Redlich, Frederick Stock. Brief Announcement: Broadcast via Mobile Agents in a Dynamic Network: Interplay of Graph Properties & Agents. *SAND 2025 (brief announcement), LIPIcs 330, 17:1–17:5*, 2025. doi:10.4230/LIPIcs.SAND.2025.17, pdf, arXiv:2504.05442 (pdf). Other versions: *SSRN preprint*, 2026, as "Broadcast via Mobile Agents in a Dynamic Network: Interplay of Graph Properties & Agents" (doi:10.2139/ssrn.6411400). Read: arXiv v2 (2026), the full version; theorem numbers refer to it. Cited on pp. 7, 325, 326, 327, 328, 665, and 710.

William K. Moses Jr., Amanda Redlich, Frederick Stock. Broadcasting Agents and Adversary: A new variation on Cops and Robbers. *arXiv preprint*, 2026. arXiv:2603.11958 (pdf). Preprint, not peer reviewed. Cited on pp. 325, 326, 327, 328, 330, 665, and 711.

Dolev Mutzari, Yonatan Aumann, Sarit Kraus. Competitive Connected Multi-robot Exploration of Unknown Graphs. *IJCAI 2026, pp. 297–305*, 2026. doi:10.24963/ijcai.2026/34. Cited on pp. 248, 251, 644, 673, and 711.

Shota Nagahama, Fukuhito Ooshita, Michiko Inoue. Terminating grid exploration with myopic luminous robots. *International Journal of Networking and Computing 12(1):73–102*, 2022. doi:10.15803/ijnc.12.1_73, arXiv:2102.06006 (pdf). Other versions: *APDCM 2021, IPDPS Workshops, pp. 586–595* (doi:10.1109/ipdpsw52791.2021.00092). First appeared in 2021. Cited on pp. 177, 180, 181, 186, 187, 188, 189, 190, 193, 661, and 705.

Shota Nagahama, Fukuhito Ooshita, Michiko Inoue. Ring exploration of myopic luminous robots with visibility more than one. *Information and Computation 292:105036*, 2023. doi:10.1016/j.ic.2023.105036. Other versions: *SSS 2019, LNCS, pp. 256–271* (doi:10.1007/978-3-030-34992-9_20). First appeared in 2019. Cited on pp. 176, 196, 197, 199, 200, and 704.

Shota Naito, Tsukasa Ninomiya, Koichi Wada. Beyond Pairwise Comparisons: Unveiling Structural Landscape of Mobile Robot Models. *SSS 2025, LNCS, pp. 408–424*, 2025. doi:10.1007/978-3-032-11127-2_31, arXiv:2508.19805 (pdf). Read: arXiv v3 (2025). Cited on pp. 27, 28, 29, 30, 34, 35, and 37.

Rikuo Nakai, Yuichi Sudo, Koichi Wada. Asynchronous Gathering Algorithms for Autonomous Mobile Robots with Lights. *SSS 2021, LNCS, pp. 410–424*, 2021. doi:10.1007/978-3-030-91081-5_27, arXiv:2109.12289 (pdf). Read: arXiv v2 (2021). Cited on pp. 66, 67, 446, 447, 651, and 705.

Keita Nakajima, Kaito Takase, Koichi Wada. Efficient Self-stabilizing Simulations of Energy-Restricted Mobile Robots by Asynchronous Luminous Mobile Robots. *SSS 2024, LNCS, pp. 141–155*, 2024. doi:10.1007/978-3-031-74498-3_10, arXiv:2403.05542 (pdf). Other versions: *SIROCCO 2024 (short paper), LNCS, pp. 494–500* (doi:10.1007/978-3-031-60603-8_28). Read: arXiv v1 (2024). Cited on p. 37.

Junya Nakamura, Sayaka Kamei, Yukiko Yamauchi. Evacuation from various types of finite two-dimensional square grid fields by a metamorphic robotic system. *Concurrency and Computation: Practice and Experience 35(14):e6628*, 2021. doi:10.1002/cpe.6628, arXiv:2010.05498 (pdf). Other versions: *CANDAR 2020, IEEE, pp. 69–78*, as "Evacuation from a Finite 2D Square Grid Field by a Metamorphic Robotic System" (doi:10.1109/CANDAR51075.2020.00016). First appeared in 2020. Read: arXiv v3 (2021). Cited on p. 509.

Alfredo Navarra, Francesco Piselli. Silent Programmable Matter: Coating. *OPODIS 2023, LIPIcs 286, 25:1*, 2023. doi:10.4230/LIPIcs.OPODIS.2023.25, pdf. Cited on pp. 46, 70, 605, 606, 607, 608, 659, and 707.

Alfredo Navarra, Francesco Piselli. Asynchronous Silent Programmable Matter: Line Formation. *SSS 2023, LNCS, pp. 598–612*, 2023. doi:10.1007/978-3-031-44274-2_44, arXiv:2307.16731 (pdf). Other versions: *DISC 2023 (brief announcement), LIPIcs 281, 45:1–45:8*, as "Brief Announcement: Line Formation in Silent Programmable Matter" (doi:10.4230/LIPIcs.DISC.2023.45, pdf). Read: arXiv v1 (2023). Cited on pp. 45, 46, 47, 52, 69, 70, 77, 600, 603, 604, 659, 672, and 707.

Alfredo Navarra, Francesco Piselli. Oblivious Robots Under Round Robin: Gathering on Rings. *IJTCS-FAW 2025, LNCS, pp. 166–180*, 2025. doi:10.1007/978-981-96-8312-3_13, arXiv:2502.03939 (pdf). Read: arXiv v2 (February 2025). Cited on pp. 141, 143, 144, 147, 148, 149, 665, and 710.

Alfredo Navarra, Francesco Piselli, Giuseppe Prencipe. Line formation and scattering in silent programmable matter. *Journal of Parallel and Distributed Computing 204:105129*, 2025. doi:10.1016/j.jpdc.2025.105129. Other versions: *AINA 2023, LNNS 654, pp. 236–247*, as "Scattering with Programmable Matter" (doi:10.1007/978-3-031-29056-5_22). First appeared in 2023. Cited on pp. 70, 604, 622, 623, 624, 625, 626, 663, and 707.

Andre Naz, Benoit Piranda, Julien Bourgeois, Seth Copen Goldstein. A distributed self-reconfiguration algorithm for cylindrical lattice-based modular robots. *NCA 2016, IEEE, pp. 254–263*, 2016. doi:10.1109/nca.2016.7778628. Cited on p. 637.

Tsukasa Ninomiya, Shota Naito, Naoki Kitamura, Yuichi Sudo, Koichi Wada. A Minimum Exact Disjunctive Basis for Two-Robot Solvability Profiles. *arXiv preprint*, 2026. arXiv:2610.17677 (pdf). Preprint, not peer reviewed. Cited on pp. 31, 32, 33, 34, 35, and 37.

Nooshin Nokhanji, Paola Flocchini, Nicola Santoro. Dynamic Line Maintenance by Hybrid Programmable Matter. *International Journal of Networking and Computing 13(1):18–47*, 2023. doi:10.15803/ijnc.13.1_18. Other versions: *IPDPS Workshops 2022, IEEE, pp. 486–495*, as "Fully Dynamic Line Maintenance by Hybrid Programmable Matter" (doi:10.1109/IPDPSW55747.2022.00087). First appeared in 2022. Cited on p. 632.

Shunhao Oh, Dana Randall, Andréa W. Richa. Brief Announcement: Foraging in Particle Systems via Self-Induced Phase Changes. *DISC 2022 (brief announcement), LIPIcs 246, 51:1–51:3*, 2022. doi:10.4230/LIPIcs.DISC.2022.51, pdf, arXiv:2208.10720 (pdf). Read: arXiv v1 (2022), the full version, titled "Foraging in Particle Systems via Self-Induced Phase Changes". Cited on p. 610.

Shunhao Oh, Joseph L. Briones, Jacob Calvert, Noah Egan, Dana Randall, Andréa W. Richa. Single Bridge Formation in Self-Organizing Particle Systems. *DISC 2024, LIPIcs 319, 34*, 2024. doi:10.4230/LIPIcs.DISC.2024.34, pdf, arXiv:2408.10830 (pdf). Read: arXiv v2 (2024). Cited on pp. 70, 618, and 619.

Takashi Okumura, Koichi Wada, Yoshiaki Katayama. Brief Announcement: Optimal Asynchronous Rendezvous for Mobile Robots with Lights. *SSS 2017 (brief announcement), LNCS, pp. 484–488*, 2017. doi:10.1007/978-3-319-69084-1_36, arXiv:1707.04449 (pdf). Other versions: *Adventures Between Lower Bounds and Higher Altitudes: Essays Dedicated to Juraj Hromkovič, LNCS 11011, pp. 434–448*, 2018, as "Rendezvous of Asynchronous Mobile Robots with Lights" (doi:10.1007/978-3-319-98355-4_25). Read: arXiv v1 (2017). Cited on p. 446.

Takashi Okumura, Koichi Wada, Xavier Défago. Optimal L-algorithms for rendezvous of asynchronous mobile robots with external-lights. *Theoretical Computer Science 979:114198*, 2023. doi:10.1016/j.tcs.2023.114198, arXiv:1804.01368 (pdf). Other versions: *OPODIS 2018, LIPIcs 125, 24:1–24:16*, as "Optimal Rendezvous L-Algorithms for Asynchronous Mobile Robots with External-Lights" (doi:10.4230/LIPIcs.OPODIS.2018.24, pdf). First appeared in 2018. Read: arXiv v1 (2018). Cited on pp. 434, 441, 447, and 702.

Fukuhito Ooshita, Sébastien Tixeuil. On the self-stabilization of mobile oblivious robots in uniform rings. *Theoretical Computer Science 568:84–96*, 2015. doi:10.1016/j.tcs.2014.12.008. Other versions: *SSS 2012, LNCS, pp. 49–63*, as "On the Self-stabilization of Mobile Oblivious Robots in Uniform Rings" (doi:10.1007/978-3-642-33536-5_6). First appeared in 2012. Cited on p. 143.

Fukuhito Ooshita, Sébastien Tixeuil. Ring exploration with myopic luminous robots. *Information and Computation 285:104702*, 2022. doi:10.1016/j.ic.2021.104702, arXiv:1805.03965 (pdf). Other versions: *SSS 2018, LNCS, pp. 301–316* (doi:10.1007/978-3-030-03232-6_20). First appeared in 2018. Read: arXiv v1 (2018). Cited on pp. 166, 174, 175, 176, 177, 178, 179, 180, 181, 189, 194, 195, 196, 197, 198, 199, 200, 201, 661, 702, and 703.

Fukuhito Ooshita, Naoki Kitamura, Ryota Eguchi, Michiko Inoue, Hirotsugu Kakugawa, Sayaka Kamei, Masahiro Shibata, Yuichi Sudo. Crash-tolerant perpetual exploration with myopic luminous robots on rings. *OPODIS 2024 (LIPIcs 324), 12:1–12:16*, 2024. doi:10.4230/LIPIcs.OPODIS.2024.12, pdf. Cited on pp. 177, 180, 195, 196, 197, 198, 661, and 709.

Fukuhito Ooshita, Hirotsugu Kakugawa, Sayaka Kamei, Masahiro Shibata. Crash-tolerant perpetual exploration of myopic luminous robots with fewer colors on rings. *CANDAR Workshops 2025, pp. 166–172*, 2025. doi:10.1109/CANDARW68385.2025.00036. Cited on pp. 177, 194, and 195.

Yaniv Oshrat, Noa Agmon, Sarit Kraus. Adversarial Fence Patrolling: Non-Uniform Policies for Asymmetric Environments. *AAAI 2020, pp. 10377–10384*, 2020. doi:10.1609/aaai.v34i06.6606. Cited on p. 568.

Dorota Osula. Minimizing the Cost of Team Exploration. *SOFSEM 2019, LNCS, pp. 392–405*, 2019. doi:10.1007/978-3-030-10801-4_31, arXiv:1705.10826 (pdf). First appeared in 2017. Cited on p. 247.

Kohei Otaka, Fabian Frei, Koichi Wada. Gathering semi-synchronously scheduled two-state robots. *Theoretical Computer Science 1068:115776*, 2026. doi:10.1016/j.tcs.2026.115776, arXiv:2408.09999 (pdf). Other versions: *SSS 2024, LNCS, pp. 209–224* (doi:10.1007/978-3-031-74498-3_15). First appeared in 2024. Read: arXiv v1 (2024). Cited on pp. 63, 66, 67, 78, 439, 440, 441, 442, 447, 448, 450, 651, and 708.

Deniz Ozsoyeller, Volkan Isler, Andrew Beveridge. Symmetric Rendezvous in Planar Environments With and Without Obstacles. *Proceedings of the AAAI Conference on Artificial Intelligence 26:2046–2052*, 2021. doi:10.1609/aaai.v26i1.8385. First appeared in 2012. Cited on p. 120.

Andreas Padalkin, Christian Scheideler, Daniel Warner. The structural power of reconfigurable circuits in the amoebot model. *Natural Computing 23(4):603–625*, 2024. doi:10.1007/s11047-024-09981-6, pdf, arXiv:2205.02610 (pdf). Other versions: *DNA 28, LIPIcs 238, 8:1–8:22*, 2022 (doi:10.4230/LIPIcs.DNA.28.8, pdf). First appeared in 2022. Read: arXiv v1 (2022); theorem numbers refer to it. Cited on p. 600.

Andreas Padalkin, Manish Kumar, Christian Scheideler. Reconfiguration and locomotion with joint movements in the amoebot model. *Autonomous Robots 49*, 2025. doi:10.1007/s10514-025-10204-9, pdf, arXiv:2305.06146 (pdf). First appeared in 2023. Read: arXiv v3 (2025). Cited on pp. 600 and 636.

Andreas Padalkin, Christian Scheideler. Polylogarithmic time algorithms for shortest path forests in programmable matter. *Distributed Computing 39(2):15*, 2026. doi:10.1007/s00446-026-00505-2, pdf, arXiv:2402.12123 (pdf). Other versions: *PODC 2024, ACM, pp. 65–75* (doi:10.1145/3662158.3662776). First appeared in 2024. Read: arXiv v1 (2024); theorem numbers refer to it. Cited on p. 600.

Linda Pagli, Giuseppe Prencipe, Giovanni Viglietta. Getting close without touching: near-gathering for autonomous mobile robots. *Distributed Computing 28(5):333–349*, 2015. doi:10.1007/s00446-015-0248-5, arXiv:1505.07168 (pdf). Other versions: *SIROCCO 2012*. First appeared in 2012. Read: arXiv v1 (2015). Cited on pp. 23, 63, 460, 461, 462, 463, 470, 682, and 697.

Malo Papin. Deterministic Collision-Free Exploration of Unknown Anonymous Graphs – A Time Efficient Algorithm. *HAL preprint hal-05690949*, 2026. link. Preprint, not peer reviewed. Cited on pp. 9, 88, 232, 234, 235, 240, 666, 697, and 712.

Irene Parada, Vera Sacristán, Rodrigo I. Silveira. A new meta-module design for efficient reconfiguration of modular robots. *Autonomous Robots 45(4):457–472*, 2021. doi:10.1007/s10514-021-09977-6. Other versions: *ICRA 2016, IEEE, pp. 5197–5202*, as "A new meta-module for efficient reconfiguration of hinged-units modular robots" (doi:10.1109/icra.2016.7487726). First appeared in 2016. Cited on p. 637.

Debasish Pattanayak, Kaushik Mondal, Partha Sarathi Mandal, Stefan Schmid. Convergence of Even Simpler Robots without Position Information. *NETYS 2017, LNCS, pp. 69–85*, 2017. doi:10.1007/978-3-319-59647-1_6, arXiv:1608.06002 (pdf). First appeared in 2016. Cited on p. 470.

Debasish Pattanayak, H. Ramesh, Partha Sarathi Mandal, Stefan Schmid. Evacuating Two Robots from Two Unknown Exits on the Perimeter of a Disk with Wireless Communication. *ICDCN 2018 (short paper)*, 2018. doi:10.1145/3154273.3154313, arXiv:1708.03792 (pdf). First appeared in 2017. Read: arXiv v1 (2017). Cited on pp. 73, 507, 511, 517, 518, 542, 653, and 702.

Debasish Pattanayak, H. Ramesh, Partha Sarathi Mandal. Chauffeuring a Crashed Robot from a Disk. *ALGOSENSORS 2019, pp. 177–191*, 2019. doi:10.1007/978-3-030-34405-4_10, arXiv:1906.03024 (pdf). Read: arXiv v1 (2019). Cited on pp. 540, 541, 542, 543, 544, 703, and 704.

Debasish Pattanayak, Kaushik Mondal, H. Ramesh, Partha Sarathi Mandal. Gathering of mobile robots with weak multiplicity detection in presence of crash-faults. *Journal of Parallel and Distributed Computing 123:145–155*, 2019. doi:10.1016/j.jpdc.2018.09.015, arXiv:1608.02432 (pdf). Other versions: *ICDCN 2017*, as "Fault-Tolerant Gathering of Mobile Robots with Weak Multiplicity Detection" (doi:10.1145/3007748.3007786). First appeared in 2016. Read: arXiv v1 (2016). Cited on pp. 24, 25, 51, 74, 432, 435, 451, 452, 453, and 700.

Debasish Pattanayak, H. Ramesh, Partha Sarathi Mandal. Collaborative Evacuation of Mobile Robots.

Proceedings of the 21st International Conference on Distributed Computing and Networking, 1-1, 2020. doi:10.1145/3369740.3373806. Cited on p. 508.

Debasish Pattanayak, Klaus-Tycho Foerster, Partha Sarathi Mandal, Stefan Schmid. Conic Formation in Presence of Faulty Robots. *ALGOSENSORS 2020, LNCS, pp. 170–185*, 2020. doi:10.1007/978-3-030-62401-9_12, arXiv:2003.01914 (pdf). Cited on p. 383.

Debasish Pattanayak, John Augustine, Partha Sarathi Mandal. Randomized gathering of asynchronous mobile robots. *Theoretical Computer Science 858:64–80*, 2021. doi:10.1016/j.tcs.2020.11.048, arXiv:1909.10004 (pdf). First appeared in 2019. Read: arXiv v1 (2019). Cited on p. 433.

Debasish Pattanayak, Gokarna Sharma, Partha Sarathi Mandal. Dispersion of Mobile Robots Tolerating Faults. *ICDCN 2021 Adjunct (WDALFR workshop)*, 2021. doi:10.1145/3427477.3429464. Cited on pp. 279 and 705.

Debasish Pattanayak, Andrzej Pelc. Computing Functions by Teams of Deterministic Finite Automata. *arXiv preprint*, 2023. arXiv:2310.01151 (pdf). Preprint, not peer reviewed. Cited on p. 369.

Debasish Pattanayak, Gokarna Sharma, Partha Sarathi Mandal. Dispersion of Mobile Robots in Spite of Faults. *SSS 2023 (LNCS 14310)*, 2023. doi:10.1007/978-3-031-44274-2_31. Cited on pp. 75, 79, 279, 280, 647, and 707.

Debasish Pattanayak, Subhash Bhagat, Sruti Gan Chaudhuri, Anisur Rahaman Molla. Maximal Independent Set via Mobile Agents. *ICDCN 2024, pp. 74–83*, 2024. doi:10.1145/3631461.3631543. Cited on pp. 61, 343, 345, 346, 347, 348, 350, 351, 353, 356, 361, 365, and 708.

Debasish Pattanayak, Andrzej Pelc. Graph exploration by a deterministic memoryless automaton with pebbles. *Discrete Applied Mathematics 356:149–160*, 2024. doi:10.1016/j.dam.2024.05.024. Cited on pp. 57, 88, 96, 209, 210, and 708.

Debasish Pattanayak, Gokarna Sharma. Time-color tradeoff on uniform circle formation by asynchronous robots. *IPDPS 2024, pp. 987–997*, 2024. doi:10.1109/IPDPS57955.2024.00092. Cited on pp. 67, 399, and 402.

Debasish Pattanayak, Andrzej Pelc. Deterministic treasure hunt and rendezvous in arbitrary connected graphs. *Information Processing Letters 185:106455*, 2024. doi:10.1016/j.ipl.2023.106455, arXiv:2310.01136 (pdf). First appeared in 2023. Read: arXiv v1 (2023). Cited on pp. 58, 109, 110, 112, 121, 130, 131, 650, and 707.

Debasish Pattanayak, Ajay D. Kshemkalyani, Manish Kumar, Anisur Rahaman Molla, Gokarna Sharma. Asynchronous Dispersion with Optimal Time Complexity. *ICPP 2026*, 2026. doi:10.1145/3832810.3832921, arXiv:2507.01298 (pdf). Other versions: *DISC 2025, brief announcement (LIPIcs 356)*, as "Brief Announcement: Optimal Dispersion Under Asynchrony" (doi:10.4230/LIPIcs.DISC.2025.63, pdf); *SPAA 2026 brief announcement*, as "Brief Announcement: Asynchronous Dispersion with Optimal Time Complexity" (doi:10.1145/3816782.3819188). First appeared in 2025. Read: arXiv v1 (2025). Cited on pp. 60, 77, 265, 266, 267, 268, and 709.

Andrzej Pelc. Deterministic gathering with crash faults. *Networks 72(2):182–199*, 2018. doi:10.1002/net.21810, arXiv:1704.08880 (pdf). First appeared in 2017. Cited on pp. 6, 13, 14, and 15.

Andrzej Pelc. Explorable Families of Graphs. *SIROCCO 2018, LNCS, pp. 165–177*, 2018. doi:10.1007/978-3-030-01325-7_17, arXiv:2303.10394 (pdf). Cited on p. 86.

Andrzej Pelc. Use of information, memory and randomization in asynchronous gathering. *Journal of Computer and System Sciences 94:193–205*, 2018. doi:10.1016/j.jcss.2017.10.005, arXiv:1709.05869 (pdf). First appeared in 2017. Cited on p. 133.

Andrzej Pelc, Ram Narayan Yadav. Cost vs. Information Tradeoffs for Treasure Hunt in the Plane. *arXiv preprint*, 2019. arXiv:1902.06090 (pdf). Preprint, not peer reviewed. Cited on p. 109.

Andrzej Pelc. Deterministic rendezvous algorithms. *Distributed Computing by Mobile Entities, LNCS 11340, pp. 423–454*, 2019. doi:10.1007/978-3-030-11072-7_17, arXiv:2303.10391 (pdf). Read: arXiv v1 (2023). Cited on pp. 5, 6, 7, 8, 11, 12, 13, 14, 15, 16, 50, 52, 53, 54, 62, 88, 110, 116, 119, 120, 121, 124, 125, 133, 209, 641, 650, and 693.

Andrzej Pelc, Ram Narayan Yadav. Using Time to Break Symmetry: Universal Deterministic Anonymous Rendezvous. *SPAA 2019, ACM, pp. 85–92*, 2019. doi:10.1145/3323165.3323182, arXiv:1810.03120 (pdf). First appeared in 2018. Cited on pp. 126, 127, 650, and 702.

Andrzej Pelc, Ram Narayan Yadav. Latecomers Help to Meet: Deterministic Anonymous Gathering in the Plane. *ICDCN 2020, pp. 1–10*, 2020. doi:10.1145/3369740.3369767, arXiv:1811.06420 (pdf). First appeared in 2018. Cited on p. 120.

Andrzej Pelc, Ram Narayan Yadav. Advice complexity of treasure hunt in geometric terrains. *Information and*

Computation 281:104705, 2021. doi:10.1016/j.ic.2021.104705, arXiv:1811.06823 (pdf). First appeared in 2018. Read: arXiv v2. Cited on p. 110.

David Peleg. Distributed Coordination Algorithms for Mobile Robot Swarms: New Directions and Challenges. *IWDC 2005, LNCS 3741, pp. 1–12*, 2005. doi:10.1007/11603771_1. Cited on pp. 18, 21, 177, and 179.

Alexandra Porter, Andrea Richa. Collaborative Computation in Self-organizing Particle Systems. *UCNC 2018, LNCS, pp. 188–203*, 2018. doi:10.1007/978-3-319-92435-9_14, arXiv:1710.07866 (pdf). First appeared in 2017. Cited on p. 601.

Pavan Poudel, Aisha Aljohani, Gokarna Sharma. Fault-tolerant complete visibility for asynchronous robots with lights under one-axis agreement. *Theoretical Computer Science 850:116–134*, 2021. doi:10.1016/j.tcs.2020.10.033. Other versions: *WALCOM 2018, LNCS, pp. 169–182* (doi:10.1007/978-3-319-75172-6_15). First appeared in 2018. Cited on pp. 417, 419, 420, 421, 422, 423, 657, and 703.

Pavan Poudel, Gokarna Sharma. Time-Optimal Gathering under Limited Visibility with One-Axis Agreement. *Information 12(11):448*, 2021. doi:10.3390/info12110448. Other versions: *SSS 2017*, as "Universally Optimal Gathering Under Limited Visibility" (doi:10.1007/978-3-319-69084-1_23). First appeared in 2017. Cited on pp. 22, 63, 432, 455, 456, 457, 652, and 701.

Subhajit Pramanick, Sai Vamshi Samala, Debasish Pattanayak, Partha Sarathi Mandal. Distributed algorithms for filling MIS vertices of an arbitrary graph by myopic luminous robots. *Theoretical Computer Science 978:114187*, 2023. doi:10.1016/j.tcs.2023.114187, arXiv:2107.04885 (pdf). Other versions: *ICDCIT 2023, LNCS, pp. 3–19*, as "Filling MIS Vertices of a Graph by Myopic Luminous Robots" (doi:10.1007/978-3-031-24848-1_1). First appeared in 2021. Cited on p. 177.

Subhajit Pramanick, Saswata Jana, Adri Bhattacharya, Partha Sarathi Mandal. Distributed Uniform Partitioning of a Region using Opaque ASYNC Luminous Mobile Robots. *ICDCN 2024, ACM, pp. 55–64*, 2024. doi:10.1145/3631461.3631555, arXiv:2311.04536 (pdf). First appeared in 2023. Read: arXiv v2 (2024). Cited on p. 466.

Subhajit Pramanick, Saswata Jana, Partha Sarathi Mandal, Gokarna Sharma. Asynchronous Gathering of Opaque Robots with Mobility Faults. *arXiv preprint*, 2025. arXiv:2509.10711 (pdf). Preprint, not peer reviewed. Cited on pp. 74 and 452.

Giuseppe Prencipe. Impossibility of gathering by a set of autonomous mobile robots. *Theoretical Computer Science 384(2–3):222–231*, 2007. doi:10.1016/j.tcs.2007.04.023. Other versions: *SIROCCO 2005*, as "On the Feasibility of Gathering by Autonomous Mobile Robots" (doi:10.1007/11429647_20). First appeared in 2005. Read: the SIROCCO 2005 version. Cited on pp. 18, 20, 21, 23, 24, 25, 51, 63, 65, 77, 431, 442, 444, 446, 671, 672, 687, and 694.

V. B. Priezzhev, Deepak Dhar, Abhishek Dhar, Supriya Krishnamurthy. Eulerian Walkers as a Model of Self-Organized Criticality. *Physical Review Letters 77(25):5079–5082*, 1996. doi:10.1103/PhysRevLett.77.5079. Cited on pp. 207 and 219.

Prajyot Pyati, Navjot Kaur, Saswata Jana, Adri Bhattacharya, Partha Sarathi Mandal. Separation of unconscious colored robots with obstructed visibility. *Theoretical Computer Science 1086:116243*, 2026. doi:10.1016/j.tcs.2026.116243, arXiv:2510.22434 (pdf). Other versions: *ICDCIT 2026, pp. 68–84*, as "Separation of Unconscious Robots with Obstructed Visibility" (doi:10.1007/978-3-032-16632-6_5). First appeared in 2025. Cited on pp. 484, 485, 487, and 710.

Ori Rappel, Joseph Ben-Asher, Alfred Bruckstein. Exploration and Coverage with Swarms of Settling Agents. *arXiv preprint*, 2022. arXiv:2209.05512 (pdf). Preprint, not peer reviewed. Cited on p. 261.

Arthur Rauch, Quentin Bramas, Stéphane Devismes, Pascal Lafourcade, Anissa Lamani. Optimal exclusive perpetual grid exploration by luminous myopic robots without common chirality. *NETYS 2021, LNCS, pp. 95–110*, 2021. doi:10.1007/978-3-030-91014-3_7, link. Cited on pp. 176, 180, 187, 188, 191, 192, 193, 194, 204, 661, and 705.

Omer Reingold, Luca Trevisan, Salil Vadhan. Pseudorandom walks on regular digraphs and the RL vs. L problem. *STOC 2006, pp. 457–466*, 2006. doi:10.1145/1132516.1132583. Other versions: *ECCC TR05-022*, 2005 (pdf, link). First appeared in 2005. Read: the ECCC report TR05-022 (2005): abstract and introduction. Cited on p. 87.

Omer Reingold. Undirected connectivity in log-space. *Journal of the ACM 55(4), article 17*, 2008. doi:10.1145/1391289.1391291. Other versions: *STOC 2005*, as "Undirected ST-connectivity in log-space" (doi:10.1145/1060590.1060647). First appeared in 2005. Cited on pp. 56, 57, 85, 87, 88, 97, 98, 108, and 694.

Ronald L. Rivest, Robert E. Schapire. Inference of finite automata using homing sequences. *Information and Computation 103(2):299–347*, 1993. doi:10.1006/inco.1993.1021. Other versions: *STOC 1989, pp. 411–420* (doi:10.1145/73007.73047). First appeared in 1989. Cited on p. 87.

Hans-Anton Rollik. Automaten in planaren Graphen. *Acta Informatica 13(3):287–298*, 1980. doi:10.1007/BF00288647. Other versions: *GI-TCS 1979, LNCS, pp. 266–275* (doi:10.1007/3-540-09118-1_28). First appeared in 1979. Cited on pp. 56, 59, 85, 89, 93, 96, and 119.

Amanpreet Singh Saini, Ashish Saxena, Kaushik Mondal. Cops and Lethal Robber on Rings: A Distributed Perspective. *arXiv preprint*, 2026. arXiv:2609.06388 (pdf). Preprint, not peer reviewed. Cited on p. 319.

Arnaud Sangnier, Nathalie Sznajder, Maria Potop-Butucaru, Sébastien Tixeuil. Parameterized verification of algorithms for oblivious robots on a ring. *Formal Methods in System Design 56(1–3):55–89*, 2020. doi:10.1007/s10703-019-00335-y, link, arXiv:1706.05193 (pdf). Other versions: *2017 Formal Methods in Computer Aided Design (FMCAD) 2017, pp. 212–219* (doi:10.23919/fmcad.2017.8102262); *FMCAD 2017, pp. 212–219*. First appeared in 2017. Cited on pp. 166, 173, and 660.

Madhumita Sardar, Deepanwita Das, Srabani Mukhopadhyaya. Grid exploration by a swarm of autonomous robots with minimum repetitions. *Theoretical Computer Science 933:67–87*, 2022. doi:10.1016/j.tcs.2022.08.029. Cited on p. 166.

Ashish Saxena, Kaushik Mondal. A further study on weak Byzantine gathering of mobile agents. *Theoretical Computer Science 1022:114892*, 2024. doi:10.1016/j.tcs.2024.114892. Other versions: *ICDCN 2024, ACM, pp. 22–31*, as "A Further Study on Weak Byzantine Gathering of Mobile Agents" (doi:10.1145/3631461.3631551). Read: the ICDCN 2024 version (open ACM copy); theorem numbers refer to it. Cited on p. 134.

Ashish Saxena, Kaushik Mondal. Path connected dynamic graphs with a study of dispersion and exploration. *Theoretical Computer Science 1050:115390*, 2025. doi:10.1016/j.tcs.2025.115390, arXiv:2504.08474 (pdf). Other versions: *ICDCN 2025*, as "Path Connected Dynamic Graphs with a Study of Efficient Dispersion" (doi:10.1145/3700838.3700864). Read: arXiv v1 (2025). Cited on pp. 286, 287, 307, and 649.

Ashish Saxena, Tanvir Kaur, Kaushik Mondal. Balanced dispersion on time-varying dynamic graphs. *Journal of Computer and System Sciences 161:103829*, 2026. doi:10.1016/j.jcss.2026.103829, arXiv:2410.04050 (pdf). Other versions: *ICDCN 2025*, as "Dispersion on Time Varying Graphs" (doi:10.1145/3700838.3700845). First appeared in 2024. Read: arXiv v3 (2025). Cited on pp. 8, 13, 286, 287, and 649.

Ashish Saxena, Kaushik Mondal. Exploration on highly dynamic graphs. *Distributed Computing 39 (2026)*, 2026. doi:10.1007/s00446-026-00516-z, arXiv:2601.13047 (pdf). Read: arXiv v1 (2026). Cited on p. 306.

Ashish Saxena, Anisur Rahaman Molla, Kaushik Mondal, Gokarna Sharma. Dynamic Graph Exploration: Semi-Synchrony and Dynamic Port Labeling. *DISC 2026, LIPIcs 397, 48:1–48:23*, 2026. doi:10.4230/LIPIcs.DISC.2026.48, pdf, arXiv:2605.14375 (pdf). Other versions: *arXiv preprint*, 2026, as "Semi-Synchronous Exploration in Dynamic Graphs" (link). Read: arXiv v1 (2026). Cited on p. 307.

Arne Schmidt, Sheryl Manzoor, Li Huang, Aaron T. Becker, Sandor P. Fekete. Efficient Parallel Self-Assembly Under Uniform Control Inputs. *IEEE Robotics and Automation Letters 3(4):3521–3528*, 2018. doi:10.1109/lra.2018.2853758, arXiv:1807.01584 (pdf). Cited on p. 637.

Hirokazu Seike, Yukiko Yamauchi. Separation of Unconscious Colored Robots. *SSS 2023, pp. 328–343*, 2023. doi:10.1007/978-3-031-44274-2_24. Cited on pp. 483, 485, 486, and 707.

Claude E. Shannon. Presentation of a maze-solving machine. *Transactions of the 8th Conference on Cybernetics (Macy Conference), pp. 173–180*, 1951. link. Cited on pp. 50, 56, 83, and 691.

Gokarna Sharma, Costas Busch, Supratik Mukhopadhyay. Bounds on Mutual Visibility Algorithms. *CCCG 2015, pp. 268–274*, 2015. pdf. Cited on p. 414.

Gokarna Sharma, Costas Busch, Supratik Mukhopadhyay. Mutual Visibility with an Optimal Number of Colors. *ALGOSENSORS 2015, pp. 196–210*, 2015. doi:10.1007/978-3-319-28472-9_15. Cited on pp. 67, 413, 416, 417, 418, 419, and 700.

Gokarna Sharma, Ramachandran Vaidyanathan, Jerry L. Trahan, Costas Busch, Suresh Rai. Complete Visibility for Robots with Lights in O(1) Time. *SSS 2016, LNCS 10083, pp. 327–345*, 2016. doi:10.1007/978-3-319-49259-9_26. Cited on pp. 417 and 420.

Gokarna Sharma, Ramachandran Vaidyanathan, Jerry L. Trahan, Costas Busch, Suresh Rai. O(log N)-Time Complete Visibility for Asynchronous Robots with Lights. *IPDPS 2017, pp. 513–522*, 2017. doi:10.1109/IPDPS.2017.51. Cited on pp. 417, 418, and 419.

Gokarna Sharma, Rusul Alsaedi, Costas Busch, Supratik Mukhopadhyay. The Complete Visibility Problem for Fat Robots with Lights. *ICDCN 2018*, 2018. doi:10.1145/3154273.3154319. Cited on pp. 413, 423, 424, and 703.

Gokarna Sharma, Ramachandran Vaidyanathan, Jerry L. Trahan. Constant-Time Complete Visibility for Robots with Lights: The Asynchronous Case. *Algorithms 14(2):56*, 2021. doi:10.3390/a14020056. Other versions: *SSS 2017, LNCS, pp. 265–281*, as "Constant-Time Complete Visibility for Asynchronous Robots with Lights" (doi:10.1007/978-3-319-69084-1_18). First appeared in 2017. Cited on pp. 25, 67, 413, 418, 419, 657, and 702.

Gokarna Sharma, Ramachandran Vaidyanathan, Jerry L. Trahan. Optimal Randomized Complete Visibility on a Grid for Asynchronous Robots with Lights. *International Journal of Networking and Computing 11(1):50–77*, 2021. doi:10.15803/ijnc.11.1_50. Other versions: *IPDPSW 2020, IEEE, pp. 607–616* (doi:10.1109/ipdpsw50202.2020.00103). First appeared in 2020. Cited on p. 414.

Avisek Sharma, Pritam Goswami, Buddhadeb Sau. Computational Power of Mobile Robots: Discrete Version. *ICDCN 2025, pp. 231–238*, 2025. doi:10.1145/3700838.3700870, arXiv:2407.05678 (pdf). First appeared in 2024. Read: arXiv v4 (2024). Cited on p. 36.

Avisek Sharma, Satakshi Ghosh, Buddhadeb Sau. Gathering of Asynchronous Robots on Circle with Limited Visibility Using Finite Communication. *SSS 2025, LNCS, pp. 442–457*, 2025. doi:10.1007/978-3-032-11127-2_33. Cited on p. 433.

Masahiro Shibata, Shinji Kawai, Fukuhito Ooshita, Hirotsugu Kakugawa, Toshimitsu Masuzawa. Partial gathering of mobile agents in asynchronous unidirectional rings. *Theoretical Computer Science 617:1–11*, 2016. doi:10.1016/j.tcs.2015.09.012. First appeared in 2015. Cited on pp. 133 and 134.

Masahiro Shibata, Toshiya Mega, Fukuhito Ooshita, Hirotsugu Kakugawa, Toshimitsu Masuzawa. Uniform deployment of mobile agents in asynchronous rings. *Journal of Parallel and Distributed Computing 119:92–106*, 2018. doi:10.1016/j.jpdc.2018.03.008. Other versions: *PODC 2016, ACM, pp. 415–424*, as "Uniform Deployment of Mobile Agents in Asynchronous Rings" (doi:10.1145/2933057.2933093). First appeared in 2016. Cited on pp. 310, 312, 313, 315, 316, and 700.

Masahiro Shibata, Daisuke Nakamura, Fukuhito Ooshita, Hirotsugu Kakugawa, Toshimitsu Masuzawa. Partial Gathering of Mobile Agents in Arbitrary Networks. *IEICE Transactions on Information and Systems E102.D(3):444–453*, 2019. doi:10.1587/transinf.2018fcp0008. Cited on p. 133.

Masahiro Shibata, Norikazu Kawata, Yuichi Sudo, Fukuhito Ooshita, Hirotsugu Kakugawa, Toshimitsu Masuzawa. Move-optimal partial gathering of mobile agents without identifiers or global knowledge in asynchronous unidirectional rings. *Theoretical Computer Science 822:92–109*, 2020. doi:10.1016/j.tcs.2020.04.002. Other versions: *SIROCCO 2019, LNCS, pp. 277–292*, as "Partial Gathering of Mobile Agents Without Identifiers or Global Knowledge in Asynchronous Unidirectional Rings" (doi:10.1007/978-3-030-24922-9_19). First appeared in 2019. Cited on p. 134.

Masahiro Shibata, Yuichi Sudo, Junya Nakamura, Yonghwan Kim. Almost uniform deployment of mobile agents in dynamic rings. *Information and Computation 289:104949*, 2022. doi:10.1016/j.ic.2022.104949. Other versions: *SSS 2020, LNCS, pp. 248–263*, as "Uniform Deployment of Mobile Agents in Dynamic Rings" (doi:10.1007/978-3-030-64348-5_20). First appeared in 2020. Cited on pp. 261, 310, 311, 312, 313, 314, 315, 316, 649, 700, and 704.

Masahiro Shibata, Masaki Ohyabu, Yuichi Sudo, Junya Nakamura, Yonghwan Kim, Yoshiaki Katayama. Visibility-optimal gathering of seven autonomous mobile robots on triangular grids. *International Journal of Networking and Computing 12(1):2–25*, 2022. doi:10.15803/ijnc.12.1_2, arXiv:2103.08172 (pdf). Other versions: *IPDPSW 2021, IEEE, pp. 566–575*, as "Gathering of seven autonomous mobile robots on triangular grids" (doi:10.1109/ipdpsw52791.2021.00090). First appeared in 2021. Cited on p. 142.

Masahiro Shibata, Sébastien Tixeuil. Semi-uniform deployment of mobile robots in perfect ℓ-ary trees. *Concurrency and Computation: Practice and Experience 35:e7432*, 2023. doi:10.1002/cpe.7432. Other versions: *CANDAR 2021, IEEE, pp. 168–174*, as "Semi-Uniform Deployment of Mobile Robots in Perfect ℓ-ary Trees" (doi:10.1109/candar53791.2021.00031). First appeared in 2021. Cited on p. 313.

Masahiro Shibata, Naoki Kitamura, Ryota Eguchi, Yuichi Sudo, Junya Nakamura, Yonghwan Kim, Yoshiaki Katayama, Toshimitsu Masuzawa, Quentin Bramas, Sébastien Tixeuil. Uniform Deployment of Mobile Robots in Complete Bipartite Graphs. *OPODIS 2025, LIPIcs 361, 34:1–34:17*, 2025. doi:10.4230/LIPIcs.OPODIS.2025.34, pdf. Cited on p. 313.

Masahiro Shibata, Yuichi Sudo, Junya Nakamura, Yonghwan Kim. Partial gathering of mobile agents in dynamic rings. *Theoretical Computer Science 1063:115684*, 2026. doi:10.1016/j.tcs.2025.115684, arXiv:2212.03457 (pdf). First appeared in 2022. Cited on p. 134.

Kohei Shimoyama, Yuichi Sudo, Hirotsugu Kakugawa, Toshimitsu Masuzawa. One Bit Agent Memory is Enough for Snap-Stabilizing Perpetual Exploration of Cactus Graphs with Distinguishable Cycles. *SSS 2022 (invited paper), LNCS 13751, pp. 19–34*, 2022. doi:10.1007/978-3-031-21017-4_2. Cited on p. 208.

Takahiro Shintaku, Yuichi Sudo, Hirotsugu Kakugawa, Toshimitsu Masuzawa. Efficient Dispersion of Mobile Agents without Global Knowledge. *SSS 2020 (LNCS 12514)*, 2020. doi:10.1007/978-3-030-64348-5_22, arXiv:2008.09379 (pdf). Read: arXiv v1 (2020). Cited on p. 267.

Prakhar Shukla, Animesh Maiti, Shivam Kumar, Subhash Bhagat. Budget-Constrained Fault-Tolerant Mutual Visibility for Autonomous Robots under the Mobility Fault Model. *arXiv preprint*, 2026. arXiv:2610.04402 (pdf). Preprint, not peer reviewed. Cited on pp. 420, 421, 657, and 711.

Prakhar Shukla, Seshunadh Tanuj Peddinti, Subhash Bhagat. Gathering Autonomous Mobile Robots Under the Adversarial Defected View Model. *arXiv preprint*, 2026. arXiv:2603.05788 (pdf). Preprint, not peer reviewed. Cited on p. 432.

Arijit Sil, Sruti Gan Chaudhuri. Line Formation by Fat Robots under Limited Visibility. *arXiv preprint*, 2017. arXiv:1710.09398 (pdf). Preprint, not peer reviewed. Cited on p. 382.

Mor Sinay, Noa Agmon, Oleg Maksimov, Sarit Kraus, David Peleg. Maintaining Communication in Multi-Robot Tree Coverage. *IJCAI 2017, pp. 4515–4522*, 2017. doi:10.24963/ijcai.2017/630. Cited on p. 247.

Andrey Viktorovich Stepkin, Alina Sergeevna Stepkina. Algorithm of simple graph exploration by a collective of agents. *Computer Research and Modeling 13(1):33–45*, 2021. doi:10.20537/2076-7633-2021-13-1-33-45. Cited on p. 158.

A. V. Stopkin. Algorithm for exploration of a simple undirected graph by a collective of agents. *Scientific Notes of Taurida National V. I. Vernadsky University, Series: Technical Sciences (2024, no. 5), pp. 303–309*, 2024. doi:10.32782/2663-5941/2024.5.1/43. Cited on p. 158.

A. V. Stopkin. Exploration of simple graphs by a collective of agents. *Mathematical Modeling and Computing 12(2):603–610*, 2025. doi:10.23939/mmc2025.02.603. Cited on p. 158.

A. V. Stopkin. Finite graph exploration by a mobile agent. *Mathematical Modeling and Computing 12(1):75–82*, 2025. doi:10.23939/mmc2025.01.075. Cited on p. 86.

A. V. Stopkin. Simple graphs exploration by a collective of agents. *Scientific Notes of Taurida National V. I. Vernadsky University, Series: Technical Sciences 36(75), no. 1:232–238*, 2025. doi:10.32782/2663-5941/2025.1.2/34. Cited on p. 158.

A. V. Stopkin. Finite Undirected Graphs Exploration by a Mobile Agent. *Mathematical Modeling and Computing 13(1):225–231*, 2026. doi:10.23939/mmc2026.01.225. Cited on p. 86.

Yuichi Sudo, Daisuke Baba, Junya Nakamura, Fukuhito Ooshita, Hirotsugu Kakugawa, Toshimitsu Masuzawa. A Single Agent Exploration in Unknown Undirected Graphs with Whiteboards. *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences E98.A(10):2117–2128*, 2015. doi:10.1587/transfun.E98.A.2117. Other versions: *WRAS 2010 (workshop at PODC 2010)*, as "An agent exploration in unknown undirected graphs with whiteboards" (doi:10.1145/1953563.1953570). First appeared in 2010. Cited on p. 208.

Yuichi Sudo, Masahiro Shibata, Junya Nakamura, Yonghwan Kim, Toshimitsu Masuzawa. Near-Linear Time Dispersion of Mobile Agents. *Distributed Computing 39(1):5*, 2026. doi:10.1007/s00446-025-00494-8, pdf, arXiv:2310.04376 (pdf). Other versions: *DISC 2024 (LIPIcs 319)* (doi:10.4230/LIPIcs.DISC.2024.38, pdf). First appeared in 2023. Read: arXiv v3 (2024). Cited on pp. 16, 57, 60, 259, 266, 267, 268, 342, 344, 345, and 645.

Yuichi Sudo, Fukuhito Ooshita, Sayaka Kamei. Self-stabilizing graph exploration by a single agent. *Theoretical Computer Science 1082:116085*, 2026. doi:10.1016/j.tcs.2026.116085, arXiv:2010.08929 (pdf). Other versions: *SIROCCO 2025, LNCS, pp. 384–400* (doi:10.1007/978-3-031-91736-3_23); *DISC 2024 (brief announcement), LIPIcs 319, 55:1–55:7*, as "Brief Announcement: Self-Stabilizing Graph Exploration by a Single Agent" (doi:10.4230/LIPIcs.DISC.2024.55, pdf). First appeared in 2020. Cited on p. 208.

Yuichi Sudo, Naoki Kitamura, Masahiro Shibata, Junya Nakamura, Sébastien Tixeuil, Toshimitsu Masuzawa, Koichi Wada. Online Graph Exploration under T-Interval Connectivity. *arXiv preprint*, 2026. arXiv:2604.04619 (pdf). Read: arXiv v1 (2026), titled "Tight Bounds on Window Size and Time for Single-Agent Graph Exploration under T-Interval Connectivity". Preprint, not peer reviewed. Cited on pp. 106, 107, 307, 644, and 711.

Xiaoming Sun, Yuan Sun, Jialin Zhang. Better Upper Bounds for Searching on a Line with Byzantine Robots. *Complexity and Approximation, LNCS 12000, pp. 151–171*, 2020. doi:10.1007/978-3-030-41672-0_9. Cited on p. 503.

Ichiro Suzuki, Masafumi Yamashita. Distributed anonymous mobile robots: formation of geometric patterns. *SIAM Journal on Computing 28(4):1347–1363*, 1999. doi:10.1137/S009753979628292X. Cited on pp. 17, 18, 19, 20, 21, 22, 26, 27, 34, 51, 53, 63, 65, 77, 379, 380, 381, 382, 398, 429, 431, 432, 434, 443, 444, 445, 454, 467, 468, 469, and 692.

Amnon Ta-Shma, Uri Zwick. Deterministic rendezvous, treasure hunts, and strongly universal exploration sequences. *ACM Transactions on Algorithms 10(3), article 12*, 2014. doi:10.1145/2601068. Other versions: *SODA 2007*. First appeared in 2007. Cited on pp. 52, 53, 57, 62, 107, 108, 109, 110, 113, 118, 121, 124, 125, 137, 138, 641, 649, and 695.

Shota Takahashi, Haruki Kanaya, Shoma Hiraoka, Ryota Eguchi, Yuichi Sudo. Recolorable Graph Exploration by an Oblivious Agent with Fewer Colors. *OPODIS 2025 (LIPIcs 361), 32:1–32:15*, 2025. doi:10.4230/LIPIcs.OPODIS.2025.32, pdf, arXiv:2505.02789 (pdf). Cited on pp. 207, 208, 209, 210, 211, 212, 214, 215, 217, 219, 662, 685, and 710.

Satoshi Terai, Koichi Wada, Yoshiaki Katayama. Gathering problems for autonomous mobile robots with lights. *Theoretical Computer Science 941:241–261*, 2023. doi:10.1016/j.tcs.2022.11.018, arXiv:1811.12068 (pdf). First appeared in 2018. Read: arXiv v1 (2018). Cited on pp. 21, 24, 66, 440, 442, 443, 446, 450, 651, 686, and 703.

Tommaso Toffoli, Norman Margolus. Programmable matter: Concepts and realization. *Physica D: Nonlinear Phenomena 47(1–2):263–272*, 1991. doi:10.1016/0167-2789(91)90296-L. Other versions: *International Journal of High Speed Computing 05(02):155–170*, 1993 (doi:10.1142/s0129053393000086). Cited on p. 42.

Yusaku Tomita, Yukiko Yamauchi, Shuji Kijima, Masafumi Yamashita. Plane Formation by Synchronous Mobile Robots without Chirality. *OPODIS 2017, LIPIcs 95, 13:1–13:17*, 2017. doi:10.4230/LIPIcs.OPODIS.2017.13, arXiv:1705.06521 (pdf). Cited on p. 383.

Masashi Tsuchida, Fukuhito Ooshita, Michiko Inoue. Byzantine-Tolerant Gathering of Mobile Agents in Arbitrary Networks with Authenticated Whiteboards. *IEICE Transactions on Information and Systems E101.D(3):602–610*, 2018. doi:10.1587/transinf.2017fcp0008. Other versions: *IEICE Transactions on Information and Systems E103.D(7):1672–1682*, 2020, as "Byzantine-Tolerant Gathering of Mobile Agents in Asynchronous Arbitrary Networks with Authenticated Whiteboards" (doi:10.1587/transinf.2019edp7311). Cited on p. 133.

Ramachandran Vaidyanathan, Costas Busch, Jerry L. Trahan, Gokarna Sharma, Suresh Rai. Logarithmic-Time Complete Visibility for Robots with Lights. *IPDPS 2015, pp. 375–384*, 2015. doi:10.1109/IPDPS.2015.52. Cited on pp. 25, 413, 420, 657, and 700.

Giovanni Viglietta. Rendezvous of Two Robots with Visible Bits. *ALGOSENSORS 2013, pp. 291–306*, 2013. doi:10.1007/978-3-642-45346-5_21, arXiv:1211.6039 (pdf). First appeared in 2012. Read: arXiv v2 (2012). Cited on pp. 18, 66, 78, 434, 446, and 697.

Qi WEI, Xuehou TAN, Bo JIANG. Competitive Strategies for Evacuating from an Unknown Affected Area. *IEICE Transactions on Information and Systems E99.D(10):2585–2590*, 2016. doi:10.1587/transinf.2016edp7118. Cited on p. 507.

Koichi Wada, Toshimitsu Masuzawa, Yuichi Sudo. Rendezvous under Variable Disorientation: The Algorithmic Power of Fixed Unit Distance. *arXiv preprint*, 2026. arXiv:2610.09713 (pdf). Preprint, not peer reviewed. Cited on pp. 23, 29, 36, 37, 66, 67, 439, 440, 447, 448, 450, 642, 651, 652, 674, and 711.

Ahmed Mouhamadou Wade. Exploration with return of highly dynamic networks. *International Journal of Advanced Research 9(10):315–319*, 2021. doi:10.21474/ijar01/13550. Cited on p. 87.

Daniel Warner, Christian Scheideler. Brief Announcement: Fault-Tolerant 3D Leader Election in the Amoebot Model. *SAND 2026 (brief announcement), LIPIcs 373, 25:1–25:6*, 2026. doi:10.4230/LIPIcs.SAND.2026.25, pdf, link. Cited on pp. 586, 592, 658, and 711.

Daniel Warner, Christian Scheideler. Brief Announcement: Leader Election with Noisy Reconfigurable Circuits in the Amoebot Model. *SAND 2026 (brief announcement), LIPIcs 373, 26:1–26:6*, 2026. doi:10.4230/LIPIcs.SAND.2026.26, pdf, link. Cited on pp. 43, 46, 586, and 595.

Jamison W. Weber, Tishya Chhabra, Andréa W. Richa, Joshua J. Daymude. Energy-Constrained Programmable Matter Under Unfair Adversaries. *OPODIS 2023, LIPIcs 286, 7:1–7:21*, 2023. doi:10.4230/LIPIcs.OPODIS.2023.7, pdf, arXiv:2309.04898 (pdf). Read: arXiv v2 (2023). Cited on pp. 628 and 629.

Qi Wei, Xuehou Tan, Bo Jiang, Lijuan Wang. On-Line Strategies for Evacuating from a Convex Region in the Plane. *Combinatorial Optimization and Applications, pp. 74-85*, 2014. doi:10.1007/978-3-319-12691-3_7. Cited on pp. 508 and 509.

Qi Wei, Xuehou Tan, Bo Jiang, Lijuan Wang. Evacuating from an Unknown Affected Area. *International Journal of Pattern Recognition and Artificial Intelligence 30, pp. 1659022*, 2016. doi:10.1142/s0218001416590229. Cited on p. 508.

Qin Xin. Faster treasure hunt and better strongly universal exploration sequences. *ISAAC 2007, pp. 549–560*, 2007. doi:10.1007/978-3-540-77120-3_48, arXiv:1204.5442 (pdf). Cited on pp. 57, 113, and 125.

Claude (Anthropic) was used in cataloguing the problems.

Qin Xin. Faster Treasure Hunt via Strongly Universal Exploration with Polylogarithmic Overhead. *arXiv preprint*, 2026. arXiv:2611.08705 (pdf). Read: arXiv v1 (2026). Preprint, not peer reviewed. Cited on pp. 58, 113, 114, 649, and 711.

Yinfeng Xu, Lan Qin. Strategies of Groups Evacuation from a Convex Region in the Plane. *Frontiers in Algorithmics and Algorithmic Aspects in Information and Management, pp. 250-260*, 2013. doi:10.1007/978-3-642-38756-2_26. Cited on pp. 508 and 509.

Ryonosuke Yamada, Tomoyuki Usami, Yukiko Yamauchi. Rendezvous and merging for two metamorphic robotic systems without global compass. *Theoretical Computer Science 1075:115906*, 2026. doi:10.1016/j.tcs.2026.115906, arXiv:2407.19175 (pdf). First appeared in 2024. Read: arXiv v1 (2024). Cited on p. 119.

Yukiko Yamauchi, Masafumi Yamashita. Randomized Pattern Formation Algorithm for Asynchronous Oblivious Mobile Robots. *DISC 2014, LNCS, pp. 137–151*, 2014. doi:10.1007/978-3-662-45174-8_10. Cited on pp. 26, 387, 388, 389, 651, and 699.

Vladimir Yanovski, Israel A. Wagner, Alfred M. Bruckstein. A distributed ant algorithm for efficiently patrolling a network. *Algorithmica 37(3):165–186*, 2003. doi:10.1007/s00453-003-1030-9. Cited on pp. 207, 209, 210, 211, 212, 219, and 693.

Erik van den Akker, Kevin Buchin, Klaus-Tycho Foerster. Multi-Agent Online Graph Exploration on Cycles and Tadpole Graphs. *SIROCCO 2024, LNCS, pp. 513–519*, 2024. doi:10.1007/978-3-031-60603-8_31, arXiv:2402.13845 (pdf).

Cited on p. 225.