Stand-Up Indulgent Gathering on Lines

Quentin Bramas, Sayaka Kamei, Anissa Lamani, Sébastien Tixeuil

Stand-Up Indulgent Gathering on Lines

Quentin Bramas[1], Sayaka Kamei[2], Anissa Lamani[1], and Sébastien Tixeuil[3]

[1] University of Strasbourg, ICube, CNRS, France.
[2] Graduate School of Advanced Science and Engineering, Hiroshima University, Japan.
[3] Sorbonne University, CNRS, LIP6, IUF, France.

Abstract. We consider a variant of the crash-fault gathering problem called stand-up indulgent gathering (SUIG). In this problem, a group of mobile robots must eventually gather at a single location, which is not known in advance. If no robots crash, they must all meet at the same location. However, if one or more robots crash at a single location, all non-crashed robots must eventually gather at that location. The SUIG problem was first introduced for robots operating in a two-dimensional continuous Euclidean space, with most solutions relying on the ability of robots to move a prescribed (real) distance at each time instant.
In this paper, we investigate the SUIG problem for robots operating in a discrete universe (i.e., a graph) where they can only move one unit of distance (i.e., to an adjacent node) at each time instant. Specifically, we focus on line-shaped networks and characterize the solvability of the SUIG problem for oblivious robots without multiplicity detection.

Keywords: Crash failure, fault-tolerance, LCM robot model

1 Introduction

1.1 Context and Motivation

Mobile robotic swarms recently received a considerable amount of attention from the Distributed Computing scientific community. Characterizing the exact hypotheses that enable solving basic problems for robots represented as disoriented (each robot has its own coordinate system) oblivious (robots cannot remember past action) dimensionless points evolving in a Euclidean space has been at the core of the researchers' goals for more than two decades. One of the key such hypotheses is the scheduling assumption [14]: robots can execute their protocol fully synchronized (FSYNC), in a completely asynchronous manner (ASYNC), of having repeatedly a fairly chosen subset of robots scheduled for synchronous execution (SSYNC).

Among the many studied problems, the *gathering* [22] plays a benchmarking role, as its simplicity to express (robots have to gather in finite time at the exact same location, not known beforehand) somewhat contradicts its computational tractability (two robots evolving assuming SSYNC scheduling cannot gather, without additional hypotheses).

As the number of robots grows, the probability that at least one of them fails increases, yet, relatively few works consider the possibility of robot failures. One of the simplest such failures is the *crash* fault, where a robot unpredictably stops executing its protocol. In the case of gathering, one should prescribe the expected behavior in the presence of crash failures. Two variants have been studied: *weak* gathering expects all correct (that is, non-crashed) robots to gather, regardless of the positions of the crashed robots, while *strong* gathering

(also known as stand-up indulgent gathering – SUIG) expects correct robots to gather at the (supposedly unique) crash location. In continuous Euclidean space, weak gathering is solvable in the SSYNC model [1,3,6,10], while SUIG (and its variant with two robots, stand up indulgent rendezvous – SUIR) is only solvable in the FSYNC model [4,5].

A recent trend [14] has been to move from the continuous environment setting to a discrete one. More precisely, in the discrete setting, robots can occupy a finite number of locations, and move from one location to another if they are neighboring. This neighborhood relation is conveniently represented by a graph whose nodes are locations, leading to the "robots on graphs" denomination. This discrete setting is better suited for describing constrained physical environments, or environments where robot positioning is only available from discrete sensors [2]. From a computational perspective, the continuous setting and the discrete setting are unrelated: on the one hand, the number of possible configurations (that, the number of robot positions) is much more constrained in the discrete setting than in the continuous setting (only a finite number of configurations exists in the discrete setting), on the other hand, the continuous setting offers algorithms designers more flexibility to solve problematic configurations (e.g., using arbitrarily small movements to break a symmetry).

In this paper, we consider the discrete setting, and aim to characterize the solvability of the SUIR and SUIG problems: in a set of locations whose neighborhood relation is represented by a line-shaped graph, robots have to gather at one single location, not known beforehand; furthermore, if one or more robots crash anytime at the same location, all robots must gather at this location.

1.2 Related Works

In graphs, in the absence of faults, mobile robot gathering was primarily considered for ring-shaped graphs [20,19,16,17,15,12,11]. For other topologies, gathering problem was considered, e.g., in finite grids [13], trees [13], tori [18], complete cliques [9], and complete bipartite graphs [9]. Most related to our problem is the (relaxed) FSYNC gathering algorithm presented by Castenow et al. [8] for grid-shaped networks where a single robot may be stationary. The main differences with our settings are as follows. First, if no robot is stationary, their [8] robots end up in a square of 2×2 rather than a single node as we require. Second, when one robot is stationary (and thus never moves), all other robots gather at the stationary robot location, *assuming a stationary robot can be detected as such when on the same node* (instead, we consider that a crashed robot cannot be detected), and assuming a stationary robot never moves from the beginning of the execution (while we consider anytime crashes). Third, they assume that initial positions are neighboring (while we characterize which patterns of initial positions are solvable).

In the continuous setting, the possibility of a robot failure was previously considered. As previously stated, the weak-gathering problem in SSYNC [1,3,6,10], and the SUIR and SUIG problems in FSYNC [4,5] were previously considered. In particular, solutions to SUIR and SUIG [4,5] make use of a level-slicing technique, that mandates them to move by a fraction of the distance to another robot. Obviously, such a technique cannot be translated to the discrete model, where robots always move by exactly one edge.

Works combining the discrete setting and the possibility of robot failures are scarce. Ooshita and Tixeuil [21] considered transient robot faults placing them at arbitrary locations, and presented a probabilistic self-stabilizing gathering algorithm in rings, assuming SSYNC, and that robots are able to exactly count how many of them occupy a particular location. Castaneda et al. [7] presented a weaker version of gathering, named edge-gathering. They provided a solution to edge-gathering in acyclic graphs, assuming that any number of robots may crash. On the one hand, their scheduling model is the most general (ASYNC); on the other hand, their robot model makes use of persistent memory (robots can remember some of their past actions, and communicate explicitly with other robots).

Overall, to our knowledge, the SUIR and SUIG problems were never addressed in the discrete setting.

1.3 Our Contribution

In this paper, we initiate the research on SUIG and SUIR feasibility for robots on line-shaped graphs, considering the vanilla model (called OBLOT [14]) where robots are oblivious (that is, they don't have access to persistent memory between activations), are *not* able to distinguish multiple occupations of a given location, and can be completely disoriented (no common direction). More precisely, we focus on both of finite/infinite lines, and study conditions that preclude or enable SUIG and SUIR solvability. As in the continuous model, we first prove that SUIG and SUIR are impossible to solve in the SSYNC model, so we concentrate on the FSYNC model. It turns out that, in FSYNC, SUIR is solvable if and only if the initial distance between the two robots is even, and that SUIG is solvable if only if the initial configuration is not edge-symmetric. Our positive results are constructive, as we provide an algorithm for each case and prove it correct. As expected, the key enabling algorithmic constructions we use for our protocols are fundamentally different from those used in continuous spaces [4,5], as robots can no longer use fractional moves to solve the problem, and can be of independent interest to build further solutions in other topologies.

The rest of the paper is organized as follows. Section 2 presents our model assumptions, Section 3 is dedicated to SUIR, and Section 4 is dedicated to SUIG. We provide concluding remarks in Section 5. Due to space constraints, additional proofs and results are presented in the appendix.

2 Model

The line consists of an infinite or finite number of nodes $u_0, u_1, u_2, \ldots$, such that a node u_i is connected to both $u_{(i-1)}$ and $u_{(i+1)}$ (if they exist). Note that in the case where the line is finite of size n, two nodes of the line, u_0 and u_{n-1} are only connected to u_1 and u_{n-2}, respectively.

Let $R = \{r_1, r_2, \ldots, r_k\}$ be the set of $k \geq 2$ autonomous robots. Robots are assumed to be anonymous (i.e., they are indistinguishable), uniform (i.e., they all execute the same program, and use no localized parameter such as a particular orientation), oblivious (i.e., they cannot remember their past actions), and disoriented (i.e., they cannot distinguish left and right).

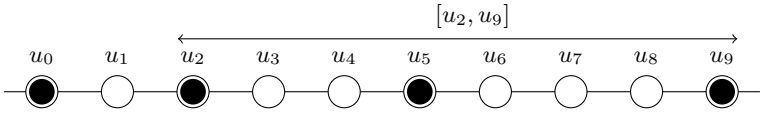

Fig. 1. Instance of a configuration in which the segment $[u_2, u_9]$ is highlighted.

We assume that robots do not know the number of robots k. In addition, they are unable to communicate directly, however, they have the ability to sense the environment including the positions of all other robots, i.e., they have infinite view. Based on the snapshot resulting of the sensing, they decide whether to move or to stay idle. Each robot r executes cycles infinitely many times, *(i)* first, r takes a snapshot of the environment to see the positions of the other robots (LOOK phase), *(ii)* according to the snapshot, r decides whether it should move and where (COMPUTE phase), and *(iii)* if r decides to move, it moves to one of its neighbor nodes depending on the choice made in COMPUTE phase (MOVE phase). We call such cycles LCM (LOOK-COMPUTE-MOVE) cycles. We consider the *FSYNC* model in which at each time instant t, called round, each robot r executes an LCM cycle synchronously with all the other robots, and the *SSYNC* model where a non-empty subset of robots chosen by an adversarial scheduler executes an LCM cycle synchronously, at each t.

A node is considered *occupied* if it contains at least one robot; otherwise, it is *empty*. If a node u contains more than one robot, it is said to have a *tower* or *multiplicity*. The ability to detect towers is called *multiplicity detection*, which can be either *global* (any robot can sense a tower on any node) or *local* (a robot can only sense a tower if it is part of it). If robots can determine the number of robots in a sensed tower, they are said to have *strong* multiplicity detection. In this work, we assume that robots do not have multiplicity detection and cannot distinguish between nodes with one robot and those with multiple robots.

As robots move and occupy nodes, their positions form the system's *configuration* $C_t = (d(u_0), d(u_1), \dots)$ at time t. Here, $d(u_i) = 0$ if node u_i is empty and $d(u_i) = 1$ if it is occupied.

Given two nodes u_i and u_j, a segment $[u_i, u_j]$ represents the set of nodes between u_i and u_j, inclusive. No assumptions are made about the state of the nodes in $[u_i, u_j]$. Any node $u \in [u_i, u_j]$ can be either empty or occupied. The number of occupied nodes in $[u_i, u_j]$ is represented by $|[u_i, u_j]|$. In Fig. 1, each node u_i, where $i \in \{2, 3, \dots, 9\}$, is part of the segment $[u_2, u_9]$. Note that $|[u_2, u_9]| = 3$ since nodes u_2, u_5, and u_9 are occupied.

For a given configuration C, let u_i be an occupied node. Node u_j is considered an *occupied neighboring node* of u_i in C if it is occupied and if $|[u_i, u_j]| = 2$. A *border node* in C is an occupied node with only one occupied neighboring node. A robot on a border node is referred to as a *border robot*. In Fig. 1, when $k = 4$, nodes u_0 and u_9 are border nodes.

The *distance* between two nodes u_i and u_j is the number of edges between them. The distance between two robots r_i and r_j is the distance between the two nodes occupied by r_i and r_j, respectively. We denote the distance between u_i and u_j (resp. r_i and r_j) $dist(u_i, u_j)$ (resp. $dist(r_i, r_j)$). Two robots or two nodes are *neighbors* (or adjacent) if the distance between them is one. A sequence of consecutive occupied nodes is a *block*. Similarly, a sequence of consecutive empty nodes is a *hole*.

An *algorithm* A is a function mapping the snapshot (obtained during the LOOK phase) to a neighbor node destination to move to (during the MOVE phase). An *execution* $\mathcal{E} = (C_0, C_1, \dots)$ of A is a sequence of configurations, where C_0 is an initial configuration, and every configuration C_{t+1} is obtained from C_t by applying A.

Let r be a robot located on node u_i at time t and let $S^+(t) = d(u_i), d(u_{i+1}), \dots, d(u_{i+m})$ and $S^-(t) = d(u_i), d(u_{i-1}), \dots d(u_{i-m'})$ be two sequences such that $m, m' \in \mathbb{N}$. Note that $m = n - i - 1$ and $m' = i$ in the case where the line is finite of size n, $m = m' = \infty$ in the case where the line is infinite. The view of robot r at time t, denoted $View_r(t)$, is defined as the pair $\{S^+(t), S^-(t)\}$ ordered in the lexicographic order.

Let C be a configuration at time t. Configuration C is said to be *symmetric* at time t if there exist two robots r and r' such that $View_r(t) = View_{r'}(t)$. In this case, r and r' are said to be symmetric robots. Let C be a symmetric configuration at time t then, C is said to be *node-symmetric* if the distance between two symmetric robots is even (i.e., if the axis of symmetry intersects with the line on a node), otherwise, C is said to be *edge-symmetric*. Finally, a non-symmetric configuration is called a *rigid* configuration.

Problem definition. A robot is said to be *crashed* at time t if it is not activated at any time $t' \geq t$. That is, a crashed robot stops execution and remains at the same position indefinitely. We assume that robots cannot identify a crashed robot in their snapshots (i.e., they are able to see the crashed robots but remain unaware of their crashed status). A crash, if any, can occur at any round of the execution. Furthermore, if more than one crash occurs, all crashes occur at the same location. In our model, since robots do not have multiplicity detection capability, a location with a single crashed robot and with multiple crashed robots are indistinguishable, and are thus equivalent. In the sequel, for simplicity, we consider at most one crashed robot.

We consider the *Stand Up Indulgent Gathering* (SUIG) problem defined in [5]. An algorithm solves the SUIG problem if, for any initial configuration C_0 (that may contain multiplicities), and for any execution $\mathcal{E} = (C_0, C_1, \dots)$, there exists a round t such that all robots (including the crashed robot, if any) gather at a single node, not known beforehand, for all $t' \geq t$. The special case with $k = 2$ is called the *Stand Up Indulgent Rendezvous* (SUIR) problem.

3 Stand Up Indulgent Rendezvous

We address in this section the case in which $k = 2$, that is, the SUIR problem. We show that SSYNC solutions do not exist when one seeks a deterministic solution (Corollary 1), and that even in FSYNC, not all initial configurations admit a solution (Theorem 1). By contrast, all other initial configurations admit a deterministic SUIR solution (Theorem 2).

Theorem 1. *Starting from a configuration where the two robots are at odd distance from each other on a line-shaped graph, the SUIR problem is unsolvable in FSYNC by deterministic oblivious robots without additional hypotheses.*

Proof. Let us first observe that in any configuration, both robots must move. Indeed, if no robot moves, then no robot will ever move, and SUIR is never achieved. If one robot only

moves, then the adversary can crash this robot, and the two robots never move, hence SUIR is never achieved.

In any configuration, two robots can either: *(i)* move both in the same direction, *(ii)* move both toward each other, or *(iii)* move both in the opposite direction. Assuming an FSYNC scheduling and no crash by any robot, in the case of *(i)*, the distance between the two robots does not change, in the case of *(ii)*, the distance decreases by two, and in the case of *(iii)*, it increases by two. Since the distance between the two robots is initially odd, then any FSYNC execution of a protocol step keeps the distance between robots odd. As a result, the distance never equals zero, and the robots never gather. □

Corollary 1. *The SUIR problem is unsolvable on a line in SSYNC without additional hypotheses.*

Proof. For the purpose of contradiction, suppose that there exists a SUIR algorithm A in SSYNC. Consider an SSYNC schedule starting from a configuration where exactly one robot is activated in each round. Since any robot advances by exactly one edge per round, and that A solves SUIR, every such execution reaches a configuration where the two robots are at distance $2i+1$, where i is an integer, that is, a configuration where robots are at odd distance from one another. From this configuration onward, the schedule becomes synchronous (as a synchronous schedule is still allowed in SSYNC). By Theorem 1, rendezvous is not achieved, a contradiction. □

By Theorem 1, we investigate the case of initial configurations where the distance between the two robots is even. It turns out that, in this case, the SUIR problem can be solved.

Theorem 2. *Starting from a configuration where the two robots are at even distance from each other, the SUIR problem is solvable in FSYNC by deterministic oblivious robots without additional hypotheses.*

Proof. Our proof is constructive. Consider the simple algorithm "go to the other robot position."

When no robot crashes, at each FSYNC round, the distance between the two robots decreases by two. Since it is initially even, it eventually reaches zero, and the robots stop moving (the other robot is on the same location), hence rendezvous is achieved.

If one robot crashes, in the following FSYNC rounds, the distance between the two robots decreases by one, until it reaches zero. Then, the correct robot stops moving (the crashed robot is on the same location), hence rendezvous is achieved. □

Observe that both Theorems 1 and 2 are valid regardless of the finiteness of the line network.

4 Stand Up Indulgent Gathering

We address the SUIG problem ($k \geq 2$) on line-shaped networks in the following. Section 4.1 first derives some impossibility results, and then Section 4.2 presents our algorithm; finally, the proof of our algorithm appears in Section 4.3.

4.1 Impossibility Results

Theorem 3 ([20]). *The gathering problem is unsolvable in FSYNC on line networks starting from an edge-symmetric configuration even with strong multiplicity detection.*

Corollary 2. *The SUIG problem is unsolvable in FSYNC on line networks starting from an edge-symmetric configuration even with strong multiplicity detection.*

Proof. Consider a FSYNC execution without crashes, and apply Theorem 3. □

As a result of Corollary 2, we suppose in the remaining of the section that initial configurations are *not* edge-symmetric.

Lemma 1. *Even starting from a configuration that is not edge-symmetric, the SUIG problem is unsolvable in SSYNC without additional hypotheses.*

Proof. The proof is by induction on the number X of occupied nodes. Suppose for the purpose of contradiction that there exists such an algorithm A.

If $X = 2$, and if the distance between the two occupied nodes is 1, consider an FSYNC schedule. All robots have the same view, so either all robots stay on their locations (and the configuration remains the same), or all robots go to the other location (and the configuration remains with $X = 2$ and a distance of 1 between the two locations). As this repeats forever, in both cases, the SUIG is not achieved, a contradiction. If $X = 2$, and if the distance between the two occupied nodes is at least 2, consider a schedule that either *(i)* executes all robots on the first location, or *(ii)* executes all robots at the second location, at every round. So, the system behaves (as robots do not use additional hypotheses such as multiplicity detection) as two robots, initially on distinct locations separated by distance at least 2. As a result, the distance between the two occupied nodes eventually becomes odd. Then, consider an FSYNC schedule, and by Theorem 1, A cannot be a solution, a contradiction.

Suppose now that for some integer X, the lemma holds. Let us show that it also holds for $X+1$. Consider an execution starting from a configuration with $X+1$ occupied nodes. Since A is a SUIG solution, any execution of A eventually creates at least one multiplicity point by having a robot r_1 on one occupied node moved to an adjacent occupied node. Consider the configuration C_b that is immediately before the creation of the multiplicity point. Then, in C_b, *all* robots at the location of r_1 make a move (this is possible in SSYNC), so the resulting configuration has X occupied nodes. By the induction hypothesis, algorithm A cannot solve SUIG from this point, a contradiction.

So, for all possible initial configurations, algorithm A cannot solve SUIG, a contradiction. □

As a result of Lemma 1, we suppose in the sequel that the scheduling is FSYNC.

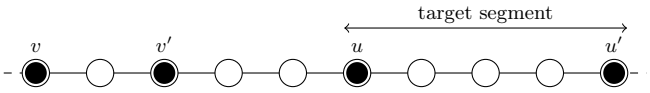

Fig. 2. Instance of a configuration in which $[u, u']$ is the target segment. Nodes v and v' are both in $O_{[u,u']}$.

4.2 Algorithm $\mathcal{A}_L$

In the following, we propose an algorithm $\mathcal{A}_L$ in FSYNC such that the initial configuration is not edge-symmetric.

Before describing our strategy, we first provide some definitions that will be used throughout this section. In given configuration C, let d_C be the largest even distance between any pair of occupied nodes and let U_C be the set of occupied nodes at distance d_C from another occupied node. If C is node-symmetric, U_C consists only of the two border nodes ($|U_C| = 2$). Then, let u and u' be nodes in U_C. By contrast, if C is rigid, then $|U_C| \geq 2$ (refer to Lemma 2). Since each robot has a unique view in a rigid configuration, let $u \in U_C$ be the node that hosts the robots with the largest view among those on a node of U_C and let u' be the occupied node at distance d_C from u. Observe that if there are two candidate nodes for u', using the view of the robots again, we can uniquely elect one of the two which are at distance d_C from u (by taking the one with the largest view). That is, u and u' can be identified uniquely in C. We refer to $[u, u']$ by the target segment in C, and denote the number of occupied nodes in $[u, u']$ by $|[u, u']|$. Finally, the set of occupied nodes which are not in $[u, u']$ is denoted by $O_{[u,u']}$. Refer to Fig. 2 for an example.

We first observe that in any configuration C in which there are at least three occupied nodes, there exist at least two occupied nodes at an even distance.

Lemma 2. *In any configuration C where there are at least three occupied nodes, there exists at least one pair of robots at an even distance from each other.*

Proof. Assume by contradiction that the lemma does not hold and let u_1, u_2, and u_3 be three distinct occupied nodes such that u_2 is located between u_1 and u_3. Assume w.l.o.g. that $d_1 = dist(u_1, u_2)$ and $d_2 = dist(u_2, u_3)$. By the assumption, both d_1 and d_2 are odd. That is, there exist $q, q' \in \mathbb{N}$, $d_1 = 2q + 1$ and $d_2 = 2q' + 1$. Hence, $d_1 + d_2$ is even since $d_1 + d_2 = 2(q + q' + 1)$. A contradiction. □

We propose, in the following, an algorithm named $\mathcal{A}_L$ that solves the SUIG problem on line-shaped networks. The main idea of the proposed strategy is to squeeze the robots by reducing the distance between the two border robots such that they eventually meet on a single node. To guarantee this meeting, robots aim to reach a configuration in which the border robots are at an even distance. Note that an edge-symmetric configuration can be reached during this process. However, in this case, we guarantee that not only one robot has crashed but also, eventually, when there are only two occupied adjacent nodes, the crashed robot is alone on its node ensuring the gathering. Let C be the current configuration. In the following, we describe robots' behavior depending on C:

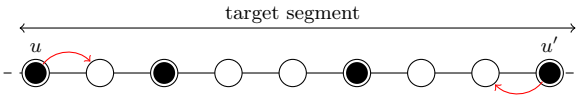

Fig. 3. Instance of a configuration in which $|O_{[u,u']}| = 0$. Robots at the border move toward an occupied node as shown by the red arrows.

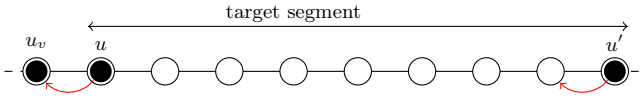

Fig. 4. Instance of a configuration of the special case in which $|O_{[u,u']}| = 1$ with only three occupied nodes. Robots on u and u' move to their adjacent node toward the node in $O_{[u,u']}$, node u_v, as shown by the red arrows.

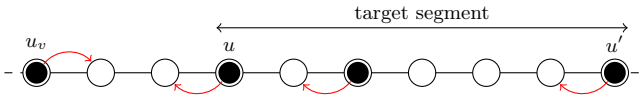

Fig. 5. Instance of a configuration in which $|O_{[u,u']}| = 1$. Robots in $[u, u']$ move to their adjacent node toward the robot in $O_{[u,u']}$ while the robots in $O_{[u,u']}$ move toward the target segment as shown by the red arrows.

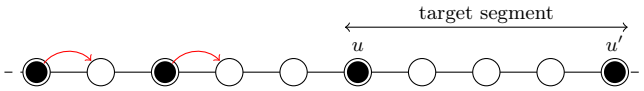

Fig. 6. Instance of a configuration in which $|O_{[u,u']}| > 1$. Robots in $O_{[u,u']}$ move to their adjacent node toward the target segment as shown by the red arrows.

1. C is edge-symmetric. The border robots move toward an occupied node.
2. Otherwise. Let $[u, u']$ be the target segment in C, and let $O_{[u,u']}$ be the set of robots located on a node, not part of $[u, u']$. Robots behave differently depending on the size of $O_{[u,u']}$:
   (a) $|O_{[u,u']}| = 0$ (u and u' host the border robots). In this case, the border robots are the ones to move. Their destination is their adjacent node toward an occupied node (refer to Fig. 3 for an example).
   (b) $|O_{[u,u']}| = 1$. Let u_v be the unique occupied node in $O_{[u,u']}$. Assume w.l.o.g. that u_v is closer to u than to u'. We address only the case in which $dist(u, u_v)$ is odd. (Note that if $dist(u, u_v)$ is even, the border nodes are at an even distance and $|O_{[u,u']}| = 0$.)
      – If the number of occupied nodes is three and u_v and u are two adjacent nodes, robots on both u and u' are the ones to move. Their destination is their adjacent node toward u_v (refer to Fig. 4).
      – Otherwise, all robots are ordered to move. More precisely, robots on a node of $[u, u']$ move to an adjacent node toward u_v while the robots on u_v move to an adjacent node toward u (refer to Fig. 5).
   (c) $|O_{[u,u']}| > 1$. In this case, the robots that are in $O_{[u,u']}$ move toward a node of $[u, u']$ (refer to Fig. 6).

4.3 Proof of the Correctness

We prove in the following the correctness of $\mathcal{A}_L$.

Lemma 3. *Let C be a non-edge-symmetric configuration, and let $[u, u']$ be the target segment. If $|O_{[u,u']}| > 1$, then all nodes in $O_{[u,u']}$ are located on the same side of $[u, u']$.*

Proof. Assume by contradiction that the lemma does not hold and assume that there exists a pair of nodes $u_1, u_2 \in O_{[u,u']}$ such that u_1 and u_2 are on different sides of $[u, u']$. Assume w.l.o.g. that u_1 is the closest to u and u_2 is the closest to u'. Let $dist(u_1, u) = d_1$ and $dist(u_2, u') = d_2$. If d_1 and d_2 are both even or both odd, $dist(u_1, u_2)$ is even. Since $dist(u_1, u_2) > dist(u, u')$, $[u, u']$ is not the target segment, a contradiction. Otherwise, assume w.l.o.g. that d_1 is even and d_2 is odd, then $dist(u_1, u')$ is even. Since $dist(u_1, u') > dist(u, u')$, $[u, u']$ is not the target segment, a contradiction. □

Lemma 4. *Starting from a non-edge-symmetric configuration C, if no robot crashes, all robots gather without multiplicity detection in $O(\mathcal{D})$ by executing $\mathcal{A}_L$, where $\mathcal{D}$ denotes the distance between the two borders in C.*

Proof. Assume by contradiction that the theorem does not hold. Let C_t be the current configuration. The following two cases are possible:

1. $\mathcal{D}$ is even. Let $u_1, u_2, \ldots, u_m$ be the sequence of nodes between the two border robots such that u_1 and u_m host a border robot. Note that the target segment is, in this case, $[u_1, u_m]$ and $|O_{[u_1,u_m]}| = 0$. By A_L, robots on u_1 and u_m are the ones to move toward respectively u_2 and u_{m-1}. By moving, the distance between the border robots decreases by two and hence remains even. By A_L, robots on u_2 and u_{m-1} are now the new borders and are the ones to move to, respectively, u_3 and u_{m-2}. As previously, the distance between the border robots decreases by two. By repeating the same process, after $\lfloor \frac{\mathcal{D}-1}{2} \rfloor$ rounds, the border robots become at distance two from each other on respectively nodes $u_{\frac{\mathcal{D}}{2}}$ and $u_{m-(\frac{\mathcal{D}}{2}+1)}$. After one additional round, all robots meet on $u_{(\frac{\mathcal{D}}{2})+1}$. Hence, the robots gather after $\frac{\mathcal{D}}{2}$ rounds. A contradiction.
2. $\mathcal{D}$ is odd. By Lemma 2, we know that there is at least one pair of robots which are at an even distance from each other. Moreover, C_t is not symmetric as we retrieve case 1 otherwise. Let u and u' be the two occupied nodes with respect to $\mathcal{A}_L$ such that $[u, u']$ is the target segment. Assume w.l.o.g. that v, the border robot in $O_{[u,u']}$, is closer to u. Let $\mathcal{D}' = dist(v, u)$. Two cases are possible:
   (a) $|O_{[u,u']}| > 1$. Then, all nodes in $O_{[u,u']}$ are on the same side of $[u, u']$ by Lemma 3. By the assumption, they are closer to u. By A_L, robots on the nodes in $O_{[u,u']}$ in C_t move toward the nodes of $[u, u']$. That is, after one round, v becomes at an even distance from u', and we retrieve case 1. Hence, we can deduce that the gathering is achieved after $\frac{\mathcal{D}-1}{2} + 1$ rounds. A contradiction.
   (b) $|O_{[u,u']}| = 1$. By the assumption and w.l.o.g., $dist(v, u) < dist(v, u')$. Let $u_{v_0}, u_{v_1}, u_{v_2}, \ldots u_{v_x}, \ldots, u_{v'_x}$ be the sequence of nodes from u_{v_0} toward u such that $u_{v_0} = v$, $u_{v_x} = u$ and $u_{v'_x} = u'$ (refer to Fig. 7 for an example). Let $\mathcal{D}' = dist(u_{v_0}, u_{v_x})$ (in Fig. 7 - configuration C_{t_0}, $u_{v_x} = u_{v_5}$ and $\mathcal{D}' = 5$). By $\mathcal{A}_L$, two cases are possible:

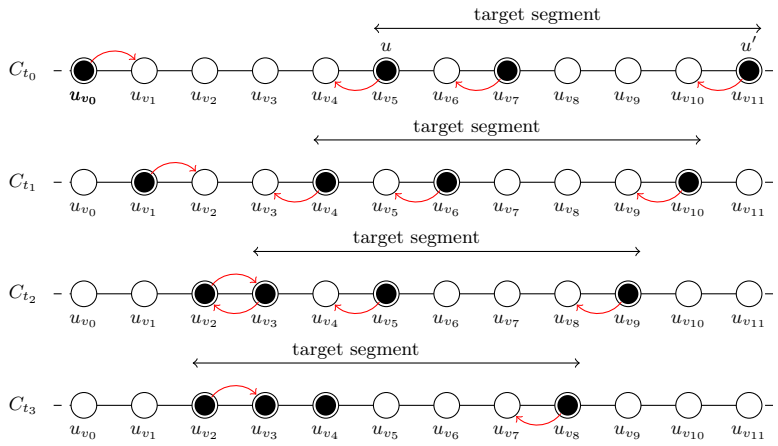

Fig. 7. Instance of an execution starting from a configuration in which $|O_{[u,u']}| = 1$ and assuming no crashes.

i. The number of occupied nodes is equal to three and u_{v_0} is adjacent to u_{v_x} then, the robots on $[u, u']$ ($[u_{v_x}, u_{v'_x}]$) move toward u_{v_0}. As there is no crashed robot, after one round, the border robots become at an even distance, and we retrieve case 1. Thus, the gathering is achieved in $\frac{\mathcal{D}-1}{2} + 1$ rounds.

ii. Otherwise, all robots on a node in $[u, u']$ move toward u_{v_0} while the robots on node u_{v_0} move toward u. Let C_{t_1} be the configuration reached once the robots move. Since there is no crashed robot by assumption, all the robots move and the distance between every robot on a node in $[u, u']$ and those on u_v decreases by two. Hence, $[u_{v_{x-1}}, u_{v_{x'-1}}]$ become the target segment in C_{t_1} and $dist(u, u') = dist(u_{v_{x-1}}, u_{v_{x'-1}})$ in C_{t1}. Moreover, $dist(u_{v_0}, u)$ in C_t is equal to $dist(u_{v_1}, u_{v_{x-1}}) - 2$ in C_{t_1}. By A_L, the robots on a node in $[u_{v_{x-1}}, u_{v_{x'-1}}]$ move toward u_{v_1} while the robots on u_{v_1} move toward $u_{v_{x-1}}$. In the configuration reached C_{t_2}, $[u_{v_{x-2}}, u_{v_{x'-2}}]$ is the target segment. By repeating the same process, after $\lfloor \frac{\mathcal{D}'}{2} \rfloor$ rounds, in configuration $C_{t_{\lfloor \frac{x}{2} \rfloor}}$, the robots initially on u in C_t become located on node $u_{v_{\lceil \frac{x}{2} \rceil}}$, the ones on u_{v_0} on $u_{v_{\lfloor \frac{x}{2} \rfloor}}$ and the ones on u' on node $u_{v_{x'-\lfloor \frac{x}{2} \rfloor}}$ (refer to Fig. 7 for an example). Again, as there is no crashed robot, $dist(u, u')$ in C_t is equal to $dist(u_{v_{\lceil \frac{x}{2} \rceil}}, u_{v_{x'-\lfloor \frac{x}{2} \rfloor}})$ and $[u_{v_{\lceil \frac{x}{2} \rceil}}, u_{v_{x'-\lfloor \frac{x}{2} \rfloor}}]$ is the target segment in $C_{t_{\lfloor \frac{x}{2} \rfloor}}$. After one additional round, we retrieve case 1. We can deduce that the gathering is achieved after $\frac{\mathcal{D}-1}{2}$ rounds, $\frac{\mathcal{D}'-1}{2}$ rounds are needed for the borders to become at an even distance and then $\frac{\mathcal{D}-\mathcal{D}'}{2}$ rounds for the robots to gather.

From the cases above, we can deduce that robots gather in $O(\mathcal{D})$ rounds. □

We focus in the following on the case in which a single robot crashes.

Lemma 5. *Starting from a configuration C where there are only two occupied nodes at distance $\mathcal{D} > 1$, one hosting the crashed robot, gathering is achieved in $\mathcal{D}$ rounds, where $\mathcal{D}$ denotes the distance between the two borders in C.*

Proof. First observe that if the crashed robot is not collocated with a non-crashed robot, after one round the robots on the other border move towards the crashed robot location, and gathering is achieved after $\mathcal{D}$ rounds.

Now assume that the crashed robot is collocated with at least one non-crashed robot. If $\mathcal{D} = 2$, then after one round, all non-crashed robots are located at the same node, adjacent to the crashed robot location, and after one more round, gathering is achieved. If $\mathcal{D} > 3$ is even, then after one round the crashed robot is alone, and the non-crashed robots form two multiplicity points and are the extremities of the target segment. So, after one more round, they both move towards the crashed robot location, and we reach a configuration with two occupied nodes, and the distance between them has decreased by two. By induction and the previous case, gathering is eventually achieved. If $\mathcal{D} = 3$, then similarly, one can show that in three rounds, gathering is achieved. If $\mathcal{D} > 3$ is odd, then similarly, one can show that after three rounds we reach configuration with two occupied nodes, and their distance has decreased by 3 (so, the distance is now even and we can apply one of the previous even cases). □

Lemma 6. *Let C be a configuration where $[u, u']$ is the target segment with $O_{[u,u']} = \{v\}$, and w.l.o.g. u' is a border. Let $\mathcal{D}$ be the distance between the two border nodes u' and v. If u' hosts a crashed robot and $dist(v, u) = 1$, then gathering is achieved in $O(\mathcal{D})$ rounds.*

Proof. Thanks to Lemma 5, it is enough to prove that, in $O(\mathcal{D})$ rounds, a configuration C' in which there are only two occupied nodes v and u' with $dist(v, u') > 1$ is reached.

By $\mathcal{A}_L$, if the number of occupied nodes in C is more than three, the robots in $[u, u']$ move toward v while the robots on v move toward a node toward $[u, u']$. That is, if there is a multiplicity on u', all non-crashed robots move to their adjacent node toward u. Thus, after one round, u' hosts only a crashed robot. As u and v are neighbors, the robots on these nodes simply exchange their positions. However, note that $[u, u']$ remains the target segment and the distance between the robots on u and nodes $v'' \in [u, u']$ with $v'' \neq u$ and $v'' \neq u'$ decreases. In the reached configuration, the same robots are ordered to move. Hence, eventually, u becomes adjacent to two occupied nodes. After one round, $|[u, u']|$ decreases. As u and v remain occupied and adjacent to each other, the robots in $[u, u']$ continue to move toward u to eventually join it. Thus, by repeating this process, $|[u, u']|$ decreases until $|[u, u']| = 2$. As the initial distance between the robots is less than $\mathcal{D}$, after at most $\mathcal{D}$ rounds, a configuration in which there are three occupied nodes u, u' and v, is reached. In this special case, by $\mathcal{A}_L$, after one round, v and u' are the only occupied nodes. Observe that since $[u, u']$ is a target segment, $dist(u, u') \geq 2$ and hence $dist(v, u') \geq 3$. After that, by $\mathcal{A}_L$, robots on v are the only ones to move. Hence the lemma holds. □

Lemma 7. *Let C be a configuration where $[u, u']$ is the target segment, $|O_{[u,u']}| = 1$ and w.l.o.g. $dist(u, v) < dist(u', v)$ where $v \in O_{[u,u']}$. Let $\mathcal{D}$ be the distance between the two border nodes u' and v in C. If u' hosts a crashed robot, then gathering is achieved in $O(\mathcal{D})$ rounds.*

Proof. Recall that, since $v \in O_{[u,u']}$, $dist(u', v)$ is odd in C. If the distance m between u and v is 1, the we apply Lemma 6, otherwise, by $\mathcal{A}_L$, the robots in v move to an empty node toward u, and all the other robots move towards v. Thus, after one round, we reach a configuration C_1 where the robots on v becomes at an even distance from u', the crashed robot location. Since the robots on u are also ordered to move toward v, the distance between the robots at v and u decreases by two.

Again, as u' hosts a crashed robot, after one more round, a configuration C_2 in which the borders are at an odd distance is reached again, and the distance between the robots at u and v is again decreased by two (or stay the same if they are adjacent in C_2 as they just swap their positions).

As the distance m between v and u is odd in C (otherwise, the border robots are at an even distance in C), we can repeat the same 2-round process ($\lceil m/4 \rceil$ times) until we reach a configuration in which the robots at u and v are at distance 1 so, by Lemma 6, gathering is achieved. □

Lemma 8. *Starting from a configuration C where the crashed robot is at a border, and the border robots are at an even distance $\mathcal{D}$, by executing $\mathcal{A}_L$, after $O(\mathcal{D})$ rounds, gathering is achieved.*

Proof. The proof is by induction on $\mathcal{D}$. As the borders are at an even distance, by $\mathcal{A}_L$, these robots are the ones to move. However, as the crashed robot does not move, after one round, the distance between the border robots $\mathcal{D}_1$ becomes odd ($\mathcal{D}_1 \geq 3$). Observe that $\mathcal{D}_1 = \mathcal{D} - 1$.

So if $\mathcal{D} = 2$, in the reached configuration C'', all the non-crashed robots are located at a node adjacent to the crashed robot location, and after one more round they all move towards the crashed robot location and the gathering is achieved.

If $\mathcal{D} > 2$, in the reached configuration C'', the following cases are possible:

1. The configuration C'' is edge-symmetric. By $\mathcal{A}_L$, the border robots are the ones to move and their destination is their adjacent node toward an occupied node. Hence, at the next round, all robots located at one border move to their adjacent node toward an occupied node. In the configuration, the distance between the two border robots $\mathcal{D}_2$ is even and $\mathcal{D}_2 = \mathcal{D} - 2$.
2. Otherwise. By Lemma 2, there exists a pair of occupied nodes that are at an even distance. As C'' is neither node-symmetric (otherwise, the borders are at even distance) nor edge-symmetric (otherwise, case 1 holds), two occupied nodes u and u' are uniquely identified such that $[u, u']$ is the target segment. Let u_x be the node that hosts the crashed robot, two cases are possible:
   (a) $u_x \in \{u, u'\}$. Assume w.l.o.g. that $u_x = u'$. Let us first consider the case in which $|O_{[u,u']}| = 1$ and let $v \in O_{[u,u']}$. If $dist(u, v) = 1$, we are done by Lemma 6. By contrast, if $dist(u, v) > 1$ then, let $u, u_1, \ldots u_m, v$ be the sequence of nodes from u to v. By Lemma 7, after one round, a configuration C'''
   Finally, if $|O_{[u,u']}| > 1$, after one round, a configuration in which the borders are at an even distance $\mathcal{D}_2 = \mathcal{D} - 2$ is reached and we apply the induction hypothesis.

(b) $u_x \in O_{[u,u']}$. Assume w.l.o.g. that $dist(u, u_x) < dist(u', u_x)$ holds. Let u_1 be the closest occupied node to u_x, and u_0 be an adjacent node on the side of u_1. Consider the case when $|O_{[u,u']}| > 1$ holds.
- If u_x hosts a multiplicity, then after one round, the non-crashed robots move to u_0 and a configuration in which $[u', u_0]$ is the target segment is reached with $|O_{[u',u_0]}| = 1$.
- If u_x hosts only a single robot (the crashed one), as only robots on u_1 are ordered to move toward u, after one round, a configuration in which $[u', u_1]$ is the target segment is reached with $|O_{[u',u_1]}| = 1$.

Hence, in both scenarios, we retrieve the case in which $|O_{[u,u']}| = 1$. Let us now focus on the case in which $|O_{[u,u']}| = 1$. By $\mathcal{A}_L$, all robots move toward u_x if $u_0 \neq u_1$. Hence, u_0 becomes occupied eventually. After one round, the distance between two border robots $\mathcal{D}_2$ become even, $\mathcal{D}_2 = \mathcal{D} - 2$, and we can use the induction hypothesis.

□

Lemma 9. *Starting from a non-edge-symmetric configuration C with one crashed robot, all robots executing $\mathcal{A}_L$ eventually gather without multiplicity detection in $O(\mathcal{D})$ rounds, where $\mathcal{D}$ denotes the distance between the two borders in C.*

Proof. First, let us consider the case where $\mathcal{D}$ is even.

1. If the crashed robot is at an equal distance from both borders, by $\mathcal{A}_L$, the border robots move toward each other. As they do, the distance between them remains even. Hence, the border robots remain the only ones to move. Eventually, all robots which are not co-located with the crashed robot become border robots and hence move. Thus, the gathering is achieved in $\frac{\mathcal{D}}{2}$ rounds.
2. If the crashed robot is a border, by Lemmas 8, we can deduce that the gathering is achieved in $O(\mathcal{D})$ rounds.
3. Otherwise, as the border robots move toward each other by $\mathcal{A}_L$, the crashed robot eventually becomes at the border. We hence retrieve case 2.

From the cases above, we can deduce that the gathering is achieved whenever a configuration in which the borders are at an even distance, is reached.

Let us now focus on the case where $\mathcal{D}$ is odd. By $\mathcal{A}_L$, two occupied nodes u and u' at the largest even distance are uniquely identified to set the target segment $[u, u']$ (recall that C is, in this case, rigid, and each robot has a unique view since the initial configuration cannot be edge-symmetric). The robots behave differently depending on the size of $O_{[u,u']}$, the set of occupied nodes outside the segment $[u, u']$. By Lemma 3, all nodes in $O_{[u,u']}$ are on the same side. Assume w.l.o.g. that for all $u_i \in O_{[u,u']}$, u_i is closer to u than u'. Two cases are possible:

1. $|O_{[u,u']}| > 1$. Let $u_f, u_{f'} \in O_{[u,u']}$ be the two farthest nodes from u such that $dist(u, u_f) > dist(u, u_{f'})$. Note that u_f is a border robot. Let u_{f+1} be u_f's adjacent node toward u'_f. If u_f does not host a crashed robot, then as the robots on u_f move toward u and those on u' remain idle by $\mathcal{A}_L$, after one round, the border robots become at an even distance

and we are done. By contrast, if u_f hosts a crashed robot, then either u_f hosts other non-crashed robots, and hence after one round, we retrieve a configuration C' in which $[u_{f+1}, u']$ is the target segment and $|O_{[u_{f+1},u']}| = 1$ or a configuration C' in which $[u_{f'}, u']$ is the target segment and $|O_{[u_{f'},u']}| = 1$ as robots on $u_{f'}$ also move toward u by $\mathcal{A}_L$. In both cases, we retrieve the following case.

2. $|O_{[u,u']}| = 1$. Let u_f be in $O_{[u,u']}$ and assume w.l.o.g. that $dist(u, u_f) < dist(u', u_f)$ (observe that u_f is a border node). If u_f hosts the crashed robot, then, after one round, the border robots are at an even distance as robots on u' move toward u_f by $\mathcal{A}_L$. Similarly, if u' hosts the crashed robot, then by Lemmas 6 and 7 after $O(\mathcal{D})$ rounds, a configuration in which the border robots are at an even distance is reached. If neither u_f nor u' hosts the crashed robot, then when the border robots move by $\mathcal{A}_L$ (other robots also move, but we focus for now on the border robots), either the distance between the two borders becomes even after one round (in the case where u_f is adjacent to u as the robots simply exchange their respective positions) or the distance between the border robots remains odd but decreases by two. Observe that in the later case, the border robots keep moving toward each other by $\mathcal{A}_L$ until one of them becomes a neighbor to a crash robot.
   Let C' be the configuration reached once a border robot becomes adjacent to a crashed robot. Let u_b be the border node that is adjacent to crashed robot, and let $u_{\overline{b}}$ be the other border node. We refer to the node that hosts the crashed robot by u_c. Since $dist(u_b, u_c) = 1$ and $dist(u_b, u_{\overline{b}})$ is odd, $dist(u_{\overline{b}}, u_c)$ is even. Hence, $|O_{[u_{\overline{b}},u_c]}| = 1$. Two cases are possible:
   - If C' hosts only three occupied nodes, then, after one round, the distance between the border robots becomes even, and we are done (recall that robots on $u_{\overline{b}}$ and u_c move toward u_b by $\mathcal{A}_L$).
   - If there are more than 3 occupied nodes and u_c hosts also non-crashed robots in C', then after one round, the distance between the border robots becomes even as the robots on $u_{\overline{b}}$ move toward u_c, those on u_b move to u_c and the non-crashed robots on u_c move toward u_b by $\mathcal{A}_L$. Hence, we are done.
   - Otherwise, after one round, the distance between the two borders remains odd as both borders move toward each other by $\mathcal{A}_L$. In the configuration reached C'', u_c becomes a new border occupied by a crashed robot and non-crashed robots. If there are only two occupied nodes in C'', by $\mathcal{A}_L$, the border robots move toward each other. That is, in the next round, the border robots become at an even distance, and we are done. If there are more than two occupied nodes in C'', by Lemmas 2 and 3, a configuration with a new target segment $[o, o']$ which includes one border node is reached. Let us first consider the case in which $u_c \in [o, o']$. If $|O_{[o,o']}| > 1$, then after one round, the border robots become at an even distance, and we are done. By contrast, if $|O_{[o,o']}| = 1$, then we are done by Lemmas 6 and 7. Next, let us consider the case where $u_c \notin [o, o']$. Let o_f be the closest occupied node of u_c. Without loss of generality, $dist(o, u_c) < dist(o', u_c)$ holds. If $|O_{[o,o']}| > 1$, then after one round, a configuration in which $[o', o_f]$ is the target segment and $u_c \in O_{[o',o_f]}$ is reached. After one additional round, we are done. Finally, if $|O_{[o,o']}| = 1$, then robots on the nodes in $[o, o']$ move toward u_c, and we are done.

From the cases above, we can deduce that the theorem holds. □

From Lemmas 4 and 9, we can deduce:

Theorem 4. *Starting from a non-edge-symmetric configuration C, algorithm $\mathcal{A}_L$ solves the SUIG problem on line-shaped networks without multiplicity detection in $O(\mathcal{D})$ rounds, where $\mathcal{D}$ denotes the distance between the two borders in C.*

5 Concluding Remarks

We initiated the research about stand-up indulgent rendezvous and gathering by oblivious mobile robots in the discrete model, studying the case of line-shaped networks. For both rendezvous and gathering cases, we characterized the initial configurations from which the problem is impossible to solve. In the case of rendezvous, a very simple algorithm solves all cases left open. In the case of gathering, we provide an algorithm that works when the starting configuration is not edge-symmetric. Our algorithms operate in the vanilla model without any additional hypotheses, and are asymptotically optimal with respect to the number of rounds to achieve rendezvous or gathering.

A number of open questions are raised by our work:

1. Is it possible to circumvent impossibility results in SSYNC using extra hypotheses (e.g., multiplicity detection)?
2. Is it possible to solve SUIR and SUIG in other topologies?

References

1. Agmon, N., Peleg, D.: Fault-tolerant gathering algorithms for autonomous mobile robots. SIAM Journal on Computing **36**(1), 56–82 (2006)
2. Balabonski, T., Courtieu, P., Pelle, R., Rieg, L., Tixeuil, S., Urbain, X.: Continuous vs. discrete asynchronous moves: A certified approach for mobile robots. In: Atig, M.F., Schwarzmann, A.A. (eds.) Networked Systems - 7th International Conference, NETYS 2019, Marrakech, Morocco, June 19-21, 2019, Revised Selected Papers. Lecture Notes in Computer Science, vol. 11704, pp. 93–109. Springer (2019). https://doi.org/10.1007/978-3-030-31277-0_7
3. Bouzid, Z., Das, S., Tixeuil, S.: Gathering of mobile robots tolerating multiple crash faults. In: IEEE 33rd International Conference on Distributed Computing Systems (ICDCS). pp. 337–346 (2013)
4. Bramas, Q., Lamani, A., Tixeuil, S.: Stand up indulgent rendezvous. In: Stabilization, Safety, and Security of Distributed Systems. SSS 2020 (2020)
5. Bramas, Q., Lamani, A., Tixeuil, S.: Stand Up Indulgent Gathering. In: Algorithms for Sensor Systems. ALGOSENSORS 2021. No. 12961 in LNCS (2021)
6. Bramas, Q., Tixeuil, S.: Wait-free gathering without chirality. In: 22nd Structural Information and Communication Complexity (SIROCCO). pp. 313–327. No. 9439 in LNCS (2015)
7. Castaneda, A., Rajsbaum, S., Alcántara, M., Flores-Penaloza, D.: Fault-tolerant robot gathering problems on graphs with arbitrary appearing times. In: 2017 IEEE International Parallel and Distributed Processing Symposium (IPDPS). pp. 493–502 (2017)
8. Castenow, J., Fischer, M., Harbig, J., Jung, D., auf der Heide, F.M.: Gathering anonymous, oblivious robots on a grid. Theoretical Computer Science **815**, 289–309 (2020)
9. Cicerone, S., DiStefano, G., Navarra, A.: Gathering robots in graphs: The central role of synchronicity. Theoretical Computer Science **849**, 99–120 (2021)
10. Défago, X., Potop-Butucaru, M., Raipin-Parvédy, P.: Self-stabilizing gathering of mobile robots under crash or byzantine faults. Distributed Computing **33**, 393–421 (2020)
11. D'Angelo, G., Navarra, A., Nisse, N.: A unified approach for gathering and exclusive searching on rings under weak assumptions. Distributed Computing **30**(1), 17–48 (2017)

12. D'Angelo, G., Stefano, G.D., AlfredoNavarra: Gathering on rings under the look–compute–move model. Distributed Computing **27**(4), 255–285 (2014)
13. D'Angelo, G., Stefano, G.D., Klasing, R., Navarra, A.: Gathering of robots on anonymous grids and trees without multiplicity detection. Theoretical Computer Science **610**, 158–168 (2016)
14. Flocchini, P., Prencipe, G., Santoro, N. (eds.): Distributed Computing by Mobile Entities, Current Researchin Moving and Computing. No. 11340 in LNCS, Springer (2019)
15. Izumi, T., Izumi, T., Kamei, S., Ooshita, F.: Time-optimal gathering algorithm of mobile robots with local weak multiplicity detection in rings. IEICE Transactions **96-A**(6), 1072–1080 (2013)
16. Kamei, S., Lamani, A., Ooshita, F., Tixeuil, S.: Asynchronous mobile robot gathering from symmetric configurations without global multiplicity detection. In: Structural Information and Communication Complexity. SIROCCO 2011. No. 6796 in LNCS (2011)
17. Kamei, S., Lamani, A., Ooshita, F., Tixeuil, S.: Gathering an even number of robots in an odd ring without global multiplicity detection. In: Mathematical Foundations of Computer Science 2012. MFCS 2012. No. 7464 in LNCS (2012)
18. Kamei, S., Lamani, A., Ooshita, F., Tixeuil, S., Wada, K.: Asynchronous Gathering in a Torus. In: 25th International Conference on Principles of Distributed Systems (OPODIS 2021). Leibniz International Proceedings in Informatics (LIPIcs), vol. 217, pp. 9:1–9:17 (2021)
19. Klasing, R., Kosowski, A., Navarra, A.: Taking advantage of symmetries: Gathering of many asynchronous oblivious robots on a ring. Theoretical Computer Science **411**(34-36), 3235—-3246 (2010)
20. Klasing, R., Markou, E., Pelc, A.: Gathering asynchronous oblivious mobile robots in a ring. Theoretical Computer Science **390**(1), 27–39 (2008)
21. Ooshita, F., Tixeuil, S.: On the self-stabilization of mobile oblivious robots in uniform rings. Theoretical Computer Science **568**, 84–96 (2015)
22. Suzuki, I., Yamashita, M.: Distributed anonymous mobile robots: Formation of geometric patterns. SIAM Journal on Computing **28**(4), 1347–1363 (1999)