Gathering on Rings for Myopic Asynchronous Robots With Lights

Sayaka Kamei
Graduate School of Engineering, Hiroshima University, Japan
s10kamei@hiroshima-u.ac.jp

Anissa Lamani
Ecole internationale des sciences du traitement de l'information, Cergy, France
anissa.lamani@eisti.eu

Fukuhito Ooshita
Graduate School of Science and Technology, Nara Institute of Science and Technology, Japan
f-oosita@is.naist.jp

Sébastien Tixeuil
Sorbonne University, Paris, France
Sebastien.Tixeuil@lip6.fr

Koichi Wada
Faculty of Science and Engineering, Hosei University, Japan
wada@hosei.ac.jp

Abstract

We investigate gathering algorithms for asynchronous autonomous mobile robots moving in uniform ring-shaped networks. Different from most work using the Look-Compute-Move (LCM) model, we assume that robots have limited visibility and lights. That is, robots can observe nodes only within a certain fixed distance, and emit a color from a set of constant number of colors. We consider gathering algorithms depending on two parameters related to the initial configuration: M_{init}, which denotes the number of nodes between two border nodes, and O_{init}, which denotes the number of nodes hosting robots between two border nodes. In both cases, a border node is a node hosting one or more robots that cannot see other robots on at least one side. Our main contribution is to prove that, if M_{init} or O_{init} is odd, gathering is always feasible with three or four colors. The proposed algorithms do not require additional assumptions, such as knowledge of the number of robots, multiplicity detection capabilities, or the assumption of towerless initial configurations. These results demonstrate the power of lights to achieve gathering of robots with limited visibility.

2012 ACM Subject Classification Theory of computation → Distributed algorithms

Keywords and phrases LCM robot system, ASYNC schedulers, myopic, luminous, ring networks

Digital Object Identifier 10.4230/LIPIcs.OPODIS.2019.27

Related Version A full version of the paper is available at [13], `http://arxiv.org/abs/1911.04757`.

Funding Supported in part by JST SICORP Grant Number JPMJSC1806, Japan.
Sayaka Kamei: Supported in part by JSPS KAKENHI No. 19K11828.
Fukuhito Ooshita: Supported in part by JSPS KAKENHI No. 18K11167.
Sébastien Tixeuil: Supported in part by Hiroshima University and by project ESTATE (Ref. ANR-16-CE25-0009-03).
Koichi Wada: Supported in part by JSPS KAKENHI No. 17K00019.

1 Introduction

A lot of research about autonomous mobile robots coordination has been conducted by the distributed computing community. The common goal of these research is to clarify the minimum capabilities for robots to achieve a given task. Hence, most work adopts weak assumptions such as: robots are identical (*i.e.*, robots execute the same algorithm and cannot be distinguished), oblivious (*i.e.*, robots have no memory to record past actions), and silent (*i.e.*, robots cannot send messages to other robots). In addition, to model the behavior of robots, most work uses the Look-Compute-Move (LCM) model introduced by Suzuki and Yamashita [18]. In the LCM model, each robot repeats executing cycles of Look, Compute and Move phases. During the Look phase, the robot takes a snapshot to observe the positions of other robots. According to this snapshot, the robot computes the next movement during the Compute phase. If the robot decides to move, it moves to the target position during the Move phase. By using the LCM model, it is possible to clarify problem solvability both continuous environments (*i.e.*, two- or three-dimensional Euclidean space) and discrete environments (*i.e.*, graph networks). State-of-the-art surveys are given in the recent book by Flocchini *et al.* [8].

In this paper, we focus on gathering in graph networks. The goal of gathering is to make all robots gather at a non-predetermined single node. Since gathering is a fundamental task of mobile robot systems and a benchmark application, numerous algorithms have been proposed for various graph network topologies. In particular, many papers focus on ring-shaped networks because symmetry breaking becomes a core difficulty, and any such solution is likely to adapt well on other topologies, as it is possible to make virtual rings over arbitrary networks and hence use ring algorithms in such networks[14, 11, 5, 3, 4].

Klasing *et al.* [14] proposed gathering algorithms for rings with *global-weak* multiplicity detection. Global-weak multiplicity detection enables a robot to detect whether the number of robots on each node is one, or more than one. However, the exact number of robots on a given node remains unknown if there is more than one robot on the node. Izumi *et al.* [11] provided a gathering algorithm for rings with *local-weak* multiplicity detection under the assumption that the initial configurations are non-symmetric and non-periodic, and that the number of robots is less than half the number of nodes. Local-weak multiplicity detection enables a robot to detect whether the number of robots on its *current* node is one, or more than one. D'Angelo *et al.* [5, 3] proposed unified ring gathering algorithms for most of the solvable initial configurations, using global-weak multiplicity detection [5], or local-weak multiplicity detection [3]. Finally, Klasing *et al.* [4] proposed gathering algorithms for grids and trees. All aforementioned work assumes *unlimited visibility*, that is, each robot can take a snapshot of the whole network graph with all occupied positions.

The unlimited visibility assumption somewhat contradicts the principle of weak mobile robots, hence several recent studies focus on *myopic* robots [9, 10, 12, 7]. A myopic robot has limited visibility, that is, it can take a snapshot of nodes (with occupying robots) only within a certain fixed distance ϕ. Not surprisingly, many problems become impossible to solve in this setting, and several strong assumptions have to be made to enable possibility results. Datta *et al.* [7] study the problem of ring exploration with different values for ϕ. Guilbault *et al.* [9] study gathering in bipartite graphs with the global-weak multiplicity detection (limited to distance ϕ) in case of $\phi = 1$, and prove that gathering is feasible only when robots form a star in the initial configuration. They also study the case of infinite lines with $\phi > 1$ [10], and prove that no universal algorithm exists in this case. In the case of rings, since a ring with even nodes is also a bipartite graph, gathering is feasible only

Table 1 Summary of gathering algorithms for myopic robots in rings. All algorithms assume that the visibility graph is connected, and that there exist two border nodes. R denotes the number of robots and H_{center} denotes the size of the center hole in the initial configuration. The global (resp., local) strong multiplicity detection enables a robot to observe the exact number of robots on every node (resp., on its current node).

	ϕ	#colors	Multiplicity detection	Assumptions
[12]	≥ 1	1	local strong	R is odd R is known towerless initial configuration
[12]	≥ 3	1	global strong	R is even R is known H_{center} is odd $H_{center} \leq \phi - 2$ towerless initial configuration
Algorithm 1	≥ 1	3	none	M_{init} is odd
Algorithm 2	≥ 2	4	none	O_{init} is odd

when three robots occupy three successive nodes. For this reason, Kamei *et al.* [12] give gathering algorithms for rings with $\phi \geq 1$ by using strong assumptions, such as knowledge of the number of robots, and strong multiplicity detection, which enables a robot to obtain the exact number of robots on a particular node. Overall, limited visibility severely hinders the possibility of gathering oblivious robots on rings.

On the other hand, completely oblivious robots (that can not remember past actions) may be too weak of a hypothesis with respect to a possible implementation on real devices, where persistent memory is widely available. Recently, enabling the possibility that robots maintain a non-volatile visible light [6] has attracted a lot of attention to improve the task solvability. A robot endowed with such a light is called a *luminous* robot. Each luminous robot can emit a light to other robots whose color is chosen among a set of colors whose size is constant. The light color is non-volatile, and so it can be used as a constant-space memory. Viglietta [19] gives a complete characterization of the rendezvous problem (that is, the gathering of two robots) on a plane using two visible colored lights assuming unlimited visibility robots. Das *et al.* [6] prove that unlimited visibility robots on two-dimensional space with a five-color light have the same computational power in the asynchronous and semi-synchronous models. Di Luna *et al.* [15] discuss how lights can be used to solve some classical distributed problems such as rendezvous and forming a sequence of patterns. The robots they assume have unlimited visibility, but they also discuss the case where robots visibility is limited by the presence of obstruction. Hence, luminous robots seem to dramatically improve the possibility to solve tasks in the LCM model.

As a result of the above observations, it becomes interesting to study the interplay between the myopic and luminous properties for LCM robots: can lights improve task solvability of myopic robots. To our knowledge, only three papers [17, 16, 2] consider this combination. Ooshita *et al.* [17] and Nagahama *et al.* [16] demonstrate that for the task of ring exploration, even a two-color light significantly improves task solvability. Bramas *et al.* [2] give exploration algorithms for myopic robots in infinite grids by using a constant-color light. To this day, the characterization of gathering feasibility for myopic luminous robots (aside from the trivial case where a single color is available) is unknown.

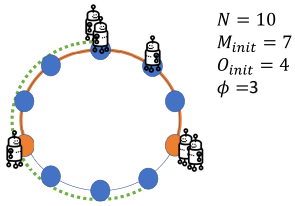

Figure 1 Example of an initial configuration where $M_{init} = 7$ and $O_{init} = 4$. Border nodes are represented in orange color. The robot on the left border node can sense the range represented by green dots.

Our contributions. We clarify the solvability of gathering for myopic luminous robots in rings. We consider the asynchronous (ASYNC) model, which is the most general timing assumption. As in a previous work [12], we focus on initial configurations such that the visibility graph[1] is connected and there exist two border nodes[2] (see Figure 1 for an example). Both assumptions are necessary for the class of *cautious* gathering algorithms (see Lemmas 24 and 25). A cautious gathering protocol never expands the span of a given visibility graph. In addition, we assume that all robots have the same color in initial configurations. We partition initial configurations using two parameters M_{init} and O_{init}; M_{init} is defined as the number of nodes between two border nodes, and O_{init} is defined as the number of nodes occupied by some robots (also see Figure 1). We can easily observe that, if both M_{init} and O_{init} are even, there exist (so-called *edge-symmetric*) initial configurations such that no algorithm achieves gathering (Corollary 22). Hence, we consider the case that M_{init} or O_{init} is odd.

On the positive side, our main contribution is to prove that, if M_{init} or O_{init} is odd, gathering is always feasible by using a constant number of colors without additional assumptions (so, no multiplicity detection is necessary) for any positive visible distance ϕ. First, for the case that M_{init} is odd and $\phi \geq 1$ holds, we give a gathering algorithm that uses three colors. Second, for the case that O_{init} is odd and $\phi \geq 2$, we give a gathering algorithm that uses four colors. Note that we assume $\phi \geq 2$ in the second algorithm because, if $\phi = 1$ holds, then $O_{init} = M_{init}$ also holds from the assumption of connected visibility graphs, so the first algorithm can be used in this case. We compare the current work with that of Kamei *et al.* [12] in Table 1. Overall, lights with a constant number of colors permit to remove most of the previously considered assumptions. For example, our algorithms do not require any multiplicity detection (that is, robots do not distinguish whether the number of robots with the same color on a single node is one or more than one). Furthermore, our algorithms solve gathering even if initial configurations include tower nodes (a tower node is a node that hosts multiple robots). These results demonstrate the power of lights to achieve mobile robot gathering with limited visibility.

Because of space limitations, we give proofs of lemmas and theorems in [13].

2 Model

We consider anonymous, disoriented and undirected rings $\mathcal{G}$ of $N(\geq 3)$ nodes $u_0, u_1, \ldots, u_{N-1}$ such that u_i is connected to both $u_{((i-1) \mod N)}$ and $u_{((i+1) \mod N)}$. On this ring, R autonomous robots collaborate to gather at one of the N nodes of the ring, not known beforehand, and remain there indefinitely.

[1] A visibility graph is defined as $\mathcal{G}_V = (\mathcal{R}, \mathcal{E}_V)$ where $\mathcal{R}$ is a set of all robots, and $\mathcal{E}_V$ is a set of pairs of robots that can observe each other.

[2] Node v is a border node if robots on v can observe other robots only in one direction.

The *distance* between two *nodes* u and v on a ring is the number of edges in a shortest path connecting them. The *distance* between two *robots* r_1 and r_2 is the distance between two nodes occupied by r_1 and r_2, respectively. Two robots or two nodes are *neighbors* if the distance between them is one.

Robots are identical, *i.e.*, they execute the same program and use no localized parameter such as an identifier or a particular orientation. Also, they are oblivious, *i.e.*, they cannot remember their past observations or actions. We assume that robots do *not* know N, the size of the ring, and R, the number of robots.

Each robot r_i maintains a variable L_i, called *light*, which spans a finite set of states called *colors*. A light is *persistent* from one computational cycle to the next: the color is not automatically reset at the end of the cycle. Let K denote the number of available light colors. Let $L_i(t)$ be the light color of r_i at time t. We assume the *full light* model: each robot r_i can see the light of other robots, but also its own light. Robots are unable to communicate with each other explicitly (*e.g.*, by sending messages), however, they can observe their environment, including the positions and colors of the other robots. We assume that besides colors, robots do *not* have multiplicity detection capability: if there are multiple robots $r_1, r_2, \ldots r_k$ in a node u, an observing robot r can detect only colors, so r can detect there are multiple robots at u if and only if at least two robots among $r_1, r_2, \ldots r_k$ have different colors. So, a robot r observing a single color at node u cannot know how many robots are located in u.

Based on the sensing result, a robot r may decide to move or to stay idle. At each time instant $t_i (1 \leq i)$, robots occupy nodes of the ring, their positions and colors form a *configuration* $C(t_i)$ of the system at time t_i. When $C(t_i)$ reaches $C(t_{i+1})$ by executing some phases between t_i and t_{i+1}, it is denoted as $C(t_i) \longrightarrow C(t_{i+1})$. The reflexive and transitive closure is denoted as $\longrightarrow^*$.

We assume that robots have limited visibility: an observing robot r at node u can only sense the robots that occupy nodes within a certain distance, denoted by ϕ ($\phi \geq 0$), from u. As robots are identical, they share the same ϕ.

Let $\mathcal{X}_i(t)$ be the set of colors of robots located in node u_i at time t. If a robot r_j located at u_i at t, the sensor of r_j outputs a sequence, $\mathcal{V}_j$, of $2\phi + 1$ set of colors:

$$\mathcal{X}_{i-\phi}(t), \ldots, \mathcal{X}_{i-1}(t), (\mathcal{X}_i(t)), \mathcal{X}_{i+1}(t), \ldots, \mathcal{X}_{i+\phi}(t).$$

This sequence $\mathcal{V}_j$ is the *view* of r_j. If the sequence $\mathcal{X}_{i+1}, \ldots, \mathcal{X}_{i+\phi}$ is equal to the sequence $\mathcal{X}_{i-1}, \ldots, \mathcal{X}_{i-\phi}$, then the view $\mathcal{V}_j$ of r_j is *symmetric*. Otherwise, it is *asymmetric*. In $\mathcal{V}_j$, a node u_k is *occupied* at instant t whenever $|\mathcal{X}_k(t)| > 0$. Conversely, if u_k is not occupied by any robot at t, then $\mathcal{X}_k(t) = \emptyset$ holds, and u_k is *empty* at t.

If there exists a node u_i such that $|\mathcal{X}_i(t)| = 1$ holds, u_i is *singly-colored*. Note that $|\mathcal{X}_i(t)|$ denotes the number of colors at node u_i, thus even if u_i is singly-colored, it may be occupied by multiple robots (sharing the same color). Now, if a node u_i is such that $|\mathcal{X}_i(t)| > 1$ holds, u_i is *multiply-colored*. As each robot has a single color, a multiply-colored node always hosts more than one robot.

In the case of a robot r_j located at a singly-colored node u_i, r_j's view $\mathcal{V}_j$ contains an $\mathcal{X}_i(t)$ that can be written as $[L_j]$. Then, if the left node of u_i contains one or more robots with color L_k, and the right node of u_i contains one or more robots with color L_l, while u_i only hosts r_j, then $\mathcal{V}_j$ can be written as $L_k[L_j]L_l$. Now, if robot r_j at node u_i occupies a multiply-colored position (with two other robots r_k and r_l having distinct colors), then

$|\mathcal{X}_i(t)| = 3$, and we can write $\mathcal{X}_i(t)$ in $\mathcal{V}_j$ as $\begin{bmatrix} L_k \\ L_l \\ [L_j] \end{bmatrix}$. When the view does not consist of a single observed node, we use brackets to distinguish the current position of the observing robot in the view and the inner bracket to explicitly state the observing robot's color.

Our algorithms are driven by observations made on the current view of a robot, so many instances of the algorithms we use *view predicates*: a Boolean function based on the current view of the robot. The predicate L_j matches any set of colors that includes color L_j, while predicate (L_j, L_k) matches any set of colors that contains L_j, L_k, or both. Now the predicate $\binom{L_1}{L_2}$ matches any set that contains *both* L_1 and L_2. Some of our algorithm rules expect that a node is singly-colored, *e.g.* with color L_k, in that case, the corresponding predicate is denoted by $L_k!$. To express predicates in a less explicit way, we use character '?' to represent any non-empty set of colors, so a set of colors $\mathcal{X}_i \neq \emptyset$ satisfies predicate '?'. The $\neg$ operator is used to negate a particular predicate P (so, $\neg P$ returns *false* whenever P returns *true* and vice versa). Also, the superscript notation P^y represents a sequence of y consecutive sets of colors, each satisfying predicate P. Observe that $y \leq \phi$. In a given configuration, if the view of a robot r_j at node u_i satisfies predicate $\emptyset^\phi[?]$ or predicate $[?]\emptyset^\phi$, then r_j is a *border robot* and u_i a *border node*. Sometimes, we require a particular color L_1 to be present at some position u_i and a particular color L_2 *not* to be present at some position u_i. For the above case, the corresponding predicate would be: $\binom{L_1}{\neg L_2}$.

In this paper, we aim at maintaining the property that at most two border nodes exist at any time. On the ring $\mathcal{G}$, let H_{max} be the size of the maximum hole (*i.e.*, the maximum sequence of empty nodes). Note that by the assumptions, at instant $t = 0$ (*i.e.*, in the initial configuration), $H_{max} > \phi$ holds. Let $\mathcal{V}'$ be the subset of nodes on a path between two border nodes u and v, such that all robots are hosted by nodes in $\mathcal{V}'$. Also, let $\mathcal{G}'$ be the subgraph of $\mathcal{G}$ induced by $\mathcal{V}'$. Note that, $\mathcal{G}'$ does not include the hole with the size H_{max}. At instant $t = 0$, let H_{init} be the maximum distance between occupied nodes in $\mathcal{G}'$, M_{init} be the number of nodes in $\mathcal{G}'$, and $O_{init} (\leq M_{init})$ be the number of occupied nodes in $\mathcal{G}'$. We assume that $\phi \geq H_{init} \geq 1$ holds. Note that, H_{init} is the size of the second maximum hole in the ring because there are two border nodes. As previously stated, no robot is aware of H_{init}, M_{init} and O_{init}. In $\mathcal{G}'$, let D denote the distance between the two border nodes. Note that, at $t = 0$, $D = M_{init} - 1$ holds.

Each robot r executes Look-Compute-Move cycles infinitely many times: (*i*) first, r takes a snapshot of the environment and obtains an ego-centered view of the current configuration (Look phase), (*ii*) according to its view, r decides to move or to stay idle and possibly changes its light color (Compute phase), (*iii*) if r decided to move, it moves to one of its neighbor nodes depending on the choice made in the Compute phase (Move phase). At each time instant t, a subset of robots is activated by an entity known as the scheduler. This scheduler is assumed to be fair, *i.e.*, all robots are activated infinitely many times in any infinite execution. In this paper, we consider the most general asynchronous model: the time between Look, Compute, and Move phases is finite but unbounded. We assume however that the move operation is atomic, that is, when a robot takes a snapshot, it sees robots colors on nodes and not on edges. Since the scheduler is allowed to interleave the different phases between robots, some robots may decide to move according to a view that is different from the current configuration. Indeed, during the compute phase, other robots may move. Both the view and the robot are in this case said to be *outdated*.

In this paper, each rule in the proposed algorithms is presented in the similar notation as in [17]: $< Label > : < Guard > :: < Statement >$. The guard is a predicate on the view $\mathcal{V}_j = \mathcal{X}_{i-\phi}, \ldots, \mathcal{X}_{i-1}, (\mathcal{X}_i), \mathcal{X}_{i+1}, \ldots, \mathcal{X}_{i+\phi}$ obtained by robot r_j at node u_i during the Look phase. If the predicate evaluates to *true*, r_j is *enabled*, otherwise, r_j is *disabled*. In the first case, the corresponding rule $< Label >$ is also said to be *enabled*. If a robot r_j is enabled, r_j may change its color and then move based on the corresponding statement during its subsequent Compute and Move phases. The statement is a pair of (*New color*, *Movement*). Movement can be $(i) \rightarrow$, meaning that r_j moves towards node u_{i+1}, $(ii) \leftarrow$, meaning that r_j moves towards node u_{i-1}, and $(iii) \perp$, meaning that r_j does not move. For simplicity, when r_j does not move (resp. r_j does not change its color), we omit *Movement* (resp. *New color*) in the statement. The label $< Label >$ is denoted as R followed by a non-negative integer (*i.e.*, R0, R1, etc.) where a smaller label indicates higher priority.

3 Algorithms

In this section, we propose two algorithms for myopic luminous robots. One is for the case that M_{init} is odd, uses three colors ($K = 3$). The other is for the case that M_{init} is even and O_{init} is odd, uses four colors ($K = 4$). We assume that the initial configurations satisfy the following conditions:

- All robots have the same color White,
- Each occupied node can have multiple robots, and
- $\phi \geq H_{init} \geq 1$ holds.

3.1 Algorithm for the case M_{init} is odd

The strategy of our algorithm is as follows: The robots on two border nodes keep their lights Red or Blue, then the algorithm can recognize that they are originally on border nodes. When robots on a border node move toward the center node, they change the color of their light to Blue or Red alternately regardless of the neighboring nodes being occupied, where initially robots become Red colors. To keep the connected visibility graph, when a border node becomes singly-colored, the border robot changes its light to Blue or Red according to the distance from the original border node and moves toward the center node and the neighboring non-border robot becomes a border robot. Eventually, two border nodes become neighboring. Then, one has Blue robots and the other has Red robots because M_{init} is odd. At the last moment, Red robots join Blue robots to achieve the gathering.

The formal description of the algorithm is in Algorithm 1. The rules of our algorithm are as follows:

- R0: If the gathering is achieved, a robot does nothing[3].
- R1: A border White robot on a singly-colored border node changes its light to Red.
- R2a and R2b: A border Red robot on a singly-colored border node changes its light to Blue and moves toward an occupied node.
- R3a and R3b: A border Blue robot on a singly-colored border node changes its light to Red and moves toward an occupied node.

[3] Note, this algorithm and the next one cannot terminate because gathering configurations are not terminating ones due to robots with outdated views even if this rule is executed. Because this rule has higher priority, if it is enabled, robots do not need to check other guards.

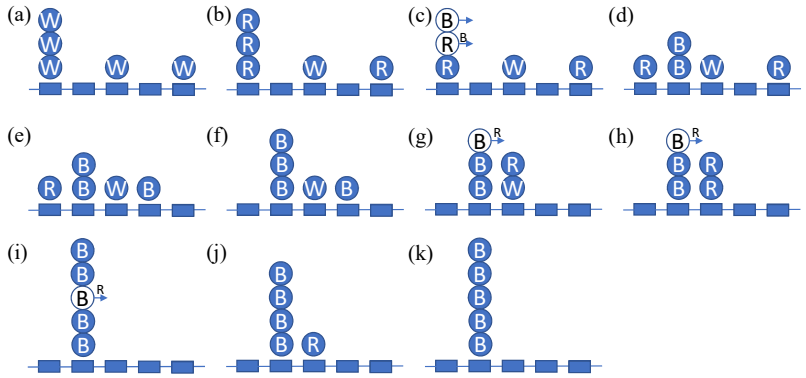

Figure 2 An execution example of Algorithm 1 with $\phi = 2$.

- R4a and R4b: When White robots become border robots, they change their color to the same color as the border Red or Blue robots.
- R5: If two border nodes are neighboring, a border Red robot on a singly-colored border node moves to the neighboring singly-colored node with Blue robots.

Figure 2 illustrates an execution example of Algorithm 1. This figure assumes $\phi = 2$. Figure 2(a) shows an initial configuration. First, border White robots change their lights to Red by R1 (Fig. 2(b)). Next, left border Red robots move by R2a. Since we consider the ASYNC model, some robots may become outdated. In Fig. 2(c), the top robot has changed its light but not yet moved, and the middle robot has looked but not yet changed its light. The outdated robots move in Fig. 2(d), and then the right border Red robot also moves in Fig. 2(e) by R2a. Here, one left border Red robot has not yet moved. However, since it still observes a White robot, it can move by R2a (Fig. 2(f)). Then, border Blue robots can move by R3b. In Fig. 2(g), one left border robot becomes outdated and the right border robot completes the movement. After the right border White robot changes its light to Red by R4a (Fig. 2(h)), the right border Red robots move by R5 (Fig. 2(i)). Note that, all robots stay at a single node but one of them is outdated. Hence the robot moves after that (Fig. 2(j)), but it can go back to the gathering node by R5 (Fig. 2(k)). Now robots have achieved gathering.

In the case of $\phi = 1$, each robot can view only its neighboring nodes. In this case, we should add some assumptions as follows:

- R2a and R3a are always disabled.
- R2b and R3b are enabled when there are White robots in the neighboring node.
- R5 is enabled when the neighboring node is singly-colored with Blue robots.

In that case, because of the connectivity of the visibility graph, there is no empty node in $\mathcal{G}'$. That is, until gathering is achieved, there is at least one White robot on the neighboring node. Thus, such assumption is natural and we can prove the correctness in the same way as other cases by deleting R2a and R3a (the case such that there is no White robot on the neighboring node but gathering is not achieved).

We now discuss the correctness of our algorithm. Because there is no rule such that non-border robot r_i can execute, we can derive the following lemma.

▶ **Lemma 1.** *When a robot r_i looks, if it is a non-border robot, it cannot execute any action.*

To discuss the correctness, we consider the time instants such that the distance D between two border nodes has just reduced at least one. The duration between them is called *mega-cycle*. Note that, D is reduced at most two by the algorithm during a mega-cycle.

Algorithm 1 Algorithm for the case that M_{init} is odd.

Colors
W (White), R (Red), B (Blue)

Rules

/* Do nothing after gathering. */
R0: $\emptyset^{\phi}[?]\emptyset^{\phi} :: \bot$

/* Start by the initial border robot. */
R1: $\emptyset^{\phi}[W!](\neg\emptyset^{\phi}) :: R$

/* Border robots on singly-colored nodes change their color alternately and move. */
R2a: $\emptyset^{\phi}[R!](\emptyset, B!)(\neg(\emptyset^{\phi-1})) :: B, \rightarrow$

R2b: $\emptyset^{\phi}[R!](W)(?^{\phi-1}) :: B, \rightarrow$

R3a: $\emptyset^{\phi}[B!](\emptyset, R!)(\neg(\emptyset^{\phi-1})) :: R, \rightarrow$

R3b: $\emptyset^{\phi}[B!](W)(?^{\phi-1}) :: R, \rightarrow$

/* When White robots become a border robot, they change their color to the same color as the border robot. */
R4a: $\emptyset^{\phi}\begin{bmatrix} R \\ [W] \end{bmatrix}(\neg\emptyset^{\phi}) :: R$

R4b: $\emptyset^{\phi}\begin{bmatrix} B \\ [W] \end{bmatrix}(\neg\emptyset^{\phi}) :: B$

/* When two border nodes are neighboring, robots gather to a node with the Blue border robots. */
R5: $\emptyset^{\phi}[R!](B!)(\emptyset^{\phi-1}) :: B, \rightarrow$

Let $t_0, t_1, \ldots t_i, \ldots$ be starting times of mega-cycles, where t_0 is the starting time of the algorithm and for each i (≥ 1), D is reduced at least one from t_{i-1} at time t_i. Letting $C(t_i)$ be the configuration at time t_i, the transition of configurations from t_i to t_{i+1} is denoted as $C(t_i) \xrightarrow{MC} C(t_{i+1})$.

Figure 3 shows a transition diagram of configurations for each mega-cycle (We prove all transitions later). The small blue box represents a node, and the circle represents a set of robots (One single circle may represent a set of robots with the same color). The doubly (resp. singly) lined arrows represent that D decreases by 2 (resp. 1). The letter in each circle represents the color of the lights. The circles in parentheses represent that they are optional. The circles in brackets represent that one of them should exist. In Conf 1-3, there is no Red or Blue robot with an outdated view (*i.e.*, all robots are after Move phases before they look). In Conf 4-7, left side border node represents that there is no Red or Blue robot with an outdated view. Right side border robots in Conf 4-7 represent that they are still working on their movements, *i.e.*, there may be robots with outdated views. The second node from the right can be empty, and there can be White robots on the node. In the right side border node, the white circles represent that the robots can be on the node, but with the outdated view. Actually, there may be White robots which do not change their color to the same as the border color, Red or Blue, yet. We omit such White robots changing to border color for simplicity, *i.e.*, they may be included in any set of White robots in border nodes in this figure[4]. For example, consider an example in Fig. 2. The initial configuration in Fig. 2(a) is represented by Init. The next mega-cycle starts in Fig. 2(e) and this configuration is represented by Conf 4. Note that mapping from left and right border nodes in Conf 1-7 to two border nodes in configurations may change during an execution. The next mega-cycle starts in Fig. 2(f) and this configuration is represented by Conf 1.

Because the border node becomes singly-colored by R4, we can derive the following lemma.

[4] However, it is considered in the proof.

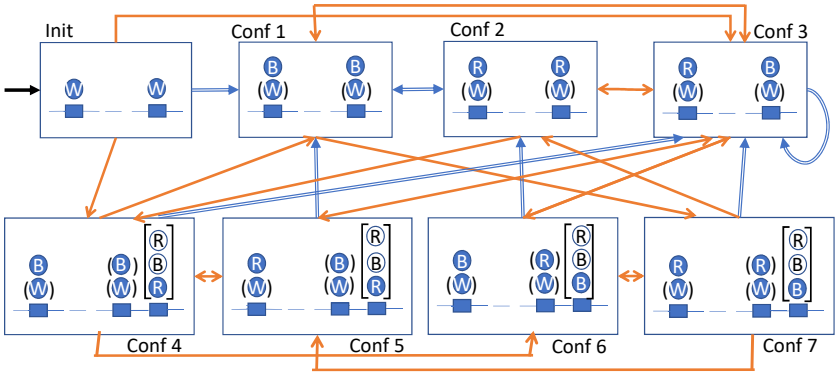

Figure 3 Transition Diagram for Algorithm 1 while $D > 2$.

▶ **Lemma 2.** *Assume that a border node contains White and γ robots, where $\gamma \in \{Red, Blue\}$ at time t. Then there exists a time $t'(> t)$ such that the border node becomes singly-colored one with γ robots. In the case that the border node contains only White robots, the border node becomes Red singly-colored.*

Because border robots on a singly-colored node eventually move to a neighboring node by R2 or R3, the following lemma holds.

▶ **Lemma 3.** *Assume that a border node u is singly-colored with Red (resp. Blue) at time t, then if there is no robot in u at time $t'(> t)$, all robots in u change their color to Blue (resp. Red), move to the neighboring node of u, and the distance between the border nodes is reduced by at least one at t'.*

We showed that the transition diagram in Figure 3 is correct while $D > 2$. In each mega-cycle, D decreases by at least one, and border robots change their color Red and Blue alternately every hop.

▶ **Lemma 4.** *From the initial configuration, D decreases monotonically and eventually becomes 2.*

▶ **Lemma 5.** *Let h be the distance from the original border node to a node u_h in $\mathcal{G}'$. If h is odd (resp. even), a Blue (resp. Red) border robot comes into u_h.*

In the following part, we consider the execution after D becomes 2.

Let Conf BW-MR be the configuration with $D = 1$ such that there are Blue robots and White robots without outdated views in a border node and there are Red robots and Blue robots with outdated views and Red robots without outdated views in the other border node, where Blue robots with outdated views will move to the other border node and Red robots with outdated views will change their color to Blue and move to the other border node (Figure 4a).

Let Conf RW-MB be the configuration with $D = 1$ such that there are Red robots and White robots without outdated views in a border node and there are Red robots and Blue robots with outdated views and Blue robots without outdated views in the other border node, where Red robots with outdated views will move to the other border node and Blue robots with outdated views will change their color to Red and move to the other border node (Figure 4b).

▶ **Lemma 6.** *After D becomes 2, the configuration becomes Conf BW-MR or Conf RW-MB.*

(a) Conf BW-MR **(b)** Conf RW-MB

Figure 4 The configurations Conf BW-MR and Conf RW-MB with $D = 1$.

To show that the gathering is achieved, by Lemmas 4 and 6, we consider the gathering only from Conf BW-MR and Conf RW-MB respectively.

▶ **Lemma 7.** *From Conf BW-MR, the gathering is achieved.*

▶ **Lemma 8.** *From Conf RW-MB, the gathering is achieved.*

By Lemmas 7 and 8, we can derive the following theorem.

▶ **Theorem 9.** *Gathering is solvable in full-light of 3 colors when M_{init} is odd.*

3.2 Algorithm for the case M_{init} is even and O_{init} is odd

In this case, we can assume $\phi > 1$ because the visibility graph is connected. The strategy of our algorithm is similar to Algorithm 1. Initially, all robots are White, and robots on two border nodes become Red in their first activation. The two border robots keep their lights Red or Blue, then the algorithm can recognize that they are originally border robots. When non-border White robots become border robots, they change their color to Red (resp., Blue) if borders that join the node have Red (resp., Blue). To keep the connected visibility graph, when a border node becomes singly-colored, the border robot moves toward the center node. At that time, if there exists a White robot in the directed neighboring node, the border robot changes its color. Otherwise, it just moves without changing its color. Eventually, two border nodes become neighboring. In this algorithm, when two border nodes are neighboring, an additional color Purple is used to decide the gathering point.

The rules of our algorithm are as follows:

- R0: If the gathering is achieved, a robot does nothing.
- R1: A border White robot on a singly-colored border node changes its light to Red.
- R2: A border Red robot on a singly-colored border node moves toward an occupied node without changing its color when there is no White robot on the neighboring node (R2a-1, R2a-2). A border Red robot moves toward an occupied node and changes its light to Blue only when there is at least one White robot on the neighboring node (R2b).
- R3: A border Blue robot on a singly-colored border node moves toward an occupied node without changing its color when there is no White robot on the neighboring node (R3a-1, R3a-2). A border Blue robot moves toward an occupied node and changes its light to Red only when there is at least one White robot on the neighboring node (R3b).
- R4: When White robots become border robots, they change their color to the same color as the border Red or Blue robots.
- R5: If two border nodes are neighboring, every robot moves to the neighboring node with Purple robots (R5a). A border Blue robot on a singly-colored border node changes its light to Purple when there are only Red robots or Red and Blue robots on the neighboring node (R5b-1, R5b-2). A border Blue robot changes its light to Purple when there is Red robot on the same node and the neighboring node is a singly-colored node with Red robots (R5b-3).

Algorithm 2 Algorithm for the case that M_{init} is even and O_{init} is odd.

Colors
W (White), R (Red), B (Blue), P (Purple)

Rules

/* Do nothing after gathering. */
R0: $\emptyset^{\phi}[?]\emptyset^{\phi} :: \bot$

/* Start by the initial border robots. */
R1: $\emptyset^{\phi}[W!](\neg\emptyset^{\phi}) :: R$

/* Border robots on singly-colored nodes move inwards. */
R2a-1: $\emptyset^{\phi}[R!](\emptyset)(\neg(\emptyset^{\phi-1})) :: \rightarrow$

R2a-2: $\emptyset^{\phi}[R!]\begin{pmatrix}\neg W\\ R\end{pmatrix}(\neg(\emptyset^{\phi-1})) :: \rightarrow$

R2b: $\emptyset^{\phi}[R!](W)(?^{\phi-1}) :: B, \rightarrow$

R3a-1: $\emptyset^{\phi}[B!](\emptyset)(\neg(\emptyset^{\phi-1})) :: \rightarrow$

R3a-2: $\emptyset^{\phi}[B!]\begin{pmatrix}\neg W\\ B\end{pmatrix}(\neg(\emptyset^{\phi-1})) :: \rightarrow$

R3b: $\emptyset^{\phi}[B!](W)(?^{\phi-1}) :: R, \rightarrow$

/* When White robots become border robots, they change their color to the same color as the border robots. */
R4a: $\emptyset^{\phi}\begin{bmatrix}R\\ [W]\end{bmatrix}(\neg\emptyset^{\phi}) :: R$

R4b: $\emptyset^{\phi}\begin{bmatrix}B\\ [W]\end{bmatrix}(\neg\emptyset^{\phi}) :: B$

/* When two border nodes are neighboring, they gather to the border node with Purple robots. */
R5a: $\emptyset^{\phi}[?](P)(\emptyset^{\phi-1}) :: \rightarrow$

R5b-1: $\emptyset^{\phi}[B!](R!)(\emptyset^{\phi-1}) :: P$

R5b-2: $\emptyset^{\phi}[B!]\begin{pmatrix}R\\ B\end{pmatrix}(\emptyset^{\phi-1}) :: P$

R5b-3: $\emptyset^{\phi}\begin{bmatrix}R\\ [B]\end{bmatrix}(R!)(\emptyset^{\phi-1}) :: P$

The formal description of the algorithm is in Algorithm 2. Figure 5 illustrates an execution example of Algorithm 2. This figure assumes $\phi = 3$. Figure 5(a) shows an initial configuration. First, the left border White robot changes its light to Red by R1 (Fig. 5(b)). Next, the left border Red robot moves by R2a-1 (Fig. 5(c)). Note that, here, the robot does not change its light. In the next movement, the left border Red robot moves to a node with a White robot by R2b, and hence it changes its light to Blue (Fig. 5(d)). Then the left border White robot changes its light to Blue (Fig. 5(e)). Left border Blue robots can move by R3a-1. In Fig. 5(f), one of them completes the movement. In this case, another robot can move by R3a-2 (Fig. 5(g)). Next, right border White robots change their lights by R1 (Fig. 5(h)). After that, left and right border robots move toward each other by R2a-1 and R3a-1. In Fig. 5(i), some Blue and Red robots meet at a node but border robots continue to move until the number of occupied nodes is at most two by R2a-2 and R3a-2 (Fig. 5(j)). After the number of occupied nodes is at most two, some robots change their lights to Purple. In this case, the left Blue robot changes its light by R5b-2 (Fig. 5(k)). After that, all robots move to the node with a Purple robot by R5a and achieve gathering (Fig. 5(l)).

We now discuss the correctness of Algorithm 2. As with Algorithm 1, since there is no rule that non-border robot can execute by the definition of Algorithm 2, the following lemma holds.

▶ **Lemma 10.** *When a robot r_i looks, if it is a non-border robot, it cannot execute any action.*

To discuss the correctness, we change the definition of mega-cycle as follows: We consider the time instants such that the number of occupied nodes with White robots among non-border nodes (denoted as $\#O_W$) has just reduced at least one. That is, mega-cycles end

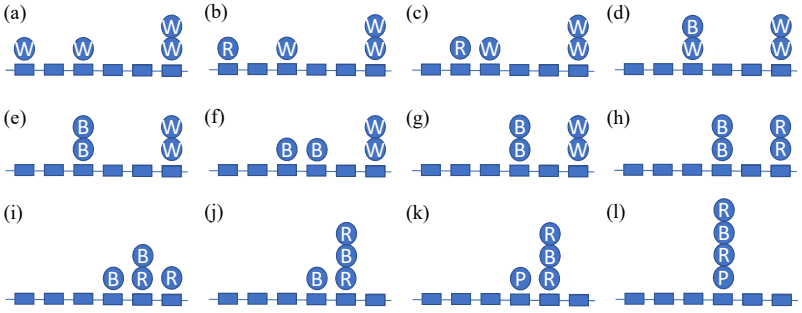

Figure 5 An execution example of Algorithm 2 with $\phi = 3$.

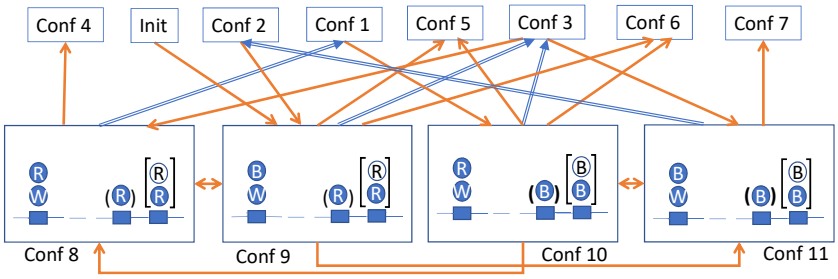

Figure 6 Transition Diagram for Algorithm 2 while $D > 2$.

when either the position of the border node moves to the nearest occupied node with White robots. If the position of the border node moves to the nearest occupied node with White robots, we say *the border absorbs White robots*.

Figure 6 shows a transition diagram of configurations for every mega-cycle. The doubly (resp. singly) lined arrows represent that $\#O_W$ is decreased by 2 (resp. 1). In the diagram, Init and Conf 1-7 are the same as those of Algorithm 1 and they have the same transitions between them as shown in Figure 3, where note that each node with W circle contains at least one White robot. In addition to these configurations, there exist four configurations Conf 8-11 in Algorithm 2. In Conf 1-3, both borders absorb White robots. In Conf 4-7, when one border absorbs White robots, the other border node is neighbored to an occupied node with a White robot. On the other hand, in Conf 8-11, when one border absorbs White robots, the other border node is not neighbored to any occupied node with a White robot.

The following lemma can be proved similarly to the proof of Lemma 3.

▶ **Lemma 11.** *Assume that a border node u is singly-colored with Red (resp. Blue) at time t and there is no robot in u at time $t'(> t)$. If the neighboring node of u (denoted as v) is an occupied node with a White robot at t, all robots in u change their color to Blue (resp. Red), move to v, and $\#O_W$ and D are reduced by at least one at t'. Otherwise, that is, when v is a node without White robots at t, all robots move to v and do not change their color and D is reduced by at least one at t'.*

By Lemma 11, each border node moves to the occupied node until either border node absorbs White robots in any mega-cycle. Transitions among Init and Conf 1-7 can be proved similarly to those in the corresponding lemmas using Lemma 11 instead of Lemma 3. The difference occurs when one border absorbs White robots and the neighboring node of the other border node is without White robots. In this case, since robots on the other border nodes moves to the neighboring node, the configuration becomes Conf 8-11. The proofs can be done similarly.

Also, we showed that the transition diagram in Figure 6 is correct while $D > 2$. Then, in each mega-cycle, because the number of occupied nodes between two borders decreases by at least one, the following lemma holds.

▶ **Lemma 12.** *From the initial configuration, $\#O_W$ decreases monotonically and eventually becomes at most one.*

▶ **Lemma 13.** *Let h be the number of occupied nodes where an original border robot r_h absorbed White robots in $\mathcal{G}'$ from the initial configuration and let u_h denote the current border node r_h is located. If h is odd (resp. even), r_h's light is Blue (resp. Red) when r_h comes into u_h.*

It can be easily verified by Figures 3 and 6 and Lemmas 12–13 that the following configurations occur when $\#O_W$ becomes at most one.

1. Conf 1, Conf 2, Conf 5 and Conf 6 ($D \geq 2$ and $\#O_W = 1$).
2. Conf 3 ($D \geq 1$ and $\#O_W = 0$) and Conf 9 ($D \geq 2$ and $\#O_W = 0$), and Conf 10 ($D \geq 2$ and $\#O_W = 0$).

In the former case, for configurations Conf 1,2,5,6($D = 2$ and $\#O_W = 1$), we have the following lemma, where Conf RW-MB and Conf BW-MR have been defined in the proof for Algorithm 1. In Conf 3($D = 1$), both border nodes may contain White robots with outdated views and contain no White robots.

▶ **Lemma 14.**
1. *Conf 1($D = 2$ and $\#O_W = 1$) $\to^*$ Conf RW-MB, or Conf 3($D = 1$)*
2. *Conf 2($D = 2$ and $\#O_W = 1$) $\to^*$ Conf BW-MR, or Conf 3($D = 1$)*
3. *Conf 5($D = 2$ and $\#O_W = 1$) $\to^*$ Conf BW-MR, or Conf 3($D = 1$)*
4. *Conf 6($D = 2$ and $\#O_W = 1$) $\to^*$ Conf RW-MB, or Conf 3($D = 1$)*

Otherwise ($D \geq 3$), we have the following transitions.

▶ **Lemma 15.**
1. *Conf 1($D \geq 3$ and $\#O_W = 1$) $\xrightarrow{MC}{}^*$ Conf 3($D \geq 1$ and $\#O_W = 0$), or Conf 10($D \geq 2$ and $\#O_W = 0$)*
2. *Conf 2($D \geq 3$ and $\#O_W = 1$)$\xrightarrow{MC}{}^*$ Conf 3($D \geq 1$ and $\#O_W = 0$), or Conf 9($D \geq 2$ and $\#O_W = 0$)*
3. *Conf 5($D \geq 3$ and $\#O_W = 1$)$\xrightarrow{MC}{}^*$ Conf 3($D \geq 1$ and $\#O_W = 0$), or Conf BW-MR($D = 1$ and $\#O_W = 0$)*
4. *Conf 6($D \geq 3$ and $\#O_W = 1$)$\xrightarrow{MC}{}^*$ Conf 3($D \geq 1$ and $\#O_W = 0$), or Conf RW-MB($D = 1$ and $\#O_W = 0$)*

We can prove that Conf BW-MR, Conf RW-MB and Conf 3($D = 1$) become gathering configuration in the following lemma.

▶ **Lemma 16.** *From configurations Conf BW-MR, Conf RW-MB and Conf 3($D = 1$), gathering is achieved.*

Then by Lemmas 14-16, it is sufficient to consider Conf 3($D \geq 2$ and $\#O_W = 0$), Conf 9($D \geq 2$ and $\#O_W = 0$), and Conf 10($D \geq 2$ and $\#O_W = 0$) for the former case.

The latter case has the following transitions. Note that, these transition does not reduce $\#O_W$ and just reduces the distance between the two border nodes. Note also that, the destinations of these transitions do not contain any White robots.

▶ **Lemma 17.**
1. *Conf 3 ($D \geq 3$ and $\#O_W = 0$) $\rightarrow^*$ Conf 3(($D = 2$ and $\#O_W = 0$) or Conf RW-BW*
2. *Conf 3($D \geq 3$ and $\#O_W = 0$)$\rightarrow^*$ Conf 10($D = 2$ and $\#O_W = 0$)*
3. *Conf 3($D \geq 3$ and $\#O_W = 0$)$\rightarrow^*$ Conf 9($D = 2$ and $\#O_W = 0$)*

Since Conf 9($D \geq 2$ and $\#O_W = 0$)(resp. Conf 10($D \geq 2$ and $\#O_W = 0$)) becomes Conf 3(($D = 2$ and $\#O_W = 0$) or Conf RW-BW, or Conf 9($D = 2$ and $\#O_W = 0$) (resp. Conf 9($D = 2$ and $\#O_W = 0$)), the correctness proof completes if we can show that these three configurations become gathering ones.

▶ **Lemma 18.** *Conf 3($D = 2$ and $\#O_W = 0$), Conf 9($D = 2$ and $\#O_W = 0$), and Conf 10($D = 2$ and $\#O_W = 0$) become gathering configurations.*

By the above discussion, we can derive the following theorem.

▶ **Theorem 19.** *Gathering is solvable in full-light of 4 colors when M_{init} is even and O_{init} is odd.*

It is an interesting open question whether gathering is solvable or not in full-light of 3 colors when M_{init} is even and O_{init} is odd.

4 Discussion

In this section, we discuss the gathering problem in other cases. First, we consider the case that M_{init} and O_{init} are even.

▶ **Definition 20.** *A configuration is* edge-view-symmetric *if there exist at least two distinct nodes hosting each at least one robot, and an edge (u_i, u_{i+1}) such that, for any integer $k \geq 0$, and for any robot r_1 at node u_{i-k}, there exists a robot r_2 at node u_{i+k+1} such that $\mathcal{V}_1 = \mathcal{V}_2$.*

▶ **Theorem 21.** *Deterministic gathering is impossible from any edge-view-symmetric configuration.*

▶ **Corollary 22.** *Starting from a configuration where M_{init} is even and O_{init} is even, and all robots have the same color, there exist initial configurations (e.g. edge-view-symmetric configurations) that a deterministic algorithm cannot gather.*

We now study the impact of an important property our algorithms satisfy: cautiousness [1].

▶ **Definition 23.** *A gathering algorithm is* cautious *if, in any execution, the direction to move is only toward other occupied nodes, i.e., robots are not adventurous and do not want to expand the covered area.*

Note that, the algorithms we provide in previous sections, only border robots move, and they only move toward occupied other nodes, hence our algorithms are cautious.

▶ **Lemma 24.** *A cautious algorithm that starts from a configuration with more than two border nodes cannot solve gathering.*

▶ **Lemma 25.** *A cautious algorithm that starts from a configuration with no border node cannot solve gathering.*

▶ **Theorem 26.** *For M_{init} even and O_{init} even, there exists no cautious gathering algorithm with $\phi = 1$, even when the initial configuration is* not *edge-view-symmetric.*

5 Conclusion

We presented the first gathering algorithms for myopic luminous robots in rings. One algorithm considers the case where M_{init} is odd, while the other is for the case where O_{init} is odd. The hypotheses used for our algorithms closely follow the impossibility results found for the other cases. Some interesting questions remain open:

- Are there any deterministic algorithms for the case where M_{init} and O_{init} are even (such solutions would have to avoid starting or ending up in an edge-view-symmetric situation)?
- Are there any algorithms for the case where M_{init} (resp. O_{init}) is odd that use fewer colors than ours? The current lower bound for odd M_{init} (resp. O_{init}) is 2 (resp. 3), but our solutions use 3 (resp. 4) colors.

References

1 Z. Bouzid, M. G. Potop-Butucaru, and S. Tixeuil. Optimal Byzantine-resilient convergence in uni-dimensional robot networks. *Theor. Comput. Sci.*, 411(34-36):3154–3168, 2010.

2 Q. Bramas, S. Devismes, and P. Lafourcade. Infinite Grid Exploration by Disoriented Robots. In *SIROCCO*, 2019.

3 G. D'Angelo, A. Navarra, and N. Nisse. A unified approach for gathering and exclusive searching on rings under weak assumptions. *Distributed Computing*, 30(1):17–48, 2017.

4 G. D'Angelo, G. D. Stefano, R. Klasing, and A. Navarra. Gathering of robots on anonymous grids and trees without multiplicity detection. *Theor. Comput. Sci.*, 610:158–168, 2016.

5 G. D'Angelo, G. D. Stefano, and A. Navarra. Gathering on rings under the Look-Compute-Move model. *Distributed Computing*, 27(4):255–285, 2014.

6 S. Das, P. Flocchini, G. Prencipe, N. Santoro, and M. Yamashita. Autonomous mobile robots with lights. *Theor. Comput. Sci.*, 609:171–184, 2016.

7 A. K. Datta, A. Lamani, L. L. Larmore, and F. Petit. Enabling Ring Exploration with Myopic Oblivious Robots. In *IPDPS*, pages 490–499, 2015.

8 P. Flocchini, G. Prencipe, and N. Santoro, editors. *Distributed Computing by Mobile Entities, Current Research in Moving and Computing*, volume 11340 of *LNCS*. Springer, 2019.

9 S. Guilbault and A. Pelc. Gathering asynchronous oblivious agents with local vision in regular bipartite graphs. *Theor. Comput. Sci.*, 509:86–96, 2013.

10 S. Guilbault and A. Pelc. Gathering Asynchronous Oblivious Agents with Restricted Vision in an Infinite Line. In *SSS*, volume 8255, pages 296–310, 2013.

11 T. Izumi, T. Izumi, S. Kamei, and F. Ooshita. Time-Optimal Gathering Algorithm of Mobile Robots with Local Weak Multiplicity Detection in Rings. *IEICE Transactions*, 96-A(6):1072–1080, 2013.

12 S. Kamei, A. Lamani, and F. Ooshita. Asynchronous Ring Gathering by Oblivious Robots with Limited Vision. In *WSSR*, pages 46–49, 2014.

13 S. Kamei, A. Lamani, F. Ooshita, S. Tixeuil, and K. Wada. Gathering on Rings for Myopic Asynchronous Robots with Lights. *arXiv:1911.04757*, 2019.

14 R. Klasing, A. Kosowski, and A. Navarra. Taking advantage of symmetries: Gathering of many asynchronous oblivious robots on a ring. *Theor. Comput. Sci.*, 411(34-36):3235–3246, 2010.

15 G. A. D. Luna and G. Viglietta. Robots with Lights. In *Distributed Computing by Mobile Entities, Current Research in Moving and Computing.*, volume 11340 of *LNCS*, pages 252–277. Springer, 2019.

16 S. Nagahama, F. Ooshita, and M. Inoue. Ring Exploration of Myopic Luminous Robots with Visibility More than One. In *SSS*, 2019.

17 F. Ooshita and S. Tixeuil. Ring Exploration with Myopic Luminous Robots. In *SSS*, pages 301–316, 2018.

18 I. Suzuki and M. Yamashita. Distributed Anonymous Mobile Robots: Formation of Geometric Patterns. *SIAM Journal on Computing*, 28(4):1347–1363, 1999.
19 G. Viglietta. Rendezvous of Two Robots with Visible Bits. In *ALGOSENSOR*, pages 291–306, 2013.