HAL open science

Asynchronous Gathering in a Torus

Sayaka Kamei, Anissa Lamani, Fukuhito Ooshita, Sébastien Tixeuil, Koichi Wada

To cite this version:

Sayaka Kamei, Anissa Lamani, Fukuhito Ooshita, Sébastien Tixeuil, Koichi Wada. Asynchronous Gathering in a Torus. OPODIS 2021 - 25th International Conference on Principles of Distributed Systems, Dec 2021, Strasbourg, France. pp.9:1–9:17, 10.4230/LIPIcs.OPODIS.2021.9 . hal-03592615

HAL Id: hal-03592615

https://hal.sorbonne-universite.fr/hal-03592615v1

Submitted on 1 Mar 2022

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Distributed under a Creative Commons Attribution 4.0 International License

Asynchronous Gathering in a Torus

Sayaka Kamei
Hiroshima University, Japan

Anissa Lamani
Strasbourg University, CNRS, ICUBE, France

Fukuhito Ooshita
Nara Institute of Science and Technology, Japan

Sébastien Tixeuil
Sorbonne University, CNRS, LIP6, France

Koichi Wada
Hosei University, Tokyo, Japan

Abstract

We consider the gathering problem for asynchronous and oblivious robots that cannot communicate explicitly with each other but are endowed with visibility sensors that allow them to see the positions of the other robots.

Most investigations on the gathering problem on the discrete universe are done on ring shaped networks due to the number of symmetric configurations. We extend in this paper the study of the gathering problem on torus shaped networks assuming robots endowed with local weak multiplicity detection. That is, robots cannot make the difference between nodes occupied by only one robot from those occupied by more than one robot unless it is their current node. Consequently, solutions based on creating a single multiplicity node as a landmark for the gathering cannot be used. We present in this paper a deterministic algorithm that solves the gathering problem starting from any rigid configuration on an asymmetric unoriented torus shaped network.

2012 ACM Subject Classification Theory of computation → Distributed algorithms

Keywords and phrases Autonomous distributed systems, Robots gathering, Torus

Digital Object Identifier 10.4230/LIPIcs.OPODIS.2021.9

Related Version *Full Version*: `https://arxiv.org/abs/2101.05421` [17]

Funding This work was partially funded by the ANR project ESTATE, ref. ANR-16-CE25-0009-03, the ANR project SAPPORO, ref. 2019-CE25-0005-1, JSPS KAKENHI No. 19K11828, 20H04140, 20K11685, and 21K11748, and by JST SICORP (Grant#JPMJSC1806).

1 Introduction

We consider autonomous robots [21] that are endowed with visibility sensors and motion actuators, yet are unable to communicate explicitly. They evolve in a discrete environment, i.e., their space is partitioned into a finite number of locations, conveniently represented by a graph, where the nodes represent the possible locations that a robot can be, and the edges denote the possibility for a robot to move from one location to another.

Those robots must collaborate to solve a collective task despite being limited to computing capabilities, inputs from the environment, etc. In particular, the robots we consider are anonymous, uniform, yet they can sense their environment and make decisions according to their own ego-centered view. In addition, they are oblivious, i.e., they do not remember their past actions. Robots operate in *cycles* that include three phases: *Look*, *Compute*, and *Move* (LCM for short). The Look phase takes a snapshot of the other robots' positions using a robot's visibility sensors. During the Compute phase, a robot computes a target destination

© Sayaka Kamei, Anissa Lamani, Fukuhito Ooshita, Sébastien Tixeuil, and Koichi Wada;
licensed under Creative Commons License CC-BY 4.0
25th International Conference on Principles of Distributed Systems (OPODIS 2021).
Editors: Quentin Bramas, Vincent Gramoli, and Alessia Milani; Article No. 9; pp. 9:1–9:17
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

based on its previous observation. The Move phase consists in moving toward the computed destination using motion actuators. Three execution models have been considered in the literature using LCM cycles, capturing the various degrees of synchrony between robots. According to current taxonomy [11], they are denoted as FSYNC, SSYNC, and ASYNC, from the stronger to the weaker. FSYNC stands for *fully synchronous*. In this model, all robots execute the LCM cycle synchronously and atomically. In the SSYNC (*semi-synchronous*) model, robots are asynchronously activated to perform cycles, yet at each activation, a robot executes one cycle atomically. With the weaker model, ASYNC (*asynchronous*), robots execute LCM in a completely independent manner. Of course, the ASYNC model is the most realistic.

In the context of robots evolving on graphs, the two benchmarking tasks are *exploration* [13] and *gathering* [4]. In this paper, we address the *gathering* problem, which requires that robots eventually all meet at a single node, not known beforehand, and terminate upon completion.

We focus on the case where the network is an *anonymous unoriented torus* (or simply *torus*, for short). The terms *anonymous* and *unoriented* mean that no robot has access to any kind of external information (*e.g.*, node identifiers, oracle, local edge labeling, etc.) allowing to identify nodes or to determine any (global or local) direction, such as North-South/East-West. Torus networks were investigated for the purpose of exploration by Devismes et al.[9].

1.1 Related Work

Mobile robot gathering on graphs was first considered for ring-shaped graphs. Klasing *et al.* [18], 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 more than one robot is on the node. Then, Izumi *et al.* [14] 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. This condition was slightly relaxed by Kamei et al. [15]. D'Angelo *et al.* [6] proposed unified ring gathering algorithms for most of the solvable initial configurations, using local-weak multiplicity detection. Overall, for rings, relatively few open cases remain [1], as algorithm synthesis was demonstrated feasible [19].

The case of gathering in tree-shaped networks was investigated by D'Angelo *et al.* [7] and by Di Stefano et al.[20]. Hypercubes were the focus of Bose at el. [2]. Complete and complete bipartite graphs were outlined by Cicerone et al. [5], and regular bipartite by Guilbault et al. [12]. Finite grids were studied by D'Angelo et al. [7], Das et al. [8], and Castenow et al. [3], while infinite grids were considered by Di Stefano et al. [20], and by Durjoy et al. [10]. Results on grids and infinite grids do not naturally extend to tori. On the one hand, the proof arguments for impossibility results on the grid can be extended for the torus, since their indistinguishability criterium remains valid. So, if a torus admits an edge symmetry (the robot positions are mirrored over an axial symmetry traversing an edge), is periodic (a non-trivial translation leaves the robot positions unchanged), or admits a rotation whose center is not a robot, the gathering is impossible on a torus. On the other hand, both the finite and the infinite grid allow algorithmic tricks to be implemented. For example, the finite grid has three classes of nodes: corners (of degree 2), borders (of degree 3), and inner nodes (of degree 4), and those three classes permit the robots to obtain some sense of direction. By

contrast, the infinite grid makes a difference between two locations: the inner space (the set of nodes within the convex hull formed by the robot positions) and the outer space (the rest of the infinite grid), which also give some sense of direction. Now, every node in a torus has degree 4, and no notion of inner/outer space can be defined. To our knowledge, torus-shaped networks were never considered before for the gathering problem. The previous work by Devismes et al [9] only considers the exploration task.

1.2 Our Contribution

We consider the problem of gathering on torus-shaped networks. In more detail, for initial configurations that are *rigid* (i.e. where each robot has a unique view of the configuration), we propose a distributed algorithm that gathers all robots to a single node, not known beforehand. We only make use of local-weak multiplicity detection: robots may only know whether at least one other robot is currently hosted at their hosting node but cannot know the exact number and are also unable to retrieve multiplicity information from other nodes. Furthermore, robots have no common notion of North and no common notion of handedness. Finally, robots operate in the most general and realistic ASYNC execution model.

2 Model

We consider a distributed system that consists of a collection of $\mathcal{K} \geq 3$ robots evolving on a non-oriented and anonymous (ℓ, L)-torus (or simply torus for short) of n nodes. Values ℓ and L are two integers such that $L < \ell$ and (definition borrowed from Devismes et al. [9]):

1. $n = \ell \times L$.
2. Let E be a finite set of edges. There exists an ordering $v_1, \ldots, v_n$ of the nodes of the torus such that $\forall i \in \{0, \ldots, n-1\}$:
   - if $i + \ell < n$, then $\{v_i, v_{(i+\ell)}\} \in E$, else $\{v_i, v_{(i+\ell) \bmod n}\} \in E$.
   - if $i + 1 \bmod \ell \neq 0$, then $\{v_i, v_{i+1}\} \in E$, else $\{v_i, v_{i-\ell+1}\} \in E$.

Given the previous ordering $v_0, \ldots, v_{n-1}$, for every $j \in \{0, \ldots, L-1\}$, the sequence $v_{j\times\ell}$, $v_{1+j\times\ell}, \ldots, v_{\ell-1+j\times\ell}$ is called an *ℓ-ring*. Similarly, for every $k \in \{0, \ldots, \ell-1\}$, the sequence v_k, $v_{k+\ell}$, $v_{k+2\times\ell}$, $\ldots, v_{k+(L-1)\times\ell}$ is called an *L-ring*.

On the torus operate $\mathcal{K} \geq 3$ identical robots, *i.e.,* they all execute the same algorithm using no local parameters, and one cannot distinguish them using their appearance. In addition, they are oblivious, *i.e.,* they cannot remember the operations performed before. No direct communication is allowed between robots; however, we assume that each robot is endowed with visibility sensors that allow him to see the position of the other robots on the torus. Robots operate in cycles that comprise three phases: *Look*, *Compute* and *Move*. During the first phase (Look), each robot takes a snapshot to see the positions of the other robots on the torus. In the second phase (Compute), they decide to either stay idle or move. In the case they decide to move, a neighboring destination is computed. Finally, in the last phase (Move), they move to the computed destination (if any).

At each instant, a subset of robots is activated for the execution by an external entity called *scheduler*. We assume that the scheduler is fair, *i.e.,* all robots are activated infinitely many times. The model considered in this paper is the asynchronous model (ASYNC), where the time between Look, Compute, and Move phases is finite but unbounded. We however assume that the move phase is instantaneous, so that when a robot performs a look operation, it sees all robots on nodes and none on edges. Still, even with instant moves, each robot may move according to an outdated view, *i.e.,* the robot takes a snapshot to see the positions of the other robots, but when it decides to move, some other robots may have moved already.

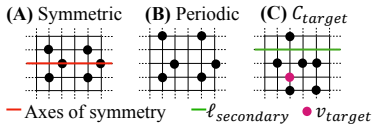

Figure 1 Instance of some defined configurations.

Let $\ell_0, \ell_1, \ldots, \ell_{L-1}$ be the sequence of ℓ-rings and let $v_{i,0}, v_{i,1}, \ldots, v_{i,\ell-1}$ be the sequence of nodes on ℓ_i (all operations on the indices are modulo ℓ for the nodes and modulo L for the ℓ-rings). By $\#r_{i,j}(t)$ we denote the number of robots on $v_{i,j}$ at time t. Node $v_{i,j}$ is empty if $\#r_{i,j}(t) = 0$. Otherwise, $v_{i,j}$ is occupied. In the case where $\#r_{i,j}(t) = 1$, we say that there is a single robot on $v_{i,j}$. By contrast, if $\#r_{i,j}(t) \geq 2$, we say that there is a multiplicity on $v_{i,j}$.

In this paper, we assume that robots have a local weak multiplicity detection *i.e.,* for any robot r, located at node u, r can only detect a multiplicity on its current node u (local). Moreover, r cannot be aware of the exact number of robots part of the multiplicity (weak).

During the process, some robots move and occupy some nodes of the torus, and their positions form the configuration of the system at that time. Initially, we assume that each node hosts at most one robot, *i.e.,* the initial configuration contains no multiplicities.

For each robot r, only a degraded vision of occupied locations is available. So, the local vision $d_{i,j}(t)$ of r about node $v_{i,j}$ at time t is 1 if $\#r_{i,j}(t) > 0$ and 0 otherwise.

For any $i, j \geq 0$, let $\delta^+_{i,j}(t)$ denote the sequence $< d_{i,j}(t), d_{i,j+1}(t), \ldots, d_{i,j+\ell-1}(t) >$, and let $\delta^-_{i,j}(t)$ denote the sequence $< d_{i,j}(t), d_{i,j-1}(t), \ldots, d_{i,j-(\ell-1)}(t) >$. Similarly, let $\Delta^{+s}_{i,j}(t)$ be the sequence $< \delta^s_{i,j}(t), \delta^s_{i+1,j}(t), \ldots, \delta^s_{i+(L-1),j}(t) >$ and $\Delta^{-s}_{i,j}(t)$ to be the sequence $< \delta^s_{i,j}(t), \delta^s_{i-1,j}(t), \ldots, \delta^s_{i-(L-1),j}(t) >$ with $s \in \{+, -\}$.

The view of a given robot r located on node $v_{i,j}$ at time t is defined as the pair $view_r(t) = (\mathcal{V}_{i,j}(t), m_j)$ where $\mathcal{V}_{i,j}(t)$ consists of the four sequences $\Delta^{++}_{i,j}, \Delta^{+-}_{i,j}, \Delta^{-+}_{i,j}, \Delta^{--}_{i,j}$ ordered in the lexicographical order and $m_j = 1$ if $v_{i,j}$ hosts a multiplicity and $m_j = 0$ otherwise.

By $view_r(t)(1)$, we refer to $\mathcal{V}_{i,j}(t)$ in $view_r(t)$. Given two robots r and r', we say that r has a larger view than r' at time t, denoted $view_r(t)(1) > view_{r'}(t)(1)$, if $view_r(t)$ is lexicographically larger than $view_{r'}(t)$. Similarly, r is said to have the largest view at time t, if for any robots $r' \neq r$, not located on the same node as r, $view_r(t)(1) > view_{r'}(t)(1)$ holds.

A configuration is said to be rigid at time t, if for any two robots r and r', located on two different nodes of the torus, $view_r(t)(1) \neq view_{r'}(t)(1)$ holds.

A configuration is said to be periodic at time t if there exist two integers i and j such that $i \neq j$, $i \neq 0 \mod \ell$, $j \neq 0 \mod L$, and for every robot $r_{(x,w)}$ located on ℓ_x at node $v_{x,w}$, $view_{r_{(x,w)}}(t)(1) = view_{r_{(x+i,w+j)}}(t)(1)$ (An example is given in Figure 1).

As defined by D'Angelo et al. [7], a configuration is said to be symmetric at time t, if the configuration is invariant after a reflection with respect to either a vertical or a horizontal axis. This axis is called the axis of symmetry (An example is given in Figure 1).

In this paper, we consider asymmetric (ℓ, L)-torus, *i.e.,* $\ell \neq L$. We assume *w.l.o.g.* that $L < \ell$. In this case, we can differentiate two sides of the torus. We denote by $nb_{\ell_i}(C)$ the number of occupied nodes on ℓ-ring ℓ_i, in configuration C. An ℓ-ring ℓ_i is said to be maximal in C if $\forall\ j \in \{0, \ldots, L-1\} \setminus \{i\}$, $nb_{\ell_j}(C) \leq nb_{\ell_i}(C)$.

Given a configuration C and two ℓ-rings ℓ_i and ℓ_j. We say that ℓ_j is adjacent to ℓ_i if $|i - j| = 1 \mod L$ holds. Similarly, we say that ℓ_j is neighbor of ℓ_i in configuration C if $nb_{\ell_j}(C) > 0$ and $nb_{\ell_k}(C) = 0$ for any $k \in \{i+1, i+2, \ldots, j-1\}$ or $k \in \{i-1, i-2, \ldots, j+1\}$. We also define $dis(x_i, x_j)$ to be a function which returns the shortest distance, in terms of hops, between x_i and x_j where x_i and x_j are two nodes of the torus. We sometimes write $x_i = r_i$ where r_i is a robot. In this case, x_i refers to the node that hosts r_i. Finally, we use

the notion of d.block to refer to a sequence of consecutive nodes in which there are occupied nodes each d hops (distance) with no other robot in between. The size of a d.block is the number of its occupied nodes.

Due to the lack of space, some details and proofs are omitted but can be found in [17].

3 Impossibility Results

This section presents impossibility results that motivate our settings.

Given a graph $G = (V, E)$ and a function $m : V \to \mathbb{N}$ associating the number of robots on a vertex v of V to v, (G, m) is a configuration whenever $\sum_{v \in V} m(v)$ is bounded and greater than zero. Let ϕ be a permutation of G's vertices that preserves its adjacency relation, so if $(u, v) \in E$, then $(\phi(u), \phi(v)) \in E'$, with $\phi(G) = (V', E')$. Note that ϕ always exists as the identity permutation fits this definition. Similarly, given a configuration (G, m), let ψ be a permutation of G's vertices that preserves its adjacency relation and such that for every node v of V, $m(v) = m(\psi(v))$. Again, ψ always exists as the identity permutation fits this definition. Given such a permutation ψ, the *cycle* C_ψ of order p that is generated by ψ is $\{\psi^0, \psi^1 = \psi, \psi^2 = \psi \circ \psi, \ldots, \psi^{p-1}\}$ such that $\psi^p = \psi^0$, where ϕ^0 is the identity. Note that C_ψ has order 1 if and only if ψ is the identity. Given a cycle C_ψ, the *orbit* of a vertex v of V is $C_\psi(v) = \{\gamma(v) | \gamma \in C_\psi\}$. Now, given a configuration $(G = (V, E), m)$, ψ is *partitive* if C_ψ has order $p > 1$, and for every $v \in V$, $|C_\psi(v)| = p$. That is, ψ is not reduced to the identity, and all nodes have the same orbit size. We now recall the Theorem of Di Stefano and Navarra for general topologies:

▶ **Theorem 1** ([20] Restated)**.** *If a configuration (G, m) admits a partitive permutation ψ, then (G, m) cannot be gathered.*

We specialize the general theorem to our setting:

▶ **Corollary 1.** *If a torus configuration is invariant by a non-empty series of non-null translations, a reflection through an edge-axis, or a non-empty series of non-null rotations whose center does* not *hold a robot; it is not gatherable.*

Next, we show that two robots cannot gather on a torus, even in FSYNC.

▶ **Theorem 2.** *Starting from a configuration with two robots a and b on different vertices in a torus with at least two vertices, gathering cannot occur, even in FSYNC.*

Finally, we show by induction that without multiplicity detection, the gathering is impossible.

▶ **Theorem 3.** *Starting from any configuration with $\mathcal{K} \geq 2$ robots with no multiplicity detection, gathering in a torus is impossible, even in SSYNC.*

4 Algorithm

When robots have only local weak multiplicity detection, multiplicities should be carefully created as the gathering becomes impossible from a configuration in which there are only two occupied nodes that both host a multiplicity. In ASYNC model, we need to be extra careful when it comes to robots with outdated views as they might create unwanted multiplicities (recall that when a robot moves the configuration might have changed as one or several robots might have moved once or many times).

Our strategy is to create a sense of direction on a torus to identify the gathering node and keep this node invariant, preventing the creation of unwanted multiplicities (if the configuration contains only two occupied nodes, one of these two nodes hosts for sure a single robot). For this purpose, robots proceed in two phases: first, they create the desired direction allowing them to identify a single node and then gather on the identified node. More precisely, let $\mathcal{C}_{target}$ be the set of configurations such that $C \in \mathcal{C}_{target}$ if the following properties are satisfied: C contains three ℓ-rings $\ell_{secondary}$, ℓ_{max} and ℓ_{target} such that:

(1) ℓ_{max} is the unique maximal ℓ-ring in C, (2) $\ell_{secondary}$ and ℓ_{target} are adjacent to ℓ_{max}. (3) $nb_{\ell_{secondary}}(C) = 0$, (4) ℓ_{target} satisfies exactly one of the following conditions:

- $nb_{\ell_{target}}(C) = 1$. We refer to the occupied node on ℓ_{target} by v_{target}.
- $nb_{\ell_{target}}(C) = 2$ and ℓ_{target} hosts a 2.block. We refer to the unique empty node in the 2.block by v_{target}.
- $nb_{\ell_{target}}(C) = 3$ and ℓ_{target} hosts a 1.block of size 3. By v_{target}, we refer to the occupied node in the middle of the 1.block.

From a configuration $C \in \mathcal{C}_{target}$, a direction can be identified: from v_{target} to ℓ_{max}. The idea is to make all robots neither on $\ell_{\max}$ nor on ℓ_{target} move to join v_{target} and then make the remaining robots gather on the node that is on $\ell_{\max}$ which is adjacent to v_{target}. To summarize, the proposed algorithm consists of two phases:

1. **Preparation Phase**. This phase starts from an arbitrary rigid configuration C_0 in which each node hosts at most one robot. Its aim is to reach a configuration $C \in \mathcal{C}_{target}$.
2. **Gathering Phase**. Starting from a configuration $C \in \mathcal{C}_{target}$, the gathering node is identified, and all robots eventually move to join it *i.e.,* the gathering is achieved.

Let us refer by $\mathcal{C}_{p_1}$ (respectively $\mathcal{C}_{p_2}$) to the set of configurations that appear during the Preparation (respectively the Gathering) phase. Let C be the current configuration, robots execute Protocol 1. Observe that $\mathcal{C}_{p_1} \cap \mathcal{C}_{p_2} = \emptyset$ and $\mathcal{C}_{target} \subset \mathcal{C}_{p_2}$.

Protocol 1 Main protocol.

```
if C ∈ C_{p_2} then
    Execute Gathering phase
else
    Execute Preparation phase
```

To ease the description of our strategy, we define predicates on a given configuration C:

- **Unique**(C): There exists a unique $i \in \{0, \ldots, L-1\}$ such that $\forall\, j \in \{0, \ldots, L-1\} \setminus \{i\}$, $nb_{\ell_j}(C) < nb_{\ell_i}(C)$.
- **Empty**(C): $(C \in \mathcal{C}_{target}) \;\wedge\; (\forall\, i \in \{0, \ldots, L-1\}$, such that $\ell_i \neq \ell_{target}$ and $\ell_i \neq \ell_{max}$, $nb_{\ell_i}(C) = 0)$.
- **Partial**(C): $(C \in \mathcal{C}_{target}) \;\wedge\; (\exists\, i \in \{0, \ldots, L-1\}$, such that $\ell_i \neq \ell_{target}$ and $\ell_i \neq \ell_{max}$, $nb_{\ell_i}(C) \neq 0)$.

Given a configuration C, Unique(C) indicates that C contains a unique maximal ℓ-ring. Empty(C) indicates that $C \in \mathcal{C}_{target}$ and all the ℓ-rings, except for ℓ_{max} and ℓ_{target}, are empty. By contrast, Partial(C) indicates that $C \in \mathcal{C}_{target}$ and there exists at least one ℓ-ring besides ℓ_{max} and ℓ_{target} that is occupied (hosts at least one occupied node).

In our algorithm, in several cases, robots in a single ℓ-ring, say ℓ_i, need to move and align themselves with respect to the positions of other robots which are on another ℓ-ring, say ℓ_k. To ease the description of the algorithm, we define a procedure referred to by **Align**(ℓ_i, ℓ_k) which makes the robots to perform such alignment *i.e.,* align robots on ℓ_i with respect to robots positions on ℓ_k. When the procedure is called in a configuration C, the following properties hold on both ℓ_i and ℓ_k:

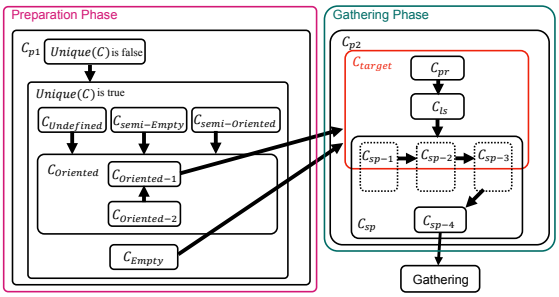

Figure 2 Transitions among all configurations.

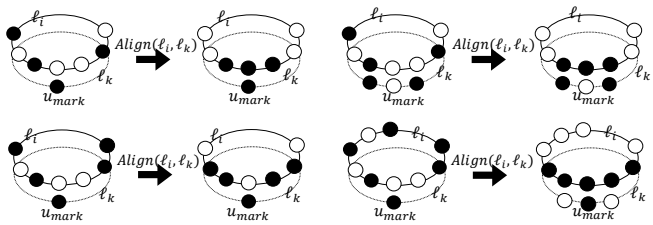

Figure 3 Some examples of Align(ℓ_i, ℓ_k).

1. $nb_{\ell_i}(C) = j$ with $j \in \{2, \ldots, 5\}$, *i.e.,* there are at least two and at most five robots on ℓ_i.
2. $nb_{\ell_i}(C) > nb_{\ell_k}(C)$ holds, and either (1) $nb_{\ell_k}(C) = 1$ or (2) $nb_{\ell_k}(C) = 2$ and ℓ_k contains a 2.block or (3) $nb_{\ell_k}(C) = 3$ and ℓ_k contains a 1.block of size 3.

Let u_{mark} be the node on ℓ_k that is: occupied if $nb_{\ell_k}(C) = 1$, empty in the 2.block if $nb_{\ell_k}(C) = 2$, occupied in the middle of the 1.block if $nb_{\ell_k}(C) = 3$. Node u_{mark} is used as a land mark to align robots on ℓ_i (a detailed description can be found in [17]). To give a better idea on the purpose of procedure **Align**, some examples are given in Figure 3. The procedure makes sure that if a multiplicity is created on ℓ_i then it is adjacent to u_{mark}. This allows the robots to keep track on multiplicities' positions and also make sure that the occupied nodes at a border of a 1.block on ℓ_i host only a single robot.

Figure 2 presents an overview of our strategy showing all the transitions among the different defined configurations in sections 4.1-4.2.

4.1 Preparation Phase

Let $C \in \mathcal{C}_{p1}$. The purpose of this phase is to reach a configuration $C' \in \mathcal{C}_{target}$ from C so that a direction is defined and the gathering node is identified. For this aim, robots first need to decrease the number of maximal ℓ-rings to reach a configuration C'' in which Unique(C'') is true. Then, from configuration C'', robots need to create both ℓ_{target} and $\ell_{secondary}$ to reach a configuration $C' \in C_{target}$. To prevent the creation of unwanted multiplicities due to robots with outdated views, most of the configurations in this phase are kept rigid.

First, let us address the case in which **Unique(C)** is false (C contains at least two maximal ℓ-rings). Robots need to decrease the number of maximal ℓ-rings to reach a configuration C' in which Unique(C') holds. Two cases are possible depending on whether there is an empty node on a maximal ℓ-ring: if a maximal ℓ-ring hosts at least one empty node then, the idea is to fill one of these empty nodes on a single maximal ℓ-rings. By contrast, if all the nodes of the maximal ℓ-rings are occupied, the idea is to create a single multiplicity on one of the maximal ℓ-rings to decrease their number gradually. Robots to move are chosen carefully

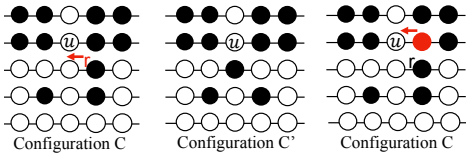

Figure 4 On the left, r is suppose to move but by moving, it creates a symmetric configuration C' shown in the middle. The robot on target-ℓ on the same L-ring as r moves to u.

in both cases, so that the configuration remains rigid. This is important to prevent having robots with outdated views. In the following, we refer to a maximal ℓ-ring by $\ell_{\max}$. Robots behavior in a configuration C in which Unique(C) holds is as follows:

1. If $nb_{\ell_{max}}(C) = \ell$ (all the nodes of $\ell_{\max}$ are occupied). Let $R_{\max}(C)$ be the set of robots on a maximal ℓ-ring. As C is rigid, all robots in $R_{\max}(C)$ have a unique view. Let ℓ_m be the maximal ℓ-ring in C that hosts the robot with the maximal view in $R_{\max}(C)$. One robot r is elected on ℓ_m to move. Its destination is one of its adjacent occupied nodes on ℓ_m. Robot r is selected as follows: Let $R_m(C) \subset R_{\max}(C)$ be the set of robots on ℓ_m which by moving to one of their adjacent occupied node on ℓ_m, the configuration reached remains rigid. Robot r is the robot in $R_m(C)$ which has the biggest view ($|R_m(C)| > 0$).
2. If $nb_{\ell_{max}}(C) < \ell$ (There is at least one empty node on $\ell_{\max}$), the idea is to fill exactly one of the empty nodes on exactly one of the maximal ℓ-ring. Let $R(C)$ be the set of robots closest to an empty node on a maximal ℓ-ring in C. Under some conditions, using the rigidity of C, one robot of $R(C)$, say r, is elected to move (the one with the largest view). Its destination is its adjacent empty node toward the closest empty node on a maximal ℓ-ring, say u, taking the shortest path. Among robots in the set $R(C)$, the one to move is the one that does not create a symmetric configuration. If no such robot exists in $R(C)$, some extra steps are taken beforehand to ensure that the configuration remains rigid. We discuss the various cases:
   - If C contains exactly two occupied ℓ-rings then, C contains only two maximal ℓ-rings. Robot r (the one to move) is the robot with the maximal view in C. Its destination is its adjacent empty node on an empty ℓ-ring (Note that this ℓ-ring exists since $L > 4$).
   - If C contains more than two occupied ℓ-rings then: let r be the robot in $R(C)$ with the largest view. By u and target-ℓ we refer to the closest empty node on a maximal ℓ-ring to r and the ℓ-ring including u. If by moving, r does not create a symmetric configuration, then r simply moves to its adjacent node toward u taking the shortest path. By contrast, if r creates a symmetric configuration by moving, then let C' be the configuration reached once r moves. Using configuration C' that each robot can compute without r moving, another robot r' in C is selected to move. We show later on that a symmetric configuration can only be reached when r either joins an empty node on the same L-ring as u for the first time or when it joins u. For the other cases, the configuration remains rigid. Hence, we only address the following two cases:
     a. Robot r joins an empty node on the same L-ring as u for the first time in C'. In this case, in C, the robot that is on target-ℓ being on the same L-ring as r moves to u (refer to Figure 4).
     b. Robot r joins u in C'. If in C' there are only two occupied ℓ-rings. The robot with the largest view which does not create a symmetric configuration is elected to move. Its destination is its adjacent empty node on an empty ℓ-ring. By contrast, if there are more than two occupied ℓ-rings in C' then robots proceed as follows:

- If the axis of symmetry lies on the unique ℓ_{max} in C' then, we are sure that there are two ℓ-rings which are maximal in C and that are symmetric with respect to the unique maximal ℓ-ring in C'. Let r be the robot located on a maximal ℓ-ring which is not on the axis of symmetry in C', that has the smallest view. Robot r is the one to move; its destination is its adjacent empty node on its ℓ-ring.
- If the axis of symmetry is perpendicular to the unique maximal ℓ-ring in C' then let T be the set of occupied ℓ-rings in C without target-ℓ. If there is an ℓ-ring in T which does not contain two 1.blocks separated by a single empty node on each side, then using the rigidity of C, a single robot on such an ℓ-ring which is the closest to the biggest 1.block is elected to move. Its destination is the closest 1.block. If there no such ℓ-ring in T (all ℓ-rings contains two 1.blocks separated by a unique empty node), then using the rigidity of C, one robot being on an ℓ-ring of T who has an empty node as a neighbor on its ℓ-ring is elected to move. Its destination is its adjacent empty node on its current ℓ-ring.

Note that we have only discussed the cases in which the reached configuration is either rigid or symmetric. This is because when r moves, it create neither a periodic nor an edge-edge symmetric configuration. This is mainly due to the fact that in C', there is a unique maximal ℓ-ring and C is assumed to be rigid.

We address now the case in which **Unique(C)** holds *i.e.,* C contains a unique maximal ℓ-ring, $\ell_{\max}$. To reach a configuration $C' \in \mathcal{C}_{target}$, robots need to move to build both $\ell_{secondary}$ and ℓ_{target} *i.e.,* one of the two adjacent ℓ-rings to ℓ_{max} needs to become empty while the other one needs to host either a single occupied node, a 2.block of size 2 or a 1.block of size 3. Let ℓ_i and ℓ_k be the two adjacent ℓ-rings to $\ell_{\max}$. Assume *w.l.o.g.* that $nb_{\ell_i}(C) \leq nb_{\ell_k}(C)$. To ease the description of this phase, we distinguish five main cases describing the possible states of ℓ_i and ℓ_k: (i) the case in which both ℓ_i and ℓ_k are empty ($C \in \mathcal{C}_{Empty}$). The idea, in this case, is to elect a single robot to join either ℓ_i or ℓ_k. (ii) the case in which ℓ_i is empty and ℓ_k hosts more than one occupied node ($C \in \mathcal{C}_{Semi-Empty}$). The idea is to make the robots on ℓ_k gather in a single node. Note that in both cases (i) and (ii), a configuration $C' \in \mathcal{C}_{target}$ is created. (iii) the case in which ℓ_i hosts a single occupied node while ℓ_k hosts at least two robots ($C \in \mathcal{C}_{Oriented}$). The unique occupied node on ℓ_i is used as a landmark to make robots on ℓ_k move and create either a 2.block of size 2 or a 1.block of size 3 ($C \in \mathcal{C}_{Oriented-2}$). Once such a block is created ($C \in \mathcal{C}_{Oriented-1}$), it is easy to free ℓ_i as the robots move to their adjacent node on ℓ_{max} (since the configuration reached $C' \in \mathcal{C}_{target}$, the multiplicity created on $\ell_{\max}$ can be identified as it is adjacent to v_{target}). (iv) the case in which both ℓ_i and ℓ_k host a unique occupied node ($C \in \mathcal{C}_{Semi-Oriented}$). The idea is to add a single robot to either ℓ_i or ℓ_k. Finally, (v) the case in which both ℓ_i and ℓ_k host more than one robot ($\mathcal{C}_{Undefined}$). The idea is to make robots elect either ℓ_i or ℓ_k and then make the robots on the elected ring gather on a single node. Both cases (iv) and (v) aim at reaching a configuration in $\mathcal{C}_{Oriented}$. More formally:

1. Set $\mathcal{C}_{Empty}$: $C \in \mathcal{C}_{Empty}$ if $nb_{\ell_i}(C) = nb_{\ell_k}(C) = 0$.
2. Set $\mathcal{C}_{Semi-Empty}$: $C \in \mathcal{C}_{Semi-Empty}$ if *w.l.o.g.* $nb_{\ell_i}(C) = 0$ and $nb_{\ell_k}(C) > 1$.
3. Set $\mathcal{C}_{Oriented}$: $C \in \mathcal{C}_{Oriented}$ if *w.l.o.g.* $nb_{\ell_i}(C) = 1$ and $nb_{\ell_k}(C) > 1$. Set $\mathcal{C}_{Oriented}$ includes:
   a. $\mathcal{C}_{Oriented-1}$. In this case either (i) $nb_{\ell_k}(C) = 3$ and ℓ_k contains a 1.block of size 3 whose middle robot is on the same L-ring as the unique occupied node on ℓ_i. (ii) $nb_{\ell_k}(C) = 2$ and ℓ_k contains a 2.block. Moreover, the unique empty node in the 2.block is on the same L-ring as the unique robot on ℓ_i.

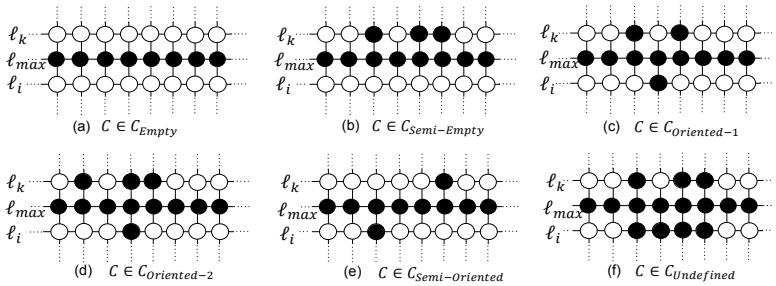

Figure 5 Instance of configurations C when Unique(C) is true.

b. $\mathcal{C}_{Oriented-2}$. Contains all the configuration in $\mathcal{C}_{oriented}$ that are not in $\mathcal{C}_{Oriented-1}$. That is, $\mathcal{C}_{Oriented-2} = \mathcal{C}_{Oriented} - \mathcal{C}_{Oriented-1}$.

4. Set $\mathcal{C}_{Semi-Oriented}$: $C \in \mathcal{C}_{Semi-Oriented}$ if *w.l.o.g.* $nb_{\ell_i}(C) = 1$ and $nb_{\ell_k}(C) = 1$.

5. Set $\mathcal{C}_{Undefined}$: $C \in \mathcal{C}_{Undefined}$ if $nb_{\ell_i}(C) > 1$ and $nb_{\ell_k}(C) > 1$.

Figure 5 presents instances of configurations in which there is a unique maximal ℓ-ring.

The behavior of the robots in each set of configurations is as follows:

1. $C \in \mathcal{C}_{Empty}$. Let ℓ_{n_i} and ℓ_{n_k} be the two neighboring ℓ-rings of ℓ_{max} (one neighboring ℓ-ring from each direction). In the case in which $\ell_{n_i} = \ell_{n_k} = \ell_{max}$ (C contains a single occupied ℓ-ring) then, using the rigidity of C, one robot from C is selected to move to its adjacent empty node outside its ℓ-ring (the scheduler chooses the direction to take). Otherwise, let R_m be the set of robots which are the closest to either ℓ_i or ℓ_k. If $|R_m| = 1$ then, the unique robot in R_m, referred to by r, is the one allowed to move. Assume *w.l.o.g.* that r is the closest to ℓ_i. The destination of r is its adjacent empty node outside its current ℓ-ring on the shortest empty path toward ℓ_i. If r is the closest to both ℓ_i and ℓ_k then the scheduler chooses the direction to take (it moves either toward ℓ_i or ℓ_k). In the case where $|R_m| > 1$ (R_m contains more than one robot) then, by using the rigidity of C, one robot r is selected. Its behavior is the same as r in the case where $|R_m| = 1$.

2. $C \in \mathcal{C}_{Semi-Empty}$. Assume *w.l.o.g.* $nb_{\ell_k}(C) > 1$ and $nb_{\ell_i}(C) = 0$. We consider two cases:

a. $nb_{\ell_k}(C) > 3$ or $nb_{\ell_k}(C) = 2$. Recall that $C \notin \mathcal{C}_{target}$. Let $\uparrow$ be the direction defined from ℓ_{max} to ℓ_k taking the shortest path and let ℓ_n be the ℓ-ring that is neighbor of ℓ_i. Observe that $\ell_n = \ell_k$ is possible (if only two ℓ-rings are occupied in C). Using the rigidity of configuration C, one robot from ℓ_n is elected to move. Its destination is its adjacent node outside ℓ_n and towards ℓ_i with respect to the direction $\uparrow$.

b. $nb_{\ell_k}(C) = 3$. Again, recall that $C \notin \mathcal{C}_{target}$. The aim is to make the three robots form a single 1.block. To this end, if the configuration contains a single d.block of size 3 with $d > 1$ then the robot in the middle of the d.block moves to its adjacent node on ℓ_k (the scheduler chooses the direction to take). By contrast, if the configuration contains a single d.block of size 2 ($d \geq 1$) then the robot not part of the d.block moves towards its adjacent empty node towards the d.block taking the shortest empty path.

3. $C \in \mathcal{C}_{Oriented}$. Let r_i be the single robot on ℓ_i.

a. $C \in \mathcal{C}_{Oriented-1}$. If $nb_{max}(C) > 4$ then the unique robot on ℓ_i moves to its adjacent node on ℓ_{max}. Otherwise, let u be the node on ℓ_{max} adjacent to a robot on ℓ_i.

- If $nb_{max}(C) = 3$ and the robots form a 1.block of size 3 whose middle robot is adjacent to u then the unique robot on ℓ_i moves to its adjacent node on ℓ_{max}. Otherwise, robots on ℓ_{max} execute **Align**(ℓ_{max}, ℓ_i).
- If $nb_{max}(C) = 4$ and u is empty, then the unique robot on ℓ_i moves to u. Otherwise (u is occupied), then let r be the robot on u.

- If r has an adjacent empty node on ℓ_{max} then r moves to one of its adjacent nodes (the scheduler chooses the node to move to in case of symmetry).
- If r does not have an adjacent empty node on ℓ_{max}, then let r' be the robot on ℓ_{max} which is adjacent to r and which does not have a neighboring robot on ℓ_{max} at distance $\lfloor \ell/2 \rfloor$. Robot r' moves to its adjacent empty node on ℓ_{max}.

b. $C \in \mathcal{C}_{Oriented-2}$. If $nb_{\ell_k}(C) = 2$ or $nb_{\ell_k}(C) = 3$ then **Align**$(\ell_k,\ \ell_i)$ is executed. Otherwise, if $nb_{\ell_k}(C) > 3$ then, $nb_{\ell_k}(C) - 2$ robots gather on the node u_k located on ℓ_k and which is on the same L-ring as the unique occupied node on ℓ_i. For this purpose, the robot on ℓ_k which is the closest to u_k with the largest view is the one allowed to move. Its destination is its adjacent node on ℓ_k toward u_k.

4. $C \in \mathcal{C}_{Semi-Oriented}$. Let ℓ_{n_i} and ℓ_{n_k} be the two neighboring ℓ-rings of ℓ_i and ℓ_k respectively. First, if *w.l.o.g.* $\ell_i = \ell_{n_k}$ ($\ell_k = \ell_{n_i}$) then, C contains only 3 occupied ℓ-rings ℓ_i, ℓ_{max} and ℓ_j. Using the rigidity of C, one robot from either ℓ_{n_i} or ℓ_{n_k} (not both) is selected to move. Its destination is its adjacent empty node outside its current ℓ-ring in the opposite direction of ℓ_{max}. Next, if $\ell_i \neq \ell_{n_k}$ ($\ell_k \neq \ell_{n_i}$) then, using the rigidity of C, a unique robot is selected to move from either ℓ_{n_i} or ℓ_{n_k} (not both). Its destination is its adjacent empty node outside its current ℓ-ring toward ℓ_i (respectively ℓ_k) if the robot was elected from ℓ_{n_i} (respectively ℓ_{n_k}). If $\ell_{n_i} = \ell_{n_k}$, the scheduler chooses the direction to take.

5. $C \in \mathcal{C}_{Undefined}$. Depending on the number of robots on ℓ_i and ℓ_k, we consider two cases::

a. $nb_{\ell_i}(C) < nb_{\ell_k}(C)$. The idea is to make robots on ℓ_i gather on ℓ_i. We define a configuration, denoted $\Gamma(C)$, built from C ignoring some ℓ-rings that will be used to identify a single node on ℓ_i on which all robots on ℓ_i will gather. If there are at least four occupied ℓ-rings in C then $\Gamma(C)$ is the configuration built from C ignoring both ℓ_i and ℓ_k. By contrast, if there are only three occupied ℓ-rings then $\Gamma(C)$ is the configuration built from C ignoring only ℓ_i. The following cases are possible:

i. Configuration $\Gamma(C)$ is rigid. Using the rigidity of $\Gamma(C)$, one node on ℓ_i, say u, is elected as the gathering node. Robots on ℓ_i move in turn to the elected node.

ii. Configuration $\Gamma(C)$ has exactly one axis of symmetry. The axis of symmetry of $\Gamma(C)$ either intersects with ℓ_i on a single node (edge-node symmetric), or on two nodes (node-node symmetric) or only on edges (edge-edge symmetric):

- $\Gamma(C)$ is node-edge symmetric: The node on ℓ_i that is on the axis of symmetry of $\Gamma(C)$ is the gathering node. Robots on ℓ_i move in turn to join it.

- $\Gamma(C)$ is node-node symmetric: Let u_1 and u_2 be the two nodes on ℓ_i on which the axis of symmetry passes through. If both nodes are occupied, then using the rigidity of C, exactly one of the two nodes is elected. Assume *w.l.o.g.* that u_1 is elected. Robots on u_1 move to their adjacent node. If both u_1 and u_2 are empty then let R be the set of robots on ℓ_i that are at the smallest distance from either u_1 or u_2. If $|R| = 1$ (Let $r \in R$ and assume *w.l.o.g.* that r is the closest to u_1) then, r moves on ℓ_i toward u_1 taking the shortest path. By contrast, if $|R| > 1$ then using the rigidity of C, exactly one robot of R is elected to move. The elected robot moves on ℓ_i toward the closest node among u_1 and u_2 taking the shortest path.

- $\Gamma(C)$ is edge-edge symmetric: assume *w.l.o.g.* that $\Gamma(C)$'s axis of symmetry of passes through ℓ_i on the two edges $e_1 = (u_1, u_2)$ and $e_1 = (u_3, u_4)$ with u_1 and u_3 being on the same side. Let $U = \{u_j,\ j \in [1-4]\}$. We consider the following cases:
  - For all $u \in U$, u is occupied. Using the rigidity of C, a single node $u \in U$ is elected. Robots on u move to their adjacent node $u' \in U$ (refer to Figure 6, (A)).
  - Three nodes of U are occupied. Assume *w.l.o.g.* that $u_1 \in U$ is the one empty. If there are robots on ℓ_i which are located on the same side as u_1 and u_3 with respect to $\Gamma(C)$'s axis of symmetry then, the robots among these which are the

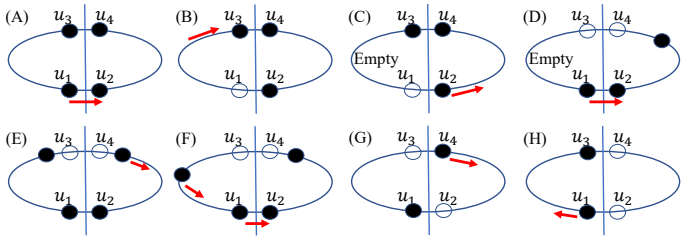

Figure 6 Case in which $\Gamma(C)$ is edge-edge symmetric.

closest to u_3 move to their adjacent node on ℓ_i toward u_3 taking the shortest path (refer to Figure 6, (B)). By contrast, if there are no robots on ℓ_i which are on the same side of u_1 and u_3 then, robots on u_2 move to their adjacent node in the opposite direction of u_1 (refer to Figure 6, (C)).

- Two nodes of U are occupied. First, assume *w.l.o.g.* that u_1 and u_2 are occupied (the case in which the two nodes are neighbors). If all robots on ℓ_i are on the same side of the axis of symmetry (assume *w.l.o.g.* that they are at the same side as u_2). Robots on u_1 are the ones to move. Their destination is u_2 (refer to Figure 6, (D)). By contrast, if there are robots on both sides of $\Gamma(C)$'s axis of symmetry then, let U' be the set of occupied nodes on ℓ_i which are the farthest from the occupied node of U which is on the side (of the axis of symmetry). If there are two such nodes (one at each side), as C is rigid, the scheduler elects exactly one of these two nodes. Let us refer to the elected node by u. Robots on u are the ones to move. Their destination is their adjacent node on ℓ_i towards the occupied node of U being on their side (refer to Figure 6, (E)). By contrast, if there is only one node in U' then, robots on the other side of the axis of symmetry are the ones to move to start from the robots that are the closest to the occupied node of U being on their side. Their destination is their adjacent ℓ_i toward the occupied node of U on their side (refer to Figure 6, (F)). Finally, if there are no robots on both sides of the axis of symmetry, then using the rigidity of C, one occupied node of U is elected. Robots on the elected node are the ones to move. Their destination is their adjacent occupied node in U.
  Next, assume *w.l.o.g.* that u_1 and u_3 are occupied (the case in which the two nodes of U are not neighbors but are at the same side of $\Gamma(C)$'s axis of symmetry). Robots on a node of U with the largest view are the ones to move. Their destination is their adjacent node in the opposite direction of a node of U (refer to Figure 6, (H)). Finally, assume *w.l.o.g.* that u_1 and u_4 are occupied (the case in which the two nodes of U are not neighbors and are in opposite sides of $\Gamma(C)$ axis of symmetry). Robots on a node of U with the largest view are the ones to move. Their destination is their adjacent node on ℓ_i, in the opposite direction of their adjacent node in U (refer to Figure 6, (G)).
- There is only one node of U that is occupied. Assume *w.l.o.g.* that u_1 is occupied. If all robots on ℓ_i are on the same side as u_1 with respect to $\Gamma(C)$'s axis of symmetry then, the closest robot to u_1 on ℓ_i are the ones to move. Its destination is its adjacent node towards u_1 taking the shortest path. By contrast, if all robots on ℓ_i are in the opposite side of the axis of symmetry of u_1 then robots on u_1 are the ones to move. Their destination is u_2. Finally, if robots on ℓ_i are on both

sides of the axis of symmetry then the closest robot to u_1 being on the same side of $\Gamma(C)$'s axis of symmetry as u_1 are the ones to move. Their destination is their adjacent node on ℓ_i towards u_1 taking the shortest path.
- All nodes of U are empty. Let d be the smallest distance between a node of $u \in U$ and a robot on the same side of $\Gamma(C)$'s axis of symmetry as u. Let R be the set of robots at distance d from a node $u \in U$. If $|R| = 1$ then the robot in R moves towards the closest node $u \in U$. By contrast, if $|R| > 1$ then, using the rigidity of C, a unique robot in R is selected to move. Its destination is its adjacent node on ℓ_i toward the closest node $u \in U$.

iii. Configuration $\Gamma(C)$ has more than one axis of symmetry. Using the rigidity of C, a single robot from $\Gamma(C)$ is elected to move. Its destination is its adjacent empty node on its current ℓ-ring. This reduces the number of axis of symmetries to either 1 or 0.

b. $nb_{\ell_i}(C) = nb_{\ell_k}(C)$. The strategy is similar to the one used in the case in which $nb_{\ell_i}(C) \neq nb_{\ell_k}(C)$. That is, by using the state of configuration $\Gamma(C)$, robots on either ℓ_i or ℓ_k gather in a single node. The difference in this case is that the robots need to elect either ℓ_i or ℓ_k. The detailed description of this case can be found in [17].

▶ **Lemma 1.** *From any initial rigid configuration $C_0 \in \mathcal{C}_{p_1}$, a configuration $C' \in \mathcal{C}_{target}$ which does not contain any robot with an outdated view, is eventually reached. Moreover, the unique maximal ℓ-ring in C' hosts at most one multiplicity node. This node (if any) is adjacent to v_{target}.*

4.2 Gathering Phase

This phase starts from a configuration $C \in \mathcal{C}_{target}$ in which a direction is defined in C (from ℓ_{target} to ℓ_{max}). The idea is to make all robots that are neither on ℓ_{target} nor ℓ_{max} move to join v_{target}. Then, make some robots on ℓ_{max} move to join v_{target} while the other align themselves with respect to v_{target} to finally gather all on the node of $\ell_{\max}$ adjacent to v_{target}. To ease the description of our algorithm, we define the following set of configurations:

1. Set C_{sp} which includes the following four sub-sets:

a. SubSet $\mathcal{C}_{sp-1}$: $C \in C_{sp-1}$ if there are exactly two occupied ℓ-rings in C denoted ℓ_i and ℓ_j respectively on which the following conditions hold: (1) ℓ_i and ℓ_j are adjacent. (2) $nb_{\ell_j}(C) < nb_{\ell_i}(C)$ (3) either :
- $nb_{\ell_i}(C) = 4$ and ℓ_i contains two 1.blocks of size 2 being at distance 2 from each other. Let u be the unique node between the two 1.blocks on ℓ_i.
- $nb_{\ell_i}(C) = 3$ or 5 and ℓ_i contains a 1.block of size $nb_{\ell_i}(C)$. Let u be the middle node of the 1.block of size $nb_{\ell_i}(C)$.

(4) Either $nb_{\ell_j}(C) = 3$ and ℓ_j contains a 1.block of size 3 whose middle node is adjacent to u or $nb_{\ell_j}(C) = 2$ and ℓ_j contains either a 2.block of size 2 whose middle node is adjacent to u or a 1.block of size 2 having one extremity adjacent to u (refer to Figure 7 for some examples).

b. SubSet $\mathcal{C}_{sp-2}$: $C \in \mathcal{C}_{sp-2}$ if $C \in \mathcal{C}_{target}$ and $nb_{\ell_{target}}(C) = 1$. In addition either one of the following conditions are verified: (1) $nb_{\ell_{max}}(C) = 4$ and on ℓ_{max} there are two 1.blocks of size 2 being at distance 2 from each other. Let u be the unique node between the two 1.blocks then u is adjacent to v_{target}. (2) $nb_{\ell_{max}} = 5$ and on ℓ_{max} there is a 1.block of size 5 whose middle robot is adjacent to v_{target}. (3) $nb_{\ell_{max}}(C) = 4$ and on ℓ_{max} there is a 1.block of size 3 having a unique occupied node at distance 2. Let u be the unique empty node between the 1.block of size 3 and the 1.block of size 1. Then u is adjacent to v_{target} (refer to Figure 7).

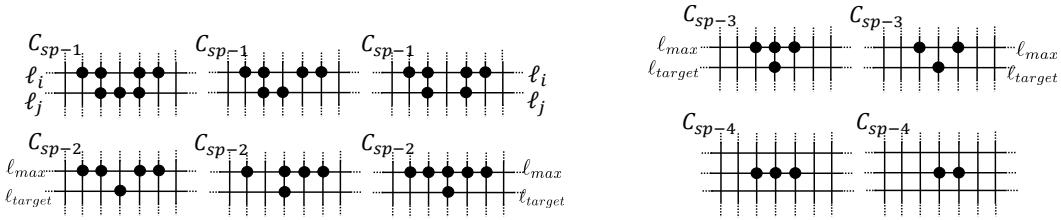

Figure 7 Set C_{sp}.

- **c.** SubSet $\mathcal{C}_{sp-3}$: $C \in \mathcal{C}_{sp-3}$ if $C \in \mathcal{C}_{target}$, $Empty(C)$ is true, $nb_{\ell_{target}}(C) = 1$ and one of the two following conditions holds: (1) $nb_{\ell_{max}}(C) = 3$ and ℓ_{max} contains an 1.block of size 3 whose middle robot is adjacent to v_{target}. (2) $nb_{\ell_{max}}(C) = 2$ and the two robots form a 2.block on ℓ_{max}. Let u be the unique empty node between the two robots on ℓ_{max}, then u is adjacent to v_{target} (refer to Figure 7).
- **d.** SubSet $\mathcal{C}_{sp-4}$: $C \in C_{sp-4}$ if there is a unique ℓ-ring that is occupied and on this ℓ-ring there are either two or three occupied nodes that form a 1.block (refer to Figure 7).

2. Set $\mathcal{C}_{pr}$: $C \in \mathcal{C}_{pr}$ if $C \in \mathcal{C}_{target}$ and $Partial(C)$ is true. That is, $\exists\ i \in \{0, \ldots, L-1\}$ such that $\ell_i \neq \ell_{max}$ and $\ell_i \neq \ell_{target}$ and $nb_{\ell_i}(C) > 0$. Note that we are sure that $C \notin \mathcal{C}_{sp}$.
3. Set $\mathcal{C}_{ls}$: $C \in \mathcal{C}_{ls}$ if $C \in \mathcal{C}_{target}$ and $C \notin \mathcal{C}_{sp}$ and $Empty(C)$. In other words, there are only two ℓ-rings that are occupied: ℓ_{max} and ℓ_{target}.

We now present the behavior of robots during the gathering phase. If the current configuration $C \in \mathcal{C}_{target}$, then we define $\uparrow$ as the direction from ℓ_{target} to ℓ_{max} taking the shortest path. Observe that $\uparrow$ can be computed by all robots and $\uparrow$ is unique (recall that ℓ_{max} is unique, and $\forall\ C \in \mathcal{C}_{target}$, $nb_{\ell_{target}}(C) \neq nb_{\ell_{secondary}}(C)$). Using $\uparrow$, we define a total order on the ℓ-rings of the torus such that $\ell_i \leq \ell_j$ if ℓ_i is not further from ℓ_{target} than ℓ_j with respect to $\uparrow$. Note that $\mathcal{C}_{p_2} = \mathcal{C}_{pr} \cup \mathcal{C}_{ls} \cup \mathcal{C}_{sp}$. Let C be the current configuration, robots behavior for each defined set is as follows:

1. $C \in \mathcal{C}_{pr}$. Let us refer by ℓ_i to the ℓ-ring that is adjacent to ℓ_{target} such that $\ell_i \neq \ell_{max}$. Depending on the number of robots on ℓ_i, two cases are possible:
   - **a.** $nb_{\ell_i}(C) > 0$. Let R_m be the set of robots on ℓ_i that are the closest to v_{target}, if
     - **i.** there is an occupied node u_i on ℓ_i that is adjacent to v_{target}, then robots on u_i are the ones to move. Their destination is v_{target}.
     - **ii.** there is no robot on ℓ_i that is adjacent to v_{target} and $nb_{\ell_i}(C) < \ell - 1$, then robots in R_m are the ones to move. Their destination is their adjacent empty node on ℓ_i on the empty path toward v_{target}.
     - **iii.** there is no robot on ℓ_i adjacent to v_{target} and $nb_{\ell_i}(C) = \ell - 1$, then let $R_{m'}$ be the set of robots that share a hole with u_i, where u_i is the node on ℓ_i that is adjacent to v_{target}. Robots in $R_{m'}$ are allowed to move only if they are not part of a multiplicity location. Their destination is the node towards u_i on the empty path.
   - **b.** $nb_{\ell_i}(C) = 0$. Let ℓ_k be the closest neighboring ℓ-ring to ℓ_{target} with respect to $\uparrow$. Let R_m be the set of robots on ℓ_k that are closest to v_{target}. Robots on R_m are the ones to move, their destination is the node outside ℓ_k and toward ℓ_{target} with respect to $\uparrow$.
2. $C \in \mathcal{C}_{ls}$. Robots aims at reaching a configuration $C' \in \mathcal{C}_{sp}$. If $nb_{\ell_{max}}(C) \leq 5$, robots on ℓ_{max} execute **Align**$(\ell_{max}, \ell_{target})$. Otherwise, robots behave as follows: Let u_1, u_2, u_3, u_4 and u_5 be a sequence of five consecutive nodes on ℓ_{max} such that u_3 is adjacent to v_{target}. If u_3 is occupied and has exactly one adjacent occupied node on ℓ_{max} (assume *w.l.o.g.* that this node is u_2) then the robot on u_2 is the one to move. Its destination is u_3. By contrast, if u_3 has either no adjacent occupied nodes on ℓ_{max}, or two adjacent occupied

nodes on ℓ_{max}, then robots on u_3 move to v_{target}. Finally, if u_3 is empty then let R be the set of robots that are closest to u_3 on ℓ_{max}. If $|R| = 2$ then both robots move to their adjacent node on ℓ_{max} toward u_3. By contrast, if $|R| = 1$, then first assume that the distance between the robot in R and u_3 is d. If there is a robot r_m on ℓ_{max} that shares a hole with u_3 and at distance $d+1$ from u_3, then r_m moves towards u_3 taking the shortest path. If no such robot exists, the robot in R moves toward u_3 taking the shortest path.

3. $C \in \mathcal{C}_{sp}$. We distinguish:
   a. $C \in \mathcal{C}_{sp-1}$. If $C \in \mathcal{C}_{target}$, then the robots on ℓ_{target} that are at the extremities of the 1.block or the 2.block move to their adjacent occupied node on ℓ_{max}. By contrast, if $C \notin \mathcal{C}_{target}$, then the robot not on ℓ_{max} that has two adjacent occupied nodes moves to its adjacent node on ℓ_{max}.
   b. $C \in \mathcal{C}_{sp-2}$. If there is a 1.block of size 3 on ℓ_{max} then the robots that are in the middle of the 1.block of size 3 move to their adjacent occupied node that has one robot at distance 2. If ℓ_{max} contains a 1.block of size 5 then the robots on ℓ_{max} that are adjacent of the extremities of the 1.block move on ℓ_{max} in the opposite direction of the extremities of the 1.block. Finally, if ℓ_{max} contains two 1.blocks of size 2 then the robots that share a hole of size 1 move toward each other.
   c. $C \in \mathcal{C}_{sp-3}$. Robots on v_{target} move to their adjacent node on ℓ_{max} (note that v_{target} can be occupied by either a single robot or a multiplicity).
   d. $C \in \mathcal{C}_{sp-4}$. If C contains a 1.block of size 3 then the robots at the extremities of the 1.block move to their adjacent occupied node. By contrast, if C contains a 1.block of size 2 then the robot that is not part of a multiplicity moves to its adjacent occupied node (it will be shown that one of the occupied nodes hosts only one robot).

We now state our main positive result.

▶ **Theorem 4.** *Assuming an (ℓ, L)-torus in which $L < \ell$ and $L > 4$ and starting from an arbitrary rigid configuration, Protocol 1 solves the gathering problem for any $\mathcal{K} \geq 3$.*

5 Concluding Remarks

We presented the first algorithm for gathering asynchronous oblivious mobile robots in a fully asynchronous model in a torus-shaped space graph. Our work raises several open questions:

1. What is the exact set of initial configurations that are gatherable? Our work considers initial rigid configurations only, and we know that periodic, edge-symmetric, and invariant through rotation (with no center robot) configurations make the problem impossible to solve. As in the case of the ring, special classes of non-rigid configuration may exist that are still gatherable.
2. The case of a square torus is intriguing: the robots would loose the ability to distinguish between the big side and the small side of the torus, so additional constraints are likely to hold if gathering remains feasible.
3. Following recent work by Kamei et al. [16] on the ring, it would be interesting to consider myopic (i.e. robot whose visibility radius is limited) yet luminous (i.e. robots that maintain a constant size state that can be communicated to other robots in the visibility range) robots in a torus.

References

1 François Bonnet, Maria Potop-Butucaru, and Sébastien Tixeuil. Asynchronous gathering in rings with 4 robots. In *Ad-hoc, Mobile, and Wireless Networks - 15th International Conference, ADHOC-NOW 2016, Lille, France, July 4-6, 2016, Proceedings*, volume 9724 of *Lecture Notes in Computer Science*, pages 311–324. Springer, 2016. doi:10.1007/978-3-319-40509-4_22.

2 Kaustav Bose, Manash Kumar Kundu, Ranendu Adhikary, and Buddhadeb Sau. Optimal gathering by asynchronous oblivious robots in hypercubes. In *Algorithms for Sensor Systems - 14th International Symposium on Algorithms and Experiments for Wireless Sensor Networks, ALGOSENSORS 2018, Helsinki, Finland, August 23-24, 2018, Revised Selected Papers*, volume 11410 of *Lecture Notes in Computer Science*, pages 102–117. Springer, 2018. doi:10.1007/978-3-030-14094-6_7.

3 Jannik Castenow, Matthias Fischer, Jonas Harbig, Daniel Jung, and Friedhelm Meyer auf der Heide. Gathering anonymous, oblivious robots on a grid. *Theoretical Computer Science*, 815:289–309, 2020. doi:10.1016/j.tcs.2020.02.018.

4 Serafino Cicerone, Gabriele Di Stefano, and Alfredo Navarra. Asynchronous robots on graphs: Gathering. In *Distributed Computing by Mobile Entities, Current Research in Moving and Computing*, volume 11340 of *Lecture Notes in Computer Science*, pages 184–217. Springer, 2019. doi:10.1007/978-3-030-11072-7_8.

5 Serafino Cicerone, Gabriele Di Stefano, and Alfredo Navarra. Gathering robots in graphs: The central role of synchronicity. *Theoretical Computer Science*, 849:99–120, 2021. doi:10.1016/j.tcs.2020.10.011.

6 Gianlorenzo D'Angelo, Alfredo Navarra, and Nicolas Nisse. A unified approach for gathering and exclusive searching on rings under weak assumptions. *Distributed Computing*, 30(1):17–48, 2017. doi:10.1007/s00446-016-0274-y.

7 Gianlorenzo D'Angelo, Gabriele Di Stefano, Ralf Klasing, and Alfredo Navarra. Gathering of robots on anonymous grids and trees without multiplicity detection. *Theoretical Computer Science*, 610:158–168, 2016. doi:10.1016/j.tcs.2014.06.045.

8 Shantanu Das, Nikos Giachoudis, Flaminia L. Luccio, and Euripides Markou. Gathering of robots in a grid with mobile faults. In *SOFSEM 2019: Theory and Practice of Computer Science - 45th International Conference on Current Trends in Theory and Practice of Computer Science, Nový Smokovec, Slovakia, January 27-30, 2019, Proceedings*, volume 11376 of *Lecture Notes in Computer Science*, pages 164–178. Springer, 2019. doi:10.1007/978-3-030-10801-4_14.

9 Stéphane Devismes, Anissa Lamani, Franck Petit, and Sébastien Tixeuil. Optimal torus exploration by oblivious robots. *Computing*, 101(9):1241–1264, 2019. doi:10.1007/s00607-018-0595-8.

10 Durjoy Dutta, Tandrima Dey, and Sruti Gan Chaudhuri. Gathering multiple robots in a ring and an infinite grid. In *Distributed Computing and Internet Technology - 13th International Conference, ICDCIT 2017, Bhubaneswar, India, January 13-16, 2017, Proceedings*, volume 10109 of *Lecture Notes in Computer Science*, pages 15–26. Springer, 2017. doi:10.1007/978-3-319-50472-8_2.

11 Paola Flocchini, Giuseppe Prencipe, and Nicola Santoro, editors. *Distributed Computing by Mobile Entities, Current Research in Moving and Computing*, volume 11340 of *Lecture Notes in Computer Science*. Springer, 2019. doi:10.1007/978-3-030-11072-7.

12 Samuel Guilbault and Andrzej Pelc. Gathering asynchronous oblivious agents with local vision in regular bipartite graphs. *Theoretical Computer Science*, 509:86–96, 2013. doi:10.1016/j.tcs.2012.07.004.

13 David Ilcinkas. Oblivious robots on graphs: Exploration. In *Distributed Computing by Mobile Entities, Current Research in Moving and Computing*, volume 11340 of *Lecture Notes in Computer Science*, pages 218–233. Springer, 2019. doi:10.1007/978-3-030-11072-7_9.

14 Tomoko Izumi, Taisuke Izumi, Sayaka Kamei, and Fukuhito Ooshita. Time-optimal gathering algorithm of mobile robots with local weak multiplicity detection in rings. *IEICE TRANSACTIONS on Fundamentals of Electronics, Communications and Computer Sciences*, 96-A(6):1072–1080, 2013. doi:10.1587/transfun.E96.A.1072.

15 Sayaka Kamei, Anissa Lamani, Fukuhito Ooshita, and Sébastien Tixeuil. Gathering an even number of robots in an odd ring without global multiplicity detection. In *Mathematical Foundations of Computer Science 2012 - 37th International Symposium, MFCS 2012, Bratislava, Slovakia, August 27-31, 2012. Proceedings*, volume 7464 of *Lecture Notes in Computer Science*, pages 542–553. Springer, 2012. doi:10.1007/978-3-642-32589-2_48.

16 Sayaka Kamei, Anissa Lamani, Fukuhito Ooshita, Sébastien Tixeuil, and Koichi Wada. Gathering on rings for myopic asynchronous robots with lights. In *23rd International Conference on Principles of Distributed Systems, OPODIS 2019, December 17-19, 2019, Neuchâtel, Switzerland*, volume 153 of *LIPIcs*, pages 27:1–27:17. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2019. doi:10.4230/LIPIcs.OPODIS.2019.27.

17 Sayaka Kamei, Anissa Lamani, Fukuhito Ooshita, Sébastien Tixeuil, and Koichi Wada. Asynchronous gathering in a torus. *CoRR*, abs/2101.05421, 2021. arXiv:2101.05421.

18 Ralf Klasing, Adrian Kosowski, and Alfredo Navarra. Taking advantage of symmetries: Gathering of many asynchronous oblivious robots on a ring. *Theoretical Computer Science*, 411(34-36):3235–3246, 2010. doi:10.1016/j.tcs.2010.05.020.

19 Laure Millet, Maria Potop-Butucaru, Nathalie Sznajder, and Sébastien Tixeuil. On the synthesis of mobile robots algorithms: The case of ring gathering. In *Stabilization, Safety, and Security of Distributed Systems - 16th International Symposium, SSS 2014, Paderborn, Germany, September 28 - October 1, 2014. Proceedings*, volume 8756 of *Lecture Notes in Computer Science*, pages 237–251. Springer, 2014. doi:10.1007/978-3-319-11764-5_17.

20 Gabriele Di Stefano and Alfredo Navarra. Optimal gathering of oblivious robots in anonymous graphs and its application on trees and rings. *Distributed Computing*, 30(2):75–86, 2017. doi:10.1007/s00446-016-0278-7.

21 Ichiro Suzuki and Masafumi Yamashita. Distributed anonymous mobile robots: Formation of geometric patterns. *SIAM Journal on Computing*, 28(4):1347–1363, 1999. doi:10.1137/S009753979628292X.