Asynchronous Gathering in a Torus
Sayaka Kamei, Anissa Lamani, Fukuhito Ooshita, Sébastien Tixeuil, Koichi Wada

To cite this version:

HAL Id: hal-03592615
https://hal.sorbonne-universite.fr/hal-03592615
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


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...
Asynchronous Gathering in a Torus

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 et al. [2]. Complete and complete bipartite graphs were outlined by Cicerone et al. [5], and regular bipartite by Guibault 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 criterion 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 \( K \geq 3 \) robots evolving on a non-oriented and anonymous \((\ell,L)\)-torus (or simply torus for short) of \( n \) nodes. Values \( \ell \) 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 \( v_0, \ldots, v_n \) of the nodes of the torus such that for every \( j \in \{0, \ldots, L-1\} \):
   - if \( i + \ell < n \), then \( \{v_i, v_{i+\ell}\} \in E \), else \( \{v_i, v_{i+\ell} \mod n\} \in E \).
   - if \( i + 1 \mod \ell \neq 0 \), 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 \( v_{j \times \ell}, v_{1+j \times \ell}, \ldots, v_{(L-1) \times \ell} \) is called an \( \ell \)-ring. Similarly, for every \( k \in \{0, \ldots, \ell - 1\} \), the sequence \( v_k, v_{k+\ell}, v_{k+2\ell}, \ldots, v_{k+(L-1)\ell} \) is called an \( L \)-ring.

   On the torus operate \( K \geq 3 \) 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.
Let \( \ell_0, \ell_1, \ldots, \ell_{L-1} \) be the sequence of \( \ell \)-rings and let \( v_{i,0}, v_{i,1}, \ldots, v_{i,\ell-1} \) be the sequence of nodes on \( \ell_i \) (all operations on the indices are modulo \( \ell \) for the nodes and modulo \( \ell \) for the \( \ell \)-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 \textit{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, \textit{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 \( i, j > 0 \), 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 \( \Delta_{i,j}^{+,s}(t) \) be the sequence \( < \delta_{i,j}^+(t), \delta_{i+1,j}^+(t), \ldots, \delta_{i+(L-1),j}^+(t) > \) and \( \Delta_{i,j}^{-,s}(t) \) to be the sequence \( < \delta_{i,j}^-(t), \delta_{i-1,j}^-(t), \ldots, \delta_{i-(L-1),j}^-(t) > \) with \( s \in \{+, -\} \).

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

By \( \text{view}_r(t)(1) \), we refer to \( V_{i,j}(t) \) in \( \text{view}_r(t) \). Given two robots \( r \) and \( r' \), we say that \( r \) has a larger view than \( r' \) at time \( t \), denoted \( \text{view}_r(t)(1) > \text{view}_{r'}(t)(1) \), if \( \text{view}_r(t) \) is lexigraphically larger than \( \text{view}_{r'}(t) \). 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 \( r \), \( \text{view}_r(t)(1) > \text{view}_{r'}(t)(1) \) 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, \( \text{view}_r(t)(1) \neq \text{view}_{r'}(t)(1) \) 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 \( \ell_x \) at node \( v_{x,w} \), \( \text{view}_{r_{(x,w)}}(t)(1) = \text{view}_{r_{(x+w+w)}(t)}(t)(1) \) (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 (\( \ell, L \))-torus, \textit{i.e.}, \( \ell \neq L \). We assume \text{w.l.o.g.} that \( L < \ell \). In this case, we can differentiate two sides of the torus. We denote by \( \text{nb}_l(C) \) the number of occupied nodes on \( \ell \)-ring \( \ell_i \), in configuration \( C \). An \( \ell \)-ring \( \ell_i \) is said to be maximal in \( C \) if \( \forall j \in \{0, \ldots, L-1\} \setminus \{i\} \), \( \text{nb}_{l_i}(C) \leq \text{nb}_{l_j}(C) \).

Given a configuration \( C \) and two \( \ell \)-rings \( \ell_i \) and \( \ell_j \). We say that \( \ell_j \) is adjacent to \( \ell_i \) if \( |i-j| = 1 \) mod \( L \) holds. Similarly, we say that \( \ell_j \) is neighbor of \( \ell_i \) in configuration \( C \) if \( \text{nb}_{l_j}(C) > 0 \) and \( \text{nb}_{l_i}(C) = 0 \) for any \( k \in \{i+1, i+2, \ldots, j-1\} \) or \( k \in \{i-1, i-2, \ldots, j+1\} \). We also define \( \text{dis}(x_i, x_j) \) 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 \( x_i = r_i \) when \( r_i \) 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 $m : V \rightarrow \mathbb{N}$ 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 $\phi$ 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 $\phi$ always exists as the identity permutation fits this definition. Similarly, given a configuration $(G, m)$, let $\psi$ 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, $\psi$ always exists as the identity permutation fits this definition. Given such a permutation $\psi$, the cycle $C_\psi$ of order $p$ that is generated by $\psi$ is \{ $\psi^0, \psi^1 = \psi, \psi^2 = \psi \circ \psi, \ldots, \psi^{p-1}$ \} such that $\psi^p = \psi^0$, where $\phi^0$ is the identity. Note that $C_\psi$ has order 1 if and only if $\psi$ is the identity. Given a cycle $C_\psi$, the orbit of a vertex $v$ of $V$ is $C_\psi(v) = \{ \gamma(v) | \gamma \in C_\psi \}$. Now, given a configuration $(G = (V, E), m)$, $\psi$ is partitive if $C_\psi$ has order $p > 1$, and for every $v \in V$, $|C_\psi(v)| = p$. That is, $\psi$ 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:

\textbf{Theorem 1} ([20] Restated). If a configuration $(G, m)$ admits a partitive permutation $\psi$, then $(G, m)$ cannot be gathered.

We specialize the general theorem to our setting:

\textbf{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.

\textbf{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.

\textbf{Theorem 3}. Starting from any configuration with $K \geq 2$ 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 idea is to make all robots neither on empty. By contrast, Partial(Preparation (respectively the Gathering) phase. Let us refer by 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 \( C \in C_{\text{target}} \).

2. Gathering Phase. Starting from a configuration \( C \in C_{\text{target}} \), the gathering node is identified, and all robots eventually move to join it i.e., the gathering is achieved.

Let us refer by \( C_{p_1} \) (respectively \( C_{p_2} \)) 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 \( C_{p_1} \cap C_{p_2} = \emptyset \) and \( C_{\text{target}} \subset C_{p_2} \).

\begin{verbatim}
if C \in C_{p_2} then
    Execute Gathering phase
else
    Execute Preparation phase
\end{verbatim}

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 \( \forall j \in \{0, \ldots, L-1\} \setminus \{i\}, nb_j(C) < nb_i(C) \).

- **Empty(\( C \))**: \( (C \in C_{\text{target}}) \land (\forall i \in \{0, \ldots, L-1\}, such that \ell_i \neq \ell_{\text{target}} and \ell_i \neq \ell_{\max}, nb_i(C) = 0) \).

- **Partial(\( C \))**: \( (C \in C_{\text{target}}) \land (\exists i \in \{0, \ldots, L-1\}, such that \ell_i \neq \ell_{\text{target}} and \ell_i \neq \ell_{\max}, nb_i(C) \neq 0) \).

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

In our algorithm, in several cases, robots in a single \( \ell \)-ring, say \( \ell_i \), need to move and align themselves with respect to the positions of other robots which are on another \( \ell \)-ring, say \( \ell_k \). To ease the description of the algorithm, we define a procedure referred to by Align(\( \ell_i, \ell_k \)) which makes the robots to perform such alignment i.e., align robots on \( \ell_i \) with respect to robots positions on \( \ell_k \). When the procedure is called in a configuration \( C \), the following properties hold on both \( \ell_i \) and \( \ell_k \):
Figure 2 Transitions among all configurations.

Figure 3 Some examples of Align($\ell_i, \ell_k$).

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

Let $u_{\text{mark}}$ be the node on $\ell_k$ that is: occupied if $\text{nb}_{\ell_k}(C) = 1$, empty in the 2.block if $\text{nb}_{\ell_k}(C) = 2$, occupied in the middle of the 1.block if $\text{nb}_{\ell_k}(C) = 3$. Node $u_{\text{mark}}$ is used as a land mark to align robots on $\ell_i$ (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 $\ell_i$ then it is adjacent to $u_{\text{mark}}$. 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 $\ell_i$ 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 $C \in C_{p1}$. The purpose of this phase is to reach a configuration $C' \in C_{\text{target}}$ 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 $\ell$-rings to reach a configuration $C''$ in which Unique($C''$) is true. Then, from configuration $C''$, robots need to create both $\ell_{\text{target}}$ and $\ell_{\text{secondary}}$ to reach a configuration $C' \in C_{\text{target}}$. 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 $\ell$-rings). Robots need to decrease the number of maximal $\ell$-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 $\ell$-ring: if a maximal $\ell$-ring hosts at least one empty node then, the idea is to fill one of these empty nodes on a single maximal $\ell$-rings. By contrast, if all the nodes of the maximal $\ell$-rings are occupied, the idea is to create a single multiplicity on one of the maximal $\ell$-rings to decrease their number gradually. Robots to move are chosen carefully...
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 \( \ell \)-ring by \( \ell_{\text{max}} \). Robots behavior in a configuration \( C \) in which Unique\((C)\) holds is as follows:

1. If \( \text{nbr}_{\text{max}}(C) = \ell \) (all the nodes of \( \ell_{\text{max}} \) are occupied). Let \( R_{\text{max}}(C) \) be the set of robots on a maximal \( \ell \)-ring. As \( C \) is rigid, all robots in \( R_{\text{max}}(C) \) have a unique view. Let \( \ell_m \) be the maximal \( \ell \)-ring in \( C \) that hosts the robot with the maximal view in \( R_{\text{max}}(C) \). One robot \( r \) is elected on \( \ell_m \) to move. Its destination is one of its adjacent occupied nodes on \( \ell_m \). Robot \( r \) is selected as follows: Let \( R_m(C) \subset R_{\text{max}}(C) \) be the set of robots on \( \ell_m \) which by moving to one of their adjacent occupied node on \( \ell_m \), 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 \( \text{nbr}_{\text{max}}(C) < \ell \) (There is at least one empty node on \( \ell_{\text{max}} \)), the idea is to fill exactly one of the empty nodes on exactly one of the maximal \( \ell \)-ring. Let \( R(C) \) be the set of robots closest to an empty node on a maximal \( \ell \)-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 \( \ell \)-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:

\[ \text{If } C \text{ contains exactly two occupied } \ell \text{-rings then, } C \text{ contains only two maximal } \ell \text{-rings.} \]

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 \( \ell \)-ring (Note that this \( \ell \)-ring exists since \( L > 4 \)).

\[ \text{If } C \text{ contains more than two occupied } \ell \text{-rings then: let } r \text{ be the robot in } R(\ell) \text{ with the largest view. By } u \text{ and target-} \ell \text{ we refer to the closest empty node on a maximal } \ell \text{-ring to } r \text{ and the } \ell \text{-ring including } u. \text{ If by moving, } r \text{ does not create a symmetric configuration, then } r \text{ simply moves to its adjacent node toward } u \text{ taking the shortest path. By contrast, if } r \text{ creates a symmetric configuration by moving, then let } C' \text{ be the configuration reached once } r \text{ moves. Using configuration } C' \text{ that each robot can compute without } r \text{ moving, another robot } r' \text{ in } C \text{ is selected to move. We show later on that a symmetric configuration can only be reached when } r \text{ either joins an empty node on the same } L \text{-ring as } u \text{ for the first time or when it joins } u. \text{ For the other cases, the configuration remains rigid. Hence, we only address the following two cases:} \]

\[ \text{a. Robot } r \text{ joins an empty node on the same } L \text{-ring as } u \text{ for the first time in } C'. \text{ In this case, in } C', \text{ the robot that is on target-} \ell \text{ being on the same } L \text{-ring as } r \text{ moves to } u \text{ (refer to Figure 4).} \]

\[ \text{b. Robot } r \text{ joins } u \text{ in } C'. \text{ If in } C' \text{ there are only two occupied } \ell \text{-rings. The robot with the largest view which does not create a symmetric configuration is elected to move. Its destination is its adjacent empty node on an empty } \ell \text{-ring. By contrast, if there are more than two occupied } \ell \text{-rings in } C' \text{ then robots proceed as follows:} \]
If the axis of symmetry lies on the unique $\ell_{\text{max}}$ in $C'$ then, we are sure that there are two $\ell$-rings which are maximal in $C$ and that are symmetric with respect to the unique maximal $\ell$-ring in $C'$. Let $r$ be the robot located on a maximal $\ell$-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 $\ell$-ring.

If the axis of symmetry is perpendicular to the unique maximal $\ell$-ring in $C'$ then let $T$ be the set of occupied $\ell$-rings in $C$ without target-$\ell$. If there is an $\ell$-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 $\ell$-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 $\ell$-ring in $T$ (all $\ell$-rings contains two 1.blocks separated by a unique empty node), then using the rigidity of $C$, one robot being on an $\ell$-ring of $T$ who has an empty node as a neighbor on its $\ell$-ring is elected to move. Its destination is its adjacent empty node on its current $\ell$-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 $\ell$-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 $\ell$-ring, $\ell_{\text{max}}$. To reach a configuration $C' \in C_{\text{target}}$, robots need to move to build both $\ell_{\text{secondary}}$ and $\ell_{\text{target}}$ i.e., one of the two adjacent $\ell$-rings to $\ell_{\text{max}}$ 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 $\ell_i$ and $\ell_k$ be the two adjacent $\ell$-rings to $\ell_{\text{max}}$. Assume w.l.o.g. that $n_{\ell_i}(C) < n_{\ell_k}(C)$. To ease the description of this phase, we distinguish five main cases describing the possible states of $\ell_i$ and $\ell_k$: (i) the case in which both $\ell_i$ and $\ell_k$ are empty ($C \in C_{\text{Empty}}$). The idea, in this case, is to elect a single robot to join either $\ell_i$ or $\ell_k$. (ii) the case in which $\ell_i$ is empty and $\ell_k$ hosts more than one occupied node ($C \in C_{\text{Semi-Empty}}$). The idea is to make the robots on $\ell_k$ gather in a single node. Note that in both cases (i) and (ii), a configuration $C' \in C_{\text{target}}$ is created. (iii) the case in which $\ell_i$ hosts a single occupied node while $\ell_k$ hosts at least two robots ($C \in C_{\text{Oriented}}$). The unique occupied node on $\ell_i$ is used as a landmark to make robots on $\ell_k$ move and create either a 2.block of size 2 or a 1.block of size 3 ($C \in C_{\text{Oriented-2}}$). Once such a block is created ($C \in C_{\text{Oriented-1}}$), it is easy to free $\ell_i$ as the robots move to their adjacent node on $\ell_{\text{max}}$ (since the configuration reached $C' \in C_{\text{target}}$, the multiplicity created on $\ell_{\text{max}}$ can be identified as it is adjacent to $\ell_{\text{target}}$). (iv) the case in which both $\ell_i$ and $\ell_k$ host a unique occupied node ($C \in C_{\text{Oriented}}$). The idea is to add a single robot to either $\ell_i$ or $\ell_k$. Finally, (v) the case in which both $\ell_i$ and $\ell_k$ host more than one robot ($C_{\text{Undefined}}$). The idea is to make robots elect either $\ell_i$ or $\ell_k$ 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 $C_{\text{Oriented}}$. More formally:

1. Set $C_{\text{Empty}}$: $C \in C_{\text{Empty}}$ if $n_{\ell_i}(C) = n_{\ell_k}(C) = 0$.
2. Set $C_{\text{Semi-Empty}}$: $C \in C_{\text{Semi-Empty}}$ if w.l.o.g. $n_{\ell_i}(C) = 0$ and $n_{\ell_k}(C) > 1$.
3. Set $C_{\text{Oriented}}$: $C \in C_{\text{Oriented}}$ if w.l.o.g. $n_{\ell_i}(C) = 1$ and $n_{\ell_k}(C) > 1$. Set $C_{\text{Oriented}}$ includes:
   a. $C_{\text{Oriented-1}}$. In this case either (i) $n_{\ell_k}(C) = 3$ and $\ell_k$ contains a 1.block of size 3 whose middle robot is on the same $L$-ring as the unique occupied node on $\ell_i$. (ii) $n_{\ell_k}(C) = 2$ and $\ell_k$ contains a 2.block. Moreover, the unique empty node in the 2.block is on the same $L$-ring as the unique robot on $\ell_i$. 
b. \( C_{\text{Oriented-2}} \). Contains all the configuration in \( C_{\text{-oriented}} \) that are not in \( C_{\text{Oriented-1}} \). That is, \( C_{\text{Oriented-2}} = C_{\text{Oriented}} - C_{\text{Oriented-1}} \).

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

5. Set \( C_{\text{Undefined}} : C \in C_{\text{Undefined}} \) 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 \( \ell \)-ring.

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

1. \( C \in C_{\text{Empty}} \). Let \( \ell_n \) and \( \ell_k \) be the two neighboring \( \ell \)-rings of \( \ell_{\text{max}} \) (one neighboring \( \ell \)-ring from each direction). In the case in which \( \ell_n = \ell_k = \ell_{\text{max}} \) (\( C \) contains a single occupied \( \ell \)-ring) then, using the rigidity of \( C \), one robot from \( C \) is selected to move to its adjacent empty node outside its \( \ell \)-ring (the scheduler chooses the direction to take). Otherwise, let \( R_m \) be the set of robots which are the closest to either \( \ell_i \) or \( \ell_k \). 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 \( \ell_i \). The destination of \( r \) is its adjacent empty node outside its current \( \ell \)-ring on the shortest empty path toward \( \ell_i \). If \( r \) is the closest to both \( \ell_i \) and \( \ell_k \) then the scheduler chooses the direction to take (it moves either toward \( \ell_i \) or \( \ell_k \)). 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. \( C \in C_{\text{Semi-Empty}} \). Assume w.l.o.g. \( nb_{\ell_i}(C) > 1 \) and \( nb_{\ell_k}(C) = 0 \). We consider two cases:
   a. \( nb_{\ell_i}(C) > 3 \) or \( nb_{\ell_k}(C) = 2 \). Recall that \( C \notin C_{\text{target}} \). Let \( \uparrow \) be the direction defined from \( \ell_{\text{max}} \) to \( \ell_k \) taking the shortest path and let \( \ell_n \) be the \( \ell \)-ring that is neighbor of \( \ell_i \). Observe that \( \ell_n = \ell_k \) is possible (if only two \( \ell \)-rings are occupied in \( C \)). Using the rigidity of configuration \( C \), one robot from \( \ell_n \) is elected to move. Its destination is its adjacent node outside \( \ell_n \) and towards \( \ell_i \) with respect to the direction \( \uparrow \).
   b. \( nb_{\ell_i}(C) = 3 \). Again, recall that \( C \notin C_{\text{target}} \). The aim is to make the three robots form a single 1.block. To this end, if the configuration contains a single 1.block of size 3 with \( d > 1 \) then the robot in the middle of the 1.block moves to its adjacent node on \( \ell_k \) (the scheduler chooses the direction to take). By contrast, if the configuration contains a single 1.block of size 2 \( (d \geq 1) \) then the robot not part of the 1.block moves towards its adjacent empty node towards the 1.block taking the shortest empty path.

3. \( C \in C_{\text{Oriented}} \). Let \( r_i \) be the single robot on \( \ell_i \).
   a. \( C \in C_{\text{Oriented-1}} \). If \( nb_{\ell_{\text{max}}}(C) > 4 \) then the unique robot on \( \ell_i \) moves to its adjacent node on \( \ell_{\text{max}} \). Otherwise, let \( u \) be the node on \( \ell_{\text{max}} \) adjacent to a robot on \( \ell_i \).
      * If \( nb_{\ell_{\text{max}}}(C) = 3 \) and the robots form a 1.block of size 3 whose middle robot is adjacent to \( u \) then the unique robot on \( \ell_i \) moves to its adjacent node on \( \ell_{\text{max}} \). Otherwise, robots on \( \ell_{\text{max}} \) execute **Align** \( (\ell_{\text{max}}, \ell_i) \).
      * If \( nb_{\ell_{\text{max}}}(C) = 4 \) and \( u \) is empty, then the unique robot on \( \ell_i \) moves to \( u \). Otherwise \( (u \) is occupied), then let \( r \) be the robot on \( u \).

\[ \text{Figure 5} \] Instance of configurations \( C \) when \( \text{Unique}(C) \) is true.
If \( r \) has an adjacent empty node on \( \ell_{\text{max}} \) 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 \( \ell_{\text{max}} \), then let \( r' \) be the robot on \( \ell_{\text{max}} \) which is adjacent to \( r \) and which does not have a neighboring robot on \( \ell_{\text{max}} \) at distance \( \lfloor \ell/2 \rfloor \). Robot \( r' \) moves to its adjacent empty node on \( \ell_{\text{max}} \).

b. \( C \in \mathcal{C}_{\text{Oriented-2}} \). 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 \( \ell_k \) and which is on the same \( L \)-ring as the unique occupied node on \( \ell_i \). For this purpose, the robot on \( \ell_k \) which is the closest to \( u_k \) with the largest view is the one allowed to move. Its destination is its adjacent node on \( \ell_k \) toward \( u_k \).

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

4. \( C \in \mathcal{C}_{\text{Undefined}} \). Depending on the number of robots on \( \ell_i \) and \( \ell_k \), we consider two cases:
   a. \( nb_{\ell_i}(C) < nb_{\ell_i}(C) \). The idea is to make robots on \( \ell_i \) gather on \( \ell_i \). We define a configuration, denoted \( \Gamma(C) \), built from \( C \) ignoring some \( \ell \)-rings that will be used to identify a single node on \( \ell_i \) on which all robots on \( \ell_i \) will gather. If there are at least four occupied \( \ell \)-rings in \( C \) then \( \Gamma(C) \) is the configuration built from \( C \) ignoring both \( \ell_i \) and \( \ell_k \). By contrast, if there are only three occupied \( \ell \)-rings then \( \Gamma(C) \) is the configuration built from \( C \) ignoring only \( \ell_i \). The following cases are possible:
      i. Configuration \( \Gamma(C) \) is rigid. Using the rigidity of \( \Gamma(C) \), one node on \( \ell_i \), say \( u \), is elected as the gathering node. Robots on \( \ell_i \) 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 \( \ell_i \) 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 \( \ell_i \) that is on the axis of symmetry of \( \Gamma(C) \) is the gathering node. Robots on \( \ell_i \) move in turn to join it.
         - \( \Gamma(C) \) is node-node symmetric: Let \( u_1 \) and \( u_2 \) be the two nodes on \( \ell_i \) 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 \( \ell_i \) 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 \( \ell_i \) 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 \( \ell_i \) 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 \( \ell_i \) 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 \( \ell_i \) 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
Asynchronous Gathering in a Torus

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 $\ell_i$ 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 $\ell_i$ 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 $\ell_i$ 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 $\ell_i$ 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 $\ell_i$, 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 $\ell_i$ 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 $\ell_i$ are the ones to move. Its destination is its adjacent node towards $u_1$ taking the shortest path. By contrast, if all robots on $\ell_i$ 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 $\ell_i$ are on both

Figure 6 Case in which $\Gamma(C)$ is edge-edge symmetric.
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 \( \ell_i \) 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 \( \ell_i \) 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 \( \ell \)-ring. This reduces the number of axis of symmetries to either 1 or 0.

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

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

### 4.2 Gathering Phase

This phase starts from a configuration \( C \in \mathcal{C}_{\text{target}} \) in which a direction is defined in \( C \) (from \( \ell_{\text{target}} \) to \( \ell_{\text{max}} \)). The idea is to make all robots that are neither on \( \ell_{\text{target}} \) nor \( \ell_{\text{max}} \) move to join \( v_{\text{target}} \). Then, make some robots on \( \ell_{\text{max}} \) move to join \( v_{\text{target}} \) while the other align themselves with respect to \( v_{\text{target}} \) to finally gather all on the node of \( \ell_{\text{max}} \) adjacent to \( v_{\text{target}} \).

To ease the description of our algorithm, we define the following set of configurations:

1. **Set \( \mathcal{C}_{\gamma} \)** which includes the following four sub-sets:
   a. **SubSet \( \mathcal{C}_{\gamma-1} \):** \( C \in \mathcal{C}_{\gamma-1} \) if there are exactly two occupied \( \ell \)-rings in \( C \) denoted \( \ell_i \) and \( \ell_j \) respectively on the following conditions hold: (1) \( \ell_i \) and \( \ell_j \) are adjacent. (2) \( nb_{\ell_i}(C) < nb_{\ell_j}(C) \) (3) either:
      - \( nb_{\ell_i}(C) = 4 \) and \( \ell_i \) 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 \( \ell_i \).
      - \( nb_{\ell_i}(C) = 3 \) or 5 and \( \ell_i \) contains a \( 1 \)-block of size \( nb_{\ell_i}(C) \). Let \( u \) be the middle node of the \( 1 \)-block of size \( nb_{\ell_i}(C) \).
   b. **SubSet \( \mathcal{C}_{\gamma-2} \):** \( C \in \mathcal{C}_{\gamma-2} \) if \( C \in \mathcal{C}_{\text{target}} \) and \( nb_{\text{target}}(C) = 1 \). In addition either one of the following conditions are verified: (1) \( nb_{\ell_{\text{max}}}(C) = 4 \) and on \( \ell_{\text{max}} \) 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 \( v_{\text{target}} \). (2) \( nb_{\ell_{\text{max}}} = 5 \) and on \( \ell_{\text{max}} \) there is a \( 1 \)-block of size 5 whose middle robot is adjacent to \( v_{\text{target}} \). (3) \( nb_{\ell_{\text{max}}}(C) = 4 \) and on \( \ell_{\text{max}} \) 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 \( v_{\text{target}} \) (refer to Figure 7).
c. SubSet $C_{sp-3}$: $C \in C_{sp-3}$ if $C \in C_{target}$, $Empty(C)$ is true, $nb_{target}(C) = 1$ and one of the two following conditions holds: (1) $nb_{\ell_{max}}(C) = 3$ and $\ell_{max}$ 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 $\ell_{max}$. Let $u$ be the unique empty node between the two robots on $\ell_{max}$, then $u$ is adjacent to $v_{target}$ (refer to Figure 7).

d. SubSet $C_{sp-4}$: $C \in C_{sp-4}$ if there is a unique $\ell$-ring that is occupied and on this $\ell$-ring there are either two or three occupied nodes that form a 1.block (refer to Figure 7).

2. Set $C_{pr}$: $C \in C_{pr}$ if $C \in C_{target}$ 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 $C \notin C_{sp}$.

3. Set $C_{ls}$: $C \in C_{ls}$ if $C \in C_{target}$ and $C \notin C_{sp}$ and $Empty(C)$. In other words, there are only two $\ell$-rings that are occupied: $\ell_{max}$ and $\ell_{target}$.

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

1. $C \in C_{pr}$. Let us refer by $\ell_{i}$ to the $\ell$-ring that is adjacent to $\ell_{target}$ such that $\ell_{i} \neq \ell_{max}$.
   Depending on the number of robots on $\ell_{i}$, two cases are possible:
   a. $nb_{\ell_{i}}(C) > 0$. Let $R_{m}$ be the set of robots on $\ell_{i}$ that are the closest to $v_{target}$, if
      i. there is an occupied node $u_{i}$ on $\ell_{i}$ 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 $\ell_{i}$ 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 $\ell_{i}$ on the empty path toward $v_{target}$.
      iii. there is no robot on $\ell_{i}$ adjacent to $v_{target}$ and $nb_{\ell_{i}}(C) = \ell - 1$, then let $R_{sp}$ be the set of robots that share a hole with $u_{i}$, where $u_{i}$ is the node on $\ell_{i}$ 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 $\ell_{k}$ be the closest neighboring $\ell$-ring to $\ell_{target}$ with respect to $\uparrow$. Let $R_{m}$ be the set of robots on $\ell_{k}$ that are closely to $v_{target}$. Robots on $R_{m}$ are the ones to move, their destination is the node outside $\ell_{k}$ and toward $\ell_{target}$ with respect to $\uparrow$.

2. $C \in C_{ls}$. Robots aims at reaching a configuration $C' \in C_{sp}$. If $nb_{\ell_{max}}(C) \leq 5$, robots on $\ell_{max}$ 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 $\ell_{max}$ such that $u_{4}$ is adjacent to $v_{target}$. If $u_{3}$ is occupied and has exactly one adjacent occupied node on $\ell_{max}$ (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 $\ell_{max}$, or two adjacent occupied
nodes on $\ell_{max}$, then robots on $u_3$ move to $v_{\text{target}}$. Finally, if $u_3$ is empty then let $R$ be the set of robots that are closest to $u_3$ on $\ell_{max}$. If $|R| = 2$ then both robots move to their adjacent node on $\ell_{max}$ 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 $\ell_{max}$ 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. $C \in C_{sp}$. We distinguish:
   a. $C \in C_{sp-1}$. If $C \in C_{\text{target}}$, then the robots on $\ell_{\text{target}}$ that are at the extremities of the 1.block or the 2.block move to their adjacent occupied node on $\ell_{max}$. By contrast, if $C \not\in C_{\text{target}}$, then the robot not on $\ell_{max}$ that has two adjacent occupied nodes moves to its adjacent node on $\ell_{max}$.
   b. $C \in C_{sp-2}$. If there is a 1.block of size 3 on $\ell_{max}$ 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 $\ell_{max}$ contains a 1.block of size 5 then the robots on $\ell_{max}$ that are adjacent of the extremities of the 1.block move on $\ell_{max}$ in the opposite direction of the extremities of the 1.block. Finally, if $\ell_{max}$ contains two 1.blocks of size 2 then the robots that share a hole of size 1 move toward each other.
   c. $C \in C_{sp-3}$. Robots on $v_{\text{target}}$ move to their adjacent node on $\ell_{max}$ (note that $v_{\text{target}}$ can be occupied by either a single robot or a multiplicity).
   d. $C \in C_{sp-4}$. 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 $(\ell, L)$-torus in which $L < \ell$ and $L > 4$ and starting from an arbitrary rigid configuration, Protocol 1 solves the gathering problem for any $K \geq 3$.

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 lose 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

Asynchronous Gathering in a Torus


