Gathering over Meeting Nodes in Infinite Grid*

Subhash Bhagat
Département d’informatique
Université du Québec en Outaouais, Canada
subhash.bhagat.math@gmail.com

Abhinav Chakraborty†, Bibhuti Das, Krishnendu Mukhopadhyaya
Advanced Computing and Microelectronics Unit
Indian Statistical Institute, Kolkata, India
{abhinav.chakraborty06, dasbibhuti905, krishnendu.mukhopadhyaya}@gmail.com

Abstract. The gathering over meeting nodes problem asks the robots to gather at one of the pre-defined meeting nodes. The robots are deployed on the nodes of an anonymous two-dimensional infinite grid, which has a subset of nodes marked as meeting nodes. Robots are identical, autonomous, anonymous and oblivious. They operate under an asynchronous scheduler. They do not have any agreement on a global coordinate system. All the initial configurations for which the problem is deterministically unsolvable have been characterized. A deterministic distributed algorithm has been proposed to solve the problem for the remaining configurations. The efficiency of the proposed algorithm is studied in terms of the number of moves required for gathering. A lower bound concerning the total number of moves required to solve the gathering problem has been derived.

Keywords: Swarm Robotics, Gathering, Meeting Nodes, Asynchronous, Look-Compute-Move Cycle

*A preliminary version of this paper appeared in the 6th International Conference, CALDAM 2020, Hyderabad, India, February 13-15, 2020.
†Address for correspondence: Advanced Computing and Microelectronics Unit, Indian Statistical Institute, Kolkata, India.
1. Introduction

A swarm of robots consists of small and inexpensive robots that work together in a cooperative environment to execute some complex tasks. A considerable amount of research in distributed computing has been focused on robot-based computing systems because of their importance in a wide range of real-world applications, like search and rescue operations, military operations, disaster management, cleaning a big surface, etc. In robot-based computing systems, gathering is one of the most widely studied problems. The gathering problem asks the mobile robots, which are initially placed at distinct locations, to gather at a common location. The common location is not fixed a priori, and the gathering should be finalized within a finite amount of time. The study of the gathering problem aims to find the minimal amount of capabilities required to solve the problem. It has been extensively studied in both the continuous [1, 2, 3] and discrete domains [4, 5, 6, 7]. In the discrete environment, the robots are deployed on the nodes of an anonymous graph.

In this paper, we have considered the gathering over meeting nodes problem in an infinite square grid. The robots and the meeting nodes are deployed on the nodes of the grid. We assume the robots to be:

- **Anonymous**: No unique identifiers.
- **Autonomous**: No central controller.
- **Identical**: Indistinguishable by their appearance.
- **Homogeneous**: All of them execute the same deterministic algorithm.
- **Oblivious**: No memory of past information.
- **Silent**: No explicit means of direct communication.
- **Disoriented**: No access to a global coordinate system, no common compass and no agreement on chirality.
- **Unlimited visibility**: Can perceive the entire graph.

When a robot becomes active, it operates according to Look-Compute-Move (LCM) cycle. In the look phase, a robot takes a snapshot of the current configuration. Based on this snapshot, it computes a destination node in the compute phase. Note that the destination node may be its current position as well. A robot moves towards its destination node in the move phase. If the destination node is the current position of the robot, then the robot performs a null movement. The topology considered in this paper is an anonymous grid graph, i.e., the nodes and the edges of the input grid graph are unlabeled. In the initial configuration, the robots are placed at distinct nodes of the grid. The input graph also comprises some pre-defined fixed nodes, which are referred to as meeting nodes. The meeting nodes are visible to the robots during the look phase, and they occupy distinct nodes of the grid. In the initial configuration, we assume that a robot may be deployed on a meeting node.

Based on the activation and timing of the robots, the following schedulers are considered in the literature.
1. **Fully-Synchronous** (FSYNC): In the *fully synchronous* (FSYNC) setting, all the robots are activated simultaneously. The activation phase of all the robots can be divided into global rounds.

2. **Semi-synchronous** (SSYNC): In this setting, a subset of robots are activated simultaneously. The activation phase of each such robot can be divided into global rounds.

3. **Asynchronous** (ASYNC): In the *asynchronous* (ASYNC) setting, there is no common notion of time. The duration of each *Look, Compute* and *Move* phases is finite, but unpredictable.

In this paper, we have assumed that the cycles are performed *asynchronously* by each robot. We assume the scheduler to be fair, i.e., each robot performs its *Look-Compute-Move* (LCM) cycle within finite time and infinitely often.

During the *Look* phase, the robots have *multiplicity detection capability*. The *global-strong multiplicity detection* capability of a robot allows a robot to count the actual number of robots in each node. If the robots have *global-weak multiplicity detection* capability, they can only detect whether multiple robots occupy a node. They cannot count the exact number of robots composing the multiplicity. If the robots are endowed with *local multiplicity detection capability* instead of the global version, then they can detect whether there is a multiplicity or the number of robots composing the multiplicity only in their current location node. In this paper, we assumed the robots to have *local-weak multiplicity detection* capability, i.e., the robots can detect whether there exists a multiplicity in their current location node.

### 1.1. Motivation

In this paper, we investigate the gathering problem in an infinite grid, where some of the nodes are designated as *meeting nodes*. In this setting, the movements of the robots are only allowed along the grid lines and the robots need to gather at one of the *meeting nodes*. The fundamental motivation behind studying the gathering over meeting nodes problem in infinite grids is to investigate the solvability of the gathering problem where both the movements of the robots and the gathering points are restricted.

1. Gathering [4, 5, 6, 7] problem has been studied in the discrete domain, where the movements of the robots are restricted along the edges. However, the gathering point was not restricted. The rationale behind considering the *meeting nodes* might be of practical use. In general, the *gathering problem* requires the robots to coordinate their movements and meet at a location that they are not aware of beforehand. However, the *gathering* may be limited to some specific regions or points. Another possibility is that robots may need to gather at one of the designated points in many real-life applications, e.g., one of the base stations or charging stations, etc.

2. Cicerone et al. [8] have studied the gathering on meeting points problem in the Euclidean plane. Though the gathering points are restricted, the robots are free to move throughout the plane. In the continuous domain, it is assumed that the robots move with high accuracy and infinite precision. In specific models, the robots can even perform *guided* movements, i.e., they can move along some specified curve [9, 10]. Moreover, the robots can move even by
infinitesimally small amounts. Even if the field of robot deployment is small, a dimensionless robot can move without creating any collision. The correctness of the algorithms relies on the accurate execution of the movements. However, the vision sensors do not have infinite precision for real-life robots with weak mechanical capabilities. In the continuous domain, a robot can travel an amount of distance that may be an irrational number. In practice, it may not always be possible to perform such infinitesimal movements with infinite precision. This motivates us to consider the problem in a grid-based terrain where the movements are restricted along the grid lines, and a robot can move to one of its neighbors in one step. However, we have assumed that the movement of a robot is instantaneous, i.e., during the look phase of each robot, the other robots are always detected on the nodes of the input grid graph. Thus, consideration of discrete environments recognizes the fact that many vision sensors produce digital and therefore, discrete snapshots of the environment. The grid topology is a natural discretization of the plane. It has numerous applications in real-life robot navigation systems, such as industrial Automated Guided Vehicles [11] and Coverage Path Planning [12], where grid type floor layouts can be suitably implemented. The restrictions imposed by the grid model on the movements of the robots make it difficult to design algorithms, as opposed to the movement of the robots in a continuous environment.

1.2. Earlier works

The gathering problem has been extensively studied in the literature [1] [2] [3] [13] [14]. In the discrete domain, the robots are deployed on the nodes of the input graph. Gathering in the discrete domain has been largely studied in rings [4] [5] [15] [16] [17] [18]. Klasing et al. [4] studied the gathering problem in an anonymous ring and proved that gathering in an anonymous ring is impossible without multiplicity detection capability. With the assumption of global-weak multiplicity detection capability, they proposed a distributed algorithm to solve the gathering problem for all the configurations having an odd number of robots and all asymmetric configurations when the number of robots is even. Klasing et al. [5] studied configurations in an anonymous ring which admits symmetries and having an even number of robots. They solved the problem for all configurations with more than eighteen robots. They proved that, for an odd number of robots, gathering is feasible if and only if the configuration is not periodic. Kamei et al. [18] studied the gathering problem in anonymous rings using local weak multiplicity detection capability. D’Angelo et al. [15] considered the gathering problem on anonymous rings with six robots. They proposed a distributed algorithm to solve the problem that assumes global-weak multiplicity detection capability of the robots.

Cicerone et al. [19], studied the gathering problem in complete bipartite graphs under FSYNC scheduler. They considered dense and symmetric graphs like complete graphs and complete bipartite graphs. They characterized the solvability of gathering in such graphs. Bose et al. [23] considered the gathering problem in hypercubes. They proposed an optimal algorithm, which minimizes the total number of moves by all the robots.

D’Angelo et al. [6], studied the gathering problem on trees and finite grids. They proved that even with global-strong multiplicity detection capability, a configuration remains ungatherable if and only if it is periodic or symmetric, with the line of symmetry passing through the edges of the grid.
For the remaining configurations, they solved the problem without assuming any multiplicity detection capability of the robots. Stefano et al. [24], studied the optimal gathering of robots in anonymous graphs. They also studied the optimal gathering problem in infinite grids [7]. In [7], they proposed a deterministic distributed algorithm that minimizes the total distance traveled by all the robots. This paper also introduced the concept of Weber-point on vertex-weighted graphs. A Weber-point is a node of the graph that minimizes the sum of the length of the shortest paths from it to each robot.

Fujinaga et al. [25] introduced the concept of fixed points or landmarks on the Euclidean plane. The landmarks covering problem requires the robot to reach a configuration where all the robots must occupy a unique fixed point or a landmark. They propose an algorithm that assumes common chirality among the robots. The proposed algorithm minimizes the total distance traveled by all the robots. Cicerone et al. [26] studied the embedded pattern formation problem without assuming any common chirality. The problem asks for a distributed algorithm that requires the robots to occupy all the fixed points within a finite amount of time. Each fixed point must be occupied by exactly one robot. The k-circle formation [27, 28] problem asks a set of robots to form disjoint circles having k robots each at distinct locations. The circles are centered at the set of fixed points. Cicerone et al. [8], studied a variant of the gathering problem on the Euclidean plane, where robots must gather at one of the predetermined points, referred to as meeting points. They defined the problem as gathering on meeting points problem. They proposed a deterministic algorithm that minimizes the total distance traveled by all the robots and minimizes the maximum distance traveled by a single robot. The proposed algorithms assume global-weak multiplicity detection capability of the robots. In our paper, we have proposed a deterministic algorithm that assumes local-weak multiplicity detection of the robots.

1.3. Our contributions

This paper considers gathering over meeting nodes problem in an infinite grid by asynchronous oblivious mobile robots. We have shown that even if the robots are endowed with multiplicity detection capability, some configurations remain ungatherable. It includes the following collection of configurations:

1. The configurations admitting a unique line of symmetry such that the line of symmetry does not contain any robots or meeting nodes.

2. The configurations admitting rotational symmetry with no robots or meeting nodes on the center of rotation.

We have proposed a deterministic distributed algorithm to solve the gathering problem for the remaining configurations. We have studied the efficiency of the proposed algorithm in terms of the total number of moves executed by the robots. A lower bound has been derived concerning the total number of movements performed by any algorithm for solving the gathering over meeting nodes problem. We have proved that any algorithm that solves the gathering over meeting nodes problem requires \( \Omega(Dn) \) moves, where \( D \) is the larger side of the initial minimum enclosing rectangle of all the robots and meeting nodes and \( n \) is the number of robots. Our proposed algorithm requires \( O(Dn) \) moves, i.e., the algorithm is asymptotically optimal.
1.4. Outline

The following section focuses on the robot model and provides some preliminary definitions and notations. These definitions and notations are relevant in understanding the problem definition. Section 3 provides the formal description of the gathering problem. A sufficient condition for the solvability of the gathering task has been stated in Section 3. In Section 4, we have provided a deterministic distributed algorithm that solves the gathering over meeting nodes. Section 5 provides a lower bound to the complexity of the gathering problem in terms of the number of moves executed by the robots to finalize the gathering. In this section, we have provided a complexity analysis for our proposed algorithm. Finally, Section 6 concludes the paper with some future directions to work with.

2. Model and definitions

2.1. Model

Robots are assumed to be autonomous, anonymous, homogeneous, dimensionless and oblivious. They do not have explicit means of communication. They have an unlimited and unobstructed visibility range, i.e., each robot can observe the entire grid. The robots do not have any agreement on a global coordinate system and chirality. Each robot perceives the configuration with respect to its local coordinate system with origin as its current position. Initially, the robots are assumed to be on the distinct nodes of the input grid. Each active robot executes Look-Compute-Move (LCM) cycle under an asynchronous scheduler. A robot can instantly move to one of its adjacent nodes along the grid lines. The movement of a robot is instantaneous, i.e., any robot performing a Look operation observes all the other robot’s positions only at the nodes of the input grid graph.

2.2. Definitions

In this subsection, we have proposed some terminologies and definitions.

- **System Configuration:**
  - $P = (\mathbb{Z}, E')$: infinite path graph where the vertex set corresponds to the set of integers $\mathbb{Z}$ and the edge set is denoted by the ordered pair $E' = \{(i, i + 1) | i \in \mathbb{Z}\}$.
  - Cartesian product of the graph $P \times P$: input grid graph.
  - $V$ and $E$: set of nodes and edges of the input grid graph, respectively.
  - $d(u, v)$: Manhattan distance between the nodes $u$ and $v$.
  - $R = \{r_1, r_2, \ldots, r_n\}$: a set of robots deployed on the nodes of the grid.
  - $r_i(t)$: position of the robot $r_i$ at time $t > 0$. When there is no ambiguity, $r$ will represent both the robot and the position occupied by it.
  - $R(t) = \{r_1(t), r_2(t), \ldots, r_n(t)\}$: multiset of robot positions at time $t$. At $t = 0$, $r_i(t) \neq r_j(t)$, for all $r_i(t), r_j(t) \in R(t)$. However, at $t > 0$, $r_i(t)$ may be equal to $r_j(t)$, for some $r_i(t), r_j(t) \in R(t)$. 

- $M = \{m_1, m_2, \ldots, m_s\}$: set of meeting nodes located at the nodes of the grid graph.
- $C(t) = (R(t), M)$: system configuration at time $t$.

- **Symmetry**: An automorphism of a graph $G = (V, E)$ is a bijective map $\phi : V \rightarrow V$ such that $u$ and $v$ are adjacent if and only if $\phi(u)$ and $\phi(v)$ are adjacent. Automorphism of graphs can be extended similarly to define automorphism of a configuration. Let $l : V \rightarrow \{0, 1, 2, 3, 4, 5\}$ be defined as a function, where:

$$l(v) = \begin{cases} 
0 & \text{if } v \text{ is an empty node} \\
1 & \text{if } v \text{ is a meeting node} \\
2 & \text{if } v \text{ is a single robot position on a meeting node} \\
3 & \text{if } v \text{ is a robot multiplicity on a meeting node} \\
4 & \text{if } v \text{ is a single robot position not on any meeting node} \\
5 & \text{if } v \text{ is a robot multiplicity not on any meeting node}
\end{cases}$$

An automorphism of a configuration $C(t)$ is an automorphism $\phi$ of the input grid graph such that $l(v) = l(\phi(v))$ for all $v \in V$. The set of all automorphisms of a configuration forms a group which is denoted by $\text{Aut}(C(t), l)$. If $|\text{Aut}(C(t), l)| = 1$, then the configuration is asymmetric. Otherwise, the configuration is said to be symmetric. Note that the function $l$ denotes the status or type of a node, i.e., $l(v)$ denotes whether the node $v$ is an empty node, a meeting node without any robot positions on it, a meeting node with a single or multiple robots on it, or a single or multiple robot positions not lying on any meeting node. We assume that the grid is embedded in the Cartesian plane. Hence, a grid can admit only three types of symmetry, namely, translation, reflection and rotation. Since the number of robots and meeting nodes is finite, translational symmetry is not possible. A unique line of symmetry characterizes a reflectional symmetry. The line of symmetry can be horizontal, vertical, or diagonal and can pass through the nodes or edges of the graph. The angle of rotation and the center of rotation characterize rotational symmetry. The angle of rotation can be $90^\circ$ or $180^\circ$, whereas the center of rotation can be a node, a center of an edge, or the center of a unit square.

- **Partitive automorphism**: Given an automorphism $\phi \in \text{Aut}(C(t), l)$, the cyclic subgroup of order $k$ generated by $\phi$ is given by $\{\phi^0, \phi^1 = \phi, \phi^2 = \phi \circ \phi, \ldots, \phi^{k-1}\}$, where $\phi^0$ denotes the identity of the cyclic subgroup. Let $H$ be any subgroup of $\text{Aut}(C(t), l)$. We define a relation $\rho$ as follows: For some $x, y \in V$, we say that $x$ and $y$ are related by the relation $\rho$ if and only if there exists an automorphism $\gamma \in H$ such that $\gamma(x) = y$. Note that the relation $\rho$ is an equivalence relation defined on the set of vertices $V$. The equivalence class of the node $x$ is defined as the orbit of $x$ and is denoted by $H(x)$. These orbits form a partition of the set $V$, since they represent disjoint equivalence classes. An automorphism $\phi \in \text{Aut}(C(t), l)$ is called partitive on $V'' \subset V$, if the cyclic subgroup $H = \{\phi^0, \phi^1 = \phi, \phi^2 = \phi \circ \phi, \ldots, \phi^{k-1}\}$ generated by $\phi$ has order $k > 1$ and is such that $|H(u)| = k$ for each $u \in V''$.

Suppose a configuration admits a unique line of symmetry $L$ such that $L$ does not pass through any node. Then, there exists an automorphism $\phi \in \text{Aut}(C(t), l)$ which is partitive on the set of
nodes $V'' = V$. The cyclic subgroup $H$ generated by $\phi$ with $k = 2$ is given by $H = \{ \phi^0, \phi^1 \}$. Similarly, assume that a configuration admits rotational symmetry where the center of rotation $c$ is not a node. If the angle of rotation is $90^\circ$, then there exists an automorphism $\phi \in \text{Aut}(C(t), l)$ which is partitive on the set of nodes $V'' = V$ and the cyclic subgroup $H$ generated by $\phi$ with $k = 4$ is given by $H = \{ \phi^0, \phi^1, \phi^2, \phi^3 \}$.

- **Configuration view**: Let $\text{MER}$ denote the minimum enclosing rectangle of $R \cup M$. $\text{MER}$ is defined as the smallest grid-aligned rectangle that contains all the robots and meeting nodes. Assume that the dimension of $\text{MER}$ is $p \times q$. We define the length of a side of $\text{MER}$ in terms of the number of grid edges on them. Let us consider the eight senary strings of length $pq$ associated with the corners of $\text{MER}$, where for each corner of $\text{MER}$, there are two senary strings defined. A senary string of length $pq$ is constructed as follows: Starting from a corner of $\text{MER}$, proceeding in the direction parallel to the width of $\text{MER}$ and scanning the entire grid sequentially, we consider all the grid lines of the $\text{MER}$ column by column. While scanning the grid, we associate $l(v)$ to each node $v$ that the string encounters. Proceeding similarly, we can define the string associated to the same corner and encounter the nodes of the grid in the direction parallel to the length of the grid. For a corner $i$, let the two strings defined are denoted by $s_{ij}$ and $s_{ik}$. Similarly, two senary strings of length $pq$ are associated with each corner of $\text{MER}$.

First, consider the case when $\text{MER}$ is a non-square rectangle. We can distinguish the two strings associated to a particular corner by considering the string which is in the direction parallel to the side of the minimum length. Consider any particular corner $i$ of $\text{MER}$. Assume that $|ij| < |ik|$. The direction parallel to $ij$ is considered as the **string direction** associated to $i$. We define $s_i = s_{ij}$ as the **string representation** associated to the corner $i$. The direction parallel to the larger side (i.e., $s_{ik}$) is defined as the **non-string direction** associated to the corner $i$. In the case of a square grid, between the two strings associated to a corner, the **string representation** is defined as the string which is lexicographically larger, i.e., $s_i = \max(s_{ij}, s_{ik})$, where the maximum is defined according to the lexicographic ordering of the strings. Note that both the strings associated with a particular corner are equal if $\text{MER}$ is symmetric with respect to the diagonal line of symmetry passing through that corner. Here, one of the directions is arbitrarily

![Diagram](image-url)

**Figure 1**: The crosses represent meeting nodes and the black circles represent robot positions. $L$ is the line of symmetry for the meeting nodes. The lexicographic largest string is $s_{AD} = 001001400004000000000040400040040100000100$. $A$ is the key corner.
selected as the string direction. If the configuration is asymmetric, we will always get a unique largest lexicographic string (In Figure 1, $s_{AD}$ is the lexicographic largest string and $AD$ is the string direction associated to $A$). Without loss of generality, let $s_i$ be the largest lexicographic string among all the strings associated to the corners of $MER$. Then we refer to $i$ as the key corner (In Figure 1, $A$ is the key corner). A corner which is not a key corner is defined as a non-key corner. The definition of the key corner is similar to one defined by Stefano et al. [7]. The configuration view of a node is defined as the tuple $(d', x)$, where $d'$ denotes the distance of a node from the key corner in the string direction and $x$ denote the status of the node, i.e., $x = l(v)$.

- Symmetry of the set $M$: We define $MER_F$ as the smallest grid-aligned rectangle that contains all the meeting nodes. Define the function $\lambda : V \rightarrow \{0, 1\}$ as follows:

$$\lambda(v) = \begin{cases} 
0 & \text{if } v \text{ is not a meeting node} \\
1 & \text{if } v \text{ is a meeting node}
\end{cases}$$

We can define a string $\alpha_i$ similar to $s_i$. The only difference is that instead of $l(v)$, each node $v$ is associated with $\lambda(v)$. If the meeting nodes are asymmetric, then there exists a unique lexicographic largest string $\alpha_i$. If the meeting nodes are not asymmetric, then the meeting nodes are said to be symmetric. The corner with which the lexicographic largest string $\alpha_i$ is associated is defined as the leading corner (In Figure 2, $\alpha_{DA} = 01000001000000010100000000001001000$ is the largest lexicographic string among the $\alpha_i's$. $D$ is the leading corner).

![Figure 2: The meeting nodes are asymmetric. D is the unique leading corner.](image)

### 3. Gathering over meeting nodes problem

In this section, we consider the problem definition for gathering. A distributed deterministic gathering algorithm for gathering $n \geq 2$ robots has been proposed. The strategy is to select a single meeting node such that all the robots agree on it and gather at that meeting node within a finite time. Since the robots are oblivious, the main objective is to maintain the invariance of the gathering meeting node during the execution of the algorithm.
3.1. Problem definition and impossibility results

In this subsection, the gathering over meeting nodes problem is formally defined in an infinite grid.

3.1.1. Problem definition:

Given a configuration \( C(t) = (R(t), M) \), the gathering over meeting nodes problem in an infinite grid asks the robots to gather at one of the meeting nodes within finite time. In an initial configuration, all the robots occupy distinct nodes of the grid. We say a configuration is final at time \( t \) if the following conditions hold:

- all the robots are on a single meeting node.
- each robot is stationary.
- any robot taking a snapshot in the look phase at time \( t \) will decide not to move.

Our proposed algorithm is a deterministic distributed algorithm that gathers all the robots at a single meeting node within a finite amount of time.

3.1.2. Partitioning of the initial configuration:

All the configurations can be partitioned into the following disjoint classes.

1. \( \mathcal{I}_1 \) – This includes the following class of configurations.
   
   (a) \( \mathcal{I}_{11} \) – \( M \) is asymmetric (Figure 2).
   
   (b) \( \mathcal{I}_{12} \) – \( M \) is symmetric with respect to a unique line of symmetry \( L \) and there exists at least one meeting node on \( L \). \( R \cup M \) is either asymmetric or symmetric with respect to \( L \) (Figure 3(a)).
   
   (c) \( \mathcal{I}_{13} \) – \( M \) is symmetric with respect to rotational symmetry with \( c \) as the center of rotation and there exists a meeting node on \( c \). \( R \cup M \) is either asymmetric or symmetric with respect to rotational symmetry (Figure 4(a)).

2. \( \mathcal{I}_2 \) – This includes the following class of configurations.
   
   (a) \( \mathcal{I}_{21} \) – \( M \) is symmetric with respect to a unique line of symmetry \( L \). \( R \cup M \) is asymmetric and there does not exist any meeting node on \( L \) (Figure 1).
   
   (b) \( \mathcal{I}_{22} \) – \( M \) is symmetric with respect to rotational symmetry. \( R \cup M \) is asymmetric and there does not exist a meeting node on \( c \).

3. \( \mathcal{I}_3 \) – This includes the following class of configurations.
   
   (a) \( \mathcal{I}_{31} \) – \( M \) is symmetric with respect to a unique line of symmetry \( L \). \( R \cup M \) is symmetric with respect to \( L \). There does not exist any meeting node on \( L \), but there exists at least one robot position on \( L \) (Figure 3(b)).
Figure 3: (a) $I_{12}$-configuration. There exist meeting nodes on $L$. (b) $I_{31}$-configuration. There exist no meeting nodes on $L$, but there exists a robot position on $L$. (c) $I_{41}$-configuration without robots or meeting nodes on $L$.

Figure 4: (a) $I_{13}$-configuration with a meeting node on $c$. (b) $I_{32}$-configuration without a meeting node on $c$, but there exists a robot position on $c$. (c) $I_{42}$-configuration without a robot or meeting node on $c$.

(b) $I_{42}$ – $M$ is symmetric with respect to rotational symmetry with $c$ as the center of rotation. $R \cup M$ is symmetric with respect to rotational symmetry. There does not exist a meeting node on $c$, but there exists a robot position on $c$. $R \cup M$ may contain either no line of symmetry or at least one line of symmetry (Figure 4(b)).

4. $I_4$ – This includes the following class of partitive configurations.

(a) $I_{41}$ – $M$ is symmetric with respect to a unique line of symmetry $L$. $R \cup M$ is symmetric with respect to $L$ and there does not exist any meeting node or robot position on $L$ (Figure 3(c)).

(b) $I_{42}$ – $M$ is symmetric with respect to rotational symmetry with $c$ as the center of rotation. $R \cup M$ is symmetric with respect to rotational symmetry and there does not exist any meeting node or robot position on $c$. $R \cup M$ may contain either no line of symmetry or at least one line of symmetry (Figure 4(c)).

Let $I$ denote the set of all initial configurations. Each time a robot is active, it observes the configuration in its Look phase and determines the current class of configuration in which it belongs without any conflict.
3.2. Impossibility result

In this subsection, we provide a sufficient condition for solving the gathering over meeting nodes problem in an infinite grid.

**Theorem 3.1.** Given an initial configuration $C(0)$, let $V' \subset V$ be a subset of nodes such that $V' \cap R(0) = \emptyset$. If there exists an automorphism $\phi$ that is partitive on the set $V \setminus V'$ and $\phi(v') = v'$, for each $v' \in V'$, then there does not exist any deterministic algorithm which can ensure gathering on a node in $V \setminus V'$.

**Proof:**

Let $H$ be the cyclic subgroup generated by $\phi$ and $k > 1$ be the size of the corresponding orbits. If possible, let algorithm $\mathcal{A}$ solve the gathering over meeting nodes problem and ensure gathering over a meeting node $m \in V \setminus V'$. This implies that starting from $C(0)$, all the robots reach a final configuration. Consider the scheduler to be fully-synchronous. Suppose, in the initial configuration, there exists a robot $r$ on a node $v \in V \setminus V'$ in the input grid graph. Since the scheduler is assumed to be fully-synchronous, all the robots in the orbit $H(v)$ are activated at the same time. As each robot in $H(v)$ has identical views, $\mathcal{A}$ cannot distinguish the robots in $H(v)$ deterministically. There exist different execution paths of the algorithm $\mathcal{A}$, but the scheduler may choose a particular execution of $\mathcal{A}$, where the destinations of each robot in $H(v)$ are the same. Since there is no robot position on $V'$, the configuration symmetry cannot be deterministically broken by allowing the robots to move from $V'$. We will prove the theorem by using induction on the number of rounds.

**Base Case:** By the assumption of the initial configuration, the configuration is partitive on the set $V \setminus V'$ at round 0.

**Inductive hypothesis:** Assume that the configuration is partitive on the set $V \setminus V'$ at round $t \geq 1$.

**Induction Step:** Let $r$ be an active robot at round $t$ that decides to move from node $v$ to node $u$. We need to prove that the configuration remains partitive on the set $V \setminus V'$ at round $t + 1$. At round $t + 1$, the following cases are to be considered.

![Figure 5](a) Each robot decides to move towards $u$ at round $t$. (b) Each robot on $H(v)$ moves towards $u$ on $L$ under the execution of $\mathcal{A}$ at round $t+1$. The configuration remains partitive after the movement. The circle on $u$ represents a multiplicity node.
1. \( v \in V \setminus V' \) and \( u \in V' \). Note that at round \( t \), the robots in \( H(v) \) have identical views and they execute the same deterministic algorithm \( A \). As a result, there exists at least one execution of \( A \) out of different execution paths of \( A \), where each robot in \( H(v) \) moves towards the same node \( u \). Each robot belonging to the other orbit \( H(v') \), where \( v' \neq v \), may move towards the same node \( u \) by the execution of \( A \). Under this execution, the configuration remains partitive on the set \( V \setminus V' \) at round \( t + 1 \) (In Figure 5(a), \( v \) is the node which is occupied by the robot \( r_1 \). Here, \( V' = L \), i.e., \( V' \) is the set of nodes belonging to the line of symmetry \( L \). In Figure 5(b), under the execution of \( A \), each robot moves towards \( u \) on \( L \)).

2. \( v \in V' \) and \( u \in V \setminus V' \). Note that, in the initial configuration, \( R(0) \cap V' = \emptyset \). Therefore, there must exist some round \( 0 < t' < t \) at which a robot \( r' \) moves from a node \( w \in V \setminus V' \) to the node \( v \in V' \). There exist different execution paths of the algorithm \( A \), but the scheduler may choose a particular execution of \( A \), where the destinations of each robot in \( H(w) \) are the same node \( v \). As a consequence, the number of robots on \( v \) at round \( t' + 1 \) is \( n = ak \), where \( a \) denotes the number of orbits (there might be different robots moving from different orbits towards \( v \)). Since each robot on \( V' \) lies on a multiplicity node \( v \), they have identical views. As the gathering must be ensured on a meeting node belonging to the set \( V \setminus V' \), there exists at least one execution of \( A \) in which \( a \) robots from \( v \) move towards \( u' \) at round \( t + 1 \), for each distinct nodes \( u' \in H(u) \). Thus, the configuration remains partitive on the set \( V \setminus V' \) at round \( t + 1 \) (In Figures 6(a) and 6(b), under the execution of \( A \), the robots on \( v \) move towards the nodes belonging to the orbit \( H(u) \) and creates multiplicity on those nodes).

![Figure 6](image-url)

**Figure 6:** (a) Each robot on \( v \) decides to move towards distinct nodes of \( H(u) \) at round \( t \). (b) The robots on \( v \) move towards distinct nodes of \( H(u) \) under the execution of \( A \) at round \( t + 1 \). The configuration remains partitive after the movement.

3. \( v \in V \setminus V' \) and \( u \in V \setminus V' \). There exists at least one execution of \( A \) in which the destinations of each robot \( r' \) on the node \( v' \) is some node \( u' \), where \( v' \in H(v) \) and \( u' \in H(u) \). Since the configuration was partitive on the set \( V \setminus V' \) at round \( t \), the configuration remains partitive on the set \( V \setminus V' \) at round \( t + 1 \) (In Figures 7(a) and 7(b), the robots on the nodes \( H(v) \) move towards the nodes belonging to \( H(u) \)).

4. \( v \in V' \) and \( u \in V' \). Note that, since in the initial configuration, \( R(0) \cap V' = \emptyset \), there must exist some round \( 0 < t' < t \) at which a robot \( r' \) moves from a node \( w \in V \setminus V' \) to the node \( v \in V' \).
There exists at least one execution of $A$, where the destinations of each robot in $H(w)$ are the same node $v$. At round $t + 1$, it might be the case that each robot on $v$ moves towards $u$, under the execution of $A$. Thus, the configuration remains partitive on the set $V \setminus V' \setminus V$ at round $t + 1$ (In Figures 8(a) and 8(b), the robots on $v$ move towards $u$).

Figure 8: (a) Each robot on $v$ decides to move towards $u$ at round $t$. (b) Each robot on $v$ moves towards $u$ at round $t + 1$. The configuration remains partitive after the movement.

Starting from $C(t)$ and with the execution of $A$, $C(t + 1)$ remains partitive on the set $V \setminus V'$ at round $t + 1$. Therefore, by the principle of mathematical induction, the configuration $C(t)$ remains partitive on the set $V \setminus V'$ at any round $t \geq 0$. Since the configuration remains partitive on the set $V \setminus V'$ at round $t + 1$, no algorithm can ensure gathering of the robots at a meeting node. In fact, to ensure gathering, there must exist a node $x \in V \setminus V'$ such that $|H(x)| = 1$, but under the execution of the algorithm $A$, the size of each orbit is $|H(x)| = k$ and $k \geq 2$, for all $x \in V \setminus V'$. This contradicts the assumption that all the robots reach a final configuration under the execution of the algorithm $A$. Thus, gathering cannot be ensured at a meeting node belonging to $V \setminus V'$.

If $C(0)$ is partitive on the node set $V \setminus V'$, then from Theorem 3.1, it follows that there must exist at least one meeting node $m \in V'$ where gathering will be finalized. In this proof, we have considered the scheduler to be fully-synchronous. Since the impossibility result holds for fully-synchronous scheduler.
and the assumption of fully-synchronous scheduler is stronger than that of asynchronous scheduler, the impossibility result holds even for asynchronous scheduler. Let $V'$ be the set of nodes on $L$, if $C(0) \in I_{41}$. Otherwise, let $V'$ be the node $\{c\}$, if $C(0) \in I_{42}$. Now, we have the following corollary:

**Corollary 3.2.** If $C(0) \in I_4$, then the gathering over meeting nodes problem is unsolvable.

**Proof:**
First, consider the case when $C(0) \in I_{41}$. This implies that $C(0)$ is partitive on the node set $V \setminus L$. According to Theorem 3.1, the gathering must be ensured at a meeting node on $L$. Since there does not exist any meeting node on $L$, gathering cannot be ensured at $L$. Therefore, the gathering over meeting nodes problem is unsolvable. The proof holds similarly in the case when $C(0) \in I_{42}$, where $C(0)$ is partitive on the node set $V \setminus \{c\}$. $\square$

**Corollary 3.3.** If an initial configuration is partitive on the set $V$, then it cannot be a final configuration.

**Proof:**
Assume to the contrary that the configuration $C(0)$ is partitive on the set $V$, and $C(t)$ can be a final configuration. This implies that starting from $C(0)$, there must exist a distributed deterministic algorithm $A$ which ensures gathering of the robots on a meeting node. First, consider the case when the configuration is symmetric with respect to a single line of symmetry $L$. Since the initial configuration is partitive on the set $V$, $L$ must be a line passing through the edges of the input grid graph. As the gathering must be ensured on a meeting node belonging to $L$, the configuration cannot be a final configuration.

Otherwise, if the configuration is symmetric with respect to rotational symmetry, the center of rotation must be a center of an edge or the center of a unit square. As the gathering must be ensured on a meeting node belonging to $c$, the configuration cannot be a final configuration. $\square$

In the rest of the paper, we assume that if a configuration admits a unique line of symmetry $L$, then $L$ passes through the nodes of the graph. Otherwise, if a configuration admits a rotational symmetry, then the center of rotation is a node. With this assumption, let $U$ denote the set of all initial configurations which are ungatherable according to Corollary 3.2. In other words, $U$ represents the following collection of the configurations.

- admitting a unique line of symmetry $L$ and no meeting nodes or robot positions on $L$.
- admitting rotational symmetry with no meeting node or robot on $c$.

4. **Algorithm**

This section describes our main algorithm $Gathering()$. The algorithm ensures gathering over a meeting node for all the initial configurations belonging to the set $I \setminus U$. The pseudo-code of the algorithm $Gathering()$ is given in Algorithm 1. We will see later that if the meeting nodes are
Algorithm 1: Gathering()

**Input:** $C(t) = (R(t), M) \in I \setminus U$

1. if $C(t) \in I_{11}$ then
   2. Each robot moves towards the meeting node $m_s$ having the highest order with respect to $O$;
3. else if $C(t) \in I_{12}$ then
   4. Each robot moves towards the meeting node $m_z$ on $L$ having the highest order with respect to $O'$;
5. else if $C(t) \in I_{13}$ then
   6. Each robot moves towards the meeting node on $c$;
7. else if $C(t) \in I_2$ then
   8. GatheringAsym();
9. else if $C(t) \in I_3$ then
10.   SymmetryBreaking();
11.   GatheringAsym();

asymmetric, then they can be ordered. Even if the meeting nodes are symmetric with respect to $L$, and there exists meeting nodes on $L$, then the meeting nodes on $L$ are orderable.

First, consider the case when the meeting nodes are asymmetric. Note that in this case, there exists a unique lexicographic largest string $\alpha_i$ (In the Figure 9, $D$ is the unique leading corner and $\alpha_{DA}$ is the unique largest lexicographic string). Consider an ordering $O$ of the meeting nodes, defined with respect to the unique leading corner. Formally, while defining the string $\alpha_i$, let $(m_1, m_2, \ldots, m_s)$ be the ordering of meeting nodes that appears in the string representation of $\alpha_i$ (In Figure 9(a), $(m_1, m_2, m_5, m_6, m_3, m_4)$ is the ordering $O$). Similarly, if the meeting nodes are symmetric with respect to a single line of symmetry $L$ and there exist meeting nodes on $L$, the meeting nodes on $L$ can be ordered according to their distances from the leading corner(s). Let $O' = (m_1, m_2, \ldots, m_z)$ be the ordering of the meeting nodes on $L$, where $z$ denote the number of meeting nodes on $L$ (In Figure 9(b), $C$ and $D$ are the leading corners. $(m_5, m_6)$ is the ordering $O'$). Hence, we have the following observations.

![Figure 9](image-url)

Figure 9: (a) The meeting nodes are asymmetric. (b) The meeting nodes are symmetric with respect to $L$, and there exists meeting nodes $m_5$ and $m_6$ on $L$.

**Observation 1.** If the meeting nodes are asymmetric, then they are orderable.

**Observation 2.** If the meeting nodes are symmetric with respect to a unique line of symmetry $L$, and there exists at least one meeting node on $L$, then the meeting nodes on $L$ are orderable.
4.1. Gathering()

In this subsection, a deterministic distributed algorithm Gathering() has been proposed to solve the gathering over meeting nodes problem in an infinite grid graph. Our proposed algorithm solves the gathering problem for all the configurations belonging to the set $\mathcal{I} \setminus \mathcal{U}$ and comprising at least two robots. The algorithm Gathering() works according to the class of configurations that each robot perceives in its local configuration view. The strategy of the algorithm is to find a single meeting node such that all the robots can agree on it and gather at that node within a finite amount of time. If $|M| = 1$, then all the robots move towards the unique meeting node and finalize the gathering. So, we assume that $|M| \geq 2$. The unique meeting node which is considered for gathering is defined as the target meeting node.

4.1.1. $\mathcal{I}_1$

Depending on whether the initial configuration $C(0)$ belongs to $\mathcal{I}_{11}$, $\mathcal{I}_{12}$ and $\mathcal{I}_{13}$, the following cases are considered:

1. $C(0) \in \mathcal{I}_{11}$. According to Observation 1, since the meeting nodes are asymmetric, they are orderable. Consider the ordering $\mathcal{O}$ of the meeting nodes, defined with respect to the unique leading corner. The meeting node $m_s$ having the highest order with respect to $\mathcal{O}$ is selected as the target meeting node. All the robots move towards $m_s$ and finalize the gathering at it (In Figure 9(a), $D$ is the leading corner and $(m_1, m_2, m_5, m_6, m_3, m_4)$ is the ordering $\mathcal{O}$. $m_4$ is the meeting node which has the highest order in $\mathcal{O}$. $m_4$ is selected as the target meeting node).

2. $C(0) \in \mathcal{I}_{12}$. There exists at least one meeting node on $L$. According to Observation 2, the meeting nodes on $L$ are orderable. Since the meeting nodes are fixed, the ordering remains invariant during the movement of the robots. Consider the ordering $\mathcal{O}' = (m_1, m_2, \ldots, m_z)$ of the meeting nodes on $L$. The meeting node $m_z$ on $L$ having the highest order with respect to $\mathcal{O}'$ is selected as the target meeting node. Each robot moves towards the meeting node $m_z$, and the gathering is finalized at $m_z$ (In Figure 9(b), $C$ and $D$ are the leading corners and $(m_5, m_6)$ is the target meeting node).

Figure 10: (a) $\mathcal{I}_{13}$-configuration with meeting node on c. $m_5$ is selected as the target meeting node. (b) The gathering is finalized by moving the robots towards $m_5$. 
the ordering $O'$. $m_6$ is the meeting node on $L$ which has the highest order among all the meeting nodes on $L$ according to $O'$. $m_6$ is selected as the target meeting node).

3. $C(0) \in I_{13}$. There exists a meeting node (say $m$) on $c$. Each robot moves towards $m$, and finalizes the gathering at $m$ (In Figure 10(a), $m_5$ is selected as the target meeting node. In Figure 10(b), each robot moves towards $m_5$).

**Lemma 4.1.** If $C(0) \in I_1$, then the target meeting node remains invariant during the execution of the algorithm Gathering() at any time $t > 0$.

**Proof:**
Depending on whether the initial configuration $C(0)$ belongs to $I_{11}$, $I_{12}$ and $I_{13}$, the following cases are considered.

**Case 1.** $C(0) \in I_{11}$. Each robot agrees on the ordering $O$ of the meeting nodes. Since the ordering remains invariant during the movement of robots, the meeting node having the highest order in $O$ also remains invariant. As a result, the target meeting node remains invariant.

**Case 2.** $C(0) \in I_{12}$. The target meeting node $m_z$ is selected as the meeting node on $L$ having the highest order with respect to $O'$. Since the ordering depends on the positions of meeting nodes, it remains invariant while the robots move towards $m_z$. Consider the case when the configuration is symmetric. Even if the configuration becomes asymmetric because of a possible pending move, $L$ remains uniquely identifiable, as it is also the line of symmetry for $M$.

**Case 3.** $C(0) \in I_{13}$. The meeting node $m$ on $c$ is selected as the target meeting node. Since $c$ is also the center of the rotational symmetry of the meeting nodes, it remains invariant while the robot moves towards it.

Hence, the target meeting nodes remains invariant during the execution of the algorithm at any time $t > 0$. ⊓⊔

**4.1.2. $I_2$**
Assume that the initial configuration $C(0) \in I_2$. In this case, the meeting nodes are symmetric, but the configuration is asymmetric. There does not exist any meeting node on $L \cup \{c\}$. Here, each robot executes GatheringAsym(). The overview of the procedure GatheringAsym() is discussed as follows.

**Overview of the procedure:** The procedure comprises the following phases: Guard Selection and Placement, Creating Multiplicity on Target Meeting Node and Finalization of Gathering. Since the robots are oblivious, each robot determines its current phase by analyzing the current configuration in its local configuration view. Note that, since the meeting nodes are symmetric, any ordering of the meeting nodes in the initial configuration depends on the robot positions and may change during the movement of the robots. In order to fix a particular ordering of the meeting nodes, a robot denoted as a guard is selected and placed in the Guard Selection and Placement phase. Each non-guard
robot moves towards the target meeting node in the Creating Multiplicity on Target Meeting Node phase. The guard is selected and placed in such a way that during the execution of the procedure GatheringAsym(), it remains uniquely identifiable by the other robots. The main strategy of the algorithm is to maintain the invariance of the target meeting node in the Creating Multiplicity on Target Meeting Node phase. Finally, in the Finalization of Gathering phase, the guard moves towards the target meeting node. While the guard moves, it moves in a shortest path.

Guard Selection and Placement: In this phase, a single robot is selected as the guard. The guard is selected and placed in such a way that it remains uniquely identifiable by the other robots during the execution of the procedure GatheringAsym(). Let MERF denote the minimum enclosing rectangle of all the meeting nodes. First, assume that the meeting nodes are symmetric with respect to a unique line of symmetry L, and there does not exist any meeting node on L. Since the configuration is asymmetric, there always exists a unique key corner. As a result, a unique robot with the maximum configuration view exists. Let $d_1$ denote the maximum distance between a meeting node from L. Similarly, let $d_2$ denote the maximum distance between a robot position from L. Next, we consider the following cases.

1. There exists at least one robot position outside the rectangle MERF. This implies that there exists at least one robot position at a distance $d_2 > d_1$ from L. If there are multiple robots at a distance $d_2$, consider the robot with the maximum configuration view. Let $r$ be the robot with the maximum configuration view. $r$ is selected as the guard and it moves towards an adjacent node away from L. This movement results in creating a unique robot which is at the maximum distance from L. (In Figures 11(a) and 11(b), $r_2$ and $r_7$ are the robots outside the MERF and at the farthest distance from L. $r_7$ is the robot with the maximum configuration view as B is the key corner. $r_7$ move towards an adjacent node).

2. Each robot position is either inside or on the rectangle MERF. This implies that any robot can be at the maximum distance $d_2$ from L and $d_2 \leq d_1$. Consider the robot farthest from L. If there are multiple such robots, consider the robot $r$ with the maximum configuration view. $r$ is selected as the guard and it moves towards an adjacent node away from L. $r$ continues its

---

**Figure 11:** (a) $A'B'C'D'$ is the minimum enclosing rectangle MERF of all meeting nodes. ABCD is the MER. (b) $r_7$ is selected as a guard and moves towards an adjacent node away from L. Finally, it moves towards its closest corner. The transformed MER is $A''BCD''$. 
movement and the moment \( r \) reaches a node which is outside the rectangle \( MER_F \), \( d_2 \) becomes greater than \( d_1 \). The rest of the procedure follows similarly from the previous case.

Once the \textit{guard} becomes the unique farthest robot from \( L \), it moves towards the closest corner of \( MER \) in the direction parallel to \( L \). If the \textit{guard} is closest to two corners of \( MER \), then it moves towards an arbitrary corner (In Figure 11(b), \( r_7 \) moves towards \( D'' \)). The procedure follows similarly when the \textit{meeting node} admits rotational symmetry, and there does not exist any \textit{meeting node} on \( c \). In that case, \( d_1 \) and \( d_2 \) are defined as the distances measured from \( c \). The pseudo-code corresponding to this phase is given in Algorithm 2.

\begin{algorithm}
\hspace*{10pt} Input: \( C(t) = (R(t), M) \)
\begin{algorithmic}[1]
\State \textbf{if} there exists at least one robot position outside \( MER_F \) \textbf{then}
\State \hspace{1em} \textbf{if} there exists exactly one robot position \( r \) outside \( MER_F \) \textbf{then}
\State \hspace{2em} \( r \) is selected as the \textit{guard} \;
\State \hspace{1em} \textbf{else}
\State \hspace{2em} The robot \( r \) farthest from \( L \cup \{c\} \) and with the maximum configuration view is selected as the \textit{guard} \;
\State \hspace{2em} \textit{r} moves towards an adjacent node away from \( L \cup \{c\} \) and finally towards its closest corner;
\State \textbf{else if} each robot is inside or on \( MER_F \) \textbf{then}
\State \hspace{1em} \textbf{if} there exists a unique robot \( r \) farthest from \( L \cup \{c\} \) \textbf{then}
\State \hspace{2em} \( r \) is selected as the \textit{guard} \;
\State \hspace{1em} \textbf{else}
\State \hspace{2em} The robot \( r \) farthest from \( L \cup \{c\} \) and with the maximum configuration view is selected as the \textit{guard} \;
\State \hspace{1em} \( r \) moves towards an adjacent node away from \( L \cup \{c\} \) and continues its movement unless it is outside \( MER_F \);
\end{algorithmic}
\end{algorithm}

\textbf{Lemma 4.2.} During the execution of the procedure \textit{GuardSelection()}, the \textit{guard} remains uniquely identifiable by the robots.

\textbf{Proof:}
First, assume that the \textit{meeting nodes} are symmetric with respect to \( L \) and there does not exist any \textit{meeting node} on \( L \). The proof follows similarly when the configuration admits rotational symmetry, and there does not exist any \textit{meeting node} on \( c \). The following cases are to be considered.

\textbf{Case 1.} There exists at least one robot position outside the rectangle \( MER_F \). Note that there may be multiple such robots. If there exists precisely one such robot \( r \), then according to the procedure \textit{GuardSelection()}, \( r \) is selected as a \textit{guard}. Otherwise, if there are multiple such robots, the robot \( r \) with the maximum configuration view is selected as the \textit{guard}. \( r \) moves towards an adjacent node \( v \) away from \( L \). The moment it reaches \( v \), it becomes the unique farthest robot from \( L \) and at least at a distance \( d_2 + 1 \) from \( L \). While the \textit{guard} moves towards the corner, it remains the unique farthest robot from \( L \). As the \textit{guard} is selected as the unique farthest robot from \( L \), it remains uniquely identifiable.

\textbf{Case 2.} Each robot position is inside or on the rectangle \( MER_F \). In this case, the robot position farthest from \( L \) is selected as a \textit{guard}. Note that there may be multiple such robots. The procedure
GuardSelection() ensures that the guard is selected as the robot $r$ with the maximum configuration view. The moment $r$ moves towards an adjacent node away from $L$, it becomes the unique farthest robot from $L$. $r$ continues its movement unless it becomes the unique robot that is outside the rectangle $MER_F$. The rest of the proof follows from Case 1.

According to Lemma 4.2, the guard is the unique robot which is farthest from $L \cup \{c\}$. As a result, the guard will not have any symmetric image with respect to $L \cup \{c\}$, and the configuration remains asymmetric. Once the guard is selected and placed, we consider the corner of $MER$, which is occupied by the guard. Starting from that corner, we scan the entire $MER$ in the direction parallel to the string direction and associate each node $v$ to $\lambda(v)$. As a result, we would get a binary string. Consider the ordering of the meeting nodes according to their positions in the string representation. We define the particular ordering by $O''$.

Creating Multiplicity on Target Meeting Node: In this phase, each non-guard moves towards the target meeting node $m$. Since $R \cup M$ is asymmetric, there exists a unique ordering of the meeting nodes with respect to the guard. Note that the ordering remains invariant unless the guard moves. Each robot agrees on the ordering $O''$ of the meeting nodes. The target meeting node $m$ is selected as the meeting node that is closest to the guard. If there are multiple such meeting nodes, consider the meeting node $m$ that has the minimum order in $O''$ as the target meeting node. Let $n_1$ denote the total number of distinct robot positions. If $n_1 \geq 3$, then each non-guard moves towards $m$ by executing MakeMultiplicity() (In Figures 12(a) and 12(b), $m_1$ is the closest meeting node from the guard. Each non-guard moves towards $m_1$). This would result in creating a robot multiplicity at $m$. Note that during this movement, the robots may create multiplicity on a meeting node other than $m$. We have to ensure that, during this phase, $m$ remains invariant and uniquely identifiable. All the non-guard robots move towards $m$ sequentially, i.e., the non-guard robots that are closest to $m$ move first. The pseudo-code corresponding to this phase is given in Algorithm 3.

Lemma 4.3. During the execution of MakeMultiplicity(), the target meeting node remains invariant.
Algorithm 3: MakeMultiplicity()

Input: \( C(t) = (R(t), M) \)

1. if there exists a unique meeting node \( m \) closest to the guard then
   2. Each robot selects \( m \) as the target meeting node;
   3. else
      4. Let \( m \) be the closest meeting node that has the minimum order in \( O'' \);
      5. Each robot selects \( m \) as the target meeting node;
      6. Let \( r \) be a closest non-guard robot which is not on \( m \);
      7. \( r \) moves towards \( m \) in a shortest path;

Proof:

Since the configuration is asymmetric, there exists a unique ordering \( O'' \) of the meeting nodes with respect to the unique guard. Note that during the execution of MakeMultiplicity(), the guard does not move. The following cases are to be considered.

Case 1. The meeting nodes are symmetric with respect to \( L \). If there exists a unique meeting node \( m \) closest to the guard, it is selected as the target meeting node. Since the position of the meeting nodes and the guard remains fixed during the execution of MakeMultiplicity(), \( m \) remains invariant. Otherwise, there are multiple closest meeting nodes from the guard. In that case, the target meeting node \( m \) is selected as the meeting node closest to the guard and that has the minimum order in \( O'' \). Since the guard does not move, the ordering \( O'' \) remains invariant. Hence, \( m \) remains invariant during the execution of MakeMultiplicity().

Case 2. The meeting nodes are symmetric with respect to rotational symmetry. The target meeting node is selected similarly, as in Case 1. The rest of the proof follows similarly.

Finalization of Gathering: In this phase, the guard executes GuardMovement(). Note that the robots are endowed with local-weak multiplicity detection capability. If \( n_1 = 2 \), then the guard can identify that it does not lie on a robot multiplicity node and on a meeting node. The guard would start moving towards the other robot position in a shortest path. All the other robots on the target meeting

Figure 13: (a) The configuration after the execution of MakeMultiplicity(). (b) \( r_7 \) moves towards \( m_1 \) and finalizes the gathering. The MER becomes \( A'B'C'D' \).
node $m$ would identify that they are already on a multiplicity node and they would not move. As a result, the robots on $m$ would remain on $m$. Since the target meeting node $m$ is selected as one of the meeting nodes closest to the guard, while the guard moves towards $m$ in a shortest path, it would not lie on any meeting node other than $m$ in its movement path. Eventually, the guard would finalize the gathering on $m$ (In Figures 13(a) and 13(b), $r_7$ observes that is not on a multiplicity node. It would move towards $m_1$). The pseudo-code corresponding to this phase is given in Algorithm 4.

**Algorithm 4:** GuardMovement()

Input: $C(t) = (R(t), M)$

1. if $n_1 = 2$ then
2. Let $r$ be the robot that does not lie on a robot multiplicity node and on a meeting node;
3. $r$ moves towards the other robot position;

4.1.3. $\mathcal{I}_3$

This subsection considers all initial configurations belonging to $\mathcal{I}_3$. Each robot executes SymmetryBreaking(). The algorithm description of the procedure SymmetryBreaking() is as follows.

![Figure 14](image-url)

Figure 14: (a) $\mathcal{I}_{31}$-configuration. (b) Transformation of $\mathcal{I}_{31}$-configuration into an asymmetric configuration by the movement of the robot $r_4$.

**Symmetry Breaking:** In this phase, all the symmetric configurations that can be transformed into asymmetric configurations are considered. A unique robot is identified that allows the transformation. We have the following cases.

1. $C(t) \in \mathcal{I}_{31}$. In this class of configurations, at least one robot exists on $L$. Let $r$ be the robot on $L$ having the maximum configuration view. $r$ moves towards an adjacent node that does not belong to $L$ (In Figures 14(a) and 14(b), $C$ and $D$ are the key corners. As a result, $r_4$ is the unique robot on $L$ that has the maximum configuration view. $r_4$ moves towards an adjacent node away from $L$, and the configuration becomes asymmetric). Note that the configuration may not belong to $\mathcal{I}_{21}$ as it may contain multiplicities.
2. $C(t) \in \mathcal{I}_{32}$. In this class of configurations, there exists a robot (say $r$) on $c$. The robot $r$ moves towards an adjacent node (In Figures 15(a) and 15(b), the robot $r_9$ on $c$ moves towards an adjacent node and the configuration transforms into an asymmetric configuration). If the configuration admits rotational symmetry with multiple lines of symmetry and there is a robot $r$ at the center, $r$ moves towards an adjacent node. This movement creates a unique line of symmetry $L'$. However, the new position of $r$ might have a multiplicity. If that happens to be the robot with the maximum view on $L'$, moving robots from there might still result in a configuration with a line of symmetry. Even so, the unique line of symmetry $L'$ would still contain at least one robot position without multiplicity, and the number of robot positions on $L'$ will be strictly less than the number of robots on the line of symmetry in the original configuration. Thus, the repeated execution of the procedure $SymmetryBreaking()$ guarantees to transform the configuration into an asymmetric configuration within a finite amount of time.

The pseudo-code corresponding to this phase is given in Algorithm 5. Once the configuration is transformed into an asymmetric configuration, the robots execute $GatheringAsym()$. Suppose a robot multiplicity is created during the execution of $SymmetryBreaking()$. In that case, the robot that is farthest from $L \cup \{c\}$ and does not lie on a multiplicity is selected as the guard. If there are multiple such farthest robots, then the robot with a higher configuration view is selected as a guard. Note that such a robot position always exists.

**Algorithm 5: SymmetryBreaking()**

```
Input: $C(t) \in \mathcal{I}_3$
1 if $C(t) \in \mathcal{I}_{31}$ then
2     Let $r$ be the robot on $L$ with the maximum configuration view;
3     Move $r$ towards any adjacent node that does not belong to $L$;
4 else if $C(t) \in \mathcal{I}_{32}$ then
5     Let $r$ be the robot on $c$;
6     Move $r$ towards any adjacent node;
```
Lemma 4.4. If $C(0) \in \mathcal{I} \setminus \mathcal{U}$, then during the execution of $Gathering()$, $C(t) \notin \mathcal{U}$, for any $t > 0$.

Proof:
According to Theorem 3.2, the ungatherable configurations are those configurations

1. admitting a unique line of symmetry $L$ and without any robot or meeting nodes on $L$.
2. admitting rotational symmetry with center of rotation $c$ and without a robot or meeting node on $c$.

Depending on whether the initial configuration $C(0)$ is in $\mathcal{I}_1$, $\mathcal{I}_2$, or $\mathcal{I}_3$, the following cases are to be considered.

Case 1. Consider the case when $C(0) \in \mathcal{I}_1$. This includes all those configurations where the meeting nodes are either asymmetric or symmetric with at least one meeting node on $L \cup \{c\}$. Since the meeting nodes are fixed nodes located on the nodes of the grid, $C(t) \notin \mathcal{U}$, for any $t > 0$.

Case 2. Consider the case when $C(0) \in \mathcal{I}_2$. Since the configuration is asymmetric, there exists a unique key corner. According to Lemma 4.2, during the execution of the procedure GuardSelection, the configuration remains asymmetric, as the guard contains no symmetric image with respect to $L \cup \{c\}$. Note that the guard is the unique farthest robot from $L \cup \{c\}$. Since the guard does not move in the Creating Multiplicity on Target Meeting node phase and each non-guards moves towards MER, the configuration remains asymmetric at any time $t > 0$. During the execution of GuardMovement(), the guard moves towards the target meeting node and finalizes the gathering. Hence, $C(t') \notin \mathcal{U}$, for any $t' > t$, where $t'$ denotes any instant of time after the execution of GuardMovement().

Case 3. Consider the case when $C(0) \in \mathcal{I}_3$. There exists at least one robot position on $L \cup \{c\}$. At some time $t > 0$, the configuration transforms into an asymmetric configuration by the execution of SymmetryBreaking(). Hence, the configuration remains asymmetric at $t' > t$, similar as in Case 2.

Hence, if $C(0) \notin \mathcal{U}$, then during the execution of $Gathering()$, $C(t) \notin \mathcal{U}$, for any $t > 0$. \hfill \qed

Without loss of generality, let $m$ be the target meeting node, selected after guards placement at time $t$. Let $d(t) = \sum_{r_i \in R(t)} d(r_i(t), m)$.

Theorem 4.5. If $C(0) \in \mathcal{I} \setminus \mathcal{U}$ with $n \geq 2$, then by the execution of the algorithm $Gathering()$, the gathering over meeting nodes problem is solved within finite time.

Proof:
According to Lemma 4.4, if the initial configuration $C(0) \notin \mathcal{U}$, then $C(t) \notin \mathcal{U}$, for any $t > 0$. Depending on whether the initial configuration $C(0)$ is in $\mathcal{I}_1$, $\mathcal{I}_2$, or $\mathcal{I}_3$, the following cases are to be considered.

Case 1. $C(t) \in \mathcal{I}_1$. According to Lemma 4.1, the target meeting node $m$ remains invariant. Let $t' > t$ be an arbitrary point of time at which at least one robot starts moving towards $m$. Therefore, $d(t') = \sum_{r_i \in R(t')} d(r_i(t'), m)$ and $d(t') < d(t)$. This implies that eventually, all the robots will reach $m$ and the gathering is finalized at $m$ within a finite amount of time.
Case 2. $C(t) \in \mathcal{I}_2$. First, consider the execution of $\text{MakeMultiplicity}()$. Let $t' > t$ be an arbitrary point of time after the guard selection and placement phase. Assume that at least one non-guard robot has completed its LCM cycle at $t'$. We have $d(t') = \sum_{r_i \in R(t')} d(r_i(t'), m)$. According to Lemma 4.3, the target meeting node remains invariant. If there is at most one robot position which is not on $m$ at time $t'$, then the execution of $\text{GuardMovement}()$ is started. Otherwise, let $r$ be any non-guard robot which has computed its LCM cycle at time $t'$. Since $r$ has moved at least one node closer to $m$, we have $d(t') < d(t)$. This implies that eventually, all the non-guard robots will reach $m$ and execution of $\text{GuardMovement}()$ will be started.

Next, we consider the execution of procedure $\text{GuardMovement}()$. Assume that at time $t''$, the guard (say $r$) starts moving towards $m$ in a shortest path. At $t''$, $d(t'') = d(r(t''), m)$ (All other robots are already on $m$). Since $r$ would move at least one node closer to $m$, $d(t_1) < d(t'')$ at $t_1 > t''$. Let $t_2 > t_1 > t''$ be the point of time at which $r$ reaches $m$. As $d(r(t_2), m) = 0$, $d(t_2) = 0$. Therefore, eventually all the robots will finalize gathering at $m$.

Case 3. $C(t) \in \mathcal{I}_3$. In this case, $\text{SymmetryBreaking}()$ is executed. The transformed configuration becomes either asymmetric, or it may admit a unique line of symmetry. By the repeated execution of $\text{SymmetryBreaking}()$, the configuration becomes asymmetric. The rest of the proof follows from Case 2.

Hence, execution of the algorithm $\text{Gathering}()$ eventually solves the gathering over meeting nodes problem within finite time. $\square$

5. Lower bounds

In this section, we study the efficiency of our algorithm in terms of the total number of moves executed by the robots. Let $n$ denote the total number of robots deployed on the nodes of the input grid graph. Consider a configuration $C(t)$, where the dimension of MER is $1 \times (n+1)$. Assume that there is a single meeting node which is placed at one of the corners of MER, and all the other $n$ nodes of the grid are occupied by the $n$ robots (In Figure 16, $m$ is the single meeting node and $MER = AB$).

Since the gathering problem requires all the $n$ robots to be placed at the unique meeting node, the total number of moves executed by the robots is given by, $1 + 2 + 3 + \ldots + n = \frac{n(n-1)}{2}$. Hence, any algorithm solving the gathering over meeting nodes problem requires $\Omega(n^2)$ moves.

Figure 16: A configuration showing the lower bound in terms of the number of movements required to finalize the gathering.

Next, assume that $D = \max\{p, q\}$, where $p$ and $q$ are the dimensions of the initial MER and if $D = \Theta(n)$, then any algorithm solves the gathering problem in $\Omega(Dn)$ moves. Hence, we have the following theorem.
Theorem 5.1. Any gathering algorithm for solving the gathering over meeting nodes problem requires \( \Omega(Dn) \) moves.

5.1. Analysis of the algorithm gathering()

• During the Symmetry Breaking phase, only \( O(1) \) moves are required in order to break the symmetry.

• In the Guard Selection and Placement phase, at most, one robot may be required to move away from the initial MER. The total number of moves during this phase is \( O(D) \).

• In the Creating Multiplicity on Target Meeting Node phase, all the non-guard robots move towards the target meeting node in a shortest path. The total number of moves in this phase is \( O(Dn) \).

• Finally, in the Finalization of Gathering phase, only the guard moves towards the target meeting node. The number of moves in this phase is \( O(D) \).

Hence, we have the following result.

Theorem 5.2. Algorithm Gathering() solves the gathering over meeting nodes problem in \( O(Dn) \) moves if the initial configuration belongs to the set \( \mathcal{I} \setminus \mathcal{U} \).

6. Conclusion

In this work, we have studied the gathering over meeting nodes problem in an infinite grid where the robots have local-weak multiplicity detection capability. A subset of all the initial configurations has been proved to be ungatherable. A deterministic distributed algorithm for solving the gathering problem has been proposed for the remaining configurations with \( n \geq 2 \), where \( n \) is the number of robots in the system. We have discussed the efficiency of the proposed algorithm in terms of the total number of moves executed by the robots. We have proved that the algorithm solves the gathering over meeting nodes problem in \( O(Dn) \) moves, where \( D \) is the length of the longer side of the initial MER.

Future Works: One immediate future direction of work would be to consider the optimal algorithms for gathering. The optimal criterion is to minimize the maximum distance traveled by any robot. Another direction of future interest would be to consider a randomized algorithm for breaking the symmetry when there are no robots or meeting nodes on the line of symmetry and the center of rotation. It would be interesting to consider gathering algorithms without any multiplicity detection capability. We may also consider the gathering problem for the configurations, where the initial configuration may contain multiplicities. If the initial deployment has multiple robots in some nodes, the problem requires a different solution.
References

[1] Cieliebak M, Flocchini P, Prencipe G, Santoro N. Solving the Robots Gathering Problem. In: Baeten JCM, Lenstra JK, Parrow J, Woeginger GJ (eds.), Automata, Languages and Programming, 30th International Colloquium, ICALP 2003, Eindhoven, The Netherlands, June 30 - July 4, 2003. Proceedings, volume 2719 of Lecture Notes in Computer Science. Springer, 2003 pp. 1181–1196. doi:10.1007/3-540-45061-0_90. URL https://doi.org/10.1007/3-540-45061-0_90.

[2] Agmon N, Peleg D. Fault-tolerant gathering algorithms for autonomous mobile robots. In: Munro JI (ed.), Proceedings of the Fifteenth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2004, New Orleans, Louisiana, USA, January 11-14, 2004. SIAM, 2004 pp. 1070–1078. URL http://dl.acm.org/citation.cfm?id=982792.982951.

[3] Bhagat S, Chaudhuri SG, Mukhopadhyaya K. Fault-tolerant gathering of asynchronous oblivious mobile robots under one-axis agreement. J. Discrete Algorithms, 2016. 36:50–62. doi:10.1016/j.jda.2015.10.005. URL https://doi.org/10.1016/j.jda.2015.10.005.

[4] Klasing R, Markou E, Pelc A. Gathering asynchronous oblivious mobile robots in a ring. Theor. Comput. Sci., 2008. 390(1):27–39. doi:10.1016/j.tcs.2007.09.032. URL https://doi.org/10.1016/j.tcs.2007.09.032.

[5] Klasing R, Kosowski A, Navarra A. Taking advantage of symmetries: Gathering of many asynchronous oblivious robots on a ring. Theor. Comput. Sci., 2010. 411(34-36):3235–3246. doi:10.1016/j.tcs.2010.05.020. URL https://doi.org/10.1016/j.tcs.2010.05.020.

[6] D’Angelo G, Stefano GD, Klasing R, Navarra A. Gathering of robots on anonymous grids and trees without multiplicity detection. Theor. Comput. Sci., 2016. 610:158–168. doi:10.1016/j.tcs.2014.06.045. URL https://doi.org/10.1016/j.tcs.2014.06.045.

[7] Stefano GD, Navarra A. Gathering of oblivious robots on infinite grids with minimum traveled distance. Inf. Comput., 2017. 254:377–391. doi:10.1016/j.ic.2016.09.004. URL https://doi.org/10.1016/j.ic.2016.09.004.

[8] Cicerone S, Stefano GD, Navarra A. Gathering of robots on meeting-points: feasibility and optimal resolution algorithms. Distributed Computing, 2018. 31(1):1–50. doi:10.1007/s00446-017-0293-3. URL https://doi.org/10.1007/s00446-017-0293-3.

[9] Pattanayak D, Mondal K, Ramesh H, Mandal PS. Gathering of mobile robots with weak multiplicity detection in presence of crash-faults. J. Parallel Distributed Comput., 2019. 123:145–155. doi:10.1016/j.jpdc.2018.09.015. URL https://doi.org/10.1016/j.jpdc.2018.09.015.

[10] Cicerone S, Stefano GD, Navarra A. Asynchronous Arbitrary Pattern Formation: the effects of a rigorous approach. Distributed Computing, 2019. 32(2):91–132. doi:10.1007/s00446-018-0325-7. URL https://doi.org/10.1007/s00446-018-0325-7.

[11] Barberá HM, Quiñonero JPC, Zamora-Izquierdo MA, Gómez-Skarmeta AF. i-Fork: a flexible AGV system using topological and grid maps. In: Proceedings of the 2003 IEEE International Conference on Robotics and Automation, ICRA 2003, September 14-19, 2003, Taipei, Taiwan. IEEE, 2003 pp. 2147–2152. doi:10.1109/ROBOT.2003.1241911. URL https://doi.org/10.1109/ROBOT.2003.1241911.

[12] Sharma G, Dutta A, Kim J. Optimal Online Coverage Path Planning with Energy Constraints. In: Elkind E, Veloso M, Agmon N, Taylor ME (eds.), Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems, AAMAS ’19, Montreal, QC, Canada, May 13-17, 2019. International Foundation for Autonomous Agents and Multiagent Systems, 2019 pp. 1189–1197. URL http://dl.acm.org/citation.cfm?id=3331820.
[13] Flocchini P. Gathering. In: Flocchini P, Prencipe G, Santoro N (eds.), Distributed Computing by Mobile Entities, Current Research in Moving and Computing, volume 11340 of Lecture Notes in Computer Science, pp. 63–82. Springer, 2019. doi:10.1007/978-3-030-11072-7_4. URL https://doi.org/10.1007/978-3-030-11072-7_4

[14] Flocchini P, Prencipe G, Santoro N. Distributed Computing by Oblivious Mobile Robots. Synthesis Lectures on Distributed Computing Theory. Morgan & Claypool Publishers, 2012. doi:10.2200/S00440ED1V01Y201208DCT010. URL https://doi.org/10.2200/S00440ED1V01Y201208DCT010

[15] D’Angelo G, Stefano GD, Navarra A. Gathering six oblivious robots on anonymous symmetric rings. J. Discrete Algorithms, 2014. 26:16–27. doi:10.1016/j.jda.2013.09.006. URL https://doi.org/10.1016/j.jda.2013.09.006

[16] D’Angelo G, Stefano GD, Navarra A. Gathering on rings under the Look-Compute-Move model. Distributed Comput., 2014. 27(4):255–285. doi:10.1007/s00444-014-0212-9. URL https://doi.org/10.1007/s00444-014-0212-9

[17] Izumi T, Izumi T, Kamei S, Ooshita F. Mobile Robots Gathering Algorithm with Local Weak Multiplicity in Rings. In: Patt-Shamir B, Ekim T (eds.), Structural Information and Communication Complexity. Springer Berlin Heidelberg, Berlin, Heidelberg. ISBN 978-3-642-13284-1, 2010 pp. 101–113.

[18] Kamei S, Lamani A, Ooshita F, Tixeuil S. Gathering an Even Number of Robots in an Odd Ring without Global Multiplicity Detection. In: Rovan B, Sassone V, Widmayer P (eds.), Mathematical Foundations of Computer Science 2012. Springer Berlin Heidelberg, Berlin, Heidelberg. ISBN 978-3-642-32589-2, 2012 pp. 542–553.

[19] Cicerone S, Stefano GD, Navarra A. Gathering robots in graphs: The central role of synchronicity. Theor. Comput. Sci., 2021. 849:99–120. doi:10.1016/j.tcs.2020.10.011. URL https://doi.org/10.1016/j.tcs.2020.10.011

[20] Cicerone S, Stefano GD, Navarra A. Gathering Synchronous Robots in Graphs: From General Properties to Dense and Symmetric Topologies. In: Censor-Hillel K, Flammini M (eds.), Structural Information and Communication Complexity - 26th International Colloquium, SIROCCO 2019, L’Aquila, Italy, July 1-4, 2019, Proceedings, volume 11639 of Lecture Notes in Computer Science. Springer, 2019 pp. 170–184. doi:10.1007/978-3-030-24922-9_12. URL https://doi.org/10.1007/978-3-030-24922-9_12

[21] Cicerone S, Stefano GD, Navarra A. On Gathering of Semi-synchronous Robots in Graphs. In: Ghaffari M, Nesterenko M, Tixeuil S, Tucci S, Yamauchi Y (eds.), Stabilization, Safety, and Security of Distributed Systems - 21st International Symposium, SSS 2019, Pisa, Italy, October 22-25, 2019, Proceedings, volume 11914 of Lecture Notes in Computer Science. Springer, 2019 pp. 84–98. doi:10.1007/978-3-030-34992-9_7. URL https://doi.org/10.1007/978-3-030-34992-9_7

[22] Bhagat S, Chakraborty A, Das B, Mukhopadhyaya K. Gathering over Meeting Nodes in Infinite Grid. In: Changat M, Das S (eds.), Algorithms and Discrete Applied Mathematics - 6th International Conference, CALDAM 2020, Hyderabad, India, February 13-15, 2020, Proceedings, volume 12016 of Lecture Notes in Computer Science. Springer, 2020 pp. 318–330. doi:10.1007/978-3-030-39219-2_26. URL https://doi.org/10.1007/978-3-030-39219-2_26

[23] Bose K, Kundu MK, Adhikary R, Sau B. Optimal Gathering by Asynchronous Oblivious Robots in Hypercubes. In: Gilbert S, Hughes D, Krishnamachari B (eds.), Algorithms for Sensor Systems. Springer International Publishing, Cham. ISBN 978-3-030-14094-6, 2019 pp. 102–117.
[24] Stefano GD, Navarra A. Optimal gathering of oblivious robots in anonymous graphs and its application on trees and rings. *Distributed Computing*, 2017. **30**(2):75–86. doi:10.1007/s00446-016-0278-7. URL https://doi.org/10.1007/s00446-016-0278-7

[25] Fujinaga N, Ono H, Kijima S, Yamashita M. Pattern Formation through Optimum Matching by Oblivious CORDA Robots. In: Lu C, Masuzawa T, Mosbah M (eds.), Principles of Distributed Systems. Springer Berlin Heidelberg, Berlin, Heidelberg. ISBN 978-3-642-17653-1, 2010 pp. 1–15.

[26] Cicerone S, Di Stefano G, Navarra A. Embedded pattern formation by asynchronous robots without chirality. *Distributed Computing*, 2019. **32**(4):291–315. doi:10.1007/s00446-018-0333-7. URL https://doi.org/10.1007/s00446-018-0333-7

[27] Bhagat S, Das B, Chakraborty A, Mukhopadhyaya K. k-Circle Formation and k-epf by Asynchronous Robots. *Algorithms*, 2021. **14**(2):62. doi:10.3390/a14020062. URL https://doi.org/10.3390/a14020062

[28] Das B, Chakraborty A, Bhagat S, Mukhopadhyaya K. k-Circle formation by disoriented asynchronous robots. *Theor. Comput. Sci.*, 2022. **916**:40–61. doi:10.1016/j.tcs.2022.03.003. URL https://doi.org/10.1016/j.tcs.2022.03.003