Gathering at Fixed Nodes by Oblivious Mobile Robots

Abhinav Chakraborty

Indian Statistical Institute

June 2023

Indian Statistical Institute

Doctoral Thesis

Gathering at Fixed Nodes by Oblivious Mobile Robots

Author:
Abhinav Chakraborty

Supervisor:
Prof. Krishnendu Mukhopadhyaya

A thesis submitted to the Indian Statistical Institute
in partial fulfilment of the requirements for the degree of
Doctor of Philosophy in Computer Science

Advanced Computing and Microelectronics Unit

Indian Statistical Institute, Kolkata

June 2023

Dedicated to
My parents

Acknowledgements

The contributions of the people who supported, encouraged, and believed in me during the process of writing my thesis cannot be adequately expressed in a few words of gratitude. However, I would like to use this chance to mention and acknowledge some of the people who have constantly supported and motivated me throughout this journey.

First, I would like to express my sincere gratitude to my supervisor, Prof. Krishnendu Mukhopadhyaya whom I had the privilege of working with. I am immensely obliged for all his help, guidance and motivation. For the past five years, his discussions with my thesis-related work have been incredibly appreciative. He offered me the flexibility to pursue the research interests, I was interested in and directed me in the right places. I am enormously grateful to him for teaching me the fundamental steps for conducting independent research.

Next, I wish to offer my gratitude to all my co-authors, including Dr. Subhash Bhagat, Mr. Bibhuti Das and Dr. Kaustav Bose. I am fortunate to have Mr. Bibhuti Das as my friend, brother and well-wisher. His encouragement, inspiration, and support are huge factors in helping me finish my thesis. He has been a huge help to me personally as well. If there is a solution to every difficulty, then without collaborating with Dr. Kaustav Bose, it would not have been simple to identify the answers to the problem. Throughout this journey, his profound insights have been a huge assistance to me in comprehending the fundamental ideas of the domain. I am fortunate to have Dr. Subhash Bhagat as one of my co-authors. His capability of designing problems is highly appreciable.

I wish to offer my gratitude to my coursework teachers, including Prof. Sandip Das, Prof. Arijit Bishnu, Prof. Subhas C. Nandy, Prof. Susmita Sur-Kolay, Prof. Ansuman Banerjee, Prof. Sourav Chakraborty, Prof. Arijit Ghosh, Prof. Goutam Paul, Prof. Subhamoy Maitra. I consider myself extremely fortunate to had such mentors during my Ph.D. journey. I would like to thank other professors, Prof. Nabanita Das, Prof. Sasanka Roy, Prof. Arijit Chakrabarty and Prof. Sasthi C. Ghosh for their expert guidance and support during the journey. I was fortunate enough to work under Prof. Pratima Panigrahi at IIT Kharagpur during my M.Tech days as a part of the project thesis.

Next, I would like to mention my joyful lab mates like Bibhuti, Rathindra, Dr. Kaustabha Ray, Dr. Avirup Das, Subhojit and the newcomer Mayank. Without their support, this journey would have been more difficult. I would like to provide a special thanks to Rathindra for his support. I was fortunate enough to have such a friend on my journey. His constant support helps me a lot in dealing with my personal matters. His vast

knowledge of the subject is quite appreciable. I was extremely lucky to have friends in ISI Kolkata like Sankar, Sucheta, Animesh, Sukriti, Anjan, Joginder, Prasun Da, Anupam, Arun, Atanu, Drimit, Suman, Suraj, Debendranath and Surochita. I would like to thank them for cheering me in my hard times. I would like to thank my seniors, Dr. Dibakar Saha, Dr. Sanjana Dey, Subhadeep Da and Dr. Shankar K. Ghosh for their support in the initial days of my Ph.D. journey. Special thanks to Dr. Ranendu Adhikary for his endless discussions and constant support. I would like to thank my junior Debasmita for her motivation and for discussing doubts during the coursework journey. I am extremely lucky to have juniors like Chandrima, Parnashree, Chandan, Banashree, Lakshmikanta, etc. I want to express my gratitude to the whole ACM unit office staff for providing me with useful information throughout the years. I am grateful to the Indian Statistical Institute, Kolkata for allowing me to pursue my Ph.D. degree.

I would like to mention Aritra, Monalisa, Dipanjana, Sudip, Amit Singh, Arindam, Ananya, Anwesha, Rana, Neha, Partha, Soumyarup, Poulami, Salil, among others with whom I have shared many beautiful memories that I will treasure forever in my graduation and master days. The most memorable times I've had with my friends were with Manish, Tamal, Debajyoti, Rohit, and Rabi. Special gratitude to Debajyoti, Tamal, and Manish for being there for me during my hard times. The support and encouragement they provided to me throughout my Ph.D. journey is something I will never forget. I would like to thank Dipsa Banerjee for providing me with mental support during my difficult times.

I want to express my gratitude to my parents, Mr. Ashok Chakraborty and Mrs. Sumita Chakraborty, for giving me the best possible upbringing, serving as my first teachers and inspiring me. This journey would be impossible without their efforts. It would have been challenging for me to pursue my academic career without their support. I was fortunate to have siblings like Epsha, Sanchari Di, Sagnika Di, Rick and Toya. I would like to thank Sanchita Roy, my aunt, for helping me make the right decisions in my life. I was fortunate enough to have Dr. Abhishek Chakraborty as my brother and Teerna Chakraborty as my sister-in-law. From time to time, I have immensely benefited while consulting them. I will never forget the sacrifices that my brother made. I do always remember his sacrifices and the way he handled the entire family in our difficult times. Last but not the least, I would mention my niece Shrinika Chakraborty as a blessing in my life. Her adorable smile brightens my day and motivates me to be a better human being.

Abhinav Chakraborty
10.01.2023

Abstract

A swarm of robots is a distributed system of small, autonomous, inexpensive mobile robots that work collaboratively to achieve some specific goal. This thesis aims to study the gathering problem, which is a fundamental coordination problem. In the initial configuration, the robots are located at distinct nodes of an input graph. The main objective of the problem is that all the robots must meet at a point. In this thesis, the deployment space of the robots is assumed to be an infinite grid graph.

This thesis considers the gathering over meeting nodes problem in an infinite grid domain. The input grid graph consists of some nodes which are marked as meeting nodes. The problem requires the mobile robots to gather at one of the pre-fixed meeting nodes. Several variations of the problem have been considered. 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 problem has also been studied under the objective function that minimizes the total number of moves made by all the robots to finalize the gathering task. A deterministic distributed algorithm has been proposed to solve the problem for all those solvable configurations, and the initial configurations for which the problem is unsolvable have been characterized. The proposed gathering algorithm is optimal with respect to the total number of moves performed by all the robots in order to finalize the gathering.

Another variation of the problem considers two disjoint and finite groups of anonymous robots. The initial configurations also consist of two finite and disjoint sets of heterogeneous meeting nodes. The objective is to design a distributed algorithm that gathers all the robots belonging to the first team at one of the meeting nodes belonging to the first type. Similarly, all the robots in the second team must gather at one of the meeting nodes belonging to the second type. The initial configurations for which the gathering problem is unsolvable have been characterized. For the remaining initial configurations, a distributed gathering algorithm has been proposed.

The parking problem can also be considered as a variation of the problem. As a solution to the parking problem, the robots need to partition themselves into groups so that each parking node contains a number of robots equal to the capacity of the node. It is assumed that the number of robots in the initial configuration represents the sum of the capacities of the parking nodes. Under this setup, we have characterized all the initial configurations and the values of k_i for which the problem is unsolvable. For the remaining configurations, a deterministic distributed algorithm was proposed.

List of Publications

Publications from the Thesis:

Journal Publications

1. Subhash Bhagat, **Abhinav Chakraborty**, Bibhuti Das, Krishnendu Mukhopadhyaya: Gathering over Meeting Nodes in Infinite Grid. Fundamenta Informaticae 187(1): 1-30 (2022)

   doi: 10.3233/FI-222128

2. Subhash Bhagat, **Abhinav Chakraborty**, Bibhuti Das, Krishnendu Mukhopadhyaya: Optimal Gathering over Meeting Nodes in Infinite Grid. International Journal of Foundations of Computer Science Vol. 34, No. 01, pp. 25-49 (2023).

   doi: 10.1142/S0129054122500174

3. **Abhinav Chakraborty**, Subhash Bhagat and Krishnendu Mukhopadhyaya: Gathering over Heterogeneous Meeting Nodes (Under Minor Revision). The Computer Journal.

   Manuscript Number: COMPJ-2022-03-0216.R1

   Date of Submission: 30/03/2022

Conference Publications

1. Subhash Bhagat, **Abhinav Chakraborty**, Bibhuti Das, Krishnendu Mukhopadhyaya: Gathering over Meeting Nodes in Infinite Grid. Algorithms and Discrete Applied Mathematics - 6th International Conference, CALDAM 2020, Hyderabad, India, February 13-15, 2020, Proceedings, Pages: 318-330

   doi: 10.1007/978-3-030-39219-2_26

2. **Abhinav Chakraborty**, Krishnendu Mukhopadhyaya: Pattern Formation Problems for Mobile Robots. ICDCN '22: 23rd International Conference on Distributed Computing and Networking, Delhi, AA, India, January 4 - 7, 2022, Pages: 236-237

   doi: 10.1145/3491003.3491301

Other Publications (Not part of the Thesis)

1. Subhash Bhagat, Bibhuti Das, **Abhinav Chakraborty**, Krishnendu Mukhopadhyaya: k-Circle Formation and k-epf by Asynchronous Robots. Algorithms 14(2): 62 (2021)

   doi: 10.3390/a14020062

2. Bibhuti Das, **Abhinav Chakraborty**, Subhash Bhagat, Krishnendu Mukhopadhyaya: k-Circle formation by disoriented asynchronous robots. Theor. Comput. Sci. 916: 40-61 (2022)

   doi: 10.1016/j.tcs.2022.03.003

3. Kaustav Bose, **Abhinav Chakraborty**, Krishnendu Mukhopadhyaya: Mutual Visibility by Fat Robots with Slim Omnidirectional Camera (Accepted)

   Manuscript Number: JPDC-D-22-00507

   Date of Acceptance: 16/05/23

   doi: 10.1016/j.jpdc.2023.104716

Contents

List of Figures

List of Tables

List of Algorithms

Chapter 1

Introduction

A *swarm of robots* is a collection of identical, small robots that work collaboratively to perform certain tasks. The study of *swarm robotics* aims to use a system of relatively small generic robots that can work together to execute complex tasks. It draws inspiration from the collective behavior of biological entities such as birds, ants, termites and bees. The tasks are executed without any centralized control. Each individual in this computational system has limited abilities to accomplish the desired goals. Since every robot is equipped with the same abilities, any entity in the system can be replaced by another. As a result, the system becomes scalable and appropriate for use in fault-prone environments. The computational aspects of these systems have been the main focus of the research. The emphasis has been on what the robots should have as minimal capabilities to solve a problem.

1.1 Motivation

Over the past two decades, there has been a significant increase in the study of co-operative behaviors of multi-robot systems. A common motivation behind building autonomous multi-robot systems is the need to perform different tasks in adverse situations where human intervention is not possible. A practical substitute for using a single powerful and expensive robot is a swarm of cheap, weak, simple robots. Compared to a traditional single robot, a swarm of robots has several benefits. Manufacturing a single powerful robot with a complicated structure and control modules can be expensive.

However, the components of an individual swarm robotic system, being very simple and generic, can be mass-produced and inexpensive. Even a sizable swarm of these robots could be significantly less expensive than a single powerful robot. Below, we list several task domains where swarm robotics could be used.

- The use of swarm robotics is suited for tasks that require patrolling a large area of space, for example, environmental monitoring. Their sensing abilities can aid in providing surveillance for quick detection of hazardous events, such as in cases when chemicals are accidentally leaked [118].
- Swarm robotics systems are suited for tasks that require low-cost designs, such as mining tasks or agricultural foraging tasks [118].
- They can be used for search and rescue missions to provide aid during natural disasters [115].
- On the battlefield, a swarm of robots can be used to build dynamic communication networks. When some of the communication nodes are hit by enemy fire, these networks can benefit from the robustness attained through re-configuring the communication nodes [109].

The subject of multi-robot systems was first studied by Suzuki et al. in the papers [116, 117], with a focus on the study of pattern formation problems. In these papers, the investigations were carried out from a computational point of view. The purpose of theoretical studies on the computational aspects of robot swarms is primarily to determine the minimal capabilities that robots need to solve a given problem. The emphasis in such works is on rigorous mathematical proofs instead of experimental or empirical evidence. In this thesis, we pursue this line of research and study the *gathering* problem in the discrete domain, which is a fundamental coordination problem.

1.2 Computational Entities

The computational entities are usually called robots. Each robot is capable of performing local computations. The robots are endowed with motorial capabilities that allow them to move. In the general model, the robots are assumed to be *autonomous*, i.e., there is no centralized control. The robots are assumed to be *anonymous*, i.e., there are no unique identifiers that they can use during computation. They are *identical* in the sense that they are indistinguishable by their appearances. They are also assumed to be *homogeneous*, i.e., they run the same algorithm and have the same capabilities. There is no global coordinate system available for the robots. Each robot has its own unit of length, an origin assumed to be the current position of the robot and an *ego-centric* Cartesian coordinate system defined by two perpendicular axes, namely X and Y axis.

1.3 Deployment Space

In general, there are mainly two settings for the spatial universe $\mathcal{U}$, where the robots operate and move: *discrete* and *continuous*. In the discrete setting, the robots are usually deployed at the nodes of an undirected and unlabeled graph. This setting mainly describes the case of mobile agents in a communication network. In the continuous universe, the robots are represented as points in the d dimensional Euclidean plane. In this thesis, we have assumed that the robots are deployed at the nodes of an infinite grid graph, where they move along the edges of the graph.

1.4 Agreement in Coordinate System

The robots do not have any agreement on the global coordinate system. Each robot has its own local coordinate system, which is ego-centric, i.e., the origin is at the current location of the robot. The robots have their own notion of unit length. The unit distance of a robot may be different from the others. However, the robots may have full or partial agreement on the direction and orientation of the coordinate axes. The following models are considered depending on the extent of consistency among local coordinate systems [62].

- **Two-axis agreement:** The robots agree on the direction and orientation of both the axes (Figure 1.1).

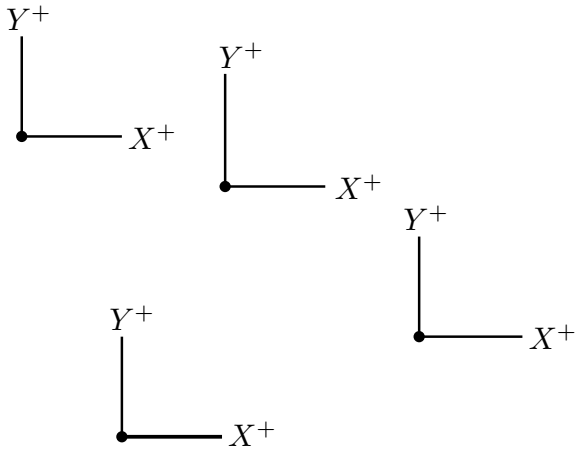

FIGURE 1.1: The robots have full axis agreement

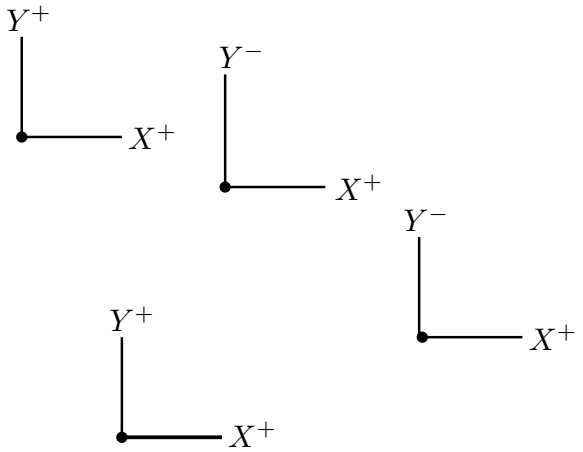

FIGURE 1.2: One-axis agreement

- **One axis agreement:** The robots agree on the direction and orientation of only one axis. The other coordinate axes may have different orientations for different robots. In Figure 1.2, the robots agree on the direction and orientation of the X-axis; different robots have different positive orientations of the positive $Y-$ axis.

- **Direction-only:** The robots agree on a common direction for both local coordinate axes. However, unlike in the case where the robots agree on the orientation

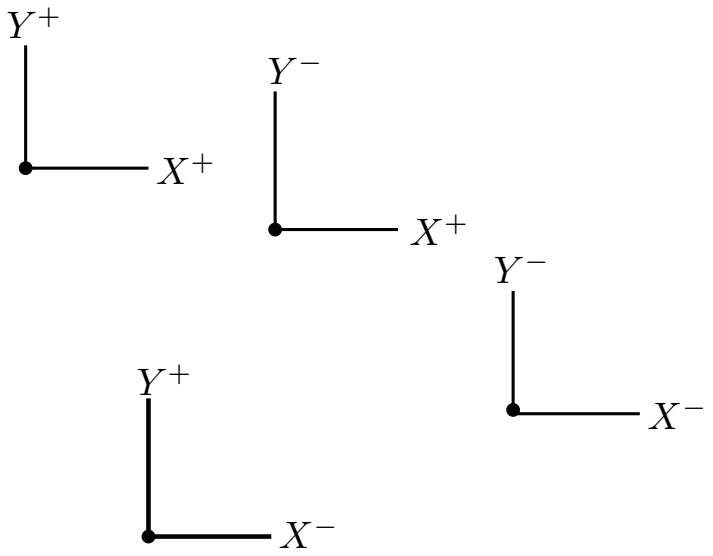

FIGURE 1.3: Direction-Only

of one of the axes, the positive orientations of the axes may not be the same in the case where the robots have direction-only agreement (Figure 1.3).

- **Axes-only:** The robots agree on the directions of both local coordinate axes. However, the positive orientations of the axes are not common and the robots do not agree on which of the two axes is the $X-$ axis and which is the $Y-$ axis (Figure 1.4).

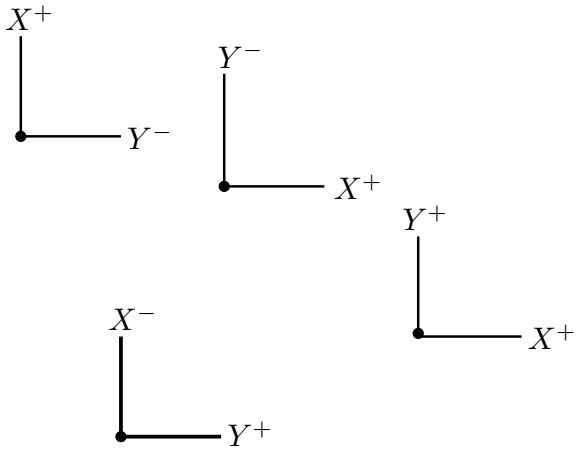

FIGURE 1.4: Axes-Only

- **Chirality:** The robots agree on a common cyclic orientation, i.e., they agree on the common handedness (clockwise and anticlockwise directions) (Figure 1.5).

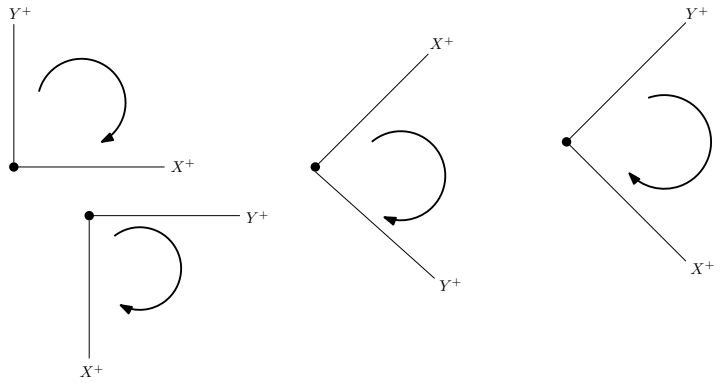

FIGURE 1.5: The robots agree on a common clockwise direction

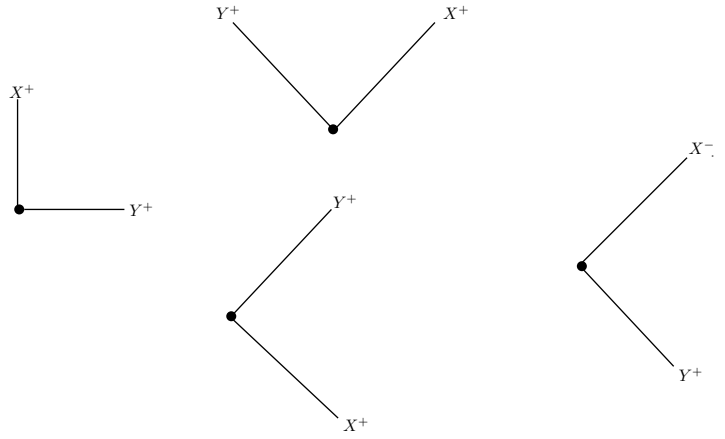

FIGURE 1.6: The robots are disoriented

- **Disoriented:** The robots do not have agreement on the common direction and orientations of the coordinate systems (Figure 1.6). In the context of the thesis, it has been assumed that the robots are disoriented.

1.5 Robots with or without Extent

Generally, in the continuous domain, the robots are viewed as points, i.e., they are assumed to be dimensionless. For $d \geq 3$, the robots may also be considered as a unit disc in the Euclidean plane or a unit ball in the d dimensional Euclidean space. The robots with extent are usually termed as *fat robots* [26, 27, 43]. However, in the discrete domain, the extent does not carry much relevance. The only assumption is that a robot fits into a node and can move along the edges of the input graph. In this thesis, we have assumed that the robots are dimensionless.

1.6 Visibility

Each robot can observe its surroundings using its visibility sensors. This enables the robots to share information implicitly among themselves. There are three main types of visibility models that have been considered in the literature.

- **Unlimited Visibility:** The most commonly used model is the *unlimited visibility model* or *full visibility model*. In this model, each robot is able to observe all the other robots.
- **Limited Visibility:** The robots can sense the presence of other robots within a predefined limited visibility range. This range is called the *visibility radius* of the robots. It is generally assumed that the visibility radius is the same for all robots. In Figure 1.7, r can see all the robots within the visibility range of γ.

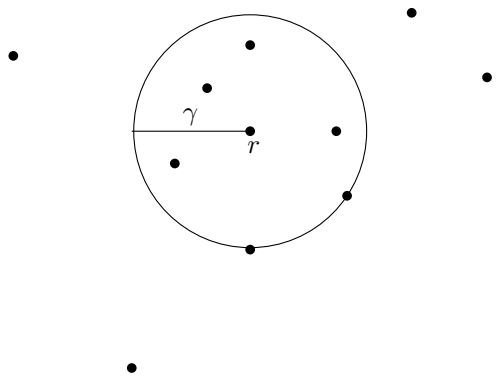

FIGURE 1.7: Limited Range Visibility Model

- **Obstructed Visibility:** In the *obstructed visibility* or *opaque robot model*, the visibility range of a robot is usually unlimited, but its visibility is obstructed by the presence of the other robots, i.e., if three opaque robots are collinear, the middle robot obstructs the vision of other two robots. In Figure 1.8, r_3 obstructs the vision of r_1 and r_4. r_2 cannot see r_5 due to the presence of r_3. However, r_1 can see r_2, r_3 and r_5. r_2 can see r_3 and r_4. r_3 can see all the other robots. r_4 and r_5 can see each other.

In this thesis, it has been assumed that the robots have unlimited visibility.

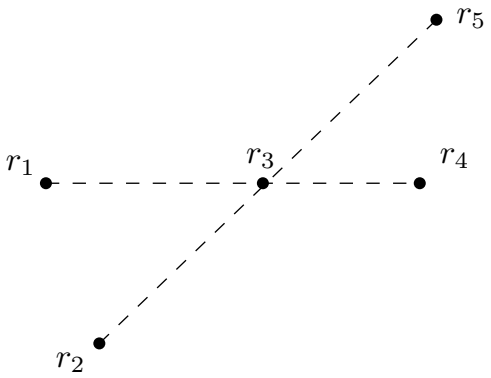

FIGURE 1.8: Obstructed Visibility Model

1.7 Look-Compute-Move cycle

At any point of time, each robot is either active or inactive. An active robot operates in *Look-Compute-Move* (LCM) cycle. Once a computational cycle is over, the robot may become inactive or may again start a fresh computational cycle. A robot executes the computational cycle repeatedly.

- During the *Look* phase, a robot uses its visibility sensors to take a snapshot of the spatial universe. This phase is instantaneous, and the outcome is a snapshot of the positions of the other robots in their own local coordinate system.
- Based on the perceived configuration, a robot computes a destination point in the *compute* phase. The algorithm is assumed to be the same for all robots.
- In the *Move* phase, a robot moves towards the computed destination point. If the destination point is the current location of the robot, the robot stays still and performs a null movement.

1.8 Activation Schedule

Three types of schedulers are mainly considered in the literature with regard to the activation schedule of the robots and the timing of the operations within their cycles.

- In the *fully-synchronous* ($\mathcal{FSYNC}$) setting, time can be logically divided into non-overlapping global rounds. In each round, all the robots are activated. They take

the snapshots at the same time and then perform their moves simultaneously. In Figure 1.9, the robots r_1, r_2 and r_3 are activated simultaneously.

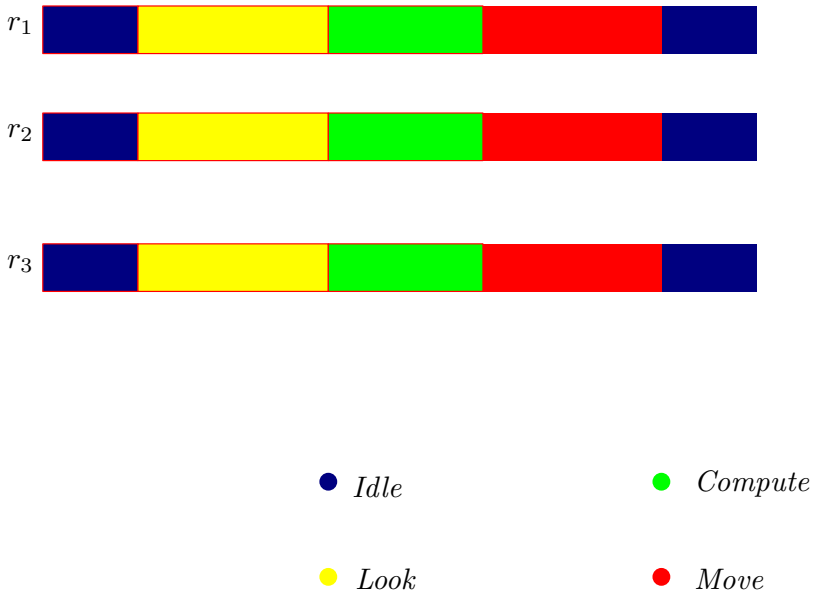

FIGURE 1.9: Fully-synchronous scheduler

- In the *semi-synchronous* ($\mathcal{SSYNC}$) setting, a subset of robots are activated at the same time and all operations are instantaneous. As a consequence, a robot is not observed by the other robots when it is in motion. In Figure 1.10, the robots r_1 and r_2 are activated simultaneously.

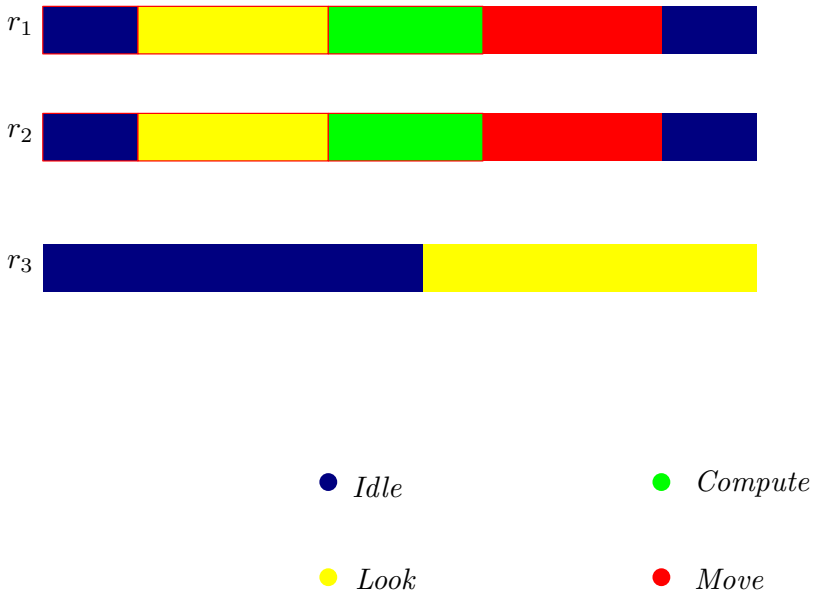

FIGURE 1.10: Semi-synchronous scheduler

- The most general model is the *asynchronous* ($\mathcal{ASYNC}$) setting. In this setting, the activation of a robot and its execution of the LCM cycle are independent. The amount of time spent in Look, Compute, Move and inactive states are finite but unbounded and unpredictable. As a result, robots do not have a common notion of time. Additionally, a robot can be seen while it is in motion, which enables computations to be made using obsolete information about its position. It should also be noted that the configuration perceived by a robot during the look phase may significantly change before it initiates a movement. In this setting, the scheduler is assumed to be fair, i.e., each robot is activated an infinite number of times. In Figure 1.11, each robot r_1, r_2 and r_3 are activated independently. In this thesis, the scheduler has been assumed to be asynchronous.

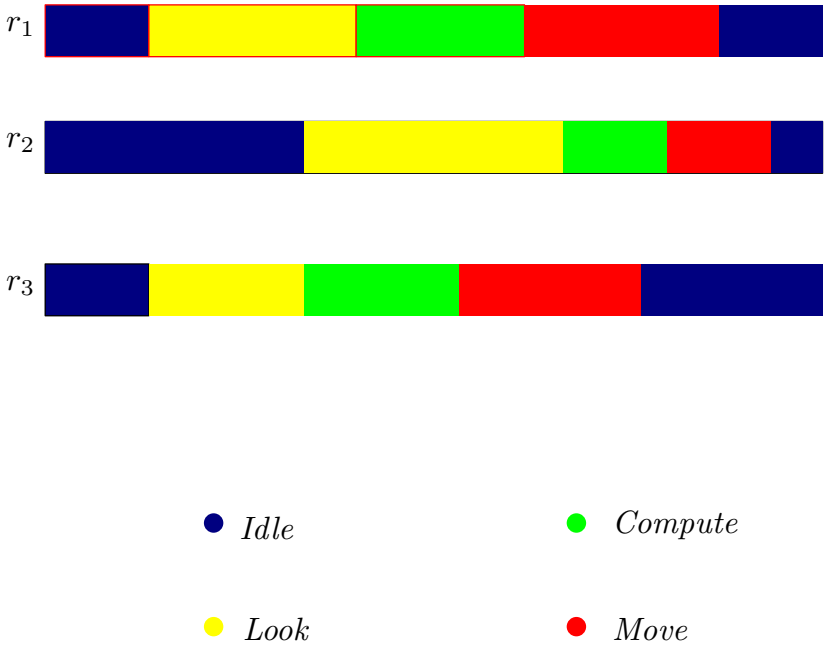

FIGURE 1.11: Asynchronous scheduler

1.9 Memory

Based on memory and communication capabilities, four models are generally used in the literature:s

- $\mathcal{OBLOT}$: The most general and well-studied model is the $\mathcal{OBLOT}$ model. In this standard model, the robots are assumed to be *silent* and *oblivious*. The robots

are silent in the sense that there is no explicit communication of information. The robots are oblivious in the sense that their memory is volatile. They do not have any memory of past observations, computations and actions. At the end of each Look-Compute-Move cycle, the memory is erased. Therefore, the computation in each cycle is based solely on what is observed in the current cycle [25, 37, 55].

- $\mathcal{LUMI}$: In this model, the robots are endowed with externally visible lights. These lights have a finite number of predetermined colors. Each color indicates a different state of the robot. While the robots are oblivious and their memories are erased in the next cycle, the color of the light is retained in the next cycle [53, 71, 95]. The lights serve both as a weak explicit communication mechanism and a form of memory.

- $\mathcal{FSTA}$: In this model, the robots are silent but have finite memory. This model is similar to the $\mathcal{LUMI}$ model. The only difference is that here the light of a robot is only visible to itself. In other words, in the Look phase, a robot observes the positions of robots visible to it but not their lights. However, it can see the color of its own light [9, 70, 71].

- $\mathcal{FCOM}$: In this model, the robots are oblivious but have finite communication bits. This model is also similar to the $\mathcal{LUMI}$ model. The difference here is that the light of a robot is only visible to others. A robot cannot observe the color of its own light [70, 71].

In this thesis, we have considered the $\mathcal{OBLOT}$ model.

1.10 Multiplicity Detection Capability

Multiple robots may occupy a single point when they are depicted as points. A *multiplicity point* is a location where multiple robots are present. If the robots are endowed with *multiplicity detection capability*, then they can detect whether there exists a multiplicity at a given point. Based on the multiplicity detection capabilities of the robots, the following models are considered:

- **Global-weak multiplicity detection capability**: This is the capability of a robot by which it can only detect whether multiple robots occupy a point. They

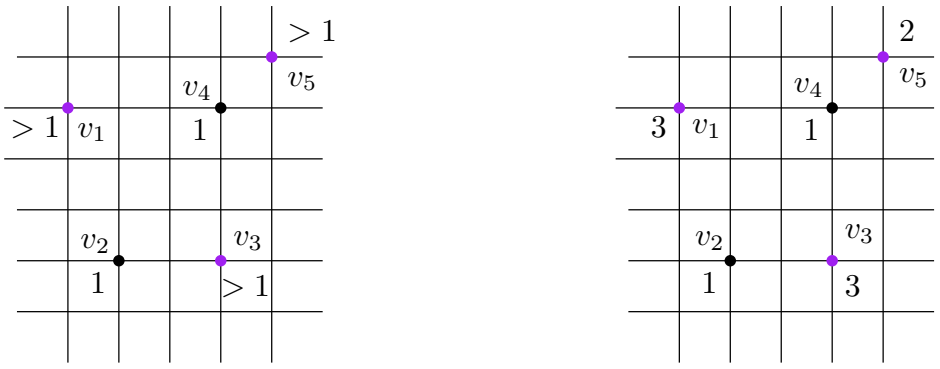

FIGURE 1.12: The nodes with purple colored implies multiplicity (a) Global-weak multiplicity detection (b) Global-strong multiplicity detection

are unable to count the exact number of robots that make up the multiplicity. If the robots have *local-weak multiplicity detection* capability rather than the global version, they can detect whether there is a multiplicity at the point where it currently resides. In Figure 1.12 (a), each robot can observe that the nodes v_1, v_3 and v_5 contain multiplicity if they are endowed with global-weak multiplicity detection capability.

- **Global-strong multiplicity detection capability**: This is the capability of a robot that allows it to count the exact number of robots composing a multiplicity. If the robots have the local version of strong multiplicity detection capability rather than the global version, they can only detect the number of robots that make up the multiplicity in their current location. In Figure 1.12 (b), each robot can observe that the nodes v_1, v_3 and v_5 contain exactly 3, 3 and 2 robots, respectively, if they are endowed with global-strong multiplicity detection capability.

In chapters 3 and 5 of the thesis, it has been assumed that the robots have weak multiplicity detection capacity. However, in chapters 4 and 6 of the thesis, the robots are equipped with global-strong multiplicity detection capability.

1.11 Movement

In the continuous domain, the robots are assumed to move along straight lines. Nevertheless, certain models also allow robots to travel along some curved trajectories [32, 103].

The movement of a robot may terminate before reaching its destination in the continuous domain. Based on this, there are mainly two models that have been considered:

- **Rigid Motion:** In this model, each robot is able to reach its destination without halting in between [37].
- **Non-rigid motion:** In this model, the movement of a robot may be stopped by the adversary before it reaches its destination. However, it is assumed that there is a distance $\delta > 0$ such that whenever a robot moves, it moves at least a distance δ towards its destination point, provided that the destination is at least δ distance away from the current position of the robot. Note that this assumption guarantees finite time reachability of the robot towards its destination point. The value of δ may not be known to the robots.

In the discrete domain, the movement of the robots is restricted along the edges of the graph and at any instant of time, a robot can move toward one of its adjacent nodes. In this thesis, since the robots are deployed at the nodes of an infinite grid graph, we assume that the movement of a robot is instantaneous. This implies that the robots cannot be seen while moving, i.e., the robots can be seen only at the nodes of the graph. In such a context, an instantaneous move also implies rigid movements.

1.12 Fundamental Geometric Problems

A large volume of works concerning different geometrical problems have been studied in the literature. One of the key issues in the field of swarm robotics research is the coordination of robots in distributed environments. The following are a few well-studied geometric problems for swarm robots:

1.12.1 Geometric Problems in Continuous Domain

- **Gathering:** In the initial configuration, the robots are located at distinct points on the plane. The main objective of the *gathering* problem is that all the robots must meet at a point. This point is not known to them a priori. This task must be accomplished within a finite amount of time [35, 38, 68, 106, 117]. A closely

related problem to the gathering problem is the *convergence* problem, where the objective is that the robots must be as close as possible instead of meeting at a particular point [5, 38, 39]. Another problem closely related to the gathering is the *near-gathering problem*, where the robots are required to be within a circle of a predetermined radius [102].

- **Scattering and Covering:** The *scattering* problem aims at arranging the robots such that no two robots share the same position in the final configuration [58, 79]. A more specific version of the scattering problem is the *covering problem*, where the objective is to disperse the robots across a bounded region of space in order to *cover* the region. There may be certain optimization criteria involved. As an example, *uniform covering* of a region requires that the region be equipartition, with each partition being assigned a robot. In order to achieve *maximum coverage* for a given *sensing range* v, the robots must scatter in a manner that maximizes the region covered by circles of radius v centered at the location of the robot.

- **Pattern Formation:** In the initial configuration, the robots are usually assumed to be at distinct positions on the plane. The *pattern formation* problem requires the robots to arrange themselves in space so as to form a pattern given as input [69, 116, 117, 122]. The pattern is generally given as a set of points in space or a geometric predicate like circle, square, etc.

- **Mutual Visibility:** The *Mutual visibility* problem requires the robots lying on the plane to attain a configuration in which robots occupy distinct locations on the plane, and no three robots are collinear [95, 96]. This problem considers opaque robots that block the vision of other robots through them.

- **Flocking:** The *flocking* problem requires the robots to move together, forming a pattern and maintaining it while moving [23]. The *guided flocking* problem is a variation of the flocking problem in which the movement of the robots is directed by an exogenous source, which is a distinct mobile entity. This entity is generally referred to as the *leader* or *guide* [99, 120, 121].

1.12.2 Geometric Problems in Discrete Domain

- **Gathering:** In the initial configuration, the robots are located at distinct nodes of an anonymous graph. The main objective of the *gathering* problem is that all the robots must meet at one of the nodes of the graph. In the discrete domain, the problem has been mostly considered in rings, grids and trees [87, 88, 113, 47, 114].

- **Pattern Formation:** In the initial configuration, the robots are assumed to be at the distinct nodes of an anonymous graph. Numerous works related to the pattern formation problem have also been considered in the grid topology, where the robots usually operate on a two-dimensional square grid [18, 30, 92, 94].

- **Mutual Visibility:** Mutual visibility in graphs refers to the concept of determining whether two nodes in a graph are mutually visible to each other. In the final configuration, the robots must attain a configuration in which they occupy distinct nodes of the graph and no three robots are collinear. In the discrete domain, the problem has been mainly studied in square grids [1, 76, 110, 111].

- **Graph Exploration:** The *graph exploration* problem [7, 8, 64, 93] is mainly of two types: *Terminating* and *Exclusive Perpetual*. The terminating variant of the exploration problem requires that each node of the input graph must be visited by at least one robot, and eventually, all the robots stop moving. The exclusive perpetual variant of the gathering problem requires that each robot must visit every vertex of the graph infinitely often. Furthermore, no two robots visit the same vertex or travel along the same edge at the same time.

- **Dispersion:** In the dispersion problem, the initial configuration involves placing the robots at the nodes of an anonymous graph with n nodes. The objective is to reposition a subset of $k \leq n$ robots such that each robot ends up at distinct nodes of the graph [6, 90].

1.13 Scope of the Thesis

This thesis aims to study the gathering problem in an infinite grid domain. The deployment space of the robots is assumed to be an infinite grid graph, where the robots are located at the nodes of the graph. The problem has been largely studied in the

continuous domain, where the robots are represented as either points or fat disks in $\mathbb{R}^2$. 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 [32, 103]. 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 collisions. 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. A robot can travel an amount of distance, which may be an irrational number. In practice, performing such infinitesimal movements with infinite precision may not always be possible. 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. Gathering [47, 87, 88, 113] 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. In this thesis, we have studied the *restricted gathering* model, where the gathering is restricted to some specific nodes. These nodes are some prefixed nodes located at the nodes of an infinite grid graph, given as input, and are visible to each robot in the initial configuration.

The *gathering over meeting nodes* problem has been studied in this thesis under different sets of assumptions. We have identified the set of all unsolvable configurations under different settings. For the remaining configurations, the deterministic distributed algorithms are proposed for the different problems considered in the thesis. We provide correctness and finite time termination of the proposed algorithms. The following problems have been addressed in this thesis:

1.13.1 Gathering over Meeting Nodes in Infinite Grid

This work considers gathering over meeting nodes problem in an infinite grid by asynchronous oblivious mobile robots. In this work, the robots are deployed at the nodes of an infinite grid graph, which has a subset of nodes marked as meeting nodes. The robots are assumed to be identical, autonomous, anonymous and oblivious. They operate under an asynchronous scheduler. They do not have any agreement on a *global*

coordinate system. However, the robots are endowed with local-weak multiplicity detection capability. We have shown that even if the robots are endowed with multiplicity detection capability, some configurations remain ungatherable. We have proposed a deterministic distributed algorithm to solve the gathering problem for the remaining configurations. The efficiency of the proposed algorithm in terms of the total number of moves executed by the robots has been studied. 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. The proposed algorithm requires $O(Dn)$ moves, i.e., the algorithm is asymptotically optimal. We have also proved that the proposed algorithm requires $O(D)$ epochs to terminate, where an epoch is the smallest interval of time within which each robot is guaranteed to execute its LCM cycle at least once. The lower bound regarding the total number of epochs required to terminate is $O(D)$, which proves that the proposed algorithm is asymptotically time optimal. The formal definition of the problem is as follows:

Problem 1. *Let $\mathcal{R}(t)$ denote the multiset of robot positions at time t and $\mathcal{M}$ be the set of meeting nodes located at the nodes of the grid. Given a configuration $C(t) = (\mathcal{R}(t), \mathcal{M})$, the gathering over meeting nodes problem in an infinite grid asks the robots to gather at one of the meeting nodes within a finite time, i.e., in the final configuration all the robots are on a single meeting node.*

1.13.2 Optimal Gathering over Weber Meeting Nodes in Infinite Grid

Efficient solutions satisfying a particular optimization criterion are preferable in terms of cost and better implementation. This work considers the *Optimal Gathering over Weber Meeting nodes* problem in infinite grids. The initial configuration comprises at least seven robots that are deployed at the nodes of an infinite grid. The optimization criterion studied in this work is the minimization of the total number of moves made by the robots to finalize the gathering. A meeting node that minimizes the sum of the distances from all the robots is termed as a *Weber meeting node*. In our restricted gathering model, the robots must gather at one of the Weber meeting nodes to ensure gathering with a minimum number of moves. We have shown that there exist some configurations where gathering over Weber meeting nodes cannot be ensured, even if

the robots are endowed with global-strong multiplicity detection capability. It has been shown that, without the assumption of global-strong multiplicity detection capability, there are configurations where gathering cannot be accomplished as soon as a multiplicity is created. However, there are initial configurations where gathering can be ensured over a meeting node but not on the set of Weber meeting nodes. This includes the configuration admitting a single line of symmetry without any robots or Weber meeting nodes on the reflection axis, but at least one meeting node exists on the reflection axis. In that case, the feasibility of the problem has been studied. The formal definition of the problem is as follows:

Problem 2. *Let $C(t) = (\mathcal{R}(t), \mathcal{M})$ be a given configuration. The optimal gathering over Weber meeting nodes problem asks the robots to gather at one of the meeting nodes of the initial configuration such that the total number of moves made by the robots is minimized, i.e., in the final configuration, all the robots are on a single Weber meeting node.*

1.13.3 Gathering over Heterogeneous Meeting Nodes

It may be possible that in many real-life applications, different tasks are required to be performed simultaneously by different groups of robots. This work considers the *gathering over heterogeneous meeting nodes problem* in infinite grids. Two finite and disjoint types of homogeneous robots are deployed at the nodes of an infinite grid graph. The grid graph also consists of two finite and disjoint sets of prefixed meeting nodes located over the nodes of the grid. The objective is to design a distributed algorithm that gathers all the robots belonging to the first team at one of the meeting nodes belonging to the first type. Similarly, all the robots in the second team must gather at one of the meeting nodes belonging to the second type. The robots can distinguish between the two types of meeting nodes. A robot can identify the team to which it belongs. However, a robot cannot identify its team members. The initial configurations for which the restricted gathering problem is unsolvable have been characterized. For the remaining initial configurations, a distributed gathering algorithm has been proposed. The proposed algorithm assumes global-weak multiplicity detection capability for the robots and solves the problem within a finite time. The proposed algorithm runs in $\Theta(dn)$ moves, where d is the diameter of the minimum enclosing rectangle of all the

robots and meeting nodes and n is the number of robots in the system. We measure the time complexity of our algorithms in epochs. The proposed algorithm runs in $O(dn)$ epochs. The formal definition of the problem is as follows:

Problem 3. *Let $C(t) = (\mathcal{R}(t), \mathcal{M})$ be a given configuration. Given two finite disjoint sets of homogeneous robots, $\mathcal{R}_1$ and $\mathcal{R}_2$, and two finite disjoint homogeneous sets of meeting nodes $\mathcal{M}_1$ and $\mathcal{M}_2$ distributed over an infinite two-dimensional grid, the gathering over heterogeneous meeting nodes problem asks the robots in $\mathcal{R}_1$ to gather at one of the meeting nodes in $\mathcal{M}_1$ and the robots in $\mathcal{R}_2$ to gather at one of the meeting nodes in $\mathcal{M}_2$.*

1.13.4 Parking Problem over Infinite Grids

The gathering over meeting nodes problem can be extended to a generalized gathering model, where each fixed node is occupied by a number of robots equal to the capacity of the fixed node in the final configuration. This work considers *parking problem over infinite grids*. The robots are deployed at the nodes of an infinite grid, which has a subset of prefixed nodes marked as parking nodes. Each parking node p_i has a capacity of k_i which is given as input and represents the maximum number of robots a parking node can accommodate. As a solution to the parking problem, robots need to partition themselves into groups so that each parking node contains a number of robots equal to the capacity of the node. It is assumed that the number of robots in the initial configuration represents the sum of the capacities of the parking nodes. Under this setup, we have characterized all the initial configurations and the values of k_i for which the problem is unsolvable. For the remaining configurations, a deterministic distributed algorithm was proposed. The formal definition of the problem is as follows:

Problem 4. *Let $\mathcal{P}$ denote the set of parking nodes. $C(t) = (\mathcal{R}(t), \mathcal{P})$ denote the system configuration at any time t. The goal of the parking problem over infinite grids is to transform any initial configuration at some time $t > 0$ into a configuration such that each parking node p_i is saturated in the final configuration., i.e., each parking node contains exactly k_i robots.*

1.14 Organisation of the Thesis

This thesis considers the gathering at fixed nodes problem in an infinite grid domain. This thesis has four contributing chapters in addition to the Introduction, Related Works and Conclusion chapters. The literature review of previously studied works that are relevant to this thesis is presented in Chapter 2. The main contribution of the thesis is presented in Chapters 3-6. In Chapter 3, the gathering over meeting nodes problem has been considered. The main objective is to gather all the robots at one of the prefixed meeting nodes. In Chapter 4, the optimal gathering over Weber meeting nodes problem is described, where the objective is to gather the robots at one of the meeting nodes such that the sum of the distances traveled by the robots is minimized. Chapter 5 considers the gathering over heterogeneous meeting nodes problem, where the objective is to gather two different types of robots at two different meeting nodes. In Chapter 6, the parking problem in an infinite grid has been considered, where the objective is to place k_i number of robots at each parking node p_i, where k_i is the capacity of a parking node p_i. The capacity of each parking node is given as input to the robots. Finally, the research work done in this thesis is summarised in Chapter 7, along with the research directions that may be pursued in the future.

Chapter 2

Related Works

Numerous studies of various geometric problems for swarm robotics have been reported in the literature [65, 66]. The basic objective of these works has been finding minimal sets of capabilities that are needed to solve the problems. One of the most important and fundamental coordination problems that has been extensively investigated in the literature is the *Gathering* problem. It is also referred to as a point formation problem. It has been studied in both the continuous and discrete domains, where the underlying topology is an undirected graph. The problem is studied from different perspectives, including the capabilities of the robots, the deployment area, environmental constraints, faults, optimization constraints, etc. This thesis emphasizes the research in the discrete domain.

2.1 Gathering in Continuous domain

2.1.1 Rendezvous problem

The problem of gathering two robots is also known as *rendezvous*. Suzuki et al. [117] studied the rendezvous problem in the $\mathcal{SSYNC}$ model. Nevertheless, even if the robots have strong multiplicity detection capabilities, they will not be able to solve the problem if they do not have any agreement on the common coordinate axes. Izumi et al. [80] investigated the problem in a scenario where the robots have unreliable compasses. They considered two types of unreliable compasses: *static compass*, which never changes

its direction during the execution of the algorithm and *dynamic compass*, which may change its direction arbitrarily. They provide a complete characterization of rendezvous solvability with respect to compasses in the $\mathcal{SSYNC}$ setting. The main objective of their study is to find the minimum magnitude of consistency between the local coordinate systems of the robots that are necessary and sufficient to solve the problem. In a recent study, Bramas et al. [21] proved that if the robots disagree on the unit distance of their coordinate systems, they are able to solve rendezvous and agree on a final common location without making any additional assumptions.

Later, many researchers have considered the rendezvous problem in a scenario where the robots are endowed with a constant number of lights. Das et al. [52] proved that rendezvous can be solved by asynchronous robots using only three colors. They proved that without any other extra assumptions, the $\mathcal{ASYNC}$ model with visible lights is more powerful than the $\mathcal{SSYNC}$ model. Viglietta [119] proved that two colors are sufficient to solve the rendezvous problem in the $\mathcal{SSYNC}$ model, but only three colors are sufficient in the $\mathcal{ASYNC}$ model if the robots can detect null distances. Additionally, he showed that if robots use only the observed colors to determine their next move, rendezvous is not possible in the $\mathcal{ASYNC}$ model with two colors. Flocchini et al. [63] investigated the problem in two different weaker settings: $\mathcal{FSTATE}$ and $\mathcal{FCOM}$. They proved that $\mathcal{FSTATE}$ robots with rigid movements could gather in $\mathcal{SSYNC}$ using only six internal states. They also proved that the rendezvous of two $\mathcal{FCOM}$ robots could be solved with only three distinct colors in the $\mathcal{SSYNC}$ setting and 12 colors in the $\mathcal{ASYNC}$ setting. Okumura et al. [100] proposed an algorithm that solves the problem with two colors in the $\mathcal{ASYNC}$ model, where the robots have rigid movements. If the robots have non-rigid movements, then the value of δ was assumed to be known. However, their solutions are not self-stabilizing, i.e., robots must have a specific light color in the initial configuration. Heriban et al. [77] proposed a self-stabilizing algorithm for the rendezvous problem. They proved that only two colors are necessary and sufficient to solve the rendezvous problem in the $\mathcal{ASYNC}$ model.

2.1.2 Gathering for more than two robots

The gathering problem for $n > 2$ robots exhibits different characteristics from the rendezvous problem in terms of computation. Cohen and Peleg [38] studied the convergence

problem for fully-synchronous robots. They proved that the gathering problem could be solved by any number of synchronous, oblivious robots without making any additional assumptions. In their solution, each robot in the system moves straight toward the center of gravity of the robot positions. In [34], Cieliebak considered the gathering problem of n autonomous mobile robots in the plane, where the robots are assumed to be non-oblivious. Thus, they can store the results of all computations from the beginning and can freely access these results in order to carry out subsequent computations. Prencipe [106] proved that gathering of $n > 2$ semi-synchronous robots is impossible if the robots do not have any agreement on the coordinate axes or in the absence of any multiplicity detection capability. Izumi et al. [80] considered the gathering problem of $n \geq 2$ semi-synchronous robots using a broader model in which the local compasses of the robots have a dynamic tilt of ϕ, where $\phi < \frac{\pi}{4}$. Their proposed algorithm for the gathering problem required rigid movements for the robots and the assumption of common chirality. Dieudonne et al. [60] proposed a self-stabilizing gathering algorithm for gathering an odd number of semi-synchronous robots, where the robots are endowed with strong multiplicity detection capability. In general, it is assumed that during their look phase, each robot can accurately sense the positions of the other robots. However, there could be some inaccuracies in the measurements of both angles and distances. Cohen et al. [40] studied the gathering problem in which robots with consistent compasses and strong multiplication detection capabilities may have measurement inaccuracies. Cieliebak et al. [36] were the first to study the gathering of $n > 2$ robots in the $\mathcal{ASYNC}$ model. They assumed that the robots have weak multiplicity detection capability.

The robots may become faulty at any stage of execution. Agmon and Peleg [4] were the first to study the gathering problem under the fault model. They proposed an algorithm that can tolerate $\frac{n-1}{3}$ Byzantine faults in the $\mathcal{FSYNC}$ model. In the $\mathcal{SSYNC}$ model, they proposed a gathering algorithm that assumes the weak multiplicity detection capability of the robots. The solution can tolerate a single crash fault. Bramas et al. [22] proposed an algorithm that can tolerate $n-1$ crash faults, assuming strong multiplicity detection of the robots in the $\mathcal{SSYNC}$ model. Bhagat et al. [11] studied the gathering problem under the crash fault model and proposed an algorithm for gathering $n \geq 2$ asynchronous oblivious robots, where the robots have an agreement of only one axis. It was assumed that the robots do not agree on a common chirality and can gather without any multiplicity detection capability. Defago et al. [54] presented a study

of the feasibility of deterministic gathering under crash and byzantine faults. They presented several randomized solutions to the problem where deterministic solutions are not possible. This was the first study that considered probabilistic solutions to solve the gathering problem. They identified conditions under which a probabilistic solution exists and conditions for which no probabilistic solution exists.

The gathering problem has also been studied under different optimization criteria. A variant of the gathering problem was studied by Cicerone et al. [31], where the gathering is accomplished at one of the meeting points. The meeting points are a finite set of points that are visible to all the robots during their look phase. They identified the initial configurations for which the gathering problem is not solvable even in the $\mathcal{FSYNC}$ model. For the rest of the configurations, they proposed a deterministic algorithm that solves the problem in the $\mathcal{ASYNC}$ model. They also studied the same problem with respect to the two optimal criteria: one by minimizing the total number of moves traveled by all the robots and the other by minimizing the maximum distance traveled by a single robot. Bhagat et al. [15] were the first to study the constrained gathering problem for a set of autonomous robots in the $\mathcal{ASYNC}$ model. It was assumed that the gathering point is not restricted, and the optimization constraint is that the maximum distance traversed by the robots should be minimized. Based on the assumption that robots possess only two bits of memory, a distributed algorithm was proposed to solve the problem in finite time for a set of $n \geq 5$ asynchronous robots. Bhagat and Molla [14] studied the same problem with the assumption that the robots do not have any lights. The robots are assumed to have local-weak multiplicity detection capability and one-axis agreement.

2.1.3 Limited Visibility Model

All the studies discussed so far have assumed that the visibility of robots is unlimited. In the limited visibility model, the robots may not have a complete view of the environment. In this model, researchers have investigated the convergence and gathering problems. Ando et al. [5] studied the convergence problem for a set of semi-synchronous robots with limited visibility. Their proposed algorithm solves the convergence problem within a finite time, where the robots are assumed to be disoriented. Flocchini et al. [67, 68] were the first to study the gathering problem for asynchronous robots in the limited

visibility model. They proved that when robots agree on the directions and orientations of both local coordinate axes, i.e., when they have consistent compasses, the gathering problem is solvable for robots with limited visibility. Di Luna et al. [97] studied the gathering problem for disoriented robots with limited visibility, such that the robots do not initially form a connected visibility graph. The system is assumed to have a leader in the form of a special configuration of three robots (called *Turing Mobile*) that coordinate the activities leading up to the robot gathering. Another version of the gathering problem is the near-gathering problem, where the robots are required to be within a circle of a predetermined radius. Pagli et al. [102] studied this problem in the $\mathcal{ASYNC}$ model with limited visibility.

2.1.4 Fat Robot Model

The fat robot model, in which the robots are represented as disks instead of dimensionless points, provides a more realistic insight into robot-based systems. Czyzowicz et al. [43, 44] initiated the study of the gathering of fat robots in the Euclidean plane. Since robots have dimensions, they can not occupy the same position at the same time. Therefore, they defined the gathering pattern as follows: The robots must form a configuration in which the union of all the disks representing the fat robots must be connected. They solve the problem of gathering robots for the initial configurations consisting of at most four robots in the asynchronous setting. Cord-Landwehr et al. [41] proposed a gathering algorithm for synchronous fat robots, where the robots are assumed to be opaque. However, they assumed that the robots knew the gathering point in advance. The main objective of the robots is to gather as close as possible to the given point. Agathangelou et al. [3] studied the gathering of $n \geq 2$ opaque fat robots under the assumption that the robots agree on a common chirality. Honorat et al. [78] considered gathering of $n \leq 4$ robots equipped with a slim-omnidirectional camera. The cameras are fitted on the top of the robots. The robots are modeled as unit disks, each having an omnidirectional camera represented as a disk of smaller size. The region obstructed by a single robot in this model is a truncated infinite cone. In this setting, they proposed a gathering algorithm. Gan Chaudhuri et al. [27] extended the study of the gathering problem for disoriented fat transparent robots in the $\mathcal{ASYNC}$ model. They proposed a distributed algorithm for gathering $n \geq 2$ asynchronous fat robots.

2.2 Gathering in Discrete Domain

2.2.1 Gathering in rings

Gathering in the discrete domain has been extensively studied in rings [17, 45, 46, 48, 49, 50, 51, 75, 81, 82, 84, 85, 87, 88, 101]. Klasing et al. [88] proved that gathering in an anonymous ring is impossible without the multiplicity detection capability of the robots. For all configurations with an odd number of robots and all asymmetric configurations with an even number of robots, the authors proposed a distributed algorithm to solve the gathering problem. They assumed that the robots are endowed with global-weak multiplicity detection capability. It was proved that the gathering is not solvable for periodic configurations and for the configurations that admit edge-edge symmetries, where the axis of symmetry passes through two edges of the ring. Klasing et al. [87] investigated configurations in an anonymous ring that allows symmetries and has an even number of robots. The robots are endowed with global-weak multiplicity detection capability. They solved the problem for all configurations with more than eighteen robots. It has been proven that gathering is possible for an odd number of robots if and only if the configuration is not periodic. These results left open the cases where the number of robots is between four and eighteen.

Most of the cases where the number of robots is four were solved by Koren et al. [89]. The cases that remained open were the $SP4$ configurations. These are the sets of all symmetric configurations of type node–edge with four robots and the odd interval cut by the axis is bigger than the even one. An interval is defined as a maximal set of consecutive empty nodes. The node-edge symmetry type means that the axis of symmetry passes through one node and one edge of the ring. D'Angelo et al. [49] proposed an algorithm for gathering all configurations that belong to the set of $\mathcal{I} \setminus (\mathcal{U} \cup SP4)$, where $\mathcal{I}$ is the set of all initial configurations and $\mathcal{U}$ is the set of all ungatherable configurations listed in [88]. The algorithm assumed global-weak multiplicity detection of the robots. D'Angelo et al. [50] considered the gathering problem on anonymous rings with six robots. They proposed a distributed algorithm to solve the problem that assumes the global-weak multiplicity detection capability of the robots.

There are other results that concern the gathering of the robots in an anonymous ring, where they are endowed with local-weak multiplicity detection capability. Izumi et al.

[81] proposed a distributed gathering algorithm for the robots deployed at the nodes of an unoriented ring, where the robots are endowed with local-weak multiplicity detection capability. They proved that the proposed algorithm is asymptotically time-optimal, i.e., the time complexity of the algorithm is $O(n)$, where n is the number of nodes. Izumi et al. [82] provided a gathering algorithm for rings with local-weak multiplicity detection capability under the assumptions that the initial configurations are non-symmetric, non-periodic and the number of robots is less than half the number of nodes. Later, Kamei et al. [83] studied the gathering problem in anonymous rings where the robots are equipped with local-weak multiplicity detection capability. They assumed that the initial configuration might be symmetric but non-periodic. Furthermore, they assumed that the number of robots k is greater than eight and the number of nodes of the ring network is an odd integer greater than $k + 3$. In [84], Kamei et al. proposed an algorithm for the case where n is odd, k is even and $10 \leq k \leq n - 5$. D'Angelo et al [46] provide a full characterization of the initial configurations for which the gathering problem could be solved. The authors characterize the initial configurations from which the gathering problem is solvable for any $k < n - 4$ and $k = 4$. They design an algorithm that solves the problem when the robots are empowered by the local-weak multiplicity detection capability.

D'Emidio et al. [56] have shown that gathering of four robots on rings of seven nodes is unsolvable in $\mathcal{ASYNC}$. Di Stefano et al. [114] studied the optimal gathering on rings with the optimization constraint that the number of moves required to accomplish the gathering should be minimized. The proposed algorithm assumes the global-strong multiplicity detection capability of the robots. Kamei et al. [85] studied the gathering algorithms for myopic luminous robots in rings, i.e., the robots have limited visibility. The proposed algorithm does not assume any multiplicity detection capability. However, the robots are endowed with lights and each robot can see the color of the lights of its own and those of the other robots.

2.2.2 Gathering in grids

D'Angelo et al. [47] studied the gathering problem on trees and finite grids. They proved that even with global-strong multiplicity detection capability, a configuration is ungatherable if and only if it is periodic or symmetric, with the line of symmetry passing

through the edges of the grid. They solved the problem for the remaining configurations without assuming any multiplicity detection capability of the robots. Cord-Landwehr et al. [42] proposed an $\mathcal{O}(n)$ time algorithm for gathering robots in a finite grid under the fully-synchronous setting. The proposed algorithm assumed that each robot could observe other robots within a fixed distance of 20 units. It was also assumed that two robots are connected if and only if they are vertical or horizontal neighbors on the grid. Castenow et al. [24] provided a $\mathcal{O}(N^2)$ algorithm for gathering on a grid in the fully synchronous setting, where the robots are assumed to be oblivious. The viewing range of the robots was assumed to be seven. Poudel et al. [104] studied the gathering problem on a finite grid, where they assumed that each robot could observe any other robot within a viewing range of $\sqrt{10}$ units and two robots are connected if and only if they are $\sqrt{2}$ units apart. The robots agree on the direction and orientation of both axes. A distributed time-optimal algorithm was proposed in the asynchronous setting. The robots gather to a point in $\mathcal{O}(D_E)$ epochs, where D_E denotes the maximum distance between any pair of robots in the initial configuration.

A Weber point is a node of the graph that minimizes the sum of the lengths of the shortest paths from it to each robot. Di Stefano et al. [113] studied the optimal gathering problem in infinite grids, with the optimization criteria that the sum of the distances traveled by each robot to accomplish the gathering must be minimized. The authors proposed a distributed algorithm and assured gathering on a Weber point by letting each robot move along the shortest paths toward such a node. However, the robots have global-strong multiplicity detection capability. Recently, Shibata et al. [112] studied the gathering problem of seven mobile robots on triangular grids. The robots agree on the orientation and direction of both axes and they operate under a fully synchronous scheduler. With visibility range one, they showed that no collision-free algorithm exists for the gathering problem. With visibility range two, they proposed a collision-free algorithm that solves the problem from any connected initial configuration. Goswami et al. [73] studied the gathering problem on an infinite triangular grid for $n \geq 2$ robots, where the robots have limited visibility.

2.2.3 Gathering in other graph topology

Stefano et al. [114] studied the optimal gathering in general graphs, where the optimization constraint is to minimize the total number of asynchronous moves performed by the robots to accomplish the gathering task. They proposed sufficient conditions for gathering in general graphs. They also proposed two distributed algorithms for trees and rings that achieve exact gathering. Here, exact gathering refers to gather the robots at one of the Weber points in the graph. Bose et al. [19] considered the gathering problem in hypercubes. They proposed an optimal algorithm that minimizes the total number of moves by all the robots. Cicerone et al. [29] studied the gathering problem in dense and symmetric graphs like complete and complete bipartite graphs. In such topologies, they fully characterized the solvability of the gathering task in the synchronous setting. Guilbaut et al. [74] proposed a complete solution for the gathering problem for regular bipartite graphs. They proved that the class of gatherable initial configurations consists of only star graphs, where all the mobile agents are adjacent to an agent. Kamei et al. [86] studied the gathering problem on torus-shaped networks, where the robots operate under an asynchronous scheduler. However, the robots are assumed to be endowed with local-weak multiplicity detection capability.

2.3 Other Problems studied in Grid topology

2.3.1 Pattern Formation Problems

The *Arbitrary Pattern Formation Problem* ($\mathcal{APF}$) was first studied by Suzuki et al. [116, 117] in the Euclidean plane. They have characterized the entire class of patterns that are formed by autonomous and anonymous robots in $\mathcal{FSYNC}$ and $\mathcal{SSYNC}$. The robots are assumed to have an unbounded amount of memory. Flocchini et al. [69] investigated the solvability of the problem in $\mathcal{ASYNC}$, where the robots are assumed to be oblivious and represented as points in the plane. Dieudonne et al. [59] studied the relationship between the $\mathcal{APF}$ problem and the *Leader Election* problem and proved that the two problems are equivalent. Yamauchi et al. [123] studied the $\mathcal{APF}$ problem in limited visibility. The *Embedded Pattern formation* problem was first studied in [72] by Fujinaga et al. 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. Cicerone et al. [33] provided a full characterization of all configurations where the embedded formation problem is solvable without chirality.

In the grid topology, the $\mathcal{APF}$ problem has been studied by several researchers in [18, 30, 91, 92, 94]. Bose et al. [18] were the first to study the problem for asynchronous oblivious robots in infinite grids. However, the proposed algorithm assumed the initial configuration to be asymmetric. The problem of pattern formation on an infinite grid was studied by Lukovszki et al. [94] for robots with limited visibility. As part of the study, the problem was investigated for robots with a constant-size memory and a common coordinate system operating in the synchronous setting. In [91], Kundu et al. studied the $\mathcal{APF}$ in infinite grids for opaque robots. Adhikary et al. [2] studied a variant of the pattern formation problem, which is the *Circle Formation* problem in an infinite grid under the assumption that the robots are oblivious, asynchronous and opaque. Cicerone [30] investigated the $\mathcal{APF}$ problem for asynchronous robots in regular tessellations graphs, where the initial configuration is assumed to be asymmetric.

2.3.2 Mutual Visibility Problems

Di Luna et al. [96] initiated the study of the mutual visibility problem for a set of semi-synchronous oblivious robots. The problem for robots with persistent memory was studied by Di Luna et al. [95] with the assumption that the robots are unaware of the total number of robots in the team. They solved the problem (a) using two colors in $\mathcal{SSYNC}$ and (b) using three colors in $\mathcal{ASYNC}$ for rigid robots. As a solution to the problem of non-rigid robots, they used (a) with two colors in $\mathcal{SSYNC}$ if the robots knew δ, (b) with three colors in $\mathcal{SSYNC}$ without knowing δ, and (c) with three colors in $\mathcal{ASYNC}$ assuming the robots agree on the direction of one coordinate axis. Bhagat et al. [10] proposed an algorithm that solves the mutual visibility problem in two rounds for a set of synchronous robots. The algorithm uses only one bit of persistent memory and guarantees collision-free movements for the robots. The proposed algorithm is optimum in terms of round complexity, the amount of memory for the $\mathcal{FSTATE}$ computational model and the number of movements for the robots.

In infinite grids, the Mutual Visibility problem was studied in [1, 76, 105, 110, 111]. Adhikary et al. [1] initiated the study of this problem in infinite grids. In [1], the

authors proposed a distributed algorithm that solves the Mutual Visibility problem for luminous robots in $\mathcal{ASYNC}$ from any initial configuration using nine colors. Sharma et al. [111] considered the opaque point robot model and proved that any algorithm solving the problem would require $\Omega(N)$ epochs. They proposed a deterministic algorithm that requires 17 colors if leader election is not required. They also proposed a randomized algorithm that requires $O(max(D, N))$ epochs and 50 colors, where D is the diameter of the initial configuration. Hector et al. [76] studied the Mutual visibility problem in the asynchronous setting by allowing all the robots to form a convex hull. The authors proposed two randomized algorithms: an $O(max\{N^2, D\})$ time algorithm using 50 colors and an $O(max\{N^{1.5}, D\})$ time algorithm using 55 colors. Poudel et al. [105] considered the mutual visibility problem by fat robots on infinite grids. The movements were not restricted to one hop or along the grid lines. A robot can directly move to any visible grid point in one step. In this setting, they proposed an algorithm that requires $O(\sqrt{N})$ epochs in the $\mathcal{ASYNC}$ model, where N is the number of robots in the system.

2.3.3 Grid Exploration

Grid exploration has been studied in [7, 8, 16, 20, 57, 98, 107, 108]. Baldoni et al. [7, 8] studied the perpetual grid exploration in the $\mathcal{FSYNC}$ model. The robots have memory and a global sense of direction. Bonnet et al. [16] investigated the exclusive perpetual exploration of grid-shaped networks using anonymous, oblivious and fully asynchronous robots. The robots do not have any sense of direction. They proved that three robots are necessary and sufficient to solve the problem, provided that the size of the grid is $n \times m$ with $3 \leq n \leq m$ or $n = 2$ and $m \geq 4$. Rauch et al. [107] studied the *Optimal Exclusive Perpetual Grid Exploration by Luminous Myopic Robot* problem. The robots are deployed at the nodes of a finite grid and they are assumed to be disoriented. The robots have limited visibility. They proposed a distributed algorithm that is optimal in terms of the visibility range, the number of robots and the number of colors. Bramas et al. [20] studied the *Optimal Exclusive Perpetual Grid Exploration by Luminous Myopic Opaque Robots* problem, where the robots are assumed to be opaque. The robots have an agreement on a common chirality. They proposed a distributed algorithm that is optimal in terms of the visibility range, the number of robots and the number of colors.

Devismes et al. [57] studied the terminating grid exploration in infinite grids. They proposed necessary and sufficient conditions for the terminating exploration of an anonymous grid by a swarm of k asynchronous oblivious robots. They proved that three robots are necessary and sufficient in all but the cases where the grid size is 2×2 and 3×3, which requires four and five robots, respectively. Sardar et al. [108] studied the terminating grid exploration problem with the objective constraint that the total number of wasteful repeated visits of the nodes that occur during exploration should be minimized.

2.3.4 Dispersion

In the initial configuration, the robots are placed at the nodes of an n- node anonymous graph. The dispersion problem requires $k \leq n$ robots to reposition themselves to reach a configuration in which each robot is at a distinct node of the graph. Augustine et al. [6] initiated the study of this problem. The problem was examined in [6] in arbitrary graphs as well as in specific graph structures like paths, rings, and trees, where it was assumed that the number of robots is equal to n. Kshemkalyani et al. [90] studied this problem on grids. They proposed two deterministic algorithms for the dispersion problem on an anonymous grid graph. The first algorithm works under the local communication model, which restricts a robot's ability to communicate with those robots that are in its current node. The second algorithm operates under the global communication model, in which any robot in the graph, presumably located at different nodes, can communicate with any other robot in the graph.

Chapter 3

Gathering over Meeting Nodes in Infinite Grid [1]

3.1 Overview

The gathering problem has been explored in detail in the literature under different settings. However, most of the studies are in the continuous domain, where the robots can move freely in the plane. This chapter investigates the gathering problem in an infinite grid, given as an input. We have assumed that some of the nodes of the infinite grid are prefixed nodes, which are designated as *meeting nodes*. The objective of our study is to gather all the robots at one of the meeting nodes, where their movements are only allowed along the grid lines. 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. The gathering problem has been extensively studied in the discrete domain [47, 87, 88, 113], where the movements of the robots are restricted only along the edges of the input graph. However, the gathering point was not assumed to be

[1] This chapter of the thesis is based on the following publications:

1. Subhash Bhagat, Abhinav Chakraborty, Bibhuti Das, Krishnendu Mukhopadhyaya: Gathering over Meeting Nodes in Infinite Grid. Fundamenta Informaticae 187(1): 1-30 (2022)
2. Subhash Bhagat, Abhinav Chakraborty, Bibhuti Das, Krishnendu Mukhopadhyaya: Gathering over Meeting Nodes in Infinite Grid. Algorithms and Discrete Applied Mathematics - 6th International Conference, CALDAM 2020, Hyderabad, India, February 13-15, 2020, Proceedings, Pages: 318-330

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, charging stations, etc.

3.2 Contribution

This chapter considers gathering over meeting nodes problem in an infinite grid by asynchronous oblivious mobile robots. The problem asks for a deterministic distributed algorithm that gathers the robots at one of the meeting nodes within a finite amount of time. 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. Configurations admitting a unique line of symmetry such that the line of symmetry does not contain any robots or meeting nodes.
2. 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 problem, starting from 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 for the total number of movements performed by any algorithm for solving the 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. It has been proved that the proposed algorithm requires $O(D)$ epochs to terminate. The lower bound regarding the total number of epochs required to terminate is $O(D)$, which proves that the proposed algorithm is asymptotically optimal with respect to the number of moves and time in epochs.

Organization: The rest of the chapter is organized as follows. The next section focuses on the robot model and provides some preliminary definitions and notations. Section 3.4 provides the formal description of the gathering problem. A sufficient condition for the solvability of the gathering task has also been stated. In Section 3.5, we presented a deterministic distributed algorithm for solving the gathering over meeting nodes problem. Section 3.6 provides a lower bound for the complexity of the gathering problem in terms of the number of moves. A lower bound for the time complexity of the algorithm has also been presented in this section, where the time has been measured in terms of epochs. In this section, we have also provided a complexity analysis for our proposed algorithm. Section 3.7 concludes the chapter with some future directions to work with.

3.3 Model and Definitions

3.3.1 Model

The robots are assumed to be autonomous, anonymous, homogeneous, dimensionless and oblivious. They do not have any 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 ego-centric local coordinate system. Initially, the robots are assumed to be at 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. We have assumed that the robots are endowed with local-weak multiplicity detection capability.

3.3.2 Definitions

In this subsection, we have proposed some terminologies and definitions. These terminologies and definitions are relevant in understanding the problem definition.

- **System Configuration:**

- $P = (\mathbb{Z}, E')$: infinite path graph where the vertex set is 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, which is the usual l_1 norm.
- $\mathcal{R} = \{r_1, r_2, \ldots, r_n\}$: a set of robots deployed at 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.
- $\mathcal{R}(t) = \{r_1(t), r_2(t), ..., r_n(t)\}$: multiset of robot positions at time t. At $t = 0$, $r_i(t) \neq r_j(t)$, for all $r_i(t), r_j(t) \in \mathcal{R}(t)$. However, at $t > 0$, $r_i(t)$ may be equal to $r_j(t)$, for some $r_i(t), r_j(t) \in \mathcal{R}(t)$.
- $\mathcal{M} = \{m_1, m_2, \ldots, m_s\}$: set of meeting nodes located at the nodes of the grid graph.
- $C(t) = (\mathcal{R}(t), \mathcal{M})$: system configuration at time t (Figure 3.1).

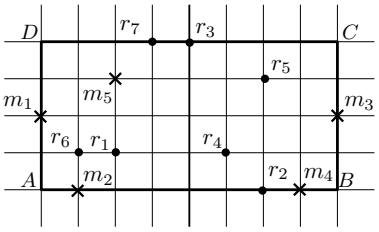

FIGURE 3.1: Example of a system configuration. The crosses represent meeting points, and the black circles represent the positions of the robots.

- **Symmetry:** An *automorphism* of a graph $G = (V, E)$ is a bijective map $\phi : V \to V$ such that u and v are adjacent if and only if $\phi(u)$ and $\phi(v)$ are adjacent. The automorphism of graphs can be extended similarly to define the automorphism of a configuration. Let $\mu_t : V \to \{0, 1, 2, 3, 4, 5\}$ at any time $t \geq 0$ be defined as a

function, where:

$$\mu_t(v) = \begin{cases} 0 & \text{if } v \text{ is an empty node} \\ 1 & \text{if } v \text{ is a meeting node without any robot positions on it} \\ 2 & \text{if } v \text{ is a single robot position on a meeting node} \\ 3 & \text{if } v \text{ is a robot multiplicity on a meeting node} \\ 4 & \text{if } v \text{ is a single robot position not on any meeting node} \\ 5 & \text{if } v \text{ is a robot multiplicity not on any meeting node} \end{cases}$$

Without any ambiguity, we denote μ_t by μ. An automorphism of a configuration $C(t)$ is an automorphism ϕ of the input grid graph such that $\mu(v) = \mu(\phi(v))$ for all $v \in V$. The set of all automorphisms of a configuration forms a group which is denoted by $Aut(C(t), \mu)$. If $|Aut(C(t), \mu)| = 1$, then the configuration is asymmetric. Otherwise, the configuration is said to be symmetric. Note that the function μ denotes the status or type of a node, i.e., $\mu(v)$ 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, a translational symmetry is not possible. A configuration admits a reflection symmetry if it is invariant after a reflection with respect to an axis passing through the geometric center of the grid. A unique line of symmetry characterizes a reflection symmetry. The line of symmetry can be horizontal, vertical, or diagonal and can pass through the nodes or edges of the graph. A configuration admits a rotational symmetry if it remains invariant with respect to rotations. The angle of rotation and the center of rotation characterize rotational symmetry. The angle of rotation can be 90° or 180°, whereas the center of rotation can be a node, a center of an edge, or the center of a unit square. Note that if a configuration admits symmetry, then no algorithm can distinguish between a robot r and its symmetric image. Consequently, a robot and its symmetric image(s) may decide to move simultaneously. A *pending move* may exist if an algorithm permits at least two robots to move at the same time. Due to the asynchronous nature of

the scheduler, it may so happen that one of the robots that is allowed to move completes its entire Look-Compute-Move cycle while one of the other robots does not perform the Move phase.

- **Partitive automorphism:** Given an automorphism $\phi \in Aut(C(t), \mu)$, the cyclic subgroup of order k generated by ϕ is given by $\{\phi^0, \phi^1 = \phi, \phi^2 = \phi \circ \phi, \ldots, \phi^{k-1}\}$, where ϕ^0 denotes the identity of the cyclic subgroup. Let H be any subgroup of $Aut(C(t), \mu)$. We define a relation ρ as follows: For some x, $y \in V$, we say that x and y are related by the relation ρ if there exists an automorphism $\gamma \in H$ such that $\gamma(x) = y$. Note that the relation ρ 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 [113] and is denoted by $H(x)$. These orbits form a partition of the set V since they represent disjoint equivalence classes. An automorphism $\phi \in Aut(C(t), \mu)$ 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 ϕ has order $k > 1$ and $|H(u)| = k$ 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 $\phi \in Aut\ (C(t),\ \mu)$ that is partitive on the set of nodes $V'' = V$. The cyclic subgroup H generated by ϕ 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 $90°$, then there exists an automorphism $\phi \in Aut\ (C(t),\ \mu)$ which is partitive on the set of nodes $V'' = V$ and the cyclic subgroup H generated by ϕ with $k = 4$ is given by $H = \{\phi^0, \phi^1, \phi^2, \phi^3\}$.

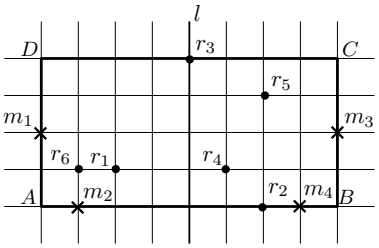

FIGURE 3.2: An example illustrating the identification of the key corner.

- **Configuration view**: Next, we define the view of a robot relative to the location of the robot and in terms of its local coordinates. For this, we first introduce the definition of the minimum enclosing rectangle. The main goal of introducing the

idea of view is to provide each robot with the same set of information about the entire configuration, which is frequently based on distances.

Let MER denote the *minimum enclosing rectangle* of $\mathcal{R} \cup \mathcal{M}$. MER is defined as the smallest grid-aligned rectangle that contains all the robots and meeting nodes. Assume that the dimension of MER is $p \times q$. We define the length of a side of MER in terms of the number of grid edges on them. A senary string of length pq is constructed as follows: Starting from a corner of MER, proceeding in the direction parallel to the width of MER and scanning the entire grid sequentially, we consider all the grid lines of MER column by column. While scanning the grid, we associate $\mu(v)$ 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 be denoted by s_{ij} and s_{ik} (In Figure 3.2, for the corner A, the strings associated are defined by s_{AB} and s_{AD}). Similarly, two senary strings of length pq are associated with each corner of MER. As a result, there are eight senary strings of length pq, which are associated with the corners of MER.

First, consider the case when MER is a non-square rectangle. We can distinguish the two strings associated to a particular corner by considering the string that progresses in the direction parallel to the side of the minimum length. Consider any particular corner i of 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 (In Figure 3.2, for the corner A, as $|AB| > |AD|$, the string direction is associated along AD. Here, s_{AD} is defined by $s_{AD} =$ 001001400004000000000000404000400401000000100). The direction parallel to the larger side (i.e., s_{ik}) is defined as the *non-string direction* associated to the corner i (In Figure 3.2, s_{AB} is the non-string direction). 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.

Lemma 3.3.1. *In an asymmetric configuration, all the strings associated to the corners of MER are different.*

Proof. For the sake of contradiction, assume that at least two strings associated to the corners of MER are equal. Without loss of generality, assume that $MER = ABCD$. We consider the following cases.

Case 1. Two strings associated to two adjacent corners are equal. Without loss of generality, assume that the strings s_{AD} and s_{BC} associated to the strings A and B are equal. This implies that for every pair of nodes a in the strings s_{AD} and b in the string s_{BC}, $\mu(a) = \mu(b)$. According to the definition of the symmetry of a configuration, it is an automorphism of a configuration such that $\mu(v) = \mu(\phi(v))$, for all $v \in V$. The fact that $\mu(a) = \mu(b)$ implies that there is a single line of symmetry that passes through the center of the grid-line joining the nodes A and B.

Case 2. Two strings associated to two non-adjacent corners are equal. It can be proved similarly that in this case, the configuration admits a rotational symmetry with the angle of rotation being 180°.

Case 3. The strings associated to one particular corner are equal. It should be noted that this can happen only when the MER is a square. It can be proved similarly that in this case, the configuration is symmetric with respect to the diagonal line of symmetry passing through that particular corner.

It can be proved similarly that if four strings associated to the corners are equal, then the configuration must admit a rotational symmetry with the angle of rotation being 90°. This proves that if at least two strings are equal, then the configuration must admit either a line of symmetry or rotational symmetry. By contradiction, this proves that if the configuration is asymmetric, then all strings associated to the corners are different. □

Corollary 3.3.2. *If a configuration is asymmetric, then there always exists a string which is lexicographically largest.*

Proof. According to Lemma 3.3.1, all the strings associated to the corners of MER are different. This implies that the strings can be ordered according to the lexicographic ordering. Hence, there always exists a lexicographically largest string among all possible strings. □

If both the strings associated with a particular corner are equal, then one of the directions is arbitrarily selected as the string direction. In Figure 3.2, l is the line of symmetry for the meeting nodes. s_{AD} is the lexicographic largest string. 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 3.2, A is the key corner. A corner that 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. [113]. If all the robots and meeting nodes lie on a single line, then MER is a $p \times 1$ rectangle with $A = D$ and $B = C$, and the length of AD and BC is 1. Note that, in this case, s_{AD} and s_{DA} refer to the same string. The meeting nodes are symmetric when the strings s_{AD} and s_{DA} are equal. 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 denotes the status of the node, i.e., $x = \mu(v)$.

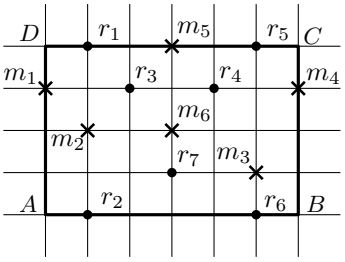

FIGURE 3.3: The meeting nodes are asymmetric and D is the unique leading corner.

Lemma 3.3.3. *If a configuration is asymmetric, there always exists a robot with the maximum configuration view.*

Proof. As the configuration is asymmetric, there always exists a unique key corner. Consider the unique key corner and the string direction s_i associated to the unique key corner i. Without loss of generality, let $\hat{O}$ be the ordering of the robots defined according to the order in which they appear in s_i. Consider the robot r, which has the highest order in $\hat{O}$. It should be noted that r has the maximum configuration view among all the robots. Hence, the proof follows. □

- **Symmetricity of the set $\mathcal{M}$:** We define MER_F as the smallest grid-aligned rectangle that contains all the meeting nodes. Define the function $f_t : V \to \{0, 1\}$

at any time t as follows:

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

We can define a string α_i similar to s_i. The only difference is that instead of $\mu(v)$, each node v is associated with $f_t(v)$. If the meeting nodes are asymmetric, then there exists a unique lexicographic largest string α_i (directly follows from the proof of Lemma 3.3.1). If the meeting nodes are not asymmetric, then the meeting nodes are said to be symmetric. The corner with which the lexicographic largest string α_i is associated is defined as the leading corner. In Figure 3.3, $\alpha_{DA} = 01000001000000010100000000001001000$, $\alpha_{AD} = 000100010000000$ 0010100000100000010, α_{BC}= 00010010000000000101000000010000010 and $\alpha_{CB} =$ 01000000100000010100000000010001000. α_{DA} is the largest lexicographic string among the $\alpha_i's$ and D is the leading corner.

3.4 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.

3.4.1 Problem Definition and Impossibility Results

This subsection formally defines the gathering over meeting nodes problem in an infinite grid.

3.4.1.1 Problem Definition:

Given a configuration $C(t) = (\mathcal{R}(t), \mathcal{M})$, the gathering over meeting nodes problem in an infinite grid asks the robots to gather at one of the meeting nodes within a 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.

- all robots are 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.

3.4.1.2 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) $\mathcal{I}_{11}-$ $\mathcal{M}$ is asymmetric (Figure 3.3).
   (b) $\mathcal{I}_{12}-$ $\mathcal{M}$ is symmetric with respect to a unique line of symmetry l and there exists at least one meeting node on l. $\mathcal{R} \cup \mathcal{M}$ may be asymmetric or symmetric with respect to l (Figure 3.4(a)).
   (c) $\mathcal{I}_{13}-$ $\mathcal{M}$ is symmetric with respect to rotational symmetry with c as the center of rotation and there exists a meeting node on c. $\mathcal{R} \cup \mathcal{M}$ may be either asymmetric or symmetric with respect to rotational symmetry (Figure 3.5(a)).
2. $\mathcal{I}_2-$ This includes the following class of configurations.

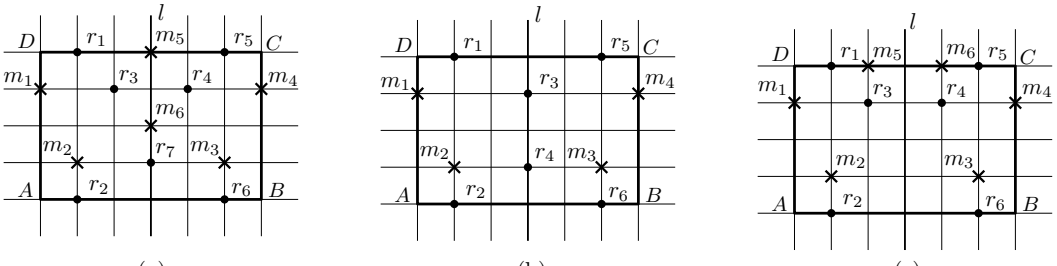

FIGURE 3.4: (a) $\mathcal{I}_{12}$- configuration: There exist meeting nodes on l. (b) $\mathcal{I}_{31}$- configuration: There do not exist any meeting nodes on l, but there exists a robot position on l. (c) $\mathcal{I}_{41}$- configuration without robots or meeting nodes on l.

   (a) $\mathcal{I}_{21}-$ $\mathcal{M}$ is symmetric with respect to a unique line of symmetry l. $\mathcal{R} \cup \mathcal{M}$ is asymmetric and there does not exist any meeting node on l (Figure 3.2).
   (b) $\mathcal{I}_{22}-$ $\mathcal{M}$ is symmetric with respect to rotational symmetry. $\mathcal{R} \cup \mathcal{M}$ is asymmetric and there does not exist a meeting node on c.

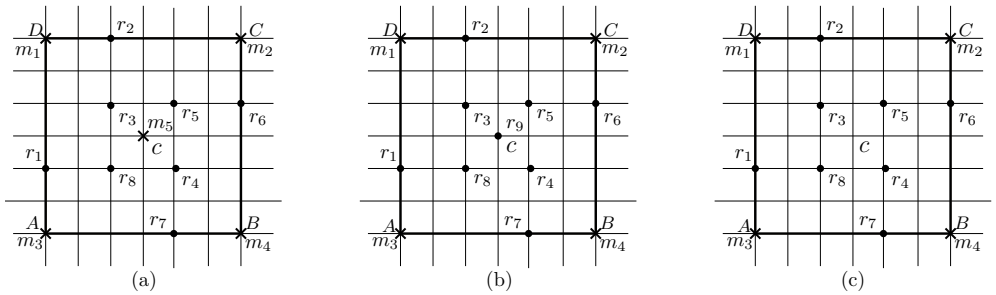

FIGURE 3.5: (a) $\mathcal{I}_{13}$- configuration with a meeting node on c. (b) $\mathcal{I}_{32}$- configuration without a meeting node on c, but there exists a robot position on c. (c) $\mathcal{I}_{42}$- configuration without a robot or meeting node on c.

3. $\mathcal{I}_3-$ This includes the following class of configurations.

   (a) $\mathcal{I}_{31}-$ $\mathcal{M}$ is symmetric with respect to a unique line of symmetry l. $\mathcal{R} \cup \mathcal{M}$ 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 3.4(b)).

   (b) $\mathcal{I}_{32}-$ $\mathcal{M}$ is symmetric with respect to rotational symmetry with c as the center of rotation. $\mathcal{R} \cup \mathcal{M}$ is symmetric with respect to rotational symmetry. There does not exist a meeting node on c, but there exists a robot position on c (Figure 3.5(b)).

4. $\mathcal{I}_4-$ This includes the following class of partitive configurations.

   (a) $\mathcal{I}_{41}-$ $\mathcal{M}$ is symmetric with respect to a unique line of symmetry l. $\mathcal{R} \cup \mathcal{M}$ is symmetric with respect to l and there does not exist any meeting node or robot position on l (Figure 3.4(c)).

   (b) $\mathcal{I}_{42}-$ $\mathcal{M}$ is symmetric with respect to rotational symmetry with c as the center of rotation. $\mathcal{R} \cup \mathcal{M}$ is symmetric with respect to rotational symmetry and there does not exist any meeting node or robot position on c (Figure 3.5(c)).

Let $\mathcal{I}$ 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 to which it belongs.

3.4.2 Impossibility Result

This subsection considers all configurations for which the gathering over meeting nodes problem is unsolvable in an infinite grid.

Theorem 3.4.1. *Given an initial configuration $C(0)$, let $V' \subset V$ be a subset of nodes such that $V' \cap \mathcal{R}(0) = \emptyset$. Assume that the robots are operated under a fully-synchronous scheduler. If there exists an automorphism ϕ 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 that can ensure gathering on a node in $V \setminus V'$.*

Proof. Let H be the cyclic subgroup generated by ϕ 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. 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 result 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.

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 $\mathcal{A}$. As a result, there exists at least one execution of $\mathcal{A}$ out of different execution paths of $\mathcal{A}$, 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 $\mathcal{A}$. Under this execution, the configuration remains partitive on the set $V \setminus V'$ at round $t + 1$. In Figure 3.6(a), v is the node that is occupied by the robot r_1.

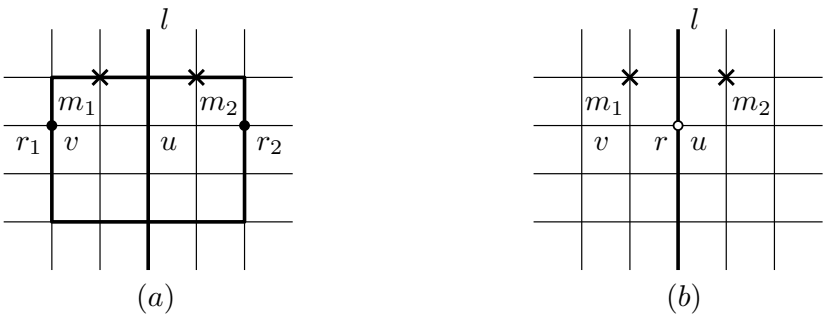

FIGURE 3.6: (a) Each robot decides to move towards u at round t. (b) Each robot on $H(v)$ moves towards u on l under the execution of $\mathcal{A}$ at round $t+1$. The configuration remains partitive after the movement. The circle on u represents a multiplicity node.

Here, $V' = l$, i.e., V' is the set of nodes belonging to the line of symmetry l. In Figure 3.6(b), under the execution of $\mathcal{A}$, each robot moves towards u on l.

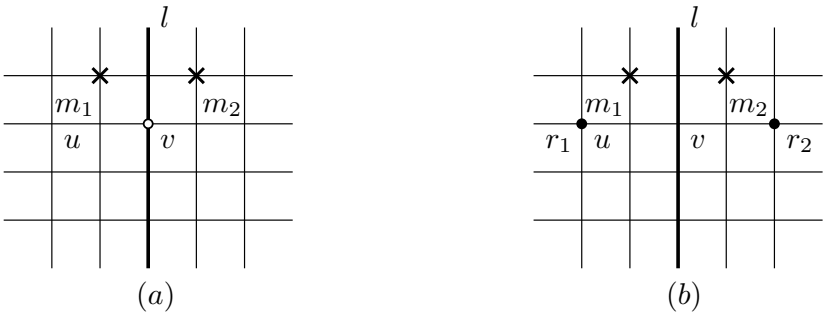

FIGURE 3.7: (a) Each robot on v decides to move towards distinct nodes of $H(u)$ at round t. (b) The robots on v move towards distinct nodes of $H(u)$ under the execution of $\mathcal{A}$ at round $t+1$. The configuration remains partitive after the movement.

2. $v \in V'$ and $u \in V \setminus V'$. Note that, in the initial configuration, $\mathcal{R}(0) \cap V' = \emptyset$. 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 $\mathcal{A}$, but the scheduler may choose a particular execution of $\mathcal{A}$, 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 $\mathcal{A}$ 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 3.7(a) and 3.7(b), under the

execution of $\mathcal{A}$, the robots on v move towards the nodes belonging to the orbit $H(u)$ and creates multiplicity at those nodes.

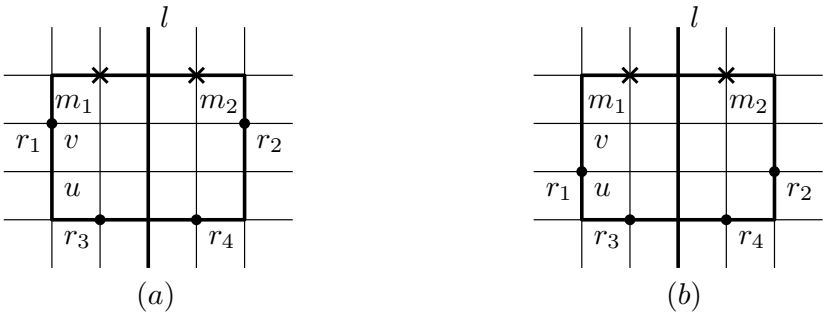

FIGURE 3.8: (a) Each robot on $H(v)$ decides to move towards $H(u)$ on $V \setminus l$ at round t. (b) Each robot on $H(v)$ decides to move towards $H(u)$ on $V \setminus l$ at round $t+1$ under the execution of $\mathcal{A}$. The configuration remains partitive after the movement.

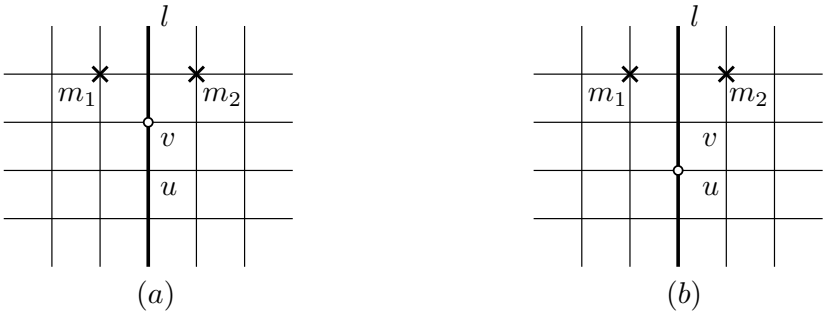

FIGURE 3.9: (a) Each robot on v decides to move towards u at round t. (b) Each robot on v moves towards u at round $t+1$. The configuration remains partitive after the movement.

3. $v \in V \setminus V'$ and $u \in V \setminus V'$. There exists at least one execution of $\mathcal{A}$ 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 3.8(a) and 3.8(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, $\mathcal{R}(0) \cap V' = \emptyset$, 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 $\mathcal{A}$, 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 $\mathcal{A}$. Thus, the configuration remains partitive on the set $V \setminus V'$ at round $t+1$ (In Figures 3.9(a) and 3.9(b), the robots on v move towards u).

Starting from $C(t)$ and with the execution of $\mathcal{A}$, $C(t+1)$ 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 $\mathcal{A}$, 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 $\mathcal{A}$. Thus, we have proved that 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 3.4.1 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. As a result, we state the following corollary without proving it.

Corollary 3.4.2. *Given an initial configuration $C(0)$, let $V' \subset V$ be a subset of nodes such that $V' \cap \mathcal{R}(0) = \emptyset$. Assume that the robots are operated under an asynchronous scheduler. If there exists an automorphism ϕ 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 that can ensure gathering on a node in $V \setminus V'$.*

Let V' be the set of nodes on l, if $C(0) \in \mathcal{I}_{41}$. Otherwise, let V' be the node $\{c\}$, if $C(0) \in \mathcal{I}_{42}$. Now, we have the following corollary:

Corollary 3.4.3. *If $C(0) \in \mathcal{I}_4$, then the gathering over meeting nodes problem is unsolvable.*

Proof. First, consider the case when $C(0) \in \mathcal{I}_{41}$. This implies that $C(0)$ is partitive on the node set $V \setminus l$. According to Theorem 3.4.1, 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 $C(0) \in \mathcal{I}_{42}$, where $C(0)$ is partitive on the node set $V \setminus \{c\}$. □

Corollary 3.4.4. *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 $\mathcal{A}$, which ensures the 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. □

For the rest of the chapter, 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 $\mathcal{U}$ denote the set of all initial configurations which are ungatherable according to Corollary 3.4.3. In other words, $\mathcal{U}$ represents the following collection of configurations.

- admitting a unique line of symmetry l and no meeting node or robot position on l.
- admitting rotational symmetry with no meeting node or robot on c.

3.5 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 $\mathcal{I} \setminus \mathcal{U}$.

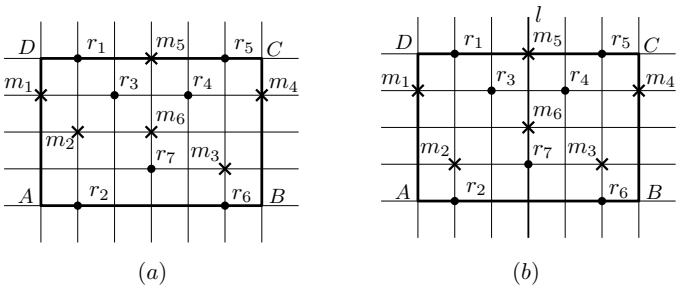

FIGURE 3.10: (a) The meeting nodes are asymmetric. (b) The meeting nodes are symmetric with respect to l, and there exists meeting nodes m_5 and m_6 on l.

The pseudo-code of the algorithm *Gathering*() is given in Algorithm 3.1. We will see later that if the meeting nodes are asymmetric, then they can be ordered. Even if the meeting nodes are symmetric with respect to l, and there exist meeting nodes on l, then the meeting nodes on l are orderable.

First, consider the case when the meeting nodes are asymmetric. Note that according to the definition of symmetry of the set $\mathcal{M}$, there exists a unique lexicographic largest string α_i associated with the unique leading corner. In Figure 3.10 (a), D is the unique leading corner and α_{DA} is the unique largest lexicographic string. Consider the largest lexicographic string α_i associated with the unique leading corner and the meeting nodes that appear in the string representation of α_i. Without loss of generality, let $\mathcal{O}_1$ be the ordering of the meeting nodes defined according to the order in which they appear in α_i. Let $\mathcal{O}_1 = (m_1, m_2, \ldots m_s)$ denote this particular ordering of the meeting nodes. In Figure 3.10(a), $(m_1, m_2, m_5, m_6, m_3, m_4)$ is the ordering $\mathcal{O}_1$. Note that this particular ordering remains invariant while the robots move towards any meeting node. While the ordering of the meeting nodes remains invariant, each robot can select the meeting nodes according to the ordering and finalize the gathering at the meeting node.

Next, assume the case when the meeting nodes are symmetric with respect to a unique line of symmetry l and there exists at least one meeting node at l. If l is a horizontal or vertical line of symmetry, then there exist two leading corners. Consider the meeting nodes on l ordered according to their appearances in the string directions associated to the leading corners. Without loss of generality, let $\mathcal{O}_2 = (m'_1, m'_2, \ldots, m'_z)$ be this particular ordering of the meeting nodes at l. In Figure 3.10(b), C and D are the leading corners. (m_5, m_6) is the ordering $\mathcal{O}_2$. Here z denotes the total number of meeting nodes lying at l. It may be noted similarly that this particular ordering remains invariant while the robots move toward any meeting node. If l is a diagonal line of symmetry and there

exists a unique leading corner, then the ordering of the meeting nodes on l is defined with respect to the distance from the leading corner.

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.*

3.5.1 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 $\mathcal{I} \backslash \mathcal{U}$ and consists of 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 $|\mathcal{M}| = 1$, then all the robots move towards the unique meeting node and finalize the gathering. Therefore, for the rest of the chapter, we assume that $|\mathcal{M}| \geq 2$. The unique meeting node, which is considered for gathering is defined as the target meeting node.

Algorithm 3.1: *Gathering*()

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

```
if C(t) ∈ I_11 then
    Each robot moves towards the meeting node m_s having the highest order with respect to
      O_1 ;
else if C(t) ∈ I_12 then
    Each robot moves towards the meeting node m_z on l having the highest order with respect
      to O_2 ;
else if C(t) ∈ I_13 then
    Each robot moves towards the meeting node on c ;
else if C(t) ∈ I_2 then
    GatheringAsym();
else if C(t) ∈ I_3 then
    SymmetryBreaking() ;
    GatheringAsym();
```

3.5.1.1 $\mathcal{I}_1$

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 $\mathcal{O}_1$ of the meeting nodes, defined with respect to the unique leading corner. The meeting node m_s having the highest order with respect to $\mathcal{O}_1$ is selected as the target meeting node. All the robots move towards m_s and finalize the gathering at it. In Figure 3.10(a), D is the leading corner and $(m_1, m_2, m_5, m_6, m_3, m_4)$ is the ordering $\mathcal{O}_1$. m_4 is the meeting node that has the highest order in $\mathcal{O}_1$. 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 $\mathcal{O}_2 = (m_1, m_2, \ldots m_z)$ of the meeting nodes on l. The meeting node m_z on l having the highest order with respect to $\mathcal{O}_2$ is selected as the target meeting node. All robots move towards the meeting node m_z, and the gathering is finalized at m_z. In Figure 3.10(b), C and D are the leading corners and (m_5, m_6) is the ordering $\mathcal{O}_2$. m_6 is the meeting node on l, which has the highest order among all the meeting nodes on l according to $\mathcal{O}_2$. m_6 is selected as the target meeting node.

3. $C(0) \in \mathcal{I}_{13}$. There exists a meeting node (say m) on c. All robots move towards m and finalize the gathering at m. In Figure 3.11(a), m_5 is selected as the target meeting node. In Figure 3.11(b), all robots move towards m_5.

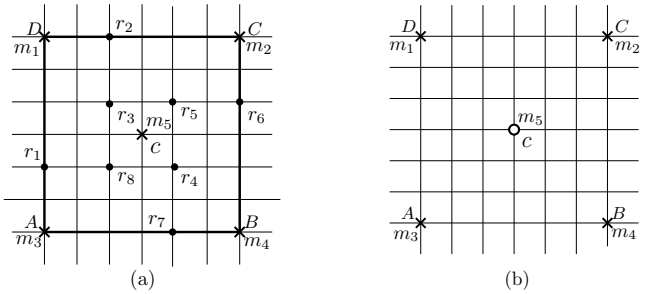

FIGURE 3.11: (a) $\mathcal{I}_{13}$- configuration with meeting node on c. m_5 is selected as the target meeting node. (b) The gathering is finalized by moving the robots towards m_5.

Lemma 3.5.1. *If $C(0) \in \mathcal{I}_1$, 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 $\mathcal{I}_{11}$, $\mathcal{I}_{12}$ and $\mathcal{I}_{13}$, the following cases are considered.

Case 1. $C(0) \in \mathcal{I}_{11}$. All robots agree on the ordering $\mathcal{O}_1$ of the meeting nodes. As the ordering $\mathcal{O}_1$ of the meeting nodes depends only on the position of the unique leading corner, the ordering remains invariant during the movement of robots. Therefore, the meeting node having the highest order in $\mathcal{O}_1$ also remains invariant. As a result, the target meeting node remains invariant.

Case 2. $C(0) \in \mathcal{I}_{12}$. The target meeting node m_z is selected as the meeting node on l having the highest order with respect to $\mathcal{O}_2$. Since the ordering depends only 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 $\mathcal{M}$.

Case 3. $C(0) \in \mathcal{I}_{13}$. 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 robots move towards it.

Hence, the target meeting nodes remain invariant during the execution of the algorithm at any time $t > 0$. □

3.5.1.2 $\mathcal{I}_2$

Assume that the initial configuration $C(0) \in \mathcal{I}_2$. In this case, the meeting nodes are symmetric, but the configuration is asymmetric. Even there does not exist any meeting node on $l \cup \{c\}$. Here, each robot executes $GatheringAsym()$. The overview of the procedure $GatheringAsym()$ is 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 toward the target meeting node. While the guard moves, it moves in a shortest path toward the target meeting node.

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 MER_F 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. We next consider the following cases.

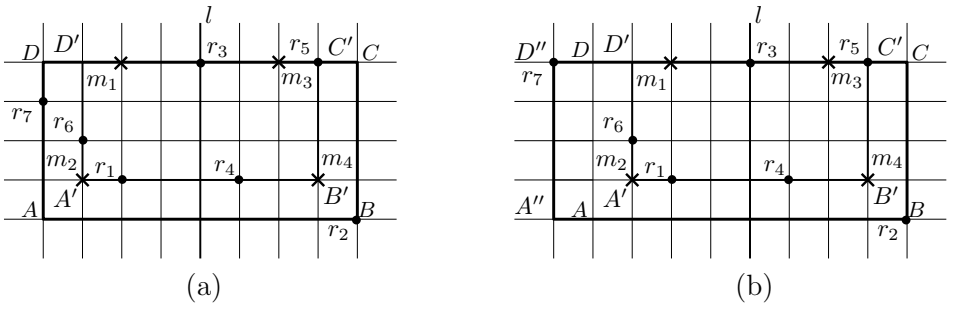

FIGURE 3.12: (a) $A'B'C'D'$ is the minimum enclosing rectangle MER_F of all meeting nodes. $ABCD$ is the MER. (b) r_7 is selected as a guard and moves towards an adjacent node away from l. Finally, it moves toward its closest corner. The transformed MER is $A''BCD''$.

1. There exists at least one robot position outside the rectangle MER_F. This implies that there exists at least one robot position at a distance $d_2 > d_1$ from l. If there are

multiple such robots at a distance d_2, consider the unique robot with the maximum configuration view. Let r be the unique 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 that is at the maximum distance from l. In Figures, 3.12(a) and 3.12(b), r_2 and r_7 are the robots outside the MER_F 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 MER_F. 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 unique robot r with the maximum configuration view. r is selected as the guard and it moves toward an adjacent node away from l. r continues its 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 like the previous case.

Once the guard becomes the unique farthest robot from l, it moves toward the closest corner of MER in the direction parallel to the string direction. If the guard is closest to two corners of MER, then it moves towards an arbitrary corner. In Figure 3.12(b), r_7 moves towards D''. The procedure progresses similarly when the meeting node admits rotational symmetry, and there does not exist any meeting node on c. In that case, d_1 and d_2 are defined as the distances from c. The pseudo-code corresponding to this phase is given in Algorithm 3.2.

Algorithm 3.2: *GuardSelection*()

Input: $C(t) = (\mathcal{R}(t), \mathcal{M})$

```
if there exists at least one robot position outside MER_F then
    if there exists exactly one robot position r outside MER_F then
        r is selected as the guard ;
    else
        The unique robot r farthest from l ∪ {c} and with the maximum configuration view is
          selected as the guard ;
    r moves toward an adjacent node away from l ∪ {c} and finally towards its closest corner;
else if each robot is inside or on MER_F then
    if there exists a unique robot r farthest from l ∪ {c} then
        r is selected as the guard;
    else
        The unique robot r farthest from l ∪ {c} and with the maximum configuration view is
          selected as the guard ;
    r moves toward an adjacent node away from l ∪ {c} and continues its movement unless it is
      outside MER_F;
```

Lemma 3.5.2. *During the execution of the procedure GuardSelection(), the guard remains uniquely identifiable by the robots.*

Proof. First, assume that the meeting nodes are symmetric with respect to l and there do not exist any meeting nodes on l. The proof follows similarly when the configuration admits rotational symmetry, and there does not exist any meeting node on c. The following cases are to be considered.

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 *GuardSelection*(), r is selected as a guard. Otherwise, if there are multiple such robots, the unique robot r with the maximum configuration view is selected as the guard. r moves towards an adjacent node v away from l. The moment it reaches v, it becomes the unique farthest robot from l. While the guard moves towards the corner, it remains the unique farthest robot from l. As the guard is selected as the unique farthest robot from l, it remains uniquely identifiable by the other robots.

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 guard. Note that there may be multiple such robots. The procedure *GuardSelection*() ensures that the guard is selected as the unique robot r, which is farthest from l and with the maximum configuration view in case of a tie. 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 outside the rectangle MER_F. The rest of the proof follows from Case 1. □

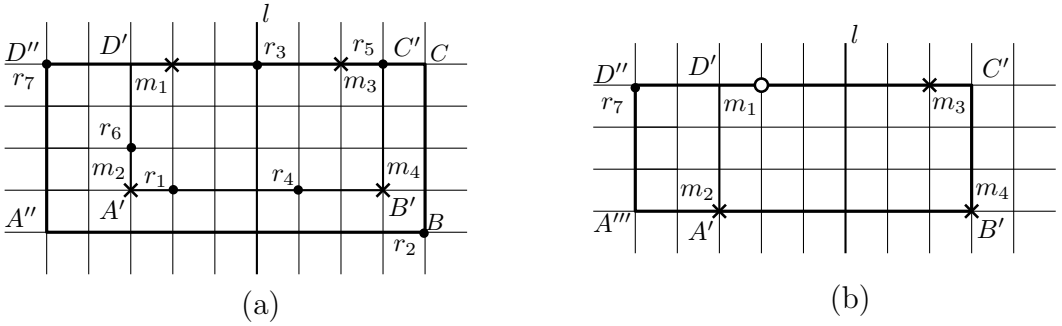

FIGURE 3.13: (a) The configuration after the execution of *GuardSelection*. m_1 is selected as the target meeting node. (b) Each non-guard moves towards m_1 and creates a multiplicity on m_1. The MER becomes $A'''B'C'D''$. The circle on m_1 represents a robot multiplicity on m_1.

According to Lemma 3.5.2, 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 as being 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 $f_t(v)$. 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 $\mathcal{O}''$.

Creating Multiplicity on Target Meeting Node: In this phase, each non-guard robot moves towards the target meeting node m. Since $\mathcal{R} \cup \mathcal{M}$ 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 unique meeting node m that has the minimum order in $\mathcal{O}''$ 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 3.13(a) and 3.13 (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 3.3.

Algorithm 3.3: $MakeMultiplicity()$

Input: $C(t) = (\mathcal{R}(t), \mathcal{M})$

```
if there exists a unique meeting node m closest to the guard then
    Each robot selects m as the target meeting node;
else
    Let m be the closest meeting node from the guard that has the minimum order in O'' ;
    Each robot selects m as the target meeting node;
Let r be a closest non-guard robot that is not on m;
r moves towards m in a shortest path ;
```

Lemma 3.5.3. *During the execution of $MakeMultiplicity()$, the target meeting node remains invariant.*

Proof. Since the configuration is asymmetric, there exists a unique ordering $\mathcal{O}''$ of the meeting nodes with respect to the unique guard. 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. When there are multiple closest meeting nodes from the guard, the target meeting node m is selected as the unique meeting node closest to the guard and that has the minimum order in $\mathcal{O}''$. Since the guard does not move, the ordering $\mathcal{O}''$ 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 as in the above case. □

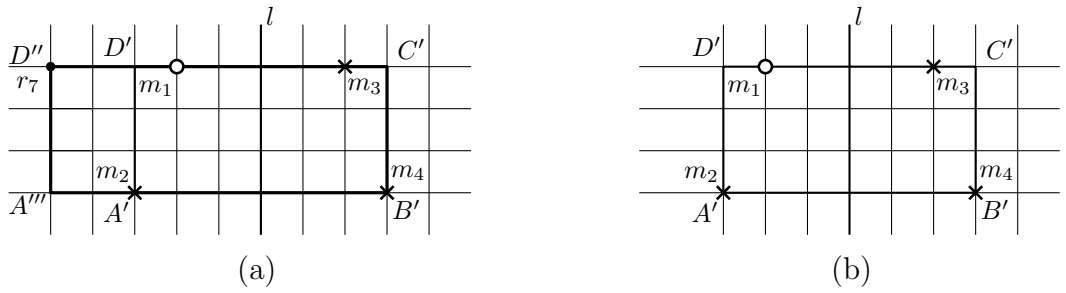

FIGURE 3.14: (a) The configuration after the execution of the procedure $MakeMultiplicity()$. (b) r_7 moves towards m_1 and finalizes the gathering. The MER becomes $A'B'C'D'$.

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 toward the other robot position in a shortest path. All the other robots on the target meeting 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 3.14(a) and 3.14(b), r_7 observes that it is not on a multiplicity node. It would move towards m_1 and finalize

the gathering at it. The pseudo-code corresponding to this phase is given in Algorithm 3.4.

Algorithm 3.4: *GuardMovement*()

Input: $C(t) = (\mathcal{R}(t), \mathcal{M})$

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

3.5.1.3 $\mathcal{I}_3$

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

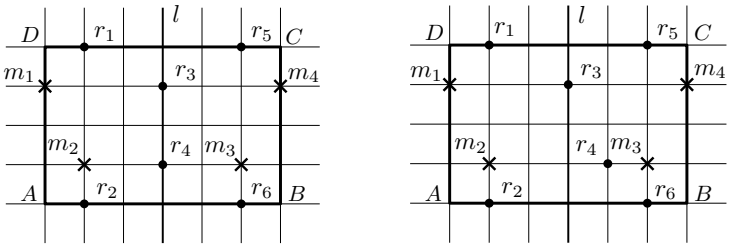

FIGURE 3.15: (a) $\mathcal{I}_{31}$-configuration. (b) Transformation of $\mathcal{I}_{31}$-configuration into an asymmetric configuration by the movement of the robot r_4.

Symmetry Breaking: In this phase, all the symmetric configurations which 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 unique robot on l having the maximum configuration view. r moves towards an adjacent node that does not belong to l. In Figures 3.15(a) and 3.15(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.

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 3.16(a) and 3.16(b), the robot

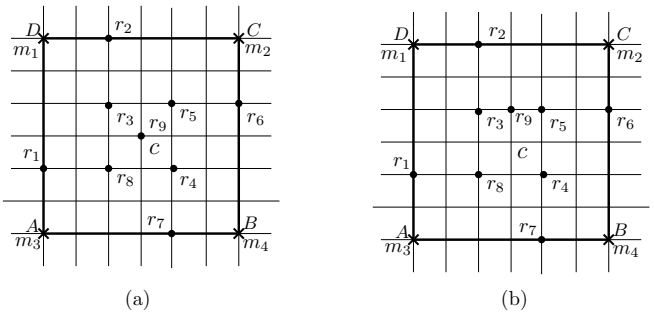

FIGURE 3.16: (a) $\mathcal{I}_{32}$-configuration. (b) Transformation of $\mathcal{I}_{32}$-configuration into an asymmetric configuration by the movement of r_9 on c.

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 3.5. Once the configuration is transformed into an asymmetric configuration, $GatheringAsym()$ is executed. 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 guard is selected as the farthest robot with the maximum configuration view. Note that such a robot position always exists.

Lemma 3.5.4. *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 3.4.3, the ungatherable configurations can be characterized by the following conditions:

Algorithm 3.5: *SymmetryBreaking*()

Input: $\mathcal{C}(t) \in \mathcal{I}_3$

```
if C(t) ∈ I_31 then
    Let r be the robot on l with the maximum configuration view ;
    Move r towards any adjacent node that does not belong to l;
else if C(t) ∈ I_32 then
    Let r be the robot on c ;
    Move r towards any adjacent node;
```

1. configurations admitting a unique line of symmetry l and without any robot or meeting nodes on l.

2. configurations admitting rotational symmetry with the 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 3.5.2, 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-guard moves towards MER_F, 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 the procedure *SymmetryBreaking*(). The configuration remains asymmetric at $t' > t$, similar as in Case 2.

This proves that if $C(0) \notin \mathcal{U}$, then during the execution of $Gathering()$, $C(t) \notin \mathcal{U}$, for any $t > 0$. □

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

Theorem 3.5.5. *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 3.5.4, 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 3.5.1, 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, $d(t') = \sum_{r_i \in \mathcal{R}(t')} d(r_i(t'), m)$ 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 $MakeMultiplicity()$. 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 $d(t') = \sum_{r_i \in \mathcal{R}(t')} d(r_i(t'), m)$. According to Lemma 3.5.3, 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 $GuardMovement()$ is started. Otherwise, let r be any non-guard robot that 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 $GuardMovement()$ will be started.

Next, we consider the execution of procedure $GuardMovement()$. 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, $SymmetryBreaking()$ is executed. The transformed configuration becomes either asymmetric or may admit a unique line of symmetry. By the repeated execution of $SymmetryBreaking()$, the configuration becomes asymmetric. The rest of the proof follows from Case 2.

Hence, execution of the algorithm $Gathering()$ eventually solves the gathering over meeting nodes problem within a finite time. □

3.6 Lower bounds

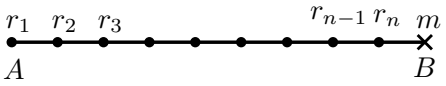

FIGURE 3.17: A configuration showing the lower bound in terms of the number of movements required to finalize the gathering.

In this section, we study the efficiency of our algorithm in terms of the total number of moves. We have also analyzed the time complexity of the proposed algorithm in terms of the number of epochs. Let n denote the total number of robots deployed at 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 that 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 3.17, 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.

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

Theorem 3.6.1. *Any gathering algorithm for solving the gathering over meeting nodes problem requires* $\Omega(Dn)$ *moves.*

Next, assume that in each epoch, the robot r_1 is the last robot that gets activated. As a result, in each epoch, r_1 moves only a unit hop towards m, which results in $\Omega(D)$ epochs for the termination to occur. Hence, we have the following theorem.

Theorem 3.6.2. *Any gathering algorithm for solving the gathering over meeting nodes problem requires $\Omega(D)$ epochs.*

3.6.1 Analysis of the Algorithm Gathering()

Number of moves

- 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 toward the target meeting node. The number of moves in this phase is $O(D)$.

Hence, we have the following result.

Theorem 3.6.3. *Algorithm Gathering() solves the gathering over meeting nodes problem in $\Theta(Dn)$ moves if the initial configuration belongs to the set $\mathcal{I} \setminus U$.*

Time Complexity

- The Symmetry Breaking phase requires only $O(1)$ epochs to terminate.
- 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 epochs 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 epochs in this phase is $O(D)$.

- Finally, in the Finalization of Gathering phase, only the guard moves toward the target meeting node. The number of moves in this phase is $O(D)$.

Hence, we have the following result.

Theorem 3.6.4. *Algorithm Gathering() solves the gathering over meeting nodes problem in $\Theta(D)$ epochs if the initial configuration belongs to the set $\mathcal{I} \setminus U$.*

3.7 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 shown 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 and epochs executed by the robots. We have proved that the algorithm solves the gathering over meeting nodes problem in $\Theta(Dn)$ moves and $\Theta(D)$ epochs, where D is the length of the larger side of the initial MER.

Chapter 4

Optimal Gathering over Weber Meeting Nodes in Infinite Grid [1]

4.1 Overview of the Problem

In Chapter 3, we studied the gathering over meeting nodes problem on a grid-based terrain. The problem requires a deterministic distributed algorithm that ensures the gathering of the robots at one of the meeting nodes. Although the proposed algorithm was asymptotically optimal with respect to the number of moves performed by the robots in order to ensure the gathering of the robots, the proposed algorithm was not optimal with respect to the exact number of moves performed by the robots. The main aim of this chapter is to study the problem under the optimization constraint that the sum of the distances traveled by the robots is minimized while accomplishing the gathering task. In order to complete the gathering task, the robots must select a unique meeting node and move towards it in such a way that the sum of the lengths of the shortest paths from each robot to the selected meeting node is minimized. A meeting node that minimizes the sum of the distances from all the robots is defined as a Weber meeting node. This chapter presents the study of optimal gathering over Weber meeting nodes problem, where the gathering is restricted to Weber meeting nodes.

[1]This chapter of the thesis is based on the following publication: Subhash Bhagat, Abhinav Chakraborty, Bibhuti Das, Krishnendu Mukhopadhyaya: Optimal Gathering over Meeting Nodes in Infinite Grid. International Journal of Foundations of Computer Science Vol. 34, No. 01, pp. 25-49 (2023)

4.2 Contribution

This chapter proposes a deterministic distributed algorithm for optimal gathering over Weber meeting nodes problem, where the initial configurations comprise at least seven robots. The robots are deployed at the nodes of an infinite grid. The optimization criterion considered in this chapter is the minimization of the total number of moves made by the robots to finalize the gathering. Di Stefano et al. [114] proved that to ensure gathering by minimizing the total number of moves, the robots must gather at one of the Weber points. In our restricted gathering model, the robots must gather at one of the Weber meeting nodes to ensure gathering with a minimum number of moves. In this chapter, it has been shown that there exist some configurations where gathering over Weber meeting nodes cannot be ensured, even if the robots are endowed with multiplicity detection capability. This includes the following collection of configurations:

1. Configurations admitting a single line of symmetry without any robot or Weber meeting node on the reflection axis.
2. Configurations admitting rotational symmetry without a robot or a meeting node on the center of rotation.

In this chapter, we assumed that the robots are endowed with global-strong multiplicity detection capability. We have shown that, without such an assumption, there are configurations where gathering cannot be accomplished as soon as a multiplicity is created. However, there are initial configurations where gathering can be ensured over a meeting node but not on the set of Weber meeting nodes. This includes the configuration admitting a single line of symmetry without any robots or Weber meeting nodes on the reflection axis, but at least one meeting node exists on the reflection axis. The feasibility of solving the gathering over meeting nodes has been studied in that case.

Organization: The following section describes the robot model and the notations used in the chapter. Section 4.4 provides the formal definition of the problem and the impossibility results for the solvability of the gathering task. Section 4.5 proposes a deterministic distributed algorithm to solve the optimal gathering over Weber meeting nodes problem. Section 4.6 describes the correctness of the proposed algorithm. Section 4.7 discusses the optimal gathering for the configurations where gathering over a meeting node can

be ensured but cannot be ensured over a Weber meeting node. Finally, in Section 4.8, we conclude the chapter with some discussion about future research.

4.3 Model and Definitions

The robots are assumed to be autonomous, anonymous, homogeneous, dimensionless and oblivious. They do not have explicit means of communication. The robots have global visibility, i.e., they can observe the entire grid. They 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 the origin as its current position. Initially, the robots are assumed to be at the distinct nodes of the input grid. Each active robot executes Look-Compute-Move(LCM) cycle under an asynchronous scheduler. 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. The robots are endowed with global-strong multiplicity detection capability.

- **Symmetry**: In Chapter 3, the robots are endowed with local-weak multiplicity detection capability. This chapter assumes that the robots are endowed with global-strong multiplicity detection capability. Therefore, in this chapter, we define symmetry slightly differently than we did in chapter 3.

  Let $\lambda_t : V \to N$ be a function denoting the number of robots on each node $v \in V$ at any time $t \geq 0$. A symmetry of a configuration is defined with respect to an automorphism of a configuration, where an automorphism of a configuration denoted by $\mathrm{Aut}((C(t),\ f_t, \lambda_t))$ is an automorphism ϕ of the input grid graph such that $f_t(v) = f_t(\phi(v))$ and $\lambda_t(v) = \lambda_t(\phi(v))$ for all $v \in V$, where f_t is the function defined in Chapter 3.

- **Weber meeting node**: In the initial configuration, the robots are deployed at the distinct nodes of the grid graph and hence $\lambda_t(v) \leq 1$, $\forall v \in V$. In the final configuration, all the robots are on a single meeting node $m \in \mathcal{M}$. In the final configuration, each robot must gather at one of the meeting nodes, with the optimization constraint that the sum of the distances traveled by the robots must be minimized. For a configuration to be final, there must exist an $m \in \mathcal{M}$ such that $\lambda_t(m) = n$ and $\lambda_t(v) = 0$ for each $v \in V \setminus \{m\}$. The *consistency* of a node $m \in \mathcal{M}$

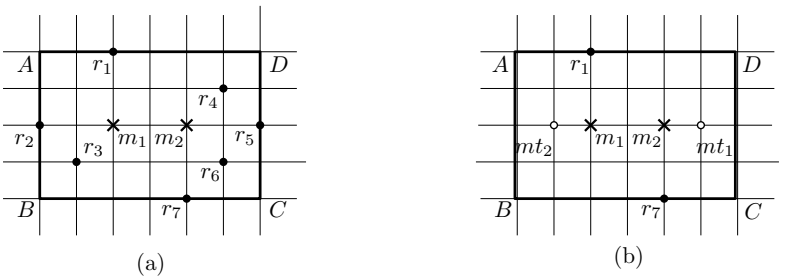

FIGURE 4.1: (a) In this figure, the crosses represent the meeting node and black circles represent robot positions. (b) m_2 remains the unique Weber meeting node, but robots will not be able to compute it correctly if they do not have global-strong multiplicity detection capability.

at any time t is defined as $c_t(m) = \sum_{v \in V} d(v, m)\ \lambda_t(v)$. A node $m \in \mathcal{M}$ is defined as a Weber meeting node if it minimizes the value $c_t(m)$. In other words, a Weber meeting node m is a meeting node that minimizes the sum of the distances from all the robots to itself. A Weber meeting node may not be unique in general. Let $W(t)$ denote the set of all Weber meeting nodes at some time t. A deterministic distributed algorithm that gathers all the robots at a Weber meeting node via the shortest paths will be optimal with respect to the total number of moves made by the robots.

The robots are equipped with global-strong multiplicity detection capability, i.e., they can detect the exact number of robots occupying any node. Without this assumption, the Weber meeting nodes cannot be detected correctly by the robots as soon as a multiplicity is created. As a result, the total number of moves made by the robots to accomplish the gathering might not be optimized. For example, consider the configuration in Figure 4.1(a). If the robots compute the Weber meeting node in this configuration, a unique Weber meeting node would be computed. In Figure 4.1(a), m_2 is the unique Weber meeting node. Due to the robot's movement, if the configuration in Figure 4.1(b) is reached where robots r_2 and r_3 move towards m_2 and create a multiplicity mt_2. Similarly, r_4, r_5 and r_6 move towards m_2 and creates a multiplicity mt_1. Without the global-strong multiplicity detection capability, robots will not be able to compute the unique Weber meeting node correctly. This example shows that without assuming strong multiplicity detection capability, the gathering over Weber meeting nodes is no longer possible as soon as a multiplicity is created.

- **Minimum Enclosing Rectangle**: As in chapter 3, $MER = ABCD$ denotes the

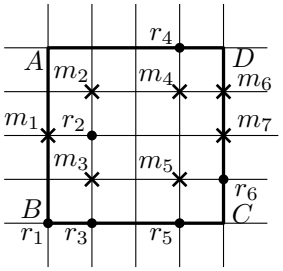

FIGURE 4.2: $ABCD$ denotes the minimum enclosing rectangle MER. Example showing the lexicographic ordering of the strings.

minimum enclosing rectangle of $\mathcal{R} \cup \mathcal{M}$. For a corner i, let the two strings defined are denoted by α_{ij} and α_{ik}. The length of any such string is pq. The strings α_{ij} and α_{ik} are as defined in chapter 3. Thus, there are a total of eight strings of distances of length pq that are obtained by traversing MER. If the meeting nodes are asymmetric, there exists a unique string, which is lexicographically maximum among all the possible strings (follows directly from Lemma 3.3.1). In Figure 4.2, the minimum lexicographic string is α_{DC}= (0, 1, 1, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 1, 0, 0). Here a_{DC}= ((0, 0), (1, 0), (1, 0), (0, 1), (0, 0), (0, 1), (1, 0), (0, 0), (1,0), (0, 1), (0, 0), (0, 0), (0, 0), (0, 0), (0, 0), (0, 0), (1, 0), (0, 1), (1, 0), (0, 1), (0, 0), (0, 0), (1, 0), (0, 0),(0, 1)). If any two possible strings are equal, then the meeting nodes are said to be symmetric. Note that the corner associated with the maximum lexicographic string is defined as a leading corner, and the string associated with the leading corner is defined as the string direction for the respective corner. In the case where MER is a square grid between the two strings associated to a corner, the string direction is defined in the direction which is lexicographically larger, i.e., $\alpha_i = max(\alpha_{ij}, \alpha_{ik})$, where the maximum is defined according to the lexicographic ordering of the strings.

- **Potential Weber meeting nodes**: In general, the Weber meeting node in an infinite grid is not unique. If it is possible to gather at one of the Weber meeting nodes, then all the robots must decide to agree on a common Weber meeting node for gathering. Depending on the symmetricity of the set $\mathcal{M}$, the number of leading corners is $1, 2$ or 4, respectively. Consider the Weber meeting nodes representing the last Weber meeting nodes in the string directions associated with the leading corners. Note that the number of Potential Weber meeting nodes can be at most eight. Let $W_p(t)$ denote the set of such Weber meeting nodes at time $t \geq 0$. The set $W_p(t)$ is defined as the set of *Potential Weber meeting nodes*.

- **Key corner**: In this chapter, the definition of a key corner is slightly different from the definition mentioned in chapter 3. We define the key corner as follows: Consider all the leading corners of MER and the strings α_i associated with each leading corner i. Assume that there exist at least two leading corners. Without loss of generality, assume that i and j are the leading corners and the strings parallel to ij and ji are the string directions associated with the leading corners. The string a_{ij} is defined as follows: Starting from the corner i, scan the grid along the string direction of i, i.e., along ij and associate the pair $(f_t(v), \lambda_t(v))$ to each node v (Figure 4.2). The string a_{ji} is defined similarly. Consider the strings a_{ij} and a_{ji}. If $C(t)$ is asymmetric, there always exists a unique string that is lexicographically smaller between the strings (The proof follows from Lemma 3.3.1. If a_{ij} is lexicographically smaller than a_{ji}, then the corner i is defined as the key corner. If $C(t)$ is symmetric, there may exist more than one key corner. Similarly, the strings b_i, for each non-leading corner i is defined.

In [114], it was proved that a Weber point remains invariant under the movement of a robot towards itself. In our restricted gathering model, where gathering can be finalized only on meeting nodes, we have the following lemma.

Lemma 4.3.1. *Let m be a Weber meeting node in a given configuration $C(t)$. Suppose $C(t')$ denotes the configuration after a single robot or a robot multiplicity moves towards the Weber meeting node m. Then the following results hold.*

1. $m \in W(t')$
2. $W(t') \subseteq W(t)$

Proof. 1. By definition we have, $c_t(m) = \sum_{v \in V} d(v, m)\lambda_t(v)$ and $W(t) = \{m | \min_{m \in M} c_t(m)\}$. Suppose $r(t) = a$ and $r(t') = b$, i.e., r has moved from the vertex a to b along a shortest path towards m in the time interval $[t, t']$. After the movement of the robot r, $\lambda_{t'}(a)$ and $\lambda_{t'}(b)$ become $\lambda_t(a)$ - 1 and $\lambda_t(b) + 1$, respectively. Since b lies on the shortest path from r to m and $d(a, b) = 1$, $c_{t'}(m)$ became $c_t(m) + d(b, m)$ - $d(a, m)$, which is again equivalent to $c_t(m)$ - 1. Therefore, $\min_{m \in M} c_{t'}(m) = \min_{m \in M} c_t(m)$ - 1 and hence $m \in W(t')$. Similarly, if a robot multiplicity moves from some vertex a to b at time t' via any shortest path towards m, then after the movement of the robot

multiplicity, $\lambda_{t'}(a)$ and $\lambda_{t'}(b)$ becomes $\lambda_t(a)$ - j and $\lambda_t(b) + j$, respectively, where $j \geq 2$ denotes the number of robots that move from node a to b. This implies that, $c_{t'}(m) = c_t(m)$- j and hence, $\min_{m \in M} c_{t'}(m) = \min_{m \in M} c_t(m)$- j. Therefore, the Weber meeting nodes in the new configuration $C(t')$ are the Weber meeting nodes of $C(t)$, which are on the some shortest path from r to m. Hence $m \in W(t')$.

2. Assume that $m \in W(t')$. This implies that m minimizes the value $c'_t(m)$. The first part of the proof implies that $\min_{m \in M} c_{t'}(m) = \min_{m \in M} c_t(m)$- j, where $j \geq 1$ denotes the number of robots that move from node a to b. In other words, no node can become a Weber meeting node if it was not before the move. Therefore, m must belong to $W(t)$ and hence $W(t') \subseteq W(t)$.

□

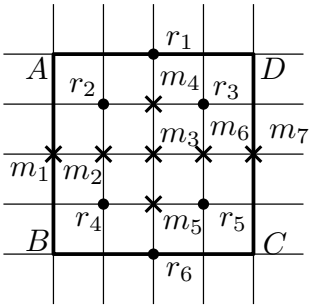

FIGURE 4.3: Multiple Weber meeting nodes in a configuration.

This lemma proves that the Weber meeting node remains invariant under the movement of robots towards itself via a shortest path. The Weber meeting node is not unique in general. In Figure 4.3, the configuration admits rotational symmetry. There are three Weber meeting nodes m_3, m_4 and m_5 in the configuration.

Observation 3. *Let $C(0)$ be any initial configuration that admits rotational symmetry. Assume that the center of the rotational symmetry c contains a meeting node m. Then m is a Weber meeting node.*

4.4 Problem Definition and Impossibility Results

In this section, we have formally defined the problem. A partitioning of the initial configurations has also been provided in this section.

4.4.1 Problem Definition

Let $C(t) = (\mathcal{R}(t), \mathcal{M})$ be a given configuration. The goal of the optimal gathering over Weber meeting nodes problem is to finalize the gathering at one of the Weber meeting nodes of $C(0)$. We have proposed a deterministic distributed algorithm that ensures gathering over a Weber meeting node, where the initial configuration consists of at least seven robots. If $|W(t)| = 1$, then all the robots finalize the gathering at the unique Weber meeting node. Otherwise, all the robots must agree on a common Weber meeting node and finalize the gathering.

4.4.2 Partitioning of the Initial Configurations

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

1. $\mathcal{I}_1-$: Any configuration for which $|W(t)| = 1$ (Figure 4.4(a)).
2. $\mathcal{I}_2-$: Any configuration for which $\mathcal{M}$ is asymmetric and $|W(t)| \geq 2$. In Figure 4.4(b), m_2 and m_3 are the Weber meeting nodes. A is the leading corner.

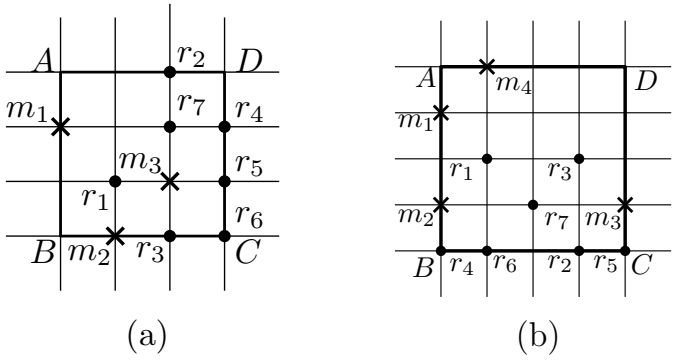

FIGURE 4.4: (a) $\mathcal{I}_1$ configuration. (b) $\mathcal{I}_2$ configuration.

3. $\mathcal{I}_3-$ Any configuration for which $\mathcal{M}$ admits a unique line of symmetry l and $|W(t)| \geq 2$. This can be further partitioned into:
   (a) $\mathcal{I}_3^a-$ $C(t)$ is asymmetric. In Figure 4.5(a), m_3 is the unique Weber meeting node A is the key corner. m_1 and m_4 are the Weber meeting nodes.
   (b) $\mathcal{I}_3^b-$ $C(t)$ is symmetric with respect to the line l. This can be further partitioned into:

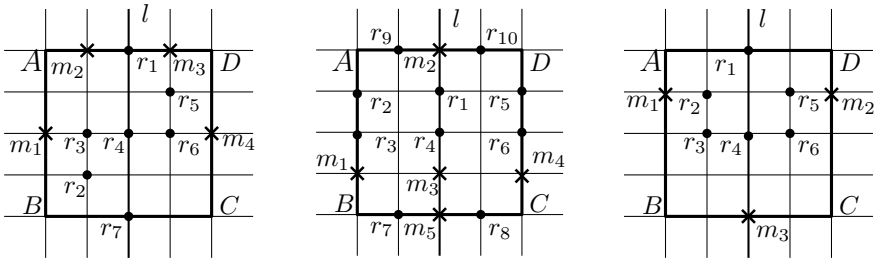

FIGURE 4.5: (a) $\mathcal{I}_3^a$ configuration. (b) $\mathcal{I}_3^{b1}$ configuration. (c) $\mathcal{I}_3^{b2}$ configuration.

- $\mathcal{I}_3^{b1}$ – there exists at least one Weber meeting node on l. In Figure 4.5(b), m_2 and m_3 are the Weber meeting nodes.
- $\mathcal{I}_3^{b2}$ – there exists at least one robot position on l but no Weber meeting nodes on l. In Figure 4.5(c), m_3 is a meeting node on l, but not a Weber meeting node on l. r_1 and r_4 are the robot positions on l.
- $\mathcal{I}_3^{b3}$ – there does not exist any Weber meeting node or robot position on l, but there may exist a meeting node on l. In Figure 4.7(a), a meeting node m_1 on l exists, but it is not a Weber meeting node. m_3 and m_4 are the Weber meeting nodes.
- $\mathcal{I}_3^{b4}$ – there does not exist any meeting node or robot position on l.

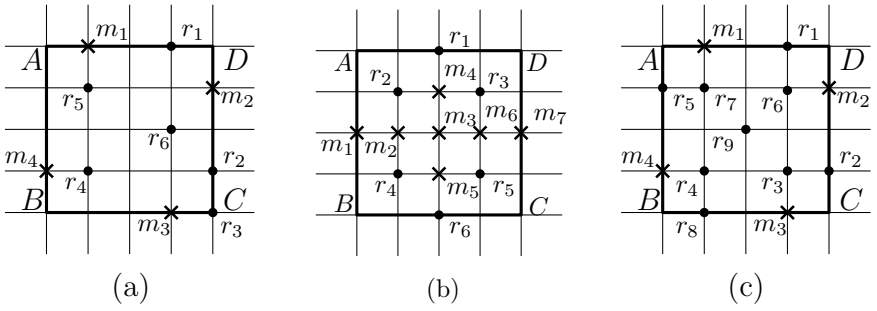

FIGURE 4.6: (a) $\mathcal{I}_4^a$ configuration. (b) $\mathcal{I}_4^{b1}$ configuration. (c) $\mathcal{I}_4^{b2}$ configuration.

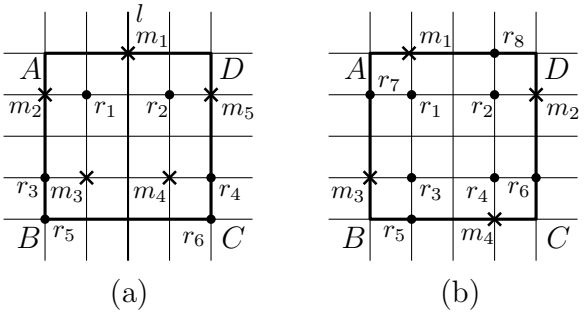

FIGURE 4.7: (a) $\mathcal{I}_3^{b3}$ configuration. (b) $\mathcal{I}_4^{b3}$ configuration without robots or meeting nodes on c.

4. $\mathcal{I}_4-$ Any configuration for which $\mathcal{M}$ admits rotational symmetry with center of rotation c and $|W(t)| \geq 2$. $\mathcal{M}$ may or may not admit multiple lines of symmetry. This can be further partitioned into:

   (a) $\mathcal{I}_4^a-$ $C(t)$ is asymmetric. In Figure 4.6(a), m_2 and m_3 are the Weber meeting nodes.

   (b) $\mathcal{I}_4^b-$ $C(t)$ is symmetric with respect to rotational symmetry, or $C(t)$ may admit a single line of symmetry. First, assume that $C(t)$ admits rotational symmetry with or without multiple lines of symmetry. This can be further partitioned into:

   - $\mathcal{I}_4^{b1}-$ there exists a meeting node on c. In Figure 4.6(b), m_3, m_4 and m_5 are the Weber meeting nodes.
   - $\mathcal{I}_4^{b2}-$ there does exist a meeting node on c, but there exists a robot position on c. In Figure 4.6(c), m_1, m_2, m_3 and m_4 are the Weber meeting nodes. Robot r_9 is on the center of rotation.
   - $\mathcal{I}_4^{b3}-$ there does not exist any meeting node or robot positions on c (Figure 4.7(b)).

If $C(t)$ admits a single line of symmetry, it might be the case that there do not exist any robot positions or meeting nodes on c, but there do exist at least one robot position or meeting nodes on the line of symmetry. In that case, the partitioning of the configuration proceeds similarly, as in $\mathcal{I}_3^b$. We assume that if the meeting nodes are symmetric with respect to a single line of symmetry, then l is the line of symmetry. Similarly, if the meeting nodes are symmetric with respect to rotational symmetry, then c is the center of rotational symmetry. Since the partitioning of the initial configurations depends only on the position of meeting nodes, which are fixed nodes, all the robots can determine the class of configuration in which it belongs without any conflict. Let $\mathcal{I}$ denote the set of all initial configurations.

Lemma 4.4.1. *If the initial configuration $C(0) \in \mathcal{I}_3^{b3} \cup \mathcal{I}_4^{b3}$, then the gathering over Weber meeting nodes problem cannot be solved.*

The proof of the above lemma can be observed as a corollary to Theorem 3.4.1, proved in Chapter 3. In Chapter 3, it was proved that $\mathcal{I}_3^{b4}$ is ungatherable. Let $\mathcal{U}$ denote the set

of all configurations for which gathering over a Weber meeting node cannot be ensured. According to Lemma 4.4.1, this includes all the configurations,

1. admitting a single line of symmetry l, and $l \cap (\mathcal{R} \cup W(t)) = \emptyset$.
2. admitting rotational symmetry with center c and $\{c\} \cap (\mathcal{R} \cup \mathcal{M}) = \emptyset$.

Note that according to Observation 3, if c is a meeting node on c, then it must be a Weber meeting node.

4.5 Algorithm

4.5.1 Overview of the Algorithm

In this subsection, a deterministic distributed algorithm has been proposed to solve the optimal gathering problem by gathering each robot at one of the Weber meeting nodes. The proposed algorithm works for all the configurations $C(t) \in \mathcal{I} \setminus (\mathcal{U} \cup \mathcal{I}_3^{b4})$ consisting of at least seven robots. The main strategy of the algorithm is to select a Weber meeting node among all the possible Potential Weber meeting nodes and allow the robots to move toward the selected Weber meeting node. The proposed algorithm mainly consists of the following phases: *Guard Selection, Target Weber meeting node Selection, Leading Robot Selection, Symmetry Breaking, Creating Multiplicity on Target Weber meeting node* and *Finalization of Gathering*. In the Target Weber meeting node Selection phase, the Potential Weber meeting node for optimal gathering is selected. The Weber meeting node selected for gathering is defined as the target Weber meeting node. A set of robots denoted as guards are selected in the Guard Selection phase. This phase slightly differs from the Guard Selection and Placement phase mentioned in Chapter 3. In this chapter, guards are selected in order to ensure that the initial MER remains invariant. In the Leading Robot Selection phase, a robot is selected as the leading robot and placed. A unique robot is selected and allowed to move toward an adjacent node in the Symmetry Breaking phase. This movement of the robot transforms a symmetric configuration into an asymmetric configuration. In this chapter, the Symmetry Breaking phase proceeds similarly as the Symmetry Breaking phase mentioned in Chapter 3. All the non-guard robots move towards the target Weber meeting node, thus creating a multiplicity on

it in the Creating Multiplicity on Target Weber meeting node phase. Finally, all the guards move towards the uniquely identifiable (robots have global-strong multiplicity detection capability) target Weber meeting node in the Finalization of Gathering phase and finalize the gathering.

4.5.2 Half-planes and Quadrants

Assume that the initial configuration $C(0)$ is asymmetric. First, consider the case when the locations of the meeting nodes are symmetric with respect to a single line of symmetry l. The line l divides MER into two half-planes. The half-planes defined in this section are open half-planes, i.e., the half-planes exclude the nodes on l. If the meeting nodes are symmetric with respect to rotational symmetry and c is the center of rotation, then consider the lines l and l', which pass through c. These lines are perpendicular to each other and divide the MER into four quadrants. The quadrants defined in this section are open quadrants, i.e., the quadrants exclude the nodes belonging to the lines l and l'. A configuration is said to be *balanced* if the following conditions hold:

1. $C(0) \in \mathcal{I}_3^a$ and the half-planes delimited by l contain an equal number of robots.
2. $C(0) \in \mathcal{I}_4^a$. Assume that there exist at least two quadrants that contain the maximum number of Potential Weber meeting nodes. Suppose more than one quadrant contains either the maximum or the minimum number of robots among all the specified quadrants. In that case, the configuration is said to be balanced.

If the initial configuration is not balanced, then it is an *unbalanced* configuration. An initial configuration $C(0)$ satisfies the following conditions:

- C_1: there exists a unique half-plane or quadrant that contains the maximum number of Potential Weber meeting nodes.
- C_2: there exists multiple half-planes or quadrants that contain the maximum number of Potential Weber meeting nodes. Any configuration $C(0)$ satisfying condition C_2 is said to satisfy C_{21}, if $C(0)$ is balanced. Otherwise, it satisfies C_{22} if the initial configuration is unbalanced.

Demarcation of the half-planes for fixing the target	
Initial Configuration $C(0)$	$\mathcal{H}^+$
satisfy C_1	The unique half-plane containing the Potential Weber meeting nodes
satisfy $C_{21} \wedge l$ is a horizontal or vertical line of symmetry	The unique half-plane not containing the key corner
satisfy $C_{21} \wedge l$ is a diagonal line of symmetry $\wedge \exists$ a unique leading corner	The half-plane which lies in the direction of AD, if a_{AD} is lexicographically larger than a_{AB}
satisfy $C_{21} \wedge l$ is a diagonal line of symmetry $\wedge \exists$ two leading corners	The half-plane containing the corners A and D, if a_{AD} is lexicographically larger than a_{CD}
satisfy C_{22}	The unique half-plane with the maximum number of robots

TABLE 4.1: Demarcation of the half-planes

- C_3: there does not exist any Potential Weber meeting node on the half-planes or on the quadrants.

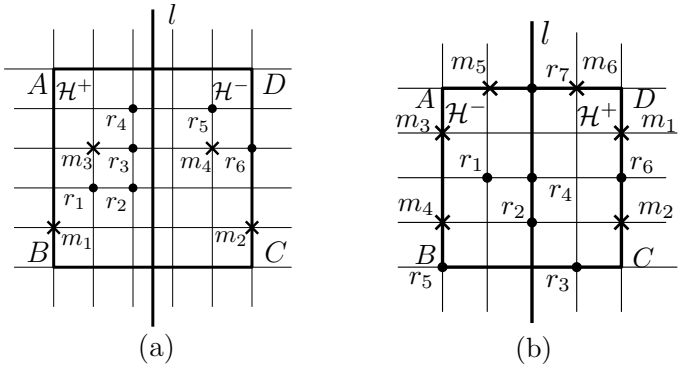

FIGURE 4.8: Examples showing the demarcation of the half-planes.

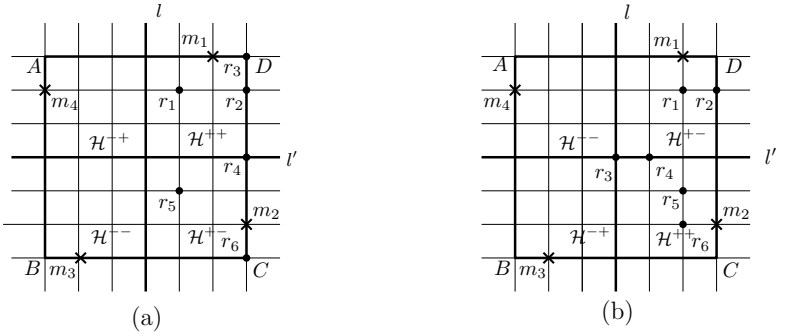

FIGURE 4.9: Examples showing the demarcations of the quadrants.

4.5.2.1 Demarcation of the Half-planes for fixing the target

Assume that the meeting nodes are symmetric with respect to a single line of symmetry l. Note that $| W_p(t) | \leq 2$. Further, assume that $| W_p(t) | = 2$ and $C(0)$ does not satisfy C_3. This implies that there exists at least one Potential Weber meeting node located at the half-planes. Note that if l is a diagonal line of symmetry, then there may exist one or two leading corners. If there exists a unique leading corner, then without loss of generality, let A be the leading corner. Otherwise, if there exist two leading corners, then assume that A and C are the leading corners and the string directions associated to the corners A and C are along the sides AD and CD, respectively. $\mathcal{H}^+$ is defined according to Table 4.1. The other half-plane delimited by l is defined as $\mathcal{H}^-$. In Figure 4.8 (a), m_3 and m_4 are the Weber meeting nodes. $\mathcal{H}^+$ is defined as the half-plane with the maximum number of robots. In Figure 4.8 (b), A and D are the leading corners. A is the key corner. m_2 and m_4 are the Weber meeting nodes. $\mathcal{H}^+$ is defined as the half-plane not containing the key corner A.

4.5.2.2 Demarcation of Quadrants for fixing the target

First, consider the case when the meeting nodes are symmetric with respect to rotational symmetry without multiple lines of symmetry and $W_p(t) \geq 2$. The quadrant $\mathcal{H}^{++}$ is defined according to Table 4.2. The other quadrants are defined as follows.

- $\mathcal{H}^{-+}$:- The quadrant adjacent to $\mathcal{H}^{++}$ with respect to the line l.
- $\mathcal{H}^{+-}$:- The quadrant adjacent to $\mathcal{H}^{++}$ with respect to the line l'.
- $\mathcal{H}^{--}$:- The quadrant which is non-adjacent to $\mathcal{H}^{++}$.

In Figure 4.9(a) m_1 and m_2 are the Weber meeting nodes. $\mathcal{H}^{++}$ denotes the unique quadrant with the maximum number of robots. In Figure 4.9 (b), m_1 and m_2 are the Weber meeting nodes. The quadrants containing the corners C and D contain the maximum number of robots. D is the key corner. $\mathcal{H}^{++}$ denotes the quadrant with the maximum number of robots and not containing the unique key corner.

If MER is a square, and the configuration admits multiple lines of symmetry, there can be at most four lines of symmetry. If there are more than two lines of symmetry, the

Demarcation of the quadrants for fixing the target	
Initial Configuration $C(0)$	$\mathcal{H}^{++}$
satisfy C_1	The unique quadrant containing the maximum number of Potential Weber meeting nodes
satisfy $C_{21} \wedge$ the angle of rotation is 180° $\wedge \exists$ at least one quadrant that contains the Potential Weber meeting nodes as well as the leading corners	The unique quadrant containing the leading corner with which the largest lexicographic string a_i is associated, and that contains the maximum number of robots
satisfy $C_{21} \wedge$ the angle of rotation is 180° $\wedge$ the quadrants that contain the Potential Weber meeting nodes do not contain the leading corners	The unique quadrant containing the non-leading corner with which the largest lexicographic string b_i is associated, and that contains the maximum number of robots
satisfy $C_{21} \wedge$ the angle of rotation is 90°	The quadrant containing the corner with which the largest lexicographic string a_i is associated, and that contains the maximum number of robots
satisfy C_{22}	The unique quadrant with the maximum number of robots
satisfy $C_3 \wedge$ unbalanced	The unique quadrant containing the minimum number of robots
satisfy $C_3 \wedge$ balanced	The unique quadrant that contains the smallest lexicographic string a_i associated with the leading corner and containing the minimum number of robots

TABLE 4.2: Demarcation of the quadrants.

two lines that are perpendicular to each other and do not pass through any corner of MER are selected and considered as l and l'. Consider the quadrants delimited by the lines l and l'. The quadrants are defined similarly, as in the case when MER admits rotational symmetry without multiple lines of symmetry.

4.5.3 Phases of the Algorithm

The proposed algorithm mainly consists of the following phases.

4.5.3.1 Guard Selection

In this phase, a set of robots is selected as guards in order to keep the initial MER invariant. If there do not exist any meeting nodes on a side of the boundary of MER, then there must exist at least one robot on that particular side of the boundary. Guards are selected in such a way that they remain uniquely identifiable. If a side of the

boundary of MER contains at least one meeting node, then a guard robot is not required for that particular side of the boundary. Therefore, consider the case when the boundary of MER does not contain any meeting nodes. Consider the robots which are on the boundary of the MER. First, assume that $C(t)$ is asymmetric. Let G denote the set of guards. Let G_C denote the set of *guard corner* and is defined as follows.

- The unique leading corner, if the meeting nodes are asymmetric.
- The leading corner contained in $\mathcal{H}^+$, if the meeting nodes are symmetric with respect to a horizontal or vertical single line of symmetry l. The unique key corner contained in $\mathcal{H}^+$, if the meeting nodes are symmetric with respect to a diagonal line of symmetry.
- The leading corner contained in $\mathcal{H}^{++}$, if the meeting nodes are symmetric with respect to rotational symmetry.

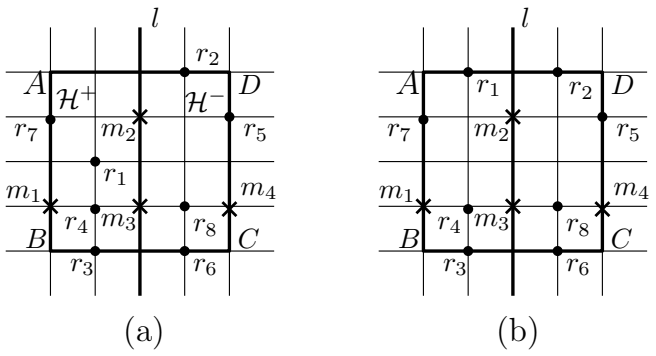

FIGURE 4.10: Example configuration showing the Guard Selection phase.

The robot positions on the sides adjacent to the unique guard corner and closest to the guard corner are considered as guards. Similarly, the robots that are farthest from the guard corner measured along the string direction and lying on the sides non-adjacent to the guard corner are also considered as guards. Note that in each case, there are exactly four guard robot positions that are selected in this phase (In Figure 4.10(a), B is the leading corner contained in $\mathcal{H}^+$. Robots r_2 and r_3 are selected as guards). If $C(t)$ is symmetric with respect to a unique line of symmetry l and l is a horizontal or vertical line of symmetry, there are exactly two leading corners. Consider the robot positions on the sides adjacent to the leading corners and which are closest to the leading corners. These two robots and their symmetric images are selected as guards. The robots which are farthest from the leading corners and lying on the side which is non-adjacent to the

Target Weber meeting node Selection	
Configuration $C(t)$	**Target Weber meeting node**
Admitting a unique Weber meeting node, i.e., $\|W(t)\| = 1$	The unique Weber meeting node
Admitting a unique Potential Weber meeting node, i.e., $\|W_p(t)\| = 1$	The unique Potential Weber meeting node
$\mathcal{I}_3 \wedge$ there exists a Weber meeting node on l	The northernmost Weber meeting node on l
$\mathcal{I}_3^a \wedge$ there does not exist any Weber meeting node on $l \wedge \|W_p(t) = 2\| \wedge l$ is a horizontal or vertical line of symmetry	The Potential Weber meeting node in $\mathcal{H}^+$. Ties are broken by considering the Potential Weber meeting node, which appears last in the string direction associated to the leading corner in $\mathcal{H}^+$
$\mathcal{I}_3^a \wedge$ there does not exist any Weber meeting node on $l \wedge \|W_p(t) = 2\| \wedge l$ is a diagonal line of symmetry $\wedge$ there exists a unique leading corner	The Potential Weber meeting node in $\mathcal{H}^+$ which appears last in the string direction associated to the unique leading corner
$\mathcal{I}_3^a \wedge$ there does not exist any Weber meeting node on $l \wedge \|W_p(t) = 2\| \wedge l$ is a diagonal line of symmetry $\wedge$ there exist two leading corners	The Potential Weber meeting node in $\mathcal{H}^{++}$ that appears last in the string direction associated to the key corner in $\mathcal{H}^+$
$\mathcal{I}_4 \wedge$ there exists a Weber meeting node on c	The Weber meeting node on c
$\mathcal{I}_4^a \wedge$ there does not exist a Weber meeting node on $c \wedge \|W_p(t)\| \geq 2 \wedge$ there does not exist any Weber meeting node on the quadrants	The Potential Weber meeting node which is closest from the unique key corner in the string direction and lying on either l or l'
$\mathcal{I}_4^a \wedge$ there does not exist a Weber meeting node on $c \wedge \|W_p(t)\| \geq 2 \wedge$ there exists a Weber meeting node on the quadrants	The Potential Weber meeting node in $\mathcal{H}^{++}$ which is farthest from the leading corner contained in $\mathcal{H}^{++}$ in the string direction.

TABLE 4.3: Target Weber meeting node selection.

leading corners are also selected as guards. Hence, there are exactly six guard robots that are selected when $C(t)$ is symmetric with respect to l. In Figure 4.10(b), B and C are the leading corners. Robots r_1, r_2, r_3 and r_6 are selected as guards. Otherwise, if l is a diagonal line of symmetry and there exists a unique leading corner, then the robots positions on the sides adjacent to the leading corner and are closest to the leading corner are selected as guards. The robot positions on the sides non-adjacent to the leading corner and farthest from the leading corner are also selected as guards. Note that they are symmetric images of each other. If there are two leading corners, the robots which are closest to the leading corners and lying on the sides adjacent to the leading corners are selected as guards. Note that if $C(t)$ is symmetric with respect to rotational symmetry, then since the center of rotational symmetry is also the center of

fixed meeting nodes and gathering is finalized at the center, the Guard Selection phase is not executed in this case.

Algorithm 4.1: *TargetWeberMeetingNodeSelection()*

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

```
if C(t) ∈ I_1 then
    Select the unique Weber meeting node m;
else if C(t) ∈ I_2 then
    Select the unique Potential Weber meeting node ;
else if C(t) ∈ I_3 then
    if C(t) is asymmetric and l ∩ W(t) ≠ ∅ then
        Select the northernmost Weber meeting node on l
    else if C(t) is asymmetric and (l ∩ W(t) = ∅ ∧ |W_p(t)| = 1) then
        Select the unique Potential Weber meeting node ;
    else if C(t) is asymmetric and (l ∩ W(t) = ∅ ∧ |W_p(t)| = 2) then
        if there exists a unique leading corner in H^+ then
            Select the Potential Weber meeting node in H^+, which appears last in the string
              direction associated to the unique leading corner;
        else if there exist two leading corners in H^+ then
            Select the Potential Weber meeting node that appears last in the string direction
              associated to the key corner in H^+ ;
    else if C(t) is symmetric with respect to the line l then
        Select the northernmost Weber meeting node on l ;
else if C(t) ∈ I_4 then
    if C(t) is asymmetric and {c} ∩ W(t) ≠ ∅ then
        Select the Weber meeting node on c;
    else if C(t) is asymmetric and ({c} ∩ W(t) = ∅∧ | W_p(t) |= 1) then
        Select the unique Potential Weber meeting node ;
    else if C(t) is asymmetric and ({c} ∩ W(t) = ∅∧ | W_p(t) |≥ 2) then
        if All the Potential Weber meeting nodes lie either on line l or l' then
            Select the Potential Weber meeting node which is farthest from the unique key
              corner in the string direction and lying on either l or l';
        else if there exists a Potential Weber meeting node lying on the quadrants then
            Select the Potential Weber meeting node in H^{++} which is farthest from the
              leading corner contained in H^{++} in the string direction ;
    else if C(t) is symmetric with a meeting node on the center of rotation c then
        Select the meeting node on c;
```

4.5.3.2 Target Weber Meeting Node Selection

In this phase, the Weber meeting node for gathering is selected. The target Weber meeting node must remain invariant during the execution of the algorithm. Depending on the class of configuration to which $C(t)$ belongs, the target Weber meeting node is selected according to Table 4.3. The pseudo-code corresponding to this phase is given in Algorithm 4.1. Consider the case when $C(t) \in \mathcal{I}_4^a$ and there exists a Weber meeting node on the quadrants. Further, assume that $|W_p(t)| \geq 2$. If there exist two string directions corresponding to the unique leading corner in $\mathcal{H}^{++}$, the target Weber meeting node is

Leading Robot Selection	
Configuration $C(t)$	**Leading Robot**
$\mathcal{I}_3^a \wedge$ there exists a robot on l	The northernmost robot on l
$\mathcal{I}_3^a \wedge$ there does not exist any robot on l	The robot closest to l and lying on $\mathcal{H}^-$. Ties are broken by considering the robot on $\mathcal{H}^-$ which is farthest from the leading corner contained in $\mathcal{H}^-$ in the string direction
$\mathcal{I}_4^a \wedge$ there exists a robot either on l or l'	The robot closest to the target Weber meeting node and lying on l or l'. Ties are broken by considering the robot either on l or l' which is closest from the leading corner contained in H^{++} in the string direction
$\mathcal{I}_4^a \wedge$ there does not exist any robot on l and $l' \wedge$ there exists a non-guard robot in a quadrant adjacent to $\mathcal{H}^{++}$	The robot lying on a quadrant adjacent to $\mathcal{H}^{++}$ and closest to the target Weber meeting node. Ties are broken by considering the robot, which is closest from the leading corner contained in H^{++} in the string direction
$\mathcal{I}_4^a \wedge$ there does not exist any robot on l and $l' \wedge$ there does not exist any non-guard robot in the quadrants adjacent to $\mathcal{H}^{++}$	The robot lying on the quadrant non-adjacent to $\mathcal{H}^{++}$ and closest to the target Weber meeting node. Ties are broken by considering the robot, which is closest from the leading corner contained in H^{++} in the string direction

TABLE 4.4: Leading Robot Selection.

selected as the Potential Weber meeting node in $\mathcal{H}^{++}$, which appears first in the string a_i.

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, the northernmost meeting node on l is defined as the meeting node on l which is farthest from the leading corner(s). Similarly, the northernmost robot on l is defined.

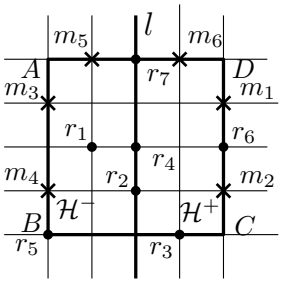

FIGURE 4.11: Balanced $\mathcal{I}_3^a$ configuration. Example configuration showing the Leading Robot Selection phase.

4.5.3.3 Leading Robot Selection

If the initial configuration is balanced and asymmetric, a robot r is selected as a leading robot in the Leading Robot Selection phase. In Figure 4.11, m_2 and m_4 are the Weber meeting nodes lying on different half-planes. m_2 is selected as the target Weber meeting node. Robot r_2 is selected as the leading robot. The leading robot moves towards the half-plane or quadrant containing the target Weber meeting node m. While r reaches the half-plane or the quadrant containing m, the configuration transforms into an unbalanced configuration, and the asymmetry of the configuration remains invariant. Since the initial configuration is balanced, assume that $C(t) \in \mathcal{I}_3 \cup \mathcal{I}_4$. Further, assume that the initial configuration does not satisfy the condition C_3. Depending on the class of configuration to which $C(t)$ belongs, the leading robot is selected according to Table 4.4. In case the configuration is in $\mathcal{I}_4^a$ and there exists a robot on l (resp. l'), the leading robot first moves along the line l (resp. l') and when it becomes collinear with m, it starts moving along l' (resp. l).

4.5.3.4 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. The following cases are to be considered.

1. $C(t) \in \mathcal{I}_3^{b2}$. In this class of configurations, at least one robot exists on l. Let r be the northernmost robot on l. r moves towards an adjacent node that does not belong to l, and the configuration becomes asymmetric.

2. $C(t) \in \mathcal{I}_4^{b2}$. In this class of configurations, there exists a robot (say r) on c. This phase proceeds similarly in the case when $C(t) \in \mathcal{I}_{32}$, considered in the Symmetry Breaking phase in Chapter 3.

4.5.3.5 Creating Multiplicity on Target Weber meeting node

The target Weber meeting node m is selected in the Target Weber meeting node Selection phase. Since there is a unique target Weber meeting node m, all the non-guard robots move towards m in the Creating Multiplicity on Target Weber meeting node phase.

Note that since the guards do not move during this phase, the MER remains invariant. As a result, m remains invariant. Eventually, a robot multiplicity is created on m while the non-guards move toward it. Depending on the class of configuration to which $C(t)$ belongs, the following cases are to be considered.

1. $C(t) \in \mathcal{I}_1$: All the robots move towards the unique Weber meeting node m. In Figure 4.4 (a), each robot moves towards the unique Weber meeting node m_3.

2. $C(t) \in \mathcal{I}_2$: All the non-guards move towards the unique Potential Weber meeting node, which is selected as the target meeting node m on l. In Figure 4.4 (b), each robot moves towards the unique Potential Weber meeting node m_3.

3. $C(t) \in \mathcal{I}_3$: If $C(t) \in \mathcal{I}_3^a$ and there exists a Weber meeting node on l, each non-guard moves towards the target meeting node m. Since the northernmost agreement depends on the position of meeting nodes, m remains invariant while the non-guards move towards it.

   Next, consider the case when $C(t) \in \mathcal{I}_3^a$ and there does not exist any Weber meeting node on l. A leading robot in the Leading Robot Selection phase transforms a balanced configuration into an unbalanced configuration. All the non-guard robots from $\mathcal{H}^-$ move towards m. This movement is required in order to ensure that $\mathcal{H}^+$ remains invariant. While such robots reach $\mathcal{H}^+$, all the non-guard robots in $\mathcal{H}^+$ move towards m, thus creating a multiplicity on m. In Figure 4.5 (a), the target Weber meeting node is m_1, lying on $\mathcal{H}^+$.

   Finally, if $C(t) \in \mathcal{I}_3^{b1}$, each non-guard which is closest to m, moves towards m either synchronously or there may be a possible pending move due to the asynchronous behavior of the scheduler. Ties are broken by considering the closest robots which are farthest from the leading corners in their respective string directions. In Figure 4.5 (b), each robot moves towards the Weber meeting node m_2 on l.

4. $C(t) \in \mathcal{I}_4$: First consider the case when the target Weber meeting node m is the center of rotational symmetry c. Each robot moves towards m on c. In Figure 4.6 (b), each robot moves towards the Weber meeting node m_3 on c.

   Next, consider the case when $C(t) \in \mathcal{I}_4^a$ and the target Weber meeting node m is on $\mathcal{H}^{++}$. After the Leading Robot Selection phase, each non-guard robot in the quadrants different from $\mathcal{H}^{++}$ as well as on the lines l or l' moves towards the target

Weber meeting node m in $\mathcal{H}^{++}$. First, the non-guards in the quadrants adjacent to $\mathcal{H}^{++}$ and the robots on l or l' move towards $\mathcal{H}^{++}$. While they reach $\mathcal{H}^{++}$, the non-guards in the quadrants non-adjacent to $\mathcal{H}^{++}$ move towards $\mathcal{H}^{++}$. This movement is required in order to ensure that $\mathcal{H}^{++}$ remains invariant. While such non-guards reach $\mathcal{H}^{++}$, each non-guards in $\mathcal{H}^{++}$ moves towards m, thus creating a robot multiplicity on m. Otherwise, consider the case when the target meeting node m is on either l or l'. First, all the non-guards in $\mathcal{H}^{++}$ move towards m. While they reach l or l', the other non-guards move towards m. This movement is required in order to ensure that $\mathcal{H}^{++}$ remains invariant. Finally, a robot multiplicity is created at m.

If there exist eight Potential Weber meeting nodes, then there exist exactly two Potential Weber meeting nodes in $\mathcal{H}^{++}$. There exist two string directions corresponding to the unique leading corner in $\mathcal{H}^{++}$. The robot closest to the target meeting node m and appearing first in a_i, which is not on m, moves towards m. After such a move of the robot, it ensures that m remains invariant during the procedure. The procedure proceeds similarly as before.

Note that if the configuration is asymmetric or symmetric with respect to rotational symmetry, there may exist at most four robot positions that are not on m during this phase. Otherwise, if the configuration is symmetric with respect to a horizontal or vertical line of symmetry, there may exist at most six robot positions that are not on m.

4.5.3.6 Finalization of Gathering

Let m be the target Weber meeting node, where a robot multiplicity is created during the Creating Multiplicity on Target Weber meeting node phase. The following are the cases in which the robots will identify that the Finalization of Gathering is in progress:

1. The configuration has at most four robot positions that are not on m, which includes robot multiplicity. Moreover, each side of the MER contains at most one robot position.

2. The configuration has exactly six robot positions that are not on m, which includes robot multiplicity. Moreover, there exist exactly two sides of the MER that contain two robot positions and are symmetric images of each other.

In this phase, all the guards move towards m. During their movement, they do not create any multiplicity on a Weber meeting node other than m. In order to ensure this, all the guards first move along the boundary of MER, and when it becomes collinear with m, it starts moving towards m. A guard robot moves by minimizing the Manhattan distance between m and itself. This implies that during their movement, no other multiplicity would be created on any other Weber meeting node and gathering would be finalized on m.

4.5.4 Optimal Gathering()

Our main algorithm, Optimal Gathering(), considers the following cases: If $C(t) \in \mathcal{I}_1$, then each robot finalizes the gathering on the unique Weber meeting node.

Otherwise, consider the case when the meeting nodes are asymmetric and there exist multiple Weber meeting nodes. There exists a unique Potential Weber meeting node. The guards are selected in the Guard Selection phase. Each non-guard moves towards the unique Potential Weber meeting node, creating a multiplicity on it. Finally, the guards move towards the multiplicity and finalize the gathering at it.

Next, consider the case when the configuration is balanced and asymmetric. A leading robot is selected in the Leading Robot Selection phase, which transforms the configuration into an unbalanced configuration. The guards are selected in the Guard Selection phase. In the Creating Multiplicity on the Target Weber meeting node phase, each non-guard moves towards the target Weber meeting node selected in the Target Weber meeting node Selection phase. Finally, the guards move towards the multiplicity and finalize the gathering.

If $C(t)$ is symmetric and there exists a Weber meeting node on $l \cup \{c\}$, the gathering is finalized at the target Weber meeting node m selected in the Target Weber meeting node Selection phase. Otherwise, if the configuration is symmetric and there exists a robot on either l or c, then in the Symmetry Breaking phase, the configuration is transformed into an asymmetric configuration, and the algorithm proceeds similarly as in the case when the configuration is asymmetric. Note that, in case $C(t) \in \mathcal{I}_3^b$, there may exist exactly six robots that are selected as guards. In case $n = 7$, there must exist at least one robot position on l. The northernmost robot on l moves towards an

adjacent node away from l if there do not exist any Weber meeting nodes on l. Hence, the configuration becomes asymmetric, and the algorithm proceeds similarly, as in the asymmetric case for $n = 7$. Otherwise, if there exists at least one Weber meeting node on l, the northernmost Weber meeting node m on l is selected as the target Weber meeting node. The closest robot on l and the northernmost, in the case of a tie, moves towards m. While each non-guard moves towards m, it remains invariant. After the non-guards reach m, m remains uniquely identifiable and the gathering is finalized in the Finalization of Gathering phase.

4.6 Correctness

In this section, we describe the correctness of our proposed algorithm. Lemmas 4.6.1 and 4.6.2 prove that the leading robot remains invariant during the movement towards its destination.

Lemma 4.6.1. *If $C(t) \in \mathcal{I}_3^a$, then in the Leading Robot Selection phase, the leading robot remains the unique robot while it moves towards the half-plane $\mathcal{H}^+$.*

Proof. Let $C(t)$ be any balanced configuration that belongs to $\mathcal{I}_3^a$. Since the configuration is balanced and asymmetric, the number of robots in the two half-planes delimited by l is equal and there exists a unique key corner. If there exists at least one robot position on l, then the northernmost robot on l is the leading robot. The northernmost robot moves towards an adjacent node away from l, and the configuration becomes unbalanced. Consider the case when there does not exist any robot position on l. Without loss of generality, assume that l is a vertical line of symmetry. Let r be the leading robot in $\mathcal{H}^-$ selected in the Leading Robot Selection phase. Without loss of generality, let A be the unique key corner and $a_{AD} = a_1, a_2, \ldots, a_{pq}$ is the unique smallest lexicographic string associated to the corner A. Similarly, let B be the other leading corner and $a_{BC} = b_1, b_2, \ldots, b_{pq}$ be the string associated to B. Let u_i and v_i denote the nodes, which the positions a_i and b_i represent in a_{AD} and a_{BC}, respectively. Since the meeting nodes are symmetric, $f_t(u_i) = f_t(v_i)$, for each $i = 1, 2 \ldots, pq$. As $a_{AD} = a_1, a_2, \ldots, a_{pq}$ is the unique smallest lexicographic string among the $a_i's$, there must exist a position k' such that $\lambda_t(u_{k'}) = 0 < \lambda_t(v_{k'}) = 1$. Without loss of generality, let i be the position of the leading robot in a_{AD}. Let k be the first position, where $\lambda_t(u_k)$ and $\lambda_t(v_k)$ differ.

Note that, $\lambda_t(u_k)$=0 and $\lambda_t(v_k) = 1$. We have to prove that after the movement of the leading robot, $a_{AD} <_l a_{BC}$, where $' <'_l$ denotes the relation that a_{AD} is lexicographically smaller than a_{BC}. Assume that at time t', the leading robot moves towards an adjacent node. Depending on the possible values of i and k in a_{AD}, the following cases are to be considered.

Case 1. The position of i is less than k in a_{AD}. While the leading robot moves towards l, $\lambda_t(u_i)$ becomes 0, but $\lambda_t(v_i)$ equals 1. After the movement of the leading robot towards an adjacent node, $a_{AD} <_l a_{BC}$.

Case 2. The position of i is equal to k in a_{AD}. Since each robot is deployed at distinct nodes of the grid in the initial configuration, this case is not possible.

Case 3. The position of i is greater than k in a_{AD}. While the leading robot moves towards l, the position k remains invariant. After the movement of the leading robot towards an adjacent node, $a_{AD} <_l a_{BC}$.

Note that, after a single movement of the leading robot towards l, it becomes the unique robot that is eligible to move towards $\mathcal{H}^+$. Since a_{AD} remains the unique lexicographically smallest string at t', $\mathcal{H}^+$ remains invariant. Clearly, after a finite number of movements towards l, H^+ remains invariant, and ultimately, the configuration becomes unbalanced. The proof is similar when the meeting nodes admit a horizontal or a diagonal line of symmetry. □

Lemma 4.6.2. *If $C(t) \in \mathcal{I}_4^a$, then in the Leading Robot Selection phase, the leading robot remains the unique robot while it moves towards the target Weber meeting node.*

Proof. Let $C(t)$ be any balanced configuration that belongs to $\mathcal{I}_4^a$. Since the configuration is balanced and asymmetric, there exist at least two quadrants that contain the maximum number of Potential Weber meeting nodes with the maximum number of robots on such quadrants. We have to prove that while the leading robot moves towards the target Weber meeting node, the quadrant $\mathcal{H}^{++}$ remains invariant. First, consider the case when the leading robot r is on either l or l'. Note that in this case, r may be one or more than one node away from $\mathcal{H}^{++}$. There is nothing to prove when r is one node away from $\mathcal{H}^{++}$. In this case, a move of r transforms the configuration into an unbalanced configuration. Therefore, consider the case when r is more than one node away from $\mathcal{H}^{++}$. Without loss of generality, first assume that r is on l. Let $MER = ABCD$

be such that the corner C is the corner diagonally opposite to A and the corners A and B are separated by line l. Similarly, A and D are the corners separated by line l'. $\mathcal{H}^{++}$ is the quadrant containing A. Let $a_{AD} = a_1, a_2, \ldots a_{pq}$ and $a_{BC} = b_1, b_2, \ldots b_{pq}$ be the strings associated to the corners A and B. While r moves along l, we have to prove that a_{AD} remains lexicographically larger than a_{BC}. It is noteworthy that a_{AD} is lexicographic larger than a_{CB} and a_{DA} while r moves. Note that we have considered the case when the string directions are along the width of the rectangle. Let i be the position of the leading robot in a_{AD} and a_{BC}. Let u_i and v_i denote the nodes, which the positions a_i and b_i represent in a_{AD} and a_{BC}, respectively. Since the meeting nodes are symmetric, $f_t(u_i) = f_t(v_i)$, for each $i = 1, 2 \ldots, pq$. After a movement of the leading robot along the line l and towards the corner A, note that the first position in which the strings a_{AD} and a_{BC} differ, remains invariant. As a result, $\mathcal{H}^{++}$ remains invariant. After a finite number of movements, the robot r becomes one node away from $\mathcal{H}^{++}$, and the proof proceeds similarly as before. The proof is similar in the case when the string directions are along the length of the rectangle. In that case, we have to compare the strings a_{AB} and a_{BA}. The proof is trivial while comparing the strings a_{AB} and a_{CD}, and a_{AB} and a_{DC}. Next, consider the case when the leading robot is on a quadrant adjacent to $\mathcal{H}^{++}$. Without loss of generality, assume that the leading robot is on $\mathcal{H}^{+-}$. While the leading robot moves, it can be observed that a_{AD} is lexicographically larger than a_{CB} and a_{DA}. We have to prove that a_{AD} remains lexicographic larger than a_{BC} while the leading robot moves. Let i and j be the positions of the leading robot in a_{AD} and a_{BC}, respectively. Note that $i < j$, as the leading robot is selected on $\mathcal{H}^{+-}$. Let k be the first position for which $b_k < a_k$. Depending on the values of i, j and k in a_{AD}, we have the following cases.

Case 1. $i < j < k$. Note that before the move of the leading robot, u_{i-1} cannot be a robot position as otherwise r would not be selected as a leading robot. After the movement of the leading robot, u_{i-1} is a robot position, but v_{i-1} cannot be a robot position as k is the first position where $a_k > b_k$. Therefore, a_{AD} remains lexicographic larger than a_{BC} while the leading robot moves.

Case 2. $i < j = k$. Note that k is the first position where the strings a_{AD} and a_{BC} differs. Since b_j represents a robot position containing the leading robot and each robot is deployed at the distinct nodes of the grid in the initial configuration, this case is not possible.

Case 3. $i < k < j$. The proof is similar to Case 1.

Case 4. $i = k < j$. This implies that a_k contains a robot position. After the move of the leading robot towards the target Weber meeting node, a_{k-1} is a robot position in a_{AD} and b_{k-1} is not a robot position. As a result, $a_{k-1} > b_{k-1}$. Therefore, a_{AD} remains lexicographic larger than a_{BC} while the leading robot moves.

Case 5. $k < i < j$. In this case, a_k must represent a robot position. After a move of r towards the target Weber meeting node, $a_k > b_k$. Therefore, a_{AD} remains lexicographic larger than a_{BC} while the leading robot moves.

The proof is similar when the string directions are along the lengths of MER. Next, consider the case when the leading robot r is selected on a quadrant non-adjacent to $\mathcal{H}^{++}$. Without loss of generality, we assume that r first starts moving towards l'. The proof is similar when r first moves towards l. While the leading robot moves, it can be observed that a_{AD} is lexicographically larger than a_{CB} and a_{DA}. We have to prove that a_{AD} remains lexicographic larger than a_{BC} while r moves. Let i and j be the positions of the leading corner in a_{AD} and a_{BC}, respectively. Note that $i > j$, as the leading robot is selected on $\mathcal{H}^{--}$. Let k be the first position for which $b_k < a_k$. Depending on the values of i, j and k in a_{AD}, we have the following cases.

Case 1. $i > j > k$. After the movement of the leading robot towards l', the position k remains invariant. Therefore, a_{AD} remains lexicographic larger than a_{BC} while the leading robot moves.

Case 2. $i > j = k$. Since each robot is deployed at the distinct nodes of the grid in the initial configuration, this case is not possible.

Case 3. $i = k > j$. There must not be a robot position at the node represented by a_{k-1}, as otherwise, r would not be selected as the leading robot. After the move of the leading robot towards the target Weber meeting node, a_{k-1} is a robot position while b_{k-1} cannot be a robot position, as k is the first position for which $b_k < a_k$. Therefore, a_{AD} remains lexicographic larger than a_{BC} while the leading robot moves.

Case 4. $k > i > j$. Note that a_{i-1} cannot be a robot position before the move, otherwise r would not be selected as the leading robot. After a move of r, a_{i-1} is a robot position, but b_{i-1} cannot be a robot position as k is the first position where a_k and b_k differ. Therefore, a_{AD} remains lexicographic larger than a_{BC} while the leading robot moves.

Case 5. $j < k < i$. After a move of r, it may be the case that $k = i - 1$ in a_{AD}. In that case, $\lambda_t(u_{i-1}) = 2$, but $\lambda_t(v_{i-1}) = 0$. Otherwise, the proof is similar as $a_k > b_k$. Therefore, a_{AD} remains lexicographic larger than a_{BC} while the leading robot moves.

As a result, the leading robot remains invariant while it moves towards its destination.

□

The next three lemmas prove that the target Weber meeting node remains invariant in the Creating Multiplicity on the Target Weber meeting node phase.

Lemma 4.6.3. *If $C(t) \in \mathcal{I}_2 \cup \mathcal{I}_3^{b1} \cup \mathcal{I}_4^{b1}$, then the target Weber meeting node remains invariant in the Creating Multiplicity on Target Weber meeting node phase.*

Proof. In the Creating Multiplicity on Target Weber meeting node phase, all the non-guard robots move towards the target Weber meeting node. According to Lemma 4.3.1, the Weber meeting node remains invariant under the movement of robots towards itself. The following cases are to be considered.

Case 1. $C(t) \in \mathcal{I}_2$: Since the meeting nodes are asymmetric, there exists a unique Potential Weber meeting node. The unique Potential Weber meeting node is selected as the target Weber meeting node. The unique Potential Weber meeting node of a configuration is defined with respect to the position of the leading corner. The leading corner remains invariant unless the MER changes. As the guards do not move in the Creating Multiplicity on Target Weber meeting node phase, the MER remains invariant. Hence, the target Weber meeting node remains invariant.

Case 2. $C(t) \in \mathcal{I}_3^{b1}$: The northernmost Weber meeting node on l is selected as the target Weber meeting node. Since the northernmost agreement depends on the position of the leading corner(s), the agreement remains invariant unless the MER changes. As the guards do not move in the Creating Multiplicity on Target Weber meeting node phase, the MER remains invariant. Hence, the target Weber meeting node remains invariant.

Case 3. $C(t) \in \mathcal{I}_4^{b1}$: The center of rotational symmetry c is the target Weber meeting node. Since c is also the center of rotational symmetry for the meeting nodes, the target Weber meeting node remains invariant.

□

Lemma 4.6.4. *If $C(t) \in \mathcal{I}_3^a$, then the target Weber meeting node remains invariant in the Creating Multiplicity on Target Weber meeting node phase.*

Proof. The meeting nodes are symmetric with respect to a single line of symmetry l. In the Creating Multiplicity on Target Weber meeting node phase, all the non-guard robots move towards the target Weber meeting node on l. Note that the MER remains invariant unless the guard moves. The following cases are to be considered.

Case 1. There exists at least one Weber meeting node on l. The northernmost Weber meeting node on l is selected as the target Weber meeting node. Since the northernmost agreement depends on the position of the leading corner(s), and the leading corner(s) remains invariant unless the MER changes, the agreement remains invariant. Hence, the target Weber meeting node remains invariant.

Case 2. $C(0)$ satisfy C_1. We have to prove that $\mathcal{H}^+$ remains invariant in the Creating Multiplicity on Target Weber meeting node phase. Note that, in this phase, all the non-guards move towards the target Weber meeting node. According to Lemma 4.3.1, the Weber meeting nodes remain invariant while the robots move towards it. As MER remains invariant unless the guard robot moves, the leading corner(s) remains invariant. Since the Potential Weber meeting nodes are defined with respect to the positions of the leading corner(s), $\mathcal{H}^+$ remains invariant. Hence, the target Weber meeting node remains invariant.

Case 3. $C(0)$ satisfy C_2. The target Weber meeting node is selected in the half-plane $\mathcal{H}^+$. We have to prove that $\mathcal{H}^+$ remains invariant while the robots move towards the target Weber meeting node. The following subcases are to be considered.

Subcase 1. $C(0)$ satisfy C_{22}. $\mathcal{H}^+$ is the half-plane that contains the maximum number of robots. All the non-guard robots in $\mathcal{H}^-$ move towards the target Weber meeting node in $\mathcal{H}^+$. During this movement of the robots, $\mathcal{H}^+$ still contains the maximum number of robots. Hence, the target Weber meeting node remains invariant.

Subcase 2. $C(0)$ satisfy C_{21}. The leading robot in $\mathcal{H}^-$ moves towards the target Weber meeting node in $\mathcal{H}^+$, resulting in transforming the configuration into an unbalanced configuration. While the leading robot moves towards $\mathcal{H}^+$, the unique lexicographic

smallest string a_i remains invariant according to Lemma 4.6.1. As the key corner remains invariant, $\mathcal{H}^+$ remains invariant. The moment, the leading robot reaches l, the configuration becomes unbalanced. The rest of the proof follows from the previous case. □

Lemma 4.6.5. *If $C(t) \in \mathcal{I}_4^a$, then the target Weber meeting node remains invariant in the Creating Multiplicity on Target Weber meeting node phase.*

Proof. Since $C(t) \in \mathcal{I}_4^a$, the meeting nodes are symmetric with respect to rotational symmetry. Let c be the center of the rotational symmetry for $\mathcal{M}$. According to Lemma 4.3.1, the Weber meeting nodes remain invariant while all the robots move towards it. The following cases are to be considered.

Case 1. There exists a Weber meeting node on c. It is selected as the target Weber meeting node. Since c is the center of rotational symmetry for the fixed meeting nodes, the target Weber meeting node remains invariant while the robots move towards it.

Case 2. $C(0)$ satisfy C_1. The target Weber meeting node is selected in $\mathcal{H}^{++}$ as the Weber meeting node, which is farthest from the leading corner in the string direction and contained in $\mathcal{H}^{++}$. Note that since the guards do not move during this phase, the MER remains invariant. As a result, the Potential Weber meeting nodes and $\mathcal{H}^{++}$ remain invariant. Hence, the target Weber meeting node remains invariant.

Case 3. $C(0)$ satisfy C_2. We have to prove that $\mathcal{H}^{++}$ remains invariant while the robots move towards the target Weber meeting node. Considering such quadrants that contain the maximum number of Potential Weber meeting nodes, the target Weber meeting node is selected as the Potential Weber meeting node in $\mathcal{H}^{++}$. Ties are broken by considering the Weber meeting node in $\mathcal{H}^{++}$, which is farthest from the leading corner in $\mathcal{H}^{++}$ in the string direction. If $C(0)$ satisfies C_{22}, first, all the non-guard robots in the quadrants adjacent to $\mathcal{H}^{++}$ and on $l \cup l'$ move towards the target Weber meeting node in $\mathcal{H}^{++}$. Finally, the other non-guard robots move towards m. Since $\mathcal{H}^{++}$ is the unique quadrant that contains the maximum number of robot positions, it still contains the maximum number of robots while all such robots reach $\mathcal{H}^{++}$. Hence, the target Weber meeting node remains invariant. Otherwise, if $C(0)$ satisfies C_{21}, there exists more than one quadrant that contains the maximum number of robot positions. Considering such quadrants and the corners contained in those quadrants. $\mathcal{H}^{++}$ is the quadrant containing the largest lexicographic string among those $a_i's$ that are associated with the

leading corners contained in such quadrants. A leading robot is selected in the Leading Robot Selection phase and is allowed to move towards the target Weber meeting node in $\mathcal{H}^{++}$. While the leading robots move towards the target Weber meeting node in $\mathcal{H}^{++}$, $\mathcal{H}^{++}$ remains invariant according to Lemma 4.6.2. As a result, the configuration becomes unbalanced. The rest of the proof follows similarly, as in the case, when $C(0)$ satisfies C_{22}.

Case 4. $C(0)$ satisfy C_3. The target Weber meeting node is selected on either l or l'. Note that, in this case, if the configuration is unbalanced, $\mathcal{H}^{++}$ is the quadrant that contains the minimum number of robots. Otherwise, if the configuration is balanced, then $\mathcal{H}^{++}$ is the quadrant containing the smallest lexicographic string among all those $a_i's$. In both cases, all the non-guard robots on $l \cup l'$, and the non-guard robots on $\mathcal{H}^{++}$, move towards the target Weber meeting node m. After such robots reach $\mathcal{H}^{++}$, $\mathcal{H}^{++}$ remains the unique quadrant with the minimum number of robots. As a result, $\mathcal{H}^{++}$ remains invariant. Hence, the target Weber meeting node remains invariant. □

The next two lemmas prove that any initial configuration $C(0) \in \mathcal{I} \setminus \mathcal{U}$, would never reach a configuration $C(t) \in \mathcal{U}$, at any point of time $t > 0$ during the execution of the algorithm Optimal Gathering().

Lemma 4.6.6. *Given $C(0) \in \mathcal{I}_3$ and $t > 0$ be an arbitrary instant of time at which at least one robot has completed its LCM cycle. If $C(0) \notin \mathcal{I}_3^{b3}$, then during the execution of the algorithm Optimal Gathering(), $C(t) \notin \mathcal{I}_3^{b3}$.*

Proof. According to Lemma 4.3.1, the Weber meeting nodes remain invariant while the robots move towards it. Since the meeting nodes admit a single line of symmetry l and there does not exist any Weber meeting node on l, assume that $C(0) \in \mathcal{I}_3^a \cup \mathcal{I}_3^{b2}$. The following cases are to be considered.

Case 1. $C(0) \in \mathcal{I}_3^a$. Note that there does not exist any Weber meeting node on l, otherwise according to Lemma 4.3.1, $C(t) \notin \mathcal{I}_3^{b3}$. Depending on the position of Potential Weber meeting nodes, the following subcases may arise.

Subcase 1. $C(0)$ satisfy C_1. All the non-guard robots in $\mathcal{H}^- \cup l$ move towards the target Weber meeting node in $\mathcal{H}^+$. So, at any arbitrary instant of time $t > 0$, $C(t)$ remains asymmetric and hence $C(t) \notin \mathcal{I}_3^{b3}$.

Subcase 2. $C(0)$ satisfy C_2. If the configuration is unbalanced, all the robots in $\mathcal{H}^- \cup l$ move towards the target Weber meeting node in $\mathcal{H}^+$. As a result, the configuration remains unbalanced and hence asymmetric. Otherwise, if the configuration is balanced, the leading robot moves towards the target Weber meeting node at some time $t' > 0$. According to Lemma 4.6.1, the configuration remains asymmetric during its movement and ultimately, the configuration becomes unbalanced. Proceeding similarly, as in the unbalanced case, at any arbitrary instant of time $t > 0$, $C(t)$ remains asymmetric and hence $C(t) \notin \mathcal{I}_3^{b3}$, where t denotes an instant of time such that $t \geq t'$.

Case 2. $C(0) \in \mathcal{I}_3^{b2}$. Assume that at time $t' > 0$, the northernmost robot on l moves towards an adjacent node away from l, which transforms the configuration into an unbalanced asymmetric configuration. The rest of the proof follows from the previous case. Hence, $C(t) \notin \mathcal{I}_3^{b3}$, where t denotes an instant of time such that $t \geq t'$. □

Lemma 4.6.7. *Given $C(0) \in \mathcal{I}_4$ and $t > 0$ be an arbitrary instant of time at which at least one robot has completed its LCM cycle. If $C(0) \notin \mathcal{I}_4^{b3}$, then during the execution of the algorithm Optimal Gathering(), $C(t) \notin \mathcal{I}_4^{b3}$.*

Proof. According to Lemma 4.3.1, the Weber meeting nodes remain invariant while the robots move towards it. Since the meeting nodes admit rotational symmetry and there does not exist any Weber meeting node on c, assume that $C(0) \in \mathcal{I}_4^a \cup \mathcal{I}_4^{b2}$. Otherwise, $C(t) \notin \mathcal{I}_4^{b3}$. The following cases are to be considered.

Case 1. $C(0) \in \mathcal{I}_4^a$. If there exists a Weber meeting node on c, then all the robots move towards it and finalize the gathering. According to Lemma 4.3.1, since the Weber meeting node remains invariant while all the robots move towards it, $C(t) \notin \mathcal{I}_4^{b3}$. Consider the case when there does not exist any Weber meeting node on c. The following subcases may arise.

Subcase 1. $C(0)$ satisfy C_1. All the non-guard robots from the other quadrants as well on l or l' move towards the target Weber meeting node in $\mathcal{H}^{++}$. So, at any arbitrary instant of time $t > 0$, $C(t)$ remains asymmetric and hence $C(t) \notin \mathcal{I}_4^{b3}$.

Subcase 2. $C(0)$ satisfy C_2. If the configuration is unbalanced, all the robots in the quadrants different from $\mathcal{H}^{++}$ as well as the robots on l or l' move towards the target Weber meeting node in $\mathcal{H}^{++}$ in the Creating Multiplicity on Target Weber meeting node phase. While such a robot reaches $\mathcal{H}^{++}$, the configuration remains unbalanced and

hence asymmetric. If the configuration is balanced, a leading robot is selected in the Leading Robot Selection phase. According to Lemma 4.6.2, the configuration remains asymmetric during the movement of the leading robot towards the target Weber meeting node at some time $t' > 0$. While the leading robot reaches $\mathcal{H}^{++}$, the configuration becomes unbalanced and remains asymmetric. So, at any arbitrary instant of time $t > 0$, $C(t) \notin \mathcal{I}_4^{b3}$, where t denotes an instant of time such that $t \geq t'$.

Subcase 3. $C(0)$ satisfy C_3. All the robots in $\mathcal{H}^{++}$, move towards the target Weber meeting node. After all the robots in $\mathcal{H}^{++}$, reach the target Weber meeting node m, all the non-guard robots from the other quadrants as well as on l or l' move towards m, thus creating a multiplicity on m. During this robot movement, $C(t)$ remains asymmetric and hence $C(t) \notin \mathcal{I}_4^{b3}$.

Case 2. $C(0) \in \mathcal{I}_4^{b2}$. Assume that at time $t' > 0$, the robot on c moves towards one of the adjacent nodes, which transforms the configuration into a configuration that may be asymmetric or admits a single line of symmetry. Proceeding similarly, as in the case of $C(0) \in \mathcal{I}_3^a \cup \mathcal{I}_4^a$, at any arbitrary instant of time $t > 0$, $C(t)$ remains asymmetric and hence $C(t) \notin \mathcal{I}_4^{b3}$, where $t \geq t'$. □

Theorem 4.6.8. *If the initial configuration belongs to the set $\mathcal{I} \setminus U$, then the algorithm Optimal Gathering() ensures gathering over Weber meeting nodes.*

Proof. Assume that $C(0) \in \mathcal{I} \setminus \mathcal{U}$. If $C(t)$ is not a final configuration for some $t \geq 0$, each active robot executes the algorithm Optimal Gathering(). According to the Lemmas 4.6.6 and 4.6.7, any initial configuration $C(0) \in \mathcal{I} \setminus \mathcal{U}$, would never reach a configuration $C(t) \in \mathcal{U}$, during the execution of the algorithm Optimal Gathering() at any point of time $t > 0$. The following cases are to be considered.

Case 1. There exists a unique Weber meeting node. All the robots move towards the unique Weber meeting node and finalize the gathering.

Case 2. There exists more than one Weber meeting node. The target Weber meeting node is selected in the Target Weber meeting node Selection phase. According to the Lemmas 4.6.3, 4.6.4 and 4.6.5, the target Weber meeting node remains invariant during the execution of the algorithm Optimal Gathering(). If $C(0)$ is a balanced configuration, then a leading robot is selected in the Leading Robot Selection phase. Lemmas 4.6.1 and 4.6.2 ensure that the leading robot remains invariant during its movement.

The movement of the leading robot ensures that the configuration transforms into an unbalanced configuration.

Without loss of generality, assume that m is the target Weber meeting node. Assume that, at any point of time t, there exists at which at least one robot r has completed its LCM cycle. If r is a non-guard robot, then it must have moved at least one unit distance towards m at time $t' > t$. Since each non-guard robot moves towards m via a shortest path in the Creating Multiplicity on Target Weber meeting node phase, this implies that eventually at time $t'' > t'$, there exists a robot multiplicity on m. Finally, in the Finalization of Gathering phase, since the robots have global strong-multiplicity detection capability, all the guard robots move towards m and finalize the gathering without creating any other multiplicity on a meeting node. Since each robot finalizes the gathering, by moving towards m via a shortest path, gathering over Weber meeting nodes is ensured. □

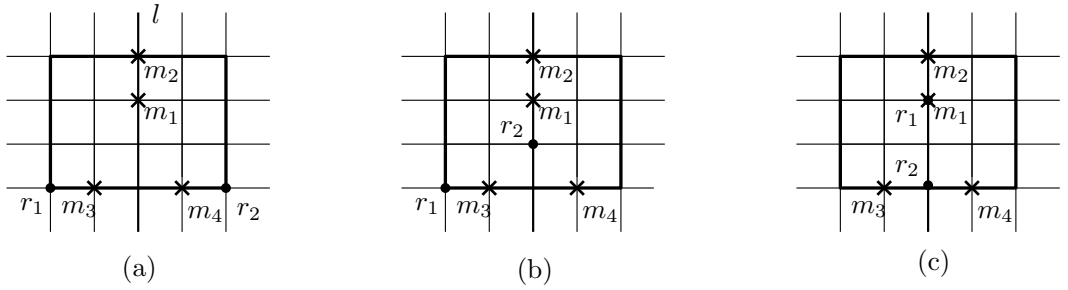

FIGURE 4.12: (a) $C(0)$, (b) $C(t_1)$, (c) $C(t_2)$

4.7 Optimal Gathering for $C(t) \in \mathcal{U}$

We have proposed a deterministic distributed algorithm that ensures gathering over a Weber meeting node for any initial configuration $C(0) \in \mathcal{I} \setminus \mathcal{U}$. Let $\mathcal{U}' \subset \mathcal{U}$ denote the set of all the initial configurations which admit a unique line of symmetry l and no Weber meeting nodes or robot positions exist on l. However, there exists at least one meeting node on l. The set $\mathcal{U}'$ includes the initial configurations for which gathering is feasible on a meeting node. Note that, if $C(t) \in \mathcal{U} \setminus \mathcal{U}'$, then it is ungatherable. To ensure gathering deterministically, the target point must lie on l. At this point of time, one optimal feasible solution for a configuration $C(0) \in \mathcal{U}'$ would be to finalize the gathering at a meeting node $m \in l$ at which the total number of moves is minimized.

Ties may be broken by considering the northernmost such meeting node. Another very important assumption that is not highlighted much in the literature is that initially, all the robots are static. The correctness of our proposed algorithm fails to hold when the optimal target point is dynamically selected. As a consequence, termination may not be guaranteed with an optimal number of moves. For example, we consider one possible execution for an initial configuration $C(0) = (\{r_1, r_2\}, \{m_1, m_2, m_3, m_4\})$ in Figure 4.12(a). At $t = 0$, m_3 and m_4 are the Weber meeting nodes. Between m_1 and m_2, the number of total moves will be minimized if the robots gather at m_1. While r_1 and r_2 start moving towards m_1, there may be a pending move due to the asynchronous behavior of the scheduler. Consider the case when r_2 has completed its LCM cycle while r_1's move is pending. At $t = t_1 > 0$, m_3 becomes the unique Weber meeting node (Figure 4.12(b)). At $t_2 > t_1$, assume that r_1 has reached m_1 and r_2 has moved by one hop distance towards m_3. At t_2, m_1 becomes the unique Weber meeting node (Figure 4.12(c)). Next, the gathering will be finalized eventually at m_1. Initially, the minimum number of moves required to finalize the gathering is 8 (Figure 4.12(a)). The number of moves required to finalize the gathering in this execution is 10. It is not guaranteed that the minimum number of moves required to finalize the gathering in the initial configuration is achievable.

4.8 Conclusion

In this chapter, the optimal gathering over Weber meeting nodes problem has been investigated over an infinite grid. The objective function is to minimize the total distance traveled by all the robots. We have characterized all the configurations for which gathering over a Weber meeting node cannot be ensured. For the remaining configurations, a deterministic distributed algorithm has been proposed that solves the gathering over Weber meeting nodes for at least seven robots.

Chapter 5

Gathering over Heterogeneous Meeting Nodes

5.1 Overview of the Problem

This chapter considers the *gathering over heterogeneous meeting nodes problem* in an infinite grid. In this problem, two finite and disjoint sets of homogeneous robots placed at the nodes of an infinite grid graph are considered. Additionally, the grid graph consists of two finite and disjoint sets of prefixed meeting nodes that are located at the nodes of the grid. The goal of the study is to propose a distributed algorithm that requires all robots in the first team to gather at one of the meeting nodes of the first type and all robots in the second team to gather at one of the meeting nodes of the second type. The robots can distinguish between the two different kinds of meeting nodes. A robot cannot identify the members of its team. However, it is assumed that a robot is capable of identifying its own team. We next discuss the motivation behind studying this problem.

In Chapter 3, we have studied the gathering over meeting nodes problem in an infinite two-dimensional grid. It may be possible that in many real-life applications, different tasks are required to be performed simultaneously by different groups of robots. We consider the following example: We have two different kinds of robots, three that clean up dust (like the iRobot Roomba) and two that mop (like the iRobot Brava). Cleaning two levels of a certain house is one of the two tasks. One or two dust cleaning robots, coupled

with a single mopping robot, are needed for the tasks, depending on how much dust has gathered at each level [61]. Bhagat et al.[13] considered two groups of homogeneous robots having two different objectives. One group of robots needs to solve the gathering problem while the other group of robots solves the Circle Formation problem. More precisely, the problem considers two teams of anonymous robots $\mathcal{R}_g$ and $\mathcal{R}_f$, where the goal of the robots in $\mathcal{R}_g$ is to achieve gathering, while the goal of the robots in $\mathcal{R}_f$ is to solve the Circle Formation problem. The robots are located at distinct points in the same space and they are unaware of which of the robots they observe is a member of their own team. A distributed algorithm was proposed that allows each team to solve its problem asynchronously and with non-rigid movements. In this chapter, we have considered two different types of homogeneous robots with the objective that they gather at two different meeting nodes in an infinite grid.

The fundamental challenge to solve the problem arises because of the anonymity and asynchrony of the robots. In [31], the gathering on meeting points problem has been considered. This problem requires that each robot must gather at one of the predetermined fixed points. The problems in [31, 13] were considered in the continuous domain. The robots can move in any arbitrary direction in the continuous domain and even by infinitesimal distance. This motivates us to consider the gathering over heterogeneous meeting nodes problem in the discrete model, where the movement of a robot is restricted along the edges of the graph and to an adjacent node in one step. In this chapter, the proposed algorithm ensures that no collision among the robots occurs at any point of time if they are located within the minimum enclosing rectangle of the meeting nodes. However, collisions exist at two different nodes, which are at a sufficiently large distance from the minimum enclosing rectangle of the meeting nodes. It is important to note that collision-free paths are not created only to prevent robot collisions. The main reason for avoiding collision is to avoid forming unstable multiplicities, which include robots from different teams. So, the main challenge arises in proposing collision-less algorithms when the robots are deployed at the nodes of an infinite grid. It should be noted that since the robots are oblivious, one of the main challenges in designing algorithms is to maintain the invariance of the targets where each robot decides to finalize the gathering. We have shown later that the target may change if there exists multiplicity at some nodes, where the multiplicity corresponds to robots of different types.

5.2 Contribution

This chapter proposes a deterministic distributed algorithm that solves the gathering over heterogeneous meeting nodes problem within a finite time. The initial configurations of the robots and the meeting nodes for which the gathering problem is unsolvable have been characterized, and a distributed gathering algorithm has been proposed for the remaining initial configurations. The proposed algorithm runs in $\Theta(dn)$ moves, where d is the diameter of the minimum enclosing rectangle of all the robots and meeting nodes in the initial configuration and n is the total number of robots in the system. We measure the time complexity of our algorithms in epochs. The proposed algorithm runs in $O(dn)$ epochs. It has been proved that the lower bound for any algorithm that solves the problem requires $\Omega(Dn)$ moves and $\Omega(D)$ epochs.

Organization: The next section describes the formal description of the model. Some basic definitions and notations are presented in this section. In this section, a partitioning of the initial configurations has also been stated. Section 5.4 is dedicated to the formal description of the algorithm along with the correctness proofs. The efficiency of the algorithm has also been studied in this section, where the efficiency is measured with respect to the number of moves and time in epochs. The chapter is concluded with some potential future directions in Section 5.5.

5.3 Gathering over Heterogeneous Meeting Nodes Problem

In this section, we first provide the basic definitions and notation necessary for the understanding of the proposed results.

- Recall from Chapters 3 and 4 that $\mathcal{R}(t)$ denotes the set of all such distinct nodes occupied by the robots in $\mathcal{R}$ at time t, i.e., $\mathcal{R}(t) \subset V$ such that each node in $\mathcal{R}(t)$ contains at least one robot on it at time t.
- $\mathcal{R}$ consists of two teams of homogeneous and disjoint sets of robots, namely $\mathcal{R}_1$ and $\mathcal{R}_2$ that are deployed at the nodes of an infinite grid graph G.

- $\mathcal{M} \subset V$ denotes the finite set of meeting nodes. The nodes in $\mathcal{M}$ are partitioned into two subsets $\mathcal{M}_1$ and $\mathcal{M}_2$, such that $\mathcal{M} = \mathcal{M}_1 \cup \mathcal{M}_2$ and $\mathcal{M}_1 \cap \mathcal{M}_2 = \phi$. The robots can distinguish between the two different types of meeting nodes.

In this chapter, we propose a gathering algorithm for solving the problem of gathering over heterogeneous meeting nodes problem. In this context, a gathering algorithm is a deterministic distributed algorithm that gathers all the robots in $\mathcal{R}_1$ (resp. $\mathcal{R}_2$) at a meeting node in $\mathcal{M}_1$ (resp. $\mathcal{M}_2$). If no deterministic algorithm can be proposed to ensure the robot's gathering at meeting nodes, we say an initial configuration is *ungatherable*. In addition, a gathering algorithm ensures that, regardless of the adversary, the robots will gather at the meeting nodes. Next, we introduce the notion of symmetry of a configuration and view of a configuration. These definitions are relevant in understanding the concepts of symmetric configurations and the symmetric configurations which are ungatherable, i.e., no gathering algorithm can ensure the gathering of such configurations.

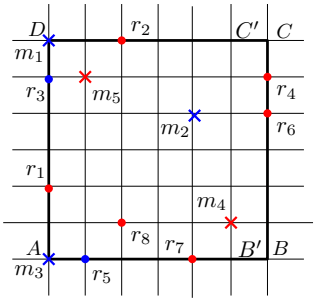

FIGURE 5.1: The crosses represent meeting nodes and the circles represent robots. The two teams of robots and meeting nodes are denoted by blue and red colors, respectively.

5.3.1 Symmetry of a configuration and Configuration View

Before proceeding to the definition of symmetry, recall from Chapter 3 that we have defined a function μ_t. In this chapter, since the set of meeting nodes is partitioned into $\mathcal{M}_1$ and $\mathcal{M}_2$, the definition of μ_t is slightly different from the definition mentioned in Chapter 3. Hence, we define the function $\mu_t : V \to \{0, 1, 2, 3, 4, 5, 6, 7, 8\}$ at time t in

the following way.

$$\mu_t(v) = \begin{cases} 0 & \text{if } v \notin \mathcal{M} \cup \mathcal{R}(t) \text{ i.e., } v \text{ is an empty node} \\ 1 & \text{if } v \in \mathcal{M}_1 \setminus \mathcal{R}(t) \text{ i.e., } v \text{ is a meeting node in } \mathcal{M}_1 \text{ not containing any robot} \\ 2 & \text{if } v \in \mathcal{M}_2 \setminus \mathcal{R}(t) \text{ i.e., } v \text{ is a meeting node in } \mathcal{M}_2 \text{ not containing any robot} \\ 3 & \text{if } v \in \mathcal{M}_1 \cap \mathcal{R}(t) \text{ s.t. } v \text{ is a meeting node in } \mathcal{M}_1 \text{ and contains exactly one robot} \\ 4 & \text{if } v \in \mathcal{M}_2 \cap \mathcal{R}(t) \text{ s.t. } v \text{ is a meeting node in } \mathcal{M}_2 \text{ and contains exactly one robot} \\ 5 & \text{if } v \in \mathcal{M}_1 \cap \mathcal{R}(t) \text{ s.t. } v \text{ is a meeting node in } \mathcal{M}_1 \text{ and contains multiple robots} \\ 6 & \text{if } v \in \mathcal{M}_2 \cap \mathcal{R}(t) \text{ s.t. } v \text{ is a meeting node in } \mathcal{M}_2 \text{ and contains multiple robots} \\ 7 & \text{if } v \in \mathcal{R}(t) \setminus \mathcal{M} \text{ s.t. } v \text{ contains exactly one robot} \\ 8 & \text{if } v \in \mathcal{R}(t) \setminus \mathcal{M} \text{ s.t. } v \text{ contains multiple robots} \end{cases}$$

Without any ambiguity, we define μ_t by μ.

- Let $C(t) = (\mathcal{R}(t),\mathcal{M})$ denote the system configuration at time t.
- The strings $s_i's$ are defined similarly as in Chapter 3. The only difference lies in defining the function μ. In Figure 5.1, DA is the string direction associated to the unique key corner D. Here, s_{DA}=17007010200007700007000000000010007000002007700000.
- **Symmetricity of the set $\mathcal{M}$:** In Chapter 3, we have defined MER_F as the smallest grid-aligned rectangle containing all the meeting nodes. Since, the set of meeting nodes $\mathcal{M}$ is partitioned into disjoint subsets $\mathcal{M}_1$ and $\mathcal{M}_2$ the modified definition of the function $f_t : V \to \{0, 1, 2\}$ as follows:

$$f_t(v) = \begin{cases} 0 & \text{if } v \text{ is not a meeting node} \\ 1 & \text{if } v \text{ is a meeting node in } \mathcal{M}_1 \\ 2 & \text{if } v \text{ is a meeting node in } \mathcal{M}_2 \end{cases}$$

The strings $\alpha_i's$ are defined similarly as in Chapter 3. Recall that the corner with which the largest lexicographic string α_i is associated is defined as the leading corner. In Figure 5.1, $AB'C'D$ is the MER_F and D is the unique leading corner. Here

$\alpha_{DC'}$=10000002000000001000000000000000002100000. The configuration view of a node is defined similarly as in Chapter 3.

5.3.2 Problem Definition and Ungatherability Results

This subsection states the formal definition of the problem and a theorem that provides a necessary condition for a configuration to be ungatherable. First, we describe the formal definition of the problem.

A set $\mathcal{R}$ consisting of $n \geq 5$ robots is deployed at the nodes of an infinite grid graph. The robots deployed are of two different types. Let $\mathcal{R}_1$ and $\mathcal{R}_2$ denote the two homogeneous and disjoint teams of robots that are deployed at the nodes of the infinite grid graph G given as input. $\mathcal{M}$ is a finite set of meeting nodes located at the nodes of the graph. The set $\mathcal{M}$ of meeting nodes consists of two disjoint sets of meeting nodes $\mathcal{M}_1$ and $\mathcal{M}_2$. The robots in each team can distinguish between the meeting nodes. However, a robot cannot identify its team members. The goal of gathering over heterogeneous meeting nodes problem is to reach a final configuration ($\mathcal{F}$), where all robots in $\mathcal{R}_1$ (resp. $\mathcal{R}_2$) gather at a meeting node in $\mathcal{M}_1$ (resp. $\mathcal{M}_2$) and the gathering must terminate within a finite amount of time.

If the initial configuration $C(0)$ is partitive on the node set $V \setminus V'$, then Theorem 3.4.1 ensures that there must exist at least one meeting node $m \in V'$ where gathering over a meeting node is ensured. In this chapter, note that the robots are unaware of the team to which the other robots belong. It may be possible that a robot in $\mathcal{R}_1$ is symmetric either with respect to a single line of symmetry or rotational symmetry, and its symmetric image is a robot belonging to $\mathcal{R}_2$. In that case, an algorithm may be proposed under the execution of which the symmetry can be broken. In other words, a gathering algorithm may exist where the symmetric robots have different execution paths. This ensures that the execution of the algorithm can break the symmetry. In order to characterize the ungatherable configurations, we first define the symmetricity of the set $\mathcal{R}$. We first define a function $h_t : V \to \{0, 1\}$ at any fixed time $t \geq 0$ by:

$$h_t(v) = \begin{cases} 0 & \text{if } v \text{ is a free node} \\ 1 & \text{if } v \text{ contains a robot position} \end{cases}$$

i.e., $h_t(v)$ is defined as an indicator variable on the set of nodes V, which equals 1 when the node contains a robot's position at time t. Without any ambiguity, we denote h_t by h.

While considering the corners of MER, we can define a string c_i similar to s_i. The only difference is that we associate $h(v)$ to each node v instead of $\mu(v)$. The strings c_i for each corner are defined similarly. Let us assume that the robots have the capability to detect the team to which the other robot belongs. In that case, we consider the following definition.

Definition 5.3.1. *Let $C(t) = (\mathcal{R}(t),\mathcal{M})$ denote the system configuration at time t. We say that if there exists a unique lexicographic largest string c_i, then the set $\mathcal{R}$ is asymmetric. Otherwise, the set $\mathcal{R}$ is said to be symmetric (Figure 5.2 (a)).*

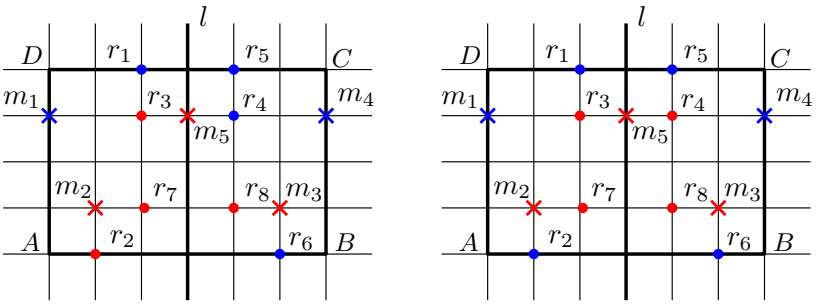

FIGURE 5.2: (a) $\mathcal{R}$ is symmetric, but $\mathcal{R}_1$ and $\mathcal{R}_2$ are not independently symmetric. (b) $\mathcal{R}$ is symmetric. $\mathcal{R}_1$ and $\mathcal{R}_2$ are independently symmetric.

We define another function color: $V \to \{0, 1, 2\}$ by:

$$color(v) = \begin{cases} 0 & \text{if } v \text{ is not a robot position} \\ 1 & \text{if } v \text{ is a robot in } \mathcal{R}_1 \\ 2 & \text{if } v \text{ is a robot in } \mathcal{R}_2 \end{cases}$$

Considering the corners of MER, we can similarly associate the pair $(h_t(v), color(v))$ to each node v and define the strings c_i.

Definition 5.3.2. *Let $C(t) = (\mathcal{R}(t),\mathcal{M})$ denote the system configuration at time t. In the definition of the symmetricity of $\mathcal{R}$, where the strings c_i are defined according to*

the color function, $\mathcal{R}$ is symmetric if and only if $\mathcal{R}_1$ and $\mathcal{R}_2$ are both independently symmetric (Figure 5.2(b)).

The next corollaries of the theorem provide a characterization of the configurations that are ungatherable in our setting and even in a stronger model than that of the model assumed in the chapter.

Corollary 5.3.3. *Assume the initial configuration $C(0)$ is such that it is symmetric with respect to l and $\mathcal{R}_1$ and $\mathcal{R}_2$ are independently symmetric. Furthermore, there do not exist meeting nodes of both types on l and $\mathcal{R} \cap l = \phi$. Then, the initial configuration is ungatherable.*

Proof. The proof of the corollary is a direct outcome of Theorem 3.4.1. Let the initial configuration $C(0)$ be the same as in the statement of Corollary 5.3.3 and $\mathcal{A}$ be a deterministic distributed algorithm that solves the gathering over heterogeneous meeting nodes problem. In other words, starting from any initial configuration $C(0)$, there exists an algorithm $\mathcal{A}$ which allows $C(0)$ to reach a configuration $\mathcal{F}$ in which the robots in $\mathcal{R}_1$ (resp. $\mathcal{R}_2$) gathers at one of the meeting nodes in $\mathcal{M}_1$ (resp. $\mathcal{M}_2$). Consider the scheduler to be fully-synchronous. According to the problem definition, since no robots exist on l, the gathering must occur at the meeting nodes on l. There exists at least one execution path of $\mathcal{A}$, in which a move of any robot r would result in the symmetric image of r performing the same move. As there does not exist any robot position on l, it is impossible to break the symmetry. Hence, in order to ensure gathering over a meeting node, there must exist at least one meeting node belonging to $\mathcal{M}_1$ and at least one meeting node belonging to $\mathcal{M}_2$ on l. Thus, the initial configuration is ungatherable. □

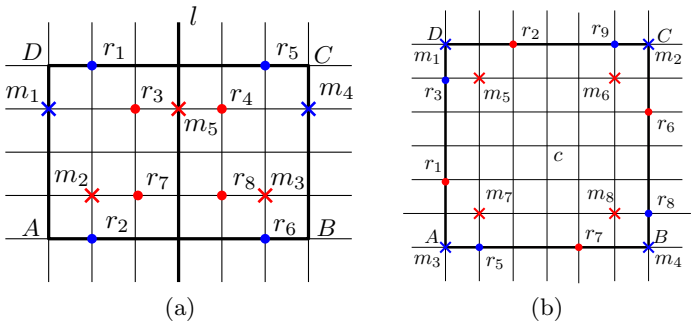

FIGURE 5.3: (a) Ungatherable configuration admitting a single line of symmetry. (b) Ungatherable configuration admitting rotational symmetry.

Corollary 5.3.4. *Assume the initial configuration $C(0)$ is such that it is symmetric with respect to rotational symmetry and $\mathcal{R}_1$ and $\mathcal{R}_2$ are independently symmetric. Furthermore, there does not exist a robot on c, i.e., $\mathcal{R} \cap \{c\} = \phi$. Then, the initial configuration is ungatherable.*

Proof. Let the initial configuration $C(0)$ be the same as in the statement of Corollary 5.3.4 and $\mathcal{A}$ be a deterministic distributed algorithm that solves the gathering over heterogeneous meeting nodes problem. In other words, starting from the initial configuration $C(0)$, there exists an algorithm $\mathcal{A}$ which allows $C(0)$ to reach a configuration $\mathcal{F}$ in which the robots in $\mathcal{R}_1$ (resp. $\mathcal{R}_2$) gathers at one of the meeting nodes in $\mathcal{M}_1$ (resp. $\mathcal{M}_2$). As there does not exist a robot position on c, a direct consequence of Theorem 3.4.1 implies the robots can gather only at c. Since the gathering must be ensured at two different meeting nodes and there may exist at most one meeting node on c, the gathering is impossible. □

We assumed that the scheduler to be fully-synchronous for the purposes of the proofs. The impossibility result holds true even if the robots are activated under an asynchronous scheduler, as the assumption of fully-synchronous scheduler is stronger than that of asynchronous scheduler. We denote the set of all ungatherable configurations $\mathcal{U}$ according to the configurations listed in the corollaries 5.3.3 and 5.3.4 (Figures 5.3 (a) and 5.3 (b)).

5.3.3 Partitioning of the Initial configuration

In this subsection, we provide a partitioning of the initial configurations. Let $\mathcal{I}$ denote the set of all initial configurations. The algorithm divides all the initial configurations which do not belong to the set $\mathcal{U}$ that it processes into the following disjoint classes, along with configurations produced during execution.

The algorithm executes according to the class of configurations each robot perceives in its local configuration view. All the initial configurations belonging to the set $\mathcal{I} \setminus \mathcal{U}$ can be partitioned into the following disjoint classes.

1. $\mathcal{I}_1$: $\mathcal{M}$ is asymmetric (Figure 5.4(a)).

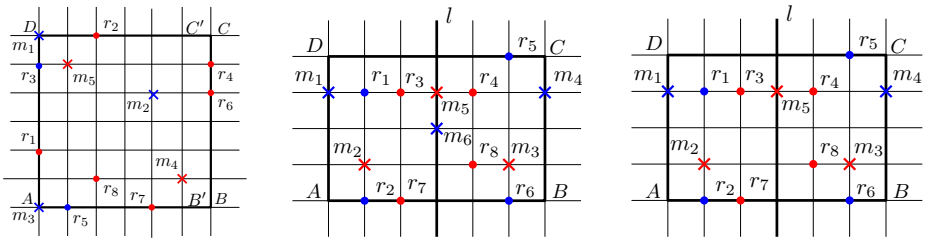

FIGURE 5.4: (a) $\mathcal{I}_1$-configuration. (b) $\mathcal{I}_2$-configuration. l contains meeting nodes $m_6 \in \mathcal{M}_1$ and $m_5 \in \mathcal{M}_2$. (c) $\mathcal{I}_3$-configuration. l does not contain any meeting node belonging to $\mathcal{M}_1$.

2. $\mathcal{I}_2$: $\mathcal{M}$ is symmetric with respect to a unique line of symmetry l and $\mathcal{C}(t)$ is asymmetric. There exist meeting nodes of both types on l (Figure 5.4(b)).

3. $\mathcal{I}_3$: $\mathcal{M}$ is symmetric with respect to a unique line of symmetry l or rotational symmetry, and $\mathcal{C}(t)$ is asymmetric. In case $\mathcal{M}$ is symmetric with respect to l, there do not exist meeting nodes of both types on l (Figure 5.4(c)).

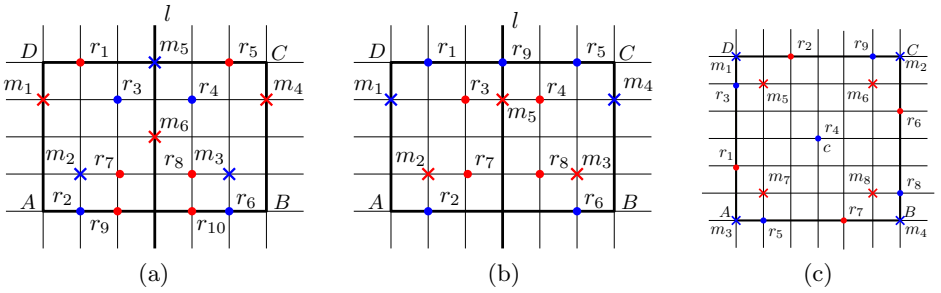

FIGURE 5.5: (a) $\mathcal{I}_{41}$-configuration with a meeting node m_5 belonging to $\mathcal{M}_1$ and a meeting node m_6 belonging to $\mathcal{M}_2$ on l. (b) $\mathcal{I}_{42}$ configuration with a robot r_9 on l. (c) $\mathcal{I}_5$-configuration with a robot position r_4 on c.

4. $\mathcal{I}_4$: $\mathcal{C}(t)$ is symmetric with respect to a unique line of symmetry l. This can be further partitioned into:

   (a) $\mathcal{I}_{41}$: There exists meeting nodes $m_1 \in \mathcal{M}$ and $m_2 \in \mathcal{M}_2$ on l (Figure 5.5(a)).

   (b) $\mathcal{I}_{42}$: There does not exist meeting nodes of both types on l, but there exists a robot position r on l (Figure 5.5 (b)).

5. $\mathcal{I}_5$: $\mathcal{C}(t)$ is symmetric with respect to rotational symmetry, with c being the center of rotation, and there exists a robot position on c (Figure 5.5(c)).

It can be checked that the set $\{\mathcal{I}_1, \mathcal{I}_2, \mathcal{I}_3, \mathcal{I}_4, \mathcal{I}_5\}$ forms a partition of the set $\mathcal{I} \setminus \mathcal{U}$.

5.4 Algorithm

This section describes our main algorithm 2-*nodeGathering*(). The algorithm solves the gathering over heterogeneous meeting nodes problem for all the initial configurations belonging to the set $\mathcal{I} \setminus \mathcal{U}$, where the initial configuration consists of at least five robots. In particular, in subsection 5.4.1, we describe the overview of the strategy underlying the algorithm. Before proceeding to the overview of the proposed algorithm, we define *stable* and *unstable* multiplicity nodes.

Definition 5.4.1. *A multiplicity node is said to be stable if the robots at the multiplicity belong to the same team of robots. Otherwise, a multiplicity node is said to be unstable.*

Note that a robot cannot distinguish between a stable and unstable multiplicity node. The main crux of the algorithm is to avoid unstable multiplicities throughout the execution. We will see later in subsection 5.3, that the avoidance of unstable multiplicities is necessary to correctly determine whether the proposed algorithm successfully accomplished the gathering process.

5.4.1 Overview

In the following subsection, we provide a general idea of our algorithm that executes according to the partitioning of the initial configurations. The following cases are to be considered.

Case 1. Consider the case when the initial configuration $C(0) \in \mathcal{I}_1 \cup \mathcal{I}_2$. First, assume the case when the meeting nodes are asymmetric, i.e., $C(0) \in \mathcal{I}_1$. According to Observation 1 in Chapter 3, let $\mathcal{O}_1$ be the ordering of the meeting nodes defined according to the order in which they appear in the string α_i. Let $\mathcal{O}_1 = (m_1, m_2, \ldots m_s)$ denote this particular ordering of the meeting nodes. The ordering of the meeting nodes remains invariant while the robots move toward the meeting nodes. As a result, the robots can select the meeting nodes according to the ordering and finalize the gathering at those meeting nodes.

Next, assume the case when the meeting nodes are symmetric with respect to a unique line of symmetry l and there exists at least one meeting node of both types on l. If l is a horizontal or vertical line of symmetry, then there exist two leading corners. According

to Observation 2 of Chapter 3, let $\mathcal{O}_2 = (m'_1, m'_2, \ldots, m'_z)$ be this particular ordering of the meeting nodes on l. If l is a diagonal line of symmetry and there exists a unique leading corner, then the ordering of the meeting nodes on l is defined with respect to the distance from the leading corner. In this case, the targets for gathering are selected with respect to the ordering $\mathcal{O}_2$. In the next subsection, the targets for gathering are formally described.

Case 2. Consider the case when $C(0) \in \mathcal{I}_3$. In this case, the meeting nodes are either symmetric with respect to a single line of symmetry l or rotational symmetry. If the meeting nodes are symmetric with respect to l, then there do not exist meeting nodes of both types on l. In this case, the procedure *GatheringAsym*() is executed. The procedure for solving the gathering in such configurations is divided into various phases. The main crux of the procedure is to maintain the asymmetry of the initial configuration. The procedure mainly consists of the following phases:

- In the *Guard Selection and Placement* (GS) phase, a robot is selected as a *guard* and placed. The guard is selected and placed in such a way that it remains invariant during the execution of the algorithm. This phase is mostly similar to Guard Selection and Placement phase described in Chapter 3. However, the destination of the guard in this case is different from the destination mentioned in Chapter 3.
- The potential target meeting nodes for gathering are selected in the *Target Meeting Nodes Selection* (TMS) phase.
- In the *Pivot Selection* (PS) phase, a set of robots is selected from the non-guard as *pivots*.
- In the *Move to Destination* (MD) phase, each non-pivot robot moves towards a destination node, which is at a sufficiently large distance from MER_F. This movement is done in such a way that no multiplicity nodes are created at the target meeting nodes during the movement.
- In the *Creating Multiplicity on the Target Meeting Node* phase, stable multiplicities are created at the target meeting nodes.
- The gathering is finalized at the target meeting nodes in the *Finalisation of Gathering* (FG) phase.

Predicates and their Definitions	
Predicates $C(0)$	Definitions
a_1	There exists meeting nodes of both types on the unique line of symmetry l
a_2	There exists at least one robot position on the unique line of symmetry l
a_3	There exists at least one robot position on the center of rotational symmetry c
a_4	The configuration is neither a nearly rotational configuration nor a nearly reflective configuration

TABLE 5.1: Predicates and their definitions

Case 3. Consider the case when $C(0) \in \mathcal{I}_{42} \cup \mathcal{I}_5$. In this class of configurations, a procedure $AllowtoMove()$ is executed to transform the symmetric configurations into asymmetric configurations. The rest of the procedure follows similarly from the previous case.

Case 4. Consider the case when $C(0) \in \mathcal{I}_{41}$. In this class of symmetric configurations, the gathering is finalized at the target meeting nodes on the unique line of symmetry.

Consider a symmetric configuration with no robot positions or meeting nodes of both types on l. Consider the case when the configuration is symmetric without any robot position or meeting nodes of both types on l. Further, assume that $\mathcal{R}_1$ and $\mathcal{R}_2$ are not independently symmetric. In that case, each robot starts moving towards the meeting nodes, which appear last in the string direction associated to the leading corners. Since the robots are not independently asymmetric, while the robots move, there exists an instant of time in which the configuration transforms into an asymmetric configuration. The rest of the procedure proceeds similarly as in the case when the configuration is asymmetric.

5.4.2 Formalization of the Algorithm 2-$nodeGathering()$

In this subsection, we provide a formal description of the algorithm that solves the gathering over heterogeneous meeting nodes problem. First, we define some predicates that are relevant to understanding the formal description of the algorithm (Table 5.1). Next, we give a formal description of the algorithm in the form of a table (Table 5.2), according to the predicates mentioned in Table 5.1.

Formalization of the Algorithm 2-*nodeGathering*()	
Initial Configuration $C(0)$	**Procedure**
$\mathcal{I}_1$	Each robot moves towards their respective targets, selected according to the ordering $\mathcal{O}_1$ of the meeting nodes
$\mathcal{I}_2$	Each robot moves towards their respective targets, selected according to the ordering $\mathcal{O}_2$ of the meeting nodes
$\mathcal{I}_3 \wedge \neg a_4$	GatheringAsym()
$\mathcal{I}_3 \wedge a_4$	AllowtoMove()
$\mathcal{I}_4 \wedge a_1$	The gathering is ensured at the meeting nodes on the unique line of symmetry
$\mathcal{I}_4 \wedge \neg a_1 \wedge a_2$	AllowtoMove()
$\mathcal{I}_5 \wedge a_3$	AllowtoMove()

TABLE 5.2: Formalization of the Algorithm 2-*nodeGathering*()

In the following subsections, the algorithm is formally described according to the class of configurations each robot perceives in its local configuration view. The pseudo-code

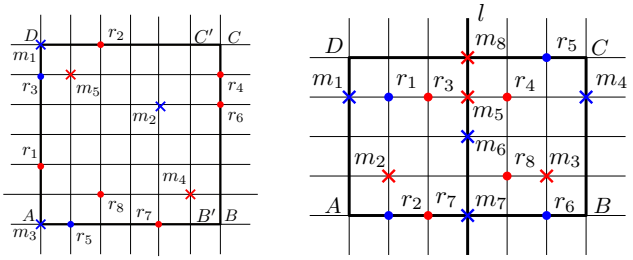

FIGURE 5.6: Example of the configurations illustrating the statements of Lemmas 5.4.2 and 5.4.3

description of the Algorithm 2-*nodeGathering*() is given in Algorithm 5.1.

5.4.2.1 $\mathcal{I}_1$

If $C(t) \in \mathcal{I}_1$, the meeting nodes are asymmetric. According to Observation 1, each robot fixes a specific ordering $\mathcal{O}_1$ of the meeting nodes. The meeting node $m_1 \in \mathcal{M}_1$ having the highest order in $\mathcal{O}_1$ among all the meeting nodes in $\mathcal{M}_1$ is selected as one of the target meeting nodes. Similarly, the meeting node $m_2 \in \mathcal{M}_2$ having the highest order in $\mathcal{O}_1$ among all the meeting nodes in $\mathcal{M}_2$ is selected as the other target meeting node. Each robot in $\mathcal{R}_1$ moves towards m_1 and finalizes the gathering at m_1. Similarly, each robot in $\mathcal{R}_2$ moves towards m_2 and finalizes the gathering at m_2. Since the ordering $\mathcal{O}_1$ depends on the position of fixed meeting nodes, the targets remain invariant while the

Algorithm 5.1: 2-*nodeGathering*()

Input: Configuration $C(t) = (\mathcal{R}(t),\mathcal{M}) \in \mathcal{I} \setminus \mathcal{U}$

if $C(t) \in \mathcal{I}_1$ **then**
  Each robot fixes a specific ordering $\mathcal{O}_1$ of the meeting nodes;
  Each robot in $\mathcal{R}_1$ (resp. $\mathcal{R}_2$) moves towards the meeting node $m_1 \in \mathcal{M}_1$ (resp. $m_2 \in \mathcal{M}_2$) having the highest order in $\mathcal{O}_1$;
else if $C(t) \in \mathcal{I}_2$ **then**
  Each robot fixes a specific ordering $\mathcal{O}_2$ of the meeting nodes on l;
  Each robot in $\mathcal{R}_1$ (resp. $\mathcal{R}_2$) moves towards the meeting node $m'_1 \in \mathcal{M}_1$ (resp. $m'_2 \in \mathcal{M}_2$) on l having the highest order in $\mathcal{O}_2$;
else if $C(t) \in \mathcal{I}_3$ *and* $C(t)$ *is neither nearly reflective nor nearly rotational configuration* **then**
  GatheringAsym();
else if $C(t) \in \mathcal{I}_3$ *and* $C(t)$ *is either a nearly reflective or a nearly rotational configuration* **then**
  AllowtoMove();
  GatheringAsym();
else if $C(t) \in \mathcal{I}_4$ **then**
  if $C(t) \in \mathcal{I}_{41}$ **then**
    GatheringSym() ;
  else if $C(t) \in \mathcal{I}_{42}$ **then**
    AllowtoMove() ;
    GatheringAsym() ;
else if $C(t) \in \mathcal{I}_5$ **then**
  AllowtoMove() ;
  GatheringAsym() ;

robot moves towards their respective targets. In Figure 5.6 (a), D is the leading corner. The ordering $\mathcal{O}_1$ is defined as $\mathcal{O}_1 = (m_1, m_5, m_2, m_4, m_3)$. m_3 and m_4 are selected as the target meeting nodes. So, we have the following lemma.

Lemma 5.4.2. *If* $C(0) \in \mathcal{I}_1$*, then the target meeting nodes remain invariant during the execution of the algorithm* 2-*nodeGathering*() *at any time* $t > 0$.

Proof. The proof follows from Lemma 3.5.1. □

5.4.2.2 $\mathcal{I}_2$

If $C(t) \in \mathcal{I}_2$, the meeting nodes are symmetric with respect to a single line of symmetry l. In addition to that, there exists at least one meeting node belonging to the set $\mathcal{M}_1$ and there exists at least one meeting node belonging to the set $\mathcal{M}_2$ at l. According to Observation 2, there exists an ordering of the meeting nodes lying on l. Each robot fixes

a specific ordering $\mathcal{O}_2$ of the meeting nodes on l. Consider the meeting node $m'_1 \in \mathcal{M}_1 \cap l$, which has the highest order in $\mathcal{O}_2$ among all the meeting nodes belonging to $\mathcal{M}_1 \cap l$. Similarly, consider the meeting node $m'_2 \in \mathcal{M}_2 \cap l$, which has the highest order in $\mathcal{O}_2$ among all the meeting nodes belonging to $\mathcal{M}_2 \cap l$. m'_1 and m'_2 are selected as the target meeting nodes. In Figure 5.6 (b), C and D are the leading corners. The ordering $\mathcal{O}_2$ is defined as, $\mathcal{O}_2 = (m_8, m_5, m_6, m_7)$. m_5 and m_7 are selected as the target meeting nodes. Each robot in $\mathcal{R}_1$ (resp. $\mathcal{R}_2$) moves towards m'_1 (resp. m'_2) and finalizes the gathering at those meeting nodes. Since l is the line of symmetry for the meeting nodes, the ordering $\mathcal{O}_2$ depends only on the position of the fixed meeting nodes. As a result, the ordering remains invariant while the robots move toward their respective targets. Hence, the target meeting nodes remain invariant during the execution of the algorithm at any time $t > 0$. Therefore, we have the following lemma.

Lemma 5.4.3. *If $C(0) \in \mathcal{I}_2$, then the target meeting nodes remain invariant during the execution of the algorithm 2-nodeGathering() at any time $t > 0$.*

Proof. The proof follows from Lemma 3.5.1. □

5.4.2.3 $\mathcal{I}_3 \cup \mathcal{I}_{42} \cup \mathcal{I}_5$

If $C(t) \in \mathcal{I}_3$, the meeting nodes are either symmetric with respect to a single line of symmetry l or rotational symmetry. If the meeting nodes are symmetric with respect to l, there do not exist meeting nodes of both types on l. In case $C(t) \in \mathcal{I}_{42} \cup \mathcal{I}_5$, the symmetric configurations are transformed into asymmetric configurations belonging to $\mathcal{I}_3$. We begin this subcase by introducing some definitions that are relevant in the algorithm design of some special class of configurations that are asymmetric and a movement of a robot towards an adjacent node transforms those configurations into symmetric configurations.

Definition 5.4.4. *Let u be a node of the input graph containing a robot position r. Consider all the half-lines starting from the node u containing the robot position r. Scan all those half-lines and associate $h_t(v)$ to each node v, that the half-line encounters. A string terminates when the last node occupied by a robot in the half-line is encountered. The four strings that are generated by the robot r are denoted by $\beta_{left}(r)$, $\beta_{right}(r)$, $\beta_{top}(r)$ and $\beta_{bottom}(r)$ [28].*

Next, assume that the initial configuration $C(0)$ is such that the meeting nodes are symmetric with respect to rotational symmetry. Let r be the robot at the center of rotation c. Then, we have the following definition.

Definition 5.4.5. *Consider all the strings generated by the robot r. The strings are said to be nearly equal if the adjacent nodes of c are occupied by the robots and can be obtained from one another by reversing one occurrence of the substring* 01*. That is, the strings can be made equal by moving a robot toward an adjacent node whose movement is pending. A configuration is said to be nearly rotational if the strings generated by the central robot r are nearly equal.*

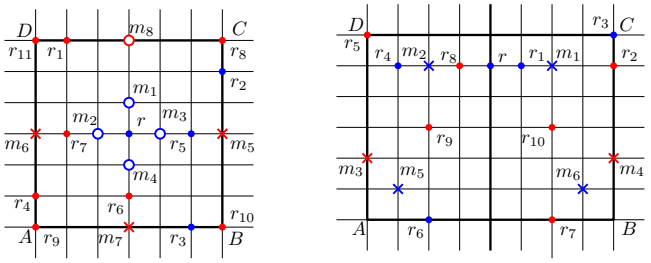

FIGURE 5.7: (a) Nearly rotational configuration. (b) Nearly reflective configuration.

Next, assume that the initial configuration $C(0)$ is such that the meeting nodes are symmetric with respect to a unique line of symmetry l. Let r be the unique robot on l that has the maximum configuration view.

Definition 5.4.6. *Consider the strings $\beta_{left}(r)$ and $\beta_{right}(r)$ as the strings generated by r and which terminate away from l. The strings $\beta_{left}(r)$ and $\beta_{right}(r)$ are said to be nearly equal if the strings $\beta_{left}(r)$ and $\beta_{right}(r)$ can be obtained from one another by just reversing one occurrence of the substring* 01*. A configuration is said to be nearly reflective, if the strings $\beta_{left}(r)$ and $\beta_{right}(r)$ are nearly equal.*

In Figure 5.7(a), the strings generated by the robot r are given by $\{110, 110, 110, 101\}$. The blue circles at m_1, m_2, m_3 and m_4 represent robot positions belonging to $\mathcal{R}_1$ at the meeting nodes of type $\mathcal{M}_1$. The red circle at the top denotes a meeting node m_8 with a robot position on it. In Figure 5.7(b), the strings generated by the robot r and which terminate away from l are given by $\{1010, 1001\}$.

Now, we will explain the Symmetry Breaking (SB) phase, in which symmetric configurations belonging to $\mathcal{I}_{42} \cup \mathcal{I}_5$ are transformed into asymmetric configurations belonging to

$\mathcal{I}_3$. Later, we will proceed with the procedure of executing the gathering of all configurations belonging to $\mathcal{I}_3$. This phase is similar to the Symmetry Breaking phase, explained in Chapter 3. However, in this chapter, this phase executes without the occurrence of any multiplicity.

SB Phase: Consider the case when the initial configuration belongs to $\mathcal{I}_{42} \cup \mathcal{I}_5$. The configurations considered in this phase include those that admit a line of symmetry l with robots on l and those that admit rotational symmetry with robots on c. A unique robot is allowed to move toward an adjacent node, thus transforming the configuration into an asymmetric configuration. While moving the unique robot on $l \cup \{c\}$, it may result in forming an unstable multiplicity at one of the target meeting nodes. In order to avoid such a scenario, each robot performs the procedure *AllowtoMove()*.

AllowtoMove(): First, assume that $C(0) \in \mathcal{I}_{42}$. Let r be the unique robot on l with the maximum configuration view and v be the node of the graph containing r. Define $N_l(v)$=$\{u : u$ is adjacent to v and $u \notin l\}$. If there exists a node $u \in N_l(v)$ such that u is not a robot position, then r moves towards u, and the configuration becomes asymmetric. Otherwise, consider the case when all the nodes belonging to $N_l(v)$ contain robot positions. The robot position representing the last 1 in the strings $\beta_{left}(r)$ and $\beta_{right}(r)$ generated by the robot r are allowed to move towards an adjacent node away from l. Due to the asynchronous behavior of the scheduler, there may be a possible pending move, and the configuration may transform into a nearly reflective configuration. The procedure AllowtoMove() moves exactly those robots whose moves are pending. As only two robots are allowed to move at any instant of time, the procedure allows the configuration to regain its symmetry by identifying the pending move. Execution of the procedure AllowtoMove() continues unless there exist nodes in $N_l(v)$ which do not contain any robot positions. While there is a space around r, then r moves towards an adjacent node. The procedure proceeds similarly, as in the case when $N_l(v)$ does not contain any robot positions.

Next, consider the case when $C(0) \in \mathcal{I}_5$. Let r be the robot on c and v be the central node of the graph containing r. Define $N_c(v) = \{u : u$ is adjacent to $v\}$. The procedure follows similarly, as in the above case, when $N_c(v)$ does not contain any robot positions. Consider the case when $N_c(v)$ contains robot positions. The robot positions representing the last 1 in the strings generated by the robot r are allowed to move towards an

adjacent node, away from c. Due to the asynchronous behavior of the scheduler, there may be a possible pending move. As a result, the configuration may become nearly rotational configuration. Execution of the procedure AllowtoMove() continues unless there exist nodes in $N_c(v)$ which do not contain any robot positions. This procedure allows regaining the symmetry of the configuration by moving the robots whose moves are pending. Suppose the configuration admits multiple lines of symmetry and $N_c(v)$ does not contain any robot positions. In that case, r moves towards an adjacent node, transforming the configuration into a configuration admitting a unique line of symmetry. The rest of the procedure follows similarly, as in the case when $C(0) \in \mathcal{I}_{42}$. Hence, we have the following lemma.

Lemma 5.4.7. *Procedure AllowtoMove() transforms any configuration $C(0) \in \mathcal{I}_{42} \cup \mathcal{I}_5$ into an asymmetric configuration which is neither a nearly reflective nor a nearly rotational configuration.*

The pseudo-code description of this phase is given in Algorithm 5.2.

Algorithm 5.2: *AllowtoMove*()

Input: $C(t) = (\mathcal{R}(t), \mathcal{M}) \in \mathcal{I}_{42} \cup \mathcal{I}_{51} \cup (\mathcal{I}_3$ such that $C(t)$ is either a nearly reflective or a nearly rotational configuration)

```
if C(t) ∈ I_42 then
    Let r be the unique robot on l with the maximum configuration view ;
    N_l(v)={u : u is adjacent to v and u ∉ l} ;
    if N_l(v) does not contain any robot positions then
        r moves towards an adjacent node ;
    else
        The last robots in the strings β_left(r) and β_right(r) move towards an adjacent node
          unless N_l(v) does not contain any robot positions ;
else if C(t) ∈ I_5 then
    Let r be the robot on the center of rotation c;
    N_c(v) : {u : u is adjacent to v};
    if N_c(v) does not contain any robot positions then
        r moves towards an adjacent node ;
    else
        The last robots in the strings generated by r move towards an adjacent node unless
          N_c(v) does not contain any robot positions ;
```

We next move to the case where the initial configuration belongs to $\mathcal{I}_3$ and such that it is neither a nearly reflective nor a nearly rotational configuration. In this case, each robot executes *GatheringAsym()*. The main crux of the procedure is to maintain the asymmetry of the configuration. The overview of the procedure *GatheringAsym*() is summarized in Table 5.3. In Table 5.3, the Boolean variable on the left is true if and only

if the condition on the right is satisfied. Note that the initial configuration corresponds to the predicate $\neg\mathcal{C}_1 \wedge \neg\mathcal{C}_2 \wedge \neg\mathcal{C}_3 \wedge \neg\mathcal{C}_4 \wedge \neg\mathcal{C}_5$, while the final configuration corresponds to the predicate $\mathcal{C}_1 \wedge \mathcal{C}_2 \wedge \mathcal{C}_3 \wedge \mathcal{C}_4 \wedge \mathcal{C}_5$. The pseudo-code corresponding to the procedure $GatheringAsym()$ is given in Algorithm 5.3.

Algorithm 5.3: $GatheringAsym()$

Input: $C(t) = (\mathcal{R}(t),\mathcal{M}) \in \mathcal{I}_3$ and $C(t)$ is neither a nearly rotational nor a nearly reflective configuration

1 $GuardSelection()$;
2 $TargetMeetingNodesSelection()$;
3 $PivotSelection()$;
4 $MovetoDestination()$;
5 $CreateMultiplicity()$;
6 $GuardMovement()$;

GS Phase: In this phase, a guard is selected and placed in such a way that it remains invariant during the execution of this phase. The main objective of this phase is to keep the target meeting nodes invariant during the execution of the algorithm. This phase proceeds similarly as in the Guard Selection and Placement phase in Chapter 3. The main difference, in this case, is the destination where the guard is placed. In this regard, the guard is placed at a sufficiently large distance from MER_F.

Consider the smallest enclosing rectangle MER_F of the meeting nodes. Assume that the meeting nodes are symmetric with respect to a unique vertical line of symmetry l. The algorithm executes similarly in the case when l is a horizontal line of symmetry. Recall from chapter 3, that d_1 is the horizontal distance from l to the boundary of MER_F,

Predicates and their Definitions	
Predicates $C(0)$	**Definition**
$\mathcal{C}_1$	The guard is placed at a distance of $d'' = max\{4d_1, 4d'_1\}$ from l, where d_1 is the length of the farthest meeting node from l and d'_1 is the length of the second farthest robot from l
$\mathcal{C}_2$	All the non-pivot robots in $\mathcal{R}_1$ (resp. $\mathcal{R}_2$) are at the nodes $\mathcal{D}_1$ ($\mathcal{D}_2$), where $\mathcal{D}_1$ is the node which is at a distance d_1 away from the guard and lies between $l \cup \{c\}$ and the guard. Similarly, $\mathcal{D}_2$ is the node that is at a horizontal distance $2d_1$ away from the guard and lies between $l \cup \{c\}$ and the guard. The pivots are the robots that are selected in the Pivot Selection (PS) phase.
$\mathcal{C}_3$	All the pivots are at the target meeting nodes
$\mathcal{C}_4$	The non-guards are at the target meeting nodes
$\mathcal{C}_5$	The guard is at the target meeting node

TABLE 5.3: Predicates and their Definitions

i.e., $2d_1$ denotes the length of the rectangle MER_F. Depending on the positions of the robots in the initial configuration, the following cases are to be considered.

1. Each robot is inside or on the rectangle MER_F. Let r be the robot at the maximum distance from l. If multiple such robots exist, the ties are broken by considering the robot with the maximum configuration view. The robot r is selected as the guard. r moves towards a node v that is at a horizontal distance of $4d_1$ from l. In Figure 5.8, r_5 is the robot farthest from l and having the maximum configuration view. In Figure 5.9, r_5 moves towards a node which is at a distance of $4d_1$ from l. Note that while the guard moves towards its destination, it remains invariant.

2. There exists a unique robot r outside the MER_F. The unique robot r is selected as the guard. The guard moves towards a node v that is at a distance of $4d_1$ from l. In this case, the dimension of the transformed MER becomes $4d_1 \times q$.

3. There exists more than one robot outside the rectangle MER_F. If there exists a unique robot r farthest from l, then r is selected as a guard. Let r' be the robot that is second farthest from l and at a distance d'_1 from l. r moves towards a node u that is at a distance of $4d'_1$ from l. In case there is more than one robot outside the rectangle MER_F, the guard r is selected as the robot which is farthest from l and with the maximum configuration view. While r moves towards an adjacent node away from l, it becomes the unique farthest robot from l. The rest of the procedure follows from the case when there exists a unique robot r farthest from l.

While the guard reaches the node at a distance $d'' = max\{4d_1, 4d'_1\}$ from l, it finally moves toward the closest corner. If there is more than one closest corner, it moves arbitrarily toward one of its closest corners. The GS phase proceeds similarly in the case when l is a diagonal line of symmetry. If l is a horizontal line of symmetry, then d_1 is defined as the vertical distance from l to the boundary of MER_F. The distance d'_1 is also measured as a vertical separation from l. If the meeting nodes are symmetric with respect to rotational symmetry, then d_1 and d'_1 denote the distance from c to the boundary of MER_F. In the following lemma, we will prove that the guard remains invariant while it moves toward its destination. The pseudo-code description of this phase is given in Algorithm 5.4.

Lemma 5.4.8. *In the GS phase, the guard remains invariant while it moves toward its destination.*

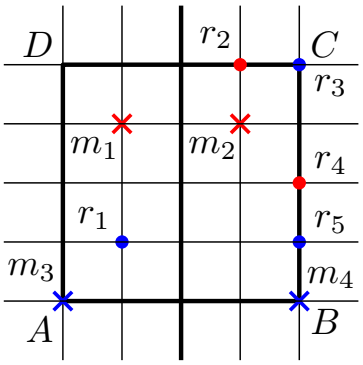

FIGURE 5.8: $\mathcal{I}_3$-configuration. Example configuration showing the GS phase.

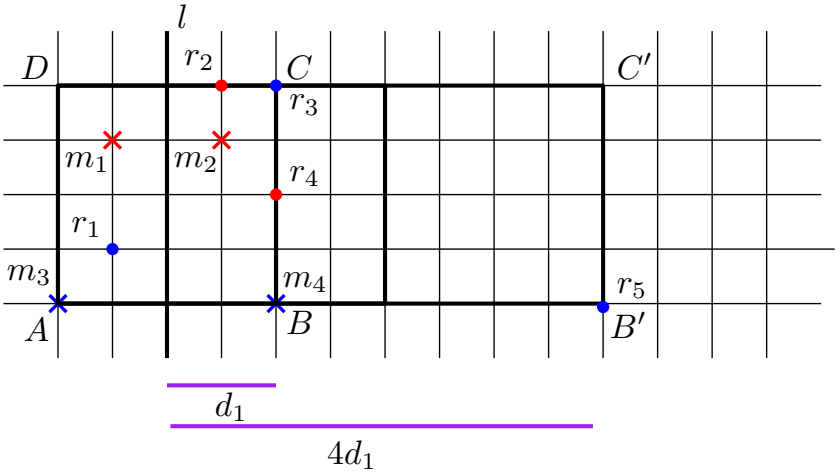

FIGURE 5.9: r_5 is the guard. r_5 moves towards a node at a large distance from $MER_F = ABCD$ and finally towards the closest corner. MER is the rectangle $AB'C'D$.

Proof. The meeting nodes are symmetric with respect to either a unique line of symmetry l or rotational symmetry. We first assume the case when the meeting nodes are symmetric with respect to a vertical line of symmetry l. The following cases are to be considered.

Case 1. Each robot is inside or on the rectangle MER_F. The guard r is selected as the robot, which is farthest from l. If there are multiple such robots, then the guard is selected as the robot which is farthest from l and has the maximum configuration view. While r moves towards an adjacent node away from l, it becomes the unique farthest from l. r remains the unique farthest robot from l while moving towards its destination. Hence, the guard becomes uniquely identifiable by the other robots.

Case 2. There exists a unique robot r outside the rectangle MER_F. r is selected as the guard. Since r is the unique robot outside MER_F, it is the unique farthest robot

Algorithm 5.4: *GuardSelection*()

Input: $C(t) = (\mathcal{R}(t),\mathcal{M}) \in \mathcal{I}_3$ and $C(t)$ is neither a nearly rotational nor a nearly reflective configuration, Length of MER_F= $2d_1$

```
if each robot is inside or on the rectangle MER_F then
    if there exists exactly one robot r farthest from l then
        r is selected as a guard and moves towards a node at a distance of 4d_1
        from l and finally towards its closest corner ;
    else
        The robot r farthest from l and having the maximum configuration view
        is selected as a guard. r moves towards a node at a distance of 4d_1 from
        l and finally towards its closest corner ;
else if there exist robots outside the rectangle MER_F then
    if there is a unique robot r outside MER_F then
        r is selected as a guard and moves towards a node at a distance of 4d_1
        from l and finally towards its closest corner;
    else if there exists more than one robot outside MER_F then
        if there exists a unique farthest robot r from l then
            r is selected as a guard and moves towards a node u that is at a
            distance of 4d'_1 from l and finally towards its closest corner, where d'_1
            is the horizontal distance from l to the second farthest robot from l;
        else
            The robot r farthest from l and having the maximum configuration
            view is selected as a guard. r moves towards an adjacent node and
            becomes the unique farthest robot from l ;
```

from l. While r moves towards its destination, the invariance property of r follows from the previous case.

Case 3. There exists more than one robot outside the rectangle MER_F. The guard is selected as the robot, which is at the maximum distance from l and with the maximum configuration view in case of a tie. While the guard r moves towards an adjacent away from l, it becomes the unique robot that is farthest from l. The invariance property of r follows from the preceding cases while r moves towards its destination.

The proof is similar when l is a horizontal or a diagonal line of symmetry. The proof is similar if the meeting nodes are symmetric with respect to rotational symmetry. The only difference lies in defining the distances d_1 and d'_1. The distances are from c rather than l. □

In the previous lemma, we have proved that the guard remains invariant while it moves towards its destination. Finally, the guard is at a distance of $d'' = max\{4d_1, 4d'_1\}$ from l. Hence, we have the following lemma.

Lemma 5.4.9. *If $C(0) \in \mathcal{I}_3$, then the GS phase terminates with $C_1 = true$.*

TMS Phase: In this phase, the target meeting nodes for the gathering over heterogeneous meeting nodes problem are selected. The guard is selected in the GS phase. Since the configuration is asymmetric, there exists an ordering of the meeting nodes with respect to the guard. Consider the ordering $\mathcal{O}$ of the meeting nodes in Chapter 3, according to their positions in the string direction associated to the corner. The ordering remains invariant unless the guard moves. Consider m' as the meeting node belonging to the set $\mathcal{M}_1$ and closest to the guard among all the meeting nodes in $\mathcal{M}_1$. If there are multiple such meeting nodes in $\mathcal{M}_1$, then m' is selected as the closest meeting node in $\mathcal{M}_1$, which has the highest order with respect to the ordering $\mathcal{O}$. Similarly, let m'' be the meeting node in $\mathcal{M}_2$, which is closest to the guard and has the highest order with respect to the ordering $\mathcal{O}$ among all the meeting nodes in $\mathcal{M}_2$. m' and m'' are selected as the target meeting nodes. In Figure 5.9, m_2 and m_4 are selected as the target meeting nodes. The pseudo-code description of this phase is given in Algorithm 5.5.

Algorithm 5.5: $TargetMeetingNodesSelection()$

Input: $C(t) = (\mathcal{R}(t),\mathcal{M}) \in \mathcal{I}_3$ and $C(t)$ is neither a nearly rotational nor a nearly reflective configuration

```
if there exists a unique meetings node m' ∈ M_1 and a unique meeting node
  m'' ∈ M_2 which is closest to the guard then
    Each robot in R_1 (resp. R_2) selects m' (resp. m'') as the target meeting
      node;
else
    Let m' (resp. m'') be the meeting node in M_1 (resp. M_2) which is closest to
      the guard and that has the highest order in O;
    Each robot in R_1 (resp. R_2) selects m' (resp. m'') as the target meeting
      node ;
```

PS Phase: In this phase, some robots are selected from the non-guards as *pivots*. The pivots guide the other non-pivot robots to move toward the target meeting nodes and finalize the gathering process. Consider the fixed ordering $\mathcal{O}$ with respect to the guard. The three robots that have the highest order in $\mathcal{O}$ are selected as the pivots. Let R_{piv} denote the set of all pivot robots. The pseudo-code description of this phase is given in Algorithm 5.6.

Algorithm 5.6: *PivotSelection*()

Input: $C(t) = (\mathcal{R}(t),\mathcal{M}) \in \mathcal{I}_3$ and $C(t)$ is neither a nearly rotational nor a nearly reflective configuration

1 The three robots that have the maximum order in $\mathcal{O}$ are selected as the pivots ;

MD Phase: In this phase, the non-guards move towards destination nodes, which are at a sufficiently large distance from the target meeting nodes. The movement is done in such a way that no unstable multiplicity nodes are created at the target meeting nodes during the movements. Note that while the guard reaches its destination node in the GS phase, the MER changes. Depending on the initial configuration, the transformed MER is such that its dimension becomes $d'' \times q$, where $d'' = max\{4d_1, 4d'_1\}$. Assume that the meeting nodes are symmetric with respect to a vertical line of symmetry l. The procedure proceeds similarly in the case when l is a horizontal or diagonal line of symmetry. The following definitions and notations are to be considered in the phase description.

- The horizontal grid line passing through the guard and perpendicular to l is denoted by $horizon_l$.
- *Destination for $\mathcal{R}_1$ ($\mathcal{D}_1$) and $\mathcal{R}_2$ ($\mathcal{D}_2$):* Let v be the node on $horizon_l$, which is at a distance d_1 away from the guard and lies between l and the guard. $\mathcal{D}_1$ is the node v. Let u be the node on $horizon_l$, which is at a horizontal distance $2d_1$ away from the guard and lies between l and the guard. $\mathcal{D}_2$ is the node u.

All the non-guard robots in $\mathcal{R}_1$ (resp. $\mathcal{R}_2$), excluding the pivot robots move towards $\mathcal{D}_1$ (resp. $\mathcal{D}_2$).

Note that while the non-guards excluding the pivots move towards their destinations, an unstable multiplicity may occur on the target meeting nodes. In order to avoid such a scenario, each robot $r_i \in \mathcal{R}_1$ (resp. $\mathcal{R}_2$) moves towards $\mathcal{D}_1$ (resp. $\mathcal{D}_2$), depending on whether $r_i \in \mathcal{R}_1$ or $\mathcal{R}_2$, if it has a free path towards $\mathcal{D}_1$ (resp. $\mathcal{D}_2$). If there exists some robot r_j within a Manhattan distance of 2 from r_i, and r_j has a free path for moving towards its destination, then r_i checks whether it has a higher order than r_j in $\mathcal{O}$. If yes, then r_i moves, otherwise, it does not move. The pseudo-code description of this phase is given in Algorithm 5.7.

Algorithm 5.7: *MovetoDestination*()

Input: $C(t) = (\mathcal{R}(t),\mathcal{M}) \in \mathcal{I}_3$ and $C(t)$ is neither a nearly rotational nor a nearly reflective configuration

Let $horizon_l$ be the horizontal grid line passing through the guard;
$\mathcal{D}_1$ is the node v lying on $horizon_l$ which is at a distance d_1 away from the guard and lies between l and the guard ;
$\mathcal{D}_2$ is the node u lying on $horizon_l$ which is at a horizontal distance $2d_1$ away from the guard and lies between l and the guard ;
if *r_i has a free path towards $\mathcal{D}_1$ or $\mathcal{D}_2$* **then**
    if *r_i has a higher order in $\mathcal{O}$ than any other r_j that is within a Manhattan distance of 2 from r_i* **then**
        r_i moves towards either $\mathcal{D}_1$ or $\mathcal{D}_2$;
    else
        r_i does not move ;
else
    r_i does not move ;

If the meeting nodes are symmetric with respect to rotational symmetry, then $\mathcal{D}_1$ and $\mathcal{D}_2$ are the destinations measured from c.

Lemma 5.4.10. *No unstable multiplicity is created at the target meeting nodes during the MD phase.*

Proof. Notice that each target meeting node lies either inside or on the boundary of the rectangle MER_F. In order to ensure that no unstable multiplicity is created at the target meeting nodes, the non-pivots must not collide while moving towards their respective destinations $\mathcal{D}_1$ and $\mathcal{D}_2$. The two robots belonging to two different teams can collide if and only if they have free paths toward their destinations and are separated by a Manhattan distance of 2 units. Consider the case when two robots have free paths towards their destinations $\mathcal{D}_1$ and $\mathcal{D}_2$ belonging to two different teams and are separated by a Manhattan distance of 2 units. The algorithm ensures that exactly one robot moves towards the destination in such a scenario by allowing the robot which has a higher order in $\mathcal{O}$ to move towards its destination. Hence, no unstable multiplicity is created at the target meeting nodes. □

At the end of the MD phase, the non-pivots are at their destinations $\mathcal{D}_1$ and $\mathcal{D}_2$. Hence, we have the following lemma.

Lemma 5.4.11. *If $C(0) \in \mathcal{I}_3$, then the MD phase terminates with $C_2 = true$.*

CM Phase: In this phase, all the non-guards move towards the target meeting nodes and creates stable multiplicities at the target meeting nodes. First, the pivot robots that are selected in the PS phase move toward the target meeting nodes. Since there are three pivot robots selected in the PS phase, by the Pigeonhole principle, at least two robots must belong to the same team of robots. The pivot robot(s) in $\mathcal{R}_1$ (resp. $\mathcal{R}_2$) move towards the target meeting nodes $m' \in \mathcal{M}_1$ (resp. $m'' \in \mathcal{M}_2$) sequentially and according to the order in $\mathcal{O}$, i.e., the pivot which is closest to the guard and having the minimum order in $\mathcal{O}$ first moves towards the target meeting nodes. While it reaches the target meeting nodes, the other pivots start moving according to their ordering in $\mathcal{O}$. A pivot moves only when it finds that each non-pivot reaches either $\mathcal{D}_1$ or $\mathcal{D}_2$, depending on whether the robot belongs to either $\mathcal{R}_1$ or $\mathcal{R}_2$. A pivot can determine whether each non-pivot has reached its targets as the non-pivots are at least d_1 distance from MER_F. This movement ensures that at least one stable multiplicity is created at one of the target meeting nodes. Since the robots have global-weak multiplicity detection capability, all the non-pivots start moving towards the target meeting nodes when a multiplicity is created at one of the target meeting nodes. First, the robots in $\mathcal{R}_2$ move towards the target meeting node m'' belonging to the set $\mathcal{M}_2$ in a shortest path. While all the non-guards in $\mathcal{R}_2$ reach m'', the non-guards in $\mathcal{R}_1$ start moving towards the target meeting node m' belonging to the set $\mathcal{M}_1$, in a free path. The robots in $\mathcal{R}_1$ will only move when it finds that there are no non-guards other than the non-guards residing in its current node within a distance d_1 from itself. As $|\mathcal{R}| \geq 5$, this movement creates stable multiplicities at the target meeting nodes. The pseudo-code description of this phase is given in Algorithm 5.8.

Lemma 5.4.12. *During the CM phase, the target meeting nodes remain invariant.*

Proof. In the TMS phase, the target meeting nodes are selected as the meeting nodes m' and m''. m' and m'' are defined with respect to the ordering $\mathcal{O}$. The ordering $\mathcal{O}$ remains invariant unless the guard moves. As a result, the ordering remains invariant while the non-guards move toward the target meeting nodes. Since the guard does not move in the CM phase, the target meeting nodes remain invariant. □

At the end of the CM phase, each non-guards are at the target meeting nodes. Hence, we have the following lemma.

Lemma 5.4.13. *If $C(0) \in \mathcal{I}_3$, the CM phase terminates with $C_3 = true$ and $C_4 = true$.*

Algorithm 5.8: *CreateMultiplicity*()

Input: $C(t) = (\mathcal{R}(t),\mathcal{M}) \in \mathcal{I}_3$ and $C(t)$ is neither a nearly rotational nor a nearly reflective configuration

Let r_1, r_2 and r_3 be the three pivot robots selected according to procedure *PivotSelection*() ;
The robot closest to the target meeting nodes and having minimum order in $\mathcal{O}$ among r_1, r_2 and r_3 move towards the target meeting nodes;
if *there exists a multiplicity at one of the target meeting nodes* **then**
    The closest non-guard and non-pivot belonging to $\mathcal{R}_2$ which is not at the target meeting nodes, first move towards m''. While the robots reach m'', the robots in $\mathcal{R}_1$ move towards m';
else
    The non-pivots do not move ;

FG Phase: In this phase, the guard moves towards one of the target meeting nodes containing stable robot multiplicities. The guard moves towards either m' or m'', depending on whether it belongs to either $\mathcal{R}_1$ or $\mathcal{R}_2$. It moves only when it is the unique robot that is not at the target meeting nodes containing multiplicities. Since the target meeting nodes are selected as the closest meeting nodes from the guard, the guard can move in a free path towards m' or m''. As the robots are equipped with global-weak multiplicity detection capability, the guard can detect whether it is on a multiplicity node. The guard moves towards its respective destinations and thus, the gathering is finalized at the target meeting nodes. The pseudo-code description of this phase is given in Algorithm 5.9. At the end of the FG phase, the gathering is finalized at the target

Algorithm 5.9: *GuardMovement*()

Input: $C(t) = (\mathcal{R}(t),\mathcal{M}) \in \mathcal{I}_3$ and $C(t)$ is neither a nearly rotational nor a nearly reflective configuration

Let r be the unique robot that does not lie on a robot multiplicity node and not on a meeting node;
r moves towards its respective target meeting node;

meeting nodes. Hence, we have the following lemma.

Lemma 5.4.14. *If $C(0) \in \mathcal{I}_3$, the CM phase terminates with $C_5 = true$.*

The lemmas 5.4.9, 5.4.11, 5.4.13 and 5.4.14 ensure that starting from any initial configuration belonging to $\mathcal{I}_3$, such that the configuration is neither a nearly rotational nor a nearly reflective configuration, reaches the final configuration within a finite amount of time.

5.4.2.4 $\mathcal{I}_{41}$

If $C(t) \in \mathcal{I}_{41}$, $C(t)$ is symmetric with respect to a single line of symmetry, with both types of meeting nodes at the line of symmetry. In this class of configuration, each robot executes $GatheringSym()$.

GatheringSym(): Consider the case when $C(0) \in \mathcal{I}_{41}$. First, assume that $\mathcal{R}_1$ and $\mathcal{R}_2$ are independently symmetric. According to Observation 2, there exists a unique ordering $\mathcal{O}_2$ of the meeting nodes on l. Consider the meeting node $m'_1 \in \mathcal{M}_1 \cap l$, which has the highest order in $\mathcal{O}_2$ among all the meeting nodes belonging to $\mathcal{M}_1 \cap l$. Similarly, consider the meeting node $m'_2 \in \mathcal{M}_2 \cap l$, which has the highest order in $\mathcal{O}_2$ among all the meeting nodes belonging to $\mathcal{M}_2 \cap l$. m'_1 and m'_2 are selected as the target meeting nodes. Each robot in $\mathcal{R}_1$ (resp. $\mathcal{R}_2$) moves towards m'_1 (resp. m'_2) only when it has a free path toward their respective targets. To ensure this, all the closest robots which are not on l and have the maximum view in case of a tie move toward their respective targets. No other robots are allowed to move unless the closest robots reach the targets. Note that exactly two symmetric robots move towards l at a particular instant of time. This movement is done to ensure that there is at most one pending move. The robots can identify this pending move by considering the previous position of the robot whose move is pending, and the possible symmetry can be re-established. This movement ensures that no multiplicity is created at the nodes unless the nodes are the target meeting nodes during the execution of the algorithm. Thus, the gathering is finalized at the target meeting nodes. While the robots not on l reach l, the robots on l move toward the target meeting nodes. The pseudo-code description is given in Algorithm 5.10.

Finally, consider the case when $C(0) \in \mathcal{I}_{41}$ and $\mathcal{R}_1$ (resp. $\mathcal{R}_2$) are not independently symmetric. The procedure follows similarly, as in the case when $C(0) \in \mathcal{I}_3$.

Algorithm 5.10: $GatheringSym()$

Input: $C(t) = (\mathcal{R}(t), \mathcal{M}) \in \mathcal{I}_{41}$

1 Let m'_1 and m'_2 be the meeting nodes on l belonging to $\mathcal{M}_1$ and $\mathcal{M}_2$ and having the maximum order in $\mathcal{O}_2$;

2 Each closest robot in $\mathcal{R}_1$ (resp. $\mathcal{R}_2$) which is not on m'_1 (resp. m'_2) and having the maximum configuration views, moves towards the target meeting nodes ;

Next, we will discuss the technical difficulty that may arise due to the presence of two different types of meeting nodes and the presence of two different classes of robots. We

will also discuss why the proposed algorithm avoids unstable multiplicities throughout the execution. First, we discuss the main reason behind the avoidance of multiplicities in our proposed algorithm.

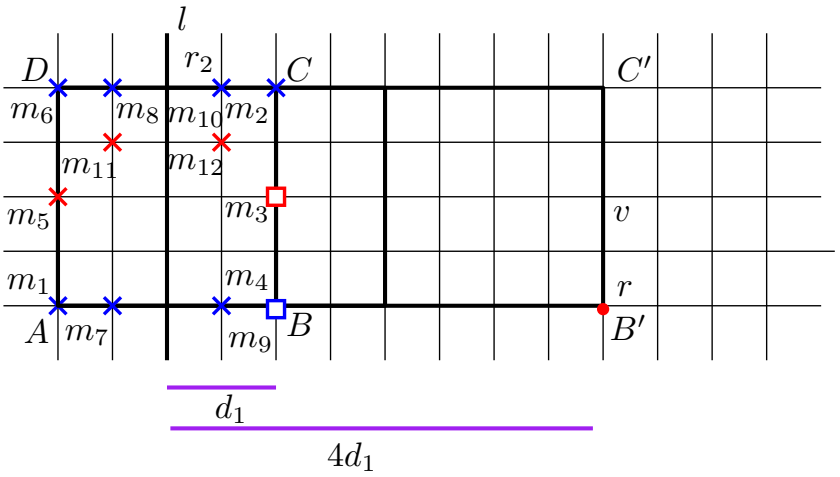

FIGURE 5.10: An example of a configuration showing that unstable multiplicity may not successfully terminate the gathering.

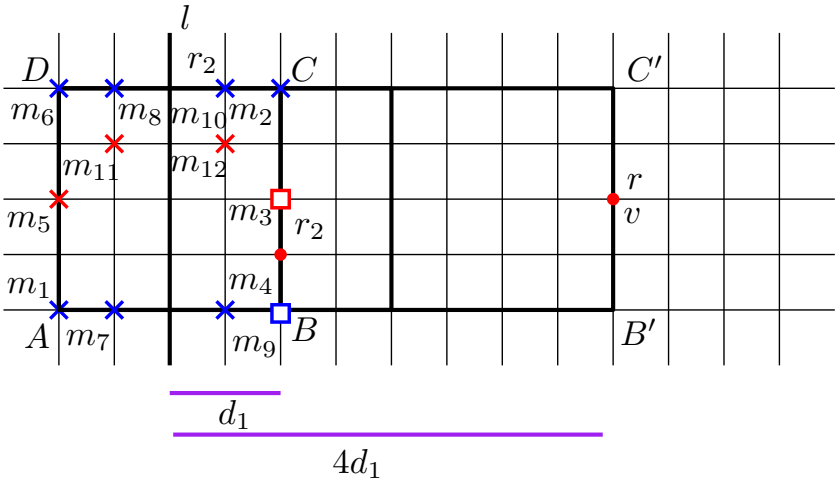

FIGURE 5.11: The guard is at the node v at time t_1.

Assume that there exist unstable multiplicity nodes at the target meeting nodes. In Figure 5.10, m_3 and m_4 are the target meeting nodes containing unstable multiplicities. Assume that at time t, the guard r is activated and it finds multiplicity nodes at the meeting nodes m_3 and m_4. The guard cannot distinguish between a stable and an unstable multiplicity. According to the procedure *GatheringAsym*(), the guard starts moving towards m_3. Assume that at time $t_1 > t$, the guard is at node v (Figure 5.11). Suppose that at the same time t_1, one robot r_2 belonging to $\mathcal{R}_2$ at m_4 starts moving towards m_3 and it is one node away from m_3. Note that the robot r_2 moves because

the destination of r_2 is m_3 and not m_4. Consequently, it might be the case that at time $t_2 > t_1$, the guard finds that there are still robots that are not at the target meeting nodes and accordingly, the guard might move towards the corner C' as it is also the closest corner from v. Now, suppose that at time $t_3 > t_2$, the guard r is at the corner C' (Figure 5.12). As a result, one of the target meeting nodes changes from m_4 to m_2. Now, suppose that the guard starts moving towards m_3 at time $t_4 > t_3$. Note that at time t_4, the robot r_2 must reach m_3, otherwise the guard would not have started moving towards its destination. Since we have assumed unstable multiplicity at m_3 it might be the case that there are multiple robots (say mt_1) belonging to $\mathcal{R}_1$ that are on m_3. At any time $t < t_1$, such robots might have decided to move towards m_4, but due to the asynchronous behavior of the scheduler, their move is pending. Consequently, there might exist a time $t_5 > t_4$, the guard is at m_3, all the robots belonging to $\mathcal{R}_1$, except the multiplicity mt_1 are at m_2, all the robots belonging to $\mathcal{R}_2$ are at m_3 and the multiplicity mt_1 is at m_4 (Figure 5.13). Thus, in the final configuration, there are three multiplicity nodes. Since the meeting nodes belonging to $\mathcal{M}_1$ are symmetric with respect to rotational symmetry, and the robots in $\mathcal{R}_1$ are symmetric in the final configuration, the gathering can not be terminated successfully.

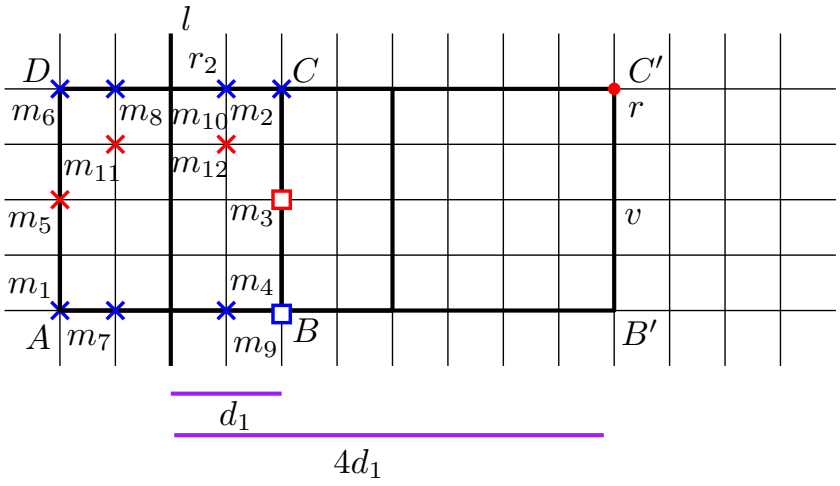

FIGURE 5.12: The guard is at the corner C' at time t_3.

5.4.3 Correctness

Theorem 5.4.15. *Algorithm 2-nodeGathering() solves the gathering over heterogeneous meeting nodes problem for any initial configuration belonging to the set* $\mathcal{I} \setminus \mathcal{U}$.

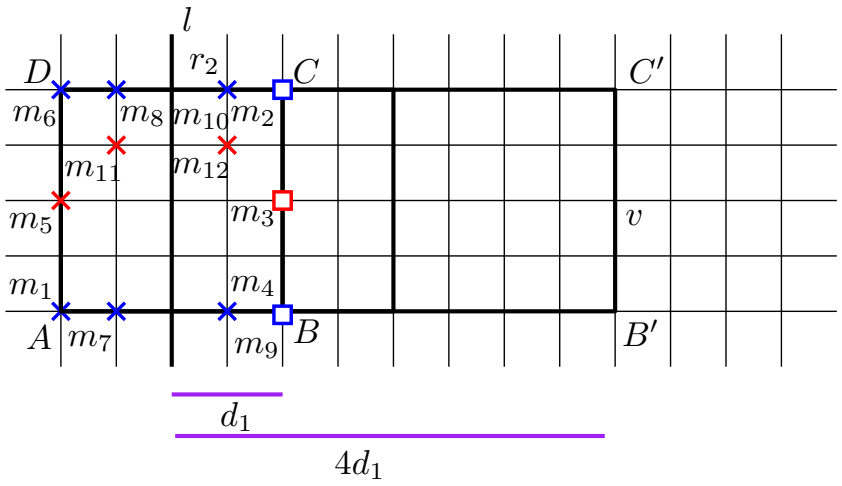

FIGURE 5.13: Final Configuration for the configuration in Figure 5.10.

Proof. Consider the following cases.

Case 1. $C(0) \in \mathcal{I}_1$. According to Lemma 5.4.2, the target meeting nodes remain invariant during the movement of the robots towards themselves. As a result, each robot in $\mathcal{R}_1$ (resp. $\mathcal{R}_2$) moves towards the target meeting nodes in $\mathcal{M}_1$ (resp. $\mathcal{M}_2$) and finalizes the gathering at those meeting nodes.

Case 2. $C(0) \in \mathcal{I}_2$. According to Lemma 5.4.3, the target meeting nodes on l remain invariant during the movement of the robots towards themselves. As a result, each robot in $\mathcal{R}_1$ (resp. $\mathcal{R}_2$) moves towards the target meeting nodes in $\mathcal{M}_1$ (resp. $\mathcal{M}_2$) and finalizes the gathering at those meeting nodes.

Case 3. $C(0) \in \mathcal{I}_3$ and $C(0)$ is neither a nearly reflective nor a nearly rotational configuration. A guard is selected and placed in the GS phase. According to Lemma 5.4.8, the guard remains invariant while it moves towards its destination. Since the guard contains no symmetric image with respect to l, the configuration remains asymmetric in the GS phase. The target meeting nodes are selected in the TMS phase according to the position of the guard. The target meeting nodes remain invariant unless the guard moves. The pivots are selected in the PS phase. Lemma 5.4.10 ensures that no unstable multiplicities are created at the target meeting nodes in the MD phase. The pivots move towards the target meeting nodes and create stable multiplicities at the target meeting nodes. Since the robots have global-weak multiplicity detection capability, each non-pivot moves towards the target meeting nodes only when it finds stable multiplicities at the target meeting nodes in the CM phase. Lemma 5.4.12 ensures that the target

meeting nodes remain invariant during the CM phase. Finally, in the FG phase, the guard moves towards its target and finalizes the gathering.

Case 4. $C(0) \in \mathcal{I}_{42} \cup \mathcal{I}_5 \cup (I_3$ such that $C(0)$ is either a nearly reflective or a nearly rotational configuration). Lemma 5.4.7 ensures that the procedure $AllowtoMove()$, transforms the configuration into an asymmetric configuration which is neither a nearly reflective nor a nearly rotational configuration. The rest of the proof follows similarly as in Case 3.

Case 5. $C(0) \in \mathcal{I}_{41}$ and $\mathcal{R}_1$ and $\mathcal{R}_2$ are independently symmetric. The procedure $GatheringSym()$ ensures that the gathering is finalized at the target meeting nodes on l, without creating unstable multiplicities at the target meeting nodes.

Case 6. $C(0) \in \mathcal{I}_{41}$ and $\mathcal{R}_1$ and $\mathcal{R}_2$ are not independently symmetric. The proof follows from Case 2. □

5.4.4 Analysis of the Algorithm

In this subsection, the efficiency of the proposed algorithm has been studied. First, the efficiency has been studied with respect to the total number of moves executed by the robots. Next, we analyzed the time complexity of the algorithm in terms of the number of epochs.

5.4.4.1 Efficiency with respect to the total number of moves

First, we discuss the lower bound regarding the total number of moves executed by any algorithm that solves the gathering over heterogeneous meeting nodes problem. Consider an infinite path graph, where $MER = AB$ is of length $n+2$ and the nodes are labelled as $v_1, v_2, v_3, \ldots, v_n, v_{n+1}, v_{n+2}$ (Figure 5.14). Assume that each of the nodes v_1, v_i where i ranges from 3 to $(n+1)$ are occupied by the robots and v_2 and v_{n+2} contain meeting nodes m_1 and m_2. Let r_i be the robot on v_i. Without loss of generality, assume that $m_1 \in \mathcal{M}_2$ and $m_2 \in \mathcal{M}_1$. The robot r_1 must move towards m_1 and the other robots must move towards m_2 in order to finalize the gathering. Under these assumptions, any gathering algorithm that solves the gathering over heterogeneous meeting nodes problem requires $1 + 2 + \ldots n - 1 + 1 = \Omega(n^2)$ moves. Therefore, if d is the diameter of MER

and $d = \Omega(n)$, then any algorithm solves the problem in $\Omega(dn)$ moves. Hence, we have the following theorem.

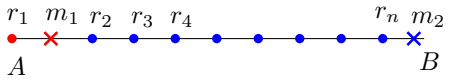

FIGURE 5.14: An example of a configuration showing the lower bound.

Theorem 5.4.16. *Any gathering algorithm solving the gathering over heterogeneous meeting nodes problem requires* $\Omega(dn)$ *moves.*

Next, we discuss the analysis of our proposed algorithm 2-*nodeGathering*() in terms of the total number of moves executed by the robots.

- In the SB phase, only the robots collinear with the unique robot on l having the maximum configuration view move towards an adjacent node. The worst-case scenario is the case when other robots occupy the nodes adjacent to the robot on l, and there is no free space available for the adjacent robots to move away from l. Therefore, it requires $O(n)$ moves. In case the configuration is symmetric with respect to rotational symmetry, it also requires $O(n)$ moves in the worst-case scenario.
- In the GS phase, only one robot is selected and allowed to move towards a node that is at a large distance from l. The distance is a constant multiple of the initial diameter of MER. Therefore, it requires $O(d)$ moves.
- In the MD phase, the non-pivots are allowed to move towards nodes that are at a sufficiently large distance from the initial MER. The distance is a constant multiple of the initial diameter of MER. Since there are $O(n)$ pivots, therefore this phase executes in $O(dn)$ moves.
- In the CM phase, all the non-guards move towards the target meeting node. The total number of moves in this phase is $O(dn)$.
- Finally, in the FG phase, only the guard moves toward the target meeting node. Therefore, the number of moves in this phase is $O(d)$.

Hence, the total number of moves performed by the algorithm is $O(dn)$.

Theorem 5.4.17. *The algorithm* 2-*nodeGathering*() *solves the gathering over heterogeneous meeting nodes problem in* $\Theta(dn)$ *moves.*

5.4.4.2 Efficiency with respect to the time complexity in epochs

Next, we discuss the analysis of our proposed algorithm 2-*nodeGathering*() in terms of the time measured in epochs.

- In the SB phase, only the robots collinear with the unique robot on l having the maximum configuration view move towards an adjacent node. The worst-case scenario is the case when the nodes adjacent to the robot on l are occupied by other robots, and there is no free space available for the adjacent robots to move away from l. In order to avoid undesirable multiplicities, the robots move only when the adjacent nodes do not contain any robot positions. Therefore, it requires $O(n)$ epochs. In case the configuration is symmetric with respect to rotational symmetry, it also requires $O(n)$ moves in the worst-case scenario. Therefore, this phase terminates in $O(n)$ epochs.
- In the GS phase, only one robot is selected and allowed to move towards a node that is at a large distance from l. The total distance traveled by the guard is a constant multiple of d_1. Therefore, this phase terminates in $O(d)$ epochs, since $d_1 = O(d)$.
- In the MD phase, the non-pivots are allowed to move towards nodes that are at a sufficiently large distance from the initial MER. The movement of the non-pivots is sequential. At any moment of time, only one robot is allowed to move toward its destination. Thus, each robot moves towards its destination in $O(d)$ epochs, and hence this phase terminates in $O(dn)$ epochs.
- In the CM phase, all the *non-guards* move towards the target meeting node. The maximum number of epochs required to terminate this phase is determined by the maximum distance from the non-guards to the target meeting nodes. Therefore, this phase terminates in $O(d)$ epochs.
- Finally, in the FG phase, only the guard moves toward the target meeting node. Therefore, this phase terminates in $O(d)$ epochs.

Hence, the algorithm terminates in $O(dn)$ epochs. Any algorithm that solves the problem for the configuration in Figure 5.14 requires $O(Dn)$ epochs to ensure the gathering. The main reason behind this requirement of time is due to the selection of the guard in

the Guard Selection and Placement phase. Note that the guard is selected in order to maintain the asymmetry of the configuration. It might be the case that no guards are required to ensure no ungatherable configurations are reached during the execution of the algorithm. If no guards are selected and placed, it might be the case that the sequential movements of the robots are avoided and the algorithm terminates in $O(D)$ epochs.

Note that a lower bound regarding the total time measured in epochs is $\Omega(d)$. Therefore, we have the following theorem.

Theorem 5.4.18. *The algorithm 2-nodeGathering() solves the gathering over heterogeneous meeting nodes problem in $O(dn)$ epochs.*

5.5 Conclusion

In this chapter, the gathering over heterogeneous meeting nodes problem has been studied in infinite square grids. We have shown that some configurations exist that remain ungatherable. The set of all ungatherable configurations is denoted by $\mathcal{U}$. A deterministic distributed algorithm has been proposed for all the remaining configurations, assuming that the initial configurations comprise at least five robots. The efficiency of the proposed algorithm has been discussed in terms of the total number of moves executed by the robots. The time complexity of the algorithm has been analyzed in terms of epochs. However, the gathering algorithm is not optimal in terms of the exact number of moves traveled by the robots in order to finalize the gathering. One immediate future task would be to consider the min-sum gathering over heterogeneous meeting nodes problem, which requires minimizing the total number of robot moves.

Chapter 6

Parking Problem in Infinite Grids

6.1 Overview of the Problem

In this chapter, the parking problem for a swarm of mobile robots has been studied. The robots are deployed at the nodes of an infinite grid, which has a subset of prefixed nodes marked as *parking nodes*. Each parking node p_i has a capacity of k_i which is given as input and represents the maximum number of robots a *parking node* can accommodate. As a solution to the parking problem, robots need to partition themselves into groups so that each parking node contains a number of robots which is equal to the capacity of the node. It is assumed that the number of robots in the initial configuration represents the sum of the capacities of the parking nodes. The robots are endowed with global-strong multiplicity detection capability.

In Chapter 3, we studied the gathering over meeting nodes problem in an infinite two-dimensional grid, where the objective is to gather all the robots at a unique fixed node. The parking problem can be thought of as an extension to the gathering over meeting nodes problem, where each fixed node is occupied by a number of robots equal to the capacity of the fixed node in the final configuration. The parking problem can be viewed as a variation of the partitioning problem [62], which requires the robots to divide themselves into m groups, each consisting of k robots while converging into a small area. Unlike the partitioning problem, the parking problem requires that each parking node must contain robots exactly equal to its given capacity in the final configuration. However, the capacities of the parking nodes may be different. Further, if the parking

nodes are configured initially with equal capacities, then the problem is reduced to the k-epf problem [12]. The k- epf problem is a generalized version of the embedded pattern formation problem, where each fixed point contains exactly k robots in the final configuration. In addition to the theoretical benefits, the parking nodes can also be seen as base stations or charging stations with some allowable capacities. The robots may need to recharge before they are corrupted.

6.2 Contribution

This chapter considers the parking problem over an infinite grid. The robots are deployed at the nodes of an infinite grid comprising some prefixed parking nodes. Each parking node p_i has a capacity k_i, which is the maximum number of robots it can accommodate. We assume that the number of robots n equals $\sum_{i=1}^{m} k_i$, where m is the total number of parking nodes. The robots are assumed to be anonymous, autonomous, homogeneous and oblivious. The robots are activated under a fair asynchronous scheduler. We have identified all the configurations and values of k_i for which the problem cannot be solved under this setup. A deterministic algorithm has been proposed to ensure the solvability of the remaining configurations. We investigated the efficiency of the proposed algorithm, where the efficiency was measured by the total number of movements performed by the robots.

Organization: The next section, Section 6.3, focuses on various terminologies and definitions that are relevant in understanding the problem definition. Section 6.4 discusses the partitioning of the initial configurations and the configurations for which the parking problem is unsolvable. Section 6.5 describes the algorithm proposed for solving the parking problem in infinite grids. Section 6.6 discusses the correctness of the proposed algorithm. Finally, in Section 6.7, the chapter concludes with some future directions.

6.3 Model and Definitions

1. **Parking Nodes:** The input grid graph comprises of some prefixed nodes designated as parking nodes. The parking nodes are located at the nodes of the input grid graph. Let $\mathcal{P} = \{p_1, p_2, ..., p_m\}$ denote the set of parking positions. In the

initial configuration, the parking nodes are located at the distinct nodes of the grid. A robot may be deployed at one of the parking nodes initially.

2. **Capacity of a parking node:** The *capacity* of a parking node is defined as the maximum number of robots the parking node can accommodate. A parking node is said to be *saturated* if it contains exactly the number of robots equal to its capacity. A parking node is said to be *unsaturated* if it is not saturated. Let $ct : V \to \mathbb{N} \cup \{0\}$ be defined as a function, where:

$$ct(v) = \begin{cases} 0 & \text{if } v \text{ is not a parking node} \\ \textit{capacity of the parking node} & \text{otherwise} \end{cases}$$

In the initial configuration, let k_i be the capacity of a parking node p_i, $\forall i = 1, 2, \ldots, m$.

3. **System Configuration:** $C(t) = (\mathcal{R}(t), \mathcal{P})$ denotes the *system configuration* at any time t.

4. **Symmetry of a Configuration:** In Chapter 4, we define $\lambda : V \to N$ as the function that denotes the number of robots on each node $v \in V$. In this chapter, an automorphism of a configuration denoted by $Aut((\mathrm{C(t)}, f_t, \lambda)$ is an automorphism ϕ of the input grid graph such that $ct(v) = ct(\phi(v))$ and $\lambda(v) = \lambda(\phi(v))$ for all $v \in V$.

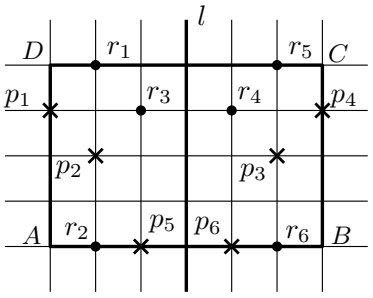

FIGURE 6.1: The configuration is symmetric with respect to l. The crosses represent parking nodes and the black circles represent robot positions.

5. **Configuration View:** In this chapter, while defining the symmetry of a configuration, we have also considered the capacities of the parking nodes. As a result, the definitions of the strings $s_i's$ are somewhat different from the definition mentioned in Chapter 3. While scanning the grid, each node v is associated to the pair

$(\lambda(v), ct(v))$ that the string encounters. In Figure 6.1, assume that the capacity of the parking nodes p_1, p_4, p_5 and p_6 are 2, respectively and the capacities of the parking nodes p_2 and p_3 are 1. It should be noted that C and D are the key corner. The lexicographic string associated with the corners C and D are $s_{CB} = s_{DA} =$ ((0,0), (0,2), (0,0), (0,0), (0,0), (1,0), (0,0), (0,1), (0,0), (1,0), (0,0), (1,0), (0,0), (0,0), (0,2), (0,0), (0,0), (0,0), (0,0), (0,0), (0,0) (1,0), (0,0), (0,0), (0,2), (1,0), (0,0), (0,1), (0,0), (1,0), (0,0), (0,2), (0,0), (0,0), (0,0)).

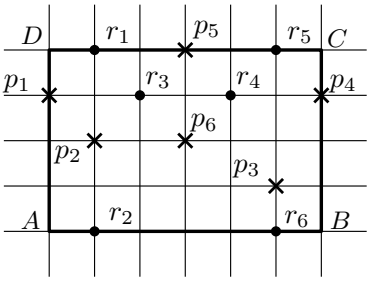

FIGURE 6.2: Example configuration indicating the case when the parking nodes are asymmetric.

6. **Symmetry of the set $\mathcal{P}$**: We may define the symmetry of the set $\mathcal{P}$ in the same way as we define the symmetry of a configuration. The smallest grid-aligned rectangle that includes all the parking nodes is defined as $M_{\mathcal{P}}$.

   In this chapter, while defining the symmetry of $\mathcal{P}$ we have also considered the capacities of the parking nodes. Each node v is associated to $ct(v)$, while scanning $\mathcal{P}$. If the parking nodes are asymmetric, a unique lexicographic largest string α_i always exists. If the parking nodes are not asymmetric, then the parking nodes are said to be symmetric. The corner with which the lexicographic largest string α_i is associated is defined as the leading corner, similar to as defined in Chapter 3. In Figure 6.2, assume that the capacity of each parking node is 1. With this assumption, $\alpha_{DA} =$ 0100000100000001010000000001001000 is the largest lexicographic string among the $\alpha_i's$. D is the leading corner. According to this definition of symmetry of the set $\mathcal{P}$, the parking nodes that are located in the symmetric positions must have equal capacities.

Definition 6.3.1. *Let $C(0)$ be any given initial configuration. A parking node p_i is said to have a higher order than the parking node p_j if it appears after p_j in the string representation α_k, associated to the leading corner k of MER. Similarly, a robot r_i has*

a higher order than r_j if it appears after r_j in the string representation α_k, associated to the leading corner k of MER.

According to this definition, if the configuration has two leading corners, then two parking nodes have the highest order exist.

6.4 Problem Definition and Impossibility Results

6.4.1 Problem Definition

Let $C(t) = ((\mathcal{R}(t), \mathcal{P})$ denote the system configuration at any time t. Parking node p_i has a capacity k_i, which is the maximum number of robots it can accommodate at any instant of time. For each parking node p_i, the capacity k_i is given as an input. The number of robots is assumed to be equal to $\sum_{i=1}^{m} k_i$, where m is the total number of parking nodes located at the nodes of an infinite grid. In an initial configuration, all the robots occupy distinct nodes of the grid. The goal of the parking problem is to transform any initial configuration into a final configuration satisfying the following properties:

- each parking node p_i is saturated, i.e., p_i contains exactly k_i robots on it.
- each robot is stationary.
- any robot taking a snapshot in the look phase at some time $t > 0$ will decide not to move.

Note that if each $k_i = 1$, the problem is reduced to the *embedded pattern formation problem.*

6.4.2 Partitioning of the initial configurations

The strategy for movement depends on the current configuration. The initial configurations can be partitioned into the following disjoint classes:

1. $\mathcal{I}_1$: The parking nodes are asymmetric (see Figure 6.2 for illustration).

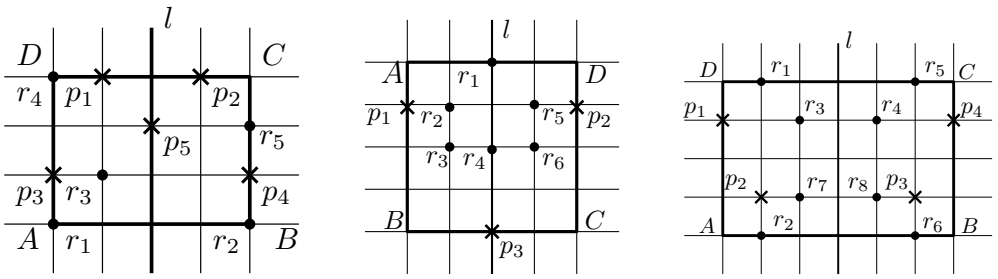

FIGURE 6.3: Examples of $\mathcal{I}_{21}$, $\mathcal{I}_{221}$ and $\mathcal{I}_{222}$ configuration.

2. $\mathcal{I}_2$: The parking nodes are symmetric with respect to a unique line of symmetry l. This class of configurations can be further partitioned into:

   (a) $\mathcal{I}_{21}$: $C(t)$ is asymmetric. In Figure 6.3(a), with the assumption that each parking node has the same capacity 1, the configuration is asymmetric while the parking nodes are symmetric with respect to l.

   (b) $\mathcal{I}_{22}$: $C(t)$ is symmetric with respect to l. This can be further partitioned into the following disjoint classes:

      i. $\mathcal{I}_{221}$: There exists at least one robot position on l. In Figure 6.3(b), with the assumption that the capacity of each parking node is 2, $C(t)$ is symmetric with respect to l and there exist robot positions r_1 and r_4 on l.

      ii. $\mathcal{I}_{222}$: There does not exist any robot position or parking nodes on l. In Figure 6.3(c), with the assumption that the capacity of each parking node is 2, $C(t)$ is symmetric with no robots or parking nodes on l.

      iii. $\mathcal{I}_{223}$: There does not exist any robot position on l, but there is at least one parking node on l. In Figure 6.4(a), with the assumption that the capacity of each parking node not at l is 1 and there exists parking node p_5 at l with capacity 2, $C(t)$ is symmetric with respect to l.

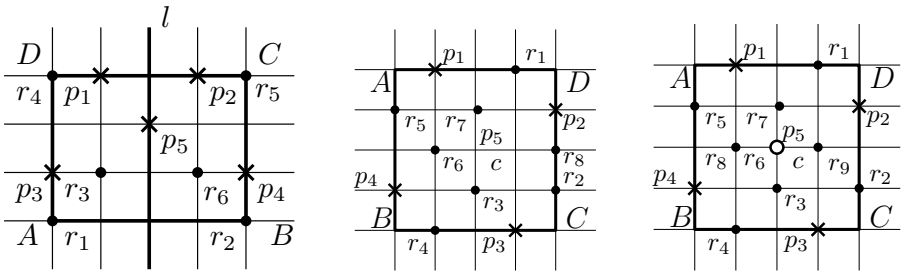

FIGURE 6.4: Examples of $\mathcal{I}_{223}$ configuration, $\mathcal{I}_{31}$ configuration and $\mathcal{I}_{321}$ configuration.

3. $\mathcal{I}_3$: The parking nodes are symmetric with respect to rotational symmetry, with c as the center of rotational symmetry. This class of configurations can be further partitioned into:

   (a) $\mathcal{I}_{31}$: $C(t)$ is asymmetric. In Figure 6.4(b), with the capacity of each parking node assumed to be 2, $C(t)$ is asymmetric; the parking nodes are symmetric with respect to rotational symmetry.

   (b) $\mathcal{I}_{32}$: $C(t)$ is symmetric with respect to c. This can be further partitioned into the following disjoint classes:

      i. $\mathcal{I}_{321}$: There exists a robot position on c. In Figure 6.4(c), with the assumption that the capacities of parking nodes p_1, p_2, p_3 and p_4 equal 1 and the capacity of the parking node p_5 equals 5, $C(t)$ is symmetric with respect to rotational symmetry. The robot r_6 is at the parking node p_5.

      ii. $\mathcal{I}_{322}$: There does not exist any robot position or parking node on c. In Figure 6.5(a), with the assumption that the capacity of each parking node is 1, $C(t)$ is symmetric with respect to rotational symmetry, without any robot or parking node on c.

      iii. $\mathcal{I}_{323}$: There exists a parking node on c, but no robot lies on c. In Figure 6.5(b), with the assumption that the capacities of the parking nodes p_1, p_2, p_3 and p_4 equals 1 and the capacity of the parking node p_5 equals 4, $C(t)$ is symmetric with respect to rotational symmetry with a parking node p_5 on c.

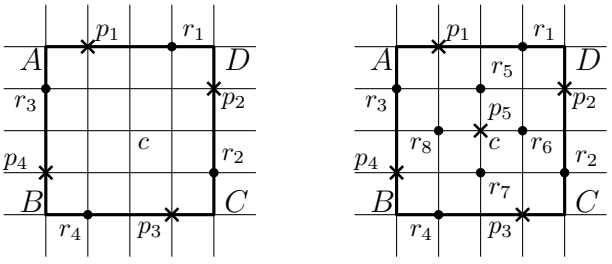

FIGURE 6.5: Examples of $\mathcal{I}_{322}$ and $\mathcal{I}_{323}$ configuration.

It can be checked that the classes defined with respect to the partitioning are exhaustive. Additionally, the partitioning can be checked to ensure that the subclasses are exhaustive. We assume that if the parking nodes admit rotational symmetry, then l and l' are perpendicular lines passing through c. Note that these lines divide the grid into

four quadrants. Consider the open quadrants, i.e., the quadrants excluding the nodes on $l \cup l'$. If there are more than two lines of symmetry, the two lines that are perpendicular to each other and do not pass through any corner of $M_{\mathcal{P}}$ are selected and considered as l and l'.

6.4.3 Impossibility Results

In this subsection, we define all those initial configurations and the values of the capacities of the parking nodes for which the parking problem is unsolvable. We next state the following impossibility results.

Lemma 6.4.1. *Let $\mathcal{A}$ be any algorithm for the parking problem in infinite grids. If there exists an execution of $\mathcal{A}$ such that the configuration $C(t)$ contains a robot multiplicity, then $\mathcal{A}$ cannot solve the parking problem.*

Proof. In the initial configuration, the capacities of the parking nodes are given as input to the robots. We have assumed that the capacity of a parking node p_i is k_i, where $k_i \geq 1$ and $i = 1(m)$. So, without loss of generality, we assume that the capacity of each parking node is 1, i.e., in the final configuration, there must be exactly one robot at each parking node. Moreover, we assume that the scheduler is fully-synchronous. Suppose at time $t > 0$, a robot multiplicity is formed at one of the nodes. The robots at the multiplicity have the same local view of the configuration. If the adversary forces both the robots composing the multiplicity to perform the same move, the multiplicity in $C(t)$ is maintained for all $C(t')$, for $t' > t$. As a result, the robots in the multiplicity cannot reach different parking nodes. This proves that under the execution of $\mathcal{A}$, the robots cannot reach the final configuration. □

This lemma ensures that during the execution of any algorithm that solves the parking problem, the robots must perform a collision-less movement at all stages of the algorithm. Suppose the robots are oblivious and not endowed with global-strong multiplicity detection capability. In that case, they cannot detect whether exactly the k_i number of robots reaches the parking node p_i. We formalize the result in the following lemma:

Lemma 6.4.2. *Without the global-strong multiplicity detection capability of the robots, the parking problem is unsolvable.*

Lemma 6.4.3. *If the initial configuration $C(0) \in \mathcal{I}_{223}$ is such that the capacity of a parking node on l is an odd integer. Then the parking problem is unsolvable.*

Proof. Assume that there exists an algorithm $\mathcal{A}$ that solves the parking problem starting from any arbitrary initial configuration in $\mathcal{I}_{223}$ and the capacity of the parking node p on l is an odd integer. Without loss of generality, we assume that the capacity of p is $2k+1$, where k is a positive integer. Consider the scheduler to be semi-synchronous with the additional constraint that a robot and its symmetric image with respect to l are activated and perform the Look-Compute-Move cycle simultaneously. In that case, a robot r and its symmetric image $\phi(r)$ have the same local view of the configuration and they execute the same deterministic algorithm $\mathcal{A}$. Assume that at time $t > 0$, there exists a $2k$ number of robots on p. Since, the capacity of p is $2k+1$, at time $t' > t$, a robot r must start moving towards p. As the configuration is symmetric, there exists at least one execution of $\mathcal{A}$ out of different execution paths, where any move that r performs according to $\mathcal{A}$ would result in a situation where $\phi(r)$ performs the same move. These movements of the robots ensure that at any moment of time, the configuration remains symmetric. Since there is no robot position on l in the initial setup and all robots move in pairs, moving a robot r to l would also move $\phi(r)$ to the same node on l. While the robots r and $\phi(r)$ move towards p, a multiplicity node is created at p, which cannot be separated by any deterministic algorithm. Hence, the parking problem is unsolvable, according to Lemma 6.4.1. □

It follows from Lemma 6.4.3 that if $C(t)$ admits multiple lines of symmetry and if there exists a parking node on a line of symmetry with odd capacity, then also the problem is unsolvable.

Corollary 6.4.4. *If the initial configuration $C(0) \in \mathcal{I}_{323}$, then the parking problem is unsolvable if the capacity of the parking node at c is neither a multiple of 4 nor 2, depending on whether the angle of rotation is either $90°$ or $180°$.*

Let $\mathcal{U}$ be the set of all configurations that are unsolvable according to Lemma 6.4.3 and Corollary 6.4.4.

6.5 Algorithm

This section proposes a deterministic distributed algorithm for solving the parking problem in infinite grids. A sufficient condition for an initial configuration to be unsolvable, i.e., when the parking problem cannot be deterministically solved, is provided by the impossibility results given in Subsection 6.4.3. The parking problem is solved using a deterministic distributed algorithm in this section for all initial configurations except those indicated in Lemma 6.4.3 and Corollary 6.4.4. The fundamental strategy of the proposed algorithm is to identify a specific target parking node and permit a number of robots to move towards it, where the number of robots is equal to the parking node's capacity. The target parking node is selected in a sequential manner and the procedure executes unless each parking node becomes saturated. The proposed algorithm mainly consists of the following phases: *Guard Selection and Placement* phase, *Target Parking Node Selection* phase, *Candidate Robot Selection* phase, *Guard Movement* phase and *Symmetry Breaking* phase. Suppose the parking nodes are symmetric and the configuration is asymmetric. In that case, a guard is selected and moved in the Guard Selection and Placement phase in such a way that the configuration remains asymmetric during the execution of the algorithm. The robots identify the current configuration and determine whether a unique parking node could be selected for parking. If the configuration is symmetric, two parking nodes may be chosen for parking at any given time during the Target Parking Node Selection phase. The number of robots equaling the capacity of the target parking node is selected in the Candidate Robot Selection phase and moves toward the target parking node in a sequential manner. When the parking node(s) with the highest orders become saturated, the next parking node is selected, which is unsaturated and has the highest order among all the unsaturated parking nodes. The process continues until all parking nodes are saturated. During the Guard Movement phase, the guard moves toward its respective target parking node. The symmetric configurations that can be changed into asymmetric configurations are taken into consideration during the Symmetry Breaking phase. A unique robot is selected and allowed to move toward an adjacent node such that the configuration becomes asymmetric. In the subsequent subsections, we observe that if the configuration is asymmetric or symmetric with a parking node on $l \cup \{c\}$, the robots can always select a unique parking node in order to begin the parking formation process.

6.5.1 Ordering of the parking nodes

In this subsection, we first consider all those configurations where the parking nodes of the configurations can be ordered uniquely. This ordering is necessary to identify a unique parking node, which will be selected by the robots in order to initialize the parking formation. So, first, consider the case when the parking nodes are asymmetric. According to the definition of the symmetry of the set $\mathcal{P}$, there exists a unique lexicographic string α_i. As a result, there exists a unique leading corner. The ordering $\mathcal{O}_1$ of the parking nodes, in case the parking nodes are asymmetric, is defined similarly as in Chapter 3. The ordering $\mathcal{O}_2$ is also defined similarly as in Chapter 3. Assume that $(p_1, p_2, \ldots, p_m)$ is the ordering $\mathcal{O}_1$ of the parking nodes. Similarly, let $\mathcal{O}_2$ be the ordering $(p_1, p_2, \ldots, p_z)$, where z denotes the number of parking nodes on l. Formally,

Observation 4. *If the parking nodes are asymmetric, then they can be ordered.*

Observation 5. *If the parking nodes are symmetric with respect to a unique line of symmetry l, then the parking nodes on l are orderable.*

6.5.2 Guard Selection and Placement

In this subsection, we describe the Guard Selection and Placement phase. This phase is similar but slight different to the phase description defined in Chapter 3. In this chapter, we have introduced logical predicates to describe this phase.

Consider the case when the parking nodes are symmetric, but the configuration is asymmetric, i.e., $C(t) \in \mathcal{I}_{21} \cup \mathcal{I}_{31}$. In this phase, a unique robot is selected as a guard and placed in such a way that the configuration remains asymmetric during the execution of the algorithm. Since the initial configuration is asymmetric, a unique largest lexicographic string exists among the $s_i's$ associated to the corners. As a result, a unique key corner and a robot with the maximum configuration view exist. The following notations are used in describing the Guard Selection and Placement phase:

- Condition C_1: There exists at least one robot position outside the rectangle $M_{\mathcal{P}}$.
- Condition C_2: Each robot is inside the rectangle $M_{\mathcal{P}}$.
- Condition C_3: There exists a unique farthest robot from $l \cup \{c\}$.

Guard Selection and Placement		
Initial Configuration ($\mathcal{I}_{21} \cup \mathcal{I}_{31}$)	**Guard**	**Position of the guard**
$C_1 \wedge C_3$	The unique robot farthest from $l \cup \{c\}$	Current position of the guard
$C_1 \wedge \neg C_3$	The unique robot furthest from $l \cup \{c\}$ and having the maximum configuration view among all the furthest robots	The unique robot moves towards an adjacent node away from $l \cup \{c\}$
$C_2 \wedge C_3$	The unique robot furthest from $l \cup \{c\}$	The guard continues its movement away from $l \cup \{c\}$, unless the condition C_1 becomes true
$C_2 \wedge \neg C_3$	The unique robot furthest from $l \cup \{c\}$ and having the maximum configuration view among all the furthest robots	The guard continues its movement away from $l \cup \{c\}$ until the condition C_1 becomes true

TABLE 6.1: Guard Selection and Placement

Note that the condition C_1 = true implies that $d_2 > d_1$ and the condition C_2 = true implies that $d_2 \leq d_1$. Depending on the class of configurations to which $C(t)$ belongs, the phase is described in Table 6.1.

In Figure 6.6 (a), D is the key corner. r_5, r_6 and r_7 are the farthest robots from l. However, r_5 is selected as the guard as it has the highest order among all the farthest robots. In Figure 6.6 (b), r_5 moves towards an adjacent node and it becomes the unique farthest robot from l. While the guard is selected and placed, as the guard is the unique farthest robot from $l \cup \{c\}$, it does not have any symmetric image with respect to $l \cup \{c\}$. In case $C(0) \in \mathcal{I}_{31}$, and the final position of the guard is either on l or l', then the guard moves towards an adjacent node away from l or l'. As a result of this movement, the guard will be positioned in one of the quadrants after this phase.

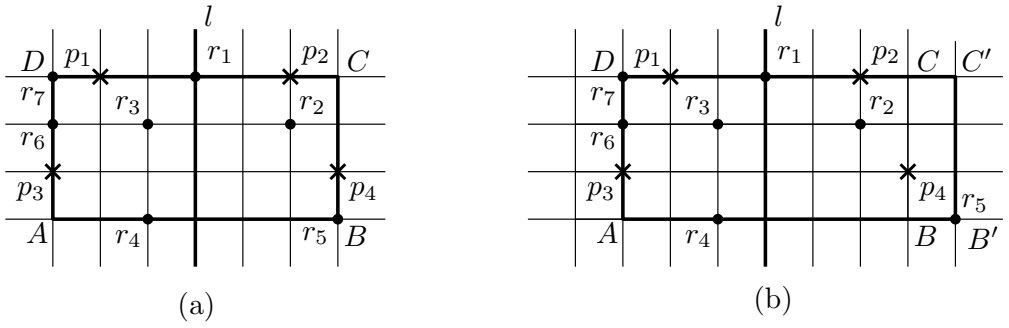

FIGURE 6.6: Example showing the Guard Selection and Placement phase.

6.5.3 Half-planes and Quadrants

In this subsection, we consider all asymmetric configurations where the parking nodes are symmetric with respect to $l \cup \{c\}$, i.e., $C(0) \in \mathcal{I}_{21} \cup \mathcal{I}_{31}$. First, consider the case when $C(0) \in \mathcal{I}_{21}$. The line of symmetry l divides the entire grid into two half-planes. We consider the open half-planes, i.e., the half-planes excluding the nodes on l. Let H_1 and H_2 denote the two half-planes delimited by l. We first introduce some notations and definitions that are relevant in understanding the concepts of demarcations of the half-planes.

1. $UP(t)$- Number of parking nodes which are unsaturated at time t.
2. *Deficit Measure of a parking node p_i ($Df_{p_i}(t)$):* The deficit measure $Df_{p_i}(t)$ of a parking node p_i at time t is defined as the deficit in the number of robots needed to have exactly k_i robots on p_i.
3. $K_1 = \sum_{p_i \in H_1} Df_{p_i}(t)$ denotes the total deficit in order to have exactly $\sum_{p_i \in H_1} k_i$ number of robots at the parking nodes belonging to the half-plane H_1.
4. $K_2 = \sum_{p_i \in H_2} Df_{p_i}(t)$ denotes the total deficit in order to have exactly $\sum_{p_i \in H_2} k_i$ number of robots at the parking nodes belonging to the half-plane H_2.

Definition 6.5.1. *Let $C(t)$ be any initial configuration belonging to the set $\mathcal{I}_{21}$. $C(t)$ is said to be unbalanced if the two half-planes delimited by l contain an unequal number of robots. Otherwise, the configuration is said to be balanced.*

We next consider the following conditions.

1. Condition C_4- There exists a unique half-plane that contains the minimum number of unsaturated parking nodes.
2. Condition C_5- $K_1 \neq K_2$
3. Condition C_6- The configuration is unbalanced.
4. Condition C_7- The configuration is balanced and $\mathcal{R} \cap l \neq \emptyset$.
5. Condition C_8- The configuration is balanced and $\mathcal{R} \cap l = \emptyset$.

Demarcation of the half-planes for fixing the target	
Initial Configuration ($\mathcal{I}_{21}$)	**$\mathcal{H}^+$**
C_4	The unique half-plane which contains the minimum number of unsaturated parked nodes
$\neg\, C_4 \wedge C_5 \wedge K_1 < K_2$	H_1
$\neg\, C_4 \wedge C_5 \wedge K_2 < K_1$	H_2
$\neg\, C_4 \wedge \neg C_5 \wedge C_6$	The unique half-plane with the maximum number of robot positions
$\neg\, C_4 \wedge \neg C_5 \wedge \neg\, C_6 \wedge C_7$	The northernmost robot on l move towards an adjacent node away from l. The unique half-plane with the maximum number of robot positions is defined as $\mathcal{H}^+$
$\neg\, C_4 \wedge \neg C_5 \wedge \neg\, C_6 \wedge \neg\, C_7 \wedge C_8$	The unique half-plane not containing the guard

TABLE 6.2: Demarcation of the half-planes

The half-plane $\mathcal{H}_{target}$ or $\mathcal{H}^+$ is defined according to Table 6.2, where the parking at the parking nodes initializes. The other half-plane is denoted by $\mathcal{H}^-$. In Figure 6.7 (a), assume that the capacities of the parking nodes p_1, p_2, p_3 and p_4 are 1, 2, 1 and 3, respectively. Therefore, $K_1 = 2$ and $K_2 = 5$. The half-plane with lesser K_i value (K_1) is selected as $\mathcal{H}^+$. In Figure 6.7 (b), assume that the capacities of the parking nodes p_1, p_2, p_3 and p_4 are 3, 2, 2 and 3, respectively. Each of the half-planes contains the same number of robots. Therefore, the configuration is balanced. The half-plane not containing the guard r_5 is defined as $\mathcal{H}^+$.

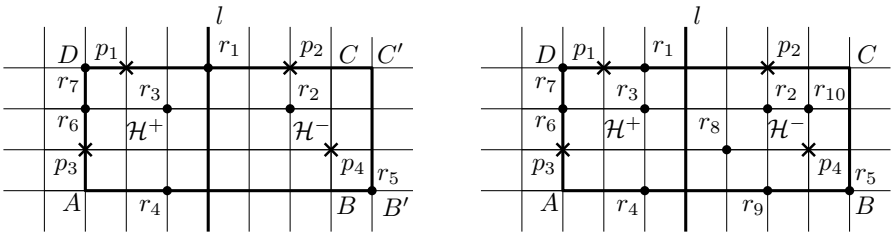

FIGURE 6.7: Example configuration showing demarcations of half-planes.

Next, assume that the parking nodes are symmetric with respect to rotational symmetry. Consider the lines l and l' that pass through the center of MER. These lines divide the entire rectangle into four quadrants. Let Q_i be the quadrants that are defined by the lines l and l', where i ranges from 1 to 4. We consider the open quadrants, i.e., the quadrants exclude the nodes belonging to l and l'. Let $L_j = \sum_{p_i \in Q_j} Df_{p_i}(t)$ denote the total deficit in order to have exactly $\sum_{p_i \in Q_j} k_i$ number of robots at the parking nodes belonging to the quadrant Q_j, where j ranges from 1 to 4.

Definition 6.5.2. *Let $C(t)$ be any initial configuration belonging to the set $\mathcal{I}_{31}$. Consider the quadrants with the minimum L_j value. $C(t)$ is said to be unbalanced if there exists a unique quadrant among such quadrant with minimum L_j value, with the maximum number of robots. Otherwise, the configuration is said to be balanced.*

According to the Definition 6.5.2, a configuration is said to be *balanced* if it contains at least two quadrants with minimum L_j value and with the maximum number of robot positions.

1. Condition C_9- There exists a unique quadrant that contains the minimum number of unsaturated nodes.
2. Condition C_{10}- There exists a unique quadrant with the minimum L_j value, for $j = \{1, 2, 3, 4\}$.
3. Condition C_{11}- The configuration is unbalanced.
4. Condition C_{12}- The configuration is balanced and $(\mathcal{R} \cap l \neq \emptyset \vee \mathcal{R} \cap l' \neq \emptyset)$.
5. Condition C_{13}- The configuration is balanced and $\mathcal{R} \cap l = \emptyset$, $\mathcal{R} \cap l' \neq \emptyset$.

Note that $\neg C_{10}$ implies there exist at least two quadrants with the minimum L_j value. The quadrant $\mathcal{Q}_{target}$ or $\mathcal{Q}^{++}$, where the parking is initialized, is defined according to Table 6.3, we consider t. The quadrants adjacent to $\mathcal{Q}^{++}$ with respect to l and l' are defined as $\mathcal{Q}^{-+}$ and $\mathcal{Q}^{+-}$ respectively. Similarly, the quadrant non-adjacent to $\mathcal{Q}^{++}$ is defined as $\mathcal{Q}^{--}$. In Figure 6.8 (a), assume that the capacities of each of the parking nodes are 2. As a result, each L_j equals 2. The quadrant with more number of robots is defined as $\mathcal{Q}^{++}$. Consider the case when there are exactly two adjacent quadrants with the minimum L_j value and with the maximum number of robots. Suppose one of such quadrants contains the guard. In that case, the quadrant not containing the guard is defined as $\mathcal{Q}^{++}$. In Figure 6.8 (b), assume that the capacities of the parking nodes p_1, p_2 and p_3 are 2 and the capacity of the parking nodes p_4 is 3. As a result, $L_1 = L_2 = L_3 = 2$ has the minimum L_j value, where Q_j is assumed as the quadrant containing the parking node p_j. However, the quadrants Q_2 and Q_3 contain the maximum number of robot positions. Hence, the configuration is balanced. $\mathcal{Q}^{++}$ is defined as the quadrant not containing the guard r_8.

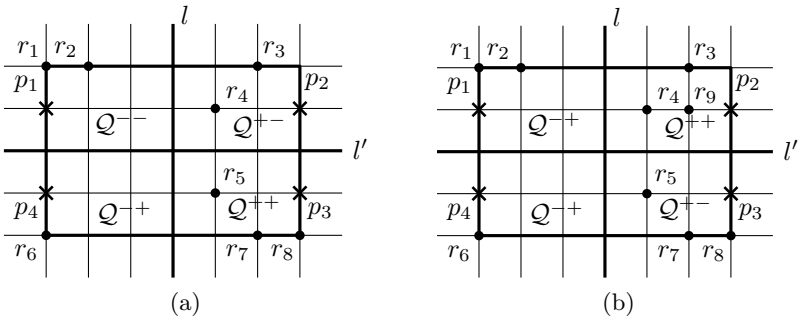

FIGURE 6.8: Example configuration showing demarcations of quadrants.

Demarcation of the quadrants for fixing the target	
Initial Configuration $C(0)$	**$\mathcal{Q}_{target}$**
C_9	The unique quadrant which contains the minimum number of unsaturated parking nodes
$\neg\, C_9 \wedge C_{10} \wedge L_j$ is minimum	Q_j
$\neg\, C_9 \wedge \neg C_{10} \wedge C_{11}$	The unique quadrant with the maximum number of robot positions
$\neg\, C_9 \wedge \neg C_{10} \wedge \neg\, C_{11} \wedge C_{12}$	The robot on l or l' with the highest order moves along $l \cup l'$ and when it is one move away from a quadrant containing maximum number of robot positions, then it moves towards an adjacent node away from $l \cup l'$. The unique quadrant with the maximum number of robot positions is $\mathcal{Q}_{target}$
$\neg\, C_9 \wedge \neg C_{10} \wedge \neg\, C_{11} \wedge \neg\, C_{12} \wedge C_{13}$	In case there are two adjacent quadrants containing the minimum L_j value and maximum number of robot positions, the unique quadrant that does not contain the guard or the quadrant that is not adjacent to the quadrant containing the guard if more than two such quadrants exist

TABLE 6.3: Demarcation of the quadrants

Definition 6.5.3. *Consider the half-lines starting from c and along the grid-lines. We denote the wedge boundaries of the quadrants delimited by the lines l and l' by $\mathcal{B}_i$, where $i = \{1, 2, 3, 4\}$.*

6.5.4 Target Parking Nodes Selection

In this phase, the target parking node for the parking problem is selected. Depending on the following classes of configurations, the phase is described in a tabular form in Table 6.4. Let p_{guard} be the closest parking node from the guard. If multiple such parking nodes exist, then the parking node closest to the guard and having maximum order is selected as p_{guard}. We first assume that the target parking nodes are selected

in $\mathcal{P} \setminus \{p_{guard}\}$. We consider the following conditions that are relevant in understanding this phase.

1. C_{14}- There exist an unsaturated parking node on l.
2. C_{15}- There exists an unsaturated parking node on c.
3. C_{16}- All the parking nodes belonging to $\mathcal{H}^+$ are saturated.
4. C'_{16}- All the parking nodes belonging to $\mathcal{H}^-$ are saturated.
5. C_{17}- All the parking nodes belonging to $\mathcal{Q}^{++}$ are saturated.
6. C_{18}- All the parking nodes at the wedge boundaries corresponding to the quadrant $\mathcal{Q}^{++}$ are saturated.
7. C_{19}- All the parking nodes belonging to the quadrant $\mathcal{Q}^{-+}$ are saturated.
8. C_{20}- All the parking nodes at the wedge boundaries corresponding to the quadrant $\mathcal{Q}^{-+}$ are saturated.
9. C_{21}- All the parking nodes belonging to the quadrant $\mathcal{Q}^{+-}$ are saturated.
10. C_{22}- All the parking nodes at the wedge boundaries corresponding to the quadrant $\mathcal{Q}^{+-}$ are saturated.
11. C_{23}- All the parking nodes belonging to $\mathcal{Q}^{--}$ are saturated.

While all the parking nodes belonging to the set $\mathcal{P} \setminus \{p_{guard}\}$ become saturated, p_{guard} becomes the target parking node. Note that $\neg C_{14}$ implies that the parking nodes are symmetric with respect to l and there either does not exist any parking node on l or each parking node on l is saturated. Similarly, $\neg C_{15}$ implies that the parking nodes are symmetric with respect to rotational symmetry and there either does not exist any parking node on c or the parking node on c is saturated. In Figure 6.7, A and B are the leading corners. p_1 is the parking node in $\mathcal{H}^+$ which has the highest order. The target parking nodes are selected in the order (p_2, p_1, p_4, p_3). Similarly, in Figure 6.8, the target parking nodes are selected in the order (p_3, p_2, p_4, p_1).

Target Parking Node Selection	
Initial Configuration $C(0)$	**Target Parking Node**
$\mathcal{I}_1$	The parking node which is unsaturated and has the highest order with respect to $\mathcal{O}_1$
$\mathcal{I}_2 \wedge C_{14}$	The parking node on l which is unsaturated and has the highest order with respect to $\mathcal{O}_2$
$\mathcal{I}_{21} \wedge \neg\, C_{14} \wedge \neg\, C_{16}$	The parking node, which is unsaturated and has the highest order in $\mathcal{H}^+$ among all the unsaturated nodes in $\mathcal{H}^+$
$\mathcal{I}_{21} \wedge \neg\, C_{14} \wedge C_{16} \wedge \neg C'_{16}$	The parking node, which is unsaturated and has the highest order in $\mathcal{H}^-$ among all the unsaturated nodes in $\mathcal{H}^-$
$\mathcal{I}_{22} \wedge \neg\, C_{14}$	The two parking nodes that have the highest order among all the unsaturated parking nodes and lying on two different half-planes
$\mathcal{I}_3 \wedge C_{15}$	The parking node on c
$\mathcal{I}_{31} \wedge \neg C_{15} \wedge \neg C_{17}$	The parking node in $\mathcal{Q}^{++}$ which is unsaturated and has the highest order among all the unsaturated parking nodes in $\mathcal{Q}^{++}$
$\mathcal{I}_{31} \wedge \neg C_{15} \wedge C_{17} \wedge \neg C_{18}$	The parking node at the wedge boundaries corresponding to $\mathcal{Q}^{++}$ which is unsaturated and has the highest order among such nodes
$\mathcal{I}_{31} \wedge \neg C_{15} \wedge C_{17} \wedge C_{18} \wedge \neg C_{19}$	The parking node in $\mathcal{Q}^{-+}$ which is unsaturated and has the highest order among all the unsaturated parking nodes in $\mathcal{Q}^{-+}$
$\mathcal{I}_{31} \wedge \neg C_{15} \wedge C_{17} \wedge C_{18} \wedge C_{19} \wedge \neg C_{20}$	The parking node at the wedge boundaries corresponding to $\mathcal{Q}^{-+}$ which is unsaturated and has the highest order among such unsaturated parking nodes
$\mathcal{I}_{31} \wedge \neg C_{15} \wedge C_{17} \wedge C_{18} \wedge C_{19} \wedge C_{20} \wedge \neg C_{21}$	The parking node in $\mathcal{Q}^{+-}$ which is unsaturated and has the highest order among all the unsaturated parking nodes in $\mathcal{Q}^{+-}$
$\mathcal{I}_{31} \wedge \neg C_{15} \wedge C_{17} \wedge C_{18} \wedge C_{19} \wedge C_{20} \wedge C_{21} \wedge \neg C_{22}$	The parking node at the wedge boundaries corresponding to $\mathcal{Q}^{+-}$ which is unsaturated and has the highest order among such unsaturated parking nodes
$\mathcal{I}_{31} \wedge \neg C_{15} \wedge C_{17} \wedge C_{18} \wedge C_{19} \wedge C_{20} \wedge C_{21} \wedge C_{22} \wedge \neg C_{23}$	The parking node in $\mathcal{Q}^{--}$ which is unsaturated and has the highest order among all the unsaturated parking nodes in $\mathcal{Q}^{--}$
$\mathcal{I}_{32} \wedge \neg C_{15}$	The two or four parking nodes which have the highest order among all the unsaturated nodes and lying on different quadrants or on the wedge boundaries

Table 6.4: Target Parking Nodes Selection

6.5.5 Candidate Robot Selection Phase

In view of Lemma 5.4.2, while a robot moves towards a parking node, it must ensure collision-free movement. Otherwise, the problem becomes unsolvable. As a result, a robot will move toward its target only when it has a path toward that target that does not contain any other robot positions. Therefore, we first consider the following definition.

Definition 6.5.4. *A path from a robot to a parking node is said to be free if it does not contain any other robot positions.*

A robot would move toward its target only when it has a free path toward it. In this phase, the *candidate robot* is selected and allowed to move toward the target parking node. Let $p \neq p_{guard}$ be the target parking node selected in the Target Parking Node Selection phase. Depending on the different classes of configurations, the following cases are to be considered.

1. $C(t)$ is asymmetric. As a result, the robots are orderable. The robot that does not lie on any saturated parking node and has the shortest free path to p is selected as the candidate robot. If multiple such robots exist, the one with the highest order among such robots is selected as the candidate robot. Once it starts moving toward its destination, it will become the unique candidate robot.

2. $C(t)$ is symmetric with respect to a single line of symmetry l and p is on l. If at least one robot exists on l, then the Symmetry Breaking phase is executed, which is discussed in the next subsection. As a result, assume that there is no robot position on l. The two closest robots, which do not lie on any saturated parking node and have shortest free paths towards p, are selected as the candidates for p. Note that since the configuration is symmetric, the two candidate robots are selected on different half-planes. If there are multiple such robots, the ties are broken by considering the robots that lie on different half-planes and have the highest order among all such robots.

3. $C(t)$ is symmetric with respect to a single line of symmetry and p is on the half-planes. The robot that does not lie on any saturated parking node and has a

shortest free path toward p is selected as the candidate robot. Note that such candidates are selected in both half-planes.

4. $C(t)$ is symmetric with respect to rotational symmetry and p is on c. The robots that are closest to p are selected as candidate robots. In this case, depending on whether the angle of rotational symmetry is 180° or 90°, two or four robots are chosen as candidates.

5. $C(t)$ is symmetric with respect to rotational symmetry, and the target parking node p is located on a quadrant as well as on the wedge boundaries. First, assume that the target parking node lies on a quadrant. The robot that does not lie on any saturated parking node and has a shortest free path toward p is selected as the candidate robot. It should be noted that such candidates are chosen from each of the four quadrants, for each target parking node. Next, assume that the target parking node is on a wedge boundary. The robot(s) not lying on any saturated parking node and having a shortest free path towards the target is (are) selected as candidate robot(s).

Next, assume that p_{guard} is the target parking node. The candidates are selected as the robot which has shortest free path towards p_{guard}. Finally, the guard move towards p_{guard}. By the choice of the target parking node p, there always exists a half-line starting from p, which does not contain any robot position. As a result, a free path always exists between the candidate robot and p.

6.5.6 Guard Movement

Assume the case when the parking nodes are symmetric and the configuration is asymmetric. The guard is selected and placed in the Guard Selection and Placement phase. In the Guard Movement phase, the guard moves toward its respective destination and the parking process is terminated. The guard moves only when it finds that, except for one, all the parking nodes have become saturated. It moves towards its destination p in a free path. By choice of p, there always exists one half-line from p in the grid, which does not contain any robot position. As a result, such a free path always exists. The guard moves towards its destination and each parking node become saturated, transforming the configuration into a final configuration.

6.5.7 Symmetry Breaking Phase

In this phase, the symmetric configurations that can be transformed into asymmetric configurations are considered. The phase description is similar to the phase description of Symmetry Breaking mentioned in Chapter 5. The list of all configurations that are considered in this phase is as follows:

1. Configurations admitting a single line of symmetry l with at least one robot position on l, i.e., $C(0) \in \mathcal{I}_{221}$.

2. Configurations admitting rotational symmetry with a robot on the center of rotational symmetry, i.e., $C(0) \in \mathcal{I}_{331}$.

Let r be the robot on l with the maximum order, in case $C(0) \in \mathcal{I}_{221}$. That is, r appears after every other robot that is on l in the string directions associated to the leading corner(s). In case, $C(0) \in \mathcal{I}_{331}$, let r be the robot on c. It should be noted that while the robot r on $l \cup \{c\}$ moves towards an adjacent node away from $l \cup \{c\}$, the configuration transforms into an asymmetric configuration. However, it might be possible that the neighbors of r contain robot positions. The movement of the robot towards an adjacent node might create a robot multiplicity node, resulting in an unsolvable configuration according to Lemma 5.4.2. As a result, there must be free space available around r so that it can move toward an adjacent node. Due to the asynchronous behavior of the scheduler, there might be a possible pending move while the adjacent robots of r move towards an adjacent node away from $l \cup \{c\}$. Recall the definition of the function $h_t : V \to \{0, 1\}$ at any fixed time t. Consider the nearly rotational and nearly reflective configurations defined in Chapter 5. The procedure $AllowtoMove()$ transforms the configuration into an asymmetric configuration.

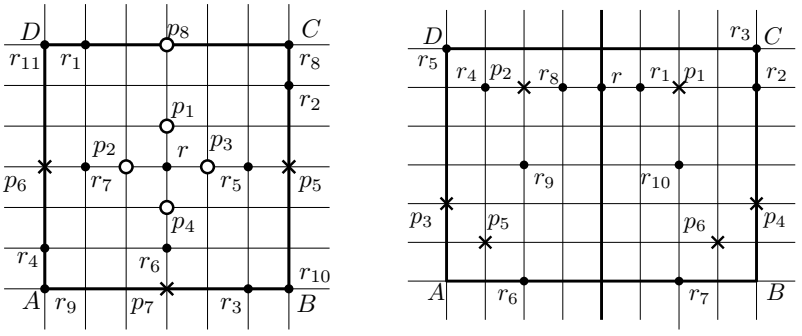

FIGURE 6.9: (a) *Nearly rotational configuration.* (b) *Nearly reflective configuration.*

In Figure 6.9(a), the *strings generated by the robot* r are given by $\{110, 110, 110, 101\}$. The circles at p_1, p_2, p_3, p_4 and p_8 denote parking nodes with robot positions on them. In Figure 6.9(b), the strings generated by the robot r and which terminate away from l are given by $\{1010, 1001\}$.

6.5.8 Parking()

This section describes the main algorithm $Parking()$, which solves the parking problem in infinite grids. The algorithm works according to the different classes of configurations. Depending on the different classes of configurations, the following cases are to be considered.

1. $\mathcal{I}_1$: In this case, the parking nodes are asymmetric. There exists a unique ordering $\mathcal{O}_1$ of the parking nodes. Since the ordering $\mathcal{O}_1$ depends on the position of the fixed parking nodes, it remains invariant during the movement of the robots. The target parking node p is selected in the Target Parking Node Selection phase, which has the highest order among all the unsaturated parking nodes. A candidate robot is selected in the Candidate Robot Selection phase, which has a shortest free path toward the target. The candidate robot moves towards the target parking node in a free path. The number of candidate robots that are selected in the Candidate Robot Selection phase for each target parking node equals its capacity. While p becomes saturated, the next target parking node is selected as the unsaturated parking node with the highest order in $\mathcal{O}_1$. The target parking nodes are selected sequentially according to the ordering $\mathcal{O}_1$, and the process continues unless each parking node becomes saturated.

2. $\mathcal{I}_2$: In this case, the parking nodes are symmetric with respect to a single line of symmetry l. However, $C(t)$ may be either asymmetric or symmetric with respect to l. First, assume the case when $C(t)$ is asymmetric, i.e., $C(0) \in \mathcal{I}_{21}$. A guard is selected in the Guard Selection and Placement phase. The configuration remains asymmetric as the guard moves towards its destination because the guard has no symmetric image with respect to l. There may or may not exist parking nodes on l. First, suppose there is at least one unsaturated parking node on l. This implies that the parking nodes at l can be ordered according to $\mathcal{O}_2$. The target parking

node p is selected on l, which has the highest order among all the unsaturated parking nodes on l. In the Candidate Robot Selection phase, a candidate robot is selected and moves in a free path towards p. The target parking nodes are selected sequentially according to the ordering $\mathcal{O}_2$ and the process continues unless each parking node on l becomes saturated.

When all the parking nodes on l become saturated, the target parking node is selected in $\mathcal{H}^+$ in the Target Parking Node Selection phase. We have proved in the Correctness section that unless all the parking nodes in $\mathcal{H}^+$ become saturated, it remains invariant. While all the parking nodes in $\mathcal{H}^+$ become saturated, the target parking node is selected in $\mathcal{H}^-$, which is unsaturated and has the highest order among all unsaturated parking nodes. Finally, the guard moves toward its destination in the Guard Movement phase only when it finds that each parking node except for one becomes saturated.

Next, assume the case when the configuration $C(t)$ is symmetric with respect to l. If $C(t) \in \mathcal{I}_{221}$, at least one robot position exists on l. The procedure $AllowtoMove()$ transforms the configuration into an asymmetric configuration. The procedure executes by allowing a robot on l to move toward an adjacent node, only when there is a free space available around r. This results in transforming the configuration into an asymmetric configuration. The rest of the procedure proceeds similarly as in the case when configuration $C(0) \in \mathcal{I}_{21}$. Next, assume the case when $C(0) \in \mathcal{I}_{222}$. The target parking nodes are selected as the unsaturated parking nodes with the highest orders. In this case, two parking nodes are selected from the two different half-planes, one from each half-plane. The candidates are selected in different half-planes and allowed to move towards their respective targets. The candidates either move synchronously or there may be a possible pending move due to the asynchronous behavior of the scheduler. If the configuration transforms into an asymmetric configuration, then the algorithm proceeds similarly as in $\mathcal{I}_{21}$. Otherwise, if the configuration regains its symmetry, then the configuration remains in $\mathcal{I}_{222}$. The algorithm proceeds unless each parking nodes become saturated. Finally, consider the case when $C(0) \in \mathcal{I}_{223}$. Note that the configuration of each parking node on l must be an even integer. Otherwise, the parking problem is unsolvable, according to Lemma 6.4.3. If an unsaturated parking node exists on l, then the target parking node is selected on l, which has the highest

order among all the unsaturated parking nodes on l. The two candidate robots selected in the Candidate Robot Selection phase move towards the target parking node either synchronously or there may be a possible pending move due to the asynchronous behavior of the scheduler. While all the parking nodes on l become saturated, the target parking nodes are selected in the two different half-planes. The rest of the procedure proceeds similarly as in the case when $C(0) \in \mathcal{I}_{222}$.

3. $\mathcal{I}_3$: In this case, the parking nodes are symmetric with respect to rotational symmetry. However, $C(t)$ may be either asymmetric or symmetric with respect to rotational symmetry. First assume the case when $C(t)$ is asymmetric, i.e., $C(0) \in \mathcal{I}_{31}$. A guard is selected and placed in the Guard Selection and Placement phase, which ensures that the configuration remains asymmetric during the execution of the algorithm. First, assume the case when there exists a parking node p on c. p is selected as the target parking node. A candidate robot is selected in the Candidate Robot Selection phase and allowed to move towards p. The number of candidates selected equals the capacity of the parking node. The target parking nodes are selected sequentially in the Target Parking Nodes Selection phase and the algorithm proceeds until each parking node except for one becomes saturated. While p becomes saturated, the target parking node is selected in $\mathcal{Q}^{++}$, which has the highest order among all the unsaturated parking nodes in $\mathcal{Q}^{++}$. We have proved in the Correctness section that $\mathcal{Q}^{++}$ remains invariant unless all the parking nodes in $\mathcal{Q}^{++}$ become saturated. While the parking nodes in $\mathcal{Q}^{++}$ become saturated, the quadrants adjacent to $\mathcal{Q}^{++}$ remain invariant. Finally, while all but one parking node becomes saturated, the guard moves toward its destination in the Guard Movement phase. The procedure terminates when the guard reaches its destination.

   Next, assume the case when $C(t)$ is symmetric with respect to rotational symmetry. If $C(t) \in \mathcal{I}_{321}$, at least one robot position exists on c. The procedure $AllowtoMove()$ transforms the configuration into an asymmetric configuration by allowing the robot on c to move toward an adjacent node, only when there is a free space available around r. This results in transforming the configuration into an asymmetric configuration. If $C(t) \in \mathcal{I}_{322}$, the target parking nodes are selected as the unsaturated parking nodes with the highest order. Two or four parking nodes are selected at any instant of time in the different quadrants, depending on whether

the angle of rotation is either 180° or 90°. The candidate robots are selected in different quadrants and allowed to move toward their respective targets. The candidates move either synchronously or there may be a possible pending move due to the asynchronous behavior of the scheduler. If the configuration transforms into an asymmetric configuration, then the algorithm proceeds similarly as in $\mathcal{I}_{31}$. Otherwise, if the configuration regains its symmetry, then the configuration remains in $\mathcal{I}_{322}$. The algorithm proceeds unless each parking node becomes saturated. Finally, consider the case when $C(0) \in \mathcal{I}_{323}$. There exists a parking node p' on c. It should be noted that the capacity of p' must be a multiple of 2. Otherwise, the configuration would be unsolvable according to Corollary 6.4.4. If p' is unsaturated, then p' is selected as the target parking node. The two or four candidate robots are selected in the Candidate Robot Selection phase, depending on whether the angle of rotation is 180° or 90°. The candidates move towards their respective targets either synchronously or there may be a possible pending move due to the asynchronous behavior of the scheduler. While p' becomes saturated, the target parking nodes are selected in the quadrants as well as on the wedge boundaries. The rest of the procedure proceeds similarly as in the case when $C(0) \in \mathcal{I}_{322}$.

6.6 Correctness

Lemma 6.6.1. *In the Guard Selection and Placement phase, the guard remains invariant while it moves towards its destination.*

Proof. The proof follows from Lemma 3.5.2. □

Lemma 6.6.2. *During the execution of the algorithm Parking(), if the parking nodes admit a single line of symmetry l, then $\mathcal{H}^+$ remains invariant.*

Proof. The following cases are to be considered.

Case 1. Condition C_4 holds. This implies that there exists a unique half-plane $\mathcal{H}^+$, which contains a minimum number of unsaturated parking nodes. As a result, there exists at least one saturated parking node in $\mathcal{H}^+$. The target parking node is selected in $\mathcal{H}^+$ in the Target Parking Nodes Selection phase, which has the highest order among all unsaturated parking nodes in $\mathcal{H}^+$. During the execution of the algorithm, no robot

at the saturated parking nodes is allowed to move. As a result, a saturated parking node can never become unsaturated, and the half-plane with the minimum number of unsaturated parking nodes remains invariant. Hence, $\mathcal{H}^+$ remains invariant.

Case 2. Condition $\neg C_4 \wedge C_5$ holds. This implies that the number of unsaturated parking nodes is the same in both the half-planes. However, $K_1 \neq K_2$. Assume that $K_1 < K_2$. The target parking node is selected in $\mathcal{H}^+$ with the minimum K_i value, $i \in \{1, 2\}$, i.e., the half-plane, which has the minimum total deficit measure. Suppose at time $t > 0$ a candidate robot is selected in the Candidate Robot Selection phase and allowed to move towards the target parking node. While this robot reaches the target parking node, the value of K_1 becomes much less than K_2. Eventually, there exists a time $t' > t$ in which at least one parking node in $\mathcal{H}^+$ becomes saturated. The rest of the proof follows from Case 1.

Case 3. Condition $\neg\ C_4 \wedge \neg C_5 \wedge C_6$ holds. This implies that $K_1 = K_2$ and the configuration is unbalanced. $\mathcal{H}^+$ is selected as the half-plane with the maximum number of robots. The symmetry of the parking nodes is also defined with respect to their capacities. As a result, according to the execution of the algorithm, a robot in $\mathcal{H}^+$ will only move to the half-plane $\mathcal{H}^-$ when all of the parking nodes in $\mathcal{H}^+$ become saturated. There will eventually be a time $t > 0$, when a robot reaches the target parking node, resulting in the value of K_1 being less than the value of K_2. The rest of the proof follows from Case 2.

Case 4. Condition $\neg\ C_4 \wedge \neg C_5 \wedge \neg\ C_6 \wedge C_7$ holds. This implies the configuration is balanced and at least one robot position exists on l. The northernmost robot on l moves towards an adjacent node and the configuration becomes unbalanced, resulting in the condition C_6 evaluating to true. The rest of the proof follows from Case 3.

Case 5. Condition $\neg\ C_4 \wedge \neg C_5 \wedge \neg\ C_6 \wedge \neg C_7 \wedge C_8$ holds. This implies the configuration is balanced and there do not exist any robot positions on l. $\mathcal{H}^+$ is selected as the half-plane that does not contain the guard. The guard moves only when each parking node except for one becomes saturated. As a result, the target parking node will be selected in $\mathcal{H}^+$, unless each parking node in $\mathcal{H}^+$ becomes saturated. Hence, $\mathcal{H}^+$ remains invariant.

□

Lemma 6.6.3. *During the execution of the algorithm Parking(), if the parking nodes admit rotational symmetry, then $\mathcal{Q}^{++}$ remains invariant.*

Proof. The following cases are to be considered.

Case 1. Condition C_9 holds. This implies that there exists a unique quadrant $\mathcal{Q}^{++}$, that contains the minimum number of unsaturated parking nodes. As a result, there exists at least one saturated parking node in $\mathcal{Q}^{++}$. The target parking node is selected in $\mathcal{Q}^{++}$ in the Target Parking Nodes Selection phase, which has the highest order among all unsaturated parking nodes in $\mathcal{Q}^{++}$. Since no robot lying on a saturated parking node is allowed to move in the Candidate Robot Selection phase, a saturated parking node can never become unsaturated. This implies that the minimum number of unsaturated parking nodes in the quadrant remains invariant. Hence, $\mathcal{Q}^{++}$ remains invariant.

Case 2. Condition $\neg\, C_9 \wedge C_{10}$ holds. This implies that the number of unsaturated parking nodes in each quadrant is the same. However, there exists a unique quadrant with a minimum L_j value, for some $j \in \{1, 2, 3, 4\}$, i.e., there exists a unique quadrant with the minimum total deficit measure. Assume that the quadrant Q_j has the minimum L_j value. The target parking node is selected in $\mathcal{Q}_j$, which is demarcated as $\mathcal{Q}^{++}$. Suppose at time $t > 0$, a candidate robot is selected in the Candidate Robot Selection phase and allowed to move towards the target parking node in $\mathcal{Q}^{++}$. While this robot reaches the target parking node, the value of L_j decreases further. As a result, $\mathcal{Q}^{++}$ remains the unique quadrant with the minimum L_j value. Eventually, there exists a time $t' > t$ at which at least one parking node in $\mathcal{Q}^{++}$ becomes saturated. The rest of the proof follows from Case 1.

Case 3. Condition $\neg\, C_9 \wedge \neg C_{10} \wedge C_{11}$ holds. This implies that there exist at least two quadrants with the minimum value of L_j. However, the configuration is unbalanced. That is, there exists a unique quadrant with the minimum value of L_j and with the maximum number of robot positions. $\mathcal{Q}^{++}$ is selected as the quadrant with the minimum L_j value and maximum number of robot positions. According to the execution of the algorithm, a robot from the quadrant $\mathcal{Q}^{++}$ will move towards one of the wedge boundaries only if each target parking node in $\mathcal{Q}^{++}$ is saturated. As a result, there exists an instant of time, where a candidate robot reaches a parking node, resulting in a unique quadrant with the minimum L_j value. The rest of the proof follows from Case 2.

Case 4. Condition $\neg\, C_9 \wedge \neg C_{10} \wedge \neg C_{11} \wedge C_{12}$ holds. The configuration is balanced. The robot with the highest order among all the robots lying on $l \cup l'$ first moves along $l \cup l'$.

Next, when it is one node away from the quadrant with the minimum L_j value and the maximum number of robot positions, then it moves towards an adjacent node. This results in transforming the configuration into an unbalanced configuration. The rest of the proof follows from Case 3.

Case 5. Condition $\neg C_9 \wedge \neg C_{10} \wedge \neg C_{11} \wedge \neg C_{12} \wedge C_{13}$ holds. This implies the configuration is balanced and there do not exist any robot positions on $l \cup l'$. $\mathcal{Q}^{++}$ is selected as the quadrant that is non-adjacent to the quadrant containing the guard or the quadrant not containing the guard if there exist exactly two quadrants containing the maximum number of robot positions and the minimum number of unsaturated parking nodes. The guard moves only when each parking node except for one becomes saturated. As a result, the quadrant non-adjacent to the quadrant containing the guard remains invariant. The target parking node will be selected in $\mathcal{Q}^{++}$ unless each parking node in $\mathcal{Q}^{++}$ becomes saturated. Eventually, there exists a time when each parking node in $\mathcal{Q}^{++}$ becomes saturated. Hence, $\mathcal{Q}^{++}$ remains invariant. □

Lemma 6.6.4. *If the configuration is such that the parking nodes admit a unique line of symmetry l, then during the execution of the algorithm Parking(), the target parking nodes remain invariant.*

Proof. The following cases are to be considered.

Case 1. There exists at least one unsaturated parking node on l. Let p be the target parking node selected on l in the Target Parking Nodes Selection phase. As a result, p is the unsaturated parking node on l, which has the highest order with respect to $\mathcal{O}_2$. Since the ordering O_2 depends only on the positions of the fixed parking nodes, the ordering remains invariant while the robot moves towards it. Hence, p remains the target parking node unless it becomes saturated.

Case 2. Each parking node on l is saturated and the configuration is asymmetric. The following subcases are to be considered.

Subcase 1. There exists an unsaturated parking node in $\mathcal{H}^+$. Let p' be the target parking node in $\mathcal{H}^+$, selected in the Target Parking Nodes Selection phase. According to Lemma 6.6.2, $\mathcal{H}^+$ remains invariant unless each parking node in $\mathcal{H}^+$ becomes saturated. Since the ordering of the parking nodes is defined with respect to the leading corners and

$\mathcal{H}^+$ remains invariant, p' remains the target parking node unless it becomes saturated. Hence, the target parking node remains invariant.

Subcase 2. Each parking node in $\mathcal{H}^+$ is saturated. Let p'' be the target parking node in $\mathcal{H}^-$, selected in the Target Parking Nodes Selection phase. It should be noted that a parking node in $\mathcal{H}^-$ is selected as a target parking node only when each parking node in $\mathcal{H}^+$ becomes saturated. Since the ordering of the parking nodes is defined with respect to the leading corners and $\mathcal{H}^+$ remains invariant, p'' remains the parking node which has the highest order in $\mathcal{H}^-$ unless it becomes saturated. Hence, the target parking node remains invariant.

Case 3. Each parking node on l is saturated and the configuration is symmetric. There are two target parking nodes selected in two different half-planes. It should be noted that the ordering of the parking nodes is defined with respect to the leading corner, and the leading corners remain invariant while the robots move toward the target. While the candidate robots move towards the target parking nodes, they remain the unsaturated parking nodes with the highest order in their respective half-planes. Hence, the target parking node remains invariant. □

Lemma 6.6.5. *If the configuration is such that the parking nodes admit rotational symmetry, then during the execution of the algorithm Parking(), the target parking nodes remain invariant.*

Proof. The following cases are to be considered.

Case 1. There exists a parking node p on c, which is unsaturated. Since c is the center of rotational symmetry for the parking nodes, it remains invariant while the robot moves towards it. As a result, p remains the target parking node unless it becomes saturated.

Case 2. There does not exist a parking node on c or the parking node on c is saturated. The configuration is asymmetric. First, assume that $p \neq p_{guard}$ is the target parking node. The following subcases are to be considered.

Subcase 1. The target parking node is selected in $\mathcal{Q}^{++}$. Let p' be the target parking node selected in $\mathcal{Q}^{++}$. As a result, p' is the unsaturated parking node in $\mathcal{Q}^{++}$, which has the highest order among all the unsaturated parking nodes in $\mathcal{Q}^{++}$. According to Lemma 6.6.3, $\mathcal{Q}^{++}$ remains invariant unless each parking node in $\mathcal{Q}^{++}$ becomes saturated. It should be noted that the leading corners are defined with respect to the position of fixed

parking nodes. The leading corners remain invariant while the robot moves toward the target parking nodes. Since the ordering of the parking nodes is defined with respect to the leading corners and $\mathcal{Q}^{++}$ remains invariant, p' remains the target parking node unless it becomes saturated. Hence, the target parking node remains invariant.

Subcase 2. The target parking node is selected on the wedge boundaries corresponding to $\mathcal{Q}^{++}$. This implies that each parking node in $\mathcal{Q}^{++}$ becomes saturated. As $\mathcal{Q}^{++}$ remains invariant according to Lemma 6.6.3, the wedge boundaries corresponding to $\mathcal{Q}^{++}$ remain invariant. Since the leading corners are defined with respect to the position of fixed parking nodes, the unsaturated parking node with the highest order on the wedge boundaries remains invariant. Hence, the target parking node remains invariant.

Subcase 3. The target parking node is selected on a quadrant adjacent to $\mathcal{Q}^{++}$. This implies that each parking node in the wedge boundaries corresponding to $\mathcal{Q}^{++}$ becomes saturated. It should be noted that since $\mathcal{Q}^{++}$ remains invariant, the quadrants adjacent to $\mathcal{Q}^{++}$ remain invariant. As the leading corners are defined with respect to the position of fixed parking nodes, the unsaturated parking node with the highest order on the quadrants $\mathcal{Q}^{+-}$ and $\mathcal{Q}^{-+}$ remains invariant. Hence, the target parking node remains invariant.

Subcase 4. The target parking node is selected on the wedge boundaries corresponding to $\mathcal{Q}^{+-}$ and $\mathcal{Q}^{-+}$. Since $\mathcal{Q}^{+-}$ and $\mathcal{Q}^{-+}$ remain invariant, the wedge boundaries corresponding to $\mathcal{Q}^{+-}$ and $\mathcal{Q}^{-+}$ remain invariant. As a result, the unsaturated parking node having the highest order on the wedges remains invariant. Hence, the target parking node remains invariant.

Subcase 5. The target parking node is selected in $\mathcal{Q}^{--}$. This implies that each parking node on the other quadrants and on the wedge boundaries becomes saturated. As a result, at this stage of the algorithm, $\mathcal{Q}^{--}$ is the unique quadrant that has unsaturated parking nodes. The target parking node is selected in $\mathcal{Q}^{--}$, which is unsaturated and has the highest order in $\mathcal{Q}^{--}$. Since $\mathcal{Q}^{--}$ is the quadrant non-adjacent to $\mathcal{Q}^{++}$ and $\mathcal{Q}^{++}$ remains invariant, the target parking node also remains invariant. Hence, the target parking node remains invariant.

Let p_{guard} be the target parking node. Note that p_{guard} would be selected as the target parking node only when each parking node becomes saturated. As a result, it remains invariant.

Case 3. There does not exist a parking node on c or the parking node on c is saturated. The configuration is symmetric with respect to rotational symmetry. In that case, the target parking nodes are selected as the unsaturated parking nodes with the highest order. It should be noted that these targets are selected either on the quadrants or on the wedge boundaries, and at any given time, there is a maximum of four parking nodes selected as targets. Since the leading corners are defined with respect to the position of fixed parking nodes, the parking nodes having the highest orders remain invariant. Hence, the target parking nodes remain invariant. □

Lemma 6.6.6. *During the Candidate Robot Selection phase, the candidate robot remains invariant.*

Proof. The following cases are to be considered.

Case 1. $C(t)$ is asymmetric. The candidate robot is selected as the robot that has a shortest free path toward the target parking node. If there exists a unique such robot, then while the candidate robot moves towards its target, it remains the unique robot that has a shortest free path towards its target. Consider the case when there are multiple such robots. Since $C(t)$ is asymmetric, the robots are orderable. The candidate is selected as the robot which has the highest order among such robots. While the candidate moves an adjacent node toward the target, it becomes the unique robot that has a shortest free path toward the target. The rest of the proof follows from the previous case when there exists a unique robot that has a shortest free path toward the target parking node. Hence, the candidate robot remains invariant.

Case 2. $C(t)$ is symmetric with respect to a single line of symmetry l and the target parking node is on l. The candidate robots are selected as the two symmetric robots that have a shortest free path toward p. While they move toward p, they remain the robots with a shortest free path toward p. If there are multiple such robots, the candidate robots are the two robots with the highest order in their respective half-planes and have a shortest free path toward p. While they move towards p, they remain the unique robots in their respective half-planes that have a shortest free path towards p. Hence, the candidate robot remains invariant.

Case 3. $C(t)$ is symmetric with respect to a single line of symmetry l and the target parking nodes are on the half-planes. It should be noted that in this case, a candidate

robot is selected in each half-plane, for each target parking node. The candidates are selected as the robots which have a shortest free path toward the target. While the candidates move toward their respective targets, it remains the unique robot that has a shortest free path toward their target. The proof proceeds similarly to the previous case when there are multiple such robots for each target parking node. Hence, the candidate robot remains invariant.

Case 4. $C(t)$ is symmetric with respect to rotational symmetry and the target parking node is on c. Depending on whether the angle of rotation is 90° or 180°, two or four candidate robots are selected. The robots with the shortest free path toward the target are selected as candidate robots. While they move toward the target, they remain the robots that have a shortest free path toward the target. Hence, the candidate robot remains invariant.

Case 5. $C(t)$ is symmetric with respect to rotational symmetry and the target parking nodes are on the quadrants or on the wedge boundaries. It should be noted that in this case, a candidate robot is selected in each quadrant or on the wedge boundaries, for each target parking node. The candidates are selected as the robots which have a shortest free path toward the target. While the candidates move toward their respective targets, it remains the unique robot that has a shortest free path toward their target. Hence, the candidate robot remains invariant. □

Lemma 6.6.7. *Procedure AllowtoMove() transforms any configuration in* $\mathcal{I}_{221} \cup \mathcal{I}_{331}$ *into an asymmetric configuration.*

Proof. The proof follows from Lemma 5.4.7. □

Theorem 6.6.8. *Algorithm Parking() solves the Parking Problem in Infinite grids for all configurations not belonging to the set* $\mathcal{U}$.

Proof. The algorithm Parking() proceeds according to the different classes of configurations. The following cases are to be considered.

Case 1. $C(0) \in \mathcal{I}_1$: Since the parking nodes are asymmetric, the parking nodes are uniquely orderable. The ordering remains invariant while the robots move toward the target parking node. The target parking node can be selected in a unique manner. According to Lemma 6.6.6, the candidate robot remains invariant while it moves toward

the target parking node. While the target parking node contains a number of robots equal to the capacity of the parking node, it becomes saturated. The process continues unless each parking node becomes saturated. Hence, the algorithm Parking() solves the parking problem for all configurations belonging to $\mathcal{I}_1$.

Case 2. $C(0) \in \mathcal{I}_2$: First consider the case when $C(0) \in \mathcal{I}_{21}$. In the Guard Selection and Placement phase, a guard is selected and placed in such a way that the configuration remains asymmetric during the execution of the algorithm. As a result, the configuration $C(t) \notin \mathcal{U}$, for any $t > 0$. The target parking node is selected in the Target Parking Nodes Selection phase, which remains invariant according to Lemma 6.6.4. The candidate robot moves toward the target parking node. According to Lemma 6.6.6, the candidate robot remains invariant while it moves toward the target parking node. When all the parking nodes become saturated, the algorithm terminates. Next, consider the case when $\mathcal{C}(0) \in \mathcal{I}_{221}$. The procedure AllowtoMove() transforms the configuration into an asymmetric configuration and the procedure proceeds similarly as before. If $C(0) \in \mathcal{I}_{222}$, the target parking node is selected in the Target Parking Nodes Selection phase, which remains invariant according to Lemma 6.6.4. The candidate robot moves toward the target parking node. According to Lemma 6.6.6, the candidate robot remains invariant while it moves toward the target parking node. The procedure terminates when all the parking nodes become saturated. In case $C(0) \in \mathcal{I}_{223}$, first the parking nodes on l become saturated and the rest of the procedure proceeds similarly as in the case when $C(0) \in \mathcal{I}_{222}$.

Case 3. $\mathcal{C}(0) \in \mathcal{I}_3$. First, consider the case when $C(0) \in \mathcal{I}_{31}$. The placement of the guard in the Guard Selection and Placement phase ensures that the configuration remains asymmetric during the execution of the algorithm Parking(). As a result, the configuration $C(t) \notin \mathcal{U}$, for any $t > 0$. The target parking node is selected in the Target Parking Nodes Selection phase, which remains invariant according to Lemma 6.6.5. The candidate robot moves toward the target parking node. According to Lemma 6.6.6, the candidate robot remains invariant while it moves toward the target parking node. When all the parking nodes become saturated, the algorithm terminates. If $C(0) \in \mathcal{I}_{321}$, then the procedure AllowtoMove() transforms the configuration into an asymmetric configuration. The procedure Parking() proceeds similarly as before. If $C(0) \in \mathcal{I}_{322}$, the target parking node is selected in the Target Parking Nodes Selection phase, which remains invariant according to Lemma 6.6.5. The candidate robot moves toward the

target parking node. According to Lemma 6.6.6, the candidate robot remains invariant while it moves toward the target parking node. The procedure terminates when all the parking nodes become saturated. In case $C(0) \in \mathcal{I}_{323}$, first the parking nodes on c become saturated and the rest of the procedure proceeds similarly as in the case when $C(0) \in \mathcal{I}_{322}$. □

6.7 Conclusion

This chapter proposed a deterministic distributed algorithm for solving the parking problem in infinite grids. The robots are placed at the nodes of an infinite grid graph, which also includes some prefixed parking nodes. In the initial configuration, for each parking node p_i, there is a capacity k_i, which is given as an input to the robots. We have characterized all the initial configurations and the values of k_i for which the problem is unsolvable, even if the robots are endowed with strong multiplicity detection capability. A deterministic algorithm has been proposed under the assumption that the robots are endowed with global-strong multiplicity detection capability. We have also proved that without this assumption, the parking problem is unsolvable.

Chapter 7

Conclusion

The main objective of this thesis has been the theoretical study of some variations of the gathering problem, where the deployment region of the robots is an infinite grid graph. In this chapter, we conclude the thesis by summarizing all the technical results from the previous chapters and also highlighting some interesting directions for future research.

In Chapter 3, we studied the gathering over meeting nodes problem in an infinite grid graph. We have proved that even if the robots are endowed with strong multiplicity detection capability, some configurations remain ungatherable. These include the configuration: which admits a single line of symmetry without any robots or meeting nodes on the line of symmetry and those which admit rotational symmetry without any robot or meeting node on the center of rotational symmetry. A distributed deterministic algorithm has been proposed to solve the problem for a set of $n \geq 2$ robots for the remaining configurations. We have studied the efficiency of the proposed algorithm with respect to the number of moves and analyzed the time complexity of the algorithm in terms of epochs. The proposed algorithm solves the problem in $\Theta(Dn)$ moves, where D is the larger side of the minimum enclosing rectangle and n is the number of robots in the system. In terms of epoch, it requires $\Theta(D)$ epochs to accomplish the gathering task.

Problem 1: One future research direction would be to investigate randomized algorithms to ensure gathering for configurations that have been shown to be ungatherable.

Problem 2: The robots are assumed to be endowed with local-weak multiplicity detection capability. Considering the problem in a setting where the robots do not have any multiplicity detection capability would be interesting.

In Chapter 4, we studied the optimal gathering over Weber meeting nodes problem in infinite grids. The objective constraint is to minimize the total number of moves required to accomplish the gathering. We have observed that if the robots gather at one of the Weber meeting nodes, the total sum of the distances traveled by each robot in order to accomplish the gathering is minimized. We have characterized all those configurations where gathering over a Weber meeting node cannot be ensured, even if the robots have global-strong multiplicity detection capability. These include the configuration: which admits a single line of symmetry without any robots or Weber meeting nodes on the line of symmetry and those which admit a rotational symmetry without any robot or meeting node on the center of rotational symmetry. For the remaining configurations, a distributed deterministic algorithm has been proposed to solve the problem for a set of $n \geq 7$ robots. In some initial configurations, the gathering cannot be ensured over a Weber meeting node. However, the gathering may still be accomplished at one of the meeting nodes. This includes the configuration that admits a single line of symmetry without any Weber meeting nodes on the reflection axis, but at least one meeting node exists on the axis. In that case, the feasibility of solving the problem has been studied.

Problem 3: The robots in this work have global-strong multiplicity detection capability. One future direction could be to characterize all initial configurations in which optimal gathering can be accomplished if the robots are equipped with global-weak multiplicity detection capability.

Problem 4: One immediate direction for future research would be to consider the problem under different optimization criteria. One optimization criterion would be to study the *min-max gathering* over meeting nodes problem, where robots must gather at a meeting point so that the maximum distance traveled by any robot is minimized.

Problem 5: Another future research direction would be to investigate the *Optimal Gathering over Weber Meeting Nodes in Infinite Grid* problem with multiplicities in the initial configuration.

In Chapter 5, we studied the gathering over heterogeneous meeting nodes problem in infinite grids. The initial configurations for which the gathering problem is not solvable have been characterized and a distributed gathering algorithm has been proposed for the remaining initial configurations. The number of robots is assumed to be at least five. The efficiency of the proposed algorithm has been discussed in terms of the total number of moves executed by the robots. The proposed algorithm runs in $\Theta(dn)$ moves, where d is the diameter of the minimum enclosing rectangle of all the robots and meeting nodes. We have also analyzed the time complexity of the algorithm in terms of the number of epochs. The proposed algorithm terminates in $O(dn)$ epochs. However, we have proved that any algorithm that solves the problem requires $O(d)$ epochs.

Problem 6: Another possible research direction would be to study distributed algorithms that close the gap between the lower and upper time bounds of *Gathering over Heterogeneous Meeting Nodes* problem.

The gathering algorithm is not optimal with respect to the exact number of moves traveled by the robots in order to finalize the gathering. It would be interesting to consider optimal algorithms that accomplish gathering by minimizing the number of moves. We define this problem as the *Min Sum Optimal Gathering over Heterogeneous Meeting Nodes* problem.

Problem 7: Investigate the *Min Sum Optimal Gathering over Heterogeneous Meeting Nodes* problem.

In Chapter 6, we studied the parking problem in infinite grids. The robots are placed at the nodes of an infinite grid graph, which also includes some prefixed parking nodes. For each parking node p_i, there is a capacity k_i. We have characterized all the initial configurations and the values of k_i for which the problem is unsolvable. A deterministic algorithm has been proposed for the remaining configurations to achieve the desired goal.

In the initial configuration, we assumed that the number of robots equaled the sum of the capacities of the parking nodes.

Problem 8: It would be interesting to investigate the problem in case the number of robots is not equal to the sum of the capacities of the parking nodes.

Contributions			
Chapters	**Problem**	**Assumptions**	**Status**
Chapter 3	Gathering over Meeting Nodes Problem	$\mathcal{ASYNC}$, local weak-multiplicity capability	Solved for $n \geq 2$ robots
Chapter 4	Optimal Gathering over Weber Meeting Nodes Problem in Infinite Grid	$\mathcal{ASYNC}$, global-strong multiplicity detection capability	Solved for $n \geq 7$ robots
Chapter 5	Gathering over Heterogeneous Meeting Nodes	$\mathcal{ASYNC}$, global-weak multiplicity detection capability	Solved for $n \geq 5$ robots
Chapter 6	Parking Problem in Infinite Grids	$\mathcal{ASYNC}$, global-strong multiplicity detection capability	Solved for $n \geq 2$ robots

TABLE 7.1: Contributions

In case the number of robots in the initial configuration is less than the sum of the capacities of the parking nodes, one interesting study could be to investigate the problem with the objective of maximizing the number of saturated parking nodes. We define this problem as the *Maximum Saturated Parking Node Problem* in infinite grids.

Problem 9: Investigate the *Maximum Saturated Parking Node Problem* in infinite grids.

We have investigated the different versions of the gathering over meeting nodes problem in this thesis, where the robots are assumed to be transparent.

Problem 10: Another interesting problem would be to consider the problem under the assumption that the robots are opaque.

Problem 11: Another variation of the problem could be to consider the gathering problem by fat robots.

Another direction of future research would be to consider the complexity analysis of the Algorithm Parking(). As a measure of complexity, either the total number of robot movements or the time measured in epochs could be considered.

Problem 12: Analyse the complexity of the algorithm Parking().

The contributions of the thesis are summarized in Table 7.1.

The main objective of building such a system of robots is to solve real-life problems in an efficient manner. The theoretical perspective of such systems has been the subject of extensive research over the past two decades. Another direction of future research could be

to design algorithms for the model with limited visibility. Situations where the visibility is limited to a certain radius would add more realistic insights into the problem. Another branch of research deals with human-centered intelligent robots. Human-centered intelligent robots have become a significant area of research that covers all aspects of robotic capabilities, including pattern recognition, intelligent control, navigation, motion planning and human-robot interaction. We expect that combining the research in these two fields in the future will lead to the development of a domain with strong theoretical foundations and efficient practical applications.

Bibliography

[1] Ranendu Adhikary, Kaustav Bose, Manash Kumar Kundu, and Buddhadeb Sau. Mutual visibility on grid by asynchronous luminous robots. *Theor. Comput. Sci.*, 922:218–247, 2022.

[2] Ranendu Adhikary, Manash Kumar Kundu, and Buddhadeb Sau. Circle formation by asynchronous opaque robots on infinite grid. *Comput. Sci.*, 22(1), 2021.

[3] Chrysovalandis Agathangelou, Chryssis Georgiou, and Marios Mavronicolas. A distributed algorithm for gathering many fat mobile robots in the plane. In Panagiota Fatourou and Gadi Taubenfeld, editors, *ACM Symposium on Principles of Distributed Computing, PODC '13, Montreal, QC, Canada, July 22-24, 2013*, pages 250–259. ACM, 2013.

[4] Noa Agmon and David Peleg. Fault-tolerant gathering algorithms for autonomous mobile robots. *SIAM J. Comput.*, 36(1):56–82, 2006.

[5] Hideki Ando, Yoshinobu Oasa, Ichiro Suzuki, and Masafumi Yamashita. Distributed memoryless point convergence algorithm for mobile robots with limited visibility. *IEEE Trans. Robotics Autom.*, 15(5):818–828, 1999.

[6] John Augustine and William K. Moses Jr. Dispersion of mobile robots: A study of memory-time trade-offs. In Paolo Bellavista and Vijay K. Garg, editors, *Proceedings of the 19th International Conference on Distributed Computing and Networking, ICDCN 2018, Varanasi, India, January 4-7, 2018*, pages 1:1–1:10. ACM, 2018.

[7] Roberto Baldoni, François Bonnet, Alessia Milani, and Michel Raynal. Anonymous graph exploration without collision by mobile robots. *Inf. Process. Lett.*, 109(2):98–103, 2008.

[8] Roberto Baldoni, François Bonnet, Alessia Milani, and Michel Raynal. On the solvability of anonymous partial grids exploration by mobile robots. In Theodore P. Baker, Alain Bui, and Sébastien Tixeuil, editors, *Principles of Distributed Systems, 12th International Conference, OPODIS 2008, Luxor, Egypt, December 15-18, 2008. Proceedings*, volume 5401 of *Lecture Notes in Computer Science*, pages 428–445. Springer, 2008.

[9] Eduardo Mesa Barrameda, Shantanu Das, and Nicola Santoro. Deployment of asynchronous robotic sensors in unknown orthogonal environments. In Sándor P. Fekete, editor, *Algorithmic Aspects of Wireless Sensor Networks*, pages 125–140, Berlin, Heidelberg, 2008. Springer Berlin Heidelberg.

[10] Subhash Bhagat. Optimum algorithm for the mutual visibility problem. In M. Sohel Rahman, Kunihiko Sadakane, and Wing-Kin Sung, editors, *WALCOM: Algorithms and Computation - 14th International Conference, WALCOM 2020, Singapore, March 31 - April 2, 2020, Proceedings*, volume 12049 of *Lecture Notes in Computer Science*, pages 31–42. Springer, 2020.

[11] Subhash Bhagat, Sruti Gan Chaudhuri, and Krishnendu Mukhopadhyaya. Fault-tolerant gathering of asynchronous oblivious mobile robots under one-axis agreement. *J. Discrete Algorithms*, 36:50–62, 2016.

[12] Subhash Bhagat, Bibhuti Das, Abhinav Chakraborty, and Krishnendu Mukhopadhyaya. k-circle formation and k-epf by asynchronous robots. *Algorithms*, 14(2):62, 2021.

[13] Subhash Bhagat, Paola Flocchini, Krishnendu Mukhopadhyaya, and Nicola Santoro. Weak robots performing conflicting tasks without knowing who is in their team. In Nandini Mukherjee and Sriram V. Pemmaraju, editors, *ICDCN 2020: 21st International Conference on Distributed Computing and Networking, Kolkata, India, January 4-7, 2020*, pages 29:1–29:6. ACM, 2020.

[14] Subhash Bhagat and Anisur Rahaman Molla. Min-max gathering of oblivious robots. In Kunal Agrawal and Yossi Azar, editors, *SPAA '21: 33rd ACM Symposium on Parallelism in Algorithms and Architectures, Virtual Event, USA, 6-8 July, 2021*, pages 420–422. ACM, 2021.

[15] Subhash Bhagat and Krishnendu Mukhopadhyaya. Optimum gathering of asynchronous robots. In Daya Ram Gaur and N. S. Narayanaswamy, editors, *Algorithms and Discrete Applied Mathematics - Third International Conference, CALDAM 2017, Sancoale, Goa, India, February 16-18, 2017, Proceedings*, volume 10156 of *Lecture Notes in Computer Science*, pages 37–49. Springer, 2017.

[16] François Bonnet, Alessia Milani, Maria Potop-Butucaru, and Sébastien Tixeuil. Asynchronous exclusive perpetual grid exploration without sense of direction. In Antonio Fernàndez Anta, Giuseppe Lipari, and Matthieu Roy, editors, *Principles of Distributed Systems*, pages 251–265, Berlin, Heidelberg, 2011. Springer Berlin Heidelberg.

[17] François Bonnet, Maria Potop-Butucaru, and Sébastien Tixeuil. Asynchronous gathering in rings with 4 robots. In Nathalie Mitton, Valeria Loscrì, and Alexandre Mouradian, editors, *Ad-hoc, Mobile, and Wireless Networks - 15th International Conference, ADHOC-NOW 2016, Lille, France, July 4-6, 2016, Proceedings*, volume 9724 of *Lecture Notes in Computer Science*, pages 311–324. Springer, 2016.

[18] Kaustav Bose, Ranendu Adhikary, Manash Kumar Kundu, and Buddhadeb Sau. Arbitrary pattern formation on infinite grid by asynchronous oblivious robots. *Theoretical Computer Science*, 815:213–227, 2020.

[19] Kaustav Bose, Manash Kumar Kundu, Ranendu Adhikary, and Buddhadeb Sau. Optimal gathering by asynchronous oblivious robots in hypercubes. In Seth Gilbert, Danny Hughes, and Bhaskar Krishnamachari, editors, *Algorithms for Sensor Systems - 14th International Symposium on Algorithms and Experiments for Wireless Sensor Networks, ALGOSENSORS 2018, Helsinki, Finland, August 23-24, 2018, Revised Selected Papers*, volume 11410 of *Lecture Notes in Computer Science*, pages 102–117. Springer, 2018.

[20] Quentin Bramas, Pascal Lafourcade, and Stéphane Devismes. Optimal exclusive perpetual grid exploration by luminous myopic opaque robots with common chirality. In *ICDCN '21: International Conference on Distributed Computing and Networking, Virtual Event, Nara, Japan, January 5-8, 2021*, pages 76–85. ACM, 2021.

[21] Quentin Bramas, Anissa Lamani, and Sébastien Tixeuil. The agreement power of disagreement. In Colette Johnen, Elad Michael Schiller, and Stefan Schmid, editors, *Stabilization, Safety, and Security of Distributed Systems - 23rd International Symposium, SSS 2021, Virtual Event, November 17-20, 2021, Proceedings*, volume 13046 of *Lecture Notes in Computer Science*, pages 273–288. Springer, 2021.

[22] Quentin Bramas and Sébastien Tixeuil. Wait-free gathering without chirality. In Christian Scheideler, editor, *Structural Information and Communication Complexity - 22nd International Colloquium, SIROCCO 2015, Montserrat, Spain, July 14-16, 2015, Post-Proceedings*, volume 9439 of *Lecture Notes in Computer Science*, pages 313–327. Springer, 2015.

[23] Davide Canepa and Maria Gradinariu Potop-Butucaru. Stabilizing flocking via leader election in robot networks. In Toshimitsu Masuzawa and Sébastien Tixeuil, editors, *Stabilization, Safety, and Security of Distributed Systems, 9th International Symposium, SSS 2007, Paris, France, November 14-16, 2007, Proceedings*, volume 4838 of *Lecture Notes in Computer Science*, pages 52–66. Springer, 2007.

[24] Jannik Castenow, Matthias Fischer, Jonas Harbig, Daniel Jung, and Friedhelm Meyer auf der Heide. Gathering anonymous, oblivious robots on a grid. *Theor. Comput. Sci.*, 815:289–309, 2020.

[25] Ioannis Chatzigiannakis, Michael Markou, and Sotiris Nikoletseas. Distributed circle formation for anonymous oblivious robots. In Celso C. Ribeiro and Simone L. Martins, editors, *Experimental and Efficient Algorithms*, pages 159–174, Berlin, Heidelberg, 2004. Springer Berlin Heidelberg.

[26] Sruti Gan Chaudhuri and Krishnendu Mukhopadhyaya. Gathering asynchronous transparent fat robots. In Tomasz Janowski and Hrushikesha Mohanty, editors, *Distributed Computing and Internet Technology, 6th International Conference, ICDCIT 2010, Bhubaneswar, India, February 15-17, 2010. Proceedings*, volume 5966 of *Lecture Notes in Computer Science*, pages 170–175. Springer, 2010.

[27] Sruti Gan Chaudhuri and Krishnendu Mukhopadhyaya. Leader election and gathering for asynchronous fat robots without common chirality. *J. Discrete Algorithms*, 33:171–192, 2015.

[28] Serafino Cicerone. Breaking symmetries on tessellation graphs via asynchronous robots. In Gennaro Cordasco, Luisa Gargano, and Adele A. Rescigno, editors, *Proceedings of the 21st Italian Conference on Theoretical Computer Science, Ischia, Italy, September 14-16, 2020*, volume 2756 of *CEUR Workshop Proceedings*, pages 122–136. CEUR-WS.org, 2020.

[29] Serafino Cicerone, Gabriele Di Stefano, and Alfredo Navarra. Gathering robots in graphs: The central role of synchronicity. *Theoretical Computer Science*, 849:99–120, 2021.

[30] Serafino Cicerone, Alessia Di Fonso, Gabriele Di Stefano, and Alfredo Navarra. Arbitrary pattern formation on infinite regular tessellation graphs. In *ICDCN '21: International Conference on Distributed Computing and Networking, Virtual Event, Nara, Japan, January 5-8, 2021*, pages 56–65. ACM, 2021.

[31] Serafino Cicerone, Gabriele Di Stefano, and Alfredo Navarra. Gathering of robots on meeting-points: feasibility and optimal resolution algorithms. *Distributed Comput.*, 31(1):1–50, 2018.

[32] Serafino Cicerone, Gabriele Di Stefano, and Alfredo Navarra. Asynchronous arbitrary pattern formation: the effects of a rigorous approach. *Distributed Comput.*, 32(2):91–132, 2019.

[33] Serafino Cicerone, Gabriele Di Stefano, and Alfredo Navarra. Embedded pattern formation by asynchronous robots without chirality. *Distributed Comput.*, 32(4):291–315, 2019.

[34] Mark Cieliebak. Gathering non-oblivious mobile robots. In Martin Farach-Colton, editor, *LATIN 2004: Theoretical Informatics, 6th Latin American Symposium, Buenos Aires, Argentina, April 5-8, 2004, Proceedings*, volume 2976 of *Lecture Notes in Computer Science*, pages 577–588. Springer, 2004.

[35] Mark Cieliebak, Paola Flocchini, Giuseppe Prencipe, and Nicola Santoro. Solving the robots gathering problem. In Jos C. M. Baeten, Jan Karel Lenstra, Joachim Parrow, and Gerhard J. Woeginger, editors, *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*, pages 1181–1196. Springer, 2003.

[36] Mark Cieliebak and Giuseppe Prencipe. Gathering autonomous mobile robots. In Christos Kaklamanis and Lefteris M. Kirousis, editors, *SIROCCO 9, Proceedings of the 9th International Colloquium on Structural Information and Communication Complexity, Andros, Greece, June 10-12, 2002*, volume 13 of *Proceedings in Informatics*, pages 57–72. Carleton Scientific, 2002.

[37] Reuven Cohen and David Peleg. Convergence properties of the gravitational algorithm in asynchronous robot systems. In Susanne Albers and Tomasz Radzik, editors, *Algorithms – ESA 2004*, pages 228–239, Berlin, Heidelberg, 2004. Springer Berlin Heidelberg.

[38] Reuven Cohen and David Peleg. Convergence properties of the gravitational algorithm in asynchronous robot systems. *SIAM J. Comput.*, 34(6):1516–1528, 2005.

[39] Reuven Cohen and David Peleg. Convergence of autonomous mobile robots with inaccurate sensors and movements. In Bruno Durand and Wolfgang Thomas, editors, *STACS 2006, 23rd Annual Symposium on Theoretical Aspects of Computer Science, Marseille, France, February 23-25, 2006, Proceedings*, volume 3884 of *Lecture Notes in Computer Science*, pages 549–560. Springer, 2006.

[40] Reuven Cohen and David Peleg. Convergence of autonomous mobile robots with inaccurate sensors and movements. *SIAM J. Comput.*, 38(1):276–302, 2008.

[41] Andreas Cord-Landwehr, Bastian Degener, Matthias Fischer, Martina Hüllmann, Barbara Kempkes, Alexander Klaas, Peter Kling, Sven Kurras, Marcus Märtens, Friedhelm Meyer auf der Heide, Christoph Raupach, Kamil Swierkot, Daniel Warner, Christoph Weddemann, and Daniel Wonisch. Collisionless gathering of robots with an extent. In Ivana Cerná, Tibor Gyimóthy, Juraj Hromkovic, Keith G. Jeffery, Rastislav Královic, Marko Vukolic, and Stefan Wolf, editors, *SOFSEM 2011: Theory and Practice of Computer Science - 37th Conference on Current Trends in Theory and Practice of Computer Science, Nový Smokovec, Slovakia, January 22-28, 2011. Proceedings*, volume 6543 of *Lecture Notes in Computer Science*, pages 178–189. Springer, 2011.

[42] Andreas Cord-Landwehr, Matthias Fischer, Daniel Jung, and Friedhelm Meyer auf der Heide. Asymptotically optimal gathering on a grid. In Christian Scheideler and Seth Gilbert, editors, *Proceedings of the 28th ACM Symposium on Parallelism in Algorithms and Architectures, SPAA 2016, Asilomar State Beach/Pacific Grove, CA, USA, July 11-13, 2016*, pages 301–312. ACM, 2016.

[43] Jurek Czyzowicz, Leszek Gasieniec, and Andrzej Pelc. Gathering few fat mobile robots in the plane. In Alexander A. Shvartsman, editor, *Principles of Distributed Systems, 10th International Conference, OPODIS 2006, Bordeaux, France, December 12-15, 2006, Proceedings*, volume 4305 of *Lecture Notes in Computer Science*, pages 350–364. Springer, 2006.

[44] Jurek Czyzowicz, Leszek Gasieniec, and Andrzej Pelc. Gathering few fat mobile robots in the plane. *Theor. Comput. Sci.*, 410(6-7):481–499, 2009.

[45] Gianlorenzo D'Angelo, Alfredo Navarra, and Nicolas Nisse. Gathering and exclusive searching on rings under minimal assumptions. In Mainak Chatterjee, Jiannong Cao, Kishore Kothapalli, and Sergio Rajsbaum, editors, *Distributed Computing and Networking - 15th International Conference, ICDCN 2014, Coimbatore, India, January 4-7, 2014. Proceedings*, volume 8314 of *Lecture Notes in Computer Science*, pages 149–164. Springer, 2014.

[46] Gianlorenzo D'Angelo, Alfredo Navarra, and Nicolas Nisse. A unified approach for gathering and exclusive searching on rings under weak assumptions. *Distributed Comput.*, 30(1):17–48, 2017.

[47] Gianlorenzo D'Angelo, Gabriele Di Stefano, Ralf Klasing, and Alfredo Navarra. Gathering of robots on anonymous grids and trees without multiplicity detection. *Theor. Comput. Sci.*, 610:158–168, 2016.

[48] Gianlorenzo D'Angelo, Gabriele Di Stefano, and Alfredo Navarra. How to gather asynchronous oblivious robots on anonymous rings. In Marcos K. Aguilera, editor, *Distributed Computing - 26th International Symposium, DISC 2012, Salvador, Brazil, October 16-18, 2012. Proceedings*, volume 7611 of *Lecture Notes in Computer Science*, pages 326–340. Springer, 2012.

[49] Gianlorenzo D'Angelo, Gabriele Di Stefano, and Alfredo Navarra. Gathering on rings under the look-compute-move model. *Distributed Comput.*, 27(4):255–285, 2014.

[50] Gianlorenzo D'Angelo, Gabriele Di Stefano, and Alfredo Navarra. Gathering six oblivious robots on anonymous symmetric rings. *J. Discrete Algorithms*, 26:16–27, 2014.

[51] Gianlorenzo D'Angelo, Gabriele Di Stefano, Alfredo Navarra, Nicolas Nisse, and Karol Suchan. Computing on rings by oblivious robots: A unified approach for different tasks. *Algorithmica*, 72(4):1055–1096, 2015.

[52] Shantanu Das, Paola Flocchini, Giuseppe Prencipe, Nicola Santoro, and Masafumi Yamashita. The power of lights: Synchronizing asynchronous robots using visible bits. In *2012 IEEE 32nd International Conference on Distributed Computing Systems, Macau, China, June 18-21, 2012*, pages 506–515. IEEE Computer Society, 2012.

[53] Shantanu Das, Paola Flocchini, Giuseppe Prencipe, Nicola Santoro, and Masafumi Yamashita. Autonomous mobile robots with lights. *Theor. Comput. Sci.*, 609:171–184, 2016.

[54] Xavier Défago, Maria Gradinariu, Stéphane Messika, and Philippe Raipin Parvédy. Fault-tolerant and self-stabilizing mobile robots gathering. In Shlomi Dolev, editor, *Distributed Computing, 20th International Symposium, DISC 2006, Stockholm, Sweden, September 18-20, 2006, Proceedings*, volume 4167 of *Lecture Notes in Computer Science*, pages 46–60. Springer, 2006.

[55] Xavier Défago and Akihiko Konagaya. Circle formation for oblivious anonymous mobile robots with no common sense of orientation. In *Proceedings of the Second ACM International Workshop on Principles of Mobile Computing*, POMC '02, page 97–104, New York, NY, USA, 2002. Association for Computing Machinery.

[56] Mattia D'Emidio, Gabriele Di Stefano, Daniele Frigioni, and Alfredo Navarra. Characterizing the computational power of mobile robots on graphs and implications for the euclidean plane. *Inf. Comput.*, 263:57–74, 2018.

[57] Stéphane Devismes, Anissa Lamani, Franck Petit, Pascal Raymond, and Sébastien Tixeuil. Terminating exploration of A grid by an optimal number of asynchronous oblivious robots. *Comput. J.*, 64(1):132–154, 2021.

[58] Yoann Dieudonné and Franck Petit. Scatter of robots. *Parallel Process. Lett.*, 19(1):175–184, 2009.

[59] Yoann Dieudonné, Franck Petit, and Vincent Villain. Leader election problem versus pattern formation problem. In Nancy A. Lynch and Alexander A. Shvartsman, editors, *Distributed Computing*, pages 267–281, Berlin, Heidelberg, 2010. Springer Berlin Heidelberg.

[60] Yoann Dieudonné and Franck Petit. Self-stabilizing gathering with strong multiplicity detection. *Theoretical Computer Science*, 428:47–57, 2012.

[61] Ayan Dutta, Vladimir Ufimtsev, Tuffa Said, Inmo Jang, and Roger Eggen. Distributed hedonic coalition formation for multi-robot task allocation. In *17th IEEE International Conference on Automation Science and Engineering, CASE 2021, Lyon, France, August 23-27, 2021*, pages 639–644. IEEE, 2021.

[62] Asaf Efrima and David Peleg. Distributed algorithms for partitioning a swarm of autonomous mobile robots. *Theor. Comput. Sci.*, 410(14):1355–1368, 2009.

[63] P. Flocchini, N. Santoro, G. Viglietta, and M. Yamashita. Rendezvous with constant memory. *Theoretical Computer Science*, 621:57–72, 2016.

[64] Paola Flocchini, David Ilcinkas, Andrzej Pelc, and Nicola Santoro. Computing without communicating: Ring exploration by asynchronous oblivious robots. *Algorithmica*, 65(3):562–583, 2013.

[65] Paola Flocchini, Giuseppe Prencipe, and Nicola Santoro. *Distributed Computing by Oblivious Mobile Robots*. Synthesis Lectures on Distributed Computing Theory. Morgan & Claypool Publishers, 2012.

[66] Paola Flocchini, Giuseppe Prencipe, and Nicola Santoro. Distributed computing by mobile entities. *Current Research in Moving and Computing*, 11340, 2019.

[67] Paola Flocchini, Giuseppe Prencipe, Nicola Santoro, and Peter Widmayer. Gathering of asynchronous oblivious robots with limited visibility. In Afonso Ferreira

and Horst Reichel, editors, *STACS 2001, 18th Annual Symposium on Theoretical Aspects of Computer Science, Dresden, Germany, February 15-17, 2001, Proceedings*, volume 2010 of *Lecture Notes in Computer Science*, pages 247–258. Springer, 2001.

[68] Paola Flocchini, Giuseppe Prencipe, Nicola Santoro, and Peter Widmayer. Gathering of asynchronous robots with limited visibility. *Theor. Comput. Sci.*, 337(1-3):147–168, 2005.

[69] Paola Flocchini, Giuseppe Prencipe, Nicola Santoro, and Peter Widmayer. Arbitrary pattern formation by asynchronous, anonymous, oblivious robots. *Theor. Comput. Sci.*, 407(1-3):412–447, 2008.

[70] Paola Flocchini, Nicola Santoro, Giovanni Viglietta, and Masafumi Yamashita. Rendezvous of two robots with constant memory. In Thomas Moscibroda and Adele A. Rescigno, editors, *Structural Information and Communication Complexity*, pages 189–200, Cham, 2013. Springer International Publishing.

[71] Paola Flocchini, Nicola Santoro, Giovanni Viglietta, and Masafumi Yamashita. Rendezvous with constant memory. *Theor. Comput. Sci.*, 621:57–72, 2016.

[72] Nao Fujinaga, Yukiko Yamauchi, Hirotaka Ono, Shuji Kijima, and Masafumi Yamashita. Pattern formation by oblivious asynchronous mobile robots. *SIAM Journal on Computing*, 44(3):740–785, 2015.

[73] Pritam Goswami, Satakshi Ghosh, Avisek Sharma, and Buddhadeb Sau. Gathering on an infinite triangular grid with limited visibility. *CoRR*, abs/2204.14042, 2022.

[74] Samuel Guilbault and Andrzej Pelc. Gathering asynchronous oblivious agents with local vision in regular bipartite graphs. *Theor. Comput. Sci.*, 509:86–96, 2013.

[75] K Haba, Taisuke Izumi, Yoshiaki Katayama, Nobuhiro Inuzuka, and Koichi Wada. On gathering problem in a ring for 2n autonomous mobile robots. *Proceedings of the 10th International Symposium on Stabilization, Safety, and Security of Distributed Systems (SSS), poster*, 01 2008.

[76] Rory Hector, Ramachandran Vaidyanathan, Gokarna Sharma, and Jerry L. Trahan. Optimal convex hull formation on a grid by asynchronous robots with lights. *IEEE Trans. Parallel Distributed Syst.*, 33(12):3532–3545, 2022.

[77] Adam Heriban, Xavier Défago, and Sébastien Tixeuil. Optimally gathering two robots. In Paolo Bellavista and Vijay K. Garg, editors, *Proceedings of the 19th International Conference on Distributed Computing and Networking, ICDCN 2018, Varanasi, India, January 4-7, 2018*, pages 3:1–3:10. ACM, 2018.

[78] Anthony Honorat, Maria Potop-Butucaru, and Sébastien Tixeuil. Gathering fat mobile robots with slim omnidirectional cameras. *Theor. Comput. Sci.*, 557:1–27, 2014.

[79] Taisuke Izumi, Maria Gradinariu Potop-Butucaru, and Sébastien Tixeuil. Connectivity-preserving scattering of mobile robots with limited visibility. In Shlomi Dolev, Jorge Arturo Cobb, Michael J. Fischer, and Moti Yung, editors, *Stabilization, Safety, and Security of Distributed Systems - 12th International Symposium, SSS 2010, New York, NY, USA, September 20-22, 2010. Proceedings*, volume 6366 of *Lecture Notes in Computer Science*, pages 319–331. Springer, 2010.

[80] Taisuke Izumi, Samia Souissi, Yoshiaki Katayama, Nobuhiro Inuzuka, Xavier Défago, Koichi Wada, and Masafumi Yamashita. The gathering problem for two oblivious robots with unreliable compasses. *SIAM J. Comput.*, 41(1):26–46, 2012.

[81] Tomoko Izumi, Taisuke Izumi, Sayaka Kamei, and Fukuhito Ooshita. Mobile robots gathering algorithm with local weak multiplicity in rings. In Boaz Patt-Shamir and Tínaz Ekim, editors, *Structural Information and Communication Complexity, 17th International Colloquium, SIROCCO 2010, Sirince, Turkey, June 7-11, 2010. Proceedings*, volume 6058 of *Lecture Notes in Computer Science*, pages 101–113. Springer, 2010.

[82] Tomoko Izumi, Taisuke Izumi, Sayaka Kamei, and Fukuhito Ooshita. Time-optimal gathering algorithm of mobile robots with local weak multiplicity detection in rings. *IEICE Trans. Fundam. Electron. Commun. Comput. Sci.*, 96-A(6):1072–1080, 2013.

[83] Sayaka Kamei, Anissa Lamani, Fukuhito Ooshita, and Sébastien Tixeuil. Asynchronous mobile robot gathering from symmetric configurations without global multiplicity detection. In Adrian Kosowski and Masafumi Yamashita, editors, *Structural Information and Communication Complexity - 18th International Colloquium, SIROCCO 2011, Gdansk, Poland, June 26-29, 2011. Proceedings*, volume 6796 of *Lecture Notes in Computer Science*, pages 150–161. Springer, 2011.

[84] Sayaka Kamei, Anissa Lamani, Fukuhito Ooshita, and Sébastien Tixeuil. Gathering an even number of robots in an odd ring without global multiplicity detection. In Branislav Rovan, Vladimiro Sassone, and Peter Widmayer, editors, *Mathematical Foundations of Computer Science 2012 - 37th International Symposium, MFCS 2012, Bratislava, Slovakia, August 27-31, 2012. Proceedings*, volume 7464 of *Lecture Notes in Computer Science*, pages 542–553. Springer, 2012.

[85] Sayaka Kamei, Anissa Lamani, Fukuhito Ooshita, Sébastien Tixeuil, and Koichi Wada. Gathering on rings for myopic asynchronous robots with lights. In Pascal Felber, Roy Friedman, Seth Gilbert, and Avery Miller, editors, *23rd International Conference on Principles of Distributed Systems, OPODIS 2019, December 17-19, 2019, Neuchâtel, Switzerland*, volume 153 of *LIPIcs*, pages 27:1–27:17. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2019.

[86] Sayaka Kamei, Anissa Lamani, Fukuhito Ooshita, Sébastien Tixeuil, and Koichi Wada. Asynchronous gathering in a torus. In Quentin Bramas, Vincent Gramoli, and Alessia Milani, editors, *25th International Conference on Principles of Distributed Systems, OPODIS 2021, December 13-15, 2021, Strasbourg, France*, volume 217 of *LIPIcs*, pages 9:1–9:17. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2021.

[87] Ralf Klasing, Adrian Kosowski, and Alfredo Navarra. Taking advantage of symmetries: Gathering of many asynchronous oblivious robots on a ring. *Theor. Comput. Sci.*, 411(34-36):3235–3246, 2010.

[88] Ralf Klasing, Euripides Markou, and Andrzej Pelc. Gathering asynchronous oblivious mobile robots in a ring. *Theor. Comput. Sci.*, 390(1):27–39, 2008.

[89] Martyna Koreń. Gathering small number of mobile asynchronous robots on ring. *Zeszyty Naukowe Wydziału ETI Politechniki Gdańskiej. Technologie Informacyjne*, 18:325–331, 2010.

[90] Ajay D. Kshemkalyani, Anisur Rahaman Molla, and Gokarna Sharma. Dispersion of mobile robots on grids. In M. Sohel Rahman, Kunihiko Sadakane, and Wing-Kin Sung, editors, *WALCOM: Algorithms and Computation - 14th International Conference, WALCOM 2020, Singapore, March 31 - April 2, 2020, Proceedings*, volume 12049 of *Lecture Notes in Computer Science*, pages 183–197. Springer, 2020.

[91] Manash Kumar Kundu, Pritam Goswami, Satakshi Ghosh, and Buddhadeb Sau. Arbitrary pattern formation by asynchronous opaque robots on infinite grid, 2022.

[92] Manash Kumar Kundu, Pritam Goswami, Satakshi Ghosh, and Buddhadeb Sau. Arbitrary pattern formation by opaque fat robots on infinite grid. *Int. J. Parallel Emergent Distributed Syst.*, 37(5):542–570, 2022.

[93] Anissa Lamani, Maria Gradinariu Potop-Butucaru, and Sébastien Tixeuil. Optimal deterministic ring exploration with oblivious asynchronous robots. In Boaz Patt-Shamir and Tínaz Ekim, editors, *Structural Information and Communication Complexity, 17th International Colloquium, SIROCCO 2010, Sirince, Turkey, June 7-11, 2010. Proceedings*, volume 6058 of *Lecture Notes in Computer Science*, pages 183–196. Springer, 2010.

[94] Tamás Lukovszki and Friedhelm Meyer auf der Heide. Fast collisionless pattern formation by anonymous, position-aware robots. In Marcos K. Aguilera, Leonardo Querzoni, and Marc Shapiro, editors, *Principles of Distributed Systems - 18th International Conference, OPODIS 2014, Cortina d'Ampezzo, Italy, December 16-19, 2014. Proceedings*, volume 8878 of *Lecture Notes in Computer Science*, pages 248–262. Springer, 2014.

[95] Giuseppe Antonio Di Luna, Paola Flocchini, Sruti Gan Chaudhuri, Federico Poloni, Nicola Santoro, and Giovanni Viglietta. Mutual visibility by luminous robots without collisions. *Inf. Comput.*, 254:392–418, 2017.

[96] Giuseppe Antonio Di Luna, Paola Flocchini, Federico Poloni, Nicola Santoro, and Giovanni Viglietta. The mutual visibility problem for oblivious robots. In *Proceedings of the 26th Canadian Conference on Computational Geometry, CCCG 2014, Halifax, Nova Scotia, Canada, 2014*. Carleton University, Ottawa, Canada, 2014.

[97] Giuseppe Antonio Di Luna, Paola Flocchini, Nicola Santoro, and Giovanni Viglietta. Turingmobile: A turing machine of oblivious mobile robots with limited visibility and its applications. In Ulrich Schmid and Josef Widder, editors, *32nd International Symposium on Distributed Computing, DISC 2018, New Orleans, LA, USA, October 15-19, 2018*, volume 121 of *LIPIcs*, pages 19:1–19:18. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2018.

[98] Shota Nagahama, Fukuhito Ooshita, and Michiko Inoue. Terminating grid exploration with myopic luminous robots. In *IEEE International Parallel and Distributed Processing Symposium Workshops, IPDPS Workshops 2021, Portland, OR, USA, June 17-21, 2021*, pages 586–595. IEEE, 2021.

[99] Iñaki Navarro, Álvaro Gutiérrez, Fernando Matía, and Félix Monasterio-Huelin. An approach to flocking of robots using minimal local sensing and common orientation. In Emilio Corchado, Ajith Abraham, and Witold Pedrycz, editors, *Hybrid Artificial Intelligence Systems*, pages 616–624, Berlin, Heidelberg, 2008. Springer Berlin Heidelberg.

[100] Takashi Okumura, Koichi Wada, and Yoshiaki Katayama. Brief announcement: Optimal asynchronous rendezvous for mobile robots with lights. In Paul G. Spirakis and Philippas Tsigas, editors, *Stabilization, Safety, and Security of Distributed Systems - 19th International Symposium, SSS 2017, Boston, MA, USA, November 5-8, 2017, Proceedings*, volume 10616 of *Lecture Notes in Computer Science*, pages 484–488. Springer, 2017.

[101] Fukuhito Ooshita and Sébastien Tixeuil. On the self-stabilization of mobile oblivious robots in uniform rings. *Theor. Comput. Sci.*, 568:84–96, 2015.

[102] Linda Pagli, Giuseppe Prencipe, and Giovanni Viglietta. Getting close without touching: near-gathering for autonomous mobile robots. *Distributed Comput.*, 28(5):333–349, 2015.

[103] Debasish Pattanayak, Kaushik Mondal, H. Ramesh, and Partha Sarathi Mandal. Gathering of mobile robots with weak multiplicity detection in presence of crash-faults. *J. Parallel Distributed Comput.*, 123:145–155, 2019.

[104] Pavan Poudel and Gokarna Sharma. Time-optimal gathering under limited visibility with one-axis agreement. *Information*, 12(11), 2021.

[105] Pavan Poudel, Gokarna Sharma, and Aisha Aljohani. Sublinear-time mutual visibility for fat oblivious robots. In *Proceedings of the 20th International Conference on Distributed Computing and Networking*, ICDCN '19, page 238–247, New York, NY, USA, 2019. Association for Computing Machinery.

[106] Giuseppe Prencipe. Impossibility of gathering by a set of autonomous mobile robots. *Theor. Comput. Sci.*, 384(2-3):222–231, 2007.

[107] Arthur Rauch, Quentin Bramas, Stéphane Devismes, Pascal Lafourcade, and Anissa Lamani. Optimal exclusive perpetual grid exploration by luminous myopic robots without common chirality. In Karima Echihabi and Roland Meyer, editors, *Networked Systems - 9th International Conference, NETYS 2021, Virtual Event, May 19-21, 2021, Proceedings*, volume 12754 of *Lecture Notes in Computer Science*, pages 95–110. Springer, 2021.

[108] Madhumita Sardar, Deepanwita Das, and Srabani Mukhopadhyaya. Grid exploration by a swarm of autonomous robots with minimum repetitions. *Theoretical Computer Science*, 933:67–87, 2022.

[109] Paul Scharre. Robotics on the battlefield: Part i: Range, persistence and daring. Technical report, Center for a New American Security, 2014.

[110] Gokarna Sharma, Ramachandran Vaidyanathan, and Jerry L. Trahan. Constant-time complete visibility for robots with lights: The asynchronous case. *Algorithms*, 14(2):56, 2021.

[111] Gokarna Sharma, Ramachandran Vaidyanathan, and Jerry L. Trahan. Optimal randomized complete visibility on a grid for asynchronous robots with lights. *Int. J. Netw. Comput.*, 11(1):50–77, 2021.

[112] Masahiro Shibata, Masaki Ohyabu, Yuichi Sudo, Junya Nakamura, Yonghwan Kim, and Yoshiaki Katayama. Gathering of seven autonomous mobile robots

on triangular grids. In *IEEE International Parallel and Distributed Processing Symposium Workshops, IPDPS Workshops 2021, Portland, OR, USA, June 17-21, 2021*, pages 566–575. IEEE, 2021.

[113] Gabriele Di Stefano and Alfredo Navarra. Gathering of oblivious robots on infinite grids with minimum traveled distance. *Inf. Comput.*, 254:377–391, 2017.

[114] Gabriele Di Stefano and Alfredo Navarra. Optimal gathering of oblivious robots in anonymous graphs and its application on trees and rings. *Distributed Computing*, 30(2):75–86, 2017.

[115] D.P. Stormont. Autonomous rescue robot swarms for first responders. In *CIHSPS 2005. Proceedings of the 2005 IEEE International Conference on Computational Intelligence for Homeland Security and Personal Safety, 2005.*, pages 151–157, 2005.

[116] Ichiro Suzuki and Masafumi Yamashita. Distributed anonymous mobile robots. In Nicola Santoro and Paul G. Spirakis, editors, *SIROCCO'96, The 3rd International Colloquium on Structural Information & Communication Complexity, Siena, Italy, June 6-8, 1996*, pages 313–330. Carleton Scientific, 1996.

[117] Ichiro Suzuki and Masafumi Yamashita. Distributed anonymous mobile robots: Formation of geometric patterns. *SIAM J. Comput.*, 28(4):1347–1363, 1999.

[118] Ying Tan and Zhong yang Zheng. Research advance in swarm robotics. *Defence Technology*, 9(1):18–39, 2013.

[119] Giovanni Viglietta. Rendezvous of two robots with visible bits. In Paola Flocchini, Jie Gao, Evangelos Kranakis, and Friedhelm Meyer auf der Heide, editors, *Algorithms for Sensor Systems - 9th International Symposium on Algorithms and Experiments for Sensor Systems, Wireless Networks and Distributed Robotics, ALGOSENSORS 2013, Sophia Antipolis, France, September 5-6, 2013, Revised Selected Papers*, volume 8243 of *Lecture Notes in Computer Science*, pages 291–306. Springer, 2013.

[120] P.K.C. Wang. Navigation strategies for multiple autonomous mobile robots moving in formation. In *Proceedings. IEEE/RSJ International Workshop on Intelligent Robots and Systems '. (IROS '89) 'The Autonomous Mobile Robots and Its Applications*, pages 486–493, 1989.

[121] Naixue Xiong, Jing He, Yan Yang, Yanxiang He, Tai-Hoon Kim, and Chuan Lin. A survey on decentralized flocking schemes for a set of autonomous mobile robots (invited paper). *J. Commun.*, 5(1):31–38, 2010.

[122] Masafumi Yamashita and Ichiro Suzuki. Characterizing geometric patterns formable by oblivious anonymous mobile robots. *Theor. Comput. Sci.*, 411(26-28):2433–2453, 2010.

[123] Yukiko Yamauchi and Masafumi Yamashita. Pattern formation by mobile robots with limited visibility. In Thomas Moscibroda and Adele A. Rescigno, editors, *Structural Information and Communication Complexity*, pages 201–212, Cham, 2013. Springer International Publishing.