Two teams of oblivious asynchronous robots performing different tasks on an infinite grid without the knowledge of its team members*

Satakshi Ghosh[1][0000−0003−1747−4037], Avisek Sharma[1][0000−0001−8940−392X], Pritam Goswami[1][0000−0002−0546−3894], and Buddhadeb Sau[1][0000−0001−7008−6135]

Jadavpur University, Kolkata, 700032
{satakshighosh.math.rs,aviseks.math.rs,pritamgoswami.math.rs, buddhadeb.sau}@jadavpuruniversity.in

Abstract. Two fundamental problems of distributed computing are Gathering and Arbitrary pattern formation (Apf). These two tasks are different in nature as in gathering robots meet at a point but in Apf robots form a fixed pattern in distinct positions.
In most of the current literature on swarm robot algorithms, it is assumed that all robots in the system perform one single task together. Two teams of oblivious robots deployed in the same system and different teams of robots performing two different works simultaneously where no robot knows the team of another robot is a new concept in the literature introduced by Bhagat et al. [ICDCN'2020].
In this work, a swarm of silent and oblivious robots are deployed on an infinite grid under an asynchronous scheduler. The robots do not have access to any global coordinates. Some of the robots are given input of an arbitrary but unique pattern. The set of robots with the given pattern is assigned with the task of forming the given pattern on the grid. The remaining robots are assigned with the task of gathering to a vertex of the grid (not fixed from earlier and not any point where a robot that is forming a pattern terminates). Each robot knows to which team it belongs, but can not recognize the team of another robot. Considering weak multiplicity detection, a distributed algorithm is presented in this paper which leads the robots with the input pattern into forming it and other robots into gathering on a vertex of the grid on which no other robot forming the pattern, terminates.

Keywords: Robots · Gathering · Arbitrary pattern formation · Infinite grid

1 Introduction

In swarm robotics robots, solving some tasks with minimum capabilities is the main focus of interest. In the last two decades, there are huge research interest when robots working on coordination problems. It is not always easy to use

* The first three authors are full time research scholars of Jadavpur University

robots with strong capability in real-life applications, as the making of these robots is not at all cost-effective. If a swarm of robots with minimum capabilities can do the same task then it is effective to use swarm robots rather than using robots with many capabilities, as the making of these robots in the swarm is very much cheaper than making robots with many capabilities. Also, it is very easy to design a robot of a swarm due to the fact that they have minimum capabilities. Depending on these capabilities there are generally four types of robot models. These models are $\mathcal{OBLOT}$, $\mathcal{FSTA}$, $\mathcal{FCOM}$ and $\mathcal{LUMI}$. In each of these models robots are assumed to be autonomous (i.e the robots do not have any central control), identical (i.e the robots are physically indistinguishable), and anonymous (i.e the robots do not have any unique identifiers). Furthermore in the $\mathcal{OBLOT}$ model, the robots are silent (i.e there is no means of communication between the robots) and oblivious (i.e the robots do not have any persistent memory to remember their previous state), in $\mathcal{FSTA}$ model the robots are silent but not oblivious, in $\mathcal{FCOM}$ model the robots are oblivious but not silent and in $\mathcal{LUMI}$ model robots are neither silent nor oblivious. The robots after getting activated operates in a Look-Compute-Move (Lcm) cycle. In Look phase a robot takes input from its surroundings and then with that input runs the algorithm in Compute phase to get a destination point as an output. The robot then goes to that destination point by moving in the Move phase. The activation of the robots is controlled by a scheduler. There are mainly three types of schedulers considered in the literature. In a synchronous scheduler, time is divided into global rounds. In a fully synchronous (FSync) scheduler, each robot is activated in all the rounds and executes Lcm cycle simultaneously. In a semi-synchronous (SSync) scheduler all robots may not get activated in each round. But the robots that are activated in the same round execute the Lcm cycle simultaneously. Lastly in the asynchronous (ASync) scheduler, there is no common notion of time, a robot can be activated at any time. There is no concept of global rounds. So there is no assumption regarding synchronization.

The Gathering and Arbitrary pattern formation are two vastly studied problems by researchers in the field of swarm robot algorithms. These are one of the fundamental tasks which can be done by autonomous robots in different settings. In the gathering problem, n number of robots initially positioned arbitrarily meets at a point not fixed from earlier within finite time. It is not always easy to meet at a point with very weak robots in the distributed system. Similarly, the Arbitrary pattern formation problem is such that robots have to form a given pattern that is given as input to the robots within finite time. In literature, there are several works that have considered either gathering or arbitrary pattern formation problem separately. But none of those works consider robots deployed in the same environment working on two different tasks. An environment of robot swarm needs periodic maintenance for making the environment robust from faults and some other factors. So If the same robot swarm deployed in the environment can do the maintenance apart from doing the specific task assigned to them, it would be more cost-effective. From this motivation and practical interest, Bhagat et al. first studied the problem where two teams of

oblivious robots work on two different tasks ([2]), namely gathering and circle formation on a plane. Here the crucial part is that a robot knows that in which team it belongs to, but it can not recognize another robot's team. The novelty of the problem would have gone away a bit if the robots are luminous and gathering team robots put a color on a light to indicate which team they belong to. But the main motivation of this problem is to extend the work for more than two different tasks and for assigning different colors for different tasks would make the number of colours unbounded. For this reason, it is convenient to solve the problem with the least possible capabilities for the robot swarm. Also $\mathcal{OBLOT}$ model is more self-stabilized and fault tolerant. For these reasons, we also have considered robots to be oblivious to our work. Now it is challenging to design a distributed algorithm by which two different teams of oblivious robots can do two different tasks simultaneously on a discrete domain because, in any discrete graph, the movements of robots become restricted. So avoiding collision becomes a great challenge.

In our work, we have provided a collision-less distributed algorithm following which two teams of oblivious robots with weak multiplication detection ability can do two different tasks namely gathering and arbitrary pattern formation simultaneously on an infinite grid under the asynchronous scheduler if the initial configuration is asymmetric.

2 Earlier works

Arbitrary pattern formation and gathering of robots are two hugely studied problems in the distributed system. In literature, there are many works on these two problems in various settings.

Arbitrary pattern formation on a plane was first studied by Suzuki and Yamashita [12]. For the grid network, the arbitrary pattern formation was first studied in [3] in $\mathcal{OBLOT}$ model with full visibility. Later in [4] authors studied the APF on a regular tessellation graph. In an infinite grid, the arbitrary pattern formation problem was studied in [10] with opaque robots and in [11] with fat robots. Similarly, for gathering there are many works in an infinite grid. In [7], authors have shown that the gathering is possible on a grid without multiplicity detection. There are many other works ([8,1,5,6]) where authors have solved gathering on an infinite rectangular grid with various assumptions. In [9] authors solved the gathering of robots on an infinite triangular grid in SSYNC scheduler under one axis agreement and with one-hop visibility. But in all these works they only consider that all robots perform one individual task. But in [2] Authors first showed that two specific but different works can be done by robots simultaneously on a plane. They showed that gathering and circle formation can be done by oblivious robots in a plane simultaneously with two different teams of robots using one axis agreement.

So here we are interested to show that two different works namely gathering and the arbitrary pattern formation of oblivious robots can be done on an infinite grid simultaneously under an asynchronous scheduler without any

axis agreement. This is the first work that considers this problem in a discrete environment.

In [4], authors considered multiplicity points in the target pattern that needs to be formed. So their work is also capable of forming an arbitrary pattern along with an additional multiplicity point. But by following their algorithm a robot on team gathering might end up forming a pattern and also a robot on team arbitrary pattern formation might end up on the multiplicity point, which is not the required solution to our problem.

3 Model and problem statement

Robots: Robots are anonymous, identical, and oblivious, i.e. they have no memory of their past rounds. They can not communicate with each other. There are two teams between the robots. One is $\mathcal{T}_{Apf}$ and the other team is $\mathcal{T}_g$. A robot r only knows that in which team it belongs to between this two. But a robot can not identify to which team another robot belongs. All robots are initially in distinct positions on the grid. The robots can see the entire grid and all other robots' positions which means they have global visibility. This implies the robots are transparent and hence the visibility of a robot can not be obstructed by another robot. Robots have no access to any common global coordinate system. They have no common notion of chirality or direction. A robot has its local view and it can calculate the positions of other robots with respect to its local coordinate system with the origin at its own position. There is no agreement on the grid about which line is x or y-axis and also about the positive or negative direction of the axes. As the robots can see the entire grid, they will set the axes of their local coordinate systems along the grid lines. Also robots have weak multiplicity detection capability, that means a robot can detect a multiplicity point but cannot count the number of robot present at a multiplicity point.

Look-Compute-Move cycles: An active robot operates according to the Look-Compute-Move cycle. In each cycle a robot takes a snapshot of the positions of the other robots according to its own local coordinate system (Look); based on this snapshot, it executes a deterministic algorithm to determine whether to stay put or to move to an adjacent grid point (Compute); and based on the algorithm the robot either remain stationary or makes a move to an adjacent grid point (Move). When the robots are oblivious they have no memory of past configurations and previous actions. After completing each Look-Compute-Move cycle, the contents in each robot's local memory are deleted.

Scheduler: We assume that robots are controlled by a fully asynchronous adversarial scheduler (ASync). The robots are activated independently and each robot executes its cycles independently. This implies the amount of time spent in Look, Compute, Move and inactive states is finite but unbounded, unpredictable and not same for different robots. The robots have no common notion of time.

Movement: In discrete domains, the movements of robots are assumed to be instantaneous. This implies that the robots are always seen on grid points, not on edges. However, in our work, we do not need this assumption. In the proposed algorithm, we assume the movements are to be instantaneous for simplicity. However, this algorithm also works without this. The movement of the robots is restricted from one grid point to one of its four neighboring grid points.

3.1 Problem description

In this work, we define a problem on an infinite grid where n oblivious, identical, autonomous robots are dispersed on the vertices of the infinite grid. In [2] they solved two conflicting tasks by robots on a plane where one team of robots gather at a point and another team of robots form a circle on the plane. But in this work robots are on an infinite grid and they solve two different distributed problems. Here are the two teams of oblivious robots where one team is $\mathcal{T}_g$ and the other team is $\mathcal{T}_{Apf}$. The goal of the robots of $\mathcal{T}_g$ is to meet all the robots of this team to a point on an infinite grid. Another team is $\mathcal{T}_{Apf}$ where the goal of this team is to form a particular pattern that is given as input. Next section provides the algorithm for solving this problem.

4 The Main Algorithm

4.1 Global coordinate agreement

Here we have to first fix the global coordinate system and then we aim to gather the $\mathcal{T}_g$ robots to the origin and form the arbitrary pattern by $\mathcal{T}_{Apf}$ robots with respect to the coordinate $(0, 2)$. So let us consider an infinite grid G as a cartesian product $P \times P$, where P is an infinite (from both sides) path graph. The infinite grid G is embedded in the Cartesian Plane R^2. In this work, some robots will gather at a point and other robots will form an arbitrary pattern on the grid. But as we know that arbitrary pattern formation and gathering is solvable depending on the symmetries of the initial configuration of the robots. So we are assuming that the initial configuration is asymmetric. A robot can form a local coordinate system aligning the axes along the grid lines but the robots do not have an access to any global coordinate system even. To form the target pattern the robots need to reach an agreement on a global coordinate system. In this subsection, we will provide the details of the procedure that allows the robots to reach an agreement on a global coordinate system.

For a given configuration ($\mathcal{C}$) formed by the robots, let the smallest enclosing rectangle, denoted by $s.rect(\mathcal{C})$, be the smallest grid-aligned rectangle that contains all the robots. Suppose the $s.rect$ of the initial configuration $\mathcal{C}_I$ is a rectangle $\mathcal{R} = ABCD$ of size $m \times n$, such that $m > n > 1$. Let $|AB| = n$. Then consider the binary string $\{p_i\}$ associated with a corner A, λ_{AB} as follows. Scan the grid from A along the side AB to B and sequentially all grid lines of $s.rect(\mathcal{C}_\mathcal{I})$ parallel to AB in the same direction. And $p_i = 0$, if the position is

unoccupied and $p_i = 1$ otherwise. Similarly construct the other binary strings λ_{BA}, λ_{CD} and λ_{DC}. Since the initial configuration is asymmetric we can find a unique lexicographically largest string. If λ_{AB} is the lexicographically largest string, then A is called the leading corner of $\mathcal{R}$.

Next, suppose $\mathcal{R}$ is an $m \times m$ square, then consider the eight binary strings $\lambda_{AB}, \lambda_{BA}, \lambda_{CD}, \lambda_{DC}, \lambda_{BC}, \lambda_{CB}, \lambda_{AD}, \lambda_{DA}$. Again since the initial configuration is asymmetric, we can find a unique lexicographically largest string among them. Hence we can find a leading corner here as well.

Next, let $\mathcal{C}_I$ be a line AB, we will have two strings λ_{AB} and λ_{BA}. Since $\mathcal{C}_I$ is asymmetric then λ_{AB} and λ_{BA} must be distinct. If λ_{AB} is lexicographically larger than λ_{BA}, then we choose A as the leading corner.

Now for either case, if λ_{AB} is the lexicographically largest string then the leading corner A is considered as the origin, and the $x-$ axis is taken along the AB line. If $\mathcal{C}_I$ is not a line then the $y-$ axis is taken along the AD line. If λ_{AB} is a line then the $y-$ coordinate of all the positions of robots is going to be zero and in this case, the $y-$ axis will be determined later. For any given asymmetric configuration $\mathcal{C}$ if λ_{AB} is the largest associated binary string to $\mathcal{C}$ then the robot causing first non zero entry in λ_{AB} is called *head* let $\mathcal{H}$ and the robot causing last non zero entry in λ_{AB} is called as *tail* let $\mathcal{T}$. We denote the i^{th} robot of the λ_{AB} string as r_{i-1}. A robot other than the head and tail are called ***Inner robot***. Further we denote $\mathcal{C}' = \mathcal{C} \setminus \{tail\}$ and $\mathcal{C}'' = \mathcal{C} \setminus \{tail, head\}$ and $\mathcal{C}''' = \mathcal{C} \setminus \{Head\}$.

Let $\mathcal{C}_{target}$ be the target configuration for the $\mathcal{T}_{Apf}$ robots and $s.rect(\mathcal{C}_{target}) = \mathcal{R}_{target}$. Let $\mathcal{R}_{target}$ is a rectangle of size $M \times N$ with $M \geq N$. We can calculate the binary strings associated with corners in the same manner as previously. $\mathcal{C}_{target}$ is expressed in the coordinate system with respect to the point $(0, 2)$, where $(0, 2)$ will be the leading corner. Let the $A'B'C'D'$ be the smallest rectangle enclosing the target pattern with $A'B' \leq B'C'$. Let $\lambda_{A'B'}$ be the largest (may not be unique) among all other strings. Then the target pattern is to be formed such that A is the origin and pattern embedded with respect to the position $(0, 2)$, $A'B'$ direction is along the positive x axis and $A'D'$ direction is along the positive y axis. If the target pattern has a symmetry then we have to choose any one among the larger string and fixed the coordinate system. So as previously said $head_{target}$ will be the first one and $tail_{target}$ will be the last one in the $s.rect$ of $\mathcal{C}_{target}$. Also, we define $\mathcal{C}_{final}$ is the configuration when all $\mathcal{T}_g$ robots are at same point and $\mathcal{C}_{target}$ configuration is formed. $\mathcal{C}'_{final} = \mathcal{C}_{final} \setminus \{tail_{target}\}$, $\mathcal{C}''_{final} = \mathcal{C}_{final} \setminus \{head_{target}, tail_{target}\}$, $\mathcal{C}'''_{final} = \mathcal{C}_{final} \setminus \{head_{target}\}$. We denote the $head_{target}$ position as t_1 and $tail_{target}$ position as t_k. Let H_i be the horizontal line having the height i from the x-axis. Let for each i there are $p(i)$ target positions on H_i. We denote the target positions of H_0 as $t_1.....t_{p(0)-1}$ from left to right. For H_1 we denote the target positions as $t_{p(0)}$ to $t_{p(0)+p(1)-1}$ from right to left. For H_2 we denote the target positions as $t_{p(0)+p(1)}$ to $t_{p(0)+p(1)+p(2)-1}$ from right to left. Similarly we can denote all other target positions on H_i, $i > 0$ except $tail_{target}$.

4.2 Brief discussion of algorithm

Let the initial configuration is $\mathcal{C}_{\mathcal{I}}$, the final configuration is $\mathcal{C}_{final}$. Robots are operating on an infinite grid. There are two teams of robots where one is $\mathcal{T}_g$ and another one is $\mathcal{T}_{Apf}$ (let us assume that $|\mathcal{T}_g| > 2$). $\mathcal{T}_g$ robots will gather at the same point on the grid and the robots of $\mathcal{T}_{Apf}$ will form a pattern on the grid. Note that the gathering point will not be a target point of the target pattern. A robot only knows in which team it belongs between $\mathcal{T}_g$ and $\mathcal{T}_{Apf}$. A robot has no information about to which team other robots belong. Robots first fix the global coordinate system. The target will form with respect to the point (0,2) and the gathering will occur at the origin. When a robot awakes up it first calculates the head robot and tail robot. In the first three phases, the head robot will move to the origin and the tail robot will expand the smallest enclosing rectangle for maintaining the asymmetry of the configuration. All the inner robots now move to the x-axis and make a line on the x-axis. Note that a line is called ***Compact line*** when there is no empty grid point between two robots. So robots on the x-axis first make the line compact then one by one robots from upward horizontal lines move down to the x-axis. After this, the robot which is not on a line say r, will move to its closest end point of the line or it will move one step upward if it belongs to. After a multiplicity point formed or calculating the position of the tail, robots on the x-axis move to the fixed target positions which are either origin or the target positions. Robots will move from right to left with respect to the position of the head sequentially. As all inner robots are on the x-axis so the robot $\in \mathcal{T}_g$ can always find the neighboring grid lines empty, so the robot can move to the origin by choosing any path to the origin if all the robots are on a line. If a robot can see the tail robot then it will choose the neighboring line of the x-axis in the direction of the tail for its movement. In this way, one by one all inner robots move to their fixed target positions. If tail sees that all inner robots move to its target positions it will move to its fixed position. If the head robot is in the gathering team then it will not move but if it is in the APF team then when without its position all the pattern formation is done it will move to its position. Note that within the algorithm no two robots collide and the coordinate system remains unchanged. Within finite time all $\mathcal{T}_g$ robots move to the same point and the $\mathcal{T}_{Apf}$ robots form the fixed pattern that is given as input.

4.3 Description of the stages

The main difficulty of this problem is a robot does not know which team another robot belongs to. The robot only knows about its own team. Here we will show that there will not arise any symmetry and no collision will occur. The global coordinate system will also not change. The algorithm is divided into five phases. In this situation, the global coordinate will fix as we mention in sec 4.1

Stage-1: Input: $\{P_4 \wedge \neg P_{14} \wedge \neg(P_5 \wedge P_6)\}$ is true.
The tail robot will move upwards and all other robots will remain static.

P_0	$\mathcal{C} = \mathcal{C}_{final}$
P_1	$\mathcal{C}' = \mathcal{C}'_{final}$
P_2	All $\mathcal{T}_g$ robots are at a same point
P_3	$\mathcal{C}_{Apf} = \mathcal{C}_{target}$
P_4	The current configuration is asymmetric
P_5	$m = \max\{N, n\} + 2$
P_6	$m = 2.\max\{M, V\}$ where V is the length of the vertical side of the smallest enclosing rectangle of $\mathcal{C}'$
P_7	The head in $\mathcal{C}$ is at the origin
P_8	$n \geq \max\{N+1, H+1, k\}$ where H is the length of the horizontal side of the smallest enclosing rectangle of $\mathcal{C}'$
P_9	Line formation on x-axis without tail
P_{10}	$\mathcal{C}'$ has a non-trivial reflectional symmetry with respect to a vertical line
P_{11}	$\mathcal{C}''' = \mathcal{C}'''_{final}$
P_{12}	$\mathcal{C}'' = \mathcal{C}''_{final}$
P_{13}	$m \geq \max\{N, n\} + 2$, $m \geq 2.\max\{M, V\}$ and m is odd
P_{14}	There exist a multiplicity point

Table 1. Boolean functions

Aim: The aim is to make $\{P_5 \wedge P_6\}$ = true. After finite number of moves by tail robot stage-1 completes with $\{P_4 \wedge \neg P_{14} \wedge P_5 \wedge P_6\}$ is true.

Stage-2: Input: $\{P_4 \wedge \neg P_{14} \wedge P_5 \wedge P_6 \wedge \neg P_7\}$ is true.
The head robot will move to origin. So head robot will move towards left if it is not initially at the origin. In this move head robot will remain head.
Aim: After finite number of moves by the head robot stage-2 completes when $\{P_4 \wedge \neg P_{14} \wedge P_5 \wedge P_6 \wedge P_7\}$ is true.

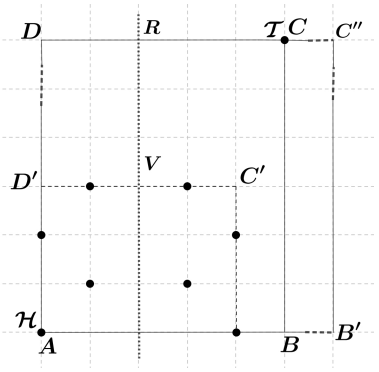

Fig. 1. Case-1: $\mathcal{C}'$ has a vertical symmetry and tail will move rightwards

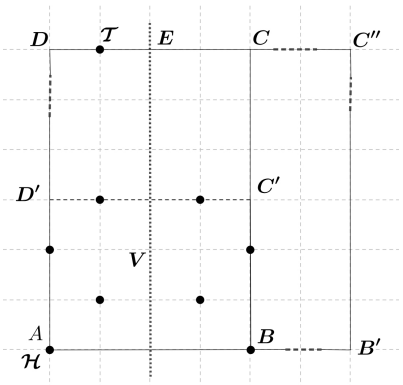

Fig. 2. Case-2: $\mathcal{C}'$ has a vertical symmetry and tail will move leftwards

Stage-3 Input: $\{P_4 \wedge \neg P_{14} \wedge P_5 \wedge P_6 \wedge P_7 \wedge \neg P_8\}$ is true.
The tail robot will move rightwards.
Aim: The main aim of this stage is to make P_8 true. After movement of robots $\{P_4 \wedge \neg P_{14} \wedge P_5 \wedge P_6 \wedge P_7 \wedge P_8\}$ is true.
In this stage the robots will check if P_{10} is true or false. If P_{10} is true then there may arise two cases:

Case-1: If tail is in the rightwards of the vertical symmetric line they it will move and make P_8 true.

Case-2: If the tail robot is in the left direction of the vertical symmetric line then it will not move rightwards. Tail will then move in leftwards upto one more step than D (fig 2). In this case the co-ordinate system will be changed.

Stage-4 Input:$\{P_4 \wedge \neg P_{14} \wedge P_5 \wedge P_6 \wedge P_7 \wedge P_8 \wedge \neg P_9\}$ is true.
The inner robots will move to the x-axis and form a line. Other than the tail robot, all other inner robots form a line on the x-axis. In phase four, the head is in origin, and all the robots on the x-axis first make the line compact, i.e. there is no empty grid point between two robots. After the x-axis becomes compact, when a robot r_i is on H_i and there are no robots in between H_i and the x-axis and the right part of r_i is empty in its horizontal line, then the robot moves to the x-axis. This procedure is done one by one by robots. In between this movement, no collision will occur. So all the inner robots other than the tail form a line on the x-axis.
Aim: When all the inner robots move to x-axis then $\{P_4 \wedge \neg P_{14} \wedge P_5 \wedge P_6 \wedge P_7 \wedge P_8 \wedge P_9\}$ is true.

Stage-5 Input: $P_4 \wedge \neg P_{14} \wedge \neg P_{13} \wedge P_7 \wedge P_9$ is true.
One robot which is not on a compact line will move to its closest end point of that line following the shortest path if it $\in \mathcal{T}_g$. As without one robot all other robots are on line so that one robot will move downwards. Then P_{14} is true. Or when $(P_4 \wedge \neg P_{14} \wedge P_7 \wedge P_9) \wedge (P_5 \wedge P_6 \wedge P_8)$ is true then that one robot will move upward from its position if it $\in \mathcal{T}_{Apf}$. By this move P_{13} will be true.
Aim: $P_4 \wedge \neg P_{14} \wedge P_7 \wedge P_9 \wedge (P_{13} \vee P_{14})$ is true..
After this phase if P_{14} is true then all robots will fix the multiplicity point as origin, the compact line is as x axis and the other perpendicular axis as y axis.

Stage-6 Input: $P_{13} \vee P_{14}$ is true.
When the tail robot will see that P_{13} is true then it will not move. An inner robot when sees that it is the rightmost robot on x-axis and P_{13} is true and there exists robots at the positions $t_{k-1}, t_{k-2}.....t_{k-i+1}$ where $1 \leq i \leq k$ (let t_k be the position of tail) then that inner robot will move upwards to t_{k-i} if it $\in \mathcal{T}_{Apf}$. If the robot $\in \mathcal{T}_g$ then it will move to the origin. Note that in this case robot can fix the global coordinate so it will move to the origin by choosing the first horizontal line in the positive direction of the y-axis. When a rightmost robot of

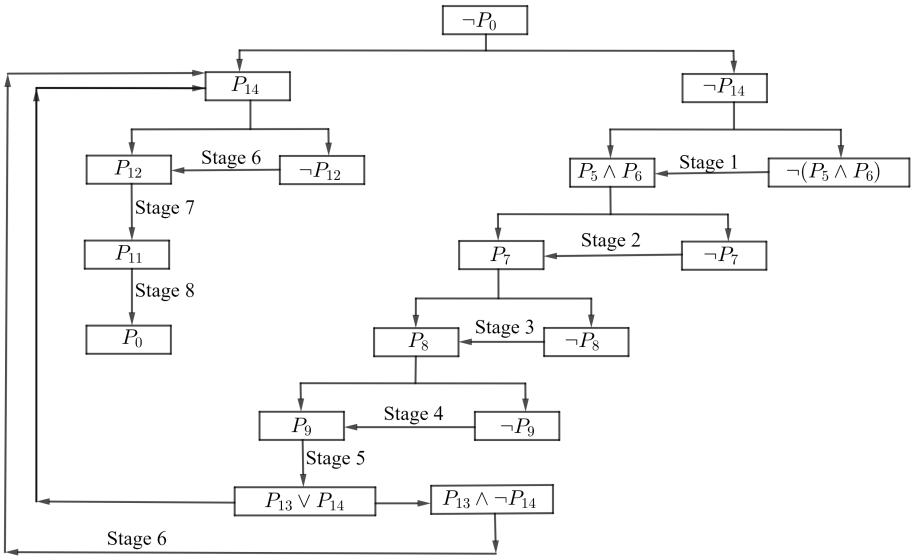

Fig. 3. Flow chart of the algorithm

team Apf on x-axis sees that P_{13} is true and no robot on target positions then it will move to t_{k-1}. Note that when a left most robot without head on x axis of Apf team sees that only $tail_{target}$ or $tail_{target}$ and $head_{target}$ are not occupied by robots then the robot will move to $tail_{target}$. Again if P_{14} is true then the rightmost robot on x-axis sees that there exist robots on $t_k, t_{k-1}.....t_{k-i+1}$ where $1 \leq i \leq k$ then that inner robot moves upward to t_{k-i} if it $\in \mathcal{T}_{Apf}$. If it $\in \mathcal{T}_g$ then move to multiplicity point. Here when a robot can see all robots on a line then it will fix that line as the x-axis and anyone perpendicular line as the y-axis and the multiplicity point as the origin. So the robot $\in \mathcal{T}_g$ moves to the origin by choosing the first horizontal line in the positive direction of y-axis. In this way, the inner robots move to their target positions one by one.
Aim: After this stage $P_{14} \wedge P_{12}$ is true.
After this stage, the multiplicity will be the origin and as the target pattern will be formed with respect to $(0, 2)$ so either there will be a robot at $tail_{target}$ or the tail robot will be at a position maintaining P_{13} true. So we can choose the larger side as the y-axis and the other one as the x-axis.

*Stage-7 Input:*In this stage $P_{14} \wedge P_{12} \wedge \neg P_{11}$ is true.
If $tail \in \mathcal{T}_{Apf}$ then move to the position of the tail target. But if $tail \in \mathcal{T}_g$ then the tail will move to the origin. If $tail \in \mathcal{T}_g$ then the tail will move downwards up to the x-axis and then move towards the multiplicity point (By P_{13} condition we can say that there will be no robot in this path of tail's movement).
Aim: P_{11} is true.

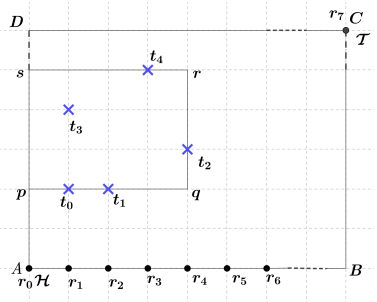

Fig. 4. Any $r_i \in \mathcal{T}_g$ moves to A and $r_i \in \mathcal{T}_{Apf}$ moves to fixed target positions one by one.

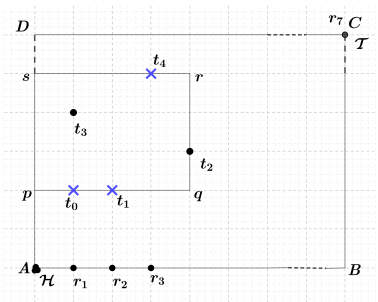

Fig. 5. A multiplicity point at A and some of the $\mathcal{T}_{Apf}$ robots form the pattern

Stage-8 Input: $P_{11} \wedge P_{14}$ is true.
If head robot is in gather team then it will not move. So then P_0 is true. But if it belongs to $\mathcal{T}_{Apf}$ then it moves to upward and then to the position of $head_{target}$.
Aim: After the movement of head then P_0 is true.

So after completing of these stages P_2 and P_3 conditions will be true. No collision will occur during the movement of robots throughout the algorithm. So the gathering and arbitrary pattern formation will be done by the robots simultaneously on an infinite grid by oblivious robots.
The proposed algorithm is depicted in the flowchart in fig 3. Starting from any configuration where $\neg P_0$ from the diagram fig 3 each directed path starting from the node where $\neg P_0$ ends at the node where P_0 is true. Hence we can conclude the theorem.

Theorem 1. *Gathering and Arbitrary pattern formation is solvable in* Async *by* $\mathcal{T}_{Apf}$ *and* $\mathcal{T}_g$ *robots from any asymmetric initial configuration.*

5 Conclusion

In this work, we claim that by our algorithm two different teams of robots can simultaneously gather and form an arbitrary pattern on an infinite grid. A robot only knows that in which team it belongs to between gathering and Apf. To our knowledge in a grid network, this is the first work where two different tasks are performed simultaneously by two different teams of robots. Here we assume the grid is infinite and the visibility is full. So for further work, we can extend this work by assuming the limited visibility and also when the grid is finite.

References

1. Bhagat, S., Chakraborty, A., Das, B., Mukhopadhyaya, K.: Gathering over meeting nodes in infinite grid. In: Conference on Algorithms and Discrete Applied Mathematics. pp. 318–330. Springer (2020)
2. Bhagat, S., Flocchini, P., Mukhopadhyaya, K., Santoro, N.: Weak robots performing conflicting tasks without knowing who is in their team. In: Mukherjee, N., Pemmaraju, S.V. (eds.) ICDCN 2020: 21st International Conference on Distributed Computing and Networking, Kolkata, India, January 4-7, 2020. pp. 29:1–29:6. ACM (2020). https://doi.org/10.1145/3369740.3369794, https://doi.org/10.1145/3369740.3369794
3. Bose, K., Adhikary, R., Kundu, M.K., Sau, B.: Arbitrary pattern formation on infinite grid by asynchronous oblivious robots. Theor. Comput. Sci. **815**, 213–227 (2020). https://doi.org/10.1016/j.tcs.2020.02.016, https://doi.org/10.1016/j.tcs.2020.02.016
4. Cicerone, S., Fonso, A.D., Stefano, G.D., Navarra, A.: Arbitrary pattern formation on infinite regular tessellation graphs. CoRR **abs/2010.14152** (2020), https://arxiv.org/abs/2010.14152
5. Cord-Landwehr, A., Fischer, M., Jung, D., Meyer auf der Heide, F.: Asymptotically optimal gathering on a grid. In: Proceedings of the 28th ACM Symposium on Parallelism in Algorithms and Architectures. pp. 301–312 (2016)
6. Das, S., Giachoudis, N., Luccio, F.L., Markou, E.: Gathering of robots in a grid with mobile faults. In: International Conference on Current Trends in Theory and Practice of Informatics. pp. 164–178. Springer (2019)
7. d'Angelo, G., Stefano, G.D., Klasing, R., Navarra, A.: Gathering of robots on anonymous grids without multiplicity detection. In: International Colloquium on Structural Information and Communication Complexity. pp. 327–338. Springer (2012)
8. Fischer, M., Jung, D., Meyer auf der Heide, F.: Gathering anonymous, oblivious robots on a grid. In: International Symposium on Algorithms and Experiments for Sensor Systems, Wireless Networks and Distributed Robotics. pp. 168–181. Springer (2017)
9. Goswami, P., Ghosh, S., Sharma, A., Sau, B.: Gathering on an infinite triangular grid with limited visibility. CoRR **abs/2204.14042** (2022). https://doi.org/10.48550/arXiv.2204.14042, https://doi.org/10.48550/arXiv.2204.14042
10. Kundu, M.K., Goswami, P., Ghosh, S., Sau, B.: Arbitrary pattern formation by asynchronous opaque robots on infinite grid. CoRR **abs/2205.03053** (2022). https://doi.org/10.48550/arXiv.2205.03053, https://doi.org/10.48550/arXiv.2205.03053
11. Kundu, M.K., Goswami, P., Ghosh, S., Sau, B.: Arbitrary pattern formation by opaque fat robots on infinite grid. Int. J. Parallel Emergent Distributed Syst. **37**(5), 542–570 (2022). https://doi.org/10.1080/17445760.2022.2088750, https://doi.org/10.1080/17445760.2022.2088750
12. Suzuki, I., Yamashita, M.: Distributed anonymous mobile robots - formation and agreement problems. In: Problems, in the Proceedings of the 3rd International Colloquium on Structural Information and Communication Complexity (SIROCCO '96. pp. 1347–1363 (1996)