Maximum Independent Set Formation on a Finite Grid by Myopic Robots

Raja Das  
rajad.math.rs@jadavpuruniversity.in  
Jadavpur University  
Kolkata, India

Avisek Sharma  
aviseks.math.rs@jadavpuruniversity.in  
Jadavpur University  
Kolkata, India

Buddhadeb Sau  
buddhadeb.sau@jadavpuruniversity.in  
Jadavpur University  
Kolkata, India

ABSTRACT
This work deals with the Maximum Independent Set (MIS) formation problem in a finite rectangular grid by autonomous robots. Suppose we are given a set of identical robots, where each robot is placed on a node of a finite rectangular grid $G$ such that no two robots are on the same node. The MIS formation problem asks to design an algorithm, executing which each robot will move autonomously and terminate at a node such that after a finite time the set of nodes occupied by the robots is a maximum independent set of $G$. We assume that robots are anonymous and silent, and they execute the same distributed algorithm.

Previous works solving this problem used one or several door nodes through which the robots enter inside the grid or the graph one by one and occupy required nodes. In this work, we propose a deterministic algorithm that solves the MIS formation problem in a more generalized scenario, i.e., when the total number of required robots to form an MIS are arbitrarily placed on the grid. The proposed algorithm works under a semi-synchronous scheduler using robots with only 2 hop visibility range and only 3 colors.

KEYWORDS
Myopic robot, Maximum Independent Set, Finite Grid, Autonomous robots, Robot with lights, Distributed algorithms

ACM Reference Format:
Raja Das, Avisek Sharma, and Buddhadeb Sau. 2018. Maximum Independent Set Formation on a Finite Grid by Myopic Robots. In Proceedings of Make sure to enter the correct conference title from your conference confirmation email (Conference acronym 'XX). ACM, New York, NY, USA, 10 pages. https://doi.org/XXXXXXX.XXXXXXX

1 INTRODUCTION
Consider a rectangular area $R$ as a bounded region in the two-dimensional Euclidean plane. We embed a rectangular grid graph $G$ in that rectangular area $R$. Let a robot with sensing capability stay on the nodes of $G$. Let depending on the sensing radius of the robot, the grid is embedded in such a way that, being placed on a node a robot can sense its immediate upward, immediate downward, immediate left, and immediate right neighbour node along with its position completely. Let a robot can move to its immediate upward, immediate downward, immediate left, and immediate right neighbour nodes through the edges of $G$. Now we want to place a set of robots on some nodes of $G$ such that each node of $G$ is sensed by at least one robot. Now cost and resilience are the major parameters to consider. We can accomplish the target in different ways. One way can be by putting robots at each node. In this way, to disconnect a node we have to disable five robots. Here the resilience is highest but the cost is maximum (See Fig. 1(a)). If we put robots on a minimum dominating set of $G$ then disabling one robot will disconnect five nodes. Here the cost is minimum but resilience is the lowest (See Fig. 1(b)). If we put robots on a maximal independent set of $G$ then disabling two robots can disconnect at most four nodes. The number of robots required in this case is one-third of the number of nodes and this method gives decent resilience (See Fig. 1(c)). If we put robots on a maximum independent set of $G$ then disabling four robots can disconnect at most five nodes. The number of robots required in this case is half of the number of nodes and this method gives good resilience (See Fig. 1(d)). So in this work, we consider robots placing on a maximum independent set of $G$.

In this paper, we give an algorithm of Maximum Independent Set (MIS) formation on a finite grid. Let a swarm of autonomous robots is placed initially on the distinct nodes of $G$. The MIS formation problem asks the robots to rearrange and take positions such that the robot occupied nodes form a MIS of $G$. The robots work...
autonomously, which means they work without any central control. The robots are homogeneous (i.e., they all run the same algorithm), identical (indistinguishable), and anonymous (without any identifier). Such robot swarms can have the capability to do certain tasks like gathering, dispersion, exploration, pattern formation, filling, etc. In some cases, robots have memory and can communicate with other robots. Based on these powers there are four types of robot models which are \textit{OBLOT, FSTA, FCOM, LUMI}. In \textit{OBLOT} model robots are silent (no communication) and oblivious (no persistent memory). In \textit{FSTA} model robots are silent and non-oblivious. In \textit{FCOM} model robots can communicate but are oblivious. In \textit{LUMI} model robots can communicate and are non-oblivious. Robots can have a finite bit of memory which is generally interpreted as a finite number of lights that can take finitely many different colors. Seeing own light is equivalent to having memory and seeing the lights of other robots is equivalent to communication. After activation, each robot follows a look-compute-move (LCM) cycle. In the look phase, the robot takes a snapshot of its surroundings in its vicinity and gets the position and states of other robots. In compute phase it runs the algorithm and gets an output. In the move phase, the robot moves to its destination node or stays at the same node depending on the output. Activation plays a big role and it is determined by the scheduler. There are generally three types of schedulers. These are (1) fully synchronous scheduler where the time is divided into global rounds and each robot work activates in each round and simultaneously executes their LCM cycle; (2) Semi synchronous scheduler where also the time is divided into global rounds but some robots activate in a round and simultaneously execute their LCM cycle; (3) Asynchronous scheduler where there is no common notion of time among robots and all robots execute their LCM cycle independently.

Vision is an important factor in performing these tasks. In [1–4] infinite visibility has been used. But infinite visibility is not practically possible due to hardware limitations. Limited visibility is more practical. Under limited visibility, a robot can see up to a certain distance in a plane and up to a certain hop in discrete space. In our work, we consider \textit{LUMI} model robots with 2 hop visibility range and 3 colors under a semi-synchronous scheduler. The robots agree on the two directions and their orientations, one which is parallel to rows of \( G \) and another which is parallel to the columns of \( G \). Hence each robot can determine its four directions. In this work, we propose an \textit{MIS} formation algorithm for a robot swarm, which is initially placed arbitrarily on the nodes of the grid. We show that the proposed algorithm forms \textit{MIS} under a semi-synchronous scheduler using robots having only two hop visibility and a light that can take three distinct colors. The next section describes all relevant works and discusses the scope of our work.

### 1.1 Related Works and Our Contributions

Using swarm robotics various types of problems have been studied like exploration [4, 11], gathering [3, 6], dispersion [8–10], pattern formation [1, 2, 5] under different model. In [1–4] robots are considered to have infinite visibility. But infinite visibility is not practically possible due to hardware limitations. Limited visibility is more practical. Robots with limited visibility are called \textit{myopic} robots. Myopic robots have been used in [6, 7, 11, 14]. A lot of problems [1, 3, 4, 7, 9, 12, 13] have been explored under grid graph. \textit{MIS} formation on a finite grid can be seen from two perspectives. One perspective is the deployment of robots through a door node. Another perspective is pattern formation. As of our knowledge, there is no algorithm for arbitrary pattern formation in a finite grid graph. To the best of our knowledge, there are only two reported work [7, 14] which considers \textit{MIS} formation problem on a graph using an autonomous robot swarm. [7] have given an \textit{MIS} filling algorithm using robots having light with three colors, 2 hop visibility for fully oriented finite grid under asynchronous scheduler. In another algorithm, they have solved the same problem using robots with seven light colors, and 3 hop visibility under an asynchronous scheduler but in an unoriented grid. [14] have given an \textit{MIS} filling algorithm for arbitrary graph with one door node using (\( \Delta + 6 \)) light color, 3 hop visibility, \( O(\log(\Delta)) \) bits of persistent storage under asynchronous scheduler. In another algorithm, they have solved the same problem with \( k (> 1) \) door nodes using \((\Delta + k + 6) \) light color, 5 hop visibility, \( O(\log(\Delta + k)) \) bits of persistent storage under semi synchronous scheduler. Another set of works is [12, 13] which are remotely related to \textit{MIS} formation problem. [12] solves the uniform scattering problem under an asynchronous scheduler and [13] solves the uniform scattering problem under a fully synchronous scheduler on a finite rectangular grid considering myopic robots. However, \textit{MIS} formation cannot be achieved by any special case or slight modification of these works.

| Work | Visibility range (hop) | Scheduler | Door number | Graph Topology | Internal Memory | Color number |
|------|------------------------|-----------|-------------|----------------|-----------------|--------------|
| 1st algorithm in [7] | 2 | ASYNC | 1 | oriented rectangular grid | None | 3 |
| 2nd algorithm in [7] | 3 | ASYNC | 1 | unoriented rectangular grid | None | 7 |
| 1st algorithm in [14] | 3 | ASYNC | 1 | Arbitrary connected Graph | \( O(\log(\Delta)) \) | \( \Delta + 6 \) |
| 2nd algorithm in [14] | 5 | SSYNC | \( k > 1 \) | Arbitrary connected Graph | \( O(\log((\Delta + 6) \log(\Delta + k))) \) | \( \Delta + k + 6 \) |
| Our Algorithm | 2 | SSYNC | None (Arbitrary initial deployment) | oriented rectangular grid | None | 3 |

In this paper from the motivation of finding a robust but cost-effective coverage of a rectangular region, we give an algorithm to form an \textit{MIS} pattern on a rectangular finite grid by luminous robots under a semi-synchronous scheduler. In contrast to [7, 14], our proposed algorithm does not use the door concept and allows to form of an \textit{MIS} starting from any initial configuration. Thus, this work generalizes the initial condition of the work in [7, 14] for rectangular grid topology. Also, there can be practical scenarios where the door concept is not possible to implement. Suppose the robots are arbitrarily placed on the grid initially. If one wants to
We assume that the size of the rectangular grid is unknown to the robots. We consider the robots equipped with motion actuators and visibility sensors. These robots move on a simple undirected connected graph \( G = (V, E) \), where \( V \) is a finite set of \( p = m \times n \) nodes and \( E \) is a finite set of \( q = (m - 1) \times n + m \times (n - 1) \) edges. \( m \) and \( n \) are positive integers greater than 1. Robots can stay on the nodes only. Robots can sense their surrounding nodes and can move through edges. We assume that \( G \) is an \( m \times n \) rectangular grid embedded on a plane, where \( m \) is the number of rows and \( n \) is the number of columns. We call the topmost row as the 1st row, the second row from the top as the 2nd row, and so on. Similarly, we call the leftmost column as the 1st column, the second column from the left as the 2nd column, and so on. We can think the grid as an \( m \times n \) matrix, where the \((i, j)\)th entry of the matrix represents the node on the \( i\)th row and \( j\)th column of the grid. \( G \) satisfies the following condition: there exists an order on the nodes of \( V = \{v_1, v_2, v_3, \ldots, v_p\} \), such that:

\[ \forall x \in \{1, 2, \ldots, p\}, (x \neq 0 \mod n) \implies \{v_x, v_{x+1}\} \in E \]

\[ \forall y \in \{1, 2, \ldots, (m - 1) \mod n\}, \{v_{y}, v_{y+n}\} \in E. \]

We assume that the size of the rectangular grid is unknown to the robots. We consider the leftmost column, rightmost column, uppermost row, and lowermost row as the west boundary, east boundary, north boundary, and south boundary respectively.

Each robot on activation executes a look-compute-move (L-C-M) cycle. In the look phase, a robot takes a snapshot of its surrounding in its vicinity. In compute phase it runs an inbuilt algorithm taking the snapshot and its previous state (if the robot is not oblivious) as an input. Then it gets a color and a position as an output. In the move phase, the robot changes its color if needed and moves to its destination node or stays at the same node.

**Synchronizer:** There are generally three types of schedulers, which are fully synchronous, semi-synchronous, and asynchronous. In a synchronous scheduler, the time is equally divided into different rounds. The robots activated in a round execute the L-C-M cycle and each phase of the L-C-M cycle is executed simultaneously by all the robots. That means, all the active robots in a round take their snapshot at the same moment, and, Compute phase and Move phase are considered to happen instantaneously. Under a fully synchronous scheduler, each robot gets activated and executes the L-C-M cycle in every round. Under semi synchronous scheduler a nonempty set of robots gets activated in a round. An adversary decides which robot gets activated in a round. In a fair adversarial scheduler, each robot gets activated infinitely often. Under an asynchronous scheduler, there is no common notion of time for the robots. Each robot independently gets activated and executes its L-C-M cycle. In this scheduler Compute phase and Move phase of robots may be different. Even the time length of two L-C-M cycles of one robot may be different. The gap between two consecutive L-C-M cycles or the time length of an L-C-M cycle of a robot is finite but can be unpredictably long. We consider the activation time and the time taken to complete an L-C-M cycle is determined by an adversary. In a fair adversarial scheduler, a robot gets activated infinitely often.

Our work is under semi synchronous scheduler.

**Visibility of robots:** A robot can see all of its neighbour nodes within 2 hop distance. Thus a robot can see 13 nodes including its position. We denote the hop distance of visibility as \( \phi \).

The first left neighbour node, second left neighbour node, first upward neighbour node, second upward neighbour node, first right neighbour node, second right neighbour node, first downward neighbour node, second downward neighbour node, north-east neighbour node, north-west neighbour node, south-east neighbour node, south-west neighbour node of a robot are denoted by \( I, l_1, u_1, u_2, r_1, r_2, d_1, d_2, ne, nw, se, sw \) respectively (See Figure 2).

**Lights:** Each robot has a light that can take three different colors. These colors are red, blue, and green. The initial color of each robot is green. The blue color indicates that the robot wants to move but its desired path is stuck by other robots. The red color indicates that the robot has reached its final position and will not move further. Here onward we shall call a robot with color red (or blue or green) as red (or blue or green) robot.

**Axes Agreement:** All robots agree on the directions of the axis parallel to rows and the axis parallel to columns. Hence robots agree on the global notion of north, south, east, west, up, down, right, and left directions. Each robot can determine the four directions from a node.

**Definition 2.1 (Maximum Independent Set).** An independent set \( I \) of a graph \( G \) is a set of nodes of \( G \) such that no two nodes of that set are adjacent. A maximum independent set \( (MIS) \) of \( G \) is an independent set of the largest possible size.

Consider an \( m \times n \) rectangular grid \( G \) where \( m \geq 2 \) is the number of rows and \( n \geq 2 \) is the number of columns present in the grid. We convert it to a door concept scenario then all robots need to gather at a corner, which might not be possible if the robots are not point robots. One might argue that all initial positions of robots can be considered as different doors and compare it with the multi-door algorithm of [14] which works under a semi-synchronous scheduler. But compared with that, our algorithm uses only a constant memory. But compared with that, our algorithm uses only a constant memory.

Figure 2: View of a robot with two hop visibility
we show that \( S = \) robots are present arbitrarily on different nodes of the rectangular grid such that there can be at most one robot on a node of the rectangular grid. Next, we state the problem formally.

**Definition 2.3 (MIS formation problem).** Suppose a set of finite robots are placed arbitrarily at distinct nodes of a finite rectangular grid \( G \). The MIS formation problem requires the robots to occupy distinct nodes of \( G \) and settle down avoiding collision such that the set of occupied nodes of \( G \) is a maximum independent set of \( G \).

The next section provides a proposed algorithm that solves MIS formation problem.

### 3 MIS FORMATION ALGORITHM

This section provides an algorithm namely, MIS Formation Algorithm that claims to solve the MIS formation problem. Different views of a robot are depicted in different figures in this section. In the figures of this section onward green, blue and red color filled circles respectively represent green robot, blue robot and red robot. The black circle indicates a node that may or may not exist. If that node exists then it can be vacant or occupied by a robot. This means a robot can ignore black circle nodes in compute phase. The black cross indicates that the node does not exist. The black diamond indicates that the node exists. Initially, all robots are colored green.

**Definition 3.1 (North-West Quadrant).** Let a robot \( r_1 \) is at \((i,j)^{th}\) node of a grid. Then the nodes having coordinates \( \{(x,y): x \leq i, y \leq j\} - \{(i,j)\} \) are called north-west quadrant of \( r_1 \).

A green robot moves at left by maintaining at least 2 hop distance from its left robot until it reaches the west boundary. After reaching the west boundary it moves upward by keeping at least 2 hop distance from its upward robot. In this way, a robot will be fixed at the northwest corner node and will be fixed first. Green robots move left by maintaining the necessary distance from their left robot until it reaches the east boundary or near a red robot (See Fig. 6).

Then it moves upward by maintaining the necessary distance from its upward robot until it reaches the north boundary or near a red robot (See Fig. 8). Thus the robot reaches a suitable node from which it can see the necessary view to becoming red (See Fig. 9).

**Definition 3.2 (Fixed robot).** When a robot becomes red, it does not move any more according to the MIS Formation Algorithm 1. This robot is called a fixed robot.

If a green robot \( r_g \) sees that its \( l_l \) (if \( r_g \) is at north boundary) or \( l_r \) (if \( r_g \) is at west boundary) or both (if \( r_g \) is neither at north boundary nor at west boundary) neighbour nodes are occupied by red robots and \( r_a \) can not move upward or left then there are two possibilities.

**Case-1:** If \( r_l \) (if \( r_g \) is not at east boundary) or \( d_l \) (if \( r_g \) is at east boundary) neighbour node of \( r_g \) is vacant then it will move 1 hop right (if \( r_g \) is not at east boundary) or 1 hop down (if \( r_g \) is at east boundary) (See Fig. 10) or (See Fig. 11).

**Case-2:** If \( r_l \) (if \( r_g \) is not at east boundary) or \( d_l \) (if \( r_g \) is at east boundary) neighbour node of \( r_g \) is occupied by a robot then \( r_g \) will turn into blue indicating that it has been stuck and wants to
move right (if \( r_a \) is not at east boundary) or down (if \( r_a \) is at east boundary) but can not move (See Fig. 13).

Now if a green robot \( r_b \) sees its \( l_1 \) neighbour node is occupied by a blue robot \( r_a \) then there are following cases.

Case-1: If \( l_1 \) (if \( r_b \) is at east boundary) neighbour node of \( r_b \) is vacant or \( d_1 \) (if \( r_b \) is at east boundary) neighbour node of \( r_b \) is vacant then it will move 1 hop right (if \( r_b \) is not at east boundary) or 1 hop down (if \( r_b \) is at east boundary) (See Fig. 14) or (See Fig. 7).

Case-2: If \( l_1 \) (if \( r_b \) is at east boundary) neighbour node of \( r_b \) is not vacant or \( d_1 \) (if \( r_b \) is at east boundary) neighbour node of \( r_b \) is not vacant.

Case-2.1: If \( u_1 \) and \( u_2 \) neighbour node of \( r_b \) is vacant then \( r_b \) will move upward by keeping necessary distance from its upward robot until it reaches north boundary or near a red robot (See Fig. 16).

Case-2.2: If \( u_1 \) or \( u_2 \) neighbour node of \( r_b \) is occupied by a non red robot then \( r_b \) will do nothing.

Case-2.3: If one of \( u_1 \) and \( u_2 \) neighbour node of \( r_b \) is occupied by a red robot and another is vacant then it will turn blue (See Fig. 17).

If a green robot \( r_d \) which is on the east boundary sees its \( u_1 \) neighbour node is occupied by a blue robot \( r_c \) then there are following cases.

Case-1: If \( l_1 \) and \( l_2 \) neighbour node of \( r_d \) both are vacant then \( r_d \) will move 1 hop left.

Case-2: Anyone between \( l_1 \) and \( l_2 \) neighbour node of \( r_d \) or both are non vacant.

Case-2.1: If \( d_1 \) neighbour node of \( r_d \) is vacant then \( r_d \) will move 1 hop down (See Fig. 15).

Case-2.2: If \( d_1 \) neighbour node of \( r_d \) occupied by a green robot then \( r_d \) will turn into blue (See Fig. 18).

**Definition 3.3 (Blue Sequence).** If a node or some consecutive nodes of a row or east boundary or both in a rectangular grid are occupied by blue robots then the \(-\) or \(-\) or \| like sequence of consecutive blue robots is called a blue sequence.

![Figure 4: Blue Sequence and Adjacent green Robot](image)

In Fig. 4 the robots \( r_1, r_2, r_3, r_4, r_5, r_6, r_7, r_8 \) forms the blue sequence.

A blue sequence can proceed till the \((m-1, n)\)th node at most. If the \((m, n)\)th node is occupied by some robot, that robot will never be blue since its \(d_1\) neighbour node does not exists and the existence of \(d_1\) neighbour node is necessary to become blue for the robots present in the east boundary.

**Definition 3.4 (Adjacent Green Robot).** Consider the blue sequence will not proceed further. If the blue sequence will not proceed further. If the blue sequence before the east boundary then the green robot at the \( r_1 \) neighbour node of the rightmost blue robot of the sequence and if the blue sequence continues through east boundary then the green robot at the \( d_1 \) neighbour node of the downmost blue robot of the sequence present in the east boundary is called the adjacent green robot of the blue sequence.

In Fig. 4 \( r_8 \) is the adjacent green robot of the blue sequence.

**Definition 3.5 (Predecessor Blue Robots).** Consider a blue robot \( r_k \) of a blue sequence. All the robots which became blue in that blue sequence before the round in which \( r_k \) became blue, are called the predecessor blue robots of \( r_k \) in that blue sequence.

In Fig. 4 \( r_1, r_2, r_3, r_4 \) and \( r_5 \) are the predecessor blue robots of \( r_6 \).

If a blue robot \( r_e \) which is not at the east boundary sees its \( r_l \) neighbour node is vacant then it turns green and moves 1 hop right (See Fig. 12).

If a blue robot \( r_e \) which is at the east boundary sees its \( l_1, l_2 \) and \( d_1 \) neighbour nodes then there are following cases.

Case-1: Both \( l_1 \) and \( l_2 \) neighbour nodes of \( r_e \) are vacant then \( r_e \) turns green and move 1 hop left (See Fig. 19).

Case-2: Any one between \( l_1 \) and \( l_2 \) neighbour nodes of \( r_e \) is not vacant and its \( d_1 \) neighbour node is vacant then it turns green and move 1 hop down (See Fig. 20).

![Figure 5: Tail and After 1 hop shifting](image)

**Definition 3.6 (Tail).** If the blue sequence continues through east boundary and a blue robot of the sequence from the east boundary leaves the sequence by moving left then the remaining blue robots of the sequence below the leaving blue robot will be called tail.

In Fig. 5 \( r_7, r_8 \) is the tail after \( r_6 \) leaves the blue sequence.

**Definition 3.7 (1 Hop Shifting).** If the adjacent green robot or any robot of the blue sequence moves from its position then each of its predecessor blue robots moves 1 hop to fill the vacant node.
and to make the starting node of the blue sequence vacant. This is called 1 hop shifting of the blue sequence.

In Fig. 5 1 hop shifting of the blue sequence of Fig. 4 has been done after the robot $r_6$ moves 1 hop left and makes its position vacant.

If a blue robot $r_e$ which is at the east boundary sees its $u_1$ neighbour node is vacant and $l_1$ neighbour node is not occupied by a blue robot then it turns green (See Fig. 21).

If a blue robot $r_e$ which is at the east boundary sees its $u_1$ neighbour node is occupied by a red robot and $l_1$ neighbour node is vacant then it turns green (See Fig. 21).

If a blue robot $r_e$ which is at the east boundary sees its $u_1$ neighbour node is occupied by a green robot then it turns green (See Fig. 21).

Now we define some sets of views.

$G_1 = \{GL_1, GL_2, GL_3, GL_4\}$  
$G_2 = \{GD_1, GD_2, GD_3, GD_4\}$

$G_3 = \{GR_1, GR_2, GR_3, GR_4, GR_5, GR_6, GR_7, GR_8\}$

$G_4 = \{GU_1, GU_2, GU_3, GU_4, GU_5, GU_6, GU_7, GU_8, GU_9, GU_{10}, GU_{11}, GU_{12}, GU_{13}, GU_{14}, GU_{15}, GU_{16}, GU_{17}, GU_{18}, GU_{19}, GU_{20}\}$

$G_5 = \{GB_1, GB_2, GB_3, GB_4, GB_5, GB_6, GB_7, GB_8, GB_9, GB_{10}, GB_{11}, GB_{12}, GB_{13}, GB_{14}\}$

$G_6 = \{G-R_1, G-R_2, G-R_3, G-R_4, G-R_5, G-R_6, G-R_7\}$
There is no robot which will move to $l_1$ neighbour node of $r$ by $(R)$ movement. So there is no $(LR)$ collision.

$(LU)$: A robot will move upward if its view belongs to $G_4$. From Fig. 16 and Fig. 8 it is clear that for $(LU)$ movement of a robot $r$, $n_e$ neighbour node of $r$ is vacant or does not exist. There is no robot which will move to $u_1$ neighbour node of $r$ by $(L)$ movement. So there is no $(LU)$ collision.

$(LD)$: A robot will move downward if its view belongs to $G_2$ or $B_3$. From Fig. 20, Fig. 7, Fig. 15 and Fig. 11 it is clear that for $(LD)$ movement of a robot $r$, $s_e$ neighbour node of $r$ does not exist. There is no robot which will move to $d_1$ neighbour node of $r$ by $(L)$ movement. So there is no $(LD)$ collision.

$(RU)$: A robot will move upward if its view belongs to $G_4$. From Fig. 8 and Fig. 16 it is clear that for $(RU)$ movement of a robot $r$, $n_w$ neighbour node of $r$ is vacant or occupied by a red robot or does not exist. There is no robot which will move to $u_1$ neighbour node of $r$ by $(R)$ movement. So there is no $(RU)$ collision.

$(RD)$: A robot will move downward if its view belongs to $G_2$ or $B_3$. From Fig. 20, Fig. 7, Fig. 15 and Fig. 11 it is clear that for $(RD)$ movement of a robot $r$, $s_w$ neighbour node of $r$ is on east boundary then $n_e$ neighbour node of $r$ is vacant or occupied by a red robot or does not exist. There is no robot which will move to $r_1$ neighbour node of $r$ by $(D)$ movement. So there is no $(RD)$ collision.

$(UD)$: A robot will move upward if its view belongs to $G_4$. From Fig. 16 and Fig. 8 it is clear that for $(UD)$ movement of a robot $r$, $u_1$ neighbour node of $r$ is vacant and $u_2$ neighbour node of $r$ is vacant.
Lemma 3.9. If all robots have turned its color to red then the set of robot occupied grid nodes forms an MIS of $G$.

Proof. First we show that the set of robot occupied grid nodes is an independent set of $G$. We show this by showing that no two red robots are adjacent. Opposite to our claim, let there be two adjacent red robots $r_1$ and $r_2$. If $r_1$ and $r_2$ are on the same column then let $r_2$ be the robot below $r_1$ and if $r_1$ and $r_2$ are on the same row then let $r_2$ be the robot right to $r_1$. Let $r_1$ and $r_2$ change its color to red in $k_1^{th}$ and $k_2^{th}$ round respectively. Now there can be two possibilities.

Case-I: $(k_1 \leq k_2)$ Since red robots never move, so throughout $k_2^{th}$ round the $r_1$ robot is at the $u_l$ or $l_1$ neighbour node of $r_2$. According to our proposed algorithm, $r_2$ will change its color to red if it sees any view belongs to the set $G_{l0}$. But no view in $G_{l0}$ allows the $u_l$ or $l_1$ neighbour node of $r_2$ to be occupied by a robot. So this leads to a contradiction.

Case-II: $(k_1 > k_2)$ In this case $r_2$ becomes red and gets fixed before $r_1$. Hence the $f_2$, $u_2$ and $w_2$ neighbours of $r_2$ must be occupied by red robots if these neighbour nodes exist and it sustains in $k_1$ round also (since red robots never move). If $r_1$ robot is at $u_l$ or $l_1$ neighbour node of $r_2$, then $u_l$ or $l_1$ neighbour node of $r_2$ exists. Hence view of $r_2$ at $k_2^{th}$ round must be one of G-R2 by G-R4 for the case when $r_1$ is at $l_1$ neighbour node of $r_2$, G-R3, G-R5, G-R6 and G-R7. In all such views either $l_1$ or $u_1$ neighbour node of $r_1$ is occupied by a red robot. Hence $r_1$ would not change its color to red in $k_1^{th}$ round, which is a contradiction.

Hence if all robots turn red then the robot occupied nodes form an independent set. Now the number of robots is $\lceil \frac{mn}{2} \rceil$ which is the maximum possible size of an independent set of $G$. Since Theorem 3.8 gives that there is no collision of robots, so all the red robots must be at distinct grid nodes. So the number of robot occupied nodes after all robots turned red is also $\lceil \frac{mn}{2} \rceil$. Thus, the independent set formed by robot occupied grid nodes is an MIS.

Lemma 3.10. If a row consists three types of robots i.e. red, blue and green then the red robots will be at left, blue robots will be at middle and green robots will be at right of the row.

Proof. A green robot becomes red, when it sees its $l_2$, $w_2$, $u_2$ neighbour nodes (if exist) are occupied by red robots and $l_1$, $u_1$ neighbour nodes (if exist) are vacant. So there cannot be any green or blue robot at left of a red robot. Thus red robots are at left of a row.

When a blue sequence starts then the $l_1$ neighbour node (if exists) of the blue robot which became blue first, is occupied by a red robot. A blue sequence is a sequence of blue robots which are at consecutive nodes. So there is no green robot at the middle of a blue sequence.

Thus the red robots will be at left, blue robots will be at middle and green robots will be at right of the row.

Lemma 3.11. If there are $\lceil \frac{n}{2} \rceil - 2$ robots present in a row consists of $(n - 1)$ nodes, then after finite round $(n - 1)^{th}$ and $(n - 2)^{th}$ node will be vacant.

Proof. In a row the distance of a red robot from its immediate left or immediate right red robot is exactly 2 hop. In a row distance of a red robot from its immediate left or immediate right green robot will be at most 2 hop since all green robots move left by keeping 2 hop distance. In a row distance of a blue robot which became blue last, from its immediate right green robot is exactly 1 hop. In a row distance of a blue robot from its immediate left or immediate right blue robot is exactly 1 hop. Thus by Lemma 3.10 in a row distance of a row from its immediate left or immediate right robot is at most 2 hop. Maximum possible number of robots is $\lceil \frac{n}{2} \rceil - 2$ (if $n$ is even) or $\lceil \frac{n}{2} \rceil - 1 - 2$ (if $n$ is odd). If possible we try to put the robots in such a way so that $(n - 1)^{th}$ and $(n - 2)^{th}$ node does not remain empty. If we put robots on even positioned nodes then $i^{th}$ robot will be at $2i^{th}$ node. $(\frac{n}{2} - 2)^{th}$ robot will be at $(n - 4)^{th}$ node. $(\frac{n}{2} - 2)^{th}$ robot will be at $(n - 3)^{th}$ node. Thus in both cases $(n - 1)^{th}$ and $(n - 2)^{th}$ node will be vacant.

Lemma 3.12. Let $r_1$ be the left most non red robot in the topmost non red robot occupied row. Let $r_1$ be a part of a blue sequence and $r_1$ be the first robot which turned blue for any one view from the Fig. 13. If the blue sequence ends at $(m - 1, n)$ node then 1 hop shifting will be done.

Proof. If the blue sequence starts at $k^{th}$ row, continues through east boundary and ends at $(m - 1, n)$ node then $1^{st}$, $2^{nd}$, ..., $i^{th}$, ..., $(k - 1)^{th}$ row each contains $u_i$ robots. $k^{th}$ row contains more than $u_i$ robots. $(k + 1)^{th}$, $(k + 2)^{th}$, ..., $m^{th}$ row together will contain less than $u_{k + 1} + u_{k + 2} + \ldots + u_m$ robots. There will be at least one row (say $p^{th}$ row) which will contain less than $u_p$ robots. If more than one such row exists then consider the top most row (say $i^{th}$ row) which contains less than $u_i$ robots. If any robot comes from below row and makes $u_i$ number robot in $i^{th}$ row, then we shall consider below $i^{th}$ row which contains less than $u_i$ robot. If this continues since the number of rows is constant we must get such a row (say $r^{th}$ row) where number of robots will be less than $u_r$ and no robots will enter from below. Without loss of generality, we consider such row as $i^{th}$ row. If $n$ is odd then $u_i$ is $\lceil \frac{n}{2} \rceil$ when $l$ is odd and $\lceil \frac{n}{2} \rceil - 1$ when $l$ is even. If $n$ is even then $u_i$ is $\lceil \frac{n}{2} \rceil$. If we consider $i^{th}$ row except the east boundary node which is occupied by a blue robot or green robot then there are $(n - 1)$ nodes and at most $\lceil \frac{n}{2} \rceil - 2$ robots. After finite round when all the robots of $i^{th}$ row except the right most blue or green robot will be at most 2 hop distance from each other, $(l, n - 2)$ and $(l, n - 1)$ node will be vacant by Lemma 3.11. Then the right most robot of $i^{th}$ row will move from $(l, n)$ node to $(l, n - 1)$ node and 1 hop shifting will be done automatically.

Theorem 3.13. MIS Formation Algorithm forms maximum independent set after finite rounds without any collisions.

Proof. Consider the uppermost row which contains at least one green or blue robot. If there is no such row then every robot
present in the grid is red. Therefore by Lemma 3.9 the proof is done.

Let there exists a row (say $k^{th}$ row) which contains at least one green or blue robot. Let $r_1$ be the leftmost non red robot on that row. $r_1$ can be green or blue. Note that all the robots present in the north-west quadrant of $r_1$ are red and they are fixed.

Case-1: $r_1$ is green.

$r_1$ continues moving left as long as it sees any views from $[GL_1, GL_2, GL_3, GL_4]$ (Fig. 6). While $r_1$ is progressing left through the row if any green robot from below row move upwards and comes to the left of $r_1$ then we will consider the new robot as $r_1$. If $r_1$ does not see any view from $[GL_1, GL_2, GL_3, GL_4]$ (Fig. 6) then it must see any view from $[GB_1, GB_2, GB_3, GB_4, GB_5, GD_1, GD_2, GR_1, GR_2, GR_3, GR_4, GR_5, GU_1, GU_2, GU_3, GU_4, GU_5, GU_6, GU_7, GU_8, GU_9, GU_10, GU_11, GU_12, G-R_1, G-R_2, G-R_3, G-R_4, G-R_5, G-R_6, G-R_7]$ (Fig. 13, Fig. 11, Fig. 10, Fig. 8, Fig. 9). If $r_1$ sees any one view from $[GB_1, GB_2, GB_3, GB_4, GB_5]$ (Fig. 13) then it turns blue and goes to Case-2. If $r_1$ sees any one view from $[GD_1, GD_2]$ (Fig. 11) then it will go to its $d_1$ neighbour node. Now we may get a new $r_1$ since there may exist some non red robot at the left in the current row. Now $r_1$ will not move to its $d_1$ neighbour node and will remain $r_1$ since it will not get any view from $[GD_1, GD_2]$ (Fig. 11). If $r_1$ sees any one view from $[GR_1, GR_2, GR_3, GR_4, GR_5]$ (Fig. 10) then it will go to its $r_1$ neighbour node. Now it will not see any view from $[GR_1, GR_2, GR_3, GR_4, GR_5]$ (Fig. 10) and $[GD_1, GD_2]$ (Fig. 11). If $r_1$ sees any one view from $[G-R_1, G-R_2, G-R_3, G-R_4, G-R_5, G-R_6, G-R_7]$ (Fig. 9) then it turns red. Else $r_1$ will see any one view from $[GU_1, GU_2, GU_3, GU_4, GU_5, GU_6, GU_7, GU_8, GU_9, GU_10, GU_11, GU_12]$ (Fig. 8) and continues moving upward until it sees any one view from $[G-R_1, G-R_2, G-R_3, G-R_4, G-R_5, G-R_6, G-R_7]$ (Fig. 9) and will turn red.

Case-2: $r_1$ is blue.

A blue robot became blue as a part of a blue sequence. Now it is either a part of a blue sequence or a part of a tail. As $r_1$ is the left most non red robot in the topmost non red robot occupied row, there can be two cases.

Case-2.1: If $r_1$ is a part of a tail then $r_1$ will be at east boundary and the topmost blue robot of the tail. If we consider $l_1$ and $u_1$ neighbour nodes of $r_1$ then there can be four type of figures .

In these three types i.e. $[BG_1, BG_3, BG_5]$ (Fig. 21), atleast one among $l_1$ and $u_1$ neighbour nodes of $r_1$ is not occupied by a red robot and $r_1$ will turn into green . The robot $r_1$ goes to Case-1 and this $r_1$ will never become blue as it was the upmost robot of a tail and all the robots which are at north-west quadrant of $r_1$ are red and atleast one among $l_1$ and $u_1$ neighbour nodes of $r_1$ is not occupied by a red robot.

If both $l_1$ and $u_1$ neighbour nodes of $r_1$ are occupied by red robots then $r_1$ goes to Case 2.2 (similar to $[GB_5]$) (Fig. 13).

Case-2.2: If $r_1$ is a part of a blue sequence then $r_1$ is the first robot which turned blue for any one view from the Fig. 13. The blue sequence can continue along the row and east boundary.

Case-2.2.1: If the blue sequence ends before $(m-1,n)$ node 1 hop shifting will be done after the adjacent green robot moves from its node. 1 hop shifting would be done before it if any blue robot of the blue sequence from the east boundary moves left.

Case-2.2.2: If the blue sequence ends at $(m-1,n)$ node 1 hop shifting will be done by Lemma 3.12.

Now $r_1$ will turn into green and goes to case-1. This $r_1$ will never become blue as its $l_1$ or $u_1$ neighbour node is vacant and all the robots which are at north-west quadrant of $r_1$ are red.

Selecting a non red robot we are making a non red robot into a red robot. Since the total number of robots is finite, after finite round all the robots will be red. Therefore by Lemma 3.9 the proof follows.

\square

4 CONCLUSION

This work presents an algorithm that forms a Maximum Independent Set (MIS) on a finite rectangular grid $G$ by myopic robots. If the size of a maximum independent set of $G$ is $k$ then initially $k$ robots are placed arbitrarily on distinct nodes of $G$. The robots are considered to be luminous and have a light that can take three distinct colors. We assume the robots agree on the global notion of north, south, east, and west direction. The robots have two hop visibility. The robots are controlled under an adversarial semi-deterministic scheduler. In construct to previous MIS formation algorithms the algorithm proposed in this work does not use the door concept. It allows the robots to form MIS from any arbitrary starting configuration. This generalizes the initial condition of the previous works for rectangular grid topology.

In this work, we assumed two visibility of robots, so as a future direction one can try proposing an MIS formation algorithm which only uses one hop visibility of robots. Further, it will be interesting to provide an algorithm for the same problem under an asynchronous scheduler.

REFERENCES

[1] Kaustav Bose, Ranendu Adhikary, Manash Kumar Kundu, and Buddhadeb Sau. 2020. Arbitrary pattern formation on infinite grid by asynchronous oblivious robots. Theoretical Computer Science 815 (2020), 213–227.

[2] Kaustav Bose, Manash Kumar Kundu, Ranendu Adhikary, and Buddhadeb Sau. 2021. Arbitrary pattern formation by asynchronous opaque robots with lights. Theoretical Computer Science 849 (2021), 138–150.

[3] Gianlorenzo d’Angelo, Gabriele Di Stefano, Ralf Klasing, and Alfredo Navarra. 2016. Gathering of robots on anonymous grids and trees without multiplicity detection. Theoretical Computer Science 610 (2016), 158–168.

[4] Stéphane Devismes, Anissa Lamani, Franck Petit, Pascal Raymond, and Sébastien Tixeuil. 2021. Terminating exploration of a grid by an optimal number of asynchronous oblivious robots. Comput. J. 64, 1 (2021), 132–154.

[5] Paola Flocchini, Giuseppe Prencipe, Nicola Santoro, and Peter Widmayer. 2008. Arbitrary pattern formation by asynchronous, anonymous, oblivious robots. Theoretical Computer Science 407, 1-3 (2008), 412–447.

[6] Sayaka Kamei, Anissa Lamani, Fukuhito Ooshita, Sébastien Tixeuil, and Koichi Wada. 2019. Gathering on rings for myopic asynchronous robots with lights. arXiv preprint arXiv:1911.04757 (2019).

[7] Sayaka Kamei and Sébastien Tixeuil. 2020. An Asynchronous Maximum Independent Set Algorithm by Myopic Luminous Robots on Grids. arXiv preprint arXiv:2012.03399 (2020).

[8] Ajay D Kshemkalyani, Anisur Rahaman Molla, and Gokarna Sharma. 2019. Fast dispersion of mobile robots on arbitrary graphs. In International Symposium on Algorithms and Experiments for Sensor Systems, Wireless Networks and Distributed Robotics. Springer, 23–40.

[9] Ajay D Kshemkalyani, Anisur Rahaman Molla, and Gokarna Sharma. 2020. Efficient dispersion of mobile robots on dynamic graphs. In 2020 IEEE 40th International Conference on Distributed Computing Systems (ICDCS). IEEE, 732–742.

[10] Shota Nagahama, Fukuhito Ooshita, and Michiko Inoue. 2019. Ring exploration of myopic luminous robots with visibility more than one. In International Symposium on Stabilizing, Safety, and Security of Distributed Systems. Springer, 256–271.
[12] Pavan Poudel and Gokarna Sharma. 2019. Time-optimal uniform scattering in a grid. In *Proceedings of the 20th International Conference on Distributed Computing and Networking*. 228–237.

[13] Pavan Poudel and Gokarna Sharma. 2020. Fast uniform scattering on a grid for asynchronous oblivious robots. In *International Symposium on Stabilizing, Safety, and Security of Distributed Systems*. Springer, 211–228.

[14] Sai Vamshi Samala, Subhajit Pramanick, Debasis Pattanayak, and Partha Sarathi Mandal. 2021. Filling MIS Vertices by Myopic Luminous Robots. *CoRR* abs/2107.04885 (2021). arXiv:2107.04885 https://arxiv.org/abs/2107.04885