An Asynchronous Maximum Independent Set Algorithm by Myopic Luminous Robots on Grids

Sayaka Kamei and Sébastien Tixeuil

Graduate School of Advanced Science and Engineering, Hiroshima University, Japan.
s10kamei@hiroshima-u.ac.jp
Sorbonne University, CNRS, LIP6, Paris, France.
Sebastien.Tixeuil@lip6.fr

Abstract. We consider the problem of constructing a maximum independent set with mobile myopic luminous robots on a grid network whose size is finite but unknown to the robots. In this setting, the robots enter the grid network one-by-one from a corner of the grid, and they eventually have to be disseminated on the grid nodes so that the occupied positions form a maximum independent set of the network. We assume that robots are asynchronous, anonymous, silent, and they execute the same distributed algorithm. In this paper, we propose two algorithms: The first one assumes the number of light colors of each robot is three and the visible range is two, but uses additional strong assumptions of port-numbering for each node. To delete this assumption, the second one assumes the number of light colors of each robot is seven and the visible range is three. In both algorithms, the number of movements is \(O(n(L + l))\) steps where \(n\) is the number of nodes and \(L\) and \(l\) are the grid dimensions.

Keywords: LCM robot systems · maximum independent set.

1 Introduction

Swarm robotics envisions groups of mobile robots self-organizing and cooperating toward the resolution of common objectives, such as patrolling, exploring and mapping disaster areas, constructing ad hoc mobile communication infrastructures to enable communication with rescue teams, etc. Our focus in this paper is the autonomous deployment of mobile robots in an unknown size rectangular area, e.g. for the purpose of establishing a communication infrastructure (if robots carry antennas) or a surveillance device (if robots carry intrusion sensors). When considering the rectangular area as a discrete structure (i.e., a graph, that depends on the antenna/sensor range: two nodes in the graph are adjacent if and only if they are within the range of the antenna/sensor), one can consider several placement strategies. Given that every location in the area must be covered by an antenna/sensor, there are two competing metrics:

1. The number of deployed robots: The cost of the deployment obviously depends linearly from the number of robots deployed.

\* Supported by project ESTATE (Ref. ANR-16-CE25-0009-03), JSPS KAKENHI No. 19K11828, and Israel & Japan Science and Technology Agency (JST) SICORP (Grant#JPMJSC1806).
2. The resilience of the infrastructure in the case robots fail unpredictably: This amounts to the number of locations that are left uncovered when a robot (or a set of robots) ceases to perform its algorithm.

Assuming full coverage is necessary, two extreme placement strategies are possible: A complete filling of each location by a robot enables maximum resilience (uncovering one location, say $C$ in Fig. 1(d), requires to disable five robots, at positions $A$, $B$, $C$, $D$ and $E$ in Fig. 1(d)) but also requires deploying one robot per location (so, the cost is highest), while a minimum dominating set strategy yields minimum cost, but poor resilience (disabling a single robot, say at $C$ in Fig. 1(c), uncovers five locations, $A$, $B$, $C$, $D$, and $E$ in Fig. 1(c)). Maximal and maximum independent set placements are somewhat more balanced, despite the fact that any robot failure will uncover its location: a maximal independent set may use as little as one-third of the robots required for a complete filling, while retaining decent resilience (e.g. in Fig. 1(a), at least two robots failures $C$ and $D$ are required to disconnect locations $A$ and $B$ beyond those initially hosting a robot); finally, a maximum independent set placement policy yields a resilience that is close to optimal (e.g. four robot failures, $A$, $B$, $C$, and $D$ disconnect only one additional location, $E$ in Fig. 1(b)) while using half of the robots required for a complete filling. In this paper, we concentrate on placing the robots according to a maximum independent set organization.

Related Works. The seminal paper for studying robotic swarms from a distributed computing perspective is due to Suzuki and Yamashita [32]. In the initial model, robots are represented as dimensionless points evolving in a bidimensional Euclidean space, and operate in Look-Compute-Move cycles. In each cycle, a robot “Looks” at its surroundings and obtains (in its own coordinate system) a snapshot containing some information about the locations of all robots. Based on this visual information, the robot “Computes” a destination location (still in its own coordinate system), and then “Moves” towards the computed location. When the robots are oblivious, the computed destination in each cycle only depends on the snapshot obtained in the current cycle (and not on the past history of actions). Then, an execution of a distributed algorithm by a robotic swarm consists in having every robot repeatedly execute its LCM cycle. Although this mathematical model is perfectly precise, it allows a great number of variants (developed over a period of 20 years by different research teams [19]), according to various dimensions, namely: sensors, memory, actuators, synchronization, and faults.

Although the seminal paper [32] focused on continuous spaces, many recent papers [19] consider robots evolving on a discrete graph (that is, robots are located on a discrete set of locations, the nodes of the graph, and may move from one location to the next if an edge exists between the two locations), as it was recently observed that discrete observations model better actual sensing devices [2]. For the particular topology we consider, the grid, many problems were previously investigated, e.g., exploration [14,15], perpetual exploration [4], scattering [3], dispersion [28], gathering [11], mutual visibility [1], pattern formation [5], and convex hull formation [21]. Similarly, the initial model considers unlimited visibility range, but actual sensors have a limited range, which makes solutions that only assume limited visibility more practical. When the evolving space is discrete, robots that can only see at a constant (in the locations
(a) Maximal Independent Set Placement  
(b) Maximum Independent Set Placement  
(c) Minimum Dominating Set Placement  
(d) Fill Placement  

Fig. 1. Possible mobile robot placements on a $7 \times 9$ grid.
graph) are called myopic. Myopic robots have successfully solved ring exploration [13], gathering in bipartite graphs [20], and gathering in ring networks [26]. Finally, another characteristic of the initial model, obliviousness, was recently dropped out in favor of a more realistic setting: luminous robots. Oblivious robots were not able to remember past actions (each new Look-Compute-Move cycle reset the local memory of the robot), while luminous robots are able to remember and communicate a finite value between two consecutive LCM cycles, using a visible light that is maintained by the robot. The number of values a robot is able to remember is tantamount to the number of different colors its light is able to show. Luminous robots were used to circumvent classical impossibility results in the oblivious model, mainly for gathering [12,33,22]. In this paper, we consider the particular combination of myopic and luminous robot model, that was previously used for ring exploration [30,29], infinite grid exploration [6], and gathering on rings [27].

The maximum independent set placement we consider in this paper is related to the benchmarking problem of geometric pattern formation initially proposed by Suzuki and Yamashita [32]. A key difference is that the target pattern is usually given explicitly to all robots (see the recent survey by Yamauchi [34]), while the maximum independent set pattern we target is only given as a constraint (as the dimensions of the grid are unknown to the robots, the exact pattern cannot be given to the robots). Unconstrained placement of robots is also known as scattering (in a continuous bidimensional Euclidean space [16,9,7], robots simply have to eventually occupy distinct positions). Evenly spreading robots in a unidimensional Euclidean space was previously investigated by Cohen and Peleg [10] and by Flocchini [17] and Flocchini et al. [18]. The bidimensional case was tackled mostly by means of simulation by Cohen and Peleg [10] and by Casteigts et al. [8]. Most related to our setting is the barrier coverage problem investigated by Hesari et al. [23]: robots have to move on a continuous line so that each portion of the line is covered by robot sensors (whose range is a fixed value) despite the robots having limited vision (whose range is twice the range of the sensor). A key difference besides the robots evolving space (continuous segment versus discrete grid) with our approach is that they consider oblivious robots and a common orientation, while we assume luminous robots and no orientation. Another closely related problem was studied by Barrière et al. [3]: uniform scattering on square grids. For uniform scattering to be solved, robots, initially at random positions, must reach a configuration where they are evenly spaced on a grid. Similarly to Hesari et al. [23], Barrière et al. [3] assume a common orientation (on both axes), that the size of the grid is \((k \times d) \times (k \times d + 1)\), where \(k \geq 2\), \(d \geq 2\), the number of robots is \((k + 1)^2\), and that each robot knows \(k\) and \(d\). They also assume that each robot has internal lights with six colors and that their visible radius is \(2d\). Under the same assumptions as Barrière et al. [3], Poudel et al. [31] proposed an algorithm needing \(O(1)\) bit memory per robot, assuming a visibility radius of \(2 \max\{d, k\}\). By contrast, we don’t assume a common orientation, we use seven or three full lights colors, and the size of the grid is arbitrary and unknown. Finally, the placement method we describe as the fill placement (see Fig. 1(d)) was investigated by Hsiang et al. [25], and by Hideg et al. [24].

In the literature, this is refereed to as the Full Light model.
Our contribution. We propose the first two solutions to the maximum independent set placement of mobile myopic luminous robots on a grid of unknown size. Robots enter at a corner of the grid, and do not share a common orientation nor chirality. In the first algorithm, each robot light can take 3 different colors, and the visibility range of each robot is two. Similarly to previous work [24], the first algorithm assumes “local” port numbers are available at each node, so that each robot can recognize its previous node. The second algorithm gets rid of the port number assumption, and executes in a completely anonymous graph. It turns out that weakening this assumption has a cost on the number of colors (7 instead of 3) and on the visibility radius (3 instead of 2). In both cases, the placement process takes $O(n(L + l))$ steps of computation, where $n$ is the number of nodes and $L$ and $l$ are the grid dimensions.

As pointed out in the above, a maximum independent set placement yields good resilience in case of robot failures for the purpose of the target application, yet makes use of half of the robots needed for a complete filling of the grid.

2 Model

We consider an anonymous, undirected connected network $G' = (V, E)$, where $V$ is a finite set of $n$ nodes $v_1, v_2, \cdots v_n$, and a specific node $v'$ (discussed below), and $E$ is a finite set of edges. We assume that the induced subgraph $G$ of $G'$ derived from the nodes except $v'$ is a $(l, L)$-grid, where $l \geq 3$ and $L \geq 3$ are two positive integers such that $l \times L = n$. Then, $G$ satisfies the following conditions: $\forall x \in [1..n], (x \mod l) \neq 0 \Rightarrow \{v_x, v_{x+1}\} \in E$, and $\forall y \in [1..l \times (L - 1)], \{v_y, v_{y+l}\} \in E$. We assume that these sizes $l$, $L$ and $n$ are unknown to the robots. Let $\delta(v)$ be the degree of node $v$ in $G'$.

The specific node $v'$ is called a Door node. Each robot enters the grid $G$ one-by-one through the Door node. We assume that $\delta(v') = 1$, and the Door node is connected to a corner node of the grid (the particular corner $v'$ is connected to is decided by an adversary). We refer to this corner as the Door corner. A robot at the Door node has to disperse through the grid while avoiding collisions. That is, two or more robots cannot occupy the same node. When the Door node becomes empty, a new robot can be placed there immediately. We use $\text{Enter\_Grid}(r_i)$ to denote an operation that makes robot $r_i$ move from the Door node to the Door corner, and $\text{Move}(r_i)$ to denote an operation that makes $r_i$ move to an adjacent node in its direction. We assume that each robot has no orientation, i.e., each robot does not know axes $x$ and $y$ of the grid in the above definition.

The distance between two nodes $v$ and $u$ is the number of edges in a shortest path connecting them. The distance between two robots $r_1$ and $r_2$ is the distance between two nodes occupied by $r_1$ and $r_2$, respectively. Two robots or two nodes are adjacent if the distance between them is one.

We assume that robots have limited visibility: an observing robot $r_i$ at node $u$ can only sense the robots that occupy nodes within a certain distance, denoted by $\phi$, from $u$. When we assume $\phi = 2$ (resp. $\phi = 3$), because we assume the network is a grid, the view of a robot is like Fig. 2(a) (resp. 2(b)) for a robot not on a border nor a corner node.

\footnote{The port numbers are local in the sense that there is no coordination between adjacent nodes to label their common edge.}
In each of these figures, the view is from a robot on the center node. For each robot $r_i$, we use $\text{view}(r_i)$ to denote the view of $r_i$. Then, we call each robot $r_j$ in $\text{view}(r_i)$ a neighboring robot of $r_i$.

Each robot $r_i$ maintains a variable $c(r_i)$ called light, which spans a finite set of states called colors. A light is persistent from one computational cycle to the next: the color is not automatically reset at the end of the cycle (see below how cycles drive the life of robots). Robot $r_i$ knows its own current color of light and can detect colors of lights of other robots in the visibility range. Robots are unable to communicate with each other explicitly (e.g., by sending messages), however, they can observe their environment, including the positions and colors of other robots, in their visibility range.

Each robot $r_i$ executes Look-Compute-Move cycles infinitely many times: (i) first, $r_i$ takes a snapshot of the environment and obtains an ego-centered view of the current configuration (Look phase), (ii) according to its view, $r_i$ decides to move or to stay idle and possibly changes its light color (Compute phase), (iii) if $r_i$ decided to move, it moves to one of its adjacent nodes depending on the choice made in the Compute phase (Move phase). At each time instant $t$, a subset of robots is activated by an entity known as the scheduler. This scheduler is assumed to be fair, i.e., all robots are activated infinitely many times in any infinite execution. In this paper, we consider the most general asynchronous model: the time between Look, Compute, and Move phases is finite but unbounded. We assume however that the move operation is atomic, that is, when a robot takes a snapshot, it sees robots’ colors on nodes and not on edges. Since the scheduler is allowed to interleave the different phases between robots, some robots may decide to move according to a view that is different from the current configuration. Indeed, during the Compute phase, other robots may move. We call a view that is different from the current configuration an outdated view, and a robot with an outdated view an outdated robot.

In this paper, the set of robots that enter the grid $G$ from a Door node constructs a maximum independent set of $G$.

---

5 The assumption that moves are atomic was show equivalent [2] to the assumption that moves are not atomic but the sensors see the robot either at the starting node or at the destination node, and no inversion of the observations is possible. For the sake of proof readability, we retain the former hypothesis.
Definition 1. An independent set $I$ of $G$ is a subset of $V \setminus \{v'\}$ such that no two nodes in $I$ are adjacent on $G$. A maximum independent set is an independent set containing the largest possible number of nodes for $G$.

3 Proposed Algorithms

In this section, we present two algorithms to construct a maximum independent set when the Door node is connected to a corner node. The first algorithm makes the assumption that outgoing edges are labeled “locally” (that is, the labels may be inconsistent for the two adjacent nodes of the edge, however, a node must assign distinct labels to different outgoing edges), and assumes that each robot is endowed with a light enabling 3 colors and has visibility radius 2. To remove the edge labeling assumption, the second algorithm makes use of more colors (7 colors are needed) and a larger visibility radius (i.e., 3). As a result, it operates in the “vanilla” Look-Compute-Move model (no labeling of nodes or edges, etc.). In both algorithms, we assume no agreement on the grid axes or directions.

3.1 Algorithm with 3 colors lights, $\phi = 2$, and port numbering

First, we propose an algorithm that assumes three light colors are available (and referred to as $F$, $p_1$, and $p_2$), and that $\phi = 2$. The color $F$ means that the robot finished the execution of the algorithm, and stops its execution. Colors $p_1$ and $p_2$ are used when the robot still did not finish its execution. We say that if the light color of a robot $r_i$ is $F$, then $r_i$ is Finished. Initially, the color of the light $c(r_i)$ for each robot $r_i$ is $p_1$.

For this algorithm, we add the following assumptions to the model in Section 2:

- For each node of the grid, adjacent nodes (except the Door node) are arranged in a fixed order, and this order is only visible for robots on the node as port numbers. The order does not change during the execution.
- Each robot can recognize the node it came from when at its current node.

These assumptions are those considered in related work for the filling problem [24]. Note that, the latter assumption can be implemented using four additional colors to remember the port number of the previous node.

The strategy of the routing to construct a maximum independent set is as Fig. 3. In this figure, the thick white circle represents a Finished robot, and the diagonal (resp. horizontal) striped circle represents a robot with $p_1$ (resp. $p_2$). First, each robot $r_i$ starts with $c(r_i) = p_1$ from the Door node (Fig. 3(a)). On the Door corner, each robot chooses an adjacent node according to the edge with the maximal port number. Each robot moves on the first border to keep the distance from its predecessor two or more hops. Each robot arrives at the first corner, then changes $c(r_i)$ to $p_2$. After that, the first robot $r_1$ goes through the second border, eventually arrives at the second corner (Fig. 3(b)), and changes $c(r_1)$ to $F$. We call this second corner the diagonal corner. The successor $r_i$ follows its predecessor $r_j$ while striving to keep a distance of at least two. If $r_i$ has $c(r_i) = p_2$, and $r_j$ is Finished two hops away, then $r_i$ changes $c(r_i)$ to $F$ (Fig. 3(c)). If $r_i$ with $c(r_i) = p_1$ observes that $r_j$ is Finished two hops away, $r_i$ changes $c(r_i)$ to
\( p_2 \) and makes the next line (Fig. 3(d)). By repeating such elementary steps, eventually, a maximum independent set can be constructed (Fig. 3(f)). Because each robot can recognize its previous node by the assumption, it can recognize its predecessor and its successor if there are two neighboring non-Finished robots.

The algorithm description is in Algorithm 1. The number of each rule represents its priority, a smaller number denoting a higher priority. In this algorithm, we use the definitions of view types in Fig. 4–9. In these figures, each view \( \text{view}(r_i) \) is from robot \( r_i \) represented by the center filled circle. The dotted circle without frame border represents the previous node wherefrom \( r_i \) moved to the current node. If \( \text{view}(r_i) \) is a view with an arrow from \( r_i \) not on the Door node (like OnCP1 in C (Fig. 6)), the arrow represents the direction of \( \text{Move}(r_i) \) operation. If \( r_i \) is on the Door node (like Door1 in D (Fig. 4)), the arrow represents the operation \( \text{Enter}_G\text{rid}(r_i) \). If the previous node is not the Door node, the circle with diagonal stripes or horizontal stripes (which is adjacent to the previous node) represents the node where \( r_i \)'s successor robot may be hosted. So, such a node may: (i) host the successor of \( r_i \), (ii) be empty, or (iii) do not exist. If the successor robot is on the diagonal (resp. horizontal) striped node, it has \( p_1 \) (resp. \( p_1 \) or \( p_2 \)). The diagonal striped node can be the Door node under the grid size hypothesis, i.e., the diagonal striped node in P1Stop cannot be the Door node due the grid size hypothesis. If the previous node is the Door node, the successor robot can be the previous node by assumption. The thick white circle represents a Finished robot that must be on the node. The circle with vertical stripes represents either a node hosting a Finished robot, or no node. A node with a dashed white square represents an empty node, if the node exists on the grid. All other empty nodes must exist on the grid and host no robot. In our classification, each type of views may include several possible views. For example, in P1Stop (Fig. 5), the upper
Algorithm 1 Algorithm for a maximum independent set placement with 3 colors light.

| Colors | $F, p_1, p_2$ |
|--------|---------------|
| Initialization | $c(r_i) = p_1$ |

**Rules on node $v$ of robot $r_i$**

0: $c(r_i) = p_1 \land \text{view}(r_i) \in D \rightarrow \text{Enter Grid}(r_i)$;

1: $c(r_i) = p_1 \land \text{view}(r_i) \in F_1 \rightarrow c(r_i) := F$;

2: $c(r_i) = p_1 \land \text{view}(r_i) \in C \rightarrow c(r_i) := p_2; \text{Move}(r_i)$;

3: $c(r_i) = p_1 \land \text{view}(r_i) \in M_1 \rightarrow \text{Move}(r_i)$;

4: $c(r_i) = p_2 \land \text{view}(r_i) \in F_2 \rightarrow c(r_i) := F$;

5: $c(r_i) = p_2 \land \text{view}(r_i) \in M_2 \rightarrow \text{Move}(r_i)$.

![Fig. 4. Definition of views in $D$ while $p_1$.](image)

![Fig. 5. Definition of views in $F_1$ while $p_1$.](image)

adjacent empty node may be a corner, then the top node with vertical stripes does not exist. Thus, in the type $P1Stop$, there are five possible views, depending on whether the top node exists or not, the bottom node exists or not, and whether the successor is in the view or not. Note that all combinations are not feasible, since e.g. if the bottom node does not exist, then the previous node is the Door node, and the successor is on the Door node, which in turn implies that the top Finished robot must exist (due to the grid size hypothesis).

**Proof of Correctness** Without loss of generality, let $L$ be the size of the border that is connected to the Door corner by an edge with a maximal port number among two edges of the Door corner. Let $l$ be the other size of the border. We call the $L$-size border connected to the Door corner “the first border”, and the $l$-size border not-connected to the Door corner “the second border”, like Fig. 3(b). Additionally, we call the second border 0-line, and count the lines in the following way: the line that is adjacent and parallel to 0-line is 1-line, and the border that is connected to the Door corner but not the first border is $(L - 1)$-line.

First, we show that robots cannot collide.

**Lemma 1.** Robots cannot collide when executing Algorithm 1.

**Proof.** If there exists an outdated robot $r_o$ that is to move using its outdated view, the outdated view is one of the types in $D, C, M_1, \text{or } M_2$ by the definition of the algorithm. Thus, if a collision with $r_o$ occurs, then $r_o$’s view is one of the types in $D, C, M_1, \text{or } M_2$. Therefore, the robots do not collide.
\[ \delta = 2 \]

(a) OnCP1

\[ \delta = 3 \]

(b) ColP1A

(c) ColP1B

Fig. 6. Definition of views in \( C \) while \( p_1 \).

\[ \delta = 3 \]

(a) StartP1

(b) MovP1

(c) GoCo

Fig. 7. Definition of views in \( M_1 \) while \( p_1 \).

\( M_2 \). In that case, because a Finished robot does not move forever, a non-Finished robot in \( r_o \)'s view may have moved, or other non-Finished robot came into the visible region of \( r_o \).

On the Door corner, if a robot \( r_i \) cannot see other robots, then its view is \( \text{StartP1} \) in \( M_1 \). Then, because \( r_i \) has \( c(r_i) = p_1 \) initially, its view becomes \( \text{MovP1} \) in \( M_1 \). By the definition of \( \text{MovP1} \), \( r_i \) moves only on the first border according to the degree of nodes until it arrives at the end of the first border, or it can see Finished robots. Then, the first border is one-way because each robot can recognize its previous node. Additionally, if \( r_i \) can see other non-Finished robots than its successor, then \( r_i \) cannot move because there is no such rule. That is, \( r_i \) keeps the distance from its predecessor (if it exists) two or more hops on the first border. Thus, on the first border (while \( c(r_i) = p_1 \)), if \( r_i \) has an outdated view in \( D, C, \) or \( M_1 \), the current configuration can be the same view type as its outdated view, because only \( r_i \)'s successor is allowed to move toward \( r_i \). Thus, \( r_i \) cannot collide with other robots while \( c(r_i) = p_1 \).

Consider when \( r_i \) arrives at the end of the first border, or can see Finished robots.

- If \( \text{view}(r_i) \) becomes \( \text{P1Stop} \) or \( \text{OnCP1F} \) in \( F_1 \), then \( r_i \) changes its color to \( F \) by Rule 1.
- If \( \text{view}(r_i) \) becomes \( \text{OnCP1} \) in \( C \), then \( r_i \) changes its color to \( p_2 \), and moves to the second border by Rule 2. After that, \( \text{view}(r_i) \) becomes \( \text{MovP2} \) in \( M_2 \) until it arrives at the diagonal corner (i.e., \( \text{OnCP2} \) in \( F_2 \)), or it can see Finished robots on the second border (i.e., \( \text{P2Stop} \) in \( F_2 \)). By the definition of \( \text{MovP2} \), each robot
moves on the second border according to the degree of nodes, and the second border is one-way. By the definition of the algorithm, there is no rule to make $r_i$ stray from the second border. Then, $r_i$ keeps the distance from its predecessor (if it exists) two or more hops on the second border.

- If $\text{view}(r_i)$ becomes GoCo in $\mathcal{M}_1$, then $r_i$ moves to the adjacent node occupied by a Finished robot on the first border and $\text{view}(r_i)$ becomes ColP1A in $\mathcal{C}$.

- If $\text{view}(r_i)$ becomes ColP1A or ColP1B in $\mathcal{C}$, then $r_i$ changes its color to $p_2$, and moves to one of the lines. Without loss of generality, let the line be $m$-line where $m > 0$. Then, robots on $(m - 1)$-line are Finished and $(m + 1)$-line is empty (if it exists on the grid) by the definition of ColP1A or ColP1B. Thus, after that, $\text{view}(r_i)$ becomes MovP2A or MovP2B in $\mathcal{M}_2$ until it arrives at the end of $m$-line (i.e., P2StopA or P2StopB in $\mathcal{F}_2$), or it can see a Finished robot on $m$-line (i.e., P2StopC in $\mathcal{F}_2$). By the definition of the algorithm, there is no rule to make $r_i$ stray from $m$-line. By the definitions of MovP2A and MovP2B, $m$-line is also one-way, and $r_i$ keeps the distance from its predecessor (if it exists on the line) two or more hops.

In any case, on each line (while $c(r_i) = p_2$), if $r_i$ has an outdated view in $\mathcal{C}$ or $\mathcal{M}_2$, then the current configuration is the same view type as its outdated view, because only $r_i$’s successor is allowed to move toward $r_i$. Thus, $r_i$ cannot collide with other robots while $c(r_i) = p_2$.

Thus, the lemma holds. \qed
Next, we show that Algorithm 1 constructs a maximum independent set.

**Lemma 2.** The first robot \( r_1 \) moves to the diagonal corner, and \( c(r_1) \) becomes \( F \) on the corner.

*Proof.* When the first robot \( r_1 \) is in the Door node, then its view is \( \text{Door} \) in \( D \). Thus, it moves to the Door corner by \( \text{Enter\_Grid}(r_1) \), and its view becomes \( \text{StartP}1 \) in \( M_1 \). Then, \( r_1 \) moves to the adjacent node through the edge with the maximal port number by Rule 3. Then, because it is the first robot, its view becomes \( \text{MovP}1 \) in \( M_1 \) and moves to the adjacent node on the first border by Rule 3.

By the proof of Lemma 1, the first border and second border are one-way, and any other robots cannot pass \( r_1 \) on these borders. Thus, \( \text{view}(r_1) \) remains \( \text{MovP}1 \), and \( r_1 \) moves on the first border according to the node degree. Therefore, \( r_1 \) arrives at the end of the first border eventually, and then \( \text{view}(r_1) \) becomes \( \text{OnCP}1 \) in \( C \). After that, by Rule 2, \( r_1 \) changes its color to \( p_2 \) and moves to the adjacent node on the second border. \( \text{view}(r_1) \) becomes \( \text{MovP}2 \) in \( M_2 \), \( r_1 \) moves towards the next (diagonal) corner through the second border according to the node degree by Rule 5, eventually \( \text{view}(r_1) \) becomes \( \text{OnCP}2 \) in \( F_2 \). By Rule 4, because \( c(r_1) = p_2 \), \( r_1 \) changes its color to \( F \) on the corner.

Thus, the lemma holds. \( \square \)

**Lemma 3.** The first \( \lceil l/2 \rceil \) robots move to the second border, and their color becomes \( F \). Additionally, nodes on the second border are occupied by a robot or empty alternately from the diagonal corner.

*Proof.* By Lemma 2, the first robot \( r_1 \) eventually becomes Finished on the diagonal corner.

First, we consider the second robot \( r_2 \), which follows \( r_1 \) in the case that \( l > 3 \). By the assumption, \( r_2 \) appears to the Door node just after \( r_1 \) enters into the grid. Because \( r_1 \) does not become Finished before it arrives at the diagonal corner, \( r_2 \) can move from the Door node only when \( \text{view}(r_2) = \text{Door} \) in \( D \) holds. After that, by the definition of the algorithm, \( r_2 \) moves in the same way as \( r_1 \), \( c(r_2) \) becomes \( p_2 \) on the end of the first border eventually and \( r_2 \) moves on the second border. Finally, \( r_2 \) can see \( r_1 \) on the diagonal corner two hops away. Then, there is no rule such that \( r_2 \) executes before \( c(r_1) \) becomes \( F \). Because of Lemma 2, \( \text{view}(r_2) \) eventually becomes \( \text{P2Stop} \) in \( F_2 \). Then, by Rule 4, \( c(r_2) \) becomes \( F \).

In the case that \( l = 3 \), when \( r_2 \) arrives at the end of the first border, \( \text{view}(r_2) \) becomes \( \text{OnCP}1 \) in \( F_1 \) and \( c(r_2) \) becomes \( F \) by Rule 1. Note that, in any case, the distance between \( r_1 \) and \( r_2 \) is two hops when they are Finished.

For the successors of \( r_2 \), we can discuss their movements in the same way as \( r_2 \). Thus, by the definitions of \( \text{OnCP}1 \) and \( \text{P2Stop} \), the distance between a robot and its successor is two hops when they are Finished on the second border. Therefore, on the second border, beginning with the diagonal corner, every even node is occupied, and the number of robots is \( \lceil l/2 \rceil \). If \( l \) is odd, when the \( \lceil l/2 \rceil \)-th robot arrives at the end of the first border, its view becomes \( \text{OnCP}1 \) and it changes its color to \( F \) by Rule 1. Otherwise, it changes its color to \( p_2 \) and moves to the second border.

Thus, the lemma holds. \( \square \)
Lemma 4. From the \( (\lceil l/2 \rceil + 1) \)-th to the \( l \)-th robots, each robot moves to the 1-line, and its color becomes \( F \). Additionally, nodes on the 1-line are empty or occupied by a robot alternately, beginning with an empty node.

Proof. By Lemma 3 \( \lceil l/2 \rceil \) robots on 0-line eventually become Finished. Let \( r_1 \) be the \( (\lceil l/2 \rceil + 1) \)-th robot, \( r_j \) be the \( (\lceil l/2 \rceil) \)-th robot that is the predecessor of \( r_l \). \( r_l \) moves from the Door node in the same way as \( r_j \) while \( c(r_i) = p_1 \). Because robots on 0-line become Finished eventually, one of the following two cases occurs: (1) if \( l \) is odd, \( \text{view}(r_j) \) becomes \( \text{GoCo} \) in \( M_1 \), because the end of the first border is occupied by \( r_j \), or (2) if \( l \) is even, \( \text{view}(r_i) \) becomes \( \text{ColP1B} \) in \( C \), because the end of the first border is empty but its adjacent node on the 0-line is occupied by \( r_l \).

In case (1), by Rule 3, \( r_j \) moves to the node in front of the end of the first border, \( \text{view}(r_i) \) becomes \( \text{ColP1A} \) in \( C \). Then, by Rule 2, \( c(r_i) \) becomes \( p_2 \) and \( r_l \) moves to 1-line. After that, if \( l = 3 \), \( \text{view}(r_i) \) becomes \( \text{P2StopA} \) in \( F_2 \) and \( r_i \) changes its color to \( F \) by Rule 4. Otherwise, because \( r_1 \) can see Finished robots on 0-line, \( \text{view}(r_i) \) becomes \( \text{MovP2A} \) in \( M_2 \). Then, because the nodes on 0-line are occupied alternately by Lemma 3 \( \text{view}(r_j) \) becomes \( \text{MovP2B} \) and \( \text{MovP2A} \) in \( M_2 \) alternately by the execution of Rule 5. Thus, \( r_i \) moves toward the other side border that is parallel to the first border by Rule 5, and \( \text{view}(r_i) \) eventually becomes \( \text{P2StopA} \) because the diagonal corner is occupied by a Finished robot (Lemma 2). Then, by Rule 4, \( c(r_i) \) eventually becomes \( F \). Because \( l \) is odd, \( \lceil l/2 \rceil \) successors of \( r_i \) follow \( r_i \), and eventually, their views become \( \text{P2StopC} \) in \( F_2 \), and they change their colors to \( F \) by Rule 4 on 1-line.

In case (2), \( r_i \) also changes its color to \( p_2 \), and moves to 1-line by Rule 2. After that, because \( r_i \) can see Finished robots on 0-line, \( \text{view}(r_i) \) becomes \( \text{MovP2B} \) in \( M_2 \). Then, in the same way as for case (1), \( l/2 - 1 \) robots including \( r_l \) become Finished on 1-line. After that, the view of the next robot \( r_l \) (\( l \)-th robot) becomes \( \text{P1Stop} \) in \( F_1 \) on the intersection between the first border and 1-line, and \( r_l \) becomes Finished by Rule 1.

Thus, the lemma holds. \( \square \)

Lemma 5. The distance between any two robots on the grid is two hops after every robot becomes Finished.

Proof. By the definitions of \( F_1 \) and \( F_2 \), the distance between a robot \( r_i \) and its predecessor is two hops after \( r_i \) becomes Finished if the predecessor is on the same line as \( r_i \). Thus, when the robots on \( m \)-line \((0 < m < L - 1)\) become Finished, if there are two adjacent Finished robots, then there is a robot \( r_j \) on \( m \)-line that cannot move from the node that is adjacent to a node occupied by a Finished robot on \((m - 1)\)-line. However, by the same argument as in Lemmas 3 and 4, if \( m \) is odd (resp. even), the nodes on \( m \)-line are occupied alternately beginning with an empty node (resp. occupied node) because the nodes on \((m - 1)\)-line are also occupied alternately beginning with an occupied node (resp. empty node). Thus, before such \( r_j \) becomes Finished, \( r_j \) has a view of type \( \text{MovP2B} \) and can move by Rule 5, i.e., it cannot exist.

Now, to consider the end of the execution of the algorithm, we consider \((L - 1)\)-line when nodes on \((L - 2)\)-line are occupied by Finished robots. The \((L - 1)\)-line is a border connected to the Door node. Then, if both \( l \) and \( L \) are odd or both are even, the view from the Door node becomes \( \text{Door1} \), otherwise \( \text{Door2} \) (See Fig. 10).
– If the view from the Door node is Door1, the robot on the Door node moves to the Door corner by Rule 0. Then, the view from the Door corner is ColP1B in C. By the above discussion, the view from the Door corner eventually becomes P1Stop in F1, thus the final robot on the Door corner becomes Finished by Rule 1. Then, any other robots cannot enter into the grid because there is no such rule.

– If the view from the Door node is Door2, the view from the Door corner is ColP1A in C. Then, the empty node v that is adjacent to the Door corner is eventually occupied by a Finished robot on (L − 1)-line. After that, any other robots on the Door node cannot enter into the grid because there is no such rule.

Thus, the lemma holds.

Lemma 6. Every robot on the grid is eventually Finished.

Proof. By the proofs of Lemmas 1–5, the transitions of the view type of each robot are shown as Fig. 11. Thus, the lemma holds.

Theorem 1. Algorithm 1 constructs a maximum independent set of occupied locations on the grid.

Proof. By Lemma 5, distances between any two occupied nodes are two. On a grid, only checkers patterns satisfy this constraint. When at least one dimension of even, there are as many occupied locations as non-occupied locations, so any checkers pattern is a maximum independent set (see Fig. 10(a) and Fig. 10(b)). When both dimensions are odd, there may be either one more occupied locations than non-occupied locations, or the contrary (See Fig. 10(c)). The situation that corresponds to the maximum independent set is the one with occupied locations in the corners, which is what our algorithm constructs. Hence, the theorem holds.

Lemma 7. When a maximum independent set is constructed, ⌈n/2⌉ robots are on the grid.

Proof. By the proofs of Lemmas 3–5 nodes on the even-numbers (resp. odd-numbers) lines are occupied by ⌈l/2⌉ (resp. ⌊l/2⌋) robots. Therefore, if L is even, the number of robots in the maximum independent set is LL/2. If L is odd, the number of robots in the maximum independent set is l[L/2] + ⌈l/2⌉. Because n = lL, the lemma holds.

To analyze the time complexity of the algorithm, we count the sum of individual executions of rules.

Theorem 2. The time complexity by Algorithm 1 is O(n(L + l)) steps.

Proof. The first robot moves L + l − 1 steps and becomes Finished, thus it executes L + l steps. The first robot moves the longest way. Therefore, by Lemma 7, the sum of the number of steps is O(n(L + l)). Thus, the theorem holds.
In this section, we relax both additional hypotheses made in Section 3.1. So, there is no local labeling of edges, and robots cannot recognize the node they came from when at a particular node. Instead, we assume $\phi = 3$, and that seven light colors are available for each robot $r_i$, whose colors are named $F$, $p_1(k_i)$, and $p_2(k_i)$ ($k_i \in \{0, 1, 2\}$). The value of $k_i$ represents the order of the robot (the notion of order is explained in detail in the sequel). Initially, the color of light $c(r_i)$ for each robot $r_i$ is $p_1(0)$, that is, $k_i = 0$.

The strategy to construct a maximum independent set is the same as Algorithm 1. However, on the Door node, the first robot chooses an adjacent node on the grid arbitrarily (that is, the choice can be taken by an adversary), and the other robots just follow it.
The algorithm description is in Algorithm 2. In this algorithm, we use the definitions of view types in Fig. 12-18. Unlike Algorithm 1, we do not use dotted circles without frames, since the previous node can no longer be recognized by the robot. The circle with diagonal stripes or horizontal stripes represents $r_i$’s successor robot, which must be there (We explain later in the text how to recognize predecessor and successor). If the successor robot is on the diagonal (resp. horizontal) striped node, it has $p_1$ (resp. $p_1$ or $p_2$). While there are two types of successor in each view type of Fig. 17-18 exactly one must be present. The waffle circle represents $r_i$’s non-Finished predecessor robot. If the waffle circle is with a thick border, the predecessor must be there. Otherwise, it may be an empty node or non-existent node. For example, in the type Door2 (Fig. 13), when the predecessor has just become Finished on the upper node with the thick white circle, the waffle circle with the dotted border is actually an empty node. The square with a question mark represents any node in Door0 (Fig. 12).

By the strategy of the routing described above, each robot enters the grid one-by-one and walks in line on the grid. Therefore, each robot has a successor, and each robot except the first one has a predecessor. In this algorithm, each robot has a variable $k_i$ to distinguish them. On the Door node, each robot $r_i$ sets its $k_i$ (Door0 in Fig. 12). If $r_i$ is the first robot, keeps $k_i = 0$. Otherwise, if its predecessor robot $r_j$ on the Door corner has $p_1(k_j)$, then $k_i$ is set to $(k_j + 1) \mod 3$. The value of $k_i$ is kept in $c(r_j)$ such that $p_1(k_i)$ and $p_2(k_i)$, and $k_i$ is not changed after that. On the Door corner, each robot $r_j$ waits for its successor $r_i$ on the Door node to set its value $k_i$ before $r_j$ moves (ColP1A1 and ColP1B1 in Fig. 15 and StartP10, StartP11, MovP13, MovP14 and GoCo1 in Fig. 16). By this mechanism, each robot $r_i$ recognizes that its neighboring non-Finished robot $r_j$ with smaller (resp. larger) $k_j$ value than $k_i$ is its predecessor (resp. successor). Let $SetC(r_i)$ be the operation such that $c(r_i) := p_1((k_j + 1) \mod 3)$, where $r_j$ is the robot on the Door corner and $c(r_j) = p_1(k_j)$.

Only the first robot selects its way arbitrarily from the Door corner (StartP10 in Fig. 16). After that, the other robot $r_i$ can move only when the distance from its pre-

---

6 If $k_i = 2$ (resp. 0, 1), it is larger than 1 (resp. 2, 0), but smaller than 0 (resp. 1, 2).
Algorithm 2 Algorithm for a maximum independent set placement with 7 colors light.

Colors $F, p_1(k_i), p_2(k_i)$, where $k_i \in \{0, 1, 2\}$

Initialization $c(r_i) = p_1(0)$

Rules on node $v$ of robot $r_i$

0-1: $c(r_i) = p_1(0) \land \text{view}(r_i) = \text{Door0} \rightarrow \text{SetC}(r_i)$;

0-2: $c(r_i) = p_1(k_i) \land \text{view}(r_i) \in D' \rightarrow \text{Enter_Grid}(r_i)$;

1: $c(r_i) = p_1(k_i) \land \text{view}(r_i) \in F' \rightarrow c(r_i) := F$;

2: $c(r_i) = p_1(k_i) \land \text{view}(r_i) \in C' \rightarrow c(r_i) := p_2(k_i); \text{Move}(r_i)$;

3: $c(r_i) = p_2(k_i) \land \text{view}(r_i) \in M'_1 \rightarrow \text{Move}(r_i)$;

4: $c(r_i) = p_2(k_i) \land \text{view}(r_i) \in F'_2 \rightarrow c(r_i) := F$;

5: $c(r_i) = p_2(k_i) \land \text{view}(r_i) \in M'_2 \rightarrow \text{Move}(r_i)$;

Proof of Correctness Without loss of generality, let $L$ be the size of the border chosen by the first robot on the Door corner in $\text{StartP10}$ in $M'_1$. Let $l$ be the other size of the border. In the same way as Algorithm 1, we define “the first border”, “the second border”, and lines.

In the following, we first show that each robot can recognize its successor and robots cannot collide.

Lemma 8. Each non-Finished robot except the first robot can recognize its predecessor and successor if it keeps two neighboring non-Finished robots.

Proof. By the assumption, each robot $r_1$ initializes its $c(r_i)$ to $p_1(0)$. The view of the first robot $r_1$ on the Door node is $\text{Door1}$ in $D'$, thus it moves to the Door corner with $c(r_1) = p_1(0)$ by Rule 0-2. After that, by the assumption, its successor $r_2$ appears on the Door node with $c(r_2) = p_1(0)$. Thus, $r_1$ cannot move until its view becomes $\text{StartP10}$.
in $\mathcal{M}_1$, i.e., $r_2$ has $c(r_2) = p_1((k_1 + 1 \mod 3)) = p_1((0 + 1 \mod 3)) = p_1(1)$. Thus, $\text{view}(r_2)$ on the Door node becomes $\text{Door0}$ and $r_2$ sets $c(r_2)$ to $p_1(1)$ by Rule 0-1. Then, $\text{view}(r_1)$ becomes $\text{StartP10}$ in $\mathcal{M}_1$, $r_1$ selects the first border arbitrarily and moves on the border by Rule 3.

Consider a robot $r_i$ sets $c(r_i)$ to $p_1(k_i)$ on the Door node in $\text{Door0}$. Then, its predecessor $r_h$ on the Door corner has $c(k_h)$ where $k_i = (k_h + 1) \mod 3$ by the definition of $\text{SetC}(r_i)$. After $r_i$ enters the grid, on the Door corner, it cannot move until its successor $r_j$ sets $c(r_j)$ to $p_1((k_j + 1) \mod 3)$ by the definitions of $\text{StartP11}$, $\text{MovP13}$, $\text{MovP14}$, $\text{GoCo1}$ (in $\mathcal{M}_1$), $\text{ColP1A1}$ or $\text{ColP1B1}$ (in $\mathcal{C}$). Therefore, each robot $r_i$ on the grid has its order modulo 3 as its value $k_i$, and the value $k_i$ is not changed after that. Thus, if each robot keeps two neighboring non-Finished robots, it can recognize its neighboring non-Finished robot with smaller (resp. larger) $k$ value than its own as its predecessor (resp. successor), lemma holds.

**Lemma 9.** While the light color is $p_1(k_i)$, each robot $r_i$ can recognize its successor, and cannot collide.

**Proof.** If there exists an outdated robot $r_o$ that is to move according to its outdated view, the view type is in $\mathcal{D}_i$, $\mathcal{C}_i$, $\mathcal{M}_1$, and $\mathcal{M}_2$ by the definition of the algorithm. Thus, if a collision with $r_o$ occurs, then $r_o$’s view type is in $\mathcal{D}_i$, $\mathcal{C}_i$, $\mathcal{M}_1$, and $\mathcal{M}_2$. In that case, because a Finished robot does not move forever, it could be that a non-Finished robot in $r_o$’s view moved, or that another non-Finished robot came into the visible region of $r_o$.

The first robot $r_1$ keeps its color $c(r_1) = p_1(0)$. On the Door node, the view of $r_1$ becomes $\text{Door1}$ in $\mathcal{D}_i$, and $r_1$ moves to the Door corner. After that, $r_1$ can move only when its view becomes $\text{StartP10}$ in $\mathcal{M}_1$, i.e., $c(r_2)$ has to be set to $p_1(1)$ by $\text{SetC}(r_2)$, where $r_2$ is $r_1$’s successor. Thus, the view of $r_2$ on the Door node becomes $\text{Door0}$, and eventually $r_2$ sets its color to $p_1(1)$. Then, $r_1$ selects one border as the first border arbitrarily in $\text{StartP10}$ and moves. Because the distance from $r_2$ becomes two, $\text{view}(r_1)$ becomes $\text{MovP10}$ and $r_1$ moves one hop. Then, because there is no rule to move for $r_1$ when the distance from $r_2$ is three, $r_1$ cannot move. Thus, $\text{view}(r_2)$ becomes $\text{Door1}$ in

![Fig. 14. Definition of views in $\mathcal{F}_1$ while $p_1$.](image-url)
A Maximum Independent Set Algorithm by Myopic Luminous Robots on Grids

\( \delta = 2 \)

Fig. 15. Definition of views in \( C' \) while \( p_1 \).

\( D' \) and \( r_2 \) enters the grid. After that, \( r_1 \) can move because the distance from \( r_2 \) is two, i.e., \( \text{MovP10} \) in \( M'_1 \) by Rule 3. Thus, by Rule 3, \( r_2 \) can move from the Door corner only when \( \text{view}(r_2) \) becomes \( \text{StartP11} \) in \( M'_1 \). That is, when \( r_2 \) can move, the distance between \( r_1 \) and \( r_2 \) is three, and \( r_2 \)'s successor \( r_3 \) has \( c(r_3) = p_1(2) \) by the definition of \( \text{StartP11} \). By the definition of \( M'_1 \), they move only on the first border according to the degree of nodes until they arrive at the end of the first border. While they move on the first border, only \( r_1 \) and \( r_3 \) are neighbors for \( r_2 \), and \( r_3 \)'s successors also follow \( r_3 \) in the same way as \( r_2 \). When the view of robots become \( \text{OnCP1} \) in \( C' \) (i.e., they arrive at the end of the first border), they change their colors from \( p_1(k_i) \) to \( p_2(k_i) \) in the same order as they entered the grid.

By the same argument, each robot \( r_i \) moves only on the first border using the degree of nodes while \( c(r_i) = p_1(k_i) \) holds, by the definition of the views in \( M'_1 \). Then, on the first border, if \( r_i \) is not on the Door node or the Door corner, \( r_i \) can move only when the distance from its predecessor \( r_h \) is three and from its successor \( r_j \) is two. That is, while \( r_i \) moves on the first border, \( r_j \) follows \( r_i \). Then, by the definition of the views in \( M'_1 \), while \( r_i \) moves on the first border, there are at most two non-Finished neighboring robots \( r_h \) and \( r_j \) for \( r_i \) and they are kept by \( r_i \)'s movement, i.e., robots move on the first border keeping in the order they entered the grid. By the definition of the algorithm, only when \( r_h \) becomes Finished two hops away by \( F'_1 \) or \( F'_2 \), the number of non-Finished neighboring robots for \( r_i \) becomes one, but \( r_i \) keeps \( r_j \) with \( k_i < k_j \) in its view and recognizes \( r_j \) as its successor. By Lemma 8 on the first border, each non-Finished robot can always recognize its successor, that is, each robot can recognize
its direction. Therefore, the first border is one-way. Thus, while $c(r_i) = p_1(k_i)$ holds, if the view is in $D'$, $C'$, or $M_1'$, $r_i$ cannot become outdated as any non-Finished robot cannot come into $r_i$’s visible region, and any non-Finished robots in $r_i$’s view cannot move. That is, each robot cannot collide with other robots.

For each robot $r_i$, when its view becomes in $F_1'$ on the first border, $r_i$ changes its color to $F$ by Rule 1. When its view belongs to $C'$ on the first border, $r_i$ changes its color from $p_1(k_i)$ to $p_2(k_i)$ and changes its direction to a line by Rule 2. Thus lemma holds.

**Lemma 10.** While the light color is $p_2(k_i)$, each robot $r_i$ can recognize its successor, and cannot collide.

**Proof.** Consider the time $t$ when each robot $r_i$ changes its color to $p_2(k_i)$ on the first border. Then, its view is in $C'$ by Rule 2, and $r_i$ moves to a line. By the proof of Lemma 9.
and the definition of the views in $C'$, $r_i$'s successor $r_j$ is two hops behind at $t$. After that, by the definition of views in $M'_2$, $r_i$ can move only when the distance from $r_j$ is two and the distance from its non-Finished predecessor $r_h$ (if exists) is three. Thus, after $r_i$ moves by the view in $M'_2$, $r_i$ cannot move unless $r_j$ moves.

- If $\text{view}(r_i)$ is $\text{OnCP1}$ at $t$, $r_i$ moves to the 0-line (i.e., the second border) and $r_j$ also follows $r_i$. After that, $\text{view}(r_i)$ becomes $\text{MovP2}$ in $M'_2$ until $r_i$ arrives at the diagonal corner (i.e., $\text{OnCP2}$ in $F'_2$) or $r_h$ becomes Finished on 0-line (i.e., $\text{P2Stop}$ in $F'_2$). By the definition of $\text{MovP2}$ in $M'_2$, $r_i$ moves on the second border according to the degree of nodes. By the definition of the algorithm, there is no rule to make $r_i$ stray from the second border. Then, by the definition of $\text{MovP2}$ in $M'_2$, $r_i$ keeps the distance from $r_j$ two or three hops and has at most two non-Finished neighboring robots, while $r_i$ moves on the second border. By this distance, these non-Finished neighboring robots are kept by the movement. Because robots on the second border keep the same order as when they entered the grid, only when $r_h$ becomes Finished by $F'_2$ (i.e., $\text{P2Stop}$ or $\text{OnCP2}$), or $r_i$ is the first robot, the number of non-Finished neighboring robots for $r_i$ becomes one. Then, $r_i$ can recognize $r_j$ as its successor, because $r_j$ is always in $\text{view}(r_i)$ and $k_i < k_j$ holds. Thus, by Lemma 8, $r_i$ can always recognize $r_j$ as its successor, and the second border is one way.
If $\text{view}(r_i)$ is $\text{ColP1A0}$ (resp. $\text{ColP1A1}$) at $t$, $r_i$ moves on a line except 0-line and $(L-1)$-line (resp. $(L-1)$-line) and $r_j$ also follows $r_i$. Without loss of generality, let the line be $m$-line where $m > 0$. Then, by the definition of $\text{ColP1A0}$ (resp. $\text{ColP1A1}$), robots on $(m-1)$-line are Finished and $(m+1)$-line is empty (if it exists on the grid). Thus, after that, because $r_j$ follows $r_i$, $\text{view}(r_i)$ becomes $\text{MovP2A}$ or $\text{MovP2B}$ in $M'_2$ while $p_2$. By the definition of the algorithm, there is no rule to make $r_i$ stray from $m$-line. By the definitions of $\text{MovP2A}$ and $\text{MovP2B}$ in $M'_2$, $r_i$ keeps the distance from $r_j$ two or three hops, and has at most two non-Finished neighboring robots, while $r_i$ moves on $m$-line. By this distance, these non-Finished neighboring robots are kept by the movement. Because robots on $m$-line keep the same order as when they entered the grid, only when $r_h$ becomes Finished on $m$-line by $F'_2$ (i.e., $P2\text{StopA}$, $P2\text{StopB}$ or $P2\text{StopC}$), or $r_i$ is the first robot for $m$-line (i.e., $r_h$ is Finished on the intersection of the first border and $(m-1)$-line in $\text{ColP1A0}$ (resp. $\text{ColP1A1}$)), the number of non-Finished neighboring robots for $r_i$ becomes one. Then, $r_i$ also recognizes $r_j$ as its successor, because $r_j$ is always in $\text{view}(r_i)$ and $k_i < k_j$ holds. Thus, by Lemma 8, $r_i$ can always recognize $r_j$ as its successor, and $m$-line is one way.

If $\text{view}(r_i)$ is $\text{ColP1B0}$ (resp. $\text{ColP1B1}$) at $t$, $r_i$ moves on a line except 0-line and $(L-1)$-line (resp. $(L-1)$-line) and $r_j$ also follows $r_i$. After that, $\text{view}(r_j)$ becomes $\text{MovP2B}$ or $\text{MovP2A}$ in $M'_2$ until $r_j$ arrives at the end of the line (i.e., $P2\text{StopA}$ or $P2\text{StopB}$ in $F'_2$) or $r_h$ becomes Finished on the same line (i.e., $P2\text{StopC}$ in $F'_2$). By the same discussion as above, $r_i$ can always recognize $r_j$ as its successor, and the line is one way.

Therefore, in any case, while $r_i$ has $p_2(k_i)$, if the view is in $C'$ or $M'_2$, then $r_i$ cannot become outdated as any non-Finished robots cannot come into $r_i$’s visible region, and any non-Finished robots in $r_i$’s view cannot move. Thus, $r_i$ cannot collide with other robots while $r_i$ has $p_2(k_i)$, and the lemma holds.

**Lemma 11.** Each non-Finished robot can recognize its successor.
Proof. By Lemma 9 (resp. Lemma 10), each non-Finished robot \( r_i \) can always recognize its successor while \( c(r_i) = p_1(k_i) \) (resp. \( c(r_i) = p_2(k_i) \)) holds. Thus, the lemma holds.

**Lemma 12.** Robots cannot collide when executing Algorithm 2.

Proof. By Lemma 9 (resp. Lemma 10), while the light color is \( p_1(k_i) \) (resp. \( p_2(k_i) \)), robots cannot collide. Because each robot cannot move after it becomes Finished, the lemma holds.

Next, we show that Algorithm 2 constructs a maximum independent set.

**Lemma 13.** The first robot \( r_1 \) moves to the diagonal corner, and \( c(r_1) \) becomes \( F \) on the corner.

Proof. By the proofs of Lemmas 9 and 10 while robots move on the grid, they keep the order they entered the grid.

By the proof of Lemma 9, \( r_1 \) eventually arrives at the end of the first border, and then \( r_1 \)'s successor \( r_2 \) is on the node three hops behind. When the distance between \( r_1 \) and \( r_2 \) becomes two, then \( \text{view}(r_1) \) becomes \( \text{OnCP1} \) in \( C' \).

After that, by the proof of Lemma 10, \( r_1 \) eventually arrives at the diagonal corner because \( r_1 \) is the first robot. When the distance between \( r_1 \) and \( r_2 \) becomes two, \( \text{view}(r_1) \) becomes \( \text{OnCP2} \) in \( F'_2 \). By Rule 4, because \( c(r_1) = p_2(k_1) \), it changes its color to \( F \) on the corner.

Thus, the lemma holds.

**Lemma 14.** The first \( \lceil l/2 \rceil \) robots move to the second border, and their colors become \( F \). Additionally, nodes on the second border are empty or occupied by a robot alternately from the diagonal corner.

Proof. By Lemma 13 the first robot \( r_1 \) eventually becomes Finished on the diagonal corner. Then, by the definition of \( \text{OnCP2} \) in \( F'_2 \) for \( r_1 \), the distance between \( r_1 \) and its successor \( r_2 \) is two.

Consider the execution of \( r_2 \) after \( c(r_1) \) becomes \( F \). If \( l \) is more than three, \( \text{view}(r_2) \) becomes \( \text{P2Stop} \) in \( F'_2 \) when the distance between \( r_2 \) and its successor \( r_3 \) becomes two. Then, by Rule 4, \( c(r_2) \) becomes \( F \). If \( l \) is three, then \( r_2 \) is at the end of the first border, thus \( \text{view}(r_2) \) becomes \( \text{OnCP1F} \) in \( F'_1 \) when the distance between \( r_2 \) and \( r_3 \) becomes two. Then, \( c(r_2) \) becomes \( F \) by Rule 1. Note that, in both cases, the distance between \( r_1 \) and \( r_2 \) remains two hops.

For the successors of \( r_2 \), we can discuss their movements in the same way as \( r_2 \). By the definitions of \( \text{OnCP1F} \) in \( F'_1 \) and \( \text{P2Stop} \) in \( F'_2 \), when robots become \( F \) on the second border, the distance between a robot and its successor is two hops because there is no rule to move to the adjacent node of the occupied node on the border. Therefore, on the second border, beginning with the diagonal corner, every even node is occupied, and the number of robots is \( \lfloor l/2 \rfloor \). If \( l \) is odd, when the \( \lceil l/2 \rceil \)-th robot \( r_i \) arrives at the end of the first border and the distance between \( r_i \) and its successor becomes two, \( r_i \)'s view becomes \( \text{OnCP1F} \) in \( F'_1 \) and \( r_i \) changes its color to \( F \) by Rule 1. Otherwise, \( r_i \) changes its color to \( p_2(k_i) \) and moves to the second border.
Thus, the lemma holds. \hfill \Box

Lemma 15. From the $([l/2] + 1)$-th to the $l$-th robots, each robot moves to the 1-line, and its color becomes $F$. Additionally, nodes on the 1-line are empty or occupied by a robot alternately, beginning with an empty node.

Proof. By Lemma 14, $[l/2]$ robots on 0-line eventually become Finished. By the definitions of $F'_1$ and $F'_2$, except on the Door corner, each robot can change its color to $F$ only when the distance from its successor is two.

Let $r_i$ be the $([l/2] + 1)$-th robot, $r_h$ be the $([l/2])$-th robot (i.e., $r_h$ is the predecessor of $r_i$), and $r_j$ be the $([l/2] + 2)$-th robot (i.e., $r_j$ is the successor of $r_i$). $r_i$ and $r_j$ move from the Door node in the same way as $r_h$ and its color becomes $2$. Additionally, nodes on the 1-line, except on the Door corner, each robot can change its color to $F$.

When the distance between $r_i$ and $r_j$ becomes two, one of the following two cases occurs: When the distance between $r_i$ and $r_j$ becomes two, (1) if $l$ is odd, $\text{view}(r_i)$ becomes $\text{GoCo}0$ or $\text{GoCo}1$ in $\mathcal{M}'$, because the end of the first border is occupied by $r_h$; or (2) if $l$ is even, $\text{view}(r_i)$ becomes $\text{CoP1B0}$, because the end of the first border is empty but its adjacent node on the second border is occupied by $r_h$.

In case (1), by Rule 3, $r_i$ moves to the node in front of the end of the first border. Then, after $r_j$ comes to the node two hops behind by $\text{MovP10}$, $\text{view}(r_i)$ becomes $\text{CoP1A0}$ in $C'$. Then, by Rule 2, $c(r_i)$ becomes $p_2(k_i)$ and $r_i$ moves to 1-line. After that, when $r_j$ comes to the node two hops away from $r_i$ by $\text{MovP11}$, if $l = 3$, $\text{view}(r_i)$ becomes $\text{P2StopA}$ in $F'_2$ and $r_i$ changes its color to $F$ by Rule 4. Otherwise, because $r_i$ can see Finished robots on 0-line, $\text{view}(r_i)$ becomes $\text{MovP2A}$ in $\mathcal{M}'$. Then, because the nodes on 0-line are occupied alternately, $\text{view}(r_i)$ becomes $\text{MovP2B}$ and $\text{MovP2A}$ in $\mathcal{M}_2'$ alternately by the execution of Rule 5. Thus, $r_i$ moves toward the other side border that is parallel to the first border by Rule 5 and $r_j$ follows $r_i$. Finally, $\text{view}(r_i)$ eventually becomes $\text{P2StopA}$ in $F'_2$ because the diagonal corner is occupied by a Finished robot (Lemma 13). Then, by Rule 4, $c(r_i)$ eventually becomes $F$. Because $l$ is even, $[l/2] - 1$ successors of $r_i$ follow $r_i$, and eventually their views become $\text{P2StopC}$ in $F'_2$, and they change their colors to $F$ by Rule 4 on 1-line.

In case (2), $r_i$ also changes its color to $p_2(k_i)$, and moves to 1-line by Rule 2. After that, because $r_i$ can see Finished robots on 0-line, $\text{view}(r_i)$ becomes $\text{MovP2B}$ in $\mathcal{M}_2'$. Then, in the same way as for case (1), $l/2 - 1$ robots including $r_i$ become Finished on 1-line. After that, the view of the next robot $r_l$ ($l$-th robot) becomes $\text{P1Stop1}$ in $F'_1$ on the intersection of the first border and 1-line, and $r_l$ becomes Finished by Rule 1. Thus, the lemma holds. \hfill \Box

Lemma 16. The distance between any two robots on the grid is two hops after every robot becomes Finished.

Proof. By the definitions of $F'_1$ and $F'_2$, the distance between a robot $r_i$ and its predecessor $r_j$ is two hops after each robot becomes Finished if $r_i$ and $r_j$ are on the same line. Thus, when the robots on $m$-line ($0 < m < L - 1$) become Finished, if there are two adjacent Finished robots to the contrary, then there is a robot $r_e$ on $m$-line that cannot move from the node that is adjacent to a node occupied by a Finished robot on $(m - 1)$-line. However, by the same argument as in Lemmas 14 and 15 if $m$ is odd (resp. even),
the nodes on $m$-line are occupied alternately beginning with an empty node (resp. occupied node) because the nodes on $(m-1)$-line are also occupied alternately beginning with an occupied node (resp. empty node). Thus, before such $r_r$ becomes Finished, $r_r$ has a view of type MovP2B and can move by Rule 5, i.e., such $r_r$ cannot exist.

Now, to consider the end of the execution of the algorithm, we consider $(L - 1)$-line when nodes on $(L - 2)$-line are occupied by Finished robots. The $(L - 1)$-line is a border connected to the Door corner. Then, if both $l$ and $L$ are odd or both are even, the view from the Door node becomes Door4 or Door3, otherwise Door2 in $D'$ (See Fig.10). Note that, in the case of Door3, the last robot $r_i$ on the $(L - 2)$-line becomes Finished by P2StopC in $F_2'$ when its successor $r_j$ arrives at the Door corner. Then, after $r_j$ moves two hops (i.e., by ColP1B1 in $C'$ and MovP2B in $M_2'$ respectively), the view from the Door node becomes Door4 for $r_j$’s successor.

– If the view from the Door node is Door4 or Door3, the robot on the Door node moves to the Door corner by Rule 0-2. Then, the view from the Door corner is ColP1B1 in $C'$. By the same discussion as above (Fig.10), the view from the Door corner eventually becomes P1Stop0 in $F_1'$, thus the final robot on the Door corner becomes Finished by Rule 1. Then, any other robots cannot enter into the grid because there is no such rule.

– If the view from the Door node is Door2, the view from the Door corner is ColP1A1 in $C'$. Then, the empty node $v$ that is adjacent to the Door corner is eventually occupied by a Finished robot on $(L - 1)$-line (Fig.10). After that, any other robots on the Door node cannot enter into the grid because there is no such rule.

Thus, the lemma holds. \hfill \qed

**Lemma 17.** Every robot on the grid is eventually Finished.

**Proof.** By the proofs of Lemmas 8–16, the transitions of the view type of each robot are shown as Fig.19. Thus, the lemma holds. \hfill \qed

By Lemma 16, distances between any two occupied nodes are two. Thus, by the same discussion as Theorem 7 we are now able to state our main result:

**Theorem 3.** Algorithm 2 constructs a maximum independent set by occupied locations on the grid.

By the proofs of Lemmas 14–16 nodes on the even-numbers (resp. odd-numbers) lines are occupied by $\lceil l/2 \rceil$ (resp. $\lfloor l/2 \rfloor$) robots. By the same discussion as Lemma 7 the following lemma holds.

**Lemma 18.** When a maximum independent set is constructed, $\lceil n/2 \rceil$ robots are on the grid.

Each robot $r_i$ sets its value $k_i$ at most once by Rule 0-1. By the same discussion of Theorem 2 the following theorem holds.

**Theorem 4.** The time complexity of Algorithm 2 is $O(n(L + l))$ steps.
4 Conclusion

We proposed two algorithms to construct a maximum independent set on an unknown size grid in the case that the Door node is connected to a corner node. One of our algorithms uses only three colors for each robot light and $\phi = 2$, but it assumes port numbering. The other uses seven colors for each robot light and $\phi = 3$, and it executes in a completely anonymous graph. Both of the time complexity are $O(n(L + l))$ steps.

Some interesting questions remain open:

- Are there any algorithms for the case where each robot has no light or two light colors? Following the results by Hesari et al. [23] for the continuous line setting, we conjecture their impossibility result for oblivious (a.k.a. no-light robots) can be extended to the discrete asynchronous and unoriented setting.
- Are there any algorithms for the case where the visibility range is less than two?
- Are there any algorithms for other assumptions of the Door node? For example, the Door node can be connected to another node, and there may be multiple Door nodes.
– Are there any algorithms that can tolerate maximum independent set reconfiguration in the case of robot crashes? We conjecture that, assuming a failing robot turns off its light (that is, crash failures can be detected by other robots), it is possible to extend our algorithm to adjust the remaining robots and introduce new ones so that the maximum independent set is reconstructed.

Additionally, we plan to design algorithms for the case of a maximal independent set placement, and minimum dominating set placement, that requires fewer robots.

References

1. Adhikary, R., Bose, K., Kundu, M.K., Sau, B.: Mutual visibility by asynchronous robots on infinite grid. In: ALGOSENSORS. pp. 83–101 (2018)
2. Balabonski, T., Courtieu, P., Pelle, R., Rieg, L., Tixeuil, S., Urbain, X.: Continuous vs. discrete asynchronous moves: A certified approach for mobile robots. In: NETYS. pp. 93–109 (2019)
3. Barrière, L., Flocchini, P., Barrameda, E.M., Santoro, N.: Uniform scattering of autonomous mobile robots in a grid. Int. J. Found. Comput. Sci. 22(3), 679–697 (2011)
4. Bonnet, F., Milani, A., Potop-Butucaru, M., Tixeuil, S.: Asynchronous exclusive perpetual grid exploration without sense of direction. In: OPODIS. pp. 251–265 (2011)
5. 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)
6. Bramas, Q., Devismes, S., Lafourcade, P.: Infinite grid exploration by disoriented robots. In: SIROCCO. pp. 340–344 (2019)
7. Bramas, Q., Tixeuil, S.: The random bit complexity of mobile robots scattering. Int. J. Found. Comput. Sci. 28(2), 111–134 (2017)
8. Casteigts, A., Albert, J., Chaumette, S., Nayak, A., Stojmenovic, I.: Biconnecting a network of mobile robots using virtual angular forces. Comput. Commun. 35(9), 1038–1046 (2012)
9. Clément, J., Défago, X., Potop-Butucaru, M.G., Izumi, T., Messika, S.: The cost of probabilistic agreement in oblivious robot networks. Inf. Process. Lett. 110(11), 431–438 (2010)
10. Cohen, R., Peleg, D.: Local spreading algorithms for autonomous robot systems. Theor. Comput. Sci. 399(1-2), 71–82 (2008)
11. D'Angelo, G., Stefano, G.D., Klasing, R., Navarra, A.: Gathering of robots on anonymous grids and trees without multiplicity detection. Theor. Comput. Sci. 610, 158–168 (2016)
12. Das, S., Flocchini, P., Prencipe, G., Santoro, N., Yamashita, M.: Autonomous mobile robots with lights. Theor. Comput. Sci. 609, 171–184 (2016)
13. Datta, A.K., Lamani, A., Larmore, L.L., Petit, F.: Ring exploration with oblivious myopic robots. In: SAFECOMP. pp. 335–342 (2013)
14. Devismes, S., Lamani, A., Petit, F., Raymond, P., Tixeuil, S.: Optimal grid exploration by asynchronous oblivious robots. In: SSS. pp. 64–76 (2012)
15. Devismes, S., Lamani, A., Petit, F., Raymond, P., Tixeuil, S.: Terminating exploration of a grid by an optimal number of asynchronous oblivious robots. The Computer Journal (2020)
16. Dieudonné, Y., Petit, F.: Scatter of robots. Parallel Process. Lett. 19(1), 175–184 (2009)
17. Flocchini, P.: Uniform Covering of Rings and Lines by Memoryless Mobile Sensors, pp. 2297–2301. Springer (2016)
18. Flocchini, P., Prencipe, G., Santoro, N.: Self-deployment of mobile sensors on a ring. Theor. Comput. Sci. 402(1), 67–80 (2008)
19. Flocchini, P., Prencipe, G., Santoro, N. (eds.): Distributed Computing by Mobile Entities, Current Research in Moving and Computing. Springer (2019)
20. Guilbault, S., Pelc, A.: Gathering asynchronous oblivious agents with local vision in regular bipartite graphs. Theor. Comput. Sci. 509, 86–96 (2013)
21. Hector, R., Vaidyanathan, R., Sharma, G., Trahan, J.L.: Optimal convex hull formation on a grid by asynchronous robots with lights. In: IPDPS. pp. 1051–1060 (2020)
22. Heriban, A., Défago, X., Tixeuil, S.: Optimally gathering two robots. In: ICDCN. pp. 3:1–3:10 (2018)
23. Hesari, M.E., Flocchini, P., Narayanan, L., Opatrny, J., Santoro, N.: Distributed barrier coverage with relocatable sensors. In: SIROCCO. pp. 235–249 (2014)
24. Hideg, A., Lukovszki, T.: Asynchronous filling by myopic luminous robots. Tech. rep., arXiv (2020)
25. Hsiang, T.R., Arkin, E.M., Bender, M.A., Fekete, S.P., Mitchell, J.S.B.: Algorithms for Rapidly Dispersing Robot Swarms in Unknown Environments, pp. 77–93. Springer (2004)
26. Kamei, S., Lamani, A., Ooshita, F.: Asynchronous ring gathering by oblivious robots with limited vision. In: WSSR. pp. 46–49 (2014)
27. Kamei, S., Lamani, A., Ooshita, F., Tixeuil, S., Wada, K.: Gathering on rings for myopic asynchronous robots with lights. In: OPODIS (2019)
28. Kshemkalyani, A., Molla, A.R., Sharma, G.: Dispersion of mobile robots on grids. In: WALCOM (2020)
29. Nagahama, S., Ooshita, F., Inoue, M.: Ring exploration of myopic luminous robots with visibility more than one. In: SSS. pp. 256–271 (2019)
30. Ooshita, F., Tixeuil, S.: Ring exploration with myopic luminous robots. In: SSS. pp. 301–316 (2018)
31. Poudel, P., Sharma, G.: Fast uniform scattering on a grid for asynchronous oblivious robots. In: SSS (2020)
32. Suzuki, I., Yamashita, M.: Distributed anonymous mobile robots: Formation of geometric patterns. SIAM Journal on Computing 28(4), 1347–1363 (1999)
33. Viglietta, G.: Rendezvous of two robots with visible bits. In: ALGOSENSOR. pp. 291–306 (2013)
34. Yamauchi, Y.: A survey on pattern formation of autonomous mobile robots: asynchrony, obliviousness and visibility. Journal of Physics: Conference Series 473, 012016 (2013)