Optimal Grid Exploration by Asynchronous Oblivious Robots*

Stéphane Devismes[1], Anissa Lamani[2], Franck Petit[3],
Pascal Raymond[1], and Sébastien Tixeuil[3]

[1] VERIMAG UMR 5104, Université Joseph Fourier, Grenoble, France
firstname.lastname@imag.fr
[2] MIS, Université de Picardie Jules Verne, Amiens, France
anissa.lamani@u-picardie.fr
[3] LIP6 UMR 7606, UPMC Sorbonne Universités, Paris, France
firstname.lastname@lip6.fr

Abstract. We consider *deterministic terminating exploration* of a grid by a team of asynchronous oblivious robots. We first consider the semi-synchronous atomic model ATOM. In this model, we exhibit the minimal number of robots to solve the problem *w.r.t.* the size of the grid. We then consider the asynchronous non-atomic model CORDA. ATOM being strictly stronger than CORDA, the previous bounds also hold in CORDA, and we propose deterministic algorithms in CORDA that matches these bounds. The above results show that except in two particular cases, 3 robots are necessary and sufficient to deterministically explore a grid of at least three nodes. The optimal number of robots for the two remaining cases is: 4 for the $(2, 2)$-Grid and 5 for the $(3, 3)$-Grid, respectively.

1 Introduction

We consider autonomous robots that are endowed with motion actuators and visibility sensors, but that are otherwise unable to communicate. Those robots must collaborate to solve a collective task, here the *deterministic terminating grid exploration* (*exploration* for short), despite being limited with respect to input from the environment, asymmetry, memory, etc. So far, two universes have been studied: the *continuous two-dimensional Euclidean space* and the *discrete universe*. In the former, robots freely move on a plane using visual sensors with perfect accuracy that permit to locate all other robots with infinite precision (*e.g.*, [1,2,3]). In the latter, the space is partitioned into a finite number of locations, conventionally represented by a graph, where the nodes represent the possible locations that a robot can take and the edges the possibility for a robot to move from one location to another (*e.g.*, [4,5,6,7,8,9,10]).

In this paper, we pursue research in the discrete universe and focus on the *exploration problem* when the network is an anonymous unoriented grid, using a team of autonomous mobile robots. Exploration requires that robots explore the grid and stop when the task completion. In other words, every node of the grid must be visited by at least one robot and the protocol eventually terminates.

* This work has been supported in part by the ANR projet R-Discover (08-ANR-CONTINT).

A.W. Richa and C. Scheideler (Eds.): SSS 2012, LNCS 7596, pp. 64–76, 2012.

The robots we consider are anonymous, uniform, and unable to communicate, however they can sense their environment and take decisions according to their own view. In addition, they are oblivious, *i.e.*, they do not remember their past actions.

The fact that robots have to stop after exploring all locations requires them to somehow remember at any time of the process which part of the graph has been visited yet. Nevertheless, under this weak scenario, robots have no memory and thus are unable to remember the various steps taken before. In addition, they are unable to communicate explicitly. Therefore the positions of the other robots are the only way to distinguish the different stages of the exploration process. The main complexity measure is then the minimal number of required robots. Since numerous symmetric configurations induce a large number of required robots, minimizing the number of robots turns out to be a difficult problem. As a matter of fact, in [8], it is shown that, in general, $\Omega(n)$ robots are necessary to explore a tree network of n nodes deterministically.

Related Work. In [7], authors proved that no deterministic exploration is possible on a ring when the number of robots k divides the number of nodes n. In the same paper, the authors proposed a deterministic algorithm that solves the problem using at least 17 robots provided that n and k are co-prime. In [10], Lamani *et al.* proved that there exists no deterministic protocol that can explore an even sized ring with $k \leq 4$ robots, even in the atomic model ATOM [3]. Impossibility results in ATOM naturally extend in the asynchronous non-atomic model CORDA [11]. Lamani *et al.* also provide in [10] a protocol in CORDA that allows 5 robots to deterministically explore any ring whose size is co-prime with 5. By contrast, four robots are necessary and sufficient to *probabilistically* explore of any ring of size at least 4 in ATOM [6,5].

To our knowledge, grid-shaped networks were only considered in the context of anonymous and oblivious robot exploration [4] for a variant of the exploration problem where robots perpetually explore all nodes in the grid. Also, contrary to this paper, the protocols presented in [4] make use of a common sense of direction for all robots (common north, south, east, and west directions) and assume an essentially synchronous scheduling.

Contribution. In this paper, we propose optimal (*w.r.t.* the number of robots) solutions for the deterministic terminating exploration of a grid-shaped network by a team of k asynchronous oblivious robots in the CORDA model.

In more details, we first consider the ATOM model, which is a strictly stronger model than CORDA. We show that it is impossible to explore a grid of at least three nodes with less than three robots. Next, we show that it is impossible to explore a $(2, 2)$-Grid with less than 4 robots, and a $(3, 3)$-Grid with less than 5 robots, respectively. The two first results hold for both deterministic and probabilistic explorations, while the latter holds only in the deterministic case. Note also that these impossibility results naturally extend to CORDA.

Then, we propose several deterministic algorithms in CORDA to exhibit the optimal number of robots allowing to explore of a given grid. Our results show that except in two particular cases, 3 robots are necessary and sufficient to deterministically explore a grid of at least three nodes. The optimal number of robots for the two remaining cases is: 4 for the $(2, 2)$-Grid and 5 for the $(3, 3)$-Grid, respectively.

The above results show that, perhaps surprisingly, exploring a grid is easier than exploring a ring. In the ring, deterministic solutions essentially require five robots [10] while probabilities enable solutions with only four robots [6,5]. In the grid, three robots are necessary and sufficient in all but two cases even for deterministic protocols, the two latter cases do require four or five robots. Also, deterministically exploring a grid requires no primality condition while deterministically exploring a ring expects the number k of robots to be co-prime with n, the number of nodes.

Roadmap. Section 2 presents the system model and the problem to be solved. Lower bounds are shown in Section 3. The deterministic general solution using three robots is given in Section 4. (Note that exploring a $(2, 2)$-Grid using 4 robots is trivially possible, henceforth not considered in this paper.) Section 5 gives some concluding remarks. Due to the lack of space, the special case with five robots is omitted, see the technical report [12] for details.

2 Preliminaries

Distributed Systems. We consider systems of autonomous mobile entities called *agents* or *robots* evolving in a *simple unoriented connected graph* $G = (V, E)$, where V is a finite set of nodes and E a finite set of edges. In G, nodes represent locations that can be sensed by robots and edges represent the possibility for a robot to move from one location to another. We assume that G is an (i, j)-*Grid* (or a Grid, for short) where i, j are two positive integers, *i.e.*, G satisfies the following two conditions: (i) $|V| = i \times j$ and (ii) there exists an order on the nodes of V, $v_1, \ldots, v_{i \cdot j}$, such that $\forall x \in [1..i \times j]$, $(x \bmod i) \neq 0 \Rightarrow \{v_x, v_{x+1}\} \in E$, and $\forall y \in [1..i \times (j-1)]$, $\{v_y, v_{y+i}\} \in E$.

We denote by $n = i \times j$ the number of nodes in G. We denote by $\delta(v)$ the degree of node v in G. Nodes of the grid are anonymous. (We may use indices, but for notation purposes only.) Moreover, given two neighboring nodes u and v, there is no explicit or implicit labelling allowing the robots to determine whether u is either on the left, on the right, above, or below v. Remark that an (i, j)-*Grid* and a (j, i)-*Grid* are isomorphic. Hence, as the nodes are anonymous, we cannot distinguish an (i, j)-*Grid* from a (j, i)-*Grid*. So, without loss of generality, we always consider (i, j)-*Grids*, where $i \leq j$. Note also that any $(1, j)$-*Grid* is isomorphic to a chain. In any (i, j)-*Grid*, if $i = 1$, then either the grid consists of one single node, or two nodes are of degree 1 and all other nodes are of degree 2; otherwise, when $i > 1$, four nodes are of degree 2 and all other nodes are of degree either 3 or 4. In any grid, the nodes of smallest degree are called *corners*. In any $(1, j)$-*Grid* with $j > 1$, the unique chain linking the two corners is called the *borderline*. In any (i, j)-*Grid* such that $i > 1$, there exist four chains $v_1, \ldots, v_m$ of length at least 2 such that $\delta(v_1) = \delta(v_m) = 2$, and $\forall x, 1 < x < m$, $\delta(v_x) = 3$, these chains are also called the *borderlines*.

Robots and Computation. Operating on G are $k \leq n$ robots. The robots do not communicate in an explicit way; however they see the position of the other robots and can acquire knowledge based on this information. We assume that the robots cannot remember any previous observation nor computation performed in any previous step. Such robots are said to be *oblivious* (or *memoryless*).

Each robot operates according to its (local) *program*. We call *protocol* a collection of k *programs*, each one operating on one single robot. Here we assume that robots are *uniform* and *anonymous*, *i.e.*, they all have the same program using no local parameter (such as an identity) that could permit to differentiate them. The program of a robot consists in executing *Look-Compute-Move* (*LCM*) *cycles* infinitely many times. That is, the robot first observes its environment (Look phase). Then, based on its observation and according its program, the robot then decides to move or stay idle (Compute phase). When the robot decides to move, it moves from its current node to a neighboring node during the Move phase.

We consider two models: the semi-synchronous and atomic model called ATOM [3], and the asynchronous non-atomic model called CORDA [11]. In both models, time is represented by an infinite sequence of instants 0, 1, 2, ... No robot has access to this global time. Moreover, every robot executes cycles infinitely many times. Each robot performs its own cycles in sequence. However, the time between two cycles of the same robot and the interleavings between cycles of different robots are decided by an *adversary*. We are interested in algorithms that correctly operate despite the choices of the adversary. In particular, our algorithms should also work even if the adversary forces the execution to be fully sequential or fully synchronous. In ATOM, each LCM cycle execution is assumed to be *atomic*: every robot that is activated (by the adversary) at instant t instantaneously executes a full cycle between t and $t+1$. In CORDA, LCM cycles are performed asynchronously by each robot: the time between Look, Compute, and Move operations is finite yet unbounded, and is decided by the adversary. The only constraint is that both Move and Look are instantaneous.

Note that in both models, any robot performing a Look operation sees all other robots on nodes and not on edges. However, in CORDA, a robot $\mathcal{R}$ may perform a Look operation at some time t, perceiving robots at some nodes, then Compute a target neighbor at some time $t' > t$, and Move to this neighbor at some later time $t'' > t'$ in which some robots are at different nodes from those previously perceived by $\mathcal{R}$ because in the meantime they moved. Hence, in CORDA robots may move based on significantly outdated perceptions. Of course, ATOM is stronger than CORDA. So, to be as general as possible, in this paper, our impossibility results are written assuming ATOM, while our algorithms assume CORDA.

Multiplicity. We assume that during the Look phase, every robot can perceive whether several robots are located on the same node or not. This ability is called *Multiplicity Detection*. We shall indicate by $d_i(t)$ the multiplicity of robots present in node u_i at instant t. We consider two kinds of multiplicity detection: the *strong* and *weak* multiplicity detections. Under the *weak* multiplicity detection, for every node u_i, d_i is a function $\mathbb{N} \mapsto \{\circ, \bot, \top\}$ defined as follows: $d_i(t)$ is equal to either $\circ$, $\bot$, or $\top$ according to u_i contains none, one or several robots at time instant t. If $d_i(t) = \circ$, then we say that u_i is *free* at instant t, otherwise u_i is said *occupied* at instant t. If $d_i(t) = \top$, then we say that u_i contains a *tower* at instant t. Under the *strong* multiplicity detection, for every node u_i, d_i is a function $\mathbb{N} \mapsto \mathbb{N}$ where $d_i(t) = j$ indicates that there are j robots in node u_i at instant t. If $d_i(t) = 0$, then we say that u_i is *free* at instant t, otherwise u_i is said *occupied* at instant t. If $d_i(t) > 1$, then we say that u_i contains a *tower (of $d_i(t)$ robots)* at instant t.

As previously, to be as general as possible, our impossibility results are written assuming the strong multiplicity detection, while our algorithms assume the weak multiplicity detection.

Configurations, Views and Execution. To define the notion of *configuration*, we need to use an arbitrary order $\prec$ on nodes. The system being anonymous, robots do not know this order. Let $v_1, \ldots, v_n$ be the list of the nodes in G ordered by $\prec$. The configuration at time t is $d_1(t), \ldots, d_n(t)$. We denote by *initial configurations* the configurations from which the system can start at time 0. Every configuration where all robots stay idle forever is said to be *terminal*. Two configurations $d_1, \ldots, d_n$ and $d'_1, \ldots, d'_n$ are *indistinguishable* (*distinguishable* otherwise) if and only if there exists an automorphism f on G satisfying the additional condition: $\forall v_i \in V$, we have $d_i = d'_j$ where $v_j = f(v_i)$.

The *view* of robot $\mathcal{R}$ at time t is a labelled graph isomorphic to G, where every node u_i is labelled by $d_i(t)$, except the node where $\mathcal{R}$ is currently located, this latter node u_j is labelled by $d_j(t), *$. (Indeed, any robot knows the multiplicity of the node where it is located.) Hence, from its view, a robot can compute the view of each other robot, and decide whether some other robots have the same view as its own.

Every decision to move is based on the view obtained during the last Look action. However, it may happen that some edges incident to a node v currently occupied by the deciding robot look identical in its view, *i.e.*, v lies on a symmetric axis of the configuration. In this case, if the robot decides to take one of these edges, it may take any of them. We assume the worst-case decision in such cases, *i.e.* the actual edge among the identically looking ones is chosen by the adversary.

We model the executions of our protocol in G by the list of configurations through which the system goes. So, an *execution* is a maximal list of configurations $\gamma_0, \ldots, \gamma_i$ such that $\forall j > 0$, we have: (i) $\gamma_{j-1} \neq \gamma_j$, (ii) γ_j is obtained from γ_{j-1} after some robots move from their locations in γ_{j-1} to a neighboring node, and (iii) For every robot $\mathcal{R}$ that moves between γ_{j-1} and γ_j, there exists $0 \leq j' \leq j$, such that $\mathcal{R}$ takes its decision to move according to its program and its view in $\gamma_{j'}$. An execution $\gamma_0, \ldots, \gamma_i$ is said to be *sequential* if and only if $\forall j > 0$, exactly one robot moves between γ_{j-1} and γ_j.

Exploration. We consider the *exploration* problem, where k robots, initially placed at different nodes, collectively explore an (i, j)-grid before stopping moving forever. By "collectively" explore we mean that every node is eventually visited by at least one robot. More formally, a protocol $\mathcal{P}$ *deterministically* (resp. *probabilistically*) solves the exploration problem if and only if every execution e of $\mathcal{P}$ starting from a *towerless* configuration[1] satisfies: (1) e terminates *in finite time* (resp. *with probability 1*), and (2) every node is visited by at least one robot during e.

Observe that the exploration problem is not defined for $k > n$ and is straightforward for $k = n$. (In this latter case the exploration is already accomplished in the initial towerless configuration.)

[1] The initial configuration must be towerless to make the exploration solvable in our model.

3 Bounds

In this section, we first show that, except for trivial case where $k = n$, if (i) robots are *oblivious*, (ii) the model is *ATOM*, and (iii) the multiplicity is *strong* (*i.e.,* the strongest possible assumptions), at least three robots are necessary to solve the (probabilistic or deterministic) exploration of any grid (Theorem 2). Moreover, in a $(2, 2)$-Grid, four robots are necessary (Theorem 3). Finally, at least five robots are necessary to solve the deterministic exploration of a $(3, 3)$-Grid (Theorem 4). In the two next sections, we show that all these bounds are also sufficient to solve the deterministic exploration in the asynchronous and non-atomic CORDA model.

Given that robots are oblivious, if there are more nodes than robots, then any terminal configuration should be distinguishable from any possible initial (towerless) configuration. So, we have:

Remark 1. *Any terminal configuration of any (probabilistic or deterministic) exploration protocol for a grid of* n *nodes using* $k < n$ *oblivious robots contains at least one tower.*

Theorem 2. *There exists no (probabilistic or deterministic) exploration protocol in ATOM using* $k \leq 2$ *oblivious robots for any* (i, j)*-Grid made of at least* 3 *nodes.*

Proof. By Remark 1, there is no protocol allowing one robot to explore any (i, j)-Grid made of at least 2 nodes. Indeed, any configuration is towerless in this case. Assume by contradiction, that there exists a protocol $\mathcal{P}$ in ATOM to explore with 2 oblivious robots an (i, j)-Grid made of at least 3 nodes. Consider a sequential execution e of $\mathcal{P}$ that terminates. (By definition, if we consider a deterministic exploration, then all executions should terminate; while if we consider a probabilistic exploration, at least one of the sequential execution should terminate.) Then, e starts from a towerless configuration (by definition) and eventually reaches a terminal configuration containing a tower (by Remark 1). As e is sequential, the two last configurations of e consist of a towerless configuration followed by a configuration containing one tower. These two configurations form a possible sequential execution that terminates while only two nodes are visited, thus a contradiction. □

Any $(2, 2)$-Grid is isomorphic to a 4-size ring. It is shown in [6] that no (probabilistic or deterministic) exploration using less than four oblivious robots is possible for any ring of size at least four in ATOM. So:

Theorem 3 ([6]). *There exists no (probabilistic or deterministic) exploration protocol using* $k \leq 3$ *oblivious robots in ATOM for a* $(2, 2)$*-Grid.*

Theorem 4. *There exists no deterministic exploration protocol in ATOM using* $k \leq 4$ *oblivious robots for a* $(3, 3)$*-Grid.*

Proof Outline. From Theorem 2, k must be greater or equal to 3. Consider first the case of $k = 3$ robots and, assume for the sake of contradiction, that there exists a deterministic protocol $\mathcal{P}$ in ATOM that uses 3 robots to explore a $(3, 3)$-Grid. Then, we can show the following claims:

1. There exist sequential executions of $\mathcal{P}$, $e = \gamma_0, \ldots, \gamma_w$, in which: (a) for every x, y with $0 \leq x < y$, γ_x and γ_y are distinguishable, and (b) only the first configuration γ_0 is towerless.
2. If there exists an execution of $\mathcal{P}$, $e = \gamma_0 \ldots \gamma_x \ldots$, where γ_x contains a tower of 3 robots, then there exists an execution e' starting with the prefix $e = \gamma_0 \ldots \gamma_x$ such that at most one new node can be visited after γ_x.
3. In any suffix $\gamma_w, \ldots, \gamma_z$ of any sequential execution of $\mathcal{P}$ where (a) for every x, y with $0 \leq x < y$, γ_x and γ_y are distinguishable, and (b) γ_w contains a tower of 2 robots, then at most 4 new nodes can be visited from γ_w before a robot of the tower moves.

Using these three claims, we can show that there exist some executions of $\mathcal{P}$ that terminate while at least one node has not been visited, a contradiction.

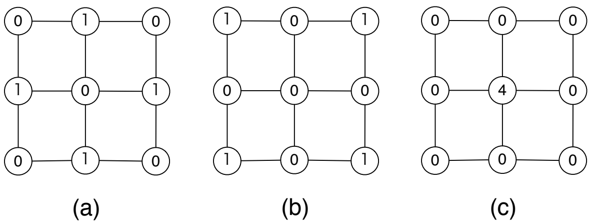

Fig. 1. Three possible configurations in a $(3, 3)$-Grid with 4 robots. Numbers inside the circles represent the multiplicity of the node.

Consider now the case of four robots. The proof consists in showing that, starting from particular configurations, the adversary can always maintain symmetries. To see this, refer to Figure 1 that depicts three possible configurations for a $(3, 3)$-Grid with 4 robots — numbers inside the circles represent the multiplicity of the node. Note that both Configuration (a) and (b) can be initial configurations. By activating the four robots synchronously and starting from Configuration (a), the adversary may lead the system in either Configuration (b) or Configuration (c). Then, in both cases, the adversary may prevent the termination of the exploration, no matter the protocol is. □

4 Deterministic Solution Using Three Robots

In this section, we focus on the deterministic exploration of a grid by three robots, in CORDA, and assuming weak multiplicity detection. Recall that there exists no deterministic solution for the exploration using three robots in a $(2, 2)$- or $(3, 3)$-grid assuming that model (Section 3). Moreover, exploring a $(1, 3)$-grid using three robots is straightforward. So, we consider all remaining cases. We split our study in two cases. An overview of the deterministic solution for any (i, j)-grid such that $j > 3$ is given in Subsection 4.1. The particular case of the $(2, 3)$-grid is solved in Subsection 4.2.

4.1 Main Algorithm

Overview. Our algorithm works according to the following three phases:

- `Set-Up`. The aim of this phase is to reach a configuration, called `Set-Up` configuration, where there is a single line of robots starting at a corner and along one of the longest borderlines of the grid—refer to Figure 2. The phase is initiated from any towerless configuration that is not a `Set-Up` configuration. Note that no tower is created during this phase. Details about this phase are given in the next subsection.
- `Orientation`. This phase follows the `Set-Up` phase and consists of a single move where the robot which is at the corner move to its adjacent occupied node. Once it has moved, a tower is created. The resulting configuration is called an `Oriented` configuration in which, the robots can agree on a common coordinate system as show in Figure 3. The node with coordinates $(0, 0)$ is the unique corner that is the closest to the tower. The x-axis is given by the vector linking the node $(0, 0)$ to the node where the tower is located. The y-axis is given by the vector linking the node $(0, 0)$ to its neighboring node that does not contain the tower.
- `Exploration`. This phase starts from an `Oriented` configuration. Note that in nodes of coordinates $(0, 0)$, $(0, 1)$, and $(0, 2)$ have been visited. So, the goal is to visit all the other nodes. To ensure that the exploration phase remains distinct from the previous phases and keep the coordinate system, we only authorize the robot that does not belong to the tower to move. This robot is called the *explorer*.
  To explore all remaining nodes, the explorer should order all coordinates in such a way that (a) $(0, 0)$ and $(0, 1)$ are before its initial position (that is $(0, 2)$) and all other coordinates are after; and (b) for all non-maximum coordinates (x, y), if (x', y') are successor of (x, y) in the order, then the nodes of coordinates (x, y) and (x', y') are neighbors. An example of such an order is $\preceq$, defined as follows: $(x, y) \preceq (x', y')$ if and only if $y < y' \vee [y = y' \wedge (x = x' \vee y \bmod 2 = 0 \wedge x < x' \vee y \bmod 2 = 1 \wedge x > x')]$.
  Using $\preceq$, the explorer moves as follows: While the explorer is not located at the node having the maximum coordinates according to $\preceq$, the explorer moves to the neighboring node whose coordinates are successors of the coordinates of its current position, as described in Figure 4.

The Set-Up Phase. In the following, we denote by $Dist(\mathcal{R}, \mathcal{R}')$ the *distance* (*i.e.*, the length of the shortest path) between the two nodes of the grid where $\mathcal{R}$ and $\mathcal{R}'$ are respectively located.

We now present the behavior of the three robots, respectively referred to as $\mathcal{R}1, \mathcal{R}2$, and $\mathcal{R}3$,[2] according to three main kinds of configurations: `Leader`, `Choice`, and `Undefined`. These classes will be split into several sub-classes.

I) The configuration is of type `Leader`: Any towerless configuration where there is exactly one robot that is at a corner of the grid. Let $\mathcal{R}1$ be this robot.
   Let consider the following subcases:

[2] Recall that robots are anonymous, so these notations are only used to ease the explanations.

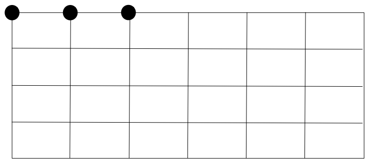

Fig. 2. `Set-Up` Configuration

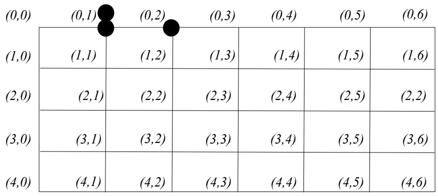

Fig. 3. Coordinate system built by the `Orientation` phase

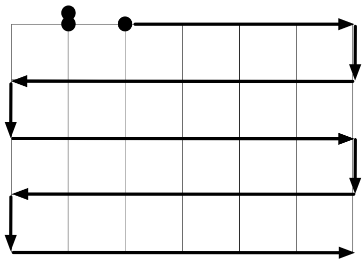

Fig. 4. Exploration phase

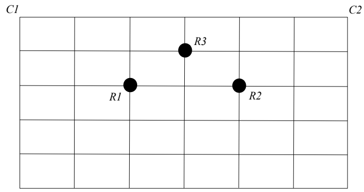

Fig. 5. Sample of a configuration of type `Undefined4-4`

A) The configuration is of type `Strict-Leader`: In such a configuration, there is no other robot on any borderline having the corner where $\mathcal{R}1$ is located as extremity. In this case, the robots that are the closest to $\mathcal{R}1$ are the ones allowed to move. Their destination is their adjacent free node on a shortest path towards the closest free node that is on a longest borderline having the corner where $\mathcal{R}1$ is located as extremity. (If there is several shortest paths, the adversary makes the choice.)

B) The configuration is of type `Half-Leader`: In such a configuration, among $\mathcal{R}2$ and $\mathcal{R}3$, only one robot, say $\mathcal{R}2$, is on a borderline having the corner where $\mathcal{R}1$ is located as extremity. Two subcases are possible:

– The configuration is of type `Half-Leader1`: $\mathcal{R}2$ is on a longest borderline. In this case, the third robot $\mathcal{R}3$ is the one allowed to move. Its destination is an adjacent free node towards a closest free node on the borderline that contains both $\mathcal{R}1$ and $\mathcal{R}2$. (If there is several shortest paths, the adversary makes the choice.)
– The configuration is of type `Half-Leader2`: $\mathcal{R}2$ is not on the longest borderline. In this case, $\mathcal{R}2$ is the one allowed to move, its destination is the adjacent free node outside the borderline, if any. In the case where there is no such a free node, $\mathcal{R}2$ moves to a free node on its own borderline. (In case of symmetry, the adversary makes the choice.)

C) The configuration is of type `All-Leader`: All the robots are on a borderline having the corner where $\mathcal{R}1$ is located as extremity. In this case, $\mathcal{R}2$ and $\mathcal{R}3$ are not necessarily on the same borderline. So, we have two subcases:

- The configuration is of type `Fully-Leader`: In such a configuration, all the robots are on the same borderline, $D1$. The two following subcases are then possible:

  (1) The configuration is of type `Fully-Leader1`: In this case, $D1$ is a longest borderline. If the robots form a line, then the `Set-Up` configuration is reached and the phase is done. Otherwise, let $\mathcal{R}2$ be the closest robot from $\mathcal{R}1$. If $\mathcal{R}1$ and $\mathcal{R}2$ are not neighbors, then $\mathcal{R}2$ is the only one allowed to move and its destination is the adjacent free node towards $\mathcal{R}1$. In the other case, $\mathcal{R}3$ is the only robot allowed to move and its destination is the adjacent free node towards $\mathcal{R}2$.
  (2) The configuration is of type `Fully-Leader2`: In this case, $D1$ is not a longest borderline. Then, the robot among $\mathcal{R}2$ and $\mathcal{R}3$ that is the closest to $\mathcal{R}1$ leaves the borderline by moving to its neighboring free node outside the borderline.
- The configuration is of type `Semi-Leader`: $\mathcal{R}2$ and $\mathcal{R}3$ are not on the same borderline. Two subcases are possible:

  (1) The configuration is of type `Semi-Leader1`: In this case, $i \neq j$. The unique robot among $\mathcal{R}2$ and $\mathcal{R}3$ which is located on a smallest borderline moves to the adjacent free node outside its borderline.
  (2) The configuration is of type `Semi-Leader2`: In this case, $i = j$. Let denote by $Dist(\mathcal{R}, \mathcal{R}')$ the *distance* (that is, the length of the shortest path) in the grid between the two nodes where $\mathcal{R}$ and $\mathcal{R}'$ are respectively located. If $Dist(\mathcal{R}1, \mathcal{R}2) \neq Dist(\mathcal{R}1, \mathcal{R}3)$, then the robot among $\mathcal{R}2$ and $\mathcal{R}3$ that is the closest to $\mathcal{R}1$ is the only one allowed to move, its destination is the adjacent free node outside the borderline. Otherwise ($Dist(\mathcal{R}1, \mathcal{R}2) = Dist(\mathcal{R}1, \mathcal{R}3)$), either (a) there is a free node between $\mathcal{R}1$ and $\mathcal{R}2$, or (b) $\mathcal{R}1$ is both neighbor of $\mathcal{R}2$ and $\mathcal{R}3$. In case (a), $\mathcal{R}1$ is the only robot allowed to move and its destination is an adjacent free node towards one of its two borderlines. (The adversary makes the choice.) In case (b), $\mathcal{R}2$ and $\mathcal{R}3$ move and their destination is their adjacent free node on their borderline.

II) The configuration is of type `Choice`: Any towerless configuration, where at least two robots are located at a corner.
We consider two cases:
A) The configuration is of type `Choice1`: In this configuration, there are exactly two robots that are located at a corner of the grid. Let $\mathcal{R}1$ and $\mathcal{R}2$ be these robots.

- In the case where $\mathcal{R}3$ is on the same borderline as either $\mathcal{R}1$ or $\mathcal{R}2$ but not both — suppose $\mathcal{R}1$ — then $\mathcal{R}2$ is the one allowed to move, its destination is the adjacent free node towards the closest free node of the borderline that contains both $\mathcal{R}1$ and $\mathcal{R}3$.
- In the case where the three robots are on the same borderline. Then:
  (1) If $Dist(\mathcal{R}1, \mathcal{R}3) \neq Dist(\mathcal{R}2, \mathcal{R}3)$, then the robot among $\mathcal{R}1$ and $\mathcal{R}2$ that is farthest to $\mathcal{R}3$ moves to the adjacent free node on the borderline towards $\mathcal{R}3$.
  (2) Otherwise ($Dist(\mathcal{R}1, \mathcal{R}3) = Dist(\mathcal{R}2, \mathcal{R}3)$), and $\mathcal{R}3$ has either or not an adjacent free node on the borderline. In the former case, $\mathcal{R}3$ moves to an adjacent free node on the borderline towards either $\mathcal{R}1$ or $\mathcal{R}2$. (The adversary makes the choice.) In the latter case, $\mathcal{R}3$ moves to its adjacent free node outside the borderline.

- If $\mathcal{R}3$ is not on any borderline, it moves to an adjacent free node on a shortest path towards the closest free node that is on a longest borderline that contains either $\mathcal{R}1$ or $\mathcal{R}2$. (In case of symmetry, the adversary makes the choice.)

B) The configuration is of type `Choice2`: In this configuration, all the robots are located at a corner. The robot allowed to move is the one that is located at a node that is common to the two borderlines of the other robots. Let $\mathcal{R}1$ be this robot. The destination of $\mathcal{R}1$ is the adjacent free node on a longest borderline. (In case of symmetry, the adversary makes the choice.)

III) The configuration is of type `Undefined`: Any towerless configuration where there is no robot that is located at any corner.
The cases below are then possible:
A) The configuration is of type `Undefined1`: In this case, $i = j$ and there is one borderline that contains two robots $\mathcal{R}1$ and $\mathcal{R}2$ such that $\mathcal{R}1$ is closer from a corner than $\mathcal{R}2$ and $\mathcal{R}3$. Let $D1$ be this borderline. $\mathcal{R}3$ is the only one allowed to move and its destination is an adjacent free node on a shortest path towards the closest free node of $D1$. (If there are several shortest paths, the adversary makes the choice.)
B) The configuration is of type `Undefined2`: It is any configuration different from `Undefined1`, where there is exactly one robot that is the closest to a corner. In this case, this robot is the only one allowed to move, its destination is an adjacent free node on a shortest path to a closest corner. (If there are several possibilities, the adversary makes the choice.)
C) The configuration is of type `Undefined3`: There are exactly two robots that are closest to a corner. Let $\mathcal{R}1$ and $\mathcal{R}2$ be these two robots.

- If $Dist(\mathcal{R}1, \mathcal{R}3) = Dist(\mathcal{R}2, \mathcal{R}3)$, then $\mathcal{R}3$ is the only one allowed to move, and either $Dist(\mathcal{R}1, \mathcal{R}3) = 1$ or $Dist(\mathcal{R}1, \mathcal{R}3) > 1$. In the former case, $\mathcal{R}3$ moves to an adjacent free node. (If there are two possibilities, the adversary makes the choice.) In the latter case, $\mathcal{R}3$ moves to an adjacent free node from which its distance to $\mathcal{R}1$ will be different from its distance to $\mathcal{R}2$. (There will be two possibilities and the adversary will make a choice.)
- If $Dist(\mathcal{R}1, \mathcal{R}3) \neq Dist(\mathcal{R}2, \mathcal{R}3)$, then the robot among $\mathcal{R}1$ and $\mathcal{R}2$ that is closest to $\mathcal{R}3$ is the only one allowed to move. Its destination is the adjacent free node that is on a shortest path to a closest corner. (If there are several possibilities, the adversary makes the choice.)

D) The configuration is of type `Undefined4`: There are three robots that are closest to a corner. Again, four cases are possible:

- The configuration is of type `Undefined4-1`: There is exactly one robot that is on a borderline. In this case, this robot is the only one allowed to move. Its destination is an adjacent free node that is on a shortest path to a closest corner. (In case of two shortest paths, the adversary breaks the symmetry in the first step.)
- The configuration is of type `Undefined4-2`: In such a configuration, there are exactly two robots on a borderline. Let $\mathcal{R}1$ and $\mathcal{R}2$ be these two robots. The robot allowed to move is $\mathcal{R}3$. Its destination is the adjacent free node towards a closest corner. (The adversary may have to break the symmetry.)
- The configuration is of type `Undefined4-3`: The three robots are on borderlines of the grid.

(1) There are more than one robot on the same borderline: In this case, there are exactly two robots on the same borderline, and let $\mathcal{R}1$ and $\mathcal{R}2$ be these robots. Then $\mathcal{R}3$ is the only one allowed to move and its destination is an adjacent free node towards a closest corner. (The adversary may have to break the symmetry.)
(2) There is at most one robot on each borderline: Exactly one borderline is perpendicular to the two others. Only the robot on that borderline moves and its destination is the adjacent node towards a closest corner. (The adversary may have to break the symmetry.)

– The configuration is of type `Undefined4-4`: In this case, there is no robot on any borderline.
(1) In the case where there are two robots, $\mathcal{R}1$ and $\mathcal{R}2$, that are closest to the same corner, and this corner is not a closest corner to $\mathcal{R}3$, then $\mathcal{R}3$ is the only robot allowed to move and its destination is an adjacent free node on a shortest path towards a closest corner. (If there are several possibilities, the adversary makes the choice.)
(2) In the case where there are two robots, $\mathcal{R}1$ and $\mathcal{R}2$, that are closest to corners $C1$ and $C2$, respectively, where $C1 \neq C2$, and $\mathcal{R}3$ is closest to both $C1$ and $C2$, then $\mathcal{R}3$ is the only one allowed to move (see Figure 5), and it moves toward $C1$ or $C2$ according to a choice of the adversary.
(3) In the case where all the robots are closest to different corners, there is one robot $\mathcal{R}1$ whom corner is between the two corners targeted by $\mathcal{R}2$ and $\mathcal{R}3$. The robot allowed to move is $\mathcal{R}1$, its destination is an adjacent free node on a shortest path towards its closest corner. (If there are several shortest paths, the adversary makes the choice.)

The next theorem can be proven using the state diagram of the algorithm:

Theorem 5. *The three phases* `Set-Up`, `Orientation`, *and* `Exploration` *deterministically solve the exploration problem with 3 oblivious robots in CORDA for any* (i, j)*-Grid such that* $j > 3$.

4.2 Exploring a (2,3)-Grid

The idea for the $(2, 3)$-Grid is rather simple. Consider the two longest borderlines of the grid. Since there are initially three isolated robots on the grid, there exists one of the two longest borderlines, say D, that contains either all the robots or exactly two robots. In the second case, the robot that is not part of D moves to the adjacent free node on the shortest path towards the free node of D. Thus, the three robots are eventually located on D. Next, the robot not located at any corner moves to one of its two neighboring occupied nodes. (The destination is chosen by the adversary.) Thus, a tower is created. Once the tower is created, the grid is oriented. Then, the single robot moves to the adjacent free node in the longest borderline that does not contain any tower. Next, it explores the nodes of this line by moving towards the tower. When it becomes neighbor of the tower, all the nodes of the $(2, 3)$-Grid have been explored.

Theorem 6. *The deterministic exploration of a* $(2, 3)$*-Grid can be solved in CORDA using 3 oblivious robots.*

5 Conclusion

We presented necessary and sufficient conditions to explore a grid with a team of k asynchronous oblivious robots. Our results show that, perhaps surprisingly, exploring a grid is easier than exploring a ring. In the ring, deterministic (respectively, probabilistic) solutions essentially require five (resp., four) robots. In the grid, three robots are necessary (even in the probabilistic case) and sufficient (even in the deterministic case) in the all but two cases, while the two remaining instances do require four and five robots, respectively. Note that the general algorithm given in that paper requires exactly three robots. It is worth investigating whether exploration of a grid of n nodes can be achieved using any number k ($3 > k \geq n - 1$) of robots, in particular when k is even.

References

1. Das, S., Flocchini, P., Santoro, N., Yamashita, M.: On the computational power of oblivious robots: forming a series of geometric patterns. In: Proceedings of the ACM Symposium on Principles of Distributed Computing (PODC 2010), pp. 267–276 (2010)
2. Dieudonné, Y., Labbani-Igbida, O., Petit, F.: Circle formation of weak robots. ACM Transactions on Adaptive and Autonomous Systems (TAAS) 3(4), 16:1–16:20 (2008)
3. Suzuki, I., Yamashita, M.: Distributed anonymous mobile robots: Formation of geometric patterns. SIAM J. Comput. 28(4), 1347–1363 (1999)
4. Baldoni, R., Bonnet, F., Milani, A., Raynal, M.: On the Solvability of Anonymous Partial Grids Exploration by Mobile Robots. In: Baker, T.P., Bui, A., Tixeuil, S. (eds.) OPODIS 2008. LNCS, vol. 5401, pp. 428–445. Springer, Heidelberg (2008)
5. Devismes, S.: Optimal exploration of small rings. In: Proceedings of the Third International Workshop on Reliability, Availability, and Security, WRAS 2010, pp. 9:1–9:6. ACM, New York (2010)
6. Devismes, S., Petit, F., Tixeuil, S.: Optimal Probabilistic Ring Exploration by Semi-synchronous Oblivious Robots. In: Kutten, S., Žerovnik, J. (eds.) SIROCCO 2009. LNCS, vol. 5869, pp. 195–208. Springer, Heidelberg (2010)
7. Flocchini, P., Ilcinkas, D., Pelc, A., Santoro, N.: Computing Without Communicating: Ring Exploration by Asynchronous Oblivious Robots. In: Tovar, E., Tsigas, P., Fouchal, H. (eds.) OPODIS 2007. LNCS, vol. 4878, pp. 105–118. Springer, Heidelberg (2007)
8. Flocchini, P., Ilcinkas, D., Pelc, A., Santoro, N.: Remembering without Memory: Tree Exploration by Asynchronous Oblivious Robots. In: Shvartsman, A.A., Felber, P. (eds.) SIROCCO 2008. LNCS, vol. 5058, pp. 33–47. Springer, Heidelberg (2008)
9. Klasing, R., Markou, E., Pelc, A.: Gathering asynchronous oblivious mobile robots in a ring. Theor. Comput. Sci. 390(1), 27–39 (2008)
10. Lamani, A., Potop-Butucaru, M.G., Tixeuil, S.: Optimal Deterministic Ring Exploration with Oblivious Asynchronous Robots. In: Patt-Shamir, B., Ekim, T. (eds.) SIROCCO 2010. LNCS, vol. 6058, pp. 183–196. Springer, Heidelberg (2010)
11. Prencipe, G.: $InstantaneousActions$ vs. $FullAsynchronicity$: Controlling and Coordinating a Set of Autonomous Mobile Robots. In: Restivo, A., Ronchi Della Rocca, S., Roversi, L. (eds.) ICTCS 2001. LNCS, vol. 2202, pp. 154–171. Springer, Heidelberg (2001)
12. Devismes, S., Lamani, A., Petit, F., Raymond, P., Tixeuil, S.: Optimal grid exploration by asynchronous oblivious robots. Technical report, UPMC (2011), `http://arxiv.org/abs/1105.2461`