Computing Without Communicating: Ring Exploration by Asynchronous Oblivious Robots

Paola Flocchini, David Ilcinkas, Andrzej Pelc, Nicola Santoro

DOI : 10.1007/978-3-540-77096-1

Computing Without Communicating: Ring Exploration by Asynchronous Oblivious Robots

Paola Flocchini[1], David Ilcinkas[2], Andrzej Pelc[3], and Nicola Santoro[4]

[1] SITE, University of Ottawa, Canada
flocchin@site.uottawa.ca
[2] CNRS, LaBRI, Université Bordeaux I, France
david.ilcinkas@labri.fr
[3] Département d'informatique, Université du Québec en Outaouais, Canada
pelc@uqo.ca
[4] School of Computer Science, Carleton University, Canada
santoro@scs.carleton.ca

Abstract. We consider the problem of exploring an anonymous unoriented ring by a team of k identical, oblivious, asynchronous mobile robots that can view the environment but cannot communicate. This weak scenario is standard when the spatial universe in which the robots operate is the two-dimentional plane, but (with one exception) has not been investigated before. We indeed show that, although the lack of these capabilities renders the problems considerably more difficult, ring exploration is still possible.
We show that the minimum number $\rho(n)$ of robots that can explore a ring of size n is $O(\log n)$ and that $\rho(n) = \Omega(\log n)$ for arbitrarily large n. On one hand we give an algorithm that explores the ring starting from any initial configuration, provided that n and k are co-prime, and we show that there always exist such k in $O(\log n)$. On the other hand we show that $\Omega(\log n)$ agents are necessary for arbitrarily large n. Notice that, when k and n are not co-prime, the problem is sometimes unsolvable (i.e., there are initial configurations for which the exploration cannot be done). This is the case, e.g., when k divides n.

1 Introduction

1.1 Framework

Recently a lot of attention has been devoted to the computational and complexity issues arising in systems of autonomous mobile entities located in a spatial universe $\mathcal{U}$. The entities have storage and processing capabilities, exhibit the same behavior (i.e., execute the same protocol), can move in $\mathcal{U}$ (their movement is constrained by the nature of $\mathcal{U}$), and are asynchronous in their actions. Depending on the context, the entities are sometimes called *agents*, other times *robots*; in the following, we use the latter. The research concern is on determining what tasks can be performed by such entities, under what conditions, and

at what cost. In particular, a central question is to determine what minimal hypotheses allow a given problem to be solved.

Depending on the nature of $\mathcal{U}$, there are two basic settings in which autonomous mobile entities are being investigated. The first setting, called sometimes *continuous universe*, is when $\mathcal{U}$ is the two-dimensional plane (e.g., [1, 9, 10, 19, 27, 29, 30]). The second setting, sometimes called *graph world* or *discrete universe*, is when $\mathcal{U}$ is a simple graph (e.g., [3, 4, 7, 12, 20, 21]). In both settings, each robot is viewed as operating in a *Look - Compute - Move* cycle. The robot observes the environment (Look), then, based on this observation, it decides to stay idle or to move (Compute), and in the latter case it moves towards its destination (Move).

Interestingly, in spite of the common features of the two settings, the researchers investigating them usually operate under two radically different assumptions on the robots' capabilities.
(1) *Communication vs Vision* - In the investigations in a graph world, the robots are assumed to communicate with each other directly; e.g., by means of tokens [6, 7], or whiteboards [12, 20], or when they meet [20]. Instead, in the studies on a continuous universe, the robots do not communicate in any explicit way; they however see the position of the other robots and can acquire knowledge from this information (e.g., see [1, 9, 10, 19, 26, 27, 29, 30]).
(2) *Persistency vs Obliviousness* - In addition to its program, each robot has a local memory (sometimes called notebook or workspace), used for computations and to store different amount of information obtained during the cycles. In all the investigations in a graph world, the local memory is possibly limited (e.g., each robot is a finite-state automaton) but almost always *persistent*: unless explicitly erased by the robot, all the information contained in the workspace will persist thoughout the robot's cycles. Instead, in the majority of the studies on a continuous universe, the robots are *oblivious*: all the information contained in the workspace is *cleared* at the end of each cycle. In other words, the robots have *no* memory of past actions and computations, and the computation is based solely on what has been determined in the current cycle. The importance of obliviousness comes from its link to *self-stabilization* and *fault-tolerance*.

Let us point out that there is nothing inherent in the nature of $\mathcal{U}$ that forces these differences in the assumptions. In other words, there is no reason why robots in a graph should not be oblivious; on the contrary, an oblivious solution would be highly desirable ensuring fault-tolerance and self stabilization. Similarly, there is nothing in the continuous domain that forbids robots from communicating explicitly; indeed, in the recent investigations on mobile sensor networks, the robots do communicate wirelessly [24].

Surprisingly, nobody has investigated how to solve problems in the discrete universe if the robots have the capabilities and limitations standard in the continuous one. In fact, with one exception, there are no studies on how a collection of asynchronous oblivious robots endowed with vision can perform a non-trivial task without any communication. The only exception is the recent investigation of the *gathering* problem in the ring [23].

In this paper, we continue this investigation and focus on a basic primitive problem in a graph world: *Exploration*, that is the process by which every node of the graph is visited by at least one robot, and we study this problem in a *ring*.

1.2 Our results

We consider the problem of exploring an anonymous ring of size n by k oblivious anonymous asynchronous robots scattered in the ring. The robots are endowed with vision but they are unable to communicate. Within finite time and regardless of the initial placement of the robots, each node must be visited by a robot and the robots must be in a configuration in which they all remain idle.

We first show that this problem is unsolvable if $k|n$. We then prove that, whenever $gcd(n, k) = 1$, for $k \geq 17$, the robots can explore the ring terminating within finite time. The proof is constructive: we present a terminating protocol that explores the ring starting from an arbitrary initial configuration, and prove its correctness.

Finally, we consider the minimum number $\rho(n)$ of robots that can explore a ring of size n. As a consequence of our positive result we show that $\rho(n)$ is $O(\log n)$. We also prove that $\rho(n) = \Omega(\log n)$ for arbitrarily large n. More precisely, there exists a constant c such that, for arbitrarily large n, we have $\rho(n) \geq c \log n$.

1.3 Related Work

Algorithms for graph exploration by mobile entities (robots) have been intensly studied in recent literature. Several scenarios have been considered. Most of the research is concerned with the case of a single robot exploring the graph. In [2, 6, 7, 14, 18] the robot explores strongly connected directed graphs and it can move only in the direction from head to tail of an edge, not vice-versa. In particular, [14] investigates the minimum time of exploration of directed graphs, and [2, 18] give improved algorithms for this problem in terms of the deficiency of the graph (i.e., the minimum number of edges to be added to make the graph Eulerian). Many papers, e.g., [15–17, 22, 25] study the scenario where the explored graph is undirected and the robot can traverse edges in both directions. In [15] the authors investigate the problem of how the availability of a map influences the efficiency of exploration. In [25] it is shown that a graph with n nodes and e edges can be explored in time $e + O(n)$. In some papers, additional restrictions on the moves of the robot are imposed. It is assumed that the robot has either a restricted tank [5, 8], forcing it to periodically return to the base for refueling, or that it is tethered, i.e., attached to the base by a rope or cable of restricted length [17].

Exploration of anonymous graphs presents different difficulties. In this case it is impossible to explore arbitrary graphs by a single robot if no marking of nodes is allowed. Hence the scenario adopted in [6, 7] allows the use of *pebbles* which the robot can drop on nodes to recognize already visited ones, and then remove them and drop in other places. The authors concentrate attention on

the minimum number of pebbles allowing efficient exploration and mapping of arbitrary directed n-node graphs. (In the case of undirected graphs, one pebble suffices for efficient exploration.) In [7] the authors compare exploration power of one robot with a constant number of pebbles to that of two cooperating robots, and give an efficient exploration algorithm for the latter scenario. In [6] it is shown that one pebble is enough if the robot knows an upper bound on the size of the graph, and $\Theta(\log \log n)$ pebbles are necessary and sufficient otherwise.

In all the above papers, except [7], exploration is performed by a single robot. Exploration by many robots has been investigated mostly in the context when moves of the robots are centrally coordinated. In [21], approximation algorithms are given for the collective exploration problem in arbitrary graphs. In [3, 4] the authors construct approximation algorithms for the collective exploration problem in weighted trees. On the other hand, in [20] the authors study the problem of distributed collective exploration of trees of unknown topology. However, the robots performing exploration have memory and can directly communicate with each other.

To the best of our knowledge, the very weak assumption of asynchronous identical robots that cannot send any messages and communicate with the environment only by observing it, has not been previously used in the context of graph exploration. It has been used, however in the case of robots moving freely in the plane (e.g., see [1, 9–11, 19, 26, 30]), where the robots were oblivious, i.e., it was assumed that they do not have any memory of past observations. Oblivious robots operate in Look-Compute-Move cycles, similar to those described in our scenario. The differences are in the amount of synchrony assumed in the execution of the cycles. In [13, 30] cycles were executed synchronously in rounds by all active robots, and the adversary could only decide which robots are active in a given cycle. In [9–11, 19, 26] they were executed asynchronously: the adversary could interleave operations arbitrarily, stop robots during the move, and schedule Look operations of some robots while others were moving.

Our scenario has been recently introduced in [23] to study the gathering problem in the ring. This scenario is very similar to the asynchronous model used in [19, 26]. The only difference with respect to [19, 26] is in the execution of Move operations. All possibilities of the adversary concerning interleaving operations performed by various robots as well as the characteristics of the robots are the same as in the model from [19, 26].

2 Preliminaries

2.1 Terminology and definitions

The network we consider is a ring of n nodes, $u_0, u_1, \ldots, u_{n-1}$; i.e., u_i is connected[5] to both u_{i-1} and u_{i+1}. The indices are used for notation purposes; in fact, the nodes are anonymous (i.e., identical) and the ring is unoriented. Operating in the ring are k identical robots; initially, at time $t = 0$, there is at most

[5] Here and in the following, all operations on the indices are modulo n.

one robot in each node. During the exploration, robots move, and at any time they occupy some nodes of the ring.

We shall indicate by $d_i(t)$ the multiplicity of robots present at node u_i at time t; more precisely $d_i(t) = 0$ indicates that there are no robots, $d_i(t) = 1$ indicates that there is exactly one robot, and $d_i(t) = 2$ indicates that there is more than one robot at u_i at time t. If $d_i(t) = 2$, we will say that there is a *tower* in u_i at time t.

Let $\delta^{+j}(t)$ denote the sequence $\delta^{+j}(t) =< d_j(t)\ d_{j+1}(t)\ \ldots d_{j+n-1}(t) >$, and let $\delta^{-j}(t)$ denote the sequence $\delta^{-j}(t) =< d_j(t)\ d_{j-1}(t)\ \ldots\ d_{j-(n-1)}(t) >$. The unordered pair[6] of sequences $\delta^{+j}(t)$ and $\delta^{-j}(t)$ describes the configuration of the system at time t viewed from node u_j. Let $\Delta^+(t) = \{\delta^{+j}(t) : 0 \leq j < n\}$ and $\Delta^-(t) = \{\delta^{-j}(t) : 0 \leq j < n\}$.

We will denote by $\delta_{max}(t)$ the lexicographically maximum sequence in $\Delta^+(t) \cup \Delta^-(t)$. It is immediate to verify that there is at most one maximal sequence in each of $\Delta^+(t)$ and $\Delta^-(t)$. A configuration is said to be *symmetric* if the maximal sequences in $\Delta^+(t)$ and $\Delta^-(t)$ are equal, and *asymmetric* otherwise.

Each robot operates in Look-Compute-Move cycles described in section 1.1. Cycles are performed asynchronously for each robot: the time between Look, Compute, and Move operations is finite but unbounded, and is decided by the adversary for each action of each robot. The only constraint is that moves are instantaneous, as in [23], and hence any robot performing a Look operation sees all other robots at nodes of the ring and not on edges. However, 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 in different nodes from those previously perceived by $\mathcal{R}$ because in the meantime they performed their Move operations. Hence robots may move based on significantly outdated perceptions. We assume that the robots can perceive, during the Look operation, if there is one or more robots in a given location; this ability, called *multiplicity detection* is a standard assumption in the continuous model [9, 23, 26]. We now describe formally what a robot perceives when performing a Look operation. Consider a robot $\mathcal{R}$ that, at time t is at node u_j and performs a Look; the result of this operation, called the *view* of $\mathcal{R}$ at time t, is precisely the unordered pair of sequences $\{\delta^{+j}(t), \delta^{-j}(t)\}$, that is, the configuration of the system at time t viewed from node u_j. We order all views as follows: order each pair $\{\delta^{+j}(t), \delta^{-j}(t)\}$ lexicographically and then use the lexicographic order on these pairs. From its view, the robot can determine $\delta_{max}(t)$, decide whether or not it is unique, and compute views of all other robots.

Let robot $\mathcal{R}$ perform in the same cycle a Look operation at time t' and a Move operation at time $t'' > t'$. We will say that $\mathcal{R}$ is *engaged to move* (or, simply *engaged*) in the open interval (t', t''); that is, $\mathcal{R}$ is engaged at any time t, where $t' < t < t''$.

[6] Since the ring is not oriented, agreement on only one of the two sequences might be impossible, and the pair cannot be ordered.

One final precision has to be added, concerning the decisions of robots made during the Compute action. Every such decision is based on the snapshot obtained during the last Look action. However it may happen that both edges incident to a node v currently occupied by the deciding robot look identical in this snapshot, i.e., v lies on a symmetry axis of the configuration. In this case if the robot decides to take one of these edges, it may take any of the two. We assume the worst-case decision in such cases, i.e., that the actual edge among the identically looking ones is chosen by an adversary.

We say that exploration of a n-node ring is possible with k robots, if there exists an algorithm which, starting from any initial configuration of the k robots without towers, allows the robots to explore the entire ring and brings all robots to a configuration in which they all remain idle. Obviously, if $n = k$, the exploration is already accomplished, hence we always assume that $k < n$.

2.2 Basic restriction

Lemma 1. *Let $k < n$. If $k|n$ then the exploration of a n-node ring with k robots is not possible.*

Proof. By contradiction, let P be a solution protocol. Choose as the initial configuration an equidistant placement of the k robots in the ring (it exists since $k|n$). Thus, initially the states of all robots are identical, say $\sigma(0)$. Clearly this state is not a terminal state. Otherwise, since $k < n$, P would terminate without exploring the ring, thus contradicting the correctness of P. Consider now an adversary that uses a synchronous scheduler and a consistent orientation of the ring. Then, at each time step t, the states of all robots continue to be identical, say $\sigma(t)$, and furthermore they are the same as those of previous steps; i.e., $\sigma(t) = \sigma(0)$ for all t. Hence the robots will never enter a terminal state, contradicting the fact that P leads within finite time to a configuration in which all robots remain idle. □

In the following we will consider the case when $gcd(n, k) = 1$, and design an algorithm that allows $k \geq 17$ robots to explore a n-node ring whenever $gcd(n, k) = 1$. Observe that if $gcd(n, k) = 1$, the configuration is either asymmetric or it is symmetric with respect to a single axis of symmetry. Therefore at most two robots can have the same view. In the symmetric case, the adjective symmetric will be used with respect to this unique axis of symmetry. Note that symmetric robots have the same view.

3 Exploration of a ring

3.1 Overview of the algorithm

The overall structure of the algorithm can be seen as a sequence of three distinct phases: *Set-Up*, *Tower-Creation*, and *Exploration*.

The purpose of the *Set-Up* phase is to transform the (arbitrary) initial configuration into one from a predetermined set of configurations (called *no-towers-final*) with special properties. More precisely, in the *Set-Up* phase, the robots create a configuration where there is a single set of consecutive nodes occupied by robots, or two such sets of the same size (called blocks). When the configuration is *no-towers-final*, the next phase begins.

The purpose of the *Tower-Creation* phase is to transform the *no-towers-final* configuration created in the previous phase, into one from a predetermined set of configurations (called *towers-completed*) in which everything is prepared for exploration to begin. More precisely, in the *Tower-Creation* phase, one or two towers are created inside each block (the number depending on the parity of the size of the block); furthermore a number of robots become uniquely identified as explorers. As soon as the configuration is *towers-completed*, the next phase begins.

During the *Exploration* phase, the ring is actually being explored. The configuration reached upon exploration depends solely on the configuration at the beginning of this phase. The set of these special *exploration-completed* configurations is uniquely identified, and once in a configuration of this type, no robots will make any further move.

The *Set-Up* phase is by far the most complicated part of the algorithm, hence we describe it in a detailed way. To simplify the presentation, the next two phases are described in detail only in the case when k is odd. The case of k even can be described and analyzed using similar techniques and is omitted.

Since the robots are oblivious (i.e., they have no recollection of actions and computations made in previous cycles), there is no explicit way for them to record which phase is the current one. This information is derived by a robot solely based on the configuration currently observed (i.e., the one obtained as a result of the Look operation). Since the determination of the phase should be non-ambiguous, each reachable configuration should be assigned to exactly one phase.

For any possible configuration we will identify a set of *players*, which are the robots deciding to move if they perform a Look operation in this configuration, and corresponding *destinations*, i.e., target neighbors. The exploration algorithm (which contains the rules describing the Compute actions in the robot's cycle) can be succinctly formulated as follows.

Algorithm Ring Exploration

If I am a *player*
move to my *destination*

3.2 Set-Up Phase

The first phase of the protocol is the *Set-Up*. The fact of being in this phase is easily recognizable by the robots since, unlike those of the other phases, the

configurations of this phase contain no towers. Precisely because they contain no towers, any configuration of this phase can be an initial configuration.

We define the *interdistance* of a configuration as the minimum distance taken over all pairs of distinct robots in the configuration. Given an arbitrary configuration of interdistance d, a *block* is a maximal set of robots, of size at least 2, forming a line with a robot every d nodes. The *size* of a block is the number of robots it contains. The *border* of a block are the two nodes occupied by the two extremal robots of the block. A robot not in a block is said to be *isolated*. A robot is said to be a *neighbor* of a block/robot if in at least one direction there is no robot between itself and the block/robot. A *leader* of a configuration is a robot from which the view is the maximal in the configuration, with respect to the order defined in Section 2.1. A block containing a leader is called a *leading block*. Otherwise it is called a *non-leading block*.

The *Set-Up* phase is described by identifying four types of configurations that form a disjoint partition of all possible configurations without towers. For each type we indicate the players and their destinations.

Type A. A configuration of *type A* is a configuration of interdistance $d \geq 1$ with at least one isolated robot. Consider an arbitrary configuration of type A and let S be the maximum among the sizes of the blocks that are neighbors of at least one isolated robot. Let $\mathcal{I}$ be the set of isolated robots that are neighbors of a block of size S such that no other isolated robot is closer to a block of size S. The players in a configuration of type A are all the robots in $\mathcal{I}$. The destination of a player is its adjacent node in the direction of the closest neighboring block of size S.

Type B. A configuration of *type B* is a configuration of interdistance $d \geq 1$, without isolated robots, and containing at least one non-leading block. More precisely, if all blocks have the same size then the configuration is of type $B1$. Otherwise, it is of type $B2$.

Consider an arbitrary configuration of type $B1$. If there is only one leader, then the player is the leader and its destination is its adjacent node outside the block it belongs to. From now on, we assume that there are two leaders. This implies that the configuration is symmetric. There are two cases. The first case is when the blocks are of size 2. Since $k \geq 17$, there are at least 9 blocks and hence there exist two symmetric blocks separated by at least three blocks on each side. (Observe that this property does not hold for $k = 16$.) The players in such a configuration are the robots of such two blocks, having the smallest view. The destination of a player is its adjacent node outside the block it belongs to. We consider now the second case, that is when the blocks are of size larger than 2. The players in such a configuration are the pair of symmetric robots that are the closest to each other among the robots at the border of a block and such that these two robots are not neighbor. The destination of a player is the adjacent node outside the block it belongs to.

Consider an arbitrary configuration of type $B2$ and let s be the minimum size of a block in the configuration. Let S be the maximum among the sizes of the

blocks that are neighbors of a block of size s and let d be the minimal distance between a block of size s and a block of size S. We define $\mathcal{T}$ as the set of robots belonging to a block of size s, neighbors of a block of size S, and at distance d from it. The players in a configuration of type $B2$ are the robots in $\mathcal{T}$ with the largest view. The destination of a player is its adjacent node in the direction of its neighboring block of size S.

Type C. A configuration of *type C* is a configuration of interdistance $d \geq 2$, without isolated robots, and such that each of its blocks is a leading block. Note that this implies that either all robots are in the same block or the robots are divided in two blocks of the same size. Moreover, there are exactly two leaders because the configuration is symmetric. The players in a configuration of type C are the two leaders. The destination of a player is its adjacent node in the direction of the block it belongs to. (This is not ambiguous because leaders are always located at the border of a block.)

Type D. A configuration of *type D* is a configuration of interdistance $d = 1$, without isolated robots, and such that each of its blocks is a leading block. Type D is the set of configurations *no-towers-final*. When such a configuration is reached, the *Set-Up* phase ends and the *Tower-Creation* phase begins.

Note that types A, B, C and D form a partition of all possible initial configurations (when $gcd(n, k) = 1$).

The general idea of the *Set-Up* phase is to create few compact blocks (interdistance 1). Each decrease of interdistance is accomplished by first decreasing the number of blocks. The following lemmas show how this progress is achieved. The proofs of most of them are ommitted due to lack of space. Theorem 1 shows that a *no-towers-final* configuration is always reached at the end.

Lemma 2. *Assume that at some time t the configuration is of type A and that the only engaged robots are isolated robots engaged to move toward a neighboring block. Then after finite time, the configuration is of type B, C or D, of the same interdistance as in time t, and no robots are engaged.*

Lemma 3. *Assume that at some time t the configuration is asymmetric, of type $B1$, and that no robots are engaged. Then after finite time, the configuration is of type $B2$, of the same interdistance as in time t, no robots are engaged, and there is one block less than at time t.*

Lemma 4. *Assume that at some time t the configuration is symmetric, of type $B1$, with blocks of size 2, and that no robots are engaged. Then after finite time, the configuration is of type C or D, of the same interdistance as in time t, and no robots are engaged.*

Lemma 5. *Assume that at some time t the configuration is symmetric, of type $B1$, with blocks of size $s \geq 3$, and that no robots are engaged. Then after finite time, the configuration is of type $B2$, C or D, of the same interdistance as in time t, no robots are engaged, and there are fewer blocks than at time t.*

Lemma 6. *Assume that at some time t the configuration is of type $B2$ and that no robots are engaged. Then after finite time, the configuration is of type B, C or D, of the same interdistance as in time t, no robots are engaged, and there are fewer blocks than at time t.*

Lemma 7. *Assume that at some time t the configuration is of type C, of interdistance $d \geq 2$, and that no robots are engaged. Then at some time $t' > t$ one of the two following situations occurs:*

- *The configuration is of type A, of interdistance $d-1$, and the only engaged robots are isolated robots engaged to move toward a block.*
- *The configuration is of type B, of interdistance $d-1$, and no robots are engaged.*

Proof. Assume that at some time t the configuration is of type C, of interdistance $d \geq 2$, and no robots are engaged. The players are the two leaders. After finite time, at least one will move. Consider the moment t_1 where the first moves. At this moment the configuration changes to type A. If the other player moved at the same time or is not engaged, we are done because the configuration is of type A, of interdistance $d-1$, and no robots are engaged.

Thus we assume that the other player $\mathcal{R}$ is engaged at time t_1. By the definition of type C configurations and the fact that $k \geq 17$, it is engaged to move toward an isolated robot $\mathcal{R}'$ that is at distance exactly d. Note that until $\mathcal{R}$ moves, there is only one block (of interdistance $d-1$) and $\mathcal{R}$ is a neighbor of it (its other neighbor is $\mathcal{R}'$). Moreover, $\mathcal{R}$ is isolated and no robots will move toward it to make a block because no other robot is engaged at the moment and because a player in a configuration of type A never moves toward an isolated robot. Therefore, the configuration will remain of type A while $\mathcal{R}$ does not move.

Consider now the time t_2 where $\mathcal{R}$ makes its move. If this move does not make it belong to a block, then the configuration is of type A, of interdistance $d-1$, and the only engaged robots are isolated robots engaged to move toward a block. Assume now that the move of $\mathcal{R}$ makes it belong to a block. Then necessarily it is a new block, of size two, and formed with robot $\mathcal{R}'$. If $\mathcal{R}'$ is not engaged then we are at a time t' satisfying the lemma. Indeed if there are isolated robots then the first situation occurs, and if there are not then the other block is larger, of size $k-2$, and thus the second situation occurs. If $\mathcal{R}'$ is engaged at time t_2, then there are no isolated robots because there is none between $\mathcal{R}$ and the other block (in the segment excluding $\mathcal{R}'$) and there is none between $\mathcal{R}'$ and the other block (in the segment excluding $\mathcal{R}$) since $\mathcal{R}'$ got engaged as an isolated robot and thus was engaged to move toward a block. Therefore, we are in the following situation: there are two blocks of sizes $k-2$ and 2, at distances at least $d+1$ (on both sides); exactly one robot of the smaller block is engaged to move toward the other block, and no other robots are engaged. Thus the configuration is of type $B2$ and after some finite time, one of the two robots of the smaller block will move. At this moment, the first situation occurs, which concludes the proof of the lemma. □

Theorem 1. *Any initial configuration is transformed after finite time into a configuration of type D (i.e.* no-towers-final*) without engaged robots.*

Proof. Let Φ be the property that the only engaged robots (if any) in a given configuration are isolated ones and they are engaged to move toward a neighboring block. For any configuration c of type A, B or C define the triple $T(c) = (d, t, x)$, where d is the interdistance of c, t is the type of c, i.e., t is A, B or C, and x is the number of blocks in c. Order all triples lexicographically, assuming that $C < B < A$. Lemmas 2 – 7 imply that any configuration c of type A, B or C satisfying property Φ is transformed after finite time either in a configuration c' of type A, B or C satisfying property Φ, such that $T(c') < T(c)$, or in a configuration of type D with no robots engaged. Since any initial configuration satisfies property Φ, this concludes the proof.

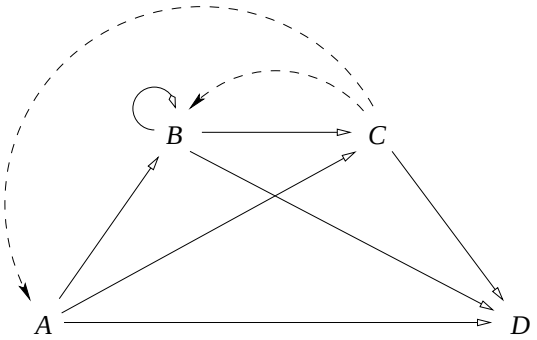

Fig. 1. Progress toward type D. Dashed arrows correspond to transitions where the interdistance decreases. The loop corresponds to a transition where the number of blocks decreases.

Figure 1 ilustrates the progress of configurations toward type D. □

3.3 Tower-Creation Phase

The second phase of the protocol is *Tower-Creation*. This phase begins with a configuration of type D, i.e., one of the configurations *no-towers-final*. The goal of this phase is to create one or two towers in each block (depending on the parity of the number of robots per block). More precisely, in a block of odd size there will be one tower, and in a block of even size there will be two towers. In a block of odd size the tower is formed by the central robot moving to its adjacent node containing the robot with the larger view. In a block of even size the two towers are formed by the two central robots moving to their other neighbors. The obtained configuration is called *towers-completed*. This is easily recognizable as each block of a *no-towers-final* configuration is transformed as follows. A block of odd size $2a+1$ is transformed into a segment of a consecutive robots followed by

an empty node, followed by a tower, followed by a segment of $a-1$ consecutive robots. A block of even size $2a$ is transformed into a segment of $a-2$ consecutive robots followed by a tower, followed by two empty nodes followed by a tower, followed by a segment of $a-2$ consecutive robots (see Figure 2).

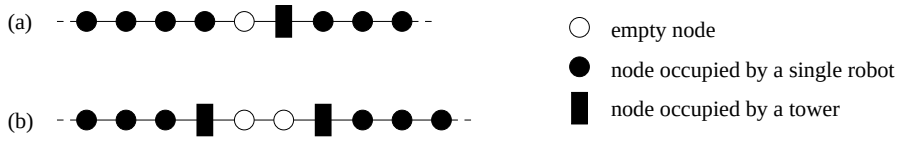

Fig. 2. Transformed blocks (a) of odd size (b) of even size

Since we limit our detailed description to the case of k odd, the only possibility is that the *no-towers-final* configuration starting the *Tower-Creation* phase consists of one block of odd size. In this case the phase consists of one move of the central robot. This robot moves to the neighbor decided by the adversary, as the configuration is symmetric.

3.4 Exploration Phase

Exploration starts when towers in the preceding phase are created. Note that the empty nodes adjacent to towers have already been explored, so the segments of empty nodes between the transformed blocks are the only ones possibly not yet explored. Each of these segments is explored in the current phase using one or two robots closest to the segment. If k is even, such a segment must lie between two segments of consecutive robots of equal size, and it is explored by the two border robots that meet in the middle of the segment (either at the extremities of the central edge, or in the central node). The obtained configuration is called *exploration-completed*.

We describe exploration in detail for k odd. In this case the configuration starting *Exploration* phase is a single transformed block of odd size, with $a = (k-1)/2$ (hence in particular $a \geq 3$), see Figure 2 (a). The unique player is the robot in the segment of $a-1$ consecutive robots, farthest from the tower. This robot moves to its empty neighbor node. In a resulting configuration with a single isolated robot, the player is this robot and it moves toward the segment of a consecutive robots. When the configuration contains $a+1$ consecutive robots followed by an empty node, followed by a tower, followed by a segment of $a-2$ consecutive robots, all robots remain idle. At this point exploration is completed.

From Theorem 1 describing the conclusion of the *Set-Up* phase and from the properties of *Tower-Creation* and *Exploration* phases we get the following result.

Theorem 2. *Let* $17 \leq k < n$. *Algorithm* Ring Exploration *allows a team of* k *robots to explore a* n*-node ring and enter a terminal state within finite time, provided* $gcd(n,k) = 1$.

4 Size of the minimum team

In this section we show that the minimum number of robots that can explore a n-node ring regardless of their initial position, is logarithmic in n. More precisely, we have the following result.

Theorem 3. *The minimum number $\rho(n)$ of robots that can explore a n-node ring has the following properties:*

1. *$\rho(n) \in O(\log n)$;*
2. *there exists a constant c such that, for infinitely many n, we have $\rho(n) \geq c \log n$.*

Proof. Let p_j denote the j-th prime, and let $p_j\#$ denote the *p_j-primorial*, that is

$$p_j\# = \Pi_{i=1}^{j} \quad p_i \tag{1}$$

An important property of the primorial is the following [28]:

$$\lim_{j\to\infty} \ (p_j\#)^{\frac{1}{p_j}} \ = \ e \, . \tag{2}$$

We will now prove each part of the theorem separately.

Part 1.

Let $f(n)$ be the smallest integer coprime with n and larger than 16. Thus, by Theorem 2, exploration is possible with $f(n)$ agents. Hence, $\rho(n) \leq f(n)$.

Take j such that $\frac{p_j\#}{13\#} \leq n < \frac{p_{j+1}\#}{13\#}$. We have $f(n) \leq p_{j+1}$. (Otherwise, all primes in $\{17, \ldots, p_{j+1}\}$ divide n and hence $n \geq \frac{p_{j+1}\#}{13\#}$, contradiction.) By property (2) we have $2 \leq (p_j\#)^{\frac{1}{p_j}}$, for sufficiently large j. Hence $2^{p_j} \leq p_j\#$, and thus $p_j \leq \log(p_j\#)$. Hence $p_{j+1} \leq \log(p_{j+1}\#) \leq \log(p_j\#) + \log p_j$. Since $p_{j+1} \leq p_j\# + 1 \leq 2 \cdot 13\# \cdot n$, we have $\rho(n) \leq f(n) \leq p_{j+1} \leq \log(2 \cdot 13\# \cdot n) + \log n$, which is at most $3 \log n$, for sufficiently large n.

Part 2.

Let n be the least common multiple of integers $1, 2, \ldots, m$. Let $g(n)$ be the smallest integer not dividing n. By Lemma 1 we have $\rho(n) \geq g(n)$. We have $g(n) \geq m + 1$. The Prime Number Theorem implies $\frac{\ln n}{m} \to 1$. Hence $\ln n \leq 2m$, for sufficiently large m. This implies the existence of a constant c such that $\rho(n) \geq g(n) \geq m + 1 > m \geq \frac{\ln n}{2} \geq c \log n$. □

It should be noted that for some specific values of n, the number $\rho(n)$ is constant. For example, if $n > 17$ is prime, then Theorem 2 shows that 17 robots can explore the n-node ring, hence $\rho(n) \leq 17$.

5 Conclusions

In this paper we have analyzed the exploration problem in rings by asynchronous robots when they are oblivious and can see the environment but cannot communicate. This is a further step in the understanding of how these robots' capabilities, standard in continuous universes, can be exploited in the discrete ones. These results open several interesting problems and pose intriguing questions. First, the complete characterization of couples (n, k) for which exploration of the ring is solvable remains open. Next, the problem of exploring other topologies and arbitrary graphs is a natural extension of this work. Moreover, since the robots cannot communicate, they have to be able to observe the environment; an immediate question is what happens if the robots can only see within a fixed distance. Accuracy of vision as well as fault-tolerance are issues that should be addressed by future research.

Acknowledgment This work was done during the stay of David Ilcinkas at the Research Chair in Distributed Computing at the Université du Québec en Outaouais and at the University of Ottawa, as a postdoctoral fellow. Andrzej Pelc was partially supported by the Research Chair in Distributed Computing at the Université du Québec en Outaouais, Paola Flocchini was partially supported by the University Research Chair of the University of Ottawa. This work was supported in part by the Natural Sciences and Engineering Research Council of Canada under Discovery grants.

References

1. N. Agmon, D. Peleg: Fault-Tolerant Gathering Algorithms for Autonomous Mobile Robots. SIAM J. Comput. 36(1): 56-82 (2006).
2. S. Albers and M. R. Henzinger, Exploring unknown environments, SIAM J. Comput. 29 (2000), 1164-1188.
3. I. Averbakh and O. Berman, A heuristic with worst-case analysis for minimax routing of two traveling salesmen on a tree, Discr. Appl. Math. 68 (1996), 17-32.
4. I. Averbakh and O. Berman, $(p-1)/(p+1)$-approximate algorithms for p-traveling salesmen problems on a tree with minmax objective, Discr. Appl. Mathematics 75 (1997), 201-216.
5. B. Awerbuch, M. Betke, R. Rivest and M. Singh, Piecemeal graph learning by a mobile robot, Proc. 8th Conf. on Comput. Learning Theory (1995), 321-328.
6. M.A. Bender, A. Fernandez, D. Ron, A. Sahai and S. Vadhan, The power of a pebble: Exploring and mapping directed graphs, Proc. 30th Ann. Symp. on Theory of Computing (STOC 1998), 269-278.
7. M.A. Bender and D. Slonim, The power of team exploration: Two robots can learn unlabeled directed graphs, Proc. 35th Ann. Symp. on Foundations of Computer Science (FOCS 1994), 75-85.
8. M. Betke, R. Rivest and M. Singh, Piecemeal learning of an unknown environment, Machine Learning 18 (1995), 231-254.
9. M. Cieliebak, P. Flocchini, G. Prencipe, N. Santoro, Solving the Robots Gathering Problem, Proc. 30th Int. Col. Automata, Languages and Programming (ICALP 2003), LNCS 2719: 1181-1196.

10. R. Cohen, D. Peleg, Robot convergence via center-of-gravity algorithms, Proc. 11th Int. Col. on Structural Information and Communication Complexity (SIROCCO 2004), LNCS 3104: 79-88.
11. J. Czyzowicz, L. Gasieniec, A. Pelc, Gathering few fat mobile robots in the plane, Proc. 10th Int. Conf. on Principles of Distributed Systems (OPODIS'2006), LNCS 4288, 744-753.
12. S. Das, P. Flocchini, S. Kutten, A. Nayak, N. Santoro, Map construction of unknown graphs by multiple agents, Theoretical Computer Science, to appear, 2007.
13. X. Défago and A. Konagaya. Circle formation for oblivious anonymous mobile robots with no common sense of orientation. In Workshop on Principles of Mobile Computing, pages 97–104, 2002.
14. X. Deng and C. H. Papadimitriou, Exploring an unknown graph, Journal of Graph Theory 32 (1999), 265-297.
15. A. Dessmark and A. Pelc, Optimal graph exploration without good maps, Proc. 10th European Symp. on Algorithms (ESA 2002), LNCS 2461, 374-386.
16. K. Diks, P. Fraigniaud, E. Kranakis and A. Pelc, Tree exploration with little memory, Proc. 13th Ann. ACM-SIAM Symp. on Discrete Algorithms (SODA 2002), 588-597.
17. C.A. Duncan, S.G. Kobourov and V.S.A. Kumar, Optimal constrained graph exploration, Proc. 12th Ann. ACM-SIAM Symp. on Discrete Algorithms (SODA 2001), 807-814.
18. R. Fleischer, G. Trippen, Exploring an unknown graph efficiently, Proc. 13th European Symp. on Algorithms (ESA 2005), 11-22.
19. P. Flocchini, G. Prencipe, N. Santoro, P. Widmayer. Hard tasks for weak robot. Proc. 10th Int. Symp. on Algorithm and Computation (ISAAC 1999), LNCS 1741, 93-102.
20. P. Fraigniaud, L. Gasieniec, D. Kowalski, A. Pelc, Collective tree exploration, Proc. Latin American Theoretical Informatics (LATIN'2004), LNCS 2976, 141-151.
21. G. N. Frederickson, M. S. Hecht and C. E. Kim, Approximation algorithms for some routing problems. SIAM J. Comput. 7 (1978), 178-193.
22. L. Gasieniec, A. Pelc, T. Radzik, X. Zhang, Tree exploration with logarithmic memory, Proc. 18th Annual ACM-SIAM Symp. on Discrete Algorithms (SODA 2007), January 2007, New Orleans, Louisiana, USA.
23. R. Klasing, E. Markou, A. Pelc, Gathering asynchronous oblivious mobile robots in a ring, Proc. 17th Int. Symp. on Algorithms and Computation (ISAAC 2006).
24. X. Li and N. Santoro, An Integrated self-deployment and coverage maintenance scheme for mobile sensor networks. Proc of 2nd Int. Conf. on Mobile Ad-Hoc and Sensors Networks (MSN'06), 2006.
25. P. Panaite and A. Pelc, Exploring unknown undirected graphs, Journal of Algorithms 33 (1999), 281-295.
26. G. Prencipe, On the feasibility of gathering by autonomous mobile robots. Proc. 12th Int. Col. on Structural Information and Communication Complexity (SIROCCO 2005), LNCS 3499: 246-261.
27. G. Prencipe, N. Santoro. Distributed algorithms for mobile robots. Proc. 5th IFIP Int. Conf. on Theoretical Computer Science (TCS 2006), 47-62.
28. S.M. Ruiz: A Result on Prime Numbers. Math. Gaz. 81 (1997), 269.
29. S. Souissi, X. Défago and M. Yamashita, Gathering asynchronous mobile robots with inaccurate compasses. In Proc. 10th Int. Conf. on Principles of Distributed Systems (OPODIS 2006), LNCS 4305, 333-349.
30. I. Suzuki, M. Yamashita, Distributed anonymous mobile robots: formation of geometric patterns. SIAM J. Comput. 28(4): 1347-1363 (1999).