Optimal Probabilistic Ring Exploration by Asynchronous Oblivious Robots

Stéphane Devismes[°] — Franck Petit[†] — Sébastien Tixeuil[⋆,‡]
[°] VERIMAG UMR 5104, Université Joseph Fourier, Grenoble (France)
[†] INRIA, LIP UMR 5668, Université de Lyon / ENS Lyon (France)
[⋆] Université Pierre et Marie Curie - Paris 6, LIP6, France

[‡] INRIA project-team Grand Large

N° 6838

February 2009

Thème NUM

ISSN 0249-6399 ISRN INRIA/RR--6838--FR+ENG

Optimal Probabilistic Ring Exploration by Asynchronous Oblivious Robots

Stéphane Devismes[°] , Franck Petit[†] , Sébastien Tixeuil[⋆,‡]
[°] VERIMAG UMR 5104, Université Joseph Fourier, Grenoble (France)
[†] INRIA, LIP UMR 5668, Université de Lyon / ENS Lyon (France)
[⋆] Université Pierre et Marie Curie - Paris 6, LIP6, France
[‡] INRIA project-team Grand Large

Thème NUM — Systèmes numériques
Projet Grand Large

Rapport de recherche n° 6838 — February 2009 — 26 pages

Abstract: We consider a team of k identical, oblivious, asynchronous mobile robots that are able to sense (*i.e.*, view) their environment, yet are unable to communicate, and evolve on a constrained path. Previous results in this weak scenario show that initial symmetry yields high lower bounds when problems are to be solved by *deterministic* robots.

In this paper, we initiate research on probabilistic bounds and solutions in this context, and focus on the *exploration* problem of anonymous unoriented rings of any size. It is known that $\Theta(\log n)$ robots are necessary and sufficient to solve the problem with k deterministic robots, provided that k and n are coprime. By contrast, we show that *four* identical probabilistic robots are necessary and sufficient to solve the same problem, also removing the coprime constraint. Our positive results are constructive.

Key-words: Robots, Anonimity, Obliviousness, Exploration

Exploration d'Anneau Probabiliste Optimale par des Robots Asynchrones et Amnésiques

Résumé : Nous considérons une équipe de k robots identiques, amnésiques, asynchrones et mobiles qui sont capables de percevoir leur environnement mais incapables de communiquer, et évoluent sur des circuits contraints. Les résultats précédents qui utilisent le même scénario montrent que la symmétrie initiale potentielle induit des bornes inférieures élevées dès lors que le problème doit être résolu par des robots déterministes.

Dans cet article, nous initions la recherche sur les bornes et sur les solutions probabilistes dans le même contexte, et nous considérons le problème de l'exploration d'anneaux anonymes et non orientés de taille quelconque. Il est connu que $\Theta(\log n)$ robots sont nécessaires et suffisants dans le cas déterministe pour résoudre le problème avec k robots, tant que k et n sont premiers entre eux. En contrepartie, nous montrons que quatre robots identiques probabilistes sont nécessaires et suffisants pour résoudre le même problème, tout en supprimant la contrainte de coprimalité. Nos résultats positifs sont constructifs.

Mots-clés : Robots, Anonymat, Amnésie, Exploration

1 Introduction

We consider autonomous robots that are endowed with visibility sensors (but that are otherwise unable to communicate) and motion actuators. Those robots must collaborate to solve a collective task, namely *exploration*, despite being limited with respect to input from the environment, asymetry, memory, etc. In this context, the exploration tasks requires every possible location to be visited by at least one robot, with the additional constraint that all robots stop moving after task completion.

Robots operate in *cycles* that comprise *look*, *compute*, and *move* phases. The look phase consists in taking a snapshot of the other robots positions using its visibility sensors. In the compute phase a robot computes a target destination based on the previous observation. The move phase simply consists in moving toward the computed destination using motion actuators.

The robots that we consider here have weak capacities: they are *anonymous* (they execute the same protocol and have no mean to distinguish themselves from the others), *oblivious* (they have no memory that is persistent between two cycles), and have no compass whatsoever (they are unable to agree on a common direction or orientation).

Related works The vast majority of literature on coordinated distributed robots considers that those robots are evolving in a *continuous* two-dimentional Euclidian space and use visual sensors with perfect accuracy that permit to locate other robots with infinite precision [2, 13, 14, 10, 6, 5].

Several works investigate restricting the capabilities of both visibility sensors and motion actuators of the robots, in order to circumvent the many impossibility results that appear in the general continuous model. In [1, 9], robots visibility sensors are supposed to be accurate within a constant range, and sense nothing beyond this range. In [9, 4], the space allowed for the motion actuator was reduced to a one-dimentional continuous one: a ring in [9], an infinite path in [4].

A recent trend was to shift from the classical continuous model to the *discrete* model. In the discrete model, space is partitioned into a *finite* number of locations. This setting is conveniently represented by a graph, where nodes represent locations that can be sensed, and where edges represent the possibility for a robot to move from one location to the other. Thus, the discrete model restricts both sensing and actuating capabilities of every robot. For each location, a robot is able to sense if the location is empty of if robots are positioned on it (instead of sensing the exact position of a robot). Also, a robot is not able to move from a position to another unless there is explicit indication to do so (*i.e.*, the two locations are connected by an edge in the representing graph). The discrete model permits to simplify many robot protocols by reasoning on finite structures (*i.e.*, graphs) rather than on infinite ones. However, as noted in most related papers [12, 11, 7, 8], this simplicity comes with the cost of extra symmetry possibilities, especially when the authorized paths are also symmetric (indeed, techniques to break formation such as those of [6] cannot be used in the discrete model).

The two main problems that have been studied in the discrete robot model are gathering [12, 11] and exploration [7, 8]. For gathering, both breaking symmetry [12] and preserving symmetry are meaningful approaches. For exploration, the fact that robots need to stop after exploring all locations requires robots to "remember" how much of the graph was explored, *i.e.*, be able to distinguish between various stages of the exploration process since robots have no persistent memory. As configurations can be distinguished only by robot positions, the main complexity measure is then the number of robots that are needed to explore a given graph. The vast number of symmetric situations induces a large number of required robots. For tree networks, [8] shows that $\Omega(n)$ robots are necessary for most n-sized tree, and that sublinear robot complexity is possible only if the maximum degree of the tree is 3. In uniform rings, [7] proves that the necessary and sufficient number of robots is $\Theta(\log n)$, although it is required that the number k of robots and the size n of the ring are coprime. Note that all previous approaches in the discrete model are *deterministic*, *i.e.*, if a robot is presented twice the same situation, its behavior is the same in both cases.

Our contribution In this paper, we initiate research on *probabilistic* bounds and solutions in the discrete robot model, and focus on the *exploration* problem of anonymous unoriented rings of any size. By contrast with [7] while in the same system setting, we show that *four* identical probabilistic robots are necessary and sufficient to solve the same problem, also removing the coprime constraint between the number of robots and the size of the ring. Our negative result show that for any ring of size at least four, there cannot exist any protocol with three robots in our setting, even if they are allowed to make use of probabilistic primitives. Our positive results are constructive, as we present a randomized protocol with four robots for any ring of size more than eight.

Outline The remaining of the paper is divided as follows. Section 2 presents the system model that we use throughout the paper. Section 3 provides evidence that no three probabilistic robots can explore every ring, while Section 4 presents our protocol with four robots. Section 5 gives some concluding remarks.

2 Model

Distributed System We consider systems of autonomous mobile entities called *agents* or *robots* evolving into a *graph*. We assume that the graph is a *ring* of n nodes, $u_0, \ldots, u_{n-1}$, *i.e.*, u_i is connected to both u_{i-1} and u_{i+1} — every computation over indices is assumed to be modulus n. The indices are used for notation purposes only: the nodes are *anonymous* and the ring is *unoriented*, *i.e.*, given two neighboring nodes u, v, there is no kind of explicit or implicit labelling allowing to determine whether u is on the right or on the left of v. Operating in the ring are $k \leq n$ anonymous robots.

A *protocol* is a collection of k *programs*, one operating on each robot. The program of a robot consists in executing *Look-Compute-Move cycles* infinitely many times. That is, the robot first observes its environment (Look phase). Based on its observation, a robot then (probabilistically or deterministically) decides — according to its program — to move or stay idle (Compute phase). When an robot decides a move, it moves to its destination during the Move phase.

The robots do not communicate in an explicit way; however they see the position of the other robots and can acquire knowledge from 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*). The robots are also *uniform* and *anonymous*, i.e, they all have the same program using no local parameter (such that an identity) allowing to differentiate any of them.

Computations Time is represented by an infinite sequence of instants 0, 1, 2, ... At every instant $t \geq 0$, a non-empty subset of robots is activated to execute a cycle. The execution of each cycle is assumed to be *atomic*: every robot that is activated at instant t instantaneously executes a full cycle between t and $t+1$. Atomicity guarantees that at any instant the robots are on some nodes of the ring but not on edges. Hence, during a Look phase, a robot sees no robot on edges.

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. More precisely $d_i(t) = j$ indicates that there is j robots in node u_i at instant t. If $d_i(t) \geq 2$, then we say that there is a *tower* in u_i at instant t (or simply there is a *tower* in u_i when it is clear from the context). We say a node u_i is *free at instant* t (or simply *free* when it is clear from the context) if $d_i(t) = 0$. Conversely, we say that u_i is *occupied at instant* t (or simply *occupied* when it is clear from the context) if $d_i(t) \neq 0$.

Given an arbitrary orientation of the ring and a node u_i, $\gamma^{+i}(t)$ (respectively, $\gamma^{-i}(t)$) denotes the sequence $\langle d_i(t)d_{i+1}(t) \ldots d_{i+n-1}(t)\rangle$ (resp., $\langle d_i(t)d_{i-1}(t) \ldots d_{i-(n-1)}(t)\rangle$). The sequence $\gamma^{-i}(t)$ is called *mirror* of $\gamma^{+i}(t)$ and conversely. Since the ring is unoriented, agreement on only one of the two sequences $\gamma^{+i}(t)$ and $\gamma^{-i}(t)$) is impossible. The (unordered) pair $\{\gamma^{+i}(t), \gamma^{-i}(t)\}$ is called the *view* of node u_i at instant t (we omit "at instant t" when it clear from the context). The view of u_i is said to be *symmetric* if and only if $\gamma^{+i}(t) = \gamma^{-i}(t)$. Otherwise, the view of u_i is said to be *asymmetric*.

By convention, we state that the *configuration* of the system at instant t is $\gamma^{+0}(t)$. Any configuration from which there is a probability 0 that a robot moves is said *terminal*. Let $\gamma = \langle x_0x_1 \ldots x_{n-1}\rangle$ be a configuration. The configuration $\langle x_ix_{i+1} \ldots x_{i+n-1}\rangle$ is obtained by rotating γ of $i \in [0 \ldots n-1]$. Two configurations γ and γ' are said *undistinguable* if and only if γ' can be obtained by rotating γ or its mirror. Two configurations that are not undistinguable are said *distinguable*. We designate by *initial configurations* the configurations from which the system can start at instant 0.

During the Look phase of some cycle, it may happen that both edges incident to a node v currently occupied by the robot look identical in the snapshot, *i.e.*, v lies on a symmetric axis of the configuration. In this case, if the robot decides to move, it may traverse any of the two edges. We assume the worst case decision in such cases, *i.e.*, that the decision to traverse one of these two edges is taken by an adversary.

We call *computation* any infinite sequence of configurations $\gamma_0, \ldots, \gamma_t, \gamma_{t+1}, \ldots$ such that (1) γ_0 is a possible initial configuration and (2) for every instant $t \geq 0$, γ_{t+1} is obtained from γ_t after some robots (at least one) execute a cycle. Any transition γ_t, γ_{t+1} is called a step of the computation. A computation c *terminates* if c contains a terminal configuration.

A *scheduler* is a predicate on computations, that is, a scheduler define a set of *admissible* computations, such that every computation in this set satisfies the scheduler predicate. Here we assume a *distributed fair* scheduler. Distributed means that, at every instant, any non-empty subset of robots can be activated. Fair means that every robot is activated infinitively often during a computation. A particular case of distributed fair scheduler is the *sequential* fair scheduler: at every instant, one robot is activated and every robot is activated infinitively often during a computation. In the following, we call sequential computation any computation that satisfies the sequential fair scheduler predicate.

Problem to be solved We consider the *exploration* problem, where k robots collectively explore a n-sized ring before stopping moving forever. More formally, a protocol $\mathcal{P}$ *deterministically* (resp. *probabilistically*) solves the exploration problem if and only if every computation c of $\mathcal{P}$ starting from a *towerless configuration* satisfies:

1. c terminates in *finite time* (resp. with *expected finite time*).
2. Every node is visited by at least one robot during c.

The previous definition implies that every initial configuration of the system in the problem we consider is *towerless*.

Using probabilistic solutions, termination is not certain, however the overall probability of non-terminating computations is 0.

3 Negative Result

In this section, we show that the exploration problem is impossible to solve in our settings (*i.e.*, oblivious robots, anonymous ring, distributed scheduler, ...) if there is less than four robots, even in a probabilistic manner (Corollary 2). The proof is made in two steps:

- The first step is based on the fact that obliviousness constraints any exploration protocol to construct an implicit memory using the configurations. We show that if the scheduler behaves sequentially, then in any case except one, it is not possible to particularize enough configurations to memorize which nodes have been visited (Theorem 1 and Lemma 5).
- The second step consists in excluding the last case (Theorem 2).

Lemmas 1 to 4 proven below are technical results that lead to Corollary 1. The latter exhibits the minimal size of a subset of particular configurations required to solve the exploration problem.

Definition 1 (MRP) *Let s be a sequence of configurations. The* minimal relevant prefix *of s, noted $\mathcal{MRP}(s)$, is the maximal subsequence of s where no two consecutive configurations are identical.*

Lemma 1 *Let $\mathcal{P}$ be any (probabilistic or deterministic) exploration protocol for k robots in a ring of n nodes. For every sequential computation c of $\mathcal{P}$ that terminates, we have $|\mathcal{MRP}(c)| \geq n - k + 1$.*

Proof. Let c be a sequential computation that terminates. In the initial configuration of c exactly k nodes are already visited because there is at most one robot in each node. So, $n - k$ nodes are dynamically visited before c terminates. As the computation is sequential, the computation contains at least $n - k + 1$ different configurations: the initial one plus one configuration per node to be dynamically visited. Hence, $|\mathcal{MRP}(c)| \geq n - k + 1$. □

Lemma 2 *Let $\mathcal{P}$ be any (probabilistic or deterministic) exploration protocol for k robots in a ring of $n > k$ nodes. For every sequential computation c of $\mathcal{P}$ that terminates, $\mathcal{MRP}(c)$ has at least $n - k + 1$ configurations containing a tower.*

Proof. Assume, by the contradiction, that there is a sequential computation c of $\mathcal{P}$ that terminates and such that $\mathcal{MRP}(c)$ has less than $n - k + 1$ configurations containing a tower.

There exists a suffix c' of c starting from a configuration α without tower followed a suffix s that only contains configurations with a tower. As α is a configuration without tower, c' is an admissible sequential computation of $\mathcal{P}$. Moreover, as c terminates, c' terminates too. Hence, $|\mathcal{MRP}(c')| = n - k + 1$ by Lemma 1 and all robots must be visited before c' reaches its terminal configuration. As a consequence, c' contains exactly $n - k$ steps of the form $\beta\beta'$ with $\beta \neq \beta'$. Now, the first of these steps in c' is a step where one robot moves to a node already occupied by another robot (remember that the computation is sequential and the first step in $\mathcal{MRP}(c')$ is a step from a configuration without tower to a configuration with a tower). Hence, c' contains at most $n - k - 1$ steps where a new node is visited: c' terminates before all robots are visited, a contradiction. □

Lemma 3 *Let $\mathcal{P}$ be any (probabilistic or deterministic) exploration protocol for k robots in a ring of $n > k$ nodes. For every sequential computation c of $\mathcal{P}$ that terminates, $\mathcal{MRP}(c)$ has at least $n - k + 1$ configurations containing a tower of less than k robots.*

Proof. Assume, by the contradiction, that there is a sequential computation c of $\mathcal{P}$ that terminates and such that $\mathcal{MRP}(c)$ has less than $n-k+1$ configurations containing a tower of less than k robots.

There exists a suffix c' of c starting from a configuration α without tower followed a suffix s that only contains configurations with a tower. As α is a configuration without tower, c' is an admissible sequential computation of $\mathcal{P}$. Moreover, as c terminates, c' terminates too. Hence, $\mathcal{MRP}(c')$ is constituted of a configuration with no tower followed by at least $n-k+1$ configurations containing a tower by Lemma 2 and all robots must be visited before c' reaches its terminal configuration.

As the first configuration of c' is without tower, for every configuration α of $\mathcal{MRP}(c')$ with a tower there exists a unique step in $\mathcal{MRP}(c')$ of the form $\alpha'\alpha$ with $\alpha' \neq \alpha$. Now, as c' is sequential, for each of these steps, if α contains a tower of k robots, then no new node is visited during $\alpha' \neq \alpha$. By contradiction assumption, there is less than $n-k+1$ of steps $\beta'\beta$ such that β contains a tower of less than k robots. Moreover, no node is visited during the first of these steps (remember that the computation is sequential and the first of these steps is a step from a configuration without tower to a configuration with a tower). Hence, less that $n+k$ steps allow to dynamically visit new nodes in c' and, as c' is sequential, c' terminates before all robots are visited, a contradiction. □

Lemma 4 *Let $\mathcal{P}$ be any (probabilistic or deterministic) exploration protocol for k robots in a ring of $n > k$ nodes. For every sequential computation c of $\mathcal{P}$ that terminates, $\mathcal{MRP}(c)$ has at least $n-k+1$ configurations containing a tower of less than k robots and any two of them are distinguable.*

Proof. Consider any sequential computation c of $\mathcal{P}$ that terminates.

By Lemma 3, $\mathcal{MRP}(c)$ has x configurations containing a tower of less than k robots where $x \geq n-k+1$.

We first show that (**) *if c contains at least two different configurations that are undistinguable, then there exists a sequential computation c' that terminates and such that $\mathcal{MRP}(c')$ has x' configurations containing a tower of less than k robots where $x' < x$.* Assume that there two different undistinguable configurations γ and γ' in c having a tower of less than k robots. Without loss of generality, assume that γ occurs at time t in c and γ' occurs at time $t' > t$ in c. Consider the two following case:

1. **γ' can be obtained by applying a rotation of i to γ.** Let p be the prefix of c from instant 0 to instant t. Let s be the suffix of c starting at

instant $t' + 1$. Let s' be the sequence obtained by applying a rotation of $-i$ to the configurations of s. As the ring and the robots are anonymous, ps' is an admissible sequential computation that terminates. Moreover, by construction $\mathcal{MRP}(ps')$ has x' configurations containing a tower of less than k robots where $x' < x$. Hence (**) is verified in this case.

2. **γ' can be obtained by applying a rotation of i to the mirror of γ.** We can prove (**) in this case by slightly modifying the proof of the previous case: we have just to apply the rotation of $-i$ to the *mirrors* of the configurations of s.

By (**), if $\mathcal{MRP}(c)$ contains less than $n-k+1$ distinguable configurations with a tower of less than k robots, it is possible to (recursively) construct an admissible computation c' of $\mathcal{P}$ such that $\mathcal{MRP}(c')$ has less than $n-k+1$ configurations containing a tower of less than k robots, a contradiction to Lemma 3. Hence, the lemma holds. □

From Lemma 4, we can deduce the following corollary:

Corollary 1 *Considering any (probabilistic or deterministic) exploration protocol for k robots in a ring of $n > k$ nodes, there exists a subset $\mathcal{S}$ of at least $n-k+1$ configurations such that:*

1. *Any two different configurations in $\mathcal{S}$ are distinguable, and*

2. *In every configuration in $\mathcal{S}$, there is a tower of less than k robots.*

Theorem 1 $\forall k, 0 \leq k < 3, \forall n > k$*, there is no exploration protocol (even probabilistic) of a n-size ring with k robots.*

Proof. First, for $k = 0$, the theorem is trivially verified. Consider then the case $k = 1$ and $k = 2$: with one robot it is impossible to construct a configuration with one tower; with two robots it is impossible to construct a configuration with one tower of less than k robots ($k = 2$). Hence, for $k = 1$ and $k = 2$, the theorem is a direct consequence of Corollary 1. □

Lemma 5 $\forall n > 4$*, there is no exploration protocol (even probabilistic) of a n-size ring with three robots.*

Proof. With three robots, the size of the maximal set of distinguable configurations containing a tower of less than three robots is $\lfloor n/2 \rfloor$. By Corollary 1, we have then the following inequality:

$$\lfloor n/2 \rfloor \geq n - k + 1$$

From this inequality, we can deduce that n must be less of equal than four and we are done. □

From this point on, we know that, assuming $k < 4$, Corollary 1 prevents the existence of any exploration protocol in any case except one: $k = 3$ and $n = 4$ (Theorem 1 and Lemma 5). Actually, assuming that the scheduler is sequential is no sufficient to show the impossibility in this latter case: Indeed, there an exploration protocol for $k = 3$ and $n = 4$ if we assume a sequential scheduler. The protocol works as shown in Figure 1.

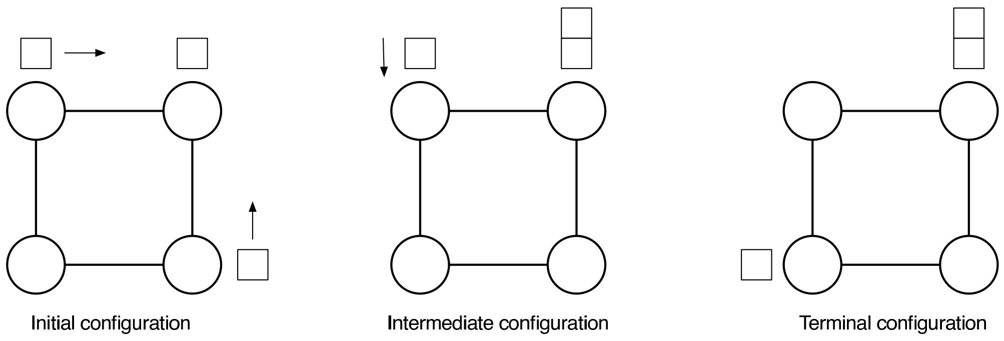

Figure 1: Protocol for $n = 4$ and $k = 3$. (The arrows show the destinations of the robots if they are activated.)

We now show the impossibility in this latter using a (non-sequential) distributed scheduler. This proof is established by enumerating and testing all possible protocols for $k = 3$ and $n = 4$.

Theorem 2 *There is no exploration protocol (even probabilistic) of a n-size ring with three robots for every $n > 3$.*

Proof. Lemma 5 excludes the existence of any exploration protocol for three robots in a ring of $n > 4$ nodes. Hence, to show this theorem, we just have to show that there is no exploration protocol for three robots working in a ring of four nodes.

Assume, by the contradiction, that there exists an exploration protocol $\mathcal{P}$ for three robots in a ring of four nodes. Then, any possible initial configuration is undistinguable with the configuration presented in Figure 2. Moreover, any possible terminal configuration contains a tower and so is undistinguable with one of the three configurations presented in Figure 3.

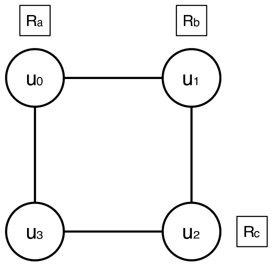

Figure 2: Initial configuration for $n = 4$ and $k = 3$. (The indices are used for notation purposes only.)

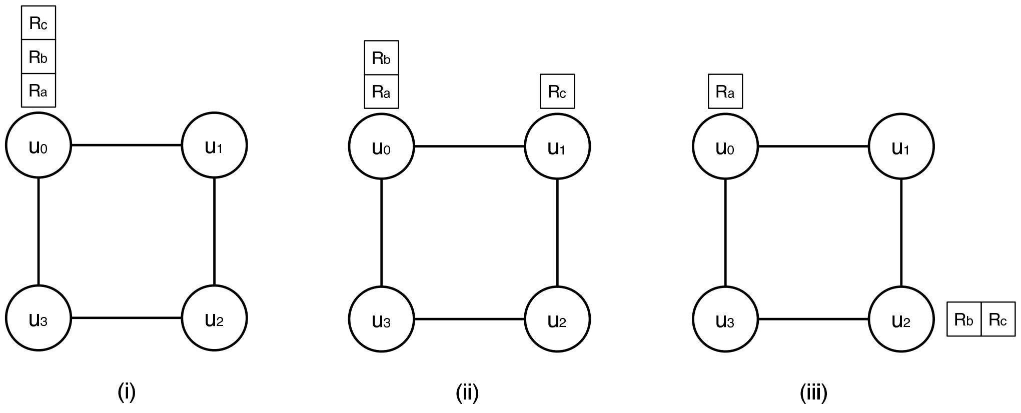

Figure 3: Terminal configurations for $n = 4$ and $k = 3$. (The indices are used for notation purposes only.)

Consider that the system is initially in the configuration of Figure 2. Three cases are possible at instant 0 using $\mathcal{P}$:

- *There is a strictly positive probability that robot R_a (resp. robot R_c) moves to node u_3 if activated by the scheduler.*[1] In this case, assume that the scheduler activates R_a until it moves. The probability that R_a eventually moves is 1 (resp. R_a moves in one step if $\mathcal{P}$ is determistic). Once R_a has moved, R_b has a strictly positive probability to move to node u_0 if activated. Assume then that the scheduler activates R_b until it moves. The probability that R_b eventually moves is 1. Repeating this scheme for R_c and so on, it is possible to construct a distributed fair computation that does not terminate in finite expected time (resp. in finite time, if if $\mathcal{P}$ is determistic), a contradiction.

- *There is a strictly positive probability that robot R_a (resp. robot R_c) moves to node u_1 if activated by the scheduler.* In this case, there an admissible computation where R_a and R_c moves to node u_1 in the first step. At instant 1, the system is in a configuration that is undistinguable with configuration (i) of Figure 3. As node u_3 is still not visited in this case, any configuration that is undistinguable with configuration (i) cannot be terminal. There is also an admissible computation where only R_a moves to node u_1 in the first step. At instant 1, the system is in a configuration that is undistinguable with configuration (ii) of Figure 3. As node u_3 is still not visited in this case, any configuration that is undistinguable with configuration (ii) cannot be terminal. Moreover, assuming that the system reaches a configuration undistinguable from configuration (i) of Figure 3 at instant 1, there is a strictly positive probability that the three robots moves (the configuration is not terminal and all robots have the same view). If they move, the adversary can choose which incident edge they traverse because the configuration is symmetric. Hence, we can obtain a configuration undistinguable with configuration (iii) of Figure 3 and where node u_3 is still not visited. Thus, any configuration that is undistinguable with configuration (iii) cannot be terminal. Hence, no configuration can be terminal, a contradiction.

- *There is a strictly positive probability that robot R_b moves if activated by the scheduler.* Assume that the scheduler activates R_b until it moves. Then, the probability that R_b eventually moves is 1. Once R_b decide to move, the adversary can choose the edge that R_b traverses because the view from R_b is symmetric. Hence, the system can reache the configuration γ: R_a is in node u_0, R_b and R_c and in node u_2. This configuration is undistinguable with

[1]If $\mathcal{P}$ is deterministic the probability is 1 and if activated, R_a moves in one step.

configuration (iii) in Figure 3 and node u_3 is still not visited. Consider the two following cases:

- *The probability that R_a moves, if activated, is 0.* Then, there is a strictly positive probability that R_c (resp. R_b) moves if activated. Assume that the scheduler activates R_a and then R_c until R_c moves. The probability that R_c eventually moves is 1 and as the the view from R_c is symmetric, the adversary can decide which edge R_c will traverse. Assume that the adversary forces R_c to go to node u_1, the system reaches a configuration undistinguable with the initial configuration. Repeating the same scheme infinitively often, we obtain a distributed fair computation that does not terminate in finite expected time, a contradiction.
- *The probability that R_a moves ,if activated, is strictly positive.* Assume that the scheduler activates R_a until it moves. Then, the probability that R_a eventually moves is 1 and as the the view from R_a is symmetric, the adversary can decide which edge R_a will traverse. Assume that R_a moves to node u_1, the system reaches the following configuration: R_a is in node u_1, R_b and R_c are in node u_2, and node u_3 is still not visited. This configuration is undistinguable with configuration (ii) in Figure 3. Consider the two following cases:
  - *The probability that R_c (resp. R_b) moves, if activated, is strictly positive.*
    - *Assume that the destination of R_c, if R_c, is node u_3.* Then, the system reaches a configuration undistinguable from initial configuration. Repeating same the scheme infinitively often, we obtain a distributed fair computation that does not terminate in finite expected time, a contradiction.
    - *Assume that the destination of R_c, if R_c moves, is node u_1.* Then, the destination of R_b, if R_b moves, is node u_1 too. Hence, there is an admissible computation where R_b and R_c move to node u_1. In this case, the system reaches a configuration that is not distinguable from configuration (i) in Figure 3 while node u_3 is still not visited. In this case, no configuration can be terminal, a contradiction.
  - *The probability that R_b (resp. R_c) moves, if activated, is 0.* Then, the probability that R_a moves is strictly positive. Consider the two following cases:

- · *Assume that the destination of R_a, if R_a, is node u_2.* In this case, there is an admissible computation where R_a move to node u_2: the system reaches a configuration that is not distinguable from configuration (i) in Figure 3 while node u_3 is still not visited. In this case, no configuration can be terminal, a contradiction.
- · *Assume that the destination of R_a, if R_a, is node u_0.* Assume that the scheduler activates R_b, R_c, and then R_a until R_a moves. The probability that R_a eventually moves is 1 and we retreive a configuration that is undistinguable with configuration γ. Repeating the same scheme infinitively often, we obtain a fair distributed computation that does not terminate in finite expected time, a contradiction.

In all cases, we obtain a contradiction: there no exploration protocol for three robots in a ring of $n > 4$ nodes and the theorem is proven. □

From Theorems 1 and 2, we can deduce the following corollary:

Corollary 2 $\forall k, 0 \leq k < 4, \forall n > k$, *there is no exploration protocol (even probabilistic) of a n-size ring with k robots.*

4 Positive Result

In this section, we propose a probabilistic exploration protocol for $\boldsymbol{k = 4}$ robots in a ring of $\boldsymbol{n > 8}$ nodes. We first define some useful terms in Subsection 4.1. We then give the general principle of the protocol in Subsection 4.2. Finally, we fully describe and prove the protocol in Subsection 4.3.

4.1 Definitions

Below, we define some terms that characterize the configurations.

We call *segment* any maximal non-empty elementary path of occupied nodes. The *length of a segment* is the number of nodes that compose it. We call *x-segment* any segment of length x. An *isolated node* is a node belonging to a 1-segment.

We call *hole* any maximal non-empty elementary path of free nodes. The *length of a hole* is the number of nodes that compose it. We call *x-hole* any hole of length x. In the hole $h = u_i, \ldots, u_k$ ($k \geq i$) the nodes u_i and u_k are terms as the *extremities* of h. We call *neighbor* of an hole any node that does not belong to the hole but is neighbor of one of its extremities. In this case, we also say that the hole is a *neighboring hole* of the node. By extension, any robot that is located at a neighboring node of a hole is also referred to as a neighbor of the hole.

We call *arrow* a maximal elementary path $u_i, \ldots, u_k$ of length at least four such that (i) u_i and u_k are occupied by one robot, (ii) $\forall j \in [i+1 \ldots k-2]$, u_j is free, and (iii) there is a tower of two robots in u_{k-1}. The node u_i is called the *arrow tail* and the node u_k is called the *arrow head*. The *size* of an arrow is the number of free nodes that compose it, *i.e.*, its the length of the arrow path minus 3. Note that the minimal size of an arrow is 1 and the maximal size is $n-4$. Note also that when there is an arrow in a configuration, the arrow is unique. An arrow is said *primary* if its size is 1. An arrow is said *final* if its size is $n-4$.

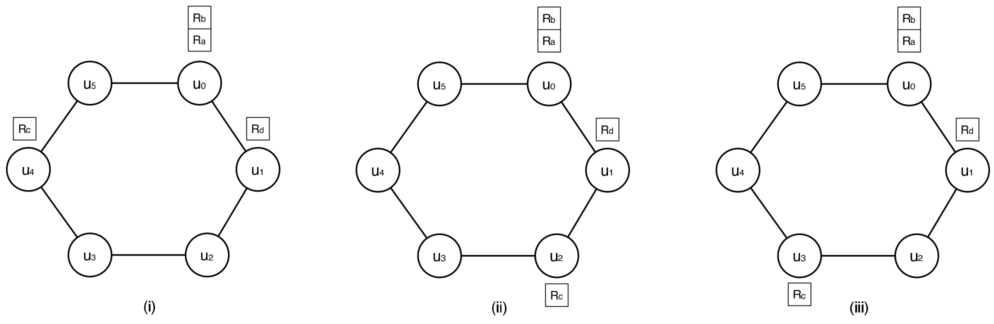

Figure 4: Arrows

Figure 4 illustrates the notion of arrows: In Configuration (i) the arrow is formed by the path u_4, u_5, u_0, u_1; the arrow is primary; the node u_4 is the tail and the node u_1 is the head. In Configuration (ii), there is a final arrow (the path u_2, u_3, u_4, u_5, u_0, u_1). Finally, the size of the arrow in Configuration (iii) (the path u_3, u_4, u_5, u_0, u_1) is 2.

4.2 Overview of the solution

Our protocol (Algorithm 1) proceeds in three distinct phases:

- **Phase I:** Starting from a configuration without tower, the robots move along the ring in such a way that (i) they never form any tower and (2) form a unique segment (a 4-segment) in finite expected time.

- **Phase** II**:** Starting from a configuration with a unique segment, the four robots form an primary arrow in finite expected time. The 4-segment is maintained until the primary arrow is formed.
- **Phase** III**:** Starting from a configuration where the four robots form a primary arrow, the arrow tail moves toward the arrow head in such way that the existence of an arrow is always maintained. The protocol terminates when robots form a final arrow. At the termination, all nodes have been visited.

Note that the protocol we propose is probabilistic. As a matter of fact, as most as possible the robots move deterministically. However, we use randomization to break the symmetry in some cases: When the system is in a symmetric configuration, the scheduler may choose synchronously to activated some processes in such way that the system stays in a symmetric configuration. To break the symmetry despite the choice of the scheduler, we proceed as follows: The activated nodes toss a coin (with a uniform probability) during their Compute phase. If they win the toss, they decide to move, otherwise they decide to stay idle. In this case, we say that the robots **try to move**. Conversely, when a process deterministically decides to move in its Compute phase, we simply say that the process **moves**.

Algorithm 1 The protocol.

```
 1: if the four robots do not form a final arrow then
 2:     if the configuration contains neither an arrow nor a 4-segment then
 3:         Execute Procedure Phase I;
 4:     else
 5:         if the configuration contains a 4-segment then
 6:             Execute Procedure Phase II;
 7:         else      /* the configuration contains an arrow */
 8:             Execute Procedure Phase III;
 9:         end if
10:     end if
11: end if
```

4.3 Detailed description of the solution

4.3.1 Phase I

Phase I is described in Algorithm 2. The aim of this phase is to eventually form a 4-segment without creating any tower during the process. Roughly speaking, in asymmetric configurations, robots moves determiniscally (Lines 3, 8, 22, 26). By contrast, in symmetric configurations, robots moves probabilistically using **Try to**

move (Lines 13 and 18). Note that in all case, we prevent the tower formation by applying the following constraint: a robot can move through a neighboring hole $\mathcal{H}$ only if its length is at least 2 or if the other neighboring robot can move through $\mathcal{H}$.

Algorithm 2 Procedure *Phase* I.

```
1:  if the configuration contains a 3-segment then
2:      if I am the isolated robot then
3:          Move toward the 3-segment through the shortest hole;
4:      end if
5:  else
6:      if the configuration contains a unique 2-segment then      /* Two robots are isolated */
7:          if I am at the closest distance from the 2-segment then
8:              Move toward the 2-segment through the hole having me and an extremity of the 2-segment as neighbors;
9:          end if
10:     else
11:         if the configuration contains (exactly) two 2-segments then
12:             if I am a neighbor of a longuest hole then
13:                 Try to move toward the other 2-segment through my neighboring hole;
14:             end if
15:         else      /* the four robots are isolated */
16:             Let $l_{max}$ be the length of the longuest hole;
17:             if every robot is neighbor of a $l_{max}$-hole then
18:                 Try to move through a neighboring $l_{max}$-hole;
19:             else
20:                 if 3 robots are neighbors of a $l_{max}$-hole then
21:                     if I am neighbor of only one $l_{max}$-hole then
22:                         Move toward the robot that is neighbor of no $l_{max}$-hole through my shortest neighboring hole;
23:                     end if
24:                 else      /* 2 robots are neighbors of the unique $l_{max}$-hole */
25:                     if I am neighbor of the unique $l_{max}$-hole then
26:                         Move through my shortest neighboring hole;
27:                     end if
28:                 end if
29:             end if
30:         end if
31:     end if
32: end if
```

The following lemma (Lemma 6) shows that no tower can by creating during Phase I. The next one (Lemma 7) shows that executing Algorithm 2, a 4-segment is eventually created.

Lemma 6 *If the configuration at instant t contains neither a 4-segment nor a tower, then the configuration at instant $t+1$ contains no tower.*

Proof. Let γ be the configuration at instant t. First, note that the robots executes *Phase* I (Algorithm 2) in γ. Note also that γ satisfies one of the following cases:

- *γ contains a 3-segment.* In this case, only the (unique) isolated robot can move and, if it does, it moves to a free node (see Line 3). Hence, no tower is created at instant $t+1$.
- *γ contains a unique 2-segment.* Two cases are possible:
  - *There is a unique isolated robot $\mathcal{R}$ at the closest distance from the 2-segment.* In this case, only $\mathcal{R}$ can move and, if it does, it moves to free node (see Line 8), so no tower is created at instant $t+1$.
  - *The two isolated robots are at the same distance from the 2-segment.* In this case, the two isolated robots can move but as they follow their shortest path to the 2-segment (see Line 8) and there is no tower in γ, they follow distinct paths and no tower is created at instant $t+1$.

  Hence, in the two subcases no tower is created at instant $t+1$.
- *γ contains two 2-segments.* In this case, as there is four robots and the size of the ring is greater than 8, the size of the longuest hole is at least three. In such a configuration, the only possible moves are the moves where robots move through one of their neighboring holes of length at least two (see Line 13). Hence, all moving robots move to a different free node: no tower is created at instant $t+1$.
- *γ contains four isolated robots.* Let l_{max} be the length of the longuest hole in γ. In this case, as there is four robots and the size of the ring n is greater than 8, $l_{max} \geq 2$. Consider then the following three subcases:
  - *Every robot is neighbor of a l_{max}-hole.* In this case, the configuration is symmetric. Every robot can move in the next step but to a neighboring hole of size at least two (see Line 18). So, all moving robots move to a different free node. Hence, no tower is created at instant $t+1$.
  - *Three robots are neighbors of a l_{max}-hole.* Let $\mathcal{R}$ be the robot that is not neighbor of any l_{max}-hole. In this case, the robots that may move (at most two) move through their neighboring hole having $\mathcal{R}$ as other neighbor (see Line 22). As $\mathcal{R}$ cannot move, no tower is created at instant $t+1$.

- *Two robots, say $\mathcal{R}_1$ and $\mathcal{R}_2$, are neighbors of the unique l_{max}-hole.* In this case, only $\mathcal{R}_1$ and $\mathcal{R}_2$ can move. If $\mathcal{R}_1$ (resp. $\mathcal{R}_2$) moves, then $\mathcal{R}_1$ (resp. $\mathcal{R}_2$) through its neighboring hole having not $\mathcal{R}_2$ (resp. $\mathcal{R}_1$) as other neighbor (see Line 26). So, all moving robots move to a different free node. As a consequence, no tower is created at instant $t+1$.

In all cases, the configuration obtained at instant $t+1$ contains no tower and the lemma holds. □

Lemma 7 *Starting from any initial configuration, the system reaches in finite expected time a configuration containing a* 4*-segment.*

Proof. Any initial configuration contains no tower. If the initial configuration contains a 4-segment, the lemma trivially holds. Consider now the case where the initial configuration contains neither a 4-segment nor a tower.

By Lemma 6, while the system does not reaches a configuration containing a 4-segment, the system remains in configurations containing no tower. For a given n-size ring network, the number of such configuration is *finite*. So, to prove the lemma, we have just to show that from any configuration containing neither a 4-segment nor a tower, there is always a strictly positive probability that the system eventually reaches a configuration containing a 4-segment (despite the choices of the scheduler). To see this, consider a configuration γ containing neither a 4-segment nor a tower and split the study into the following cases:

1. *γ contains a* 3*-segment.* In this case, only the unique isolated robot can move and by the fairness property, it eventually does: it moves toward the 3-segment through the shortest hole (see Line 3). So, until the system reaches a configuration containing a 4-segment, only the isolated robot moves and at each move the length of the shortest hole decreases. Hence, the system reaches a configuration containing a 4-segment *in finite time.*

2. *γ contains a unique* 2*-segment.* Following the same scheme as in the previous case, we can see that the system reaches a configuration containing a 4-segment *in finite time.*

3. *γ contains two* 2*-segments.* In this case, the robots that are neighbors of a longuest hole (at least two) can *try* to move (see Line 13). So, by fairness property, a non-empty set of these robots, say S, is eventually activated by the scheduler. Now, every robot in S decides with a uniform probability to

move or not. So, there a strictly positive probability that only one robot in S decides to move. In this case, we retreive the previous case and we are done.

4. γ *contains four isolated nodes.* Let l_{max} be the length of the longuest hole in γ. Let study the following subcases:

   (a) *Only two robots are neighbors of a l_{max}-hole.* In this case, the two robots that are neighbors of the unique l_{max}-hole can move. So, by fairness property, either one or both of them eventually move through their shortest neighboring hole (see Line 26). After such moves, either (*i*) the system is still in a configuration containing four isolated nodes and where two robots are neighbors of a unique longuest hole but the size of the longuest hole increased, or (*ii*) the system is in a configuration containing a unique 2-segment, or (*iii*) the system is in a configuration containing two 2-segments. Hence, the system reaches in finite time a configuration satisfying (*ii*) or (*iii*), *i.e.*, we eventually retreive the cases 2 or 3, and we are done.

   (b) *Exactly three robots are neighbors of a l_{max}-hole.* Let $\mathcal{R}_0$ be the robot that is not neighbor of a l_{max}-hole. Let $\mathcal{R}_1$ and $\mathcal{R}_2$ be the two robots that are neighbor of exactly one l_{max}-hole. In this case, only $\mathcal{R}_1$ and $\mathcal{R}_2$ can move (see Line 22) and by fairness property at least one of them eventually does. If only one of them moves, then we retreive Subcase 4.(a) or Case 2, and we are done. If both $\mathcal{R}_1$ and $\mathcal{R}_2$ move, then the system reaches (*i*) either a configuration where exactly three robots are neighbors of a longuest hole of length $l_{max} + 1$ or (*ii*) a configuration containing a 3-segment. In Case (*i*), if we repeat the argument, we can see that we eventually retreive Subcase 4.(a), Case 1, or Case 2, and we are done. In Case (*ii*), we directly retreive Case 1 and we are done.

   (c) *The four robots are neighbors of a l_{max}-hole.* In this case, the configuration is symmetric and all robots *try* move (see Line 26). Now, despite the choice of the scheduler, there is a strictly positive probability that only one robot probabilistically decides to move. In this case, the robot moves through one of its neighboring l_{max}-hole of size at least two (to provide the tower creation). As a consequence, we retreive Subcases 4.(a) or 4.(b) and we are done.

Hence, in all cases there is a strictly positive probability that the system eventually reaches a configuration containing a 4-segment from γ and the lemma holds. □

4.3.2 Phase II

Phase II is described in Algorithm 3: Starting from a configuration where there is a 4-segment on nodes $u_i, u_{i+1}, u_{i+2}, u_{i+3}$, the system eventually reaches a configuration where a primary arrow is formed on nodes $u_i, u_{i+1}, u_{i+2}, u_{i+3}$. To that goal, we proceed as follows: Let $\mathcal{R}_1$ and $\mathcal{R}_2$ be the robots located at the nodes u_{i+1} and u_{i+2} of the 4-segment. $\mathcal{R}_1$ and $\mathcal{R}_2$ try to move to u_{i+2} and u_{i+1}, respectively. Eventually only one of these robots moves and we are done, as proven in the two next lemmas.

Algorithm 3 Procedure *Phase* II.

```
1: if I am not located at an extremity of the 4-segment then
2:     Try to move toward my neighboring node that is not an extremity of the 4-segment;
3: end if
```

Lemma 8 *Let γ be a configuration containing a 4-segment $u_i, u_{i+1}, u_{i+2}, u_{i+3}$. If γ is the configuration at instant t, then the configuration at instant $t+1$ is either identical to γ or the configuration containing the primary arrow $u_i, u_{i+1}, u_{i+2}, u_{i+3}$.*

Proof. Let $\mathcal{R}_1$ (resp. $\mathcal{R}_2$) be the robot located at node u_{i+1} (resp. u_{i+2}) in γ. In γ, all robots executes Algorithm 3 (see Algorithm 1). So, from γ, only $\mathcal{R}_1$ and $\mathcal{R}_2$ can move: $\mathcal{R}_1$ can move to node u_{i+2} and $\mathcal{R}_2$ can move to node u_{i+1} (see Algorithm 3). When one or both of these robots, we obtain a configuration containing either a 4-segment or a primary arrow in $u_i, u_{i+1}, u_{i+2}, u_{i+3}$ and the lemma holds. □

Lemma 9 *From a configuration containing a 4-segment, the system reaches a configuration containing a primary arrow in finite expected time.*

Proof. By Lemma 8, we know that starting from a configuration γ containing a 4-segment, the system either remains in the same configuration or reaches a configuration containing a primary arrow. Let $\mathcal{R}_1$ and $\mathcal{R}_2$ be the robots that are not located at the extremity of the 4-segment in γ. Only $\mathcal{R}_1$ and $\mathcal{R}_2$ can (probabilistically) decide to move in γ. Also, by the fairness property, eventually one or both of them are activated. Now, despite the choice of the scheduler, there is a strictly positive probability that only one of them probabilistically decide to move: in this case, the system reaches a configuration containing a primary arrow (see Algorithm 3) and we are done. □

4.3.3 **Phase** III

Phase III is described in Algorithm 4. This phase is fully deterministic: Let $\mathcal{H}$ be the hole between the tail and the head of arrow. The robot located at the arrow tail traverses $\mathcal{H}$. When it is done, the system is in a terminal configuration containing a final arrow: all nodes have been visited as shown is the theorem below.

Algorithm 4 Procedure *Phase* III.

```
1: if I am the arrow tail then
2:     Move toward the arrow head through the hole having me and the arrow head as neighbor;
3: end if
```

Theorem 3 *Algorithm 1 is a probabilistic exploration protocol for* $\boldsymbol{4}$ *robots in a ring of* $\boldsymbol{n > 8}$ *nodes.*

Proof. The proof of the theorem is based on the two following claims:

1. *Any configuration containing a final arrow is terminal.*

   Proof: Immediate, see Line 1 of Algorithm 1.

2. *From a configuration containing a non-final arrow of length x, the system eventually reaches a configuration containing a $x + 1$-arrow.*

   Proof: In such a configuration, only the arrow tail can move. By the fairness property, the robot located at the arrow tail moves *in finite time*: it moves through its neighboring hole having the arrow head as other neighbor (see Algorithm 4). As a consequence, the size of the arrow is incremented to $x + 1$ and we are done.

Using the two previous claims, we now prove the lemma in two step:

- **Termination.** *Any computation of Algorithm 1 terminates in finite expected time.*

  Proof: Immediate from Lemmas 7, 9, Claims 1 and 2.

- **Partial Correctness.** *When a computation of Algorithm 1 terminates, any node has been visited.*

  Proof: By Lemma 7, starting from any initial configuration, the system reaches in finite expected time a configuration containing a 4-segment say

$u_i, u_{i+1}, u_{i+2}, u_{i+3}$. By Lemmas 8 and 9, from this configuration the system reaches in finite expected time a configuration containing an arrow on $u_i, u_{i+1}, u_{i+2}, u_{i+3}$. Hence, when the phase III starts, nodes u_i, u_{i+1}, u_{i+2}, and u_{i+3} are already visited. By Claim 2, the robots executes then Algorithm 4 until the computation terminates. Let $\mathcal{P}$ be the path $u_{i-1}, \ldots, u_{i-n+4}$. By Claim 2, until the computation terminated, only the robot located at the arrow tail can move and it it move following $\mathcal{P}$. Hence, when the computation terminates all nodes of $\mathcal{P}$ have been visited (*i.e.*, nodes u_{i-1}, …, u_{i-n+4}) and, as nodes u_i, u_{i+1}, u_{i+2}, u_{i+3} have also been visited, we are done.

□

5 Conclusion

We provided evidence that for the exploration problem in uniform rings, randomization could shift complexity from $\Theta(\log n)$ to $\Theta(1)$. While applying randomization to other problem instances is an interesting topic for further research, we would like to point out immediate open questions raised by our work:

1. Though we were able to provide a general algorithm for any n (strictly) greater than eight, it seems that ad hoc solutions have to be designed when n is between five and eight (included).
2. Our protocol is optimal with respect to the number of robots. However, the efficiency (in terms of exploring time) is only proved to be finite. Actually computing the convergence time from our proof argument is feasible, but it would be more interesting to study how the number of robots relates to the time complexity of exploration, as it seems natural that more robots will explore the ring faster.

References

[1] H. Ando, Y. Oasa, I. Suzuki, and M. Yamashita. Distributed memoryless point convergence algorithm for mobilerobots with limited visibility. *IEEE Transactions on Robotics and Automation*, 1999.

[2] Yuichi Asahiro, Satoshi Fujita, Ichiro Suzuki, and Masafumi Yamashita. A self-stabilizing marching algorithm for a group of oblivious robots. In Baker et al. [3], pages 125–144.

[3] Theodore P. Baker, Alain Bui, and Sébastien Tixeuil, editors. *Principles of Distributed Systems, 12th International Conference, OPODIS 2008, Luxor, Egypt, December 15-18, 2008. Proceedings*, volume 5401 of *Lecture Notes in Computer Science*. Springer, 2008.

[4] Zohir Bouzid, Maria Gradinariu Potop-Butucaru, and Sébastien Tixeuil. Byzantine-resilient convergence in oblivious robot networks. In *International Conference on Distributed Systems and Networks (ICDCN 2009)*, January 2009.

[5] Y Dieudonné and F Petit. Scatter of weak mobile robots. *Parallel Processing Letters*, 19(1):175–184, 2009.

[6] Yoann Dieudonné, Ouiddad Labbani-Igbida, and Franck Petit. Circle formation of weak mobile robots. *TAAS*, 3(4), 2008.

[7] Paola Flocchini, David Ilcinkas, Andrzej Pelc, and Nicola Santoro. Computing without communicating: Ring exploration by asynchronous oblivious robots. In Eduardo Tovar, Philippas Tsigas, and Hacène Fouchal, editors, *OPODIS*, volume 4878 of *Lecture Notes in Computer Science*, pages 105–118. Springer, 2007.

[8] Paola Flocchini, David Ilcinkas, Andrzej Pelc, and Nicola Santoro. Remembering without memory: Tree exploration by asynchronous oblivious robots. In Alexander A. Shvartsman and Pascal Felber, editors, *SIROCCO*, volume 5058 of *Lecture Notes in Computer Science*, pages 33–47. Springer, 2008.

[9] Paola Flocchini, Giuseppe Prencipe, Nicola Santoro, and Peter Widmayer. Gathering of asynchronous robots with limited visibility. *Theor. Comput. Sci.*, 337(1-3):147–168, 2005.

[10] Paola Flocchini, Giuseppe Prencipe, Nicola Santoro, and Peter Widmayer. Arbitrary pattern formation by asynchronous, anonymous, oblivious robots. *Theor. Comput. Sci.*, 407(1-3):412–447, 2008.

[11] Ralf Klasing, Adrian Kosowski, and Alfredo Navarra. Taking advantage of symmetries: Gathering of asynchronous oblivious robots on a ring. In Baker et al. [3], pages 446–462.

[12] Ralf Klasing, Euripides Markou, and Andrzej Pelc. Gathering asynchronous oblivious mobile robots in a ring. *Theor. Comput. Sci.*, 390(1):27–39, 2008.

[13] Samia Souissi, Xavier Défago, and Masafumi Yamashita. Using eventually consistent compasses to gather memory-less mobile robots with limited visibility. *TAAS*, 4(1), 2009.

[14] Ichiro Suzuki and Masafumi Yamashita. Distributed anonymous mobile robots: Formation of geometric patterns. *SIAM J. Comput.*, 28(4):1347–1363, 1999.

Unité de recherche INRIA Futurs
Parc Club Orsay Université - ZAC des Vignes
4, rue Jacques Monod - 91893 ORSAY Cedex (France)

Unité de recherche INRIA Lorraine : LORIA, Technopôle de Nancy-Brabois - Campus scientifique
615, rue du Jardin Botanique - BP 101 - 54602 Villers-lès-Nancy Cedex (France)
Unité de recherche INRIA Rennes : IRISA, Campus universitaire de Beaulieu - 35042 Rennes Cedex (France)
Unité de recherche INRIA Rhône-Alpes : 655, avenue de l'Europe - 38334 Montbonnot Saint-Ismier (France)
Unité de recherche INRIA Rocquencourt : Domaine de Voluceau - Rocquencourt - BP 105 - 78153 Le Chesnay Cedex (France)
Unité de recherche INRIA Sophia Antipolis : 2004, route des Lucioles - BP 93 - 06902 Sophia Antipolis Cedex (France)

Éditeur
INRIA - Domaine de Voluceau - Rocquencourt, BP 105 - 78153 Le Chesnay Cedex (France)
http://www.inria.fr
ISSN 0249-6399