非同期リングにおけるモバイルエージェント均一配置アルゴリズム

Masahiro Shibata †, Fukuhito Ooshita◇, Hirotsugu Kakugawa†, and Toshimitsu Masuzawa†

†Graduate School of Information Science and Technology, Osaka University
1-5 Yamadaoka, Suita, Osaka, 565-0871, Japan
{m-sibata, kakugawa, masuzawa}@ist.osaka-u.ac.jp
◇Graduate School of Information Science, NAIST
Takayama 8916-5, Ikoma, Nara 630-0192, Japan
f-oosita@is.naist.jp

Abstract. In this paper, we consider the uniform deployment problem of mobile agents in asynchronous uniditrectional ring networks. The uniform deployment problem requires agents to uniformly spread in finite time. In this paper, we consider the uniform deployment problem for the case that agents know neither the number of nodes nor agents. At first we show that, when termination detection is required, there exists no algorithm to solve the uniform deployment problem. For this reason, we consider the relaxed uniform deployment problem that does not require termination detection, and we propose an algorithm to solve the relaxed uniform deployment problem. This algorithm requires $O(k/\ell \log(n/\ell))$ memory per agent, $O(n/\ell)$ time, and $O(kn/\ell)$ total moves, where n is the number of nodes, k is the number of agents, and ℓ is the symmetry of the initial configuration ($\ell \geq 1$). Note that both the algorithms achieve the uniform deployment from any initial configuration, which is a striking difference from the rendezvous problem because the rendezvous problem is not solvable from some initial configurations.
keyword: distributed system, mobile agent, uniform deployment, ring network, token

1 Introduction

A *distributed system* is a system that consists of a set of computers (*nodes*) and communication links. Recently, distributed systems have become large and design of distributed systems has become complicated. As a way to design distributed systems, (mobile) agents have attracted a lot of attention [1]. Agents can traverse the system and process tasks on each node, and hence they can simplify design of distributed systems [2].

Many fundamental problems for cooperation of mobile agents have been studied. For instance, Suzuki et al. [3] considered a *gossip problem*, which requires all agents to share their information. They proposed gossip algorithms on the assumption that agents can communicate with others staying at the same node and can use whiteboard on each node. Another fundamental and the most investigated problem is the *rendezvous problem*, which requires all agents to meet at a single node. The rendezvous problem is considered in rings [4, 5], torus [6], trees [7], and arbitrary networks [8]. Some works assume that agents can use whiteboard on each node, and others assume that agents can use only tokens, which are markers that agents can release on nodes. Chalopin et al. [9] considered *black hole search*, which makes agents locate a particularly dangerous node called black hole. They investigated the minimum resources (the numbers of agents and tokens) necessary for locating all links incident to the black hole.

In this paper, we consider the *uniform deployment* (or *uniform scattering*) problem, which requires all agents to spread uniformly. From a practical point of view, the uniform deployment is useful for the network management. For instance, if agents that can repair faulty nodes are deployed uniformly, such agents can quickly reach and repair faulty nodes after the faults are detected. If agents with database replicas are deployed uniformly, each node can quickly access the database. The uniform deployment is interesting to investigate also from a theoretical point of view. The problem exhibits a striking contrast to the rendezvous: the uniform deployment aims to attain the symmetry of agent locations while the rendezvous aims to break the symmetry. It is well known that the symmetry breaking is difficult (and sometimes impossible) to attain in distributed systems. Consequently, it is interesting to clarify how easily the uniform deployment can be attained compared to the rendezvous.

As related works, Flocchini et al. [10] and Elor et al. [11] considered the uniform deployment problem in ring networks, while Barriere et al. [12] considered it in grid networks. All of them propose uniform deployment algorithms under the assumption that agents are oblivious (or memoryless) but can observe multiple nodes within its visibility range. On the other hand, Mega et al. [13] considered the uniform deployment problem for agents that have memory but cannot observe nodes except for their currently visiting nodes. They assumed

Table 1. Results in each model

	First result in [13]	Second result in [13]	Model 1	Model 2
Knowledge of k	Available	Available	Not Available	Not Available
Termination detection	Required	Required	Required	Not Required
Solvable / Unsolvable	Solvable	Solvable	Unsolvable	Solvable
Agent memory	$O(k \log n)$	$O(n)$	-	$O(k/\ell \log(n/\ell))$
Time complexity	$O(n)$	$O(n \log k)$	-	$O(n/\ell)$
Total moves	$O(kn)$	$O(kn)$	-	$O(kn/\ell)$

n: the number of nodes, k: the number of agents, ℓ: the symmetry of the initial configuration

agents with knowledge of the number of agents and proposed two move-optimal algorithms to solve the uniform deployment problem in unidirectional synchronous ring networks. In addition to the two algorithms, they considered the trade-off between the time complexity and the memory requirement par agent.

In this paper, we consider the uniform deployment problem in unidirectional asynchronous ring networks. Similarly to [13], we consider agents that have memory but cannot observe nodes except for their currently visiting node. Each agent initially has a token and can release it on a visiting nodes. After a token is released at some node, agents cannot remove such a token. Different from [13], we assume that agents have no knowledge of the number of agents or nodes. At first we show that, when termination detection is required, there exists no algorithm to solve the uniform deployment problem. Intuitively, it is due to impossibility of finding k or n when the initial configuration has symmetry: when an agent misestimates these at smaller numbers than actual ones, it prematurely terminates and the uniform deployment cannot be achieved. For this reason, we consider the relaxed uniform deployment problem that does not require termination detection, and we propose an algorithm to solve the relaxed uniform deployment problem. In this algorithm, each agent estimates k and n (possibly at smaller values than actual ones) and behaves based on the estimation. Thus, the efficiency of the algorithm depends on the estimation. To evaluate the efficiency, we introduce the following parameter ℓ to denote the symmetry degree of an initial configuration: we say that an initial configuration has symmetry ℓ when the its distance sequence can be represented as ℓ-times repetition of some non-periodic sequence. For example, an asymmetric initial configuration has symmetry 1, and the symmetry becomes larger for a higher symmetric initial configuration. Note that agents cannot know ℓ but the efficiency depends on it. Using the symmetry parameter ℓ, the efficiency of the algorithm is denoted as follows: this algorithm requires $O(k/\ell \log(n/\ell))$ memory per agent, $O(n/\ell)$ time, and $O(kn/\ell)$ total moves. This result shows a natural but interesting property: the algorithm achieves the uniform deployment more efficiently when the initial configuration has higher symmetry. For an asymmetric initial configuration, this algorithm requires $O(k \log n)$ memory per agent, $O(n)$ time, and $O(kn)$ total moves. However, when ℓ is $\omega(1)$, this algorithm requires $o(k \log n)$ memory per agent, $o(n)$ time, and $o(kn)$ total moves. When ℓ is $\Omega(n)$, this algorithm requires $O(1)$ memory per agent, $O(1)$ time, and $O(k)$ total moves.

Note that all proposed algorithms achieve the uniform deployment from any initial configuration, which is a striking difference from the rendezvous problem because the rendezvous problem is not solvable from some initial configurations. Due to limitation of space, we describe several proofs of lemmas and theorems in the appendix. In Table 1, we compare our contributions with results in [13].[1]

2 Preliminaries

2.1 System model

A *unidirectional ring network* R is defined as 2-tuple $R = (V, E)$, where V is a set of anonymous nodes (i.e, nodes having no IDs) and E is a set of unidirectional links. We denote by $n\,(= |V|)$ the number of nodes. Then, we define $V = \{v_0, v_1, \ldots, v_{n-1}\}$ and $E = \{e_0, e_1, \ldots, e_{n-1}\}\,(e_i = (v_i, v_{(i+1) \bmod n}))$. For simplicity, operations to an index of a node assume calculation under modulo n, that is, $v_{(i+1) \bmod n}$ is simply represented by v_{i+1}. We define the direction from v_i to v_{i+1} as the *forward* direction. In addition, we define the j-th $(j \neq 0)$ forward agent a' of agent a as the agent that exists in the a's forward direction and there are $j-1$ agents between a and a'. Moreover, the *distance* from v_i to v_j $(0 \leq i, j \leq n-1)$ is defined to be $(j - i) \bmod n$.

An agent is a state machine having an *initial state*. Let $A = \{a_0, a_1, \ldots, a_{k-1}\}$ be a set of $k\,(\leq n)$ agents. For simplicity, operations to an index of an agent assume calculation under modulo k. Since the ring is unidirectional, agents staying at v_i can move only to v_{i+1}. We assume that agents are anonymous (i.e., agents have no IDs). In addition, each agent initially has a 1-bit memory called *token* and can release it on a node that it visits. After a token is released at some node, agents cannot remove the token. Note that since agents are anonymous, they cannot recognize the owner of each token. Moreover, we assume that agents can send a message of any size to

[1] The model in [13] can be easily to applied to our paper.

any agent at the same node. We consider two types of agents: agents with knowledge of k and agents with no knowledge of k or n.

Each agent executes the following five operations in an atomic action: 1) The agent reaches a node, say v (or it starts operations at v), 2) the agent receives messages (if any), 3) the agent executes local computation at v, 4) the agent sends a message to agents v (if any) if it decides to send a message, and 5) the agent leaves v if it decides to move. We assume that agents move through a link in a FIFO manner, that is, when agent a_p leaves v_i after agent a_q leaves v_i, a_p reaches v_{i+1} after a_q reaches v_{i+1}. We consider an *asynchronous* system, that is, the time for each agent to perform an operation is finite but unbounded.

A (global) *configuration* C is defined as a 5-tuple $C = (S, T, M, P, Q)$. The first element S is a k-tuple $S = (s_0, s_1, \ldots, s_{k-1})$ representing the agent states where s_i is the state (including the state of holding a token or not) of $a_i\,(0 \leq i \leq k-1)$. The second element T is an n-tuple $T = (t_0, t_1, \ldots, t_{n-1})$ denoting the node states where t_i is the state (i.e., the number of tokens) of $v_i\,(0 \leq i \leq n-1)$. The third element M is a k-tuple $M = (m_0, m_1 \ldots m_{k-1})$, where m_i is a sequence of messages that are sent to a_i and not received by a_i. The remaining elements P and Q represent the positions of agents. The element P is an n-tuple $P = (p_0, p_1, \ldots, p_{n-1})$, where p_i is a set of agents staying at node v_i $(0 \leq i \leq n-1)$. The element Q is an n-tuple $Q = (q_0, q_1, \ldots, q_{n-1})$, where q_i is a sequence of agents residing in the FIFO queue corresponding to link (v_{i-1}, v_i) $(0 \leq i \leq n-1)$. Hence, agents in q_i are those in transit from v_{i-1} to v_i.

We denote by $\mathcal{C}$ the set of all the possible configurations. In *initial configuration* $C_0 \in \mathcal{C}$, all agents are in the initial states and placed at distinct nodes respectively, and no node has any token. In addition, in C_0 the node where agent a stays is called the *home node* of a and denoted by $v_{HOME}(a)$. We assume that in C_0 agent a is in the head of queue q_i if v_i is the home node of a. This assures that agent a starts the algorithm at $v_{HOME}(a)$ before any other agent visits $v_{HOME}(a)$, that is, a is the first agent that takes an action at $v_{HOME}(a)$.

In addition, we define *periodic rings*. For initial configuration C_0, we define the *distance sequence* of agent a_i as $D_i(C_0) = (d^i_0(C_0), \ldots, d^i_{k-1}(C_0))$, where $d^i_j(C_0)$ is the distance from the j-th forward agent of a_i to the $(j+1)$-th forward agent of a_i in C_0. Then, we define the distance sequence of configuration C_0 as the lexicographically minimum sequence among $\{D_i(C_0) | a_i \in A\}$, and we denote it by $D(C_0)$. In addition, let $\mathit{shift}(D, x) = (d_x, d_{x+1}, \ldots, d_{k-1}, d_0, d_1, \ldots, d_{x-1})$ for sequence $D = (d_0, d_1, \ldots, d_{k-1})$. Then, when $D(C_0) = \mathit{shift}(D(C_0), x)$ holds for some x such that $0 < x < k$ holds, we say the ring is *periodic*. Otherwise, we say the ring is *not periodic*.

A *schedule* is an infinite sequence of agents. A schedule $X = \rho_1, \rho_2, \ldots$ is fair if every agent appears in X infinitely often. An infinite sequence of configurations $E = C_0, C_1, \ldots$ is called an *execution* from C_0 if there exists a fair schedule $X = \rho_1, \rho_2, \ldots$ that satisfies the following conditions for each h $(h > 0)$:

- If $\rho_h \in p_i$ holds for some i in a configuration C_h, the states of ρ_h and v_i in C_{h-1} are changed to those in C_h by a local computation of ρ_h. Let $a_j = \rho_h$. If $m_j \neq \emptyset$, all messages in m_j are delivered to a_j and consumed, that is, m_j becomes $\emptyset$. In addition if ρ_h sends a message to some agent a_l at v_i, the message is appended to the tail of sequence m_l. Moreover if ρ_h releases its token at v_i, t_i changes, that is, the number of tokens at v_i increments. After this if ρ_h decides to move to v_{i+1}, ρ_h is removed from p_i and is appended to the tail of sequence q_{i+1}. If ρ_h decides to stay, ρ_h is still in p_i. The other elements in C_{h-1} are the same as those in C_h.

- If ρ_h is at the head of q_i for some i in a configuration C_h, ρ_h moves to v_i, that is, ρ_h is removed from q_i. Then, the states of ρ_h and v_i in C_{h-1} are changed to those in C_h by a local computation of ρ_h. If ρ_h sends a message to some agent a_l at v_i, the message is appended to the tail of sequence m_l. Moreover if ρ_h releases its token at v_i, t_i changes, that is, the number of tokens at v_i increments. After this if ρ_h decides to move to v_{i+1}, ρ_h is appended to the tail of sequence q_{i+1}. If ρ_h decides to stay, ρ_h is inserted in p_i. The other elements in C_{h-1} are the same as those in C_h.

- Otherwise, C_{h-1} is the same as C_h.

2.2 The uniform deployment problem

The uniform deployment problem requires $k\,(\geq 2)$ agents to spread uniformly in the ring, that is, the distance between any two *adjacent agents* should become identical like Fig. 1. Here, we say two agents are adjacent when there exists no agent between them. However, we should consider the case that n is not a multiple of k. In this case, we aim to distribute the agents so that the distance d of two adjacent agents should satisfy $\lfloor n/k \rfloor \leq d \leq \lceil n/k \rceil$.

We consider *the uniform deployment problem with termination detection* and *the uniform deployment problem without termination detection*. At first, we define the uniform deployment problem with termination detection. In this case, *a halt state* is defined as follows: when agent a_i enters a halt state, it terminates the algorithm, that is, a_i neither changes its state nor leaves the current node even if another agent sends a message to a_i. Hence

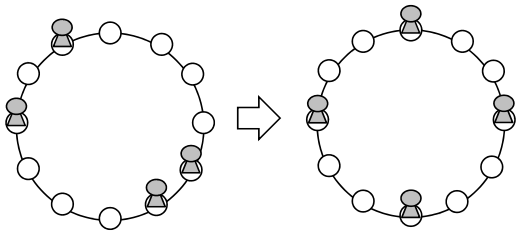


Fig. 1. An example of the uniform deployment ($n = 16, k = 4, d = 3$)

if an agent enters a halt state, it can detect its termination. Now, we define the uniform deployment problem with termination detection as follows.

Definition 1. *An algorithm solves the uniform deployment problem* with *termination detection if any execution satisfies the following conditions.*

- *All agents change their states to halt states in finite time.*
- *When all agents are in the halt states, $q_i = \emptyset$ holds for any $q_i \in Q$ and each distance d of two adjacent agents satisfies $\lfloor n/k \rfloor \leq d \leq \lceil n/k \rceil$.*

Next, we define the uniform deployment problem without termination detection. In this case, *a suspended state* is defined as follows: when agent a_i enters a suspended state, it neither changes its state nor leaves the current node unless another agent sends a message to a_i. If a_i receives a message, it can resume its behavior and leave the current node. The uniform deployment problem without termination detection allows agents to stop in suspended states.

Definition 2. *An algorithm solves the uniform deployment problem* without *termination detection if any execution satisfies the following conditions.*

- *All agents change their states to suspended states in finite time.*
- *When all agents are in the suspended states, $q_i = \emptyset$ holds for any $q_i \in Q$ and each distance d of two adjacent agents satisfies $\lfloor n/k \rfloor \leq d \leq \lceil n/k \rceil$.*

For the uniform deployment problem, we have the following lower bound of total moves.

Theorem 1. *When $k \leq cn$ holds for some constant $c\,(c < 1)$, a lower bound of the total moves to solve the uniform deployment problem (with or without termination detection) is $\Omega(kn)$ even if agents have knowledge of k.*

Proof. We consider the initial configuration such that all agents stay in a quarter part of the ring like Fig. 2. In this case, the ring is divided into four quarter parts, and in the initial configuration, all agents are in the part a (we assume $k \leq n/4$). To achieve the uniform deployment, $k/4$ agents need to move to the part c, the opposite part of a, and each of them must move at least $n/4$ times. Thus the total number of moves is at least $(k/4) \times (n/4) = kn/16$. □

Next, we evaluate the *time complexity* as the time required to achieve the uniform deployment. Since there is no assumption about the period of each action of agents in asynchronous systems, it is impossible to measure the exact time. Instead we consider the *ideal time complexity*, which is defined as the execution time under the following assumptions: 1) The time required for an agent to move from a node to its neighboring node is at most one, and 2) the time required for local computation is ignored (i.e., zero). In the following, we simply use terms "time complexity" and "time" instead of "ideal time complexity". Then, we can show the following theorem similarly to Theorem 1.

Theorem 2. *A lower bound of the time complexity to solve the uniform deployment problem (with or without termination detection) is $\Omega(n)$.*

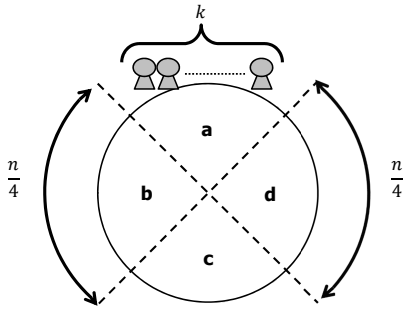


Fig. 2. The initial configuration to derive a lower bound $\Omega(kn)$ of the total moves

3 Impossibility result

In this section, we consider the uniform deployment problem with termination detection. We have the following theorem.

Theorem 3. *Even when the ring is not periodic, there exists no algorithm to solve the uniform deployment problem with termination detection.*

Proof. We prove the theorem by contradiction, that is, we assume that there exists algorithm $\mathcal{A}$ to solve the uniform deployment problem with termination detection.

At first, let us consider n-node ring R and the initial configuration C_0 such that k agents $a_0, a_1, \ldots, a_{k-1}$ exist in this order. Let $V = \{v_0, v_1, \ldots, v_{n-1}\}$ and assume that $d = n/k$ is a positive integer. From hypothesis, there is an execution E_R of $\mathcal{A}$ to solve the uniform deployment problem in R. We define $T(E_R)$ as the length of E_R and denote $E_R = C_0, C_1, \ldots, C_{T(E_R)}$. Note that in $C_{T(E_R)}$, all agents are in the halt states and every distance between two adjacent agents is d.

Next, let us consider a larger ring R' consisting of $2qn + 2n$ nodes, where q is the minimum integer such that $qn \geq T(E_R)$ holds. Let $V' = \{v'_0, v'_1, \ldots, v'_{2qn+2n-1}\}$. We consider the initial configuration C'_0 such that $kq + k$ agents $a'_0, a'_1, \ldots a'_{kq+k-1}$ exist in this order in R'. Then in R', the interval of the uniform deployment is $2d$. In addition, we define the initial position of each agent in R' as follows. Let $v_{f(i)}$ be the node where agent a_i initially stays in R. We assume that $f(i) < f(i+1)$ holds if $i \neq k-1$ and $f(i) > f(i+1)$ holds otherwise. Then, we assume that agent a'_i initially stays at node $v'_{f(i \mod k)+n\cdot\lfloor i/k \rfloor}$. That is, the initial positions for R are repeated from v'_0 to v'_{qn+n-1}, and there is no agent from v'_{qn+n} to $v'_{2qn+2n-1}$. For each node v'_j in R', we define $C_v(v'_j) = v_{j \mod n}$ as the corresponding node of v'_j in R. In the following, we show that each agent $a'_i\,(0 \leq i \leq k-1)$ behaves in the exactly same way as agent a_i in R and a'_i enters a halt state at the same time as a_i. Then, the distance between the two adjacent agents is d, which contradicts that the interval of the uniform deployment in R' is $2d$.

At first, we have the following lemma. We define the *local configuration* of node v as the 2-tuple that consists of the state of v and the states of all agents at v.

Lemma 1. *Let us consider execution $E_{R'} = C'_0, C'_1, \ldots, C'_{T(E_R)}, \ldots$ for ring R'. We define $V'_t = \{v'_t, v'_{t+1}, \ldots, v'_{qn+n-1}\}$. For any $t \leq E(T)$, configuration c'_t satisfies the following condition: for each $v'_j \in V'_t$, the local configuration of v'_j in C'_t is the same as that of $C_v(v'_j)$ in C_t.*

Proof. We prove Lemma 1 by induction on t. For $t = 0$, Lemma 1 holds from the definition of R'. Next, we show that when Lemma 1 holds for $t\,(t < T(E_R))$, Lemma 1 holds for $t + 1$.

From the hypothesis, for each $v'_j \in V'_{t+1}$ the local configurations of v'_{j-1} and v'_j in C'_t are the same as those of $C_v(v'_{j-1})$ and $C_v(v'_j)$ in C_t respectively. Hence, agents at v'_{j-1} and v'_j in C'_t behave in the exactly same way as those at $C_v(v'_{j-1})$ and $C_v(v'_j)$ in C_t. Since only agents at nodes v'_{j-1} and v'_j can change the local configuration of v'_j in unidirectional rings, the local configuration of v'_j in C'_{t+1} is the same as that of $C_v(v'_j)$ in C_{t+1}.

Therefore, we have the lemma. □

From Lemma 1, in $C'_{T(E_R)}$ local configuration of each node in $V^* = \{v'_{qn}, v'_{qn+1}, \ldots, v'_{qn+n-1}\} \subseteq V'_{T(E_R)}$ is the same as that of the corresponding node in $C_{T(E_R)}$. Note that the set of nodes corresponding to nodes in V^* is equal to V, and every agent in V^* also stops in the halt state in configuration $C'_{T(E_R)}$. Hence in $C'_{T(E_R)}$, there exist k agents in the halt states in V^*. Then, the distance between the adjacent agents in V^* is d, which is a contradiction.

Therefore, we have the theorem. □

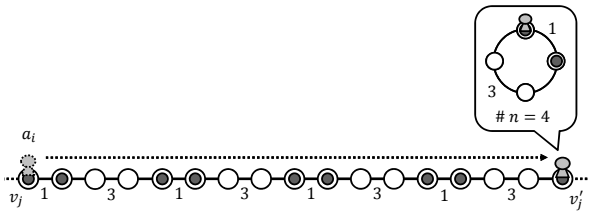


Fig. 3. An example that an agent guesses the number of nodes

4 An algorithm for uniform deployment without termination detection

4.1 Proposed algorithm

In this section, we propose an algorithm to solve the uniform deployment problem without termination detection and we show that this algorithm requires $O(k/\ell \log(n/\ell))$ memory per agent, $O(n/\ell)$ time, and $O(kn/\ell)$ total moves, where ℓ is the symmetry of the initial configuration. At first, we consider the case that the ring is not periodic. After this, we show that our proposed algorithm achieves the uniform deployment also in periodic rings.

4.1.1 Case for non-periodic rings The algorithm consists of three phases: guessing phase, patrolling phase, and deployment phase. In the guessing phase, each agent a_i moves in the ring and guesses the number of nodes. At the end of this phase, we can show that at least one agent guesses the correct number n of nodes. In the patrolling phase, a_i moves in the ring several times depending on its guessed number of nodes. During the movement, if a_i visits the node where another agent exists, a_i sends its guessed number of nodes (with some information) to the agent. By this behavior, we can show that every agent eventually gets the correct number n of nodes and its correct target node. In the deployment phase, a_i moves to its target node and enters a suspended state. After this, if a_i receives a message and recognizes that it mistakenly guesses the number of nodes, a_i decides its new target node from the message and moves there. For simplicity we assume $n = ck$ for some positive integer c in the following description, and this restriction can be removed Appendx A. In addition for sequence Y, we define $Y^1 = Y$ and $Y^{l+1} = Y^l \cdot Y$.

Guessing phase. In the guessing phase, each agent a_i firstly releases its token at its home node. After this, a_i moves in the ring and memorizes the distance *dis* between two adjacent token nodes. Agent a_i continues such a behavior until it completes guessing the number of nodes. Concretely, a_i continues to move until it observes the same distance sequence four times consecutively. After this, a_i considers it travelled four times around the ring and guesses the number of nodes: if a_i observes the same distance sequence four times consecutively when a_i visits $4n'$ nodes, a_i guesses n' as the number of nodes. For example, let us consider Fig. 3. Each number in the figure represents the distance between two adjacent token nodes. Agent a_i moves from node v_j to v_j' and gets the distance sequence $D = (1, 3, 1, 3, 1, 3, 1, 3) = (1, 3)^4$. Then, a_i guesses 4 as the number of nodes. By this behavior, we can show that 1) at least one agent guesses the correct number n of nodes, and 2) if the guessed number n' is not correct, $n' \leq n/2$ holds. The pseudocode is described in Algorithm 1. Variable k' represents the guessed number of agents (tokens) in the ring, and variable *nodes* represents the number of nodes a_i has ever visited.

Patrolling phase. In the patrolling phase, a_i moves $8n'$ times. Finishing this behavior, a_i considers it traveled twelve times around the ring from the beginning about its guessed number of nodes n'. During the movement, a_i may observe some agent a_h staying at some node. In this case, a_h may guess the incorrect number of nodes and prematurely stop moving at an incorrect target node. Hence if a_i observes such an agent, a_i sends $n', k', nodes$, and D to a_h. By this behavior, we can show that every agent eventually gets the correct number n of nodes and its correct target node. The pseudocode is described in Algorithm 2.

Deployment phase. In the deployment phase, agent a_i selects its target node and moves there as follows. Let $D = (d_0, d_1, \ldots, d_{k'-1})^4$ be the distance sequence a_i obtained in the guessing phase, where d_j is the distance from the j-th token node it found to the $(j+1)$-th token node. Note that, $a_i's$ home node $v_{HOME}(a_i)$ is considered as the 0-th token node. In addition, x be the minimum number such that $shift(D, x) = D_{min}$ holds, where D_{min} is the lexicographically minimum distance sequence among $\{shift(D, x) | 0 \leq x \leq k' - 1\}$. Then, a_i selects base node v_{base} where the agent whose distance sequence is D_{min} initially stays. For example in Fig. 4 (a), we assume that each agent finishes the patrolling phase and returns to its home node. Then agents select the node where agent a_0 initially exists as a base node because a_0's distance sequence is the lexicographically minimum. In addition, a_i considers that it is *rank*-th agent ($0 \leq rank \leq k' - 1$) from v_{base} (the agent staying

Algorithm 1 The behavior of agent a_i in the guessing phase

Behavior of Agent a_i

/* guessing phase */
release a token at its home node $v_{HOME}(a_i)$
while $n' = 0$ **do**
  move to the next token node and get the distance *dis* between two token nodes
  $D[i] = dis$, $i = i + 1$
  if $(i \mod 4 = 0) \wedge (\forall x\ (0 \leq x \leq i/4 - 1)$
  $D[x] = D[x + i/4] = D[x + 2 \times i/4] = D[x + 3 \times i/4])$ **then**
    // completing guessing the numbers of nodes and tokens
    $k' = i/4$
    $n' = D[0] + D[1] + \cdots + D[k' - 1]$
    $nodes = 4n'$
  end if
end while
change to the patrolling phase

Algorithm 2 The behavior of agent a_i in the patrolling phase

Behavior of Agent a_i

/* patrolling phase */
while $nodes \neq 12n'$ **do**
  move to the forward node
  $nodes = nodes + 1$
  if there exists another agent a_h **then** send $(n', k', nodes, D[])$ to a_h
end while
change to the deployment phase

at v_{base} is considered as 0-th agent). Let *disBase* be the distance between the current node and the v_{base} (if a_i already stays at v_{base}, we assume that $disBase = n'$). At first, a_i moves *disBase* times and reaches v_{base}. After this, a_i moves to its target node by moving $rank \times n/k$ times and enters a suspended state. In Fig. 4 (b), each agent firstly moves to v_{base}, moves to its target node, and enters a suspended state. When all agents enter suspended states, agents solve the uniform deployment problem.

However, a_i may stay at an incorrect target node with incorrect guessed number of nodes. In this case, a_i eventually receives a message from another agent a_ℓ. Let n'_ℓ, k'_ℓ, $nodes_\ell$, and D_ℓ be the guessed number of nodes, the guessed number of agents, the number of nodes ever visited, and the distance sequence included in a message from a_l respectively. If $n' \leq n'_\ell/2$ holds and there exists t such that $(\forall i\,(0 \leq i \leq 4k' - 1)$ $D[i] = D_\ell[i + t]) \wedge (D_\ell[0] + \cdots D_\ell[t - 1] = nodes_\ell - nodes)$ hold, it means that a_l guesses at least twice number of nodes than a_i and memorizes a_i's whole distance sequence D as a part of D_ℓ. Then, a_i recognizes that it mistakenly guesses the number of nodes and resumes its behavior. Concretely, a_i firstly moves $12n'_\ell - nodes$ times. Note that, we show later that $12n'_\ell - nodes$ is positive. Then, a_i considers it traveled twelve times around the ring from the beginning about new guessed number of nodes n'_ℓ. After this, it decides the new base node and its new target node from n'_ℓ, k'_ℓ, $nodes_\ell$ and D_ℓ, moves to its new target node as mentioned before, and enters a suspended state again. When all agents enter suspended states, agents solve the uniform deployment problem. The pseudocode is described in Algorithm 3.

An example As an example, let us consider the ring like Fig. 5. This ring is not periodic but has some *periodic subsequence*, that is, some agent observes 4-times repeated subsequence before it travels once around the ring. In such a ring, some agent mistakenly guesses the number of nodes and enters a suspended state at an incorrect target node. However in this case, we can show that at least one agent a_i guesses the correct number n of nodes and informs prematurely suspending agents of n during the patrolling phase. Let us consider the behavior of agents a_1 and a_2. For simplicity, we assume that they behave in a synchronous manner. In the guessing phase, agent a_2 gets the distance sequence $D = (1, 3, 1, 3, 1, 3, 1, 3) = (1, 3)^4$ and guesses 4 as the number of nodes, which is incorrect (Fig. 5 (a) to Fig. 5 (b)). After this a_2 executes the patrolling and deployment phases, and enters a suspended state at incorrect target node v'_j (Fig. 5 (b) to Fig. 5 (c)). On the other hand, agent a_1 is still in the guessing phase. When a_1 observes $D = (11, 1, 3, 1, 3, 1, 3, 1, 3)^4$, it completes the guessing phase and guesses the correct number of nodes 27. After this in the patrolling phase, a_1 observes a_2 at v'_j, sends its guessed number of nodes with other information to a_2 (Fig. 5 (c) to Fig. 5 (d)), and moves to its target node. When a_2 receives the message from a_1, it recognizes that it mistakenly guesses the number of nodes and resumes its behavior.

In the following, we show that every agents eventually gets the correct number n of nodes and its correct target node. To show this, we use the following lemmas.

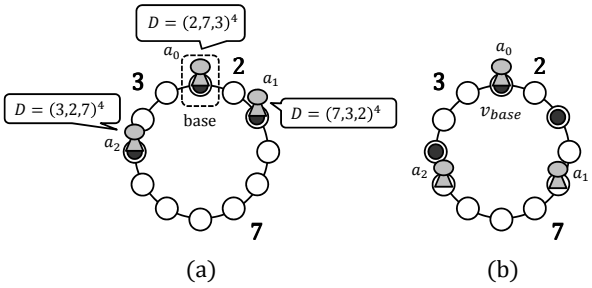


Fig. 4. An example of the deployment phase ($n = 12, k = 3, d = 4$)

Algorithm 3 The behavior of agent a_i in the deployment phase

Behavior of Agent a_i

1: /* deployment phase */
2: let D_{min} be the lexicographically minimum sequence among $\{shift(D, x)|0 \leq x \leq k' - 1\}$
3: $rank = \min\{x \geq 0|shift(D, x) = D_{min}\}$
4: $disBase = D[0] + D[1] + \cdots + D[k - 1 - rank]$
5: move $disBase$ times
6: $nodes = nodes + disBase$
7: move $rank \times n/k$ times
8: $nodes = nodes + rank \times n/k$
9: change its state to a suspended state
10:
11: /* behavior in the suspended state */
12: wait at the current node until a_i receives $(n'_\ell, k'_\ell, nodes_\ell, D_\ell[\,])$ from some agent a_ℓ
13: **if** $(n' \leq n'_\ell/2) \wedge$ (there exists t such that $(\forall i\,(0 \leq i \leq 4k' - 1)$ $D[i] = D_\ell[i + t]) \wedge (D_\ell[0] + \cdots D_\ell[t - 1] = nodes_\ell - nodes)$ hold) **then**
14: // a_i recognizes that it misunderstands the number of nodes
15: $n' = n'_\ell$, $k' = k'_\ell$, $D[\,] = shift(D_\ell[\,], t)$
16: move $12n' - nodes$ times
17: $nodes = 12n'$
18: go to line 2
19: **end if**

Lemma 2. [14] *Consider an p-length sequence $A = a_0, \ldots, a_{p-1}$ and an p'-length sequence $B = b_0, \ldots, b_{p'-1}$ such that $p' < p$ holds. If B^3 is the prefix of A^3, either $p' \leq p/2$ holds or B is periodic.*

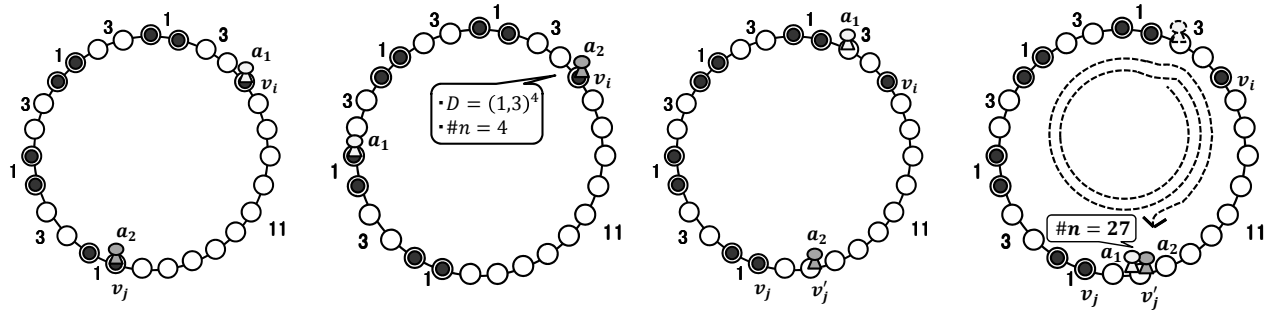
Lemma 3. *If agent a_ℓ guesses the incorrect number of nodes n_ℓ (i.e., $n_\ell \neq n$ holds), $n_\ell \leq n/2$ holds.*

Proof. Let $k_\ell\,(< k)$ be the number of agents (tokens) guessed by a_ℓ. Since a_ℓ observes $4k_\ell$ tokens in the guessing phase, it stores the same distance sequence $(D[0], \ldots, D[k_\ell - 1])$ four times, that is, $(D[0], \ldots, D[4k_\ell - 1]) = (D[0], \ldots, D[k_\ell - 1])^4$ holds. Then, $n_\ell = D[0] + \cdots + D[k_\ell - 1]$ holds. On the other hand since the number of tokens in the ring is $k > k_\ell$, sequence $(D[0], \ldots, D[k_\ell - 1])^4$ is the prefix of $(D[0], \ldots, D[k - 1])^4$. Note that, $n = D[0] + \cdots + D[k - 1]$ holds. Then from Lemma 2, $(D[0], \ldots, D[k_\ell - 1])$ is periodic or $k_\ell \leq k/2$ holds. If $D([0], \ldots, D[k_\ell - 1])$ is periodic, there exists $k'_\ell < k_\ell$ such that $(D[0], \ldots, D[4k'_\ell - 1]) = (D[0], \ldots, D[k'_\ell - 1])^4$ holds. This is a contradiction because a_ℓ should guess n_ℓ as the number of nodes. Hence, $k_\ell \leq k/2$ holds. Then since $(D[0], \ldots, D[k_\ell - 1])$ is the prefix of $(D[0], \ldots, D[k - 1])$, $(D[0], \ldots, D[k - 1]) = (D[0], \ldots, D[k_\ell - 1], D[0], \ldots, D[k_\ell - 1], D[2k_\ell], D[2k_\ell + 1], \ldots)$ holds. Thus, $(D[0] + \cdots + D[k_\ell - 1]) \leq (D[0] + \cdots + D[k - 1])/2$ holds, that is, $n_\ell \leq n/2$ holds. Therefore, we have the lemma. □

Then, we have the following lemmas.

Lemma 4. *If ring R is not periodic, at least one agent guesses the correct number n of nodes and gets distance sequence D of the initial configuration in R.*

Proof. We show that at least one agent guesses the correct number n of nodes. Then from Algorithm 1 and 3, the agent clearly gets the distance sequence D for the initial concifgration in R. We prove the lemma by contradiction, that is, we assume that the number of nodes guessed by each agent is less than n. Without loss of generality, we assume that in the initial configuration agents $a_0, a_1, \ldots, a_{k-1}$ exist in this order. We define



(a) (b) (c) (d)

Fig. 5. An example in the ring having some periodic subsequence ($n = 27, k = 9, d = 3$)

n_i as the number of nodes guessed by a_i and D_i as the distance sequence observed by a_i. In addition, let S_i be the distance sequence such that $D_i = S_i^4$ holds.

Let a_m be the agent that guesses the maximum number of nodes $n_m\,(< n)$ among all agents, and let $\ell = |S_m|\,(< k)$. We assume that the distance sequence a_m observes in Algorithm 1 is D_m=$(d_0^m, \ldots, d_{\ell-1}^m, d_\ell^m, \ldots, d_{2\ell-1}^m, d_{2\ell}^m, \ldots, l_{3\ell-1}^m, l_{3\ell}^m, \ldots d_{4\ell-1}^m)$=$(d_0^m, \ldots, d_{\ell-1}^m)^4 = S_m^4$. Note that, $S_m = (d_0^m, \ldots, d_{\ell-1}^m)$ is not periodic and $\forall j\ (0 \le j \le \ell - 1)\ d_j^m = d_{j+\ell}^m = d_{j+2\ell}^m = d_{j+3\ell}^m$ holds.

Next, let us consider the agent $a_{m+\ell}$. Then, either $n_{m+\ell} < n_m$ or $n_{m+\ell} = n_m$ holds because n_m is the maximum. We show that $n_{m+\ell} = n_m$ always holds by contradiction, that is, we assume that $n_{m+\ell} < n_m$ holds. Then, $|S_{m+\ell}| < |S_m|$ clearly holds. Consequently, $S_{m+\ell}^3$ is the prefix of S_m^3 because $a_{m+\ell}$ gets the distance sequence $(d_\ell^m, \ldots, d_{2\ell-1}^m) = S_m$ when it observes ℓ tokens. Then from Lemma 2, either $|S_{m+\ell}| \le |S_m|/2$ holds or $S_{m+\ell}$ is periodic. If $|S_{m+\ell}| \le |S_m|/2$ holds, agent a_m observes $S_{m+\ell}^4$ before observing S_m^4 because $(d_0^m, \ldots, d_{2\ell-1}^m) = (d_\ell^m, \ldots, d_{3\ell-1}^m)$ contains $S_{m+\ell}^4$ as its prefix. Consequently, a_m guesses $n_{m+\ell} < n_m$ as the number of nodes, which is a contradiction. If $S_{m+\ell}$ is periodic, $S_{m+\ell} = (S'_{m+\ell})^t$ holds for some distance sequence $S'_{m+\ell}$ and some positive integer t ($S'_{m+\ell}$ is not periodic and $|S'_{m+\ell}| \le |S_{m+\ell}|/2$ holds). Hence, a_m observes $(S'_{m+\ell})^4$ before observing S_m^4 and the number of nodes a_m guesses is less than n_m, which is also a contradiction. Therefore, $n_{m+\ell} = n_m$ holds.

Let $m(i) = m + i\ell$ and $A_m = \{a_{m(i)} | \, i \ge 0\}$. As mentioned above, $n_m = n_{m+\ell}$ and $S_{m(0)} = S_{m(1)} = S_m$ hold. In addition, $a_{m(1)}$ observes the same distance sequence of length $4|S_m|$ as $a_{m(0)}$. Hence recursively, $a_{m(i+1)}$ observes the same distance sequence of length $4|S_m|$ as $a_{m(i)}$ and consequently each agent in A_m observes S_m as the first ℓ consecutive distances. When k is divided by ℓ, since every agent $a_{m(i)}$ observes S_m as the first ℓ consecutive distances and $\ell < k$ holds, the ring is periodic, which is a contradiction. In the following, we consider the case that k is not divided by ℓ and show that $S_{m(0)}(= S_m)$ is periodic in this case. When k is not divided by ℓ, $k = \alpha\ell + \beta\,(0 < \beta < \ell)$ holds for some integers α and β. Let $m(\alpha) = m + \alpha\ell$. Then, the prefix of $S_{m(0)}$ is identical to the suffix of $S_{m(\alpha)}$ because the trajectories of $a_{m(0)}$ and $a_{m(\alpha)}$ include the same part of the ring. We assume that t elements are overlapped, that is, $(d_0^{m(0)}, \ldots, d_{t-1}^{m(0)}) = (d_{\ell-t}^{m(\alpha)}, \ldots, d_{\ell-1}^{m(\alpha)})$ holds. In addition since $S_{m(0)} = S_{m(\alpha)}$ holds, $(d_0^{m(0)}, \ldots, d_{\ell-1}^{m(0)}) = (d_0^{m(\alpha)}, \ldots, d_{\ell-1}^{m(\alpha)})$ holds. If $t > \ell/2$ holds, $(d_t^{m(0)}, \ldots, d_{\ell-1}^{m(0)}) = (d_0^{m(\alpha)}, \ldots, d_{\ell-t-1}^{m(\alpha)})$ holds. Hence, $shift(S_{m(0)}, t) = S_{m(\alpha)} = S_{m(0)}$ holds. If $t \le \ell/2$ holds, $(d_{\ell-t}^{m(0)}, \ldots, d_{\ell-1}^{m(0)}) = (d_{\ell-t}^{m(\alpha)}, \ldots, d_{\ell-1}^{m(\alpha)}) = (d_0^{m(0)}, \ldots, d_{t-1}^{m(0)}) = (d_0^{m(\alpha)}, \ldots, d_{t-1}^{m(0)})$ and $(d_t^{m(0)}, \ldots, d_{\ell-t-1}^{m(0)}) = (d_t^{m(\alpha)}, \ldots, d_{\ell-t-1}^{m(\alpha)})$ hold. Thus, $(d_t^{m(0)}, \ldots, d_{\ell-1}^{m(0)}) = (d_0^{m(\alpha)}, \ldots, d_{\ell-t-1}^{m(\alpha)})$ holds. Consequently, $shift(S_{m(0)}, t) = S_{m(\alpha)} = S_{m(0)}$ holds. Therefore, $S_{m(0)}$ is periodic since $0 < t < \ell$ holds. However, this contradicts the assumption that $S_{m(0)}\,(= S_m)$ is not periodic.

Therefore, we have the lemma. □

Lemma 5. *If ring R is not periodic, every agent eventually gets the correct number n of nodes and distance sequence D of the initial configuration in R.*

Proof. We show that all agents eventually get the correct number n of nodes. Then from Algorithms 1 to 3, all agents can clearly get distance sequence D of the initial configuration in R. We prove the lemma by contradiction, that is, we assume that when all agents are in the suspended states, there exists at least one agent a_h whose guessed number of nodes n' is less than n. Then from Lemma 3, $n' \le n/2$ holds. On the other hand from Lemma 4, at least one agent a_c guesses the correct number n of nodes. In the following we show that a_c observes a_h during the patrolling phase and sends its guessed number of nodes n to a_h, which contradicts the assumption of $n' < n$.

At first, let us consider the number of nodes a_h visits. Let n_1 be the number of nodes a_h guesses in the guessing phase. From Algorithms 1 to 3, a_h moves at most $14n_1$ times by the time a_h enters a suspended state for the first time. After this, we assume that a_h receives messages and updates its guessed number of nodes to $n_2, n_3, \ldots, n_l = n'$ in this order. When a_h updates it guessed number of node to n_2, a_h's total moves at that point (i.e, *nodes*) is at most $7n_2$ since $n_1 \leq n_2/2$ holds. Hence, $12n_2 - nodes$ is clearly positive. Then, a_h firstly moves in the ring until its total moves becomes $12n_2$ by moving $12n_2 - nodes$ times. After this, a_h moves to a new target node and enters a suspended state again. This requires at most $14n_2$ total moves. Then since $n_3 \leq n_2/2$ holds from Algorithm 3, *nodes* is at most $7n_3$ and $12n_3 - nodes$ is clearly positive. Thus recursively, we can show that $12n_i - nodes$ is always positive ($2 \leq i \leq l$) and a_h's total moves unless it does not get the correct number n of nodes is at most $14n' \leq 7n$. On the other hand, agent a_c moves $8n$ times in the patrolling phase. Thus, a_c clearly observes a_h during the patrolling phase and sends its guessed number n of nodes to a_h, which is a contradiction.

Therefore, we have the lemma. □

Finally, we have the following lemma.

Lemma 6. *When ring R is not periodic, agents solve the uniform deployment problem without termination detection.*

Proof. From Lemma 5, all agents eventually get the corect number n of nodes and distance sequence D for the initital configuration in R. Then, each agent can compute its correct target node from D and move there. Thus, we have the lemma. □

4.1.2 Case for periodic rings Next, let us consider the case that the ring is periodic. Let R' be a periodic ring and D' be the distance sequence of the initial configuration in R'. We say R' is a (N, ℓ)-node ring if there exists a non-periodic distance sequence D such that $D' = D^\ell$ holds and the total sum of elements of D is N. Then, $n = N\ell$ holds and ℓ is equivalent to the symmetry of the initial configuration in R'. We call the ring R with the distance sequence D the *fundamental ring of* R' (e.g. Fig. 6). Note that a non-periodic ring can be denoted by a $(n, 1)$-node ring. In addition for each agent a_i in R, there exist ℓ agents in R' such that the distance sequence of each agent is ℓ-times repetition of the distance sequence of a_i. We say such agents in R' are *corresponding agents* of agent a_i in R and denote by a_i^j ($0 \leq j \leq \ell - 1$). We assume that agents $a_i^0, a_i^1, \ldots, a_i^{\ell-1}$ are deployed in this order and operations to an above index of a_i^j assume calculation under modulo ℓ. Then, the distance from a_i^j to a_i^{j+1} is N. In this case, all agents eventually guess the incorrect number $N = n/\ell$ of nodes, but we can show that agents can solve the uniform deployment problem similarly to in R. Concretely from Algorithms in Section 4.1, each agent moves to its target node after considering, based on the guessed number N of nodes, it traveled twelve times around the ring. This means that each agent stays at its target node during its twelfth or thirteenth circulations in the ring of the guessed size N, which guarantees that when all agents are in the suspended states, no agents stay at the same node and they can achieve the uniform deployment. For example, let us consider rings in Fig. 6. Ring R' is the (6,2)-node periodic ring and R is the fundamental ring of R'. In R, each agent guesses the correct number 6 of nodes in the guessing phase and moves to its correct target node (Fig. 6 (a)). On the other hand in R', each agent also guesses the number 6 of nodes, which is incorrect (Fig. 6 (b)). By Algorithms 1 to 3, each agent moves to its target node after considering, based on the guessed size 6, it travelled twelve times around the ring, that is, after each agent moves 72 times (actually, each agent travelled six times around ring R'). This guarantees that when all agents are in the suspended states, no agents stay at the same node and they can achieve the uniform deployment (Fig. 6 (c)).

Now, we have the following lemmas, which can be proved similarly to the case of non-periodic rings.

Lemma 7. *Let R' be a (N, ℓ)-node periodic ring and R be the fundamental ring of R'. Let a_i in R be the agent guessing the number N of nodes in the guessing phase. Then in R', agent a_i^j ($0 \leq j \leq \ell - 1$) corresponding to a_i also guesses the number N of nodes.*

Proof. From the definition of R', a_i^j observes the same distance sequence as that of a_i. In addition since agents have no knowledge of k or n, agents determine their guessed number of nodes depending only on the distance sequence they observe. Thus, a_i^j guesses the same number of nodes as that of a_i. □

Lemma 8. *Let R' be a (N, ℓ)-node periodic ring and R be the fundamental ring of R'. Then in R', every agent eventually gets the number N of nodes and distance sequence D of the initial configuration in R.*

Proof. We show that all agents eventually get the number n of nodes. Then from Algorithms 1 to 3, all agents can clearly get distance sequence D of the initial configuration in R. We prove the lemma by contradiction, that is, we assume that when all agents are in the suspended states, there exists at least one agent a_h whose guessed number of nodes n' is less than N. On the other hand from Lemma 7, there exists agent a_c^j ($0 \leq j \leq \ell - 1$)

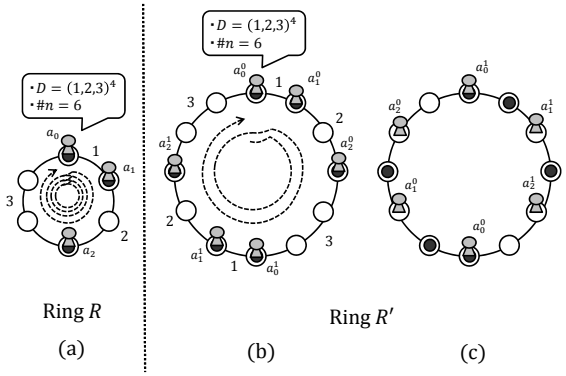


Fig. 6. An example for the periodic ring

guessing the number N of nodes in the guessing phase. Let $A_c = \{a_c^0, a_c^1 \ldots, a_c^{\ell-1}\}$. In the following, we show that some agent in A_c observes a_h during the patrolling phase and sends its guessed number N of nodes to a_h, which contradicts the assumption of $n' < N$.

At first, let us consider the number of nodes a_h visits. Similarly to the case for non-periodic rings, when a_h updates its guessed number of nodes from n'' to n', it firstly moves in the ring until its total moves becomes $12n'$ by moving $12n' - \mathit{nodes}$ times. After this, a_h moves to a new target node and enters a suspended state again. This requires at most $14n'$ total moves. Hence unless a_h does not get the number n of nodes, its total moves is at most $14n' \leq 7N$.

On the other hand from Lemma 7, there exists agent a_c^j in A_c such that it guesses N as the number of nodes and the distance from $v_{HOME}(a_c^j)$ to $v_{HOME}(a_h)$ is less than N. Remind that, $v_{HOME}(a)$ is the home node of agent a. Then, let us consider the behavior of agent a_c^{j-4}. Agent a_c^{j-4} firstly moves $4N$ times and finishes the guessing phase at node $v_{HOME}(a_c^j)$. After this, a_c^{j-4} moves $8N$ times from $v_{HOME}(a_c^j)$ in the patrolling phase. On the other hand, a_h moves at most $7N$ times from $v_{HOME}(a_h)$. Since the distance from $v_{HOME}(a_c^j)$ to $v_{HOME}(a_h)$ is less than N, a_c^{j-4} observes a_h during the patrolling phase and sends the number N of nodes to a_h, which is a contradiction.

Therefore, we have the lemma. □

Lemma 9. *Even when ring R' is periodic, agents solve the uniform deployment problem without termination detection.*

Proof. From Lemma 8, all agents eventually get the number N of nodes and distance sequence D of the initial configuration in R, where R is the fundamental ring of R'. From Algorithm 3 when agent a_i^j gets the number N of nodes it firstly moves in the ring until its total moves becomes $12N$. Then, a_i^j is at $v_{HOME}(a_i^{j+12})$. After this, a_i^j computes its target node from D and moves there, which requries at most $2N$ moves. Hence, a_i^j eventually stays between $v_{HOME}(a_i^{j+12})$ and $v_{HOME}(a_i^{j+14})$. This mean that letting v_{base} (resp., v'_{base}) be the base node exsiting between $v_{HOME}(a_i^{j+12})$ and $v_{HOME}(a_i^{j+13})$ (resp., $v_{HOME}(a_i^{j+13})$ and $v_{HOME}(a_i^{j+14})$) a_i^j eventually stays between v_{base} and v'_{base}. Moerover, it crarly holds total moves of each of a_i^j $(0 \leq j \leq \ell - 1)$ are the same. Thus when all agents are in the supended states, no agents stay at the same node and agents can achive the uniform deployment.

Therefore, we have the lemma. □

Finally, we have the following theorem for (N, ℓ)-node rings.

Theorem 4. *For agents with no knowledge of k or n, the proposed algorithm solves the uniform deployment problem without termination detection. This algorithm requires $O(k/\ell \log(n/\ell))$ memory per agent, $O(n/\ell)$ time, and $O(kn/\ell)$ total moves.*

Proof. From Lemmas 6 and 9, agents solve the uniform deployment problem. In the following, we analyze complexity measures.

At first, we evaluate the memory requirement per agent. Each agent eventually gets the distance sequence $D = (d_0, d_1, \ldots, d_{(4k/\ell)-1})$. Since each d_i is at most n/ℓ, this sequence requires $O(k/\ell \log(n/\ell))$ memory. Moreover, the other variables require $O(\log(n/\ell))$ bit memory. Therefore, the memory requirement per agent is $O(k/\ell \log(n/\ell))$.

Next, we analyze the time complexity. Let $A_{correct}$ be the set of agents that guess the number $n/\ell\,(= N)$ of nodes in the guessing phase. Each agent $a_c \in A_{correct}$ moves its correct target node without stopping on

the way, which requires at most $14n/\ell$ unit times. In addition from the proof of Lemmas 5 and 8, each agent $a_h \notin A_{correct}$ gets the number n/ℓ of nodes within $12n/\ell$ unit times. After this, a_h requires at most $14n/\ell$ unit times to moves to its correct target node. Thus, the time complexity is $O(n/\ell)$.

At last, we analyze the total number of agent moves. Each agentrequires at most $14n/\ell$ moves to move to its target node. Thus, the total number of agent moves is $O(kn/\ell)$. □

5 Conclusion

In this paper, we considered the uniform deployment problem of mobile agents in asynchronous unidirectional ring networks. At first we showed that, when termination detection is required, there exists no algorithm to solve the uniform deployment problem. Next, we proposed an algorithm to solve the uniform deployment problem without termination detection in non-periodic rings. This algorithm requires $O(k/\ell \log(n/\ell))$ memory per agent, $O(n/\ell)$ time, and $O(kn/\ell)$ total moves. As a future work, we want to consider the lower bound of memory requirement per agent. We conjecture that it is $\Omega(\log n)$, and if the conjecture is correct, we can show that the first algorithm is asymptotically optimal in terms of memory requirement.

References

1. D. Kotz S. R. Gray, G. Cybenko, A.R. Peterson, and D. Rus. D'agents: Applications and performance of a mobile-agent system. *Software: Practice and Experience*, 32(6):543–573, 2002.
2. D.B. Lange and M. Oshima. Seven good reasons for mobile agents. *Communications of the ACM*, 42(3):88–89, 1999.
3. T. Suzuki, T. Izumi, F. Ooshita, H. Kakugawa, and T. Masuzawa. Move-optimal gossiping among mobile agents, Theoretical Computer Science. 393(1):90–101, 2008.
4. E. Kranakis, D. Krozanc, and E. Markou. The mobile agent rendezvous problem in the ring, Synthesis Lectures on Distributed Computing Theory, Vol. 1. pages 1–122, 2010.
5. P. Flocchini, E. Kranakis, D. Krizanc, N. Santoro, and C. Sawchuk. Multiple mobile agent rendezvous in a ring, LATIN, LNCS, Vol. 2976. pages 599–608, 2004.
6. E. Kranakis, D. Krizanc, and E. Markou. Mobile agent rendezvous in a synchronous torus, LATIN LNCS, Vol. 3887. pages 653–664, 2006.
7. P. Fraigniaud and A. Pelc. Deterministic rendezvous in trees with little memory, disc, LNCS, Vol. 6950. pages 242–256, 2008.
8. A.Kosowski J. Czyzowicz and A. Pelc. How to meet when you forget: Log-space rendezvous in arbitrary graphs. *DISC*, 25(2):165–178, 2012.
9. A. Labourel J. Chalopin, S. Das and E. Markou. Tight bounds for scattered black hole search in a ring. In *SIROCCO*, pages 186–197. Springer, 2011.
10. P. Flocchini, G. Prencipe, and N. Santoro. Self-deployment of mobile sensors on a ring. *Theoretical Computer Science*, 402(1):67–80, 2008.
11. E. Yotam and B. M. Alfred. Uniform multi-agent deployment on a ring. *Theoretical Computer Science*, 412(8):783–795, 2011.
12. E. Mesa-Barrameda L. Barriere, P. Flocchini and N. Santoro. Uniform scattering of autonomous mobile robots in a grid. *International Journal of Foundations of Computer Science*, 22(03):679–697, 2011.
13. T. Mega, F.Ooshita, H. Kakugawa, and T. Masuzawa. Algorithms for uniform deployment of mobile agents on synchronous rings. *COMP*, 112(24):9–16, 2012.
14. S. Kawai, F. Ooshita, H. Kakugawa, and T. Masuzawa. Randomized rendezvous of mobile agents in anonymous unidirectional ring networks, SIROCCO LNCS, Vol. 7355. pages 303–314, 2012.

Appendix

A The uniform deployment for the case of $n \neq ck$

To remove the restriction of $n = ck$ imposed in Section 4.1, only the parts for determining the target nodes and for moving to a target node should be modified. In the case that n is not a multiple of k, the distance between some adjacent target nodes should be $\lceil n/k \rceil$ or $\lfloor n/k \rfloor$.

The target nodes should be determined by each agent so that the decisions of different agents should be identical. Since all the agents recognize the same nodes as the base nodes, the common target nodes can be determined using the base node as a reference node: Let b be the number of the base nodes, and $r = n \bmod k$. The distance of every pair of adjacent base nodes is identical even in the case of $n \neq ck$, and is $n/b = (\lfloor n/k \rfloor \times k + r)/b = \lfloor n/k \rfloor \times k/b + r/b$ (notice that k/b and r/b are integers). This implies that we should select $k/b - 1$ target nodes between two adjacent base nodes so that the first r/b intervals between adjacent target nodes should be $\lceil n/k \rceil$ and others should be $\lfloor n/k \rfloor$. With considering the above, each agent can determine its own target node by local computation so that all the agents can spread over the ring to achieve the uniform deployment.