Asynchronous mobile robot gathering from symmetric configurations without global multiplicity detection

Sayaka Kamei, Anissa Lamani, Fukuhito Ooshita, Sébastien Tixeuil

Asynchronous mobile robot gathering from symmetric configurations without global multiplicity detection

Sayaka Kamei[*]
Dept. of Information Engineering, Hiroshima University, Japan
s-kamei@se.hiroshima-u.ac.jp

Anissa Lamani
Université de picardie Jules Vernes, Amiens, France
anissa.lamani@u-picardie.fr

Fukuhito Ooshita
Graduate School of Information Science and Technology, Osaka University, Suita, 565-0871, Japan
f-oosita@ist.osaka-u.ac.jp

Sébastien Tixeuil
LIP6 UMR 7606, Université Pierre et Marie Curie, France
Sebastien.Tixeuil@lip6.fr

Abstract

We consider a set of k autonomous robots that are endowed with visibility sensors (but that are otherwise unable to communicate) and motion actuators. Those robots must collaborate to reach a single vertex that is unknown beforehand, and to remain there hereafter. Previous works on gathering in ring-shaped networks suggest that there exists a tradeoff between the size of the set of potential initial configurations, and the power of the sensing capabilities of the robots (*i.e.* the larger the initial configuration set, the most powerful the sensor needs to be). We prove that there is no such trade off. We propose a gathering protocol for an odd number of robots in a ring-shaped network that allows symmetric but not periodic configurations as initial configurations, yet uses only local weak multiplicity detection. Robots are assumed to be anonymous and oblivious, and the execution model is the non-atomic CORDA model with asynchronous fair scheduling. Our protocol allows the largest set of initial configurations (with respect to impossibility results) yet uses the weakest multiplicity detector to date. The time complexity of our protocol is $O(n^2)$, where n denotes the size of the ring. Compared to previous work that also uses local weak multiplicity detection, we do *not* have the constraint that $k < n/2$ (here, we simply have $2 < k < n-3$).

Keywords: Gathering, Discrete Universe, Local Weak Multiplicity Detection, Asynchrony, Robots.

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 *gathering*, despite being limited with respect to input from the environment, asymmetry, memory, etc. The

---

area where robots have to gather is modeled as a graph and the gathering task requires every robot to reach a single vertex that is unknown beforehand, and to remain there hereafter.

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 among its neighbors, based on the previous observation. The move phase simply consists in moving toward the computed destination using motion actuators. We consider an asynchronous computing model, i.e., there may be a finite but unbounded time between any two phases of a robot's cycle. Asynchrony makes the problem hard since a robot can decide to move according to an old snapshot of the system and different robots may be in different phases of their cycles at the same time.

Moreover, 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 in the ring).

1.1 Related Work

While the vast majority of literature on coordinated distributed robots considers that those robots are evolving in a *continuous* two-dimensional Euclidean space and use visual sensors with perfect accuracy that permit to locate other robots with infinite precision, 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 or 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 [15, 13, 6, 5, 14, 1, 9, 7, 10, 11], this simplicity comes with the cost of extra symmetry possibilities, especially when the authorized paths are also symmetric.

In this paper, we focus on the discrete universe where two main problems have been investigated under these weak assumptions. The *exploration problem* consists in exploring a given graph using a minimal number of robots. Explorations come in two flavours: *with stop* (at the end of the exploration all robots must remain idle) [6, 5, 14] and *perpetual* (every node is visited infinitely often by every robot) [1]. The second studied problem is the *gathering* problem where a set of robots has to gather in one single location, not defined in advance, and remain on this location [9, 7, 10, 11].

The gathering problem was well studied in the continuous model with various assumptions [3, 2, 8, 17]. In the discrete model, deterministic algorithms have been proposed to solve the gathering problem in a ring-shaped network, which enables many problems to appear due to the high number of symmetric configurations. In [15, 13, 4], symmetry was broken by enabling robots to distinguish themselves using labels, in [7], symmetry was broken using tokens. The case of anonymous, asynchronous and oblivious robots was investigated only recently in this context. It should be noted that if the configuration is periodic and edge symmetric, no deterministic solution can exist [11]. The first two solutions [11, 10] are complementary: [11] is based on breaking the symmetry whereas [10] takes advantage of symmetries. However, both [11] and [10] make the assumption that robots are endowed with the ability to distinguish nodes that host one robot from nodes that host two robots or more in the entire network (this property is referred to in the literature as *global* weak multiplicity detection). This ability weakens the gathering problem because it is sufficient for a protocol to ensure that a single multiplicity point exists to have all robots gather in this point, so it reduces the gathering problem to the creation of a single multiplicity point.

Investigating the feasibility of gathering with weaker multiplicity detectors was recently addressed in [9]. In this paper, robots are only able to test that their current hosting node is a multiplicity node (*i.e.* hosts at least two robots). This assumption (referred to in the literature as *local* weak multiplicity detection) is obviously weaker than the global weak multiplicity detection, but is also more realistic as far as sensing devices are concerned. The downside of [9] compared to [10] is that only rigid configurations (*i.e.* non symmetric configuration) are allowed as initial configurations (as in [11]), while [10] allowed symmetric but not periodic configurations to be used as initial ones. Also, [9] requires that $k < n/2$ even in the case of non-symmetric configurations.

1.2 Our Contribution

We propose a gathering protocol for an odd number of robots in a ring-shaped network that allows symmetric but not periodic configurations as initial configurations, yet uses only local weak multiplicity detection. Robots are assumed to be anonymous and oblivious, and the execution model is the non-atomic CORDA model with asynchronous fair scheduling. Our protocol allows the largest set of initial configurations (with respect to impossibility results) yet uses the weakest multiplicity detector to date. The time complexity of our protocol is $O(n^2)$, where n denotes the size of the ring. By contrast to [9], k may be greater than $n/2$, as our constraint is simply that $2 < k < n - 3$ and k is odd.

2 Preliminaries

2.1 System Model

We consider here the case of an anonymous, unoriented and undirected ring of n nodes $u_0, u_1, ..., u_{(n-1)}$ such as u_i is connected to both $u_{(i-1)}$ and $u_{(i+1)}$. Note that since no labeling is enabled (anonymous), there is no way to distinguish between nodes, or between edges.

On this ring, k robots operate in distributed way in order to accomplish a common task that is to gather in one location not known in advance. We assume that k is odd. The set of robots considered here are *identical*; they execute the same program using no local parameters and one cannot distinguish them using their appearance, and are *oblivious*, which means that they have no memory of past events, they can't remember the last observations or the last steps taken before. In addition, they are unable to communicate directly, however, they have the ability to sense the environment including the position of the other robots. Based on the configuration resulting of the sensing, they decide whether to move or to stay idle. Each robot executes cycles infinitely many times, (1) first, it catches a sight of the environment to see the position of the other robots (look phase), (2) according to the observation, it decides to move or not (compute phase), (3) if it decides to move, it moves to its neighbor node towards a target destination (move phase).

At instant t, a subset of robots are activated by an entity known as *the scheduler*. The scheduler can be seen as an external entity that selects some robots for execution, this scheduler is considered to be fair, which means that, all robots must be activated infinitely many times. The *CORDA model* [16] enables the interleaving of phases by the scheduler (For instance, one robot can perform a look operation while another is moving). The model considered in our case is the CORDA model with the following constraint: the Move operation is instantaneous *i.e.* when a robot takes a snapshot of its environment, it sees the other robots on nodes and not on edges. However, since the scheduler is allowed to interleave the different operations, robots can move according to an outdated view since during the Compute phase, some robots may have moved.

During the process, some robots move, and at any time occupy nodes of the ring, their positions form a configuration of the system at that time. We assume that, at instant $t = 0$ (*i.e.*, at the initial configuration), some of the nodes on the ring are occupied by robots, such as, each node contains at most one robot. If

there is no robot on a node, we call the node *empty node*. The segment $[u_p, u_q]$ is defined by the sequence $(u_p, u_{p+1}, \cdots, u_{q-1}, u_q)$ of consecutive nodes in the ring, such as all the nodes of the sequence are empty except u_p and u_q that contain at least one robot. The distance D_p^t of segment $[u_p, u_q]$ in the configuration of time t is equal to the number of nodes in $[u_p, u_q]$ minus 1. We define a *hole* as the maximal set of consecutive empty nodes. That is, in the segment $[u_p, u_q]$, $(u_{p+1}, \cdots, u_{q-1})$ is a hole. The size of a hole is the number of free nodes that compose it, the border of the hole are the two empty nodes who are part of this hole, having one robot as a neighbor.

We say that there is a *tower* at some node u_i, if at this node there is more than one robot (Recall that this tower is distinguishable only locally).

When a robot takes a snapshot of the current configuration on node u_i at time t, it has a *view* of the system at this node. In the configuration $C(t)$, we assume $[u_1, u_2], [u_2, u_3], \cdots, [u_w, u_1]$ are consecutive segments in a given direction of the ring. Then, the view of a robot on node u_1 at $C(t)$ is represented by $(\max\{(D_1^t, D_2^t, \cdots, D_w^t), (D_w^t, D_{w-1}^t, \cdots, D_1^t)\}, m_1^t)$, where m_1^t is true if there is a tower at this node, and sequence $(a_i, a_{i+1}, \cdots, a_j)$ is larger than $(b_i, b_{i+1}, \cdots, b_j)$ if there is $h(i \leq h \leq j)$ such that $a_l = b_l$ for $i \leq l \leq h-1$ and $a_h > b_h$. It is stressed from the definition that robots don't make difference between a node containing one robot and those containing more. However, they can detect m^t of the current node, *i.e.* whether they are alone on the node or not (they have a local weak multiplicity detection).

When $(D_1^t, D_2^t, \cdots, D_w^t) = (D_w^t, D_{w-1}^t, \cdots, D_1^t)$, we say that the view on u_i is *symmetric*, otherwise we say that the view on u_i is *asymmetric*. Note that when the view is symmetric, both edges incident to u_i look identical to the robot located at that node. In the case the robot on this node is activated we assume the worst scenario allowing the scheduler to take the decision on the direction to be taken.

Configurations that have no tower are classified into three classes in [12]. Configuration is called *periodic* if it is represented by a configuration of at least two copies of a sub-sequence. Configuration is called *symmetric* if the ring contains a single axis of symmetry. Otherwise, the configuration is called *rigid*. For these configurations, the following lemma is proved in [11].

Lemma 1 *If a configuration is rigid, all robots have distinct views. If a configuration is symmetric and non-periodic, there exists exactly one axis of symmetry.*

This lemma implies that, if a configuration is symmetric and non-periodic, at most two robots have the same view.

We now define some useful terms that will be used to describe our algorithm. We denote by the *inter-distance d* the minimum distance taken among distances between each pair of distinct robots (in term of the number of edges). Given a configuration of inter-distance d, a *d.block* is any maximal elementary path where there is a robot every d edges. The border of a *d*.block are the two external robots of the *d*.block. The size of a *d*.block is the number of robots that it contains. We call the *d*.block whose size is biggest the *biggest d.block*. A robot that is not in any *d*.block is said to be an *isolated robot*.

We evaluate the time complexity of algorithms with asynchronous rounds. An asynchronous round is defined as the shortest fragment of an execution where each robot performs a move phase at least once.

2.2 Problem to be solved

The problem considered in our work is the gathering problem, where k robots have to agree on one location (one node of the ring) not known in advance in order to gather on it, and that before stopping there forever.

3 Algorithm

To achieve the gathering, we propose the algorithm composed of two phases. The first phase is to build a configuration that contains a single 1.block and no isolated robots without creating any tower regardless

of their positions in the initial configuration (provided that there is no tower and the configuration is aperiodic.) The second phase is to achieve the gathering from any configuration that contains a single 1.block and no isolated robots. Note that, since each robot is oblivious, it has to decide the current phase by observing the current configuration. To realize it, we define a special configuration set C_{sp} which includes any configuration that contains a single 1.block and no isolated robots. We give the behavior of robots for each configuration in C_{sp}, and guarantee that the gathering is eventually achieved from any configuration in C_{sp} without moving out of C_{sp}. We combine the algorithms for the first phase and the second phase in the following way: Each robot executes the algorithm for the second phase if the current configuration is in C_{sp}, and executes one for the first phase otherwise. By this way, as soon as the system becomes a configuration in C_{sp} during the first phase, the system moves to the second phase and the gathering is eventually achieved.

3.1 First phase: An algorithm to construct a single 1.block

In this section, we provide the algorithm for the first phase, that is, the algorithm to construct a configuration with a single 1.block. The strategy is as follows; In the initial configuration, robots search the biggest d.block B_1, and then robots that are not on B_1 move to join B_1. Then, we can get a single d.block. In the single d.block, there is a robot on the axis of symmetry because the number of robots is odd. When the nearest robots from the robot on the axis of symmetry move to the axis of symmetry, then we can get a $d-1$.block B_2, and robots that are not on B_2 move toward B_2 and join B_2. By repeating this way, we can get a single 1.block.

We will distinguish three types of configurations as follows:

- **Configuration of type** 1. In this configuration, there is only a single d.block such as $d > 1$, that is, all the robots are part of the d.block. Note that the configuration is in this case symmetric, and since there is an odd number of robots, we are sure that there is one robot on the axis of symmetry.

  If the configuration is this type, the robots that are allowed to move are the two symmetric robots that are the closest to the robot on the axis. Their destination is their adjacent empty node towards the robot on the axis of symmetry. (Note that the inter-distance has decreased.)

- **Configuration of type** 2. In this configuration, all the robots belong to d.blocks (that is, there are no isolated robots) and all the d.blocks have the same size.

  If the configuration is this type, the robots neighboring to hole and with the maximum view are allowed to move to their adjacent empty nodes. If there exists such a configuration with more than one robot and two of them may move face-to-face on the hole on the axis of symmetry, then they withdraw their candidacy and other robots with the second maximum view are allowed to move.

- **Configuration of type** 3. In this configuration, the configuration is not type 1 and 2, *i.e.*, all the other cases. Then, there is at least one biggest d.block whose size is the biggest.

  - If there exists an isolated robot that is neighboring to the biggest d.block, then it is allowed to move to the adjacent empty node towards the nearest neighboring biggest d.block. If there exist more than one such isolated robots, then only robots that are closest to the biggest d.block among them are allowed to move. If there exist more than one such isolated robots, then only robots with the maximum view among such isolated robots are allowed to move. The destination is their adjacent empty nodes towards one of the nearest neighboring biggest d.blocks.

  - If there exist no isolated robot that is neighboring to the biggest d.block, the robot that does not belong to the biggest d.block and is neighboring to the biggest d.block is allowed to move. If there exists more than one such a robot, then only robots with the maximum view among them are allowed to move. The destination is their adjacent empty node towards one of the nearest neighboring biggest d.blocks. (Note that the size of the biggest d.block has increased.)

Correctness of the algorithm In the followings, we prove the correctness of our algorithm.

5

Lemma 2 *From any non-periodic initial configuration without tower, the algorithm does not create a periodic configuration.*

Proof: Assume that, after a robot A moves, the system reaches a periodic configuration C^*. Let C be the configuration that A observed to decide the movement. The important remark is that, since we assume an odd number of robots, any periodic configuration should have at least three d.blocks with the same size or at least three isolated robots.

C is a configuration of type 1. Then, C has a single d.block and there is another robot B that is allowed to move. After configuration C, three cases are possible: A moves before B moves, A moves after B moves, or A and B move at the same time. After the movement of all the cases, the configuration C^* has exactly one $(d-1)$.block and thus C^* is not periodic.

C is a configuration of type 2. Let s be the size of d.blocks in C (Remind that all d.blocks have the same size). Since the number of robots is odd, $s \geq 3$ holds.

- If C is not symmetric, only A is allowed to move. When A moves, it either becomes an isolated robot or joins another d.block. Then, C^* has exactly one isolated robot or exactly one d.block with size $s+1$. Therefore, C^* is not periodic.
- If C is symmetric, there is another robot B that is allowed to move in C.
  - If A moves before B moves, C^* is not periodic similarly to the non-symmetric case.
  - If A and B move at the same time, they become isolated robots or join other d.blocks. Then, three cases are possible: C^* has exactly two isolated robots, C^* has exactly two d.blocks with size $s+1$, or C^* has exactly one d.block with size $s+2$. For all the cases C^* is not periodic.
  - Consider the case that B moves before A moves. Then, B becomes an isolated robot or joins another d.block. Let C' be the configuration after B moves. If A moves in C', C^* is not periodic since C^* is the same one as A and B move at the same time in C. Consequently, the remaining case is that other robots other than A move in C'.

    First, we consider the case that B joins d.block in C'. Then, the d.block becomes a single biggest d.block. This implies that all other robots move toward this d.block. Consequently, the following configurations contain exactly one biggest d.block, and thus C^* is not periodic.

    Second, we consider the case that B is an isolated robot in C'. Since only B is an isolated robot in C', only B is allowed to move in C' and it moves toward its neighboring d.block. If A moves before B joins the d.block, C^* contains exactly two isolated robots, and thus C^* is not periodic. After B joins the d.block, C^* is not periodic similarly to the previous case.

C is a configuration of type 3. Let s be the size of biggest d.blocks in C.

- If C is not symmetric, only A is allowed to move. When A moves, it either becomes an isolated robot or joins another d.block. In the latter case, C^* has exactly one d.block with size $s+1$, and thus C^* is not periodic. In the previous case, there may exist multiple isolated robots. However, A is the only one isolated robot such that the distance to its neighboring biggest d.block is the minimum. This means C^* is not periodic.
- If C is symmetric, there is another robot B that is allowed to move in C.
  - If A moves before B moves, C^* is not periodic similarly to the non-symmetric case.
  - If A and B move at the same time, they become isolated robots or join other d.blocks. In the latter case, C^* has exactly two d.block with size $s+1$ or exactly one d.block with size $s+2$, and thus C^* is not periodic. In the previous case, A and B are only two isolated robots such that the distance to its neighboring biggest d.block is the minimum. For both cases, C^* is not periodic.
  - Consider the case that B moves before A moves. Then, B becomes an isolated robot or joins another d.block. Let C' be the configuration after B moves. If A moves in C', C^* is not periodic since C^* is the same one as A and B move at the same time in C. Consequently, the remaining case is that other robots other than A move in C'.

First, we consider the case that B joins d.block in C'. Then, the d.block becomes a single biggest d.block. This implies that all other robots move toward this d.block. Consequently, the following configurations contain exactly one biggest d.block, and thus C^* is not periodic.

Second, we consider the case that B is an isolated robot in C'. Then, B is the only one isolated robot such that the distance to its neighboring biggest d.block is the minimum. Consequently, only B is allowed to move in C' and it moves toward its neighboring d.block. Even if A moves before B joins the d.block, B is the only one isolated robot such that the distance to its neighboring biggest d.block is the minimum. Consequently, in this case C^* is not periodic. After B joins the d.block, C^* is not periodic similarly to the previous case.

For all cases, C^* is not periodic; thus, a contradiction.   □

Lemma 3 *No tower is created before reaching a configuration with a single 1.block for the first time.*

Proof: If each robot that is allowed to move immediately moves until other robots take new snapshots, that is, no robot has outdated view, then it is clear that no tower is created.

Assume that a tower is created. Then, two robots A and B were allowed to move in a configuration, but the scheduler activates only A, and other robot C takes a snapshot after the movement of A before B moves. Because B moves based on the outdated view, if B and C moves face-to-face, then B and C may make a tower.

By the algorithm, in a view of a configuration, two robots are allowed to move if and only if the configuration is symmetric, because the maximum view is only one for each configuration other than symmetric configurations. If the configuration is not symmetric, only one robot E is allowed to move and the view of each robot does not change until E moves. Therefore, we should consider only symmetric configurations, and A and B are two symmetric robots.

- Consider the configuration of type 1 as before A moves. Then, there is a robot F on the axis of symmetry, and F is not allowed to move. The robots A and B are neighbor to F. In the case where the scheduler activates only A and A moves, then A and F create a new $d-1$.block. By the algorithm of the type 3(1), the closest isolated robot to this new $d-1$.bock is allowed to move in the new view. However, it is robot B, because the distance from B to the $d-1$.block is d but from other neighbor of the $d-1$.block is $d+1$. Therefore, other robots cannot move, and this is a contradiction.

- Consider the configuration of type 2 as before A moves. Then, by the exception, the robots that are face-to-face on the hole on the axis of symmetry are not allowed to move. Therefore, A and B are not neighboring to such hole on the axis of symmetry. After A moves, A becomes isolated or joins the other d.block, and the d.block A belonged to becomes not the biggest on the new view. If A becomes isolated, by the algorithm of type 3(1), A are allowed to move on the new view. Therefore, A can move and others than B cannot move until it joins the neighboring biggest d.block. After A joins the other d.block, the configuration becomes type 3(2) and the d.block D A belongs to is the biggest while B does not move. After that, other processes C neighboring to D can move toward D. Because B is not neighboring to D, C and B cannot move face-to-face. This is a contradiction.

- Consider the configuration of type 3(1) as before A moves. Then, A and B are isolated robots that are neighboring to the biggest d.block. Their destinations are the empty nodes towards the nearest neighboring biggest d.blocks respectively. The configuration after A move is type 3(1) until A join the nearest neighboring biggest d.block D. Then, A and B are allowed to move and others cannot move until A joins D. Consider the case after A joins D.
  - If there exist other isolated robot C neighboring to D, then it can move toward D because the configuration becomes type 3(1). However, C and B cannot move face-to-face because C are neighboring to D and B moves toward the border of other d.block. This is a contradiction.

– If there does not exist other isolated robot neighboring to *D*, then the configuration becomes type 3(2) on the new view. Then, the robot *C* neighboring to *D* can move toward *D*. However, *C* and *B* cannot move face-to-face because *C* are neighboring to *D* and *B* moves toward the other neighboring *d*.block. This is a contradiction.

- Consider the configuration of type 3(2) as before *A* moves. Then, *A* and *B* are neighboring to the biggest *d*.blocks and are members of any (not biggest) *d*.blocks. After *A* moves, *A* becomes isolated until *A* joins the biggest *d*.block *D* that is the destination of *A*. By the algorithm of type 3(1), *A* is allowed to move on the new view and others cannot move. After *A* joins *D*, *D* becomes biggest, the configuration becomes type 3(2). Then, the other process *C* neighboring to *D* can move to *D*. However, *B* and *C* cannot move face-to-face because *C* are neighboring to *D* and *B* moves toward the other neighboring *d*.block. This is a contradiction.

From the cases above, we can deduct that no tower is created before the gathering process. □

From Lemmas 2 and 3, the configuration is always non-periodic and does not have a tower from any non-periodic initial configuration without tower. Since configurations are not periodic, there exist one or two robots that are allowed to move unless the configuration contains a single 1.block.

Lemma 4 *Let C be a configuration such that its inter-distance is d and the size of the biggest d.block is s ($s \leq k - 1$). From configuration C, the configuration becomes such that the size of the biggest d.block is at least $s + 1$ in $O(n)$ rounds.*

Proof: From configurations of type 2 and type 3, at least one robot neighboring to the biggest *d*.block is allowed to move. Consequently, the robot moves in $O(1)$ rounds. If the robot joins the biggest *d*.block, the lemma holds.

If the robot becomes an isolated robot, the robot is allowed to move toward the biggest *d*.block by the configurations of type 3 (1). Consequently the robot joins the biggest *d*.block in $O(n)$ rounds, and thus the lemma holds. □

Lemma 5 *Let C be a configuration such that its inter-distance is d. From configuration C, the configuration becomes such that there is only single d.block in $O(kn)$ rounds.*

Proof: From Lemma 4, the size of the biggest *d*.block becomes larger in $O(n)$ rounds. Thus, the size of the biggest *d*.block becomes *k* in $O(kn)$ rounds. Since the configuration that has a *d*.block with size *k* is the one such that there is only single *d*.block. Therefore, the lemma holds. □

Lemma 6 *Let C be a configuration such that there is only single d.block ($d \geq 2$). From configuration C, the configuration becomes one such that there is only single $(d-1)$.block in $O(kn)$ rounds.*

Proof: From the configuration of type 1, the configuration becomes one such that there is $(d-1)$.block in $O(1)$ rounds. After that, the configuration becomes one such that there is only single $(d-1)$.block in $O(kn)$ rounds by Lemma 5. Therefore, the lemma holds. □

Lemma 7 *From any non-periodic initial configuration without tower, the configuration becomes one such that there is only single 1.block in $O(n^2)$ rounds.*

Proof: Let *d* be the inter-distance of the initial configuration. From the initial configuration, the configuration becomes one such that there is a single *d*.block in $O(kn)$ rounds by Lemma 5. Since the inter-distance becomes smaller in $O(kn)$ rounds by Lemma 6, the configuration becomes one such that there is only single 1.block in $O(dkn)$ rounds. Since $d \leq n/k$ holds, the lemma holds. □

8

3.2 Second phase: An algorithm to achieve the gathering

In this section, we provide the algorithm for the second phase, that is, the algorithm to achieve the gathering from any configuration with a single 1.block. As described in the beginning of this section, to separate the behavior from the one to construct a single 1.block, we define a special configuration set C_{sp} that includes any configuration with a single 1.block. Our algorithm guarantees that the system achieves the gathering from any configuration in C_{sp} without moving out of C_{sp}. We combine two algorithms for the first phase and the second phase in the following way: Each robot executes the algorithm for the second phase if the current configuration is in C_{sp}, and executes one for the first phase otherwise. By this way, as soon as the system becomes a configuration in C_{sp} during the first phase, the system moves to the second phase and the gathering is eventually achieved. Note that the system moves to the second phase without creating a single 1.block if it reaches a configuration in C_{sp} before creating a single 1.block.

The strategy of the second phase is as follows. When a configuration with a single 1.block is reached, the configuration becomes symmetric. Note that since there is an odd number of robots, we are sure that there is one robot $R1$ that is on the axis of symmetry. The two robots that are neighbor of $R1$ move towards $R1$. Thus $R1$ will have two neighboring holes of size 1. The robots that are neighbor of such a hole not being on the axis of symmetry move towards the hole. By repeating this process, a new 1.block is created (Note that its size has decreased and the tower is on the axis of symmetry). Consequently robots can repeat the behavior and achieve the gathering. Note that due to the asynchrony of the system, the configuration may contain a single 1.block of size 2. In this case, one of the two nodes of the block contains a tower (the other is occupied by a single robot). Since we assume a local weak multiplicity detection, only the robot that does not belong to a tower can move. Thus, the system can achieve the gathering.

In the followings, we define the special configuration set C_{sp} and the behavior of robots in the configurations. To simplify the explanation, we define a block as a maximal consecutive nodes where every node is occupied by some robots. The size $Size(B)$ of a block B denotes the number of nodes in the block. Then, we regard an isolated node as a block of size 1.

The configuration set C_{sp} is partitioned into five subsets: Single block C_{sb}, block leader C_{bl}, semi-single block C_{ssb}, semi-twin C_{st}, semi-block leader C_{sbl}. That is, $C_{sp} = C_{sb} \cup C_{bl} \cup C_{ssb} \cup C_{st} \cup C_{sbl}$ holds. We provide the definition of each set and the behavior of robots. Note that, although the definition of configurations specifies the position of a tower, each robot can recognize the configuration without detecting the position of a tower if the configuration is in C_{sp}.

- **Single block.** A configuration C is a single block configuration (denoted by $C \in C_{sb}$) if and only if there exists exactly one block $B0$ such that $Size(B0)$ is odd or equal to 2. Note that If $Size(B0)$ is equal to 2, one node of $B0$ is a tower and the other node is occupied by one robot. If $Size(B0)$ is odd, letting v_t be the center node of $B0$, no node other than v_t is a tower.

  In this configuration, robots move as follows: 1) If $Size(B0)$ is equal to 2, the robot that is not on a tower moves to the neighboring tower. 2) If $Size(B0)$ is odd, the configuration is symmetric and hence there exists one robot on the axis of symmetry (Let this robot be $R1$). Then, the robots that are neighbors of $R1$ move towards $R1$.

- **Block leader.** A configuration C is a block leader configuration (denoted by $C \in C_{bl}$) if and only if the following conditions hold (see Figure 3): 1) There exist exactly three blocks $B0$, $B1$, and $B2$ such that $Size(B0)$ is odd and $Size(B1) = Size(B2)$. 2) Blocks $B0$ and $B1$ share a hole of size 1 as their neighbors. 3) Blocks $B0$ and $B2$ share a hole of size 1 as their neighbors. 4) Letting v_t be the center node in $B0$, no node other than v_t is a tower. Note that, since $k < n - 3$ implies that there exist at least four free nodes, robots can recognize $B0$, $B1$, and $B2$ exactly.

  In this configuration, the robots in $B1$ and $B2$ that share a hole with B_0 as its neighbor move towards $B0$.

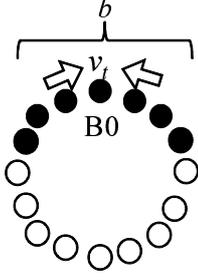- **Semi-single block.** A configuration C is a semi-single block configuration (denoted by $C \in C_{ssb}$) if

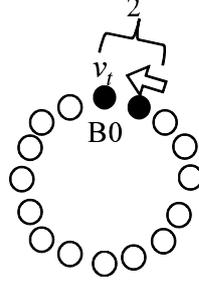Figure 1: Single block $C_{sb}(b)$ $(b > 2)$

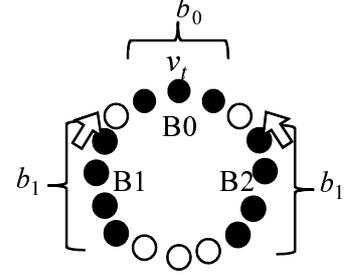Figure 2: Single block $C_{sb}(2)$
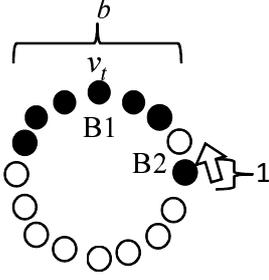

Figure 3: Block leader $C_{bl}(b_0, b_1)$

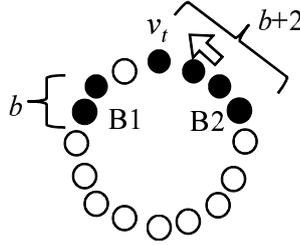Figure 4: Semi-single block $C_{ssb}(b)$
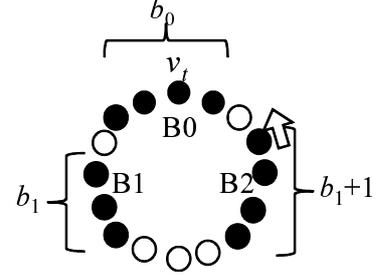

Figure 5: Semi-twin $C_{st}(b)$

Figure 6: Semi-block leader $C_{sbl}(b_0, b_1)$

and only if the following conditions hold (see Figure 4): 1) There exist exactly two blocks $B1$ and $B2$ such that $Size(B2) = 1$ and $Size(B1)$ is even (Note that this implies $Size(B1) + Size(B2)$ is odd.). 2) Blocks $B1$ and $B2$ share a hole of size 1 as their neighbors. 3) Letting v_t be a node in $B1$ that is the $(Size(B1)/2)$-th node from the side sharing a hole with $B2$, no node other than v_t is a tower.

In this configuration, the robot in $B2$ moves towards $B1$.

- **Semi-twin.** A configuration C is a semi-twin configuration (denoted by $C \in \mathcal{C}_{st}$) if and only if the following conditions hold (see Figure 5). 1) There exist exactly two blocks $B1$ and $B2$ such that $Size(B2) = Size(B1) + 2$ (Note that this implies $Size(B1) + Size(B2)$ is even, which is distinguishable from semi-single block configurations). 2) Blocks $B1$ and $B2$ share a hole of size 1 as their neighbors. 3) Letting v_t be a node in $B2$ that is the neighbor of a hole shared by $B1$ and $B2$, no node other than v_t is a tower.

In this configuration, the robot in $B2$ that is a neighbor of v_t moves towards v_t.

- **Semi-block leader.** A configuration C is a semi-block leader configuration (denoted by $C \in \mathcal{C}_{sbl}$) if and only if the following conditions hold (see Figure 6). 1) There exist exactly three blocks $B0$, $B1$, and $B2$ such that $Size(B0)$ is even and $Size(B2) = Size(B1) + 1$. 2) Blocks $B0$ and $B1$ share a hole of size 1 as their neighbors. 3) Blocks $B0$ and $B2$ share a hole of size 1 as their neighbors. 4) Letting v_t be a node in $B0$ that is the $(Size(B0)/2)$-th node from the side sharing a hole with $B2$, no node other than v_t is a tower. Note that, since $k < n - 3$ implies that there exist at least four free nodes, robots can recognize $B0$, $B1$, and $B2$ exactly.

In this configuration, the robot in $B2$ that shares a hole with $B0$ as a neighbor moves towards $B0$.

Correctness of the algorithm In the followings, we prove the correctness of our algorithm. To prove the correctness, we define the following notations.

- $\mathcal{C}_{sb}(b)$: A set of single block configurations such that $Size(B0) = b$.

10

- $C_{bl}(b_0, b_1)$: a set of block leader configurations such that $Size(B0) = b_0$ and $Size(B1) = Size(B2) = b_1$.
- $C_{ssb}(b)$: A set of semi-single block configurations such that $Size(B1) = b$.
- $C_{st}(b)$: A set of semi-twin configurations such that $Size(B1) = b$.
- $C_{sbl}(b_0, b_1)$: A set of semi-block leader configurations such that $Size(B0) = b_0$ and $Size(B1) = b_1$.

Note that every configuration in C_{sp} has at most one node that can be a tower (denoted by v_t in the definition). We denote such a node by a tower-construction node. In addition, we define an outdated robot as the robot that observes the outdated configuration and tries to move based on the outdated configuration.

From Lemmas 8 to 18, we show that, from any configuration $C \in C_{sp}$ with no outdated robots, the system achieves the gathering. However, some robots may move based on the configuration in the first phase. That is, some robots may observe the configuration in the first phase and try to move, however the configuration reaches one in the second phase before they move. Thus, in Lemma 19, we show that the system also achieves the gathering from such configurations with outdated robots.

Lemma 8 *From any single block configuration $C \in C_{sb}(b)$ ($b \geq 5$) with no outdated robots, the system reaches a configuration $C' \in C_{bl}(1, (b-3)/2)$ with no outdated robots in $O(1)$ rounds.*

Proof: In C the robots that are neighbors of the tower-construction node can move. Two sub cases are possible. First, the scheduler makes the two robots move at the same time. Once the robots move, they join the tower-construction node. Then, the configuration becomes one in $C_{bl}(1, (b-3)/2)$ and there exist no outdated robots. In the second case, the scheduler activates the two robots separately. In this case, one of the two robots first joins the tower-construction node and the configuration becomes one in $C_{st}((b-3)/2)$. After that, the other robot joins the tower-construction node (No other robots can move in this configuration). Then, the configuration becomes one in $C_{bl}(1, (b-3)/2)$ and there exist no outdated robots. In both cases, the transition requires at most $O(1)$ rounds and thus the lemma holds. □

Lemma 9 *From any single block configuration $C \in C_{sb}(3)$ with no outdated robots, the system achieves the gathering in $O(1)$ rounds.*

Proof: In C the robots that are neighbors of the tower-construction node can move. Two sub cases are possible. First, the scheduler makes the two robots move at the same time. Once the robots move, they join the tower-construction node. Then, the system achieves the gathering. In the second case, the scheduler activates the two robots separately. In this case, one of the two robots first joins the tower-construction node and the configuration becomes one in $C_{sb}(2)$. After that, the other robot joins the tower-construction node because robots on the tower never move due to the local multiplicity detection. And thus, the system achieves the gathering. In both cases, the transition requires at most $O(1)$ rounds and thus the lemma holds. □

Lemma 10 *From any block leader configuration $C \in C_{bl}(b_0, b_1)$ ($b_1 \geq 2$) with no outdated robots, the system reaches a configuration $C' \in C_{bl}(b_0 + 2, b_1 - 1)$ with no outdated robots in $O(1)$ rounds.*

Proof: In C the robots in $B1$ and $B2$ that share a hole of size 1 with $B0$ can move. Two sub cases are possible. First, the scheduler makes the two robots move at the same time. Once the robots move, they join $B0$. Since the size of $B0$ is increased by two and the size of $B1$ and $B2$ is decreased by one, the configuration becomes one in $C_{bl}(b_0 + 2, b_1 - 1)$. In addition, there exist no outdated robots. The second possibility is that the scheduler activates the two robots separately. In this case, one of the two robots first joins $B0$. Then, since the size of $B0$ is increased by one and the size of either $B1$ or $B2$ is decreased by one, the configuration becomes one in $C_{sbl}(b_0 + 1, b_1 - 1)$. After that, the other robot joins $B0$ (No other robots can move in this configuration). Then, the configuration becomes one in $C_{bl}(b_0 + 2, b_1 - 1)$ and there exist no outdated robots. In both cases, the transition requires at most $O(1)$ rounds and thus the lemma holds. □

11

Lemma 11 *From any block leader configuration $C \in \mathcal{C}_{bl}(b_0, 1)$ with no outdated robots, the system reaches a configuration $C' \in \mathcal{C}_{sb}(b_0 + 2)$ with no outdated robots in $O(1)$ rounds.*

Proof: In C the robots in $B1$ and $B2$ can move. Two sub cases are possible: *(i)* The scheduler makes the two robots move at the same time, then once the robots move, they join $B0$. Then, the system reaches a configuration in $\mathcal{C}_{sb}(b_0 + 2)$ with no outdated robots. *(ii)* The scheduler activates the two robots separately. In this case, one of the two robots first joins $B0$. Then, the configuration becomes one in $\mathcal{C}_{ssb}(b_0 + 1)$. After that, the other robot joins $B0$, and thus the configuration becomes one in $\mathcal{C}_{sb}(b_0 + 2)$ with no outdated robots. In both cases, the transition requires at most $O(1)$ rounds and thus the lemma holds. □

Lemma 12 *From any single block configuration $C \in \mathcal{C}_{sb}(b)$ $(b \geq 5)$ with no outdated robots, the system reaches a configuration $C' \in \mathcal{C}_{sb}(b-2)$ with no outdated robots in $O(k)$ rounds.*

Proof: From Lemma 8, the configuration becomes one in $\mathcal{C}_{bl}(1, (b-3)/2)$ in $O(1)$ rounds. After that, from Lemmas 10 and 11, the configuration becomes $\mathcal{C}_{sb}(b-2)$ in at most $O((b-3)/2)$ rounds. Since $b \leq k$, the lemma holds. □

Lemma 13 *From any single block configuration $C \in \mathcal{C}_{sb}(b)$ with no outdated robots, the system achieves the gathering in $O(k^2)$ rounds.*

Proof: From Lemma 12, if $b \geq 5$, the size of the block is decreased by two in $O(k)$ rounds. From Lemma 9, if the size of the block is 3, the system achieves the gathering in $O(1)$ rounds. □

Lemma 13 says that the system achieves the gathering in $O(k^2)$ rounds from any single block configuration in $\mathcal{C}_{sb}$. For other configurations, we can show the following lemmas.

Lemma 14 *From any block leader configuration $C \in \mathcal{C}_{bl}$ with no outdated robots, the system achieves the gathering in $O(k^2)$ rounds.*

Proof: Consider configuration $C \in \mathcal{C}_{bl}(b_0, b_1)$. From Lemmas 10 and 11, the configuration becomes $\mathcal{C}_{sb}(b_0 + 2b_1)$ in $O(k)$ rounds since $b_1 \leq k$ holds. After that, from Lemma 13, the system achieves the gathering in $O(k^2)$ rounds. □

Lemma 15 *From any semi-single block configuration $C \in \mathcal{C}_{ssb}$ with no outdated robots, the system achieves the gathering in $O(k^2)$ rounds.*

Proof: Consider configuration $C \in \mathcal{C}_{ssb}(b)$. Then, the configuration becomes $\mathcal{C}_{sb}(b+1)$ in $O(1)$ rounds. After that, from Lemma 13, the system achieves the gathering in $O(k^2)$ rounds. □

Lemma 16 *From any semi-twin configuration $C \in \mathcal{C}_{st}$ with no outdated robots, the system achieves the gathering in $O(k^2)$ rounds.*

Proof: Consider configuration $C \in \mathcal{C}_{st}(b)$. Then, the configuration becomes $\mathcal{C}_{bl}(1, b)$ in $O(1)$ rounds. After that, from Lemma 14, the system achieves the gathering in $O(k^2)$ rounds. □

Lemma 17 *From any semi-block leader configuration $C \in \mathcal{C}_{sbl}$ with no outdated robots, the system achieves the gathering in $O(k^2)$ rounds.*

Proof: Consider configuration $C \in \mathcal{C}_{sbl}(b_0, b_1)$. Then, the configuration becomes $\mathcal{C}_{bl}(b_0 + 1, b_1)$ in $O(1)$ rounds. After that, from Lemma 14, the system achieves the gathering in $O(k^2)$ rounds. □

From Lemmas 13, 14, 15, 16 and 17, we have the following lemma.

Lemma 18 *From any configuration $C \in \mathcal{C}_{sp}$ with no outdated robots, the system achieves the gathering in $O(k^2)$ rounds.*

Lemma 19 *From any configuration $C \in \mathcal{C}_{sp}$ with outdated robots, the system achieves the gathering in $O(k^2)$ rounds.*

Proof: We call the algorithm to construct a single 1.block Alg1, and the algorithm to achieve the gathering Alg2.

To construct a configuration $C \in \mathcal{C}_{sp}$ with outdated robots during Alg1, two robots P and Q have to observe a symmetric configuration C^* if they are activated by the scheduler they will move. From the behavior of Alg1, P and Q move toward their neighboring biggest d.blocks. Since P and Q observe a symmetric configuration, the directions of their movements are different each other.

We assume that P was activated by the scheduler however it executed only the look phase thus it doesn't move based on a configuration in Alg1, and the system reaches a configuration $C \in \mathcal{C}_{sp}$ by the behavior of Q (and other robots that move after Q joins the biggest d.block). Note that P and Q are isolated robots or the border of a block in C^*.

We have two possible types of configurations as C based on the behavior of Q.

- We say C is of TypeA if Q joins its neighboring biggest d.block between C^* and C. In this case, the join of Q creates exactly one biggest d.block. From the behavior of Alg1, other robots move toward this biggest d.block and thus there exists only one biggest d.block in C. In addition, since Q and other robots do not move to the block of P from the behavior of Alg1, the biggest d.block in C does not include P.
- We say C is of TypeB if Q does not join its neighboring biggest d.block between C^* and C. In this case, Q is an isolated robot in C. In addition, positions of non-isolated robots other than P and Q are the same in C and C^*.

We consider five cases according to the type of the configuration C: single block, block leader, semi-single block, semi-twin, and semi-block leader.

Single block Consider the case that C is single block. However, if C is of TypeA, there exist at least two blocks. If C is of TypeB, there exist at least one isolated robot. Both cases contradict the single block configuration.

Semi-twin Consider the case that C is semi-twin. Since the number of nodes occupied by robots is even, there exists a tower. However, since the behavior of Alg1 does not make a tower, this is a contradiction.

Semi-single block Consider the case that C is semi-single block.

If C is of TypeA, since P is not in the biggest block, P is in $B2$. If P moves to $B1$ by the same direction of Alg2, the system reaches a single block configuration. If P moves to the opposite direction, the system reaches a configuration with a single biggest block and one isolated robot P. In this configuration, only P can move by Alg1 and the system reaches a semi-single block configuration with no outdated robots. Therefore, the system reaches a single block configuration by Alg2 and achieves the gathering in $O(k^2)$ rounds.

If C is of TypeB, Q is in $B2$. However, this means Q moved out from $B1$ and thus the configuration was single block before Q moves. Thus, this is a contradiction.

Block leader Consider the case that C is block leader.

First, we assume $b_0 > b_1$, that is, $B0$ is the biggest block in C. Without loss of generality, we assume P is in $B1$ because the size of $B1$ and $B2$ is same.

13

- Consider the case that C is of TypeA.
  - Consider the case that the size of $B1$ is bigger than 1, then the size of $B2$ is also bigger than 1. If P is the border of $B1$ that shares a hole with $B0$, the destinations of P by Alg1 and Alg2 are the same. If P is the other border of $B1$, the biggest d.block of the destination in C^* is $B2$. Because $2 < k < n - 3$, the size of hole H between P and $B2$ is more than two. Because C^* is symmetric and $B2$ does not move between C^* and C and the size of holes other than H is one, it is a contradiction.
  - Consider the case that the size of $B1$ is equal to 1, the the size of $B2$ is also equal to 1. Then the destination of $B2$ is $B0$ by Alg2. If the destination of P is $B0$, the system achieves the semi-single block or single block with no outdated robots. If the destination of P is $B2$, the system reaches a configuration with a single biggest block and at least one isolated robot P with no outdated robots. However, by Alg1, the robot on the biggest block cannot move, and the isolated robots are the one that move to the biggest block. Therefore, the system reaches a semi-single block configuration or single block configuration with no outdated robots and achieves the gathering in $O(k^2)$ rounds.
- Consider the case that C is of TypeB. Because Q is an isolated robot, Q is part of $B2$ which is of size 1. Thus, $B1$ is also of size 1.

Second, we assume $b_0 < b_1$. Then, there exist two biggest d.blocks, and thus C is of TypeB. (Note that, in TypeA, the biggest d.block is only one.) Therefore, Q is an isolated robot and in $B0$. Without loss of generality, we assume that the destination of Q is $B2$ because the size of $B1$ and $B2$ is same. Since the size of a hole between $B1$ and Q is one in C, Q belongs to $B1$ in C^*. This implies that the size of $B1$ is $b_1 + 1$ and the size of $B2$ is b_1 in C^*. Since Q moves to the smaller block in C^*, this is a contradiction.

Finally, we assume $b_0 = b_1$. There are three biggest d.blocks. If C is of TypeA, because the biggest d.block is only one to which Q belongs. This is a contradiction. If C is of TypeB, because Q is an isolated robot, each size of $B0$, $B1$ and $B2$ is equal to 1. Then, because there are only three robots and C^* is symmetric, P and Q are not $B0$ and the destination of both of them is $B0$ by type 1 of Alg1. This destination is same as by Alg2.

Semi-block leader Consider the case that C is semi-block leader. Then, $B0$ or $B2$ is the biggest block in C.

If $b_0 = b_1 + 1$, there are two biggest blocks. This implies C is of TypeB, and then Q is an isolated robot. Consequently, $B1$ contains only Q, and thus $B0$ and $B2$ contain two robots. Since Q moves to the biggest block in C^*, Q is an isolated robot in C^* (Otherwise, Q is in a block $B0$ or $B2$ with the size three). Remind that P and Q are symmetric in C^*. However, P is in block $B0$ or $B2$. This is a contradiction, and thus this case never happens.

If $b_0 > b_1 + 1$, $B0$ is the biggest block in C. If C is of TypeA, Q joins $B0$. Then, P is in $B1$ or $B2$. By the definition of $B2$, the size of $B2$ is bigger than 1.

- Consider the case that P is in $B1$. If the destination of P in Alg1 is to $B0$, Q joins from $B2$ because C^* is symmetric and both destination of P and Q is $B0$. However, because the size of $B2$ is bigger than $B1$, it is a contradiction. If the destination of Q in Alg1 is to $B2$, then the position of $B2$ does not change from C^* because the size of $B2$ is bigger than 1. However, because the size of hole between $B1$ and $B2$ is more than 2 and C^* is symmetric, then there is another hole which size is more than 2, it is a contradiction.
- Consider the case that P is in $B2$. If the destination of P in Alg1 is to $B0$, then it is same as the destination by Alg2 and the other robot does not move. Therefore, the system achieves the gathering in $O(k^2)$ rounds. If the destination of P in Alg1 is to $B1$ and the size of $B1$ is bigger than 1, then the position of $B1$ does not change from C^*. However, because the size of hole between $B1$ and $B2$ is more

than 2 and C^* is symmetric, then there is another hole which size is more than 2, it is a contradiction. If the destination of P in Alg1 is to $B1$ and the size of $B1$ is equal to 1, then $B1$ moves to $B0$, the size of $B2$ is 2, and the robot R in $B2$ other than P moves to $B0$. Then, the members of $B0$ cannot move. If P moves, after R joins $B0$, then new destination of P is to $B0$ by both of Alg1 and Alg2. Therefore, the system achieves the gathering in $O(k^2)$ rounds.

If C is of TypeB, Q is an isolated robot and thus $B1$ contains only Q. Consequently, $B2$ contains two robots. Since Q moves toward the biggest block, Q moves toward $B0$. This means P is in $B2$ and moves toward $B0$. However, since P has a hole of size one in its direction, Q should also have a hole of size one in its direction in C^*. Then, Q joins a block immediately after Q moves, and thus C never becomes of TypeB. Therefore, this case never happens.

If $b_0 < b_1 + 1$, then $B2$ is the biggest d.block. If C is of TypeA, Q joins $B2$. Then, P is in $B0$ or $B1$.

- Consider the case that P is in $B0$. If the destination of P is $B1$, then Q joins $B2$ from $B0$ because the size of a hole between $B0$ and $B2$ is one. However, in C^*, the configuration is block leader, it is a contradiction. If the destination of P is $B2$, then Q joins $B2$ from the side of $B1$. If Q is a member of $B1$ in C^*, then the size of $B1$ is bigger than $B2$. If Q is not a member of $B1$ in C^*, then until all block members to which Q belongs in C^* join $B2$, the size of hole between the block and $B2$ is 1 because the size of hole between P and $B2$ is 1. Therefore, in C, all members to which Q belongs in C^* join $B2$. By considering the size of $B1$ and $B2$, Q is an isolated robot in C^*. Therefore, P in $B0$ is an isolated robot, that is, the number of robots in this case is 4. This is a contradiction.
- Consider the case that P is in $B1$.
  - Consider the case that P is a neighbor to $B2$. Then, in C, Q moves to $B2$, so Q is in $B0$. Because both size of holes between $B1$ and $B0$ and between $B0$ and $B2$ is 1, and P and Q are symmetric in C^*, the size of a hole between $B1$ and $B2$ is also 1. Because $n \geq k + 4$, it is a contradiction.
  - Consider the case that P is a neighbor to $B0$. Then, P tries to move toward $B0$. Since P and Q are symmetric in C^*, Q joins $B2$ from the side of $B1$.
    Remind that the difference between the size of $B1$ and that of $B2$ is one in C and the size of $B1$ does not increase from C^* to C. Therefore, the size of $B1$ and that of $B2$ are the same immediately before Q joins $B2$. On the other hand, since P and Q are symmetric in C^* and P belongs to $B1$, Q also belongs to $B1$. This implies the size of $B1$ is bigger than that of $B2$ in C^*. This is a contradiction.

If C is of TypeB, Q is an isolated robot.

- Consider the case that P is in $B0$. Then, the member of $B1$ is only Q, and the size of $B2$ is $b_1 + 1 = 2$. Because $b_0 < b_1 + 1 = 2$, the member of $B0$ is only P. Then, the number of robot is 4, and this is a contradiction.
- Consider the case that P is in $B1$. Then, the member of $B0$ is only Q. Since Q moves toward the biggest block, Q moves toward $B2$. This implies Q is in $B1$ in C^*, and consequently there exist only two blocks with the same size in C^*. Since P and Q are symmetric in C^*, the sizes of holes in both directions are the same. This implies C^* is periodic, and this is a contradiction.
- Consider the case that P is in $B2$. Then, since P is in the biggest block in C, P is also in the biggest block in C^*. Consequently, P moves toward the neighboring biggest block which is the same size as $B2$. This implies the size of $B0$ is at least $b_1 + 1$, and this is a contradiction.

□

From Lemmas 7, 18 and 19, we have the following theorem.

Theorem 1 *From any non-periodic initial configuration without tower, the system achieves the gathering in $O(n^2)$ rounds.*

4 Concluding remarks

We presented a new protocol for mobile robot gathering on a ring-shaped network. Contrary to previous approaches, our solution neither assumes that global multiplicity detection is available nor that the network is started from a non-symmetric initial configuration. Nevertheless, we retain very weak system assumptions: robots are oblivious and anonymous, and their scheduling is both non-atomic and asynchronous. We would like to point out some open questions raised by our work. First, the recent work of [5] showed that for the *exploration with stop* problem, randomized algorithm enabled that periodic and symmetric initial configurations are used as initial ones. However the proposed approach is not suitable for the non-atomic CORDA model. It would be interesting to consider randomized protocols for the gathering problem to bypass impossibility results. Second, investigating the feasibility of gathering without any multiplicity detection mechanism looks challenging. Only the final configuration with a single node hosting robots could be differentiated from other configurations, even if robots are given as input the exact number of robots.

References

[1] Lélia Blin, Alessia Milani, Maria Potop-Butucaru, and Sébastien Tixeuil. Exclusive perpetual ring exploration without chirality. In *DISC*, volume 6343, pages 312–327, 2010.

[2] Mark Cieliebak. Gathering non-oblivious mobile robots. In *LATIN*, pages 577–588, 2004.

[3] Mark Cieliebak, Paola Flocchini, Giuseppe Prencipe, and Nicola Santoro. Solving the robots gathering problem. In *ICALP*, pages 1181–1196, 2003.

[4] Anders Dessmark, Pierre Fraigniaud, Dariusz R. Kowalski, and Andrzej Pelc. Deterministic rendezvous in graphs. *Algorithmica*, 46(1):69–96, 2006.

[5] Stéphane Devismes, Franck Petit, and Sébastien Tixeuil. Optimal probabilistic ring exploration by semi-synchronous oblivious robots. In *SIROCCO*, pages 195–208, 2009.

[6] P. Flocchini, D. Ilcinkas, A. Pelc, and N. Santoro. Computing without communicating: Ring exploration by asynchronous oblivious robots. *OPODIS*, pages 105–118, 2007.

[7] Paola Flocchini, Evangelos Kranakis, Danny Krizanc, Nicola Santoro, and Cindy Sawchuk. Multiple mobile agent rendezvous in a ring. In *LATIN*, pages 599–608, 2004.

[8] 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.

[9] Tomoko Izumi, Taisuke Izumi, Sayaka Kamei, and Fukuhito Ooshita. Mobile robots gathering algorithm with local weak multiplicity in rings. In *SIROCCO*, pages 101–113, 2010.

[10] Ralf Klasing, Adrian Kosowski, and Alfredo Navarra. Taking advantage of symmetries: Gathering of asynchronous oblivious robots on a ring. In *OPODIS*, pages 446–462, 2008.

[11] Ralf Klasing, Euripides Markou, and Andrzej Pelc. Gathering asynchronous oblivious mobile robots in a ring. In *ISAAC*, pages 744–753, 2006.

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

[13] Dariusz R. Kowalski and Andrzej Pelc. Polynomial deterministic rendezvous in arbitrary graphs. In *ISAAC*, pages 644–656, 2004.

[14] Anissa Lamani, Maria Gradinariu Potop-Butucaru, and Sébastien Tixeuil. Optimal deterministic ring exploration with oblivious asynchronous robots. In *SIROCCO*, pages 183–196, 2010.

[15] Gianluca De Marco, Luisa Gargano, Evangelos Kranakis, Danny Krizanc, Andrzej Pelc, and Ugo Vaccaro. Asynchronous deterministic rendezvous in graphs. *Theor. Comput. Sci.*, 355(3):315–326, 2006.

[16] G. Prencipe. CORDA: Distributed coordination of a set of autonomous mobile robots. In *Proc. 4th European Research Seminar on Advances in Distributed Systems (ERSADS'01)*, pages 185–190, Bertinoro, Italy, May 2001.

[17] Giuseppe Prencipe. On the feasibility of gathering by autonomous mobile robots. In *SIROCCO*, pages 246–261, 2005.