PAPER

Time-Optimal Self-Stabilizing Leader Election on Rings in Population Protocols*

Daisuke YOKOTA[†], **Yuichi SUDO**[††a)], ***Nonmembers***, ***and*** **Toshimitsu MASUZAWA**[†], ***Member***

SUMMARY We propose a self-stabilizing leader election protocol on directed rings in the model of population protocols. Given an upper bound N on the population size n, the proposed protocol elects a unique leader within $O(nN)$ expected steps starting from any configuration and uses $O(N)$ states. This convergence time is optimal if a given upper bound N is asymptotically tight, i.e., $N = O(n)$.

key words: *leader election, self-stabilizing, population protocols*

1. Introduction

In this paper, We consider the *population protocol* (PP) model [2]. A network called *population* consists of a large number of finite-state automata, called *agents*. Agents make *interactions* (i.e., pairwise communication) with each other to update their states. The interactions are opportunistic, i.e., they are unpredictable for the agents. A population is modeled by a graph $G = (V, E)$, where V represents the set of agents, and E indicates which pair of agents can interact. Each pair of agents $(u, v) \in E$ has interactions infinitely often, while each pair of agents $(u', v') \notin E$ never has an interaction. At each time step, one pair of agents chosen uniformly at random from all pairs in E has an interaction. This assumption enables us to evaluate time complexities of population protocols. Almost all studies in the population protocol model make this assumption when they evaluate time complexities of population protocols. In the field of population protocols, many efforts have been devoted to devising protocols for a complete graph, i.e., a population where every pair of agents interacts infinitely often. In addition, several studies [2]–[12] have investigated populations represented by communication graphs other than complete graphs.

Self-stabilization [13] is a fault-tolerant property whereby, even when any number and kinds of faults occur, the network can autonomously recover from the faults. Formally, self-stabilization is defined as follows: (i) starting from an arbitrary configuration, a network eventually reaches a *safe configuration* (*convergence*), and (ii) once a network reaches a safe configuration, it maintains its specification forever (*closure*). Self-stabilization is of great importance in the PP model because self-stabilization tolerates any finite number of transient faults, and this is a necessary property in a network consisting of a large number of inexpensive and unreliable nodes.

Consequently, many studies have been devoted to self-stabilizing population protocols [3]–[6], [9]–[12], [14]–[19]. For example, Angluin et al. [3] proposed self-stabilizing protocols for a variety of problems, i.e., leader election in rings, token circulation in rings with a pre-selected leader, 2-hop coloring in degree-bounded graphs, consistent global orientation in undirected rings, and spanning tree construction in regular graphs. Sudo et al. [10], [12] gave a self-stabilizing 2-hop coloring protocol that uses a much smaller memory space of agents. Sudo et al. [20] investigates the possibility of self-stabilizing protocols for leader election, ranking, degree recognition, and neighbor recognition on arbitrary graphs.

Many of the above studies on self-stabilizing population protocols have focused on self-stabilizing leader election (SS-LE) because leader election is one of the most fundamental problems in the PP model. Several important protocols [2], [3], [21] require a pre-selected unique leader. In particular, Angluin et al. [21] show that all semi-linear predicates can be solved very quickly if we have a unique leader. The goal of the leader election problem is electing exactly one agent as a leader in the population. Unfortunately, SS-LE is impossible to solve without an additional assumption even if we focus only on complete graphs [3], [14], [20]. The studies to overcome this impossibility in the literature are roughly classified into four categories. (Some of the studies belong to multiple categories.)

The first category [14], [20], [22] assumes that every agent knows the exact number of agents. With this assumption, Cai et al. [14] gave an SS-LE protocol for complete graphs, Burman et al. [22] gave faster protocols in the same setting, and Sudo et al. [20] gave an SS-LE protocol for arbitrary graphs.

The second category [4], [5], [15] employs *oracles*, a kind of failure detectors. Fischer and Jiang [15] introduced an oracle $\Omega?$ that eventually tells all agents whether or not at least one leader exists. They proposed two SS-LE protocols using $\Omega?$, one for complete graphs and the other for rings. Canepa et al. [5] proposed two SS-LE protocols that use

Manuscript received October 19, 2020.
Manuscript revised April 17, 2021.
Manuscript publicized June 3, 2021.

[†]The authors are with Osaka University, Suita-shi, 565-0871 Japan.
[††]The author is with Hosei University, Koganei-shi, 184-8584 Japan.

*This work was supported by JAPS KAKENHI Grant No. 19H04085 and 20H04140. An extended abstract of this paper [1] appears in the proceedigs of the 22nd International Symposium on Stabilizing, Safety, and Security of Distributed Systems.

a) E-mail: sudo@hosei.ac.jp

DOI: 10.1587/transfun.2020EAP1125

Table 1 Self-stabilizing leader election on rings.

	Assumption	Convergence Time	#states
[3]	n is not multiple of a given k	$\Theta(n^3)$	$O(1)$
[15]	oracle $\Omega?$	$\Theta(n^3)$	$O(1)$
[6]	none	exponential	$O(1)$
this work	$n \leq N$ for a given N	$O(nN)$	$O(N)$

$\Omega?$, i.e., a deterministic protocol for trees and a randomized protocol for arbitrary graphs. Beauquier et al. [4] presented a deterministic SS-LE protocol for arbitrary graphs that uses two copies of $\Omega?$.

The third category [9]–[12], [16]–[19] slightly relaxes the requirement of the original self-stabilization and gave *loosely-stabilizing* leader election protocols. Specifically, the studies of this category allow a population to deviate from the specification of the problem (i.e., a unique leader) after the population satisfies the specification for an extremely long time. This concept was introduced by [18]. The protocols given by [16]–[19] work for complete graphs and those given by [9]–[12] work for arbitrary graphs. Recently, Sudo et al. [17] gave a time-optimal loosely-stabilizing leader election protocol for complete graphs: given a parameter $\tau \geq 1$, an execution of their protocol reaches a configuration with a unique leader within $O(\tau n \log n)$ expected steps starting from any configuration, and thereafter, it keeps the unique leader for $\Omega(n^\tau)$ expected steps, where n is the number of agents in the population.

The forth category [3], [6], [15], [23] restricts the topology of a graph to avoid the impossibility of SS-LE. A class $\mathcal{G}$ of graphs is called *simple* if there does not exist a graph in $\mathcal{G}$ which contains two disjoint subgraphs that are also in $\mathcal{G}$. Angluin et al. [3] proves that there exists no SS-LE protocol that works for all the graphs in any non-simple class. Thus, if we focus on a simple class of graphs, there may exist an SS-LE protocol for all graphs in the class. As a typical example, the class of *rings* is simple. Angluin et al. [3] gave an SS-LE protocol that works for all rings whose sizes are not multiples of a given integer k (in particular, rings of odd size). They posed a question whether SS-LE is solvable or not for general rings (i.e., rings of any size) without any oracle or knowledge such as the exact number of agents in the population, while Fischer and Jiang [15] solves SS-LE for general rings using oracle $\Omega?$. This question had been open for a decade until Chen and Chen [6] recently gave an SS-LE protocol for general rings. These three protocols given by [3], [6], [15] use only a constant number of states per agent. The expected convergence times (i.e., the expected numbers of steps required to elect a unique leader starting from any configuration) of the protocols proposed by [3], [15] are $\Theta(n^3)$, while the protocol given by [6] requires an exponentially long convergence time. The oracle $\Omega?$ only guarantees that it *eventually* reports to each agent whether there exists a leader in the population. Here, the convergence time of the protocol of [15] is bounded by $\Theta(n^3)$ assuming that the oracle immediately reports the absence of the leader to each agent. All of the three protocols assume that the rings are oriented or *directed*. However, this assumption is not essential because Angluin et al. [3] also presented a self-stabilizing ring orientation protocol, which gave a common sense of direction to all agents in the ring. In this paper, we also consider directed rings. Very recently, Chen and Chen [23] generalized their work on the rings for regular graphs.

1.1 Our Contribution

This paper belongs to the fourth category. We propose an SS-LE protocol P_{RL} for directed rings. Specifically, given an upper bound N on the population size n, P_{RL} elects a unique leader in $O(nN)$ expected steps for all directed rings (whose size is at most N). One can easily prove that no protocol can solve SS-LE in $o(n^2)$ expected steps. Thus, P_{RL} is time-optimal if a given upper bound N is asymptotically tight, i.e., $N = O(n)$. The results are summarized in Table 1.

The main contribution of this paper is a novel mechanism that largely improves the number of steps required to decrease the number of leaders to one when there are multiple leaders in the population. The mechanism requires only $O(n^2)$ expected steps, while the existing three SS-LE protocols for rings [3], [6], [15] requires $\Omega(n^3)$ expected steps to decrease the number of leaders to one. Our mechanism requires only $O(1)$ states, which is the same as the existing three protocols [3], [6], [15]. (Protocol P_{RL} requires $O(N)$ states only to detect the absence of a leader.) Thus, if we assume an oracle that reports to each leader an absence of a leader within $O(n^2)$ expected steps, we immediately obtain an SS-LE protocol with $O(n^2)$ expected convergence time and constant space per agent by using the proposed mechanism to remove leaders. We leave open an interesting question whether or not this oracle can be implemented with $o(N)$ states.

2. Preliminaries

In this section, we describe the formal definitions of our computation model.

A *population* is a simple and weakly connected digraph $G = (V, E)$, where V ($|V| \geq 2$) is a set of *agents* and $E \subseteq V \times V$ is a set of arcs. Each arc represents a possible *interaction* (or communication between two agents): If $(u, v) \in E$, agents u and v can interact with each other, where u serves as an *initiator* and v serves as a *responder*. If $(u, v) \notin E$, agents u and v never have an interaction. In this paper, we consider only a population represented by a *directed ring*, i.e., we assume that $V = \{u_0, u_1, \ldots, u_{n-1}\}$ and $E = \{(u_i, u_{i+1 \bmod n}) \mid i = 0, 1, \ldots, n-1\}$. Here, we use the indices of the agents only for simplicity of description. The agents are *anonymous*, i.e., they do not have unique identifiers. We call $u_{i-1 \bmod n}$ and

$u_{i+1 \bmod n}$ the left neighbor and the right neighbor of u_i, respectively. We omit "modulo by n" (i.e., mod n) in the index of agents when no confusion occurs.

A *protocol* $P(Q, Y, T, \pi_{out})$ consists of a finite set Q of states, a finite set Y of output symbols, transition function $T : Q \times Q \to Q \times Q$, and an output function $\pi_{out} : Q \to Y$. When an interaction between two agents occurs, T determines the next states of the two agents based on their current states. The *output of an agent* is determined by π_{out}: the output of agent v with state $q \in Q$ is $\pi_{out}(q)$. We assume that all agents have a common knowledge N on n such that $n \leq N = O(poly(n))$. Thus, the parameters Q, Y, T, and π_{out} can depend on the knowledge N. However, for simplicity, we do not explicitly write protocol P as parameterized with N, e.g., $P_N = (Q_N, Y_N, T_N, \pi_{\mathrm{out},N})$.

A *configuration* is a mapping $C : V \to Q$ that specifies the states of all the agents. Since a set Q of states is given by protocol $P(Q, Y, T, \pi_{out})$, the set of all (possible) configurations depends on P. We denote this set by $C_{\mathrm{all}}(P)$. We simply denote it by C_{all} when protocol P is clear from the context. We say that configuration C changes to C' by an interaction $e = (u_i, u_{i+1})$, denoted by $C \xrightarrow{e} C'$ if we have $(C'(u_i), C'(u_{i+1})) = T(C(u_i), C(u_{i+1}))$ and $C'(v) = C(v)$ for all $v \in V \setminus \{u_i, u_{i+1}\}$. We simply write $C \to C'$ if there exits $e \in E$ such that $C \xrightarrow{e} C'$. We say that a configuration C' is reachable from C if there exists a sequence of configurations $C_0, C_1, \ldots, C_k$ such that $C = C_0$, $C' = C_k$, and $C_i \to C_{i+1}$ for all $i = 0, 1, \ldots, k-1$. We also say that a set $\mathcal{C}$ of configurations is *closed* if no configuration outside $\mathcal{C}$ is reachable from a configuration in $\mathcal{C}$.

A scheduler determines which interaction occurs at each time step (or just *step*). In this paper, we consider a *uniformly random scheduler* $\boldsymbol{\Gamma} = \Gamma_0, \Gamma_1, \ldots$: each $\Gamma_t \in E$ is a random variable such that $\Pr(\Gamma_t = (u_i, u_{i+1})) = 1/n$ for any $t \geq 0$ and $i = 0, 1, \ldots, n-1$. Each Γ_t represents the interaction that occurs at step t. Given an initial configuration C_0, the *execution* of protocol P under $\boldsymbol{\Gamma}$ is defined as $\Xi_P(C_0, \boldsymbol{\Gamma}) = C_0, C_1, \ldots$ such that $C_t \xrightarrow{\Gamma_t} C_{t+1}$ for all $t \geq 0$. We denote $\Xi_P(C_0, \boldsymbol{\Gamma})$ simply by $\Xi_P(C_0)$ when no confusion occurs.

We address the self-stabilizing leader election problem in this paper. For simplicity, we give the definition of a self-stabilizing leader election protocol instead of giving the definitions of self-stabilization and the leader election problem separately.

Definition 1 (Self-stabilizing Leader Election)**.** For any protocol P, we say that a configuration C of P is *safe* if (i) exactly one agent outputs L (leader) and all other agents output F (follower) in C, and (ii) at every configuration reachable from C, all agents keep the same outputs as those in C. A protocol P is a *self-stabilizing leader election (SS-LE) protocol* if $\Xi_P(C_0, \boldsymbol{\Gamma})$ reaches a safe configuration with probability 1.

We evaluate a SS-LE protocol P with two metrics: *the expected convergence time* and *the number of states*. For a configuration $C_0 \in C_{\mathrm{all}}(P)$, let t_{P,C_0} be the expected number of steps until $\Xi_P(C_0, \boldsymbol{\Gamma})$ reaches a safe configuration. The expected convergence time of P is defined as $\max_{C_0 \in C_{\mathrm{all}}(P)} t_{P,C_0}$. The number of states of $P = (Q, Y, T, O)$ is simply $|Q|$.

3. Self-Stabilizing Leader Election Protocol

In this section, we propose a SS-LE protocol P_{RL} that works in any directed ring consisting of N agents or less. The expected convergence time is $O(nN)$, and the number of states is $O(N)$. Thus, P_{RL} is time-optimal if a given upper bound N of n is asymptotically tight, i.e., $N = O(n)$.

The pseudocode of P_{RL} is given in Algorithm 1, which describes how two agents l and r update their states, i.e., their variables, when they have an interaction. Here, l and r represents the initiator and the responder in the interaction, respectively. That is, l is the left neighbor of r, and r is the right neighbor of l. We denote the value of the variable `var` at agent $v \in V$ by $v.\mathtt{var}$. Similarly, we denote the variable `var` in state $q \in Q$ by $q.\mathtt{var}$. In this algorithm, each agent $v \in V$ maintains an *output* variable $v.\mathtt{leader} \in \{0, 1\}$, according to which it determines its output. Agent v outputs L when $v.\mathtt{leader} = 1$ and outputs F when $v.\mathtt{leader} = 0$. We say that v is a *leader* if $v.\mathtt{leader} = 1$; otherwise v is called a *follower*. For each $u_i \in V$, we define the distance to *the nearest left leader* and the distance to *the nearest right leader* of u_i as $d_L(i) = \min\{j \geq 0 \mid u_{i-j}.\mathtt{leader} = 1\}$ and $d_R(i) = \min\{j \geq 0 \mid u_{i+j}.\mathtt{leader} = 1\}$, respectively. When there is no leader in the ring, we define $d_L(i) = d_R(i) = \infty$.

Algorithm P_{RL} consists of two parts: the leader creation part (Lines 1–5) and the leader elimination part (Lines 6–16). Since P_{RL} is a self-stabilizing protocol, it has to handle any initial configuration, where there may be no leader or multiple leaders. The leader creation part creates a new leader when there is no leader, while the leader elimination part decreases the number of leaders to one when there are two or more leaders.

3.1 Leader Elimination

We are inspired by the algorithm of [15] to design the leader elimination part (Lines 6–16). Roughly speaking, the strategy of [15] can be described as follows.

- Each agent may have a *bullet* and/or a *shield*.
- A leader always fires a bullet: each time a leader u_{i+1} having no bullet interacts with an agent u_i, the leader u_{i+1} makes a bullet.
- Bullets move from left to right in the ring: each time u_i having a bullet interacts with u_{i+1}, the bullet moves from u_i to u_{i+1}.
- Conversely, shields move from right to left: each time u_{i+1} having a shield interacts with u_i, the shield moves from u_{i+1} to u_i.
- Each time two agents both with bullets (resp. shields) have an interaction, the left bullet (resp. the right shield) disappears.
- When a bullet and a shield pass each other, i.e., u_i with

Algorithm 1: P_{RL}

Variables:
$\texttt{leader} \in \{0,1\}, \texttt{bullet} \in \{0,1,2\}, \texttt{shield} \in \{0,1\}$
$\texttt{signal} \in \{0,1\}, \texttt{distL} \in \{0,1,\ldots,N\}$

Interaction between initiator l and responder r:

1 $l.\texttt{distL} \leftarrow \begin{cases} 0 & l.\texttt{leader} = 1 \\ l.\texttt{distL} & \text{otherwise} \end{cases}$

2 $r.\texttt{distL} \leftarrow \begin{cases} 0 & r.\texttt{leader} = 1 \\ \min(l.\texttt{distL}+1, N) & r.\texttt{leader} = r.\texttt{bullet} = 0 \\ r.\texttt{distL} & \text{otherwise} \end{cases}$

3 **if** $r.\texttt{distL} = N$ **then**
4 $\quad r.\texttt{leader} \leftarrow 1; r.\texttt{distL} \leftarrow 0$
5 $\quad r.\texttt{bullet} \leftarrow 2; r.\texttt{shield} \leftarrow 1; r.\texttt{signal} \leftarrow 0$

6 **if** $l.\texttt{leader} = l.\texttt{signal} = 1$ **then**
7 $\quad l.\texttt{bullet} \leftarrow 2; l.\texttt{shield} \leftarrow 1; l.\texttt{signal} \leftarrow 0$

8 **if** $r.\texttt{leader} = r.\texttt{signal} = 1$ **then**
9 $\quad r.\texttt{bullet} \leftarrow 1; r.\texttt{shield} \leftarrow 0; r.\texttt{signal} \leftarrow 0$

10 **if** $l.\texttt{bullet} > 0 \wedge r.\texttt{leader} = 1$ **then**
11 $\quad r.\texttt{leader} \leftarrow \begin{cases} 0 & l.\texttt{bullet} = 2 \wedge r.\texttt{shield} = 0 \\ 1 & \text{otherwise} \end{cases}$
12 $\quad l.\texttt{bullet} \leftarrow 0$

13 **else if** $l.\texttt{bullet} > 0 \wedge r.\texttt{leader} = 0$ **then**
14 $\quad r.\texttt{bullet} \leftarrow \begin{cases} l.\texttt{bullet} & r.\texttt{bullet} = 0 \\ r.\texttt{bullet} & r.\texttt{bullet} > 0 \end{cases}$
15 $\quad l.\texttt{bullet} \leftarrow 0; r.\texttt{signal} \leftarrow 0$

16 $l.\texttt{signal} \leftarrow \max(l.\texttt{signal}, r.\texttt{signal}, r.\texttt{leader})$

a bullet and u_{i+1} with a shield have an interaction, the bullet disappears.
- When a bullet moves to a leader without a shield, the leader is *killed* (i.e., becomes a follower).

The algorithm of [15] assumes an oracle, called an *eventual leader detector* $\Omega?$, which detects and tells each agent whether a leader exists or not, when there is continuously a leader or there is continuously no leader. A follower becomes a leader when it is reported by $\Omega?$ that there is no leader in the population. At this time, the new leader simultaneously generates both a shield and a bullet. One can easily observe that by the above strategy together with oracle $\Omega?$, the population eventually reaches a configuration after which there is always one fixed leader. However, the algorithm of [15] requires $\Omega(n^3)$ steps to elect one leader in the worst case even if oracle $\Omega?$ can immediately report to each agent whether there is a leader in the population.

We drastically modify the above strategy of [15] for the leader elimination part of P_{RL} to decrease the number of leaders to one within $O(n^2)$ steps. First, a shield never moves in our algorithm. Only leaders have shields. A leader sometimes generates a shield and sometimes breaks a shield. Second, a leader does not always fire a bullet. Instead, a leader fires a new bullet only after it detects that the last bullet it fired reaches a (possibly different) leader. Leaders use *bullet-absence signals* for the detection. Roughly speaking, every leader always sends a bullet-absence signal to its left neighbor and the signal propagates from right to left. On the other hand, a bullets move from left to right, and a bullet always deletes bullet-absence signals when it finds them. Thus, a leader receives a bullet-absence signal only when its last fired bullet reaches a (possibly different) leader. Third, we have two kinds of bullets: *live bullets* and *dummy bullets*. A live bullet kills a leader without a shield. However, a dummy bullet does not have capability to kill a leader. When a leader decides to fire a new bullet, the bullet becomes live or dummy with the probability $1/2$ each. When a leader fires a live bullet, it simultaneously generates a shield (if it does not have a shield). When a leader fires a dummy bullet, it breaks the shield if it has a shield. Thus, roughly speaking, each leader is *shielded* (i.e., has a shield) with probability $1/2$ at each step. Therefore, when a live bullet reaches a leader, the leader is killed with probability $1/2$. This strategy is well designed: not all leaders kill each other simultaneously because a leader must be shielded if it fired a live bullet in the last shot. As a result, the number of leaders eventually decreases to one.

In what follows, we explain how we implement this strategy. Each agent v maintains variables $v.\texttt{bullet} \in \{0,1,2\}$, $v.\texttt{shield} \in \{0,1\}$, and $v.\texttt{signal} \in \{0,1\}$. As their names imply, $v.\texttt{bullet} = 0$ (resp. $v.\texttt{bullet} = 1$, $v.\texttt{bullet} = 2$) indicates that v is now conveying no bullet (resp. a dummy bullet, a live bullet), while $v.\texttt{shield} = 1$ indicates that v is shielded. Unlike the protocol of [15], we ignore the value of $v.\texttt{shield}$ for any follower v. A variable $\texttt{signal}$ is used by a leader to detect that the last bullet it fired already disappeared. Specifically, $v.\texttt{signal} = 1$ indicates that v is propagating a *bullet-absence signal*. A leader always generates a bullet-absence signal in its left neighbor when it interacts with its left neighbor (Line 16). This signal propagates from right to left (Line 16), while a bullet moves from left to right (Lines 14 and 15). A bullet disables a bullet-absence signal regardless of whether it is live or dummy, i.e., $u_{i+1}.\texttt{signal}$ is reset to 0 when two agents u_i and u_{i+1} such that $u_i.\texttt{bullet} > 0$ and $u_{i+1}.\texttt{signal} = 1$ have an interaction (Lines 14 and 15). Thus, a bullet-absence signal propagates to a leader only after the last bullet fired by the leader disappears (Fig. 1).

When a leader u_i receives a bullet-absence signal from its right neighbor u_{i+1}, u_i waits for its next interaction to extract randomness from the uniformly random scheduler. At

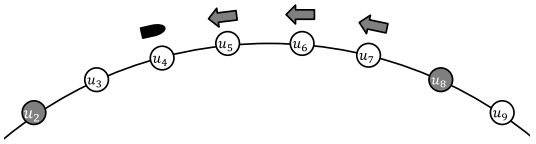


Fig. 1 A bullet and bullet-absence signals. The gray and white circles represent leaders and followers respectively. The gray (left) arrows represent bullet-absence signals. Bullet-absence signals move from right to left, while a bullet moves from left to right. A bullet deletes a bullet-absence signal when it meets the signal. Thus, after a leader fires a bullet b, it will never receives a bullet-absence signal before b disappears.

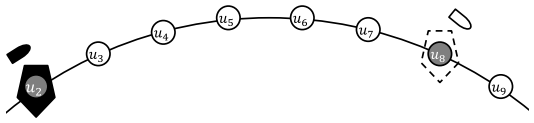


Fig. 2 A shield and bullets. A black bullet represents a live bullet and white one represents a dummy bullet. When a leader fires a live bullet, it simultaneously generates a shield if it does not have a shield. When a leader fires a dummy bullet, it breaks the shield if it has a shield.

the next interaction, by the definition of the uniformly random scheduler, u_i meets its right neighbor u_{i+1} with probability $1/2$ and its left neighbor u_{i-1} with probability $1/2$. In the former case, u_i fires a live bullet and becomes shielded (Line 7). In the latter case, u_i fires a dummy bullet and becomes unshielded (Lines 9) (Fig. 2). In both cases, the received signal is deleted (Lines 7 and 9). The fired bullet moves from left to right each time the agent with the bullet, say u_i, interacts with its right neighbor u_{i+1} (Lines 14 and 15). However, the bullet disappears without moving to u_{i+1} if u_{i+1} already has another bullet at this time. Suppose that the bullet now reaches a leader. If the bullet is live and the leader is not shielded at that time, the leader is killed by the bullet (Line 11). The bullet disappears at this time regardless of whether the bullet is alive and/or the leader is shielded (Line 12).

3.2 Leader Creation

The leader creation part is simple (Lines 1–5). Each agent u_i estimates $d_L(i)$ and stores the estimated value on variable $u_i.\texttt{distL} \in \{0, 1, \ldots, N\}$. Specifically, at each interaction (u_i, u_{i+1}), agents u_i and u_{i+1} update their `distL` as follows (Lines 1 and 2): (i) u_i (resp. u_{i+1}) resets its `distL` to zero if u_i (resp. u_{i+1}) is a leader, and (ii) if u_{i+1} is not a leader and does not have a bullet, $\min(l.\texttt{distL} + 1, N)$ is substituted for $u_{i+1}.\texttt{distL}$. Thus, if there is no leader in the population, some agent v eventually increases $v.\texttt{distL}$ to N, and at that time, the agent decides that there is no leader. Then, this agent becomes a leader, executing $v.\texttt{leader} \leftarrow 1$ and $v.\texttt{distL} \leftarrow 0$ (Line 4). At the same time, v fires a live bullet, generates a shield, and disables a bullet-absence signal (Line 5). This live bullet prevents the new leader from being killed until it disappears. This is because (i) the leader is shielded when it is created, (ii) it never receives a bullet-absence signal before this live bullet disappears (Fig. 1), and (iii) the leader fires a bullet again and becomes unshielded only when it receives an bullet-absence signal.

As mentioned above, at interaction (u_i, u_{i+1}), the distance propagation does not occur if u_{i+1} is a leader. This exception helps us to simplify the analysis of the convergence time, i.e., we can easily get an upper bound on the expected number of steps before each bullet disappears. Note that there are two cases that a bullet disappears: (i) when it reaches a leader, and (ii) when it reaches another bullet. The first case includes an interaction (u_i, u_{i+1}) where $u_i.\texttt{distL} \geq N - 1$ holds and u_{i+1} becomes a leader by Line 4. Formally, at an interaction (u_i, u_{i+1}) such that $u_i.\texttt{bullet} \geq 1$, we say that a bullet located at u_i disappears if $u_{i+1}.\texttt{leader} = 1$, $u_{i+1}.\texttt{bullet} \geq 1$, or $u_i.\texttt{distL} \geq N - 1$. We have the following lemma thanks to the above exception.

Lemma 1. *Every bullet disappears before it moves (from left to right) N times.*

We should note that the leader creation part may create a leader even when there is one or more leaders, thus this part may prevent the leader elimination part from decreasing the number of leaders to one. Fortunately, as we will see in Sect. 4, within $O(nN)$ steps in expectation, the population reaches a configuration after which no new leader is created.

4. Correctness and Time Complexity

In this section, we prove that P_{RL} is a SS-LE protocol on directed rings of any size $n(\leq N)$ and that the expected convergence time of P_{RL} is $O(nN)$. In Sect. 4.1, we define a set $\mathcal{S}_{\mathrm{RL}}$ of configurations and prove that every configuration in $\mathcal{S}_{\mathrm{RL}}$ is safe. In Sect. 4.2, we prove that the population starting from any configuration reaches a configuration in $\mathcal{S}_{\mathrm{RL}}$ within $O(nN)$ steps in expectation.

4.1 Safe Configurations

In this paper, we use several functions whose return values depend on a configuration, such as $d_L(i)$ and $d_R(i)$. When a configuration should be specified, we explicitly write a configuration as the first argument of those functions. For example, we write $d_L(C, i)$ and $d_R(C, i)$ to denote $d_L(i)$ and $d_R(i)$ in a configuration C, respectively.

In protocol P_{RL}, leaders kill each other by firing live bullets to decrease the number of leaders to one. However, it is undesirable that all leaders are killed and the number of leaders becomes zero. Therefore, a live bullet should not kill a leader if it is the last leader (i.e., the unique leader) in the population. We say that a live bullet located at agent u_i is *peaceful* when the following predicate holds:

$$\mathit{Peaceful}(i) \stackrel{\text{def}}{\equiv} \begin{pmatrix} u_{i-d_L(i)}.\texttt{shield} = 1 \\ \wedge\, \forall j = 0, 1, \ldots, d_L(i) : \\ \qquad\qquad u_{i-j}.\texttt{signal} = 0 \end{pmatrix}.$$

A peaceful bullet could kill some leader, but it never kills the last leader in the population because its nearest left leader is shielded. A peaceful bullet never becomes non-peaceful; because letting u_i be the agent at which the bullet is located, the agents $u_{i-d_L(i)}, u_{i-d_L(i)+1}, \ldots, u_i$ will never have a bullet-absence signal thus $u_{i-d_L(i)}$ never becomes unshielded before the bullet disappears. At the beginning of an execution, there may be one or more non-peaceful live bullets. However, every newly-fired live bullet is peaceful because a leader becomes shielded and disables the bullet-absence signal when it fires a live bullet. Thus, once the population reaches a configuration where every live bullet is peaceful and there is one or more leaders, the number of leaders never

becomes zero. Formally, we define the set of such configurations as follows:

$$C_{\mathrm{PB}} = \left\{ C \in C_{\mathrm{all}} \;\middle|\; \begin{array}{l} \exists u_i \in V : C(u_i).\mathtt{leader} = 1 \\ \wedge \forall u_j \in V : \begin{pmatrix} C(u_j).\mathtt{bullet} = 2 \\ \Rightarrow \mathit{Peaceful}(C, j) \end{pmatrix} \end{array} \right\}.$$

The following lemma holds from the above discussion.

Lemma 2. *C_{PB} is closed.*

Thus, once the population reaches a configuration in C_{PB}, there is always one or more leaders.

In protocol P_{RL}, a new leader is created when `distL` of some agent reaches N. We require this mechanism to create a new leader when there is no leader. However, it is undesirable that a new leader is created when there is already one or more leaders. We say that an agent u_i is *secure* when the following predicate holds:

$$\mathit{Secure}(i) \stackrel{\mathrm{def}}{\equiv} \begin{cases} u_i.\mathtt{distL} = 0 & u_i.\mathtt{leader} = 1 \\ u_i.\mathtt{distL} \leq N - d_R(i) & \text{otherwise.} \end{cases}$$

One may think that no leader is created once the population reaches a configuration in C_{PB} such that all agents are secure. Unfortunately, this does not hold. For example, consider the case $n = N = 100$ and a configuration $C \in C_{\mathrm{PB}}$ where

- only two agents u_0 and u_{50} are leaders,
- $u_0.\mathtt{distL} = u_{50}.\mathtt{distL} = 0$,
- $u_i.\mathtt{distL} = 100 - d_R(i)$ for all $i = 1, 2, \ldots, 49, 51, 52, \ldots, 100$,
- u_{49} carries a live (and peaceful) bullet in C, i.e., $u_{49}.\mathtt{bullet} = 2$, and
- u_{50} is not shielded, i.e., $u_{50}.\mathtt{shield} = 0$.

Note that the above condition does not contradict the assumption $C \in C_{\mathrm{PB}}$. In this configuration, all agents are secure. However, starting from this configuration, the population may create a new leader even when another leader exists. In configuration C, $u_{49}.\mathtt{distL} = 99$. If u_{49} and u_{50} have two interactions in a row, then u_{50} becomes a follower in the first interaction, and $u_{49}.\mathtt{distL} + 1 = 100$ is substituted for $u_{50}.\mathtt{distL}$ and u_{50} becomes a leader again in the second interaction (even though u_0 is a leader during this period) (Fig. 3).

We introduce the definition of *modest bullets* to clarify

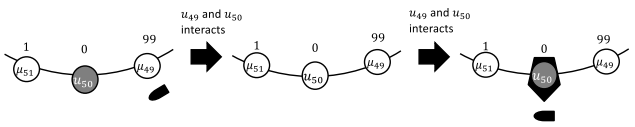


Fig. 3 The example of an execution where a new leader is created from a configuration in C_{PB} in which all agents are secure. A gray circle represents a leader, while a white circle represents a follower. An integer attached to an agent represents the value of `distL` in the agent. At the first interaction, the unshielded leader u_{50} is killed and becomes a follower. Note that this interaction makes u_{49} insecure. At the second interaction, $u_{49}.\mathtt{distL} + 1 = 100$ is substituted for $u_{50}.\mathtt{distL}$, which makes u_{50} a leader again.

the condition by which a new leader is no longer created. A live bullet located at u_i is said to be *modest* when the following predicate holds:

$$\mathit{Modest}(i) \stackrel{\mathrm{def}}{\equiv} \mathit{Peaceful}(i) \wedge \begin{pmatrix} \forall j = 0, 1, \ldots, d_L(i) : \\ u_{i-j}.\mathtt{distL} \leq d_L(i - j) \end{pmatrix}.$$

As we will see soon, a new leader is no longer created in an execution starting from a configuration in C_{PB} where all agents are secure and all live bullets are modest. Note that in the above example, a live bullet located at u_{49} in C is not modest. We define a set C_{NI} of configurations as follows:

$$C_{\mathrm{NI}} = \left\{ C \in C_{\mathrm{PB}} \;\middle|\; \begin{array}{l} \forall u_i \in V : \mathit{Secure}(C, i) \\ \qquad \wedge \begin{pmatrix} C(u_i).\mathtt{bullet} = 2 \\ \Rightarrow \mathit{Modest}(C, i) \end{pmatrix} \end{array} \right\}.$$

Lemma 3. *A modest bullet never becomes non-modest.*

Proof. Let C and C' be any configurations such that $C \to C'$ and b be a modest bullet in C. Assume for contradiction that b does not disappear and b becomes non-modest in $C \to C'$. Let u_i and $u_{i'}$ be the agents at which b is located in C and C', respectively ($i' \in \{i, i+1\}$). Since a peaceful bullet never becomes non-peaceful, b is still peaceful in C'. By definition of a modest bullet, in C, agent u_{i-j} satisfies $u_{i-j}.\mathtt{distL} \leq d_L(i - j)$ for all $j = 0, 1, \ldots, d_L(C, i)$. Thus, none of $u_i, u_{i-1}, \ldots, u_{i-d_L(C,i)+1}$ becomes a leader in $C \to C'$, Moreover, $u_{i-d_L(C,i)}$ is shielded in C and thus never becomes a follower in $C \to C'$. This yields that the nearest left leader of b does not change in $C \to C'$, i.e., $i - d_L(C, i) = i' - d_L(C', i')$. Therefore, for all $j = 0, 1, \ldots, d_L(C, i)$, u_{i-j} still satisfies $u_{i-j}.\mathtt{distL} \leq d_L(i - j)$ in C'. Since b is not modest in C', b must move u_i to u_{i+1} in $C \to C'$, and $u_{i+1}.\mathtt{distL} > d_L(C', i + 1)$ must hold in C'. However, in $C \to C'$, $u_{i+1}.\mathtt{distL}$ is updated to $u_i.\mathtt{distL} + 1 \leq d_L(C, i) + 1 = d_L(C', i + 1)$, a contradiction. □

Lemma 4. *A newly-fired live bullet is modest.*

Proof. Assume that a leader u_i fires a live bullet b at interaction (u_i, u_{i+1}) in $C \to C'$. Bullet b immediately disappears by Lines 12 and 15 if u_{i+1} is a leader or has a bullet in C. Otherwise, b moves to u_{i+1}. Then, $u_i.\mathtt{shield} = 1$, $u_i.\mathtt{distL} = 0$, $u_{i+1}.\mathtt{distL} = 1$, and $u_i.\mathtt{signal} = u_{i+1}.\mathtt{signal} = 0$ must hold in C', which yields that b is modest in C'. □

Lemma 5. *Let C be any configuration where all live bullets are modest and C' any configuration such that $C \to C'$. Then, a secure agent u_i becomes insecure in $C \to C'$ only if u_i interacts with u_{i-1} in $C \to C'$ and u_{i-1} is insecure in C.*

Proof. Assume that u_i becomes insecure in $C \to C'$. First, consider the case that no leader becomes a follower in $C \to C'$. Then, u_i must change the value of $u_i.\mathtt{distL}$ to become insecure. If u_i interacts with u_{i+1}, $u_i.\mathtt{distL}$ does not change or becomes zero (Line 1). Thus, u_i must interact with u_{i-1} and increase $u_i.\mathtt{distL}$ to a value greater than

$N - d_R(C',i) \geq N - d_R(C,i-1) + 1$, hence u_{i-1} must be insecure in C. Next, consider the case that a leader u_j becomes a follower in $C \to C'$. If $u_j \neq u_{i+d_R(C,i)}$, $d_R(C,i) = d_R(C',i)$ and $C(u_i).\texttt{distL} = C'(u_i).\texttt{distL}$ hold, which violates the assumption that u_i becomes insecure. Thus, $u_j = u_{i+d_R(C,i)}$ holds. However, this means that a modest bullet is located at $u_i, u_{i+1}, \ldots, u_{j-1}$ in C. Thus, by definition of a modest bullet, $C'(u_i).\texttt{distL} \leq C(u_i).\texttt{distL} \leq d_L(C,i) \leq N - d_R(C',i)$ holds. Hence, u_i is still secure in C', which violates the assumption. To conclude, u_i must interact with u_{i-1} in $C \to C'$ and u_{i-1} must be insecure in C. □

Lemma 6. *C_{NI} is closed.*

Proof. Immediately follows from Lemmas 2, 3, 4, and 5. □

Lemma 7. *No new leader is created in any execution starting from any configuration in C_{NI}.*

Proof. Since C_{NI} is closed by Lemma 6, every configuration that appears in an execution starting from a configuration in C_{NI} is also in C_{NI}. All agents are secure in a configuration in C_{NI}. Thus, $u_i.\texttt{distL} \leq N-1$ holds for all $u_i \in V$ and $u_i.\texttt{distL} = N-1$ holds only if u_{i+1} is a leader. Therefore, in a configuration in C_{NI}, no agent increases its distL to N, thus no new leader is created. □

Finally, we define $\mathcal{S}_{\mathrm{RL}}$ as the set of all configurations included in C_{NI} where there is exactly one leader.

Lemma 8. *$\mathcal{S}_{\mathrm{RL}}$ is closed and includes only safe configurations.*

Proof. Let C be any configuration in $\mathcal{S}_{\mathrm{RL}}$ and C' any configuration such that $C \to C'$. Since C_{NI} is closed by Lemma 6 and exactly one agent is a leader in C, it suffices to show that no one changes its output (i.e., the value of variable leader) in $C \to C'$. Since $C \in C_{\mathrm{PB}}$, the unique leader in C is never killed in $C \to C'$. By Lemma 7, no other agent becomes a leader in $C \to C'$. Thus, no agent changes its output in $C \to C'$. □

4.2 Convergence

In this subsection, we prove that an execution of P_{RL} starting from any configuration in $C_{\mathrm{all}}(P_{RL})$ reaches a configuration in $\mathcal{S}_{\mathrm{RL}}$ within $O(nN)$ steps in expectation. Formally, for any $C \in C_{\mathrm{all}}(P_{RL})$ and $\mathcal{S} \subseteq C_{\mathrm{all}}(P_{RL})$, we define $ECT(C,\mathcal{S})$ as the expected number of steps that execution $\Xi_{P_{RL}}(C,\mathbf{\Gamma})$ requires to reach a configuration in $\mathcal{S}$. The goal of this subsection is to prove $\max_{C \in C_{\mathrm{all}}(P_{RL})} ECT(C,\mathcal{S}_{\mathrm{RL}}) = O(nN)$. We give this upper bound by showing $\max_{C \in C_{\mathrm{all}}(P_{RL})} ECT(C,C_{\mathrm{NI}}) = O(nN)$ and $\max_{C \in C_{\mathrm{NI}}} ECT(C,\mathcal{S}_{\mathrm{RL}}) = O(n^2)$ in Lemmas 11 and 13, respectively.

In this subsection, we denote interaction (u_i, u_{i+1}) by e_i. In addition, for any two sequences of interactions $s = e_{k_0}, e_{k_1}, \ldots, e_{k_h}$ and $s' = e_{k'_0}, e_{k'_1}, \ldots, e_{k'_j}$, we define $s \cdot s' = e_{k_0}, e_{k_1}, \ldots, e_{k_h}, e_{k'_0}, e_{k'_1}, \ldots, e_{k'_j}$. That is, we use "$\cdot$" for the concatenation operator. For any sequence s of interactions and integer $i \geq 1$, we define s^i by induction: $s^1 = s$ and $s^i = s \cdot s^{i-1}$. For any $i, j \in \{0, 1, \ldots, n-1\}$, we define $seq_R(i,j) = e_i, e_{i+1}, \ldots, e_j$ and $seq_L(i,j) = e_i, e_{i-1}, \ldots, e_j$.

Definition 2. Let $\gamma = e_{k_1}, e_{k_2}, \ldots, e_{k_h}$ be a sequence of interactions. We say that γ occurs within l steps when $e_{k_1}, e_{k_2}, \ldots, e_{k_h}$ occurs in this order (not necessarily in a row) within l steps. Formally, the event "γ occurs within l steps from a time step t" is defined as the following event: $\Gamma_{t_i} = e_{k_i}$ holds for all $i = 1, 2, \ldots, h$ for some sequence of integers $t \leq t_1 < t_2 < \cdots < t_h \leq t + l - 1$. We say that from step t, γ completes at step $t + l$ if γ occurs within l steps but does not occur within $l - 1$ steps. When t is clear from the context, we write "γ occurs within l steps" and "γ completes at step l", for simplicity.

Lemma 9. *From any time step, a sequence $\gamma = e_{k_1}, e_{k_2}, \ldots, e_{k_h}$ with length h occurs within nh steps in expectation.*

Proof. For any interaction e_i, at each step, e_i occurs with probability $1/n$. Thus, e_i occurs within n steps in expectation. Therefore, γ occurs within nh steps in expectation. □

Lemma 10. *Let C be a configuration where no leader exists. In execution $\Xi = \Xi_{P_{RL}}(C,\mathbf{\Gamma})$, a leader is created within $O(nN)$ steps in expectation.*

Proof. Let $\gamma = (seq_R(0, n-1))^{\lceil N/n \rceil + 1}$. Since the length of γ is at most $N + 2n$, γ occurs within $3nN$ or fewer steps in expectation by Lemma 9. Thus, it suffices to show that a leader is created before γ completes in Ξ. Assume for contradiction that no leader is created before γ completes in Ξ.

First, consider the case that there is at least one bullet in C. Let B be the set of bullets that exist in C. Before γ completes, all bullets in B disappear by Lemma 1, while a bullet disappears only if it reaches a leader or another bullet as mentioned in Sect. 3.2. Thus, at least one bullet in B disappears by reaching a bullet not included in B, since there is no leader during the period by the above assumption. However, no bullet $b \notin B$ exists during the period because a follower never creates a new bullet. This is a contradiction.

Second, consider the case that there is no bullet in C. Let u_i be any agent with the minimum distL in C. Each time $seq_R(i, i-1)$ completes, $u_i.\texttt{distL}$ increases at least by n unless a new leader is created. Since $(seq_R(i, i-1))^{\lceil N/n \rceil}$ completes earlier than $\gamma = (seq_R(0, n-1))^{\lceil N/n \rceil + 1}$, a new leader is created before γ completes, a contradiction. □

Lemma 11. $\max_{C \in C_{\mathrm{all}}(P_{RL})} ECT(C,C_{\mathrm{NI}}) = O(nN)$.

Proof. Let C_0 be any configuration in $C_{\mathrm{all}}(P_{RL})$ and consider $\Xi = \Xi_{P_{RL}}(C_0,\mathbf{\Gamma})$. All bullets that exist in C_0 disappear before $\gamma = (seq_R(0, n-1))^{\lceil N/n \rceil + 1}$ completes by Lemma 1, while γ occurs within $O(nN)$ steps in expectation by Lemma 9. Thus, by Lemmas 3, 4 and 10, within $O(nN)$ steps in expectation, Ξ reaches a configuration C' where all live bullets are modest,

there is at least one leader, and every leader u_i satisfies $u_i.\texttt{distL} = 0$.

Let Ξ' be the suffix of Ξ after Ξ reaches C'. In the rest of this proof, we show that Ξ' reaches a configuration in C_{NI} within $O(n^2)$ steps. Since C_{PB} is closed (Lemma 2) and $C' \in C_{\mathrm{PB}}$, there is always at least one leader in Ξ'. Thus, by Lemmas 3 and 4, it suffices to show that Ξ' reaches a configuration where all agents are secure within $O(n^2)$ steps in expectation.

Here, we have the following two claims.

Claim 1. *In Ξ', once an agent u_i becomes a leader, u_i is always secure thereafter (even after it becomes a follower).*

Proof. Suppose that u_i is a leader in some point of Ξ'. At this time, $u_i.\texttt{distL} = 0$ (Lines 1 and 2). As long as u_i is a leader, u_i is secure. Agent u_i becomes a follower only when a live bullet reaches u_i (Line 11). In Ξ', all live bullets are modest. This yields that, when u_i becomes a follower, all agents $u_i, u_{i-1}, \ldots, u_{i-d_L(i)}$ are secure. Thus, letting $u_j = u_{i-d_L(i)}$, agent u_i is secure as long as u_j is secure. Similarly, u_j is secure as long as u_j is a leader. Even if u_j becomes a follower, there is a leader u_k such that u_j is secure as long as u_k is a leader, and so on. Therefore, u_i never becomes insecure. □

Claim 2. *In Ξ', an insecure agent u_i becomes secure if it interacts with u_{i-1} when u_{i-1} is secure.*

Proof. Let $C \to C'$ be any transition (u_{i-1}, u_i) that appears in Ξ such that u_{i-1} is secure in C. If u_{i-1} is a leader in C (thus in C'), $C'(u_i).\texttt{distL} \le 1 \le N - d_R(C', i)$. Otherwise, $d_R(C', i) = d_R(C', i-1) + 1$ must hold. By Lemma 5, u_{i-1} is still secure in C', hence $C'(u_i).\texttt{distL} \le C'(u_{i-1}).\texttt{distL} + 1 \le N - d_R(C', i-1) + 1 = N - d_R(C', i)$. Thus, u_i is secure in C' in both cases. □

Let u_i be a leader in C'. By Lemma 5 and Claims 1 and 2, all agents are secure when $seq_R(i, i-2)$ completes. This requires $O(n^2)$ expected steps by Lemma 9. □

Lemma 12. *Let C_0 be a configuration in C_{NI} where there are at least two leaders. Let u_i, u_j, and u_k be the leaders in C_0 such that $i = j - d_L(C_0, j)$ and $j = k - d_L(C_0, k)$, i.e., u_i is the nearest left leader of u_j and u_j is the nearest left leader of u_k in C_0. ($u_i = u_k$ may hold.) Let $d_1 = d_L(C_0, j)$ and $d_2 = d_L(C_0, k)$. Then, in an execution $\Xi = \Xi_{P_{RL}}(C_0, \mathbf{\Gamma}) = C_0, C_1, \ldots$, the event that one of the three leaders becomes a follower occurs within $O(n(d_1 + d_2))$ steps in expectation.*

Proof. Let T_{del} be the minimum integer t such that u_i, u_j, or u_k is a follower in C_t. Our goal is to prove $\mathbf{E}[T_{\mathrm{del}}] = O(n(d_1 + d_2))$. Let $V' = \{u_i, u_{i+1}, \ldots, u_k\}$. Consider the protocol, denoted by P'_{RL}, that can be obtained by removing Line 11 from P_{RL}. No leader becomes a follower in any execution of P'_{RL}. Let $\Xi' = \Xi_{P'_{RL}}(C_0, \mathbf{\Gamma}) = D_0, D_1, \ldots$. Of course, Ξ and Ξ' can be different. However, each agent in V' always has the same state both in Ξ and Ξ' before u_i, u_j, or u_k becomes a follower in Ξ. Formally, $C_t(u_s) = D_t(u_s)$ holds for all $u_s \in V'$ and all $t = 0, 1, \ldots, T_{\mathrm{del}} - 1$. Therefore, we have $T_{\mathrm{del}} \le T'_{\mathrm{del}}$, where T'_{del} is the minimum integer t such that a live bullet reaches u_j in $D_{t-1} \to D_t$ and u_j is unshielded at this interaction, i.e., $D_{t-1}(u_j).\texttt{shield} = 0$; because if u_j is still a leader in C_t, u_i or u_k must become a follower in $C_0, C_1, \ldots, C_{t-1}$. Thus, it suffices to show that $T'_{\mathrm{del}} = O(n(d_1 + d_2))$.

Again, no leader becomes a follower in Ξ'. In addition, no leader is created in Ξ' by Lemma 7. Leader u_j fires a bullet at least once in Ξ' before or when $\gamma = seq_R(j, k-1) \cdot seq_L(k-1, j) \cdot e_j$ completes. Thereafter, at any step, u_j is shielded with probability at least $1/2$ for the following reasons.

- Each time u_j fires a bullet, it fires a live bullet and becomes shielded with probability $1/2$, and fires a dummy bullet and becomes unshielded with probability $1/2$.
- Live bullets fired by u_j reach u_k not later than dummy bullets fired by u_j because live bullets are initially located at u_{j+1} when they are fired by u_j, while dummy bullets are initially located at u_j when they are fired by u_j.

After u_j fires a bullet for the first time, u_i fires a bullet at least once and the bullet reaches v_j in Ξ' before or when $\gamma' = seq_R(i, j-1) \cdot seq_L(j-1, i) \cdot seq_R(i, j-1)$ completes. This bullet is a live one with probability $1/2$, while u_j is unshielded at this time with probability at least $1/2$, as mentioned above. Thus, each time $\gamma \cdot \gamma'$ completes, the event that a live bullet reaches u_j at the time u_j is shielded occurs with probability at least $1/4$. Since $\gamma \cdot \gamma'$ occurs within $O(n(d_1 + d_2))$ steps in expectation by Lemma 9, we can conclude that $\mathbf{E}[T'_{\mathrm{del}}] = O(n(d_1 + s_2))$. □

Lemma 13. $\max_{C \in C_{\mathrm{NI}}} ECT(C, \mathcal{S}_{\mathrm{RL}}) = O(n^2)$.

Proof. Let C_0 be any configuration in C_{NI} and let $\Xi = \Xi_{P_{RL}}(C_0, \mathbf{\Gamma})$. By Lemmas 6 and 7, the number of leaders is monotonically non-increasing and never becomes zero in Ξ. For any real number x, define $\mathcal{L}_x$ as the set of configurations where the number of leaders is at most x. First, we prove the following claim.

Claim 3. *Let $\alpha = 12/11$. For any sufficiently large integer $k = O(1)$, if $C_0 \in (\mathcal{L}_{\alpha^{k+1}} \setminus \mathcal{L}_{\alpha^k}) \cap C_{\mathrm{NI}}$, execution Ξ reaches a configuration in $\mathcal{L}_{\alpha^k} \cap C_{\mathrm{NI}}$ within $O(n^2/\alpha^k)$ steps in expectation.*

Proof. Let $l_0, l_1, \ldots, l_{s-1} = u_{\pi_0}, u_{\pi_1}, \ldots, u_{\pi_{s-1}}$ be the leaders in C, where $\pi_0 < \pi_1 < \cdots < \pi_{s-1}$. We say that l_j and $l_{j+1 \bmod s}$ are neighboring leaders for each $j = 0, 1, \ldots, s-1$. Since there are s leaders in C_0, there are at least $3s/4$ leaders $l_i = u_{\pi_i}$ such that $d_L(\pi_{i+1 \bmod s}) \le 4n/s$. Thus, there are at least $n/2$ leaders $l_j = u_{\pi_j}$ such that $d_L(\pi_{j+1 \bmod s}) \le 4n/s$ and $d_L(\pi_{j+2 \bmod s}) \le 4n/s$ in C_0. Let S_L be the set of all such leaders. For each $l_j \in S_L$, by Lemma 12 and Markov inequality, l_j, $l_{j+1 \bmod s}$, or $l_{j+2 \bmod s}$ becomes a follower within $O(n^2/s)$ steps with probability $1/2$. Generally, if $X_0, X_1, \ldots, X_i$ are (possibly non-independent) events each of

which occurs with probability at least $1/2$, at least half of the events occur with probability at least $1/2$. Thus, with probability $1/2$, for at least half of the leaders l_j in S_L, the event that l_j, $l_{j+1 \bmod s}$, or $l_{j+2 \bmod s}$ becomes a follower occurs within $O(n^2/s)$ steps. Thus, at least $|S_L| \cdot (1/2) \cdot (1/3) = s/12$ leaders become followers within $O(n^2/s)$ steps with probability at least $1/2 = \Omega(1)$. Repeating this analysis, we observe that Ξ reaches a configuration in $\mathcal{L}_{\alpha^k} \cap C_{\mathrm{NI}}$ within $O(n^2/\alpha^k)$ steps in expectation. □

By Claim 3, for sufficiently large integer $k = O(1)$, the number of leaders becomes a constant (i.e., $O(\alpha^k) = O(1)$) within $\sum_{i=k}^{\lceil \log_\alpha n \rceil} O(n^2/\alpha^i) = O(n^2)$ steps in expectation in Ξ. Thereafter, by Lemma 12, the number of leaders decreases to one within $O(n^2)$ steps in expectation. □

Lemmas 8, 11, and 13 give the following main theorem.

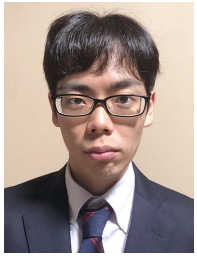
Theorem 1. *Given an integer N, P_{RL} is a self-stabilizing leader election protocol for any directed rings of any size $n \leq N$. The convergence time is $O(nN)$. The number of states is $O(N)$.*

5. Conclusion

We presented a self-stabilizing leader election protocol for directed rings in population protocols, given an upper bound N of the population size n. Specifically, an execution of the protocol starting from any initial configuration elects a unique leader within $O(nN)$ steps in expectation, by using $O(N)$ states per agent. If a given knowledge N is asymptotically tight, i.e., $N = O(n)$, this protocol is time-optimal.

References

[1] D. Yokota, Y. Sudo, and T. Masuzawa, "Time-optimal self-stabilizing leader election on rings in population protocols," Proc. 22nd International Symposium on Stabilizing, Safety, and Security of Distributed Systems, pp.301–316, 2020.
[2] D. Angluin, J. Aspnes, Z. Diamadi, M.J. Fischer, and R. Peralta, "Computation in networks of passively mobile finite-state sensors," Distrib. Comput., vol.18, no.4, pp.235–253, 2006.
[3] D. Angluin, J. Aspnes, M.J. Fischer, and H. Jiang, "Self-stabilizing population protocols," ACM Trans. Auton. Adapt. Syst., vol.3, no.4, pp.1–28, 2008.
[4] J. Beauquier, P. Blanchard, and J. Burman, "Self-stabilizing leader election in population protocols over arbitrary communication graphs," International Conference on Principles of Distributed Systems, pp.38–52, 2013.
[5] D. Canepa and M.G. Potop-Butucaru, "Stabilizing leader election in population protocols," http://hal.inria.fr/inria-00166632, 2007.
[6] H.P. Chen and H.L. Chen, "Self-stabilizing leader election," Proc. 38th ACM Symposium on Principles of Distributed Computing, pp.53–59, 2019.
[7] G. Cordasco and L. Gargano, "Space-optimal proportion consensus with population protocols," International Symposium on Stabilization, Safety, and Security of Distributed Systems, pp.384–398, 2017.
[8] G.B. Mertzios, S.E. Nikoletseas, C.L. Raptopoulos, and P.G. Spirakis, "Determining majority in networks with local interactions and very small local memory," International Colloquium on Automata, Languages, and Programming, pp.871–882, 2014.
[9] Y. Sudo, F. Ooshita, H. Kakugawa, and T. Masuzawa, "Loosely-stabilizing leader election on arbitrary graphs in population protocols," International Conference on Principles of Distributed Systems, pp.339–354, 2014.
[10] Y. Sudo, T. Masuzawa, A.K. Datta, and L.L. Larmore, "The same speed timer in population protocols," 36th IEEE International Conference on Distributed Computing Systems, pp.252–261, 2016.
[11] Y. Sudo, F. Ooshita, H. Kakugawa, and T. Masuzawa, "Loosely stabilizing leader election on arbitrary graphs in population protocols without identifiers or random numbers," IEICE Trans. Inf. & Syst., vol.E103-D, no.3, pp.489–499, March 2020.
[12] Y. Sudo, F. Ooshita, H. Kakugawa, T. Masuzawa, A.K. Datta, and L.L. Larmore, "Loosely-stabilizing leader election for arbitrary graphs in population protocol model," IEEE Trans. Parallel Distrib. Syst., vol.30, no.6, pp.1359–1373, 2018.
[13] E. Dijkstra, "Self-stabilizing systems in spite of distributed control," Commun. ACM, vol.17, no.11, pp.643–644, 1974.
[14] S. Cai, T. Izumi, and K. Wada, "How to prove impossibility under global fairness: On space complexity of self-stabilizing leader election on a population protocol model," Theory Comput. Syst., vol.50, no.3, pp.433–445, 2012.
[15] M.J. Fischer and H. Jiang, "Self-stabilizing leader election in networks of finite-state anonymous agents," International Conference on Principles of Distributed Systems, pp.395–409, 2006.
[16] T. Izumi, "On space and time complexity of loosely-stabilizing leader election," International Colloquium on Structural Information and Communication Complexity, pp.299–312, 2015.
[17] Y. Sudo, R. Eguchi, T. Izumi, and T. Masuzawa, "Time-optimal loosely-stabilizing leader election in population protocols," arXiv preprint arXiv:2005.09944, 2020.
[18] Y. Sudo, J. Nakamura, Y. Yamauchi, F. Ooshita, H. Kakugawa, and T. Masuzawa, "Loosely-stabilizing leader election in a population protocol model," Theor. Comput. Sci., vol.444, pp.100–112, 2012.
[19] Y. Sudo, F. Ooshita, H. Kakugawa, T. Masuzawa, A.K. Datta, and L.L. Larmore, "Loosely-stabilizing leader election with polylogarithmic convergence time," Theor. Comput. Sci., vol.806, pp.617–631, 2020.
[20] Y. Sudo, M. Shibata, J. Nakamura, Y. Kim, and T. Masuzawa, "The power of global knowledge on self-stabilizing population protocols," International Colloquium on Structural Information and Communication Complexity, pp.237–254, 2020.
[21] D. Angluin, J. Aspnes, and D. Eisenstat, "Fast computation by population protocols with a leader," Distrib. Comput., vol.21, no.3, pp.183–199, 2008.
[22] J. Burman, D. Doty, T. Nowak, E.E. Severson, and C. Xu, "Efficient self-stabilizing leader election in population protocols," arXiv preprint arXiv:1907.06068, 2019.
[23] H.P. Chen and H.L. Chen, "Self-stabilizing leader election in regular graphs," Proc. 39th Symposium on Principles of Distributed Computing, pp.210–217, 2020.



Daisuke Yokota received the B.E. degree in computer science from Osaka University in 2020. He is now a master's student at the Graduate School of Information Science and Technology, Osaka University. His research interests include distributed algorithms.

Yuichi Sudo received the B.E., M.E., and Ph.D. degrees in information science and technology from Osaka University in 2009, 2011, and 2015, respectively. He worked at NTT Corporation and was engaged in research on network security during 2011–2017. He was an assistant professor with the Graduate School of Information Science and Technology, Osaka University, during 2017–2021. He has been an associate professor at Faculty of Computer and Information Sciences, Hosei University, since April 2021. His research interests include distributed algorithms and graph theory. He is a member of IEEE and EATCS.

Toshimitsu Masuzawa received the B.E., M.E. and D.E. degrees in computer science from Osaka University in 1982, 1984 and 1987. He had worked at Osaka University during 1987–1994, and was an associate professor of Graduate School of Information Science, Nara Institute of Science and Technology (NAIST) during 1994–2000. He is now a professor of Graduate School of Information Science and Technology, Osaka University. He was also a visiting associate professor of Department of Computer Science, Cornell University between 1993–1994. His research interests include distributed algorithms, parallel algorithms and graph theory. He is a member of ACM, IEEE, IEICE and IPSJ.