Theoretical Computer Science 444 (2012) 100–112

Contents lists available at SciVerse ScienceDirect

Theoretical Computer Science

journal homepage: www.elsevier.com/locate/tcs

Loosely-stabilizing leader election in a population protocol model

Yuichi Sudo [a,*], Junya Nakamura [a], Yukiko Yamauchi [b], Fukuhito Ooshita [a], Hirotsugu Kakugawa [a], Toshimitsu Masuzawa [a]

[a] Graduate School of Information Science and Technology, Osaka University, 1-5 Yamadaoka, Suita, Osaka, 565-0871, Japan

[b] Graduate School of Information Science and Electrical Engineering, Kyushu University, 744 Motooka, Nishi-ku, Fukuoka 819-0395, Japan

ARTICLE INFO

Keywords:
Self-stabilization
Population protocol
Leader election
Loose-stabilization

ABSTRACT

A self-stabilizing protocol guarantees that starting from any arbitrary initial configuration, a system eventually comes to satisfy its specification and keeps the specification forever. Although self-stabilizing protocols show excellent fault-tolerance against any transient faults (e.g. memory crash), designing self-stabilizing protocols is difficult and, what is worse, might be impossible due to the severe requirements. To circumvent the difficulty and impossibility, we introduce a novel notion of *loose-stabilization*, that relaxes the closure requirement of self-stabilization; starting from any arbitrary configuration, a system comes to satisfy its specification in a relatively short time, and it keeps the specification *not forever but for a long time*. To show the effectiveness and feasibility of this new concept, we present a probabilistic loosely-stabilizing leader election protocol in the Probabilistic Population Protocol (PPP) model of complete networks. Starting from any configuration, the protocol elects a unique leader within $O(nN \log n)$ expected steps and keeps the unique leader for $\Omega(Ne^N)$ expected steps, where n is the network size (not known to the protocol) and N is a known upper bound of n. This result proves that introduction of the loose-stabilization circumvents the already-known impossibility result; the self-stabilizing leader election problem in the PPP model of complete networks cannot be solved without the knowledge of the exact network size.

1. Introduction

A distributed system is a collection of autonomous computational entities (processes) connected by communication links. Fault tolerance of distributed systems has attracted more and more attention since distributed systems are prone to faults. A *self-stabilizing system* [1,2] has a desirable property that, even when any transient fault (e.g. memory crash) hits the system, it can autonomously recover from the fault. The notion of self-stabilization is described as follows: (i) starting from any arbitrary initial configuration, a system eventually reaches a *safe configuration* (*convergence*), and (ii) once a system reaches a safe configuration, then it keeps its specification forever (*closure*). Although self-stabilizing systems provide excellent fault-tolerance as mentioned above, designing self-stabilizing protocols is difficult and, what is worse, might be impossible due to the severe requirements of self-stabilization.

To circumvent this difficulty and impossibility, many researchers have tried to relax the severe requirement of self-stabilization and proposed a lot of variants. *Probabilistic self-stabilization* [3] guarantees convergence to a safe configuration with probability 1 starting from any arbitrary configuration. *Quasi-stabilization* [4] guarantees convergence to a safe configuration only when all processes in the system start with the program counters of value 0. *Weak-stabilization* [5] guarantees that starting from any arbitrary configuration, there exists an execution that reaches a safe configuration.

* Corresponding author. Tel.: +81 6 6879 4118; fax: +81 6 6879 4119.
E-mail addresses: y-sudou@ist.osaka-u.ac.jp (Y. Sudo), junya-n@ist.osaka-u.ac.jp (J. Nakamura), y-yamauchi@is.naist.jp (Y. Yamauchi), f-oosita@ist.osaka-u.ac.jp (F. Ooshita), kakugawa@ist.osaka-u.ac.jp (H. Kakugawa), masuzawa@ist.osaka-u.ac.jp (T. Masuzawa).

0304-3975/$ – see front matter © 2012 Elsevier B.V. All rights reserved.
doi:10.1016/j.tcs.2012.01.007

Devismes et al. [6] investigated the relations among self, probabilistic and weak stabilization. A notable characteristic common to all the above variants is that they relax only the convergence requirement but not the closure requirement of self-stabilization. As we shall see later, we relax the closure requirement to introduce a novel notion of loose-stabilization.

In 2004, Angluin et al. [7] introduced the *population protocol* model, which represents wireless sensor networks of anonymous mobile sensing devices. This new model has many interesting properties and has been studied by a number of papers [7–13]. In this model, two devices (agents) communicate with each other only when they come sufficiently close to each other. (We call this event an *interaction*.) Then, each of the two agents change its state depending on its states and the states of the other agent. The population protocol model, for example, represents a flock of birds such that each bird is equipped with a sensing device of small transmission range.

In the population protocol model proposed in [7], a fairness condition called global fairness is assumed; Roughly speaking, any pair of agents must have interactions infinitely many times. Under this condition, the intervals of the interactions are not bounded at all: It may happen that agents u and v do not have an interaction while u' and v' have a interaction a thousand times. Therefore, the assumption of the global fairness is not strong enough to evaluate time complexity while it is strong enough to guarantee reachability to some configuration. Thus, in this paper, to evaluate time complexity (in terms of expected time), the global fairness is replaced by the assumption that, at each time, a pair of agents is selected for interaction at uniform random. This model is called the *Probabilistic Population Protocol* (PPP) and has been studied well in the literature [7,9,13]. In this paper, we adopt the PPP model because evaluating time is crucial in the concept of loose-stabilization we introduce later.

Self-stabilizing leader election in the population protocol model of complete networks is an important problem and has been considered by several papers [10–12]. Unfortunately, this problem is unsolvable if a protocol must work on the network of finite but any arbitrary size. (This fact is proven by Angluin et al. [10].[1]) Hence, if one wants to solve this problem, the additional condition or information is needed. Cai et al. [11] show that the information of the exact network size is necessary to solve the problem. In other words, for any two distinct positive integers n and n', there exists no self-stabilizing leader election protocol that works both on the complete network of size n and on that of size n'. Furthermore, they also show that the knowledge of the exact network size n is sufficient for the problem. Actually, their proposed protocol solves the self-stabilizing leader election with that knowledge and $O(\log n)$ memory space of each agent. Fischer and Jiang [12] use a failure detector (or oracle) $\Omega?$ to solve the problem with no knowledge of the network size. The detector $\Omega?$ tells each agent whether a leader exists or not among n agents of the network. What $\Omega?$ tells the agents is not always correct, but guaranteed to become correct eventually if a leader exists continuously, or if no leader exists continuously. The impossibility result of [11] holds even in the PPP model and even if we can use infinite space of each agent. That is, without knowledge of the exact network size, there is no probabilistic stabilizing leader election protocol even if the infinite space of each agent is available. This negative result is derived from the proof of [11] with slight modification. (The proof is attached in the appendix.)

Our contribution. To circumvent difficulty and impossibility in designing self-stabilizing protocols, we introduce a novel notion of *loose-stabilization*, which relaxes the closure requirement of self-stabilization. To the best of our knowledge, this is the first trial to relax the closure requirement and not the convergence requirement. Intuitively, the notion of loose-stabilization is described as follows: (i) starting from any arbitrary configuration, a system reaches a *loosely-safe configuration* within a short time (*convergence*), and (ii) once a system reaches a loosely-safe configuration, then it keeps its specification for a long time (*loose-closure*). In other words, we relax the closure requirement by allowing a system to deviate from its specification even after a loosely-safe configuration but only after a long period satisfying the specification. The requirement of fast convergence is added to guarantee that the specification is satisfied in most of the system running time. Actually, the loose-stabilization is practically equivalent to self-stabilization if the specification is kept for a significantly long time (e.g. exponential order with the network size) after the loosely-safe configuration.

Even though loose-stabilization relaxes the requirements of self-stabilization, from a practical perspective, the notion of loose-stabilization suits the purpose of fault-tolerance just as well, if not better. Self-stabilization has great importance for networks prone to faults, where the probability of fault occurrence is not negligible and faults occur repeatedly and intermittently: self-stabilizing protocols can recover from faults and work correctly during the fault-free periods. Since the length of the fault-free period is commonly estimated by, for example, MTBF (mean time between faults), the *permanent closure* of self-stabilization (to permanently satisfy the specification after convergence) seems to be an exaggerated requirement. To such a situation, loose-stabilization is particularly appropriate if, after convergence, it satisfies the specification in a sufficiently long period (compared to the MTBF).

Several definitions for the above notion can be formulated. In this paper, we give a concrete definition of *probabilistic loose-stabilization*, which ensures fast convergence and a long period of closure in terms of *expected time*.

To show effectiveness and feasibility of loose-stabilization, we present a probabilistic loosely-stabilizing leader election protocol in the PPP model of complete networks. The protocol uses the knowledge of an upper bound, say N, of the network size: the protocol works correctly on any complete network of size N or less. Starting from any arbitrary configuration, the protocol elects a unique leader within $O(nN \log n)$ expected steps, and then, keeps the unique leader for $\Omega(Ne^N)$ expected steps where n is the actual network size. This result discloses an evidence that introduction of the loose-stabilization can

[1] They prove this impossibility for a certain class of topology, called non-simple class. This class includes complete networks, directed line networks, and connected networks with a certain degree bound etc.

Table 1
Leader election protocols in the literature and in this paper

	Property	Information	Memory space	Model
[11]'s	self-stabilization	exact n	$O(\log n)$	original
[12]'s	self-stabilization	$\Omega?$	$O(1)$	original
ours	loose-stabilization	upper bound $N \geq n$	$O(\log N)$	PPP

circumvent impossibility results on self-stabilization; the self-stabilizing leader election in the PPP model of complete networks cannot be solved even in a probabilistic way without knowledge of the exact network size (as mentioned above).

In Table 1, our result and the other results that overcome the impossibility of self-stabilizing leader election is shown. Our result substantially relaxes the needed information of [11] while our solution costs a slightly larger space of agents than that of [11] and does not attain original self-stabilization. Unlike the protocol of [12], we do not use any kind of external entity such as a failure detector while the protocol of [12] preserves the traditional restriction on space of the population protocol: only constant space is available at the each agent. This restriction is introduced by [7] to implement a protocol on a sensor network of tiny devices. Although our protocol violates the restriction, this violation does not matter in a practical application because the consumption space or our algorithm is only log-space as well as [10,11]. Actually, the space of $\log N$ is only 20 bit even if N is one million, thus, the space can be considered small enough to be implemented at any device.

2. Preliminaries

In this section, we define the probabilistic population protocol model and the concept of probabilistic loose-stabilization. Throughout this paper, we use the notation $\mathrm{pre}_l(s)$ for describing the prefix of sequence s of length l.

A *population* consists of a collection of finite state sensing devices called *agents*. Each agent has its own state and updates the state by communication with other agents in pairs,[2] called *interactions*. We represent a population by simple directed graph $G(V, E)$: vertex set $V = \{0, 1, \ldots, n-1\}$ $(n \geq 2)$ represents the set of agents, and edge set $E \subseteq V \times V$ represents the set of possible interactions. If $(u, v) \in E$, agents u and v can interact (or communicate) with each other in such a way that u serves as an *initiator* and v serves as a *responder*. In this paper, we assume that a population $G(V, E)$ is a directed complete graph, that is, the edge set E is equal to $\{(u, v) \mid u, v \in V,\ u \neq v\}$.

A *protocol* $P(Q, Y, O, \delta)$ consists of a finite set Q of states, a finite set Y of output symbols, an output function $O : Q \to Y$, and a transition function $\delta : Q \times Q \to Q \times Q$. The *output of an agent* is determined by O: When the state of an agent is $p \in Q$, the output of the agent is $O(p)$. When an interaction between two agents happens, δ determines the next states of the two agents, the initiator and the responder. For agent u with state p and agent v with state q, the equation $\delta(p, q) = (p', q')$ indicates that the states of u (the initiator) and v (the responder) become p' and q' respectively after the interaction (u, v).

A *configuration* is a mapping $C : V \to Q$ that specifies the states of all agents in a population. We denote by $\mathcal{C}_{\mathrm{all}}(P)$ the set of all configurations of P. Let C and C' be configurations, and let u and v be distinct agents. We say that C changes to C' by an interaction $r = (u, v)$, denoted by $C \xrightarrow{r} C'$, if we have $(C'(u), C'(v)) = \delta(C(u), C(v))$ and $C'(w) = C(w)$ for all $w \in V$ except u and v.[3]

An *interaction sequence* $\gamma = (u_0, v_0), (u_1, v_1), \ldots$ is an infinite sequence of interactions. For each $t \geq 0$, we denote u_t and v_t by $\gamma_1(t)$ and $\gamma_2(t)$ respectively, and denote (u_t, v_t) by $\gamma(t)$. We call $\gamma(t)$ *the interaction at time t in* γ. We say that agent v *joins in* interaction $\gamma(t)$ when $v \in \{\gamma_1(t), \gamma_2(t)\}$.

An *execution* is an infinite sequence of configurations. Given an interaction sequence γ and an initial configuration C_0, the execution of protocol P is uniquely defined as $\Xi_P(C_0, \gamma) = C_0, C_1, \ldots$ such that $C_t \xrightarrow{\gamma(t)} C_{t+1}$ for all $t \geq 0$.

A scheduler determines which interaction happens at each time t $(t \geq 0)$. In this paper, we consider a uniformly random scheduler: the interaction at each time is chosen uniformly at random from all possible interactions. We represent the choice of this scheduler by the interaction sequence Γ: each $\Gamma(t)$ is a random variable such that $\Pr(\Gamma(t) = (u, v)) = 1/|E| = 1/(n(n-1))$ for any arbitrary interactions $(u, v) \in E$ and for any integer $t \geq 0$.

2.1. Specification

In this section, we introduce the concept of *specification* and define the specification of leader election problem.

For protocol $P(Q, Y, O, \delta)$ and configuration $C \in \mathcal{C}_{\mathrm{all}}(P)$, we view the composite function $O \circ C : V \to Y$ as the output of C and denote it by $O(C)$. For a sequence of configurations $T = C_0, C_1, \ldots$, we define output sequence $OT_P(T)$ as $O(C_0), O(C_1), \ldots$.

A *specification* of a problem defines the condition the output sequence should satisfy. Formally, a specification $SP(Y)$ is a set consisting of sequences of functions $V \to Y$. (We omit Y from the notation $SP(Y)$ when it is clear from the context.) Let $\Xi = C_0, C_1, \ldots$ be an execution of protocol P. We say that execution Ξ satisfies specification SP if and only

[2] This means that an agent can communicate simultaneously with only one agent.

[3] This definition implies that interactions between two agents happen sequentially, that is, exactly one pair of agents interact at any time.

if $OT_P(\Xi) \in SP$ holds. When $OT_P(\mathrm{pre}_{t+1}(\Xi)) \in SP$ holds, Ξ is considered to satisfy SP until time t. (Remind the index: $\mathrm{pre}_{t+1}(\Xi) = C_0, C_1, \ldots, C_t$.) In this paper, we assume that any specification SP is prefix closed: $X \in SP \Rightarrow \mathrm{pre}_l(X) \in SP$ holds for any positive integer l.

Definition 1 (*Leader Election Problem*)**.** We denote by le the set of all assignments $\omega : V \to \{F, L\}$ such that for some $v_l \in V$, $\omega(v_l) = L$ and for all $v \neq v_l$, $\omega(v) = F$. The leader election specification $LE(\{F, L\})$ is defined as $LE(\{F, L\}) = \{T = w^k \mid w \in le,\ 1 \leq k \leq \infty\}$ where w^k is the sequence of consecutive assignments w with length k, that is,

$$w^k = \underbrace{w, w, \ldots, w}_{k}.$$

Informally, $LE\{F, L\}$ requires that any legitimate execution has one *static* leader agent with the output symbol L and $n - 1$ non-leader (follower) agents with the output symbol F through its all configurations. Here, "static" means that the leader must continue to be a leader and any other agent must not become a leader during the execution. This specification requires that the outputs of all agents should be stable, but allows that agents continue to change their states. Moreover, the specification does not require termination detection. Since interactions happen an infinite amount of times, the execution continues forever and never terminates.

2.2. Probabilistic loose-stabilization

In this section, we define the notion of *probabilistic loose-stabilization*. For the proof of the impossibility result, we also define *probabilistic stabilization*.

Firstly, we define *holding time* $HT_P(\Xi, SP)$ for protocol $P(Q, Y, O, \delta)$, execution Ξ of P and specification $SP(Y)$. This represents how long Ξ satisfies SP from time 0. If $OT_P(\Xi) \in SP$ holds, then we define $HT_P(\Xi, SP) = \infty$. If $OT_P(\mathrm{pre}_1(\Xi)) \notin SP$ holds, then we define $HT_P(\Xi, SP) = 0$. Otherwise, we have some t such that $OT_P(\mathrm{pre}_t(\Xi)) \in SP$ and $OT_P(\mathrm{pre}_{t+1}(\Xi)) \notin SP$. Such t is uniquely determined from the prefix-closed property of SP. Then we define $HT_P(\Xi, SP) = t$.

Secondly, we define *convergence time* $CT_P(\Xi, \mathcal{C})$ for a set $\mathcal{C} \subseteq \mathcal{C}_{\mathrm{all}}(P)$ of configurations. This represents how long it takes for $\Xi = C_0, C_1, \ldots$ to reach a configuration in $\mathcal{C}$. If $C_0 \in \mathcal{C}$ holds, then we define $CT_P(\Xi, \mathcal{C}) = 0$. If $C_t \in \mathcal{C}$ does not hold for any time $t \geq 0$, then we define $CT_P(\Xi, \mathcal{C}) = \infty$. Otherwise, we have some t such that $C_t \notin \mathcal{C}$ and $C_t \in \mathcal{C}$. Then, we define $CT_P(\Xi, \mathcal{C}) = \min\{t \mid C_t \notin \mathcal{C} \wedge C_{t+1} \in \mathcal{C}\}$.

We denote $\mathbf{E}[HT_P(\Xi_P(C, \Gamma), SP)]$ by $EHT_P(C, SP)$ for any configuration $C \in \mathcal{C}_{\mathrm{all}}(P)$, where $\mathbf{E}[X]$ denotes the expected value of random variable X. Starting from C, the execution of P satisfies SP for $EHT_P(C, SP)$ expected time. Similarly, we denote $\mathbf{E}[CT_P(\Xi_P(D, \Gamma), \mathcal{C})]$ by $ECT_P(D, \mathcal{C})$ for any configuration $D \in \mathcal{C}_{\mathrm{all}}(P)$. Starting from D, the execution of P reaches a configuration in $\mathcal{C}$ within $ECT_P(D, \mathcal{C})$ expected time.

Definition 2 (*Probabilistic Loose-stabilization*)**.** Let α and β be real numbers ($\alpha \geq 0$, $\beta > 0$). A protocol $P(Q, Y, O, \delta)$ is (α, β)-probabilistic loosely-stabilizing for specification $SP(Y)$ if a nonempty set $\mathcal{S}$ of configurations exists such that:

$$\max_{C \in \mathcal{C}_{\mathrm{all}}(P)} ECT_P(C, \mathcal{S}) \leq \alpha,$$

$$\min_{C \in \mathcal{S}} EHT_P(C, SP) \geq \beta.$$

Notice that the second inequality implies any configuration $C \in \mathcal{S}$ is legitimate in the sense that the execution C (of length one) is contained in specification SP. We say that any configuration in S is loosely-safe. Intuitively, probabilistic loose-stabilization requires that any execution starting from any configuration reaches a loosely-safe configuration within time α, and after that, the execution satisfies the specification for time β (in terms of expected time). An (α, β)-probabilistic loosely-stabilizing protocol is quite useful if β is sufficiently large (e.g. exponential order with n) and α is relatively small (e.g. low polynomial order with n).

3. Protocol P_{LE}

In this section, we present a leader election protocol $P_{LE}(Q, \{F, L\}, O, \delta)$, which uses the knowledge of an upper bound N of the network size n. The protocol has a design parameter s. When s is adequately set depending on N, it is $(O(nN \log n), \Omega(Ne^N))$-probabilistic loosely-stabilizing for LE.

Each agent has one *leader bit* and a *timer* that takes an integer value in $[0, s]$, i.e. $Q = \{-, l\} \times \{0, 1, \ldots, s\}$. For state p, we denote the first element (leader bit) of p by $p.leader$ and the second element (timer) of p by $p.time$. The output function O is defined as follows: if the leader bit of an agent is l, then the output of the agent is L, otherwise F. We call an agent with the leader bit $l\,(-)$ a leader (non-leader, respectively). We describe the transition function δ by pattern rules in Fig. 1. Given any pair of states (p, q), the pair of the next states $\delta(p, q)$ is defined as follows: (i) if (p, q) matches the left side of exactly one rule, $\delta(p, q)$ is determined by the right side of the rule, and (ii) if there are two or more matched rules, we apply the rule with smallest rule number among them. The symbol $*$ means "don't care", that is, $*$ matches any value of the timer. Note that this five rules are collectively exhaustive.

$$
\begin{array}{lllll}
\text{R1.} & ((l,*),(l,*)) & \to & ((l,s),(-,s)) & \\
\text{R2.} & ((l,*),(-,*)) & \to & ((l,s),(-,s)) & \\
\text{R3.} & ((-,*),(l,*)) & \to & ((-,s),(l,s)) & \\
\text{R4.} & ((-,0),(-,0)) & \to & ((l,s),(-,s)) & \\
\text{R5.} & ((-,i),(-,j)) & \to & ((-,f),(-,f)) & \\
 & & & (0 \le i,j \le s,\ f = \max(i,j)-1) &
\end{array}
$$

Fig. 1. The transition function δ of P_{LE}.

If two leaders interact, the initiator remains a leader and the responder becomes a non-leader (R1). If a leader and a non-leader interact, the leader bits of both the agents do not change (R2, R3). In every interaction in which one or two leaders join, the timers of both the agents are reset to the full timer value s (R1, R2, and R3). We call this event *timer reset*. A new leader is created only when two non-leaders with timer value 0 interact (R4). We call this event *timeout*. If two non-leaders interact where either or both the agents have non-zero timer, then at least one of the two agents decrements its timer value by 1 (R5). R5 plays another role of *propagating the higher timer value*: intuitively, when two non-leaders interact, the timer of a lower value is set to the other (higher) value (minus 1).

In a configuration containing at least one leader, timeout rarely happens because of frequent occurrences of timer reset and propagation of higher timer value. On the other hand, in a configuration containing no leader, timeout happens in a relatively short time because of no possibility of timer reset. Hence, starting from any configuration, removing leaders by R1 or creating a leader by R4 eventually brings the population to a configuration with exactly one leader. The following two properties hold clearly.

Lemma 1. *Once a configuration with one or more leaders is reached, the number of leaders cannot become 0 thereafter.*

Lemma 2. *Once a unique leader is elected, specification LE holds until the next timeout happens.*

As a set of loosely-safe configuration, we adopt $\mathcal{S}_{\mathrm{half}}$, which consists of all the configurations in which exactly one leader exists and the timer value of every agent is greater than or equal to $s/2$. From the above explanation for P_{LE}, one can intuitively observe the following two properties: starting from any configuration, the population reaches a configuration in $\mathcal{S}_{\mathrm{half}}$ within a relatively short time (*convergence*), and once a configuration in $\mathcal{S}_{\mathrm{half}}$ is reached, the specification (the unique and static leader) is kept for an extremely long time (*loose-closure*). In Section 4, we show rigorously how fast P_{LE} converges to a loosely-safe configuration, and how long P_{LE} maintains the specification of leader election after a loosely-safe configuration is reached.

Discussion. The idea of our protocol is similar to that of Fisher and Jiang's protocol [12]. Their protocol uses a failure detector $\Omega?$, which tells each agent whether the leader exists or not among the n agents of the population. The idea of their protocol is roughly as follows: (i) when two leaders meet, one of them remains a leader, and the other becomes a non-leader, (ii) when a leader and a non-leader meet, they do not change anything, and (iii) when two non-leaders meet, they do not change anything if $\Omega?$ tells them a leader exists in the population; otherwise, one of the two becomes a leader. If we view the timeout mechanism of our protocol as a detector that reports whether a leader exists or not, then our protocol and their protocol [12] are the same. Actually, the authors of [12] mentioned that, in a practical application, timeout mechanism is effective for implementing $\Omega?$. However, in theory, our timeout mechanism and any other mechanism cannot provide $\Omega?$ accurately due to the impossibility result as mentioned in Section 1.

4. Analysis and proofs

Assume that we set design parameter s so that s is a multiple of 96 and $s \ge 3n$. (For simplicity, we use the notation $s^* = s/96$.) In this section, we prove that under this assumption, P_{LE} is $(O(ns \log n), \Omega(se^{s^*}))$-probabilistic loosely-stabilizing for LE. To claim it, we prove the following two expressions:

$$\max_{C \in \mathcal{C}_{\mathrm{all}}(P_{LE})} ECT_{P_{LE}}(C, \mathcal{S}_{\mathrm{half}}) = O(ns \log n), \tag{1}$$

$$\min_{C \in \mathcal{S}_{\mathrm{half}}} EHT_{P_{LE}}(C, LE) = \Omega\left(s \cdot e^{s^*}\right). \tag{2}$$

In this section, we omit P_{LE} from some expressions when the protocol under consideration is clear from the context; for example we denote $\mathcal{C}_{\mathrm{all}}(P_{LE})$, $OT_{P_{LE}}$ and $ECT_{P_{LE}}$ simply by $\mathcal{C}_{\mathrm{all}}$, OT and ECT respectively. And, we use the following four subsets $\mathcal{L}_{\mathrm{one}}$, $\mathcal{L}$, $\mathcal{C}_{\mathrm{half}}$ and $\mathcal{L}_{\mathrm{half}}$ of $\mathcal{C}_{\mathrm{all}}$ in addition to $\mathcal{S}_{\mathrm{half}}$;

$$\mathcal{L}_{\mathrm{one}} = \{C \in \mathcal{C}_{\mathrm{all}} \mid \#l(C) = 1\},$$
$$\mathcal{L} = \{C \in \mathcal{C}_{\mathrm{all}} \mid \#l(C) \ge 1\},$$
$$\mathcal{C}_{\mathrm{half}} = \left\{C \in \mathcal{C}_{\mathrm{all}} \mid \forall v \in V,\ C(v).time \ge \frac{s}{2}\right\},$$
$$\mathcal{L}_{\mathrm{half}} = \mathcal{L} \cap \mathcal{C}_{\mathrm{half}},$$

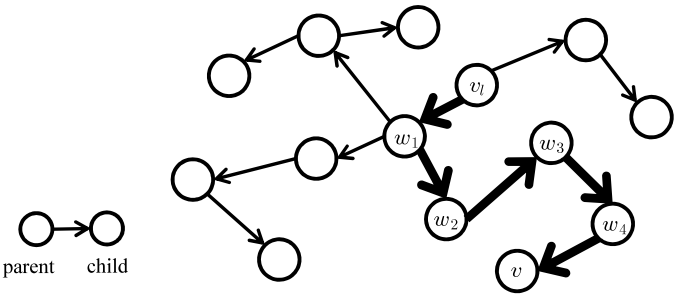

Fig. 2. The tree made from parent–child relation of infection: virtual agent of v' migrates from v_l to v through the path $v_l \to w_1 \to w_2 \to w_3 \to w_4 \to v$ (depicted by **bold face**).

where $\#l(C)$ represents the number of leaders in configuration C, i.e. $\#l(C) = |\{v \in V | v.leader = l\}|$. The set $\mathcal{L}_{\text{one}}$ represents the set of all configurations in which exactly one leader exists while $\mathcal{L}$ represents the set of all configurations in which one or more leaders exist. The set $\mathcal{C}_{\text{half}}$ represents the set of all configurations in which the timer value of every agent is greater than or equal to $s/2$. Note that $\mathcal{S}_{\text{half}}$ is equal to $\mathcal{L}_{\text{one}} \cap \mathcal{C}_{\text{half}}$.

In the rest of this section, firstly, we introduce the notion of *epidemic* (presented in [9]) and *virtual agents* in Section 4.1. Next, by using these tools, we prove Eq. (2) in Section 4.2 and Eq. (1) in Section 4.3.

4.1. Epidemic and virtual agents

In this section, we introduce the notion of *epidemic* (presented in [9]) and *virtual agents*.

To begin with, the notion of epidemic is introduced. Let C_0 be a configuration in $\mathcal{L}_{\text{one}}$, and let $v_l \in V$ be the unique leader in C_0. Let γ be an interaction sequence. The *epidemic function* $I_{C_0,\gamma}(t)$ $(t = 0, 1, \dots)$ that returns a set of agents is defined as follows: $I_{C_0,\gamma}(0) = \{v_l\}$, and $I_{C_0,\gamma}(t) = I_{C_0,\gamma}(t-1) \cup Add_{C_0,\gamma}(t-1)$ for any $t \geq 1$ where $Add_{C_0,\gamma}(i)$ is defined as

$$Add_{C_0,\gamma}(i) = \begin{cases} \{\gamma_1(i), \gamma_2(i)\} & \text{if } \gamma_1(i) \in I_{C_0,\gamma}(i) \ \vee \ \gamma_2(i) \in I_{C_0,\gamma}(i) \\ \emptyset & \text{otherwise} \end{cases}$$

for any integer $i \geq 0$. We say that if $v \in I_{C_0,\gamma}(t)$, then v is *infected* at time t in the epidemic starting from C_0 and spreading under γ, otherwise v is *infection-free* at time t in the epidemic. At time 0, only v_l is infected. An infection-free agent becomes infected when it interacts with an infected agent. Once an agent becomes infected, it remains infected thereafter. We define the *infected time* $T_{C_0,\gamma}(v)$ of agent $v \neq v_l$ as the integer $t \geq 0$ that satisfies $v \notin I_{C_0,\gamma}(t)$ and $v \in I_{C_0,\gamma}(t+1)$. We naturally define $T_{C_0,\gamma}(v_l) = 0$.

In the following, we define the *virtual agent* $VA_{C_0,\gamma}(v)$ of each agent $v \in V$. We assume that all the agents eventually become infected, that is, $I_{C_0,\gamma}(t') = V$ holds for some $t' \geq 0$. The virtual agent $VA_{C_0,\gamma}(v)$ is not defined if no such t' exists for C_0 and γ. Let v be any agent other than v_l. We define the *parent* of v as the agent that infects v in time $T_{C_0,\gamma}(v)$. This parent–child relation uniquely makes the rooted spanning tree, the root of which is v_l. In this tree, the path from v_l to v, $v_l = w_0 \to w_1 \to w_2 \to \cdots \to w_m = v$, uniquely exists. The virtual agent $VA_{C_0,\gamma}(v)$ is a virtual entity that migrates from v_l to v through the path (Fig. 2). This notion is formalized as *the location of the virtual agent* $L_{C_0,\gamma}(v,t)$ $(t \geq 0)$, which is defined as follows:

$$L_{C_0,\gamma}(v,t) = \begin{cases} v_l & (0 \leq t \leq t_1) \\ w_i & (t_i + 1 \leq t \leq t_{i+1},\ 1 \leq i \leq m-1) \\ v & (t \geq t_m + 1 = T_{C_0,\gamma}(v) + 1), \end{cases}$$

where $t_i = T_{C_0,\gamma}(w_i)$. For the leader agent v_l, we define $L_{C_0,\gamma}(v_l, t) = v_l$ for any $t \geq 0$.

Let v be an agent in V.[4] For simplicity, we denote the virtual agent $VA_{C_0,\gamma}(v)$ by v' here. We say that the virtual agent v' joins in interaction $\gamma(t)$ if agent $L_{C_0,\gamma}(v,t)$ joins in $\gamma(t)$, and we define indicator variable $VJ_{C_0,\gamma}(v,t)$ for any $t \geq 0$ as follows: if v' joins in $\gamma(t)$, then $VJ_{C_0,\gamma}(v,t) = 1$, otherwise $VJ_{C_0,\gamma}(v,t) = 0$. *The number of virtual interactions* of v, denoted by $VI_{C_0,\gamma}(v,t)$, is defined as follows:

$$VI_{C_0,\gamma}(v,t) = \sum_{i=0}^{t-1} VJ_{C_0,\gamma}(v,i).$$

Intuitively, $VI_{C_0,\gamma}(v,t)$ is the number of interactions in which v' joins between time 0 and time $t-1$.

Thanks to propagation of a higher timer value, the virtual agent v' brings a large timer value to v with high probability when v' reaches v through the infecting path. Actually, the timer of v at time $T_{C_0,\gamma}(v) + 1$ (just after v is infected) is at least

[4] Note that v can be v_l.

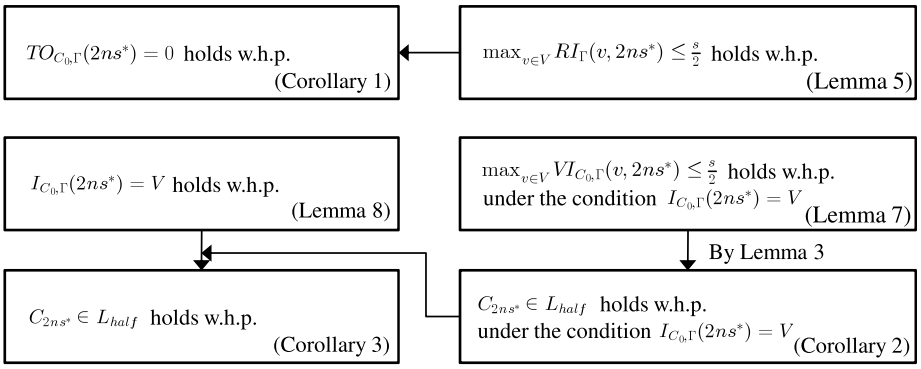

Fig. 3. The structure of the proof for Eq. (3): w.h.p. means "with high probability".

$s - VI_{C_0,\gamma}(v, T_{C_0,\gamma}(v) + 1)$ because one interaction decrements the timer of v' by at most 1. The following lemma trivially holds.

Lemma 3. *Let C_0 be a configuration in $\mathcal{L}_{\text{one}}$ and let γ be an interaction sequence. Let $\Xi(C_0, \gamma) = C_0, C_1, \ldots$. The following predicate holds for any integer $t \ge 0$:*

$$I_{C_0,\gamma}(t) = V \;\Rightarrow\; \forall v \in V, C_t(v).time \ge s - VI_{C_0,\gamma}(v, t).$$

If timeout happens, another new leader is created. This leader may change v_l to be a non-leader. Note that the above lemma holds even if such a situation happens, and v_l becomes a non-leader.

In addition to the number of virtual interactions, we define *the number of real interactions* of v as $RI_\gamma(v, t) = \sum_{i=0}^{t-1} RJ_\gamma(v, i)$, where $RJ_\gamma(v, t)$ is indicator variable such that if v joins in $\gamma(t)$ then $RJ_\gamma(v, t) = 1$, otherwise $RJ_\gamma(v, t) = 0$. Intuitively, $RI_\gamma(v, t)$ is the number of interactions in which v joins between time 0 and time $t - 1$.

4.2. Expected holding time

In this section, we prove Eq. (2). For a positive integer t, a configuration $C_0 \in \mathcal{C}_{\text{all}}$ and an interaction sequence γ, we define indicator variable $TO_{C_0,\gamma}(t)$ as follows: if timeout happens in execution $\Xi_{P_{LE}}(C_0, \gamma)$ between time 0 and time $t - 1$, that is, at least one of the interactions $\gamma(0), \gamma(1), \ldots, \gamma(t-1)$ causes timeout in $\Xi(C_0, \gamma)$, then $TO_{C_0,\gamma}(t) = 1$, otherwise $TO_{C_0,\gamma}(t) = 0$. For convenience, we define $TO_{C_0,\gamma}(0) = 0$. As a sufficient condition for Eq. (2), we focus on the following inequality for any configuration C_0 in $\mathcal{S}_{\text{half}}$:

$$\Pr\left(TO_{C_0,\Gamma}(2ns^*) = 0 \;\wedge\; C_{2ns^*} \in \mathcal{S}_{\text{half}}\right) \ge 1 - 2n \cdot e^{-s^*}, \tag{3}$$

where $\Xi(C_0, \Gamma) = C_0, C_1, \ldots, C_{2ns^*}, \ldots$.

Lemma 4. *Expression Eq. (2) holds if Eq. (3) holds for any configuration C_0 in $\mathcal{S}_{\text{half}}$.*

Proof. Assume that Eq. (3) holds for any configuration C_0 in $\mathcal{S}_{\text{half}}$. Then, from Lemma 2, the following inequality holds:

$$EHT(C_0, LE) \ge \left(1 - 2n \cdot e^{-s^*}\right)\left(2ns^* + \min_{C\in\mathcal{S}_{\text{half}}} EHT(C, LE)\right).$$

Since C_0 is *any* configuration in $\mathcal{S}_{\text{half}}$, we have

$$\min_{C\in\mathcal{S}_{\text{half}}} EHT(C, LE) \ge \left(1 - 2n \cdot e^{-s^*}\right)\left(2ns^* + \min_{C\in\mathcal{S}_{\text{half}}} EHT(C, LE)\right).$$

Solving this inequality gives us Eq. (2). □

In the following, we show that Eq. (3) holds for any configuration $C_0 \in \mathcal{S}_{\text{half}}$. The structure of the proof is shown in Fig. 3. Firstly, we bound the number of real interactions probabilistically (Lemma 5), from which we get the result that the probability of $TO_{C_0,\Gamma}(2ns^*) = 0$ is sufficiently close to 1 (Corollary 1). Secondly, we bound the number of virtual interactions probabilistically (Lemma 7). From this and Lemma 3 together, we prove that $C_{2ns^*} \in \mathcal{L}_{\text{half}}$ holds with sufficiently high probability under the condition $I_{C_0,\Gamma}(2ns^*) = V$ (Corollary 2). We also prove that $I_{C_0,\Gamma}(2ns^*) = V$ holds with sufficiently high probability (Lemma 8). By Corollary 2 and Lemma 8, we get the result that the probability of $C_{2ns^*} \in \mathcal{L}_{\text{half}}$ is sufficiently close to 1 (Corollary 3). The combination of Corollaries 1 and 3 directly leads to Eq. (3).

Lemma 5. $\Pr(\max_{v\in V} RI_\Gamma(v, 2ns^*) \le \frac{s}{2}) \ge 1 - n \cdot \exp(-24s^*)$.

Proof. At each time t, any agent v joins in $\Gamma(t)$ with probability $\frac{2}{n}$. Hence, $RI_\Gamma(v, 2ns^*)$ is a binomial random variable, and $RI_\Gamma(v, 2ns^*) \sim B(2ns^*, \frac{2}{n})$ holds. As one of the Chernoff bounds, $\Pr(Y \geq R) \leq 2^{-R}$ holds for any binomial random variable Y and any real number $R \geq 6 \cdot \mathbf{E}[Y]$ [14, (4.3)]. Since $\frac{s}{2} \geq 6\mathbf{E}[RI_\Gamma(v, 2ns^*)] = \frac{s}{4}$ and $2^{-1/2} < e^{-1/4}$, we obtain

$$\Pr\left(RI_\Gamma\left(v, 2ns^*\right) \geq \frac{s}{2}\right) \leq 2^{-s/2} < \exp\left(-\frac{s}{4}\right) = \exp(-24s^*).$$

We achieve the lemma by summing up all the above probabilities with respect to $v \in V$. □

Corollary 1. *The inequality* $\Pr(TO_{C_0,\Gamma}(2ns^*) = 0) \geq 1 - n \cdot \exp(-24s^*)$ *holds for any configuration* C_0 *in* $\mathcal{C}_{\text{half}}$.

Proof. Since $C_0 \in \mathcal{C}_{\text{half}}$, timeout happens by time $2ns^* - 1$ (i.e. $TO_{C_0,\Gamma}(2ns^*) = 1$) only when some agent joins in at least $\frac{s}{2} + 1$ interactions between time 0 and time $2ns^* - 1$. Hence, the corollary follows from Lemma 5. □

Next, in Lemma 7, we bound the number of virtual interactions probabilistically. Apparently, it seems that $VJ_{C_0,\Gamma}(v, t)$ and $RJ_\Gamma(v, t)$ have the same probability distribution for any $v \in V$ and $t \geq 0$. However, this is not true. Surprisingly, the interaction at time t influences the location of the virtual agent of v at the same time t. Hence, $\Pr(VJ_{C_0,\Gamma}(v, t) = 1) = \Pr(L_{C_0,\Gamma}(v, t) \in \{\Gamma_1(t), \Gamma_2(t)\})$ is not equal to $\frac{2}{n}$ and very hard to calculate. Therefore, we must take a different approach for Lemma 7 from that of Lemma 5. To begin with, we introduce the following lemma as a tool.

Lemma 6. *Let* C_0 *be a configuration in* $\mathcal{L}_{\text{one}}$ *and let* $X(i, p)$ *be a binomial random variable such that* $X(i, p) \sim B(i, p)$. *Then, the following expression holds for any* $v \in V$ *and any integers* $t \geq n$ *and* $j \geq 0$*:*

$$\Pr(VI_{C_0,\Gamma}(v, t) \geq j + n - 1 \mid I_{C_0,\Gamma}(t) = V) \leq \Pr(X(t, 4/n) \geq j).$$

Proof. Assume $I_{C_0\Gamma}(t) = V$ and let $v_l \in V$ be the unique leader in C_0. We define the *infecting time set IT* as $\bigcup_{v \in V \setminus \{v_l\}} \{T_{C_0,\Gamma}(v)\}$, and the *non-infecting time set NIT* as $\{0, 1, \ldots, t-1\} \setminus IT$. Let v be any agent in V, and let $NVI = \sum_{t' \in NIT} VJ_{C_0,\Gamma}(v, t')$. Since $|IT| = n - 1$, the inequality $VI_{C_0,\Gamma}(v, t) \leq NVI + n - 1$ immediately follows. Therefore, it is sufficient for our proof to show $\Pr(NVI \geq j \mid I_{C_0,\Gamma}(t) = V) \leq \Pr(X(t, 4/n) \geq j)$.

Let t' be any integer in $\{0, 1, \ldots, t-1\}$ and let $m = |I_{C_0,\Gamma}(t')|$. Consider the case $t' \in NIT$. Then, the interaction $\Gamma(t')$ must be an interaction such that both agents $\Gamma_1(t')$ and $\Gamma_2(t')$ belong to $I_{C_0,\Gamma}(t')$ or both the agents belong to $V \setminus I_{C_0,\Gamma}(t')$. Otherwise, some infection-free agent becomes infected at time t', contradicting $t' \notin IT$. And, $L_{C_0,\Gamma}(v, t') \in I_{C_0,\Gamma}(t')$ clearly holds by the definition of virtual agents. Thus, letting ${}_0C_2 = {}_1C_2 = 0$, we have

$$\begin{aligned}\Pr(VJ_{C_0,\Gamma}(v, t') = 1 \mid I_{C_0,\Gamma}(t) = V \wedge t' \in NIT) &= \frac{m-1}{{}_mC_2 + {}_{n-m}C_2} \\ &\leq \frac{4}{n}.\end{aligned}$$

See Lemma B.1 for the last inequality.

Note that this upper bound $4/n$ of the probability is independent from any interaction at any time other than t'. Hence, for any set S consisting of $t - n + 1$ distinct integers in $[0, t-1]$, we have

$$\Pr\left(NVI \geq j \mid I_{C_0,\Gamma}(t) = V \wedge NIT = S\right) \leq \Pr\left(X\left(t - n + 1, \frac{4}{n}\right) \geq j\right).$$

Therefore, the following inequality holds and so does the lemma.

$$\begin{aligned}\Pr(NVI \geq j \mid I_{C_0,\Gamma}(t) = V) &\leq \Pr\left(X\left(t - n + 1, \frac{4}{n}\right) \geq j\right) \\ &\leq \Pr\left(X\left(t, \frac{4}{n}\right) \geq j\right). \quad \square\end{aligned}$$

Lemma 7. *Let* C_0 *be a configuration in* $\mathcal{L}_{\text{one}}$. *The following inequality holds:*

$$\Pr\left(\max_{v \in V} VI_{C_0,\Gamma}\left(v, 2ns^*\right) \leq \frac{s}{2} \mid I_{C_0,\Gamma}\left(2ns^*\right) = V\right) \geq 1 - n \cdot \exp\left(-\frac{8s^*}{3}\right). \tag{4}$$

Proof. Let v be any agent, and let $X(i, p)$ be a binomial variable such that $X(i, p) \sim B(i, p)$. By Lemma 6 and the assumption $s \geq 3n$, we have

$$\begin{aligned}&\Pr\left(VI_{C_0,\Gamma}\left(v, 2ns^*\right) \geq \frac{s}{2} \mid I_{C_0,\Gamma}\left(2ns^*\right) = V\right) \\ &\leq \Pr\left(VI_{C_0,\Gamma}\left(v, 2ns^*\right) \geq \frac{s}{6} + n - 1 \mid I_{C_0,\Gamma}\left(2ns^*\right) = V\right) \ \because \frac{s}{2} \geq \frac{s}{6} + n - 1 \\ &\leq \Pr\left(X\left(2ns^*, \frac{4}{n}\right) \geq 16s^*\right).\end{aligned}$$

As one of the Chernoff bounds, $\Pr(Y \geq (1+\delta)\mathbf{E}[Y]) \leq \exp(-\delta^2\mathbf{E}[Y]/3)$ holds for any binomial random variable Y and any real number δ ($0 \leq \delta \leq 1$) [14, (4.2)], it follows that $\Pr(X(2ns^*, 4/n) \geq 16s^*) \leq \exp(-8s^*/3)$. (We set $\delta = 1$.) We obtain Eq. (4) by summing up all the probabilities with respect to $v \in V$. □

The following corollary is directly obtained from Lemmas 3 and 7.

Corollary 2. *Let C_0 be a configuration in $\mathcal{L}_{\text{one}}$ and let $\Xi(C_0, \Gamma) = C_0, C_1, \ldots$. Then, $\Pr(C_{2ns^*} \in \mathcal{L}_{\text{half}} \mid I_{C_0,\Gamma}(2ns^*) = V) \geq 1 - n \cdot \exp(-8s^*/3)$ holds.*

Lemma 8. $\Pr(I_{C_0,\Gamma}(2ns^*) = V) \geq 1 - n \cdot \exp(-s^*)$ *holds for any $C_0 \in \mathcal{L}_{\text{one}}$.*

Proof. For each k ($2 \leq k \leq n$), we define $T(k)$ as integer t such that $|I_{C_0,\Gamma}(t-1)| = k-1$ and $|I_{C_0,\Gamma}(t)| = k$, and define $T(1) = 0$. Intuitively, $T(k)$ is the first time at which there exists k infected agents in the population. Let $X_{\text{pre}} = T(\lceil \frac{n+1}{2} \rceil)$ and $X_{\text{post}} = T(n) - T(n - \lceil \frac{n+1}{2} \rceil + 1)$. Angluin et al. found in [9] that $T(k)$ and $T(n) - T(n-k+1)$ have the same probability distribution for any k ($1 \leq k \leq n$). Hence, so do X_{pre} and X_{post}. And, $X_{\text{pre}} + X_{\text{post}} \geq T(n)$ holds because $\lceil \frac{n+1}{2} \rceil \geq n - \lceil \frac{n+1}{2} \rceil + 1$. We denote $T(n - \lceil \frac{n+1}{2} \rceil + 1)$ by T_{half} and let $X_v = \max(T_{C_0,\Gamma}(v) - T_{\text{half}}, 0)$ for any agent v. Informally, X_v is the number of interactions that occurs between time T_{half} and the time at which agent v becomes infected. Consider the case $v \notin I_{C_0,\Gamma}(T_{\text{half}})$. At any time $t \geq T_{\text{half}}$, at least $n - \lceil \frac{n+1}{2} \rceil + 1$ ($\geq \frac{n}{2}$) agents are infected. Therefore, each interaction at time $t \geq T_{\text{half}}$ infects v with the probability of at least $\frac{1}{{}_nC_2} \cdot \frac{n}{2} \geq \frac{1}{n}$, and hence, we have $\Pr(X_v > ns^*) \leq (1 - \frac{1}{n})^{ns^*} \leq e^{-s^*}$. Since the number of infection-free agents at time T_{half} is at most $\frac{n}{2}$, $\Pr(X_{\text{post}} > ns^*) \leq \Pr(\bigvee_{v \in V}(X_v > ns^*)) \leq \frac{n}{2} \cdot \exp(-s^*)$ holds. By the equivalence of the distribution of X_{pre} and X_{post}, we have

$$\Pr\left(I_{C_0,\Gamma}\left(2ns^*\right) \neq V\right) \leq \Pr\left(X_{\text{pre}} > ns^*\right) + \Pr\left(X_{\text{post}} > ns^*\right) \leq n \cdot e^{-s^*}. \quad \square$$

Corollary 2 and Lemma 8 together lead to the following corollary.

Corollary 3. *Let C_0 be a configuration in $\mathcal{L}_{\text{one}}$ and let $\Xi(C_0, \Gamma) = C_0, C_1, \ldots, C_{2ns^*}, \ldots$. Then, $\Pr(C_{2ns^*} \in \mathcal{L}_{\text{half}}) \geq 1 - n \cdot \exp(-8s^*/3) - n \cdot \exp(-s^*)$ holds.*

Lemma 9. *Expression Eq. (2) holds.*

Proof. We have $\exp(-24s^*) + \exp(-\frac{8s^*}{3}) \leq \exp(-s^*)$ for any $s^* \geq 1$. Therefore, Eq. (3) holds for any configuration $C_0 \in \mathcal{S}_{\text{half}}$ from Corollaries 1 and 3. We achieve Eq. (2) by Lemma 4. □

4.3. Expected convergence time

Next, we show Eq. (1) to complete our proof. The following inequality clearly holds:

$$\max_{C \in \mathcal{C}_{\text{all}}} ECT(C, \mathcal{S}_{\text{half}}) \leq \max_{C \in \mathcal{C}_{\text{all}}} ECT(C, \mathcal{L}) + \max_{C \in \mathcal{L}} ECT(C, \mathcal{L}_{\text{half}}) + \max_{C \in \mathcal{L}_{\text{half}}} ECT(C, \mathcal{S}_{\text{half}}). \tag{5}$$

Therefore, it suffices to show that each term in the right side of Eq. (5) belongs to $O(ns \log n)$. We show that the first term belongs to $O(ns \log n)$ in Lemma 10, and the second and the third term belongs to $O(ns)$ in Lemmas 11 and 15 respectively.

Lemma 10. $\max_{C \in \mathcal{C}_{\text{all}}} ECT(C, \mathcal{L})$ *belongs to $O(ns \log n)$.*

Proof. We define $\nu(C, i)$ ($0 \leq i \leq s$) as the number of agents with timer value i in configuration C, i.e. $\nu(C, i) = |\{v \in V \mid C(v).time = i\}|$. For any integer i, j ($0 \leq i \leq s,\ 1 \leq j \leq n$) we denote by $\mathcal{W}_{i,j}$ the set of all configurations in which there exists no leader, the maximum timer value of all agents is i, and $\nu(C, i) = j$ holds. Note that $\mathcal{W}_{0,j}$ is empty when $j \neq n$.

We denote $\max_{C \in \mathcal{W}_{i,j}} ECT(C, \mathcal{C}_{\text{all}} \setminus \mathcal{W}_{i,j})$ by $w_{i,j}$. The execution starting from $\mathcal{W}_{i,j}$ gets out of $\mathcal{W}_{i,j}$ if one of the j agents with the largest timer value joins in an interaction. Furthermore, after that, the execution never comes back to $\mathcal{W}_{i,j}$ again. Since one of the j agents is selected with probability $\frac{j(j-1)+2j(n-j)}{n(n-1)} \geq \frac{j}{n}$, we have $w_{i,j} \leq \frac{n}{j}$, and thus

$$\max_{C \in \mathcal{C}_{\text{all}}} ECT(C, \mathcal{L}) \leq w_{0,n} + \sum_{i=1}^{s} \sum_{j=1}^{n} w_{i,j} \leq 1 + ns \cdot H(n) = O(ns \log n) \tag{6}$$

holds where H is the harmonic function. □

In Section 4.1, the definitions of epidemic and virtual agents stand on the assumption that there exists *exactly* one leader in the initial configuration C_0 (i.e. $C_0 \in \mathcal{L}_{\text{one}}$). However, this assumption can be relaxed as follows: there exists *at least* one leader in C_0 (i.e. $C_0 \in \mathcal{L}$). With defining v_l as any arbitrary leader in C_0, we can redefine $I_{C_0,\gamma}(t)$ and $VI_{C_0,\gamma}(v, t)$ for any $C_0 \in \mathcal{L}$ in the same manner as Section 4.1. Then, Corollary 3 holds not only for any $C_0 \in \mathcal{L}_{\text{one}}$ but also for any $C_0 \in \mathcal{L}$.

Lemma 11. $\max_{C \in \mathcal{L}} ECT(C, \mathcal{L}_{\text{half}})$ *belongs to $O(ns)$.*

Proof. Let C_0 be a configuration in $\mathcal{L}$ and let $\Xi(C_0, \Gamma) = C_0, C_1, \ldots$. By Corollary 3, we have

$$\Pr(C_{2ns^*} \in \mathcal{L}_{\text{half}}) \geq 1 - 2n \cdot \exp\left(-s^*\right) = 1 - o(1). \tag{7}$$

Since $C_{2ns^*} \in \mathcal{L}$ holds by Lemma 1, we have

$$\max_{C\in\mathcal{L}} ECT(C, \mathcal{L}_{\text{half}}) \leq 2ns^* + o(1) \cdot \max_{C\in\mathcal{L}} ECT(C, \mathcal{L}_{\text{half}}).$$

Solving this inequality gives us $\max_{C\in\mathcal{L}} ECT(C, \mathcal{L}_{\text{half}}) = O(ns)$. □

In the following, we prove Lemma 15 ($\max_{C\in\mathcal{L}_{\text{half}}} ECT(C, \mathcal{S}_{\text{half}}) = O(ns)$). Informally, the lemma holds for the following reason.

- When we can ignore the occurrence of timeout, at most n leaders kill each other, and one unique leader is elected within $O(n^2) \leq O(ns)$ expected interactions. (We shall see this fact in Lemma 13.)
- From Corollaries 1 and 3, one can observe that timeout happens only extremely rarely if the execution starts from a configuration in $\mathcal{L}_{\text{half}}$. Hence, the probability that timeout happens by time $O(n^2)$ is negligible. Furthermore, although the execution may get out of $\mathcal{L}_{\text{half}}$, it comes back to $\mathcal{L}_{\text{half}}$ by time $O(ns)$ with very high probability (Corollary 3).

To conclude, a configuration in $\mathcal{S}_{\text{half}}$ is likely to be reached by time $O(ns)$.

From now, we show the formal proof. To avoid complicated analysis of conditional probability, we introduced a protocol P'_{LE}, which eliminate timeout mechanism from P_{LE}. Specifically, P'_{LE} is the protocol obtained from P_{LE} by replacing rule R4 in the transition function δ with the following rule R4':

R4' $\quad ((-, 0), (-, 0)) \ \rightarrow \ ((-, 0), (-, 0))$.

The state sets of P_{LE} and P'_{LE} are identical, and hence, so are the $\mathcal{C}_{\text{all}}(P_{LE})$ and $\mathcal{C}_{\text{all}}(P'_{LE})$.

Lemma 12. *Let C_0 be a configuration in $\mathcal{C}_{\text{all}}$ and let γ be an interaction sequence. Let $\Xi_{P_{LE}}(C_0, \gamma) = C_0, C_1, \ldots$ and $\Xi_{P'_{LE}}(C_0, \gamma) = C_0, D_1, D_2, \ldots$. The following predicate holds for any $t \geq 0$:*

$$D_t \in \mathcal{L}_{\text{one}} \wedge C_t \in \mathcal{C}_{\text{half}} \wedge TO_{C_0,\gamma}(t) = 0 \ \Rightarrow \ C_t \in \mathcal{S}_{\text{half}}.$$

Proof. Assume that $D_t \in \mathcal{L}_{\text{one}}$, $C_t \in \mathcal{C}_{\text{half}}$ and $TO_{C_0,\gamma}(t) = 0$ holds. Note that executions $\Xi_{P_{LE}}(C_0, \gamma)$ and $\Xi_{P'_{LE}}(C_0, \gamma)$ have no difference until timeout happens. By the assumption $TO_{C_0,\gamma}(t) = 0$, we have the equality $C_t = D_t$, and hence, $C_t \in \mathcal{L}_{\text{one}} \cap \mathcal{C}_{\text{half}} = \mathcal{S}_{\text{half}}$ holds. □

Lemma 13 (*Angluin et al. [7]*)**.** $\max_{C\in\mathcal{L}} ECT_{P'_{LE}}(C, \mathcal{L}_{\text{one}}) = (n-1)^2$.

Proof. The number of leaders decrease by 1 when two leaders have an interaction. In each interaction, two of i leaders have an interaction with probability ${}_iC_2/{}_nC_2$. Hence, we have

$$\max_{C\in\mathcal{L}} ECT_{P'_{LE}}(C, \mathcal{L}_{\text{one}}) = \sum_{i=2}^{n} \frac{{}_nC_2}{{}_iC_2} = (n-1)^2. \quad \square$$

In what follows, we use an integer $r = \lceil \frac{2n^2}{2ns^*} \rceil \cdot 2ns^*$.

Lemma 14. *Let C_0 be a configuration in $\mathcal{L}_{\text{half}}$ and let $\Xi_{P_{LE}}(C_0, \Gamma) = C_0, C_1, \ldots$. Then, the following inequality holds:*

$$\Pr(C_r \in \mathcal{S}_{\text{half}}) \geq \frac{1}{2} - o(1).$$

Proof. Let $\Xi_{P'_{LE}}(C_0, \Gamma) = C_0, D_1, D_2, \ldots$. By Lemma 12, it suffices to show $\Pr(D_r \in \mathcal{L}_{\text{one}} \wedge C_r \in \mathcal{C}_{\text{half}} \wedge TO_{C_0,\Gamma}(r) = 0) \geq \frac{1}{2} - o(1)$. By Lemma 13 and Markov's inequality, we have

$$\begin{aligned}\Pr(D_r \notin \mathcal{L}_{\text{one}}) &= \Pr(CT(\Xi_{P'_{LE}}(C_0, \Gamma), \mathcal{L}_{\text{one}}) > r) \\ &\leq \frac{ECT_{P'_{LE}}(C_0, \Gamma)}{r} \leq \frac{(n-1)^2}{2n^2} \leq \frac{1}{2}.\end{aligned}$$

Next, we show a lower bound of $\Pr(C_r \in \mathcal{C}_{\text{half}} \wedge TO_{C_0,\Gamma}(r) = 0)$. By Lemma 5 and Corollary 3, we have the following inequality:

$$\Pr\left(C_{2ns^*} \in \mathcal{L}_{\text{half}} \wedge TO_{C_0,\Gamma}(2ns^*) = 0\right) \geq 1 - 2n \cdot e^{-s^*}.$$

Hence, we have

$$\begin{aligned}\Pr\left(C_r \in \mathcal{L}_{\text{half}} \wedge TO_{C_0,\Gamma}(r) = 0\right) &\geq \left(1 - 2n \cdot e^{-s^*}\right)^{\lceil 2n^2/2ns^* \rceil} \\ &\geq 1 - 2n \cdot e^{-s^*} \cdot \left\lceil \frac{n}{s^*} \right\rceil \qquad \text{(see Lemma B.2)} \\ &\geq 1 - o(1).\end{aligned}$$

Thus, we have $\Pr(D_r \in \mathcal{L}_{\text{one}} \wedge C_r \in \mathcal{C}_{\text{half}} \wedge TO_{C_0,\Gamma}(r) = 0) \geq \frac{1}{2} - o(1)$. □

Lemma 15. $\max_{C\in\mathcal{L}_{\mathrm{half}}} ECT(C, \mathcal{S}_{\mathrm{half}})$ *belongs to* $O(ns)$.

Proof. Let C_0 be a configuration in $\mathcal{L}_{\mathrm{half}}$ and let $\Xi_{P_{LE}} = C_0, C_1, \ldots$. By Lemmas 1 and 14, we have

$$\max_{C\in\mathcal{L}_{\mathrm{half}}} ECT(C, \mathcal{S}_{\mathrm{half}}) \le r + p\cdot\left(\max_{C\in\mathcal{L}} ECT(C, \mathcal{L}_{\mathrm{half}}) + \max_{C\in\mathcal{L}_{\mathrm{half}}} ECT(C, \mathcal{S}_{\mathrm{half}})\right).$$

where $p = \frac{1}{2} + o(1)$. Solving this inequality gives $\max_{C\in\mathcal{L}_{\mathrm{half}}} ECT(C, \mathcal{S}_{\mathrm{half}}) = O(r) = O(ns)$. □

Thus, we obtain Eq. (1) from Lemmas 10, 11 and 15 and Eq. (5). Now, we have proven Eqs. (1) and (2), which directly lead to the following theorem.

Theorem 1. *P_{LE} is $(O(ns\log n), \Omega(se^{s/96}))$-probabilistic loosely-stabilizing for specification LE if s is a multiple of 96 and $s \ge 3n$.*

Recall that P_{LE} knows an upper bound N of n. When we set s to be $96N$, P_{LE} realize $(O(nN\log n), \Omega(Ne^N))$-probabilistic loose-stabilization for *LE*. That is, P_{LE} realizes fast convergence to a loosely-safe configuration (low polynomial order time) and extremely long maintenance of its specification (exponential order time).

Before now, we analyzed the holding time and convergence time of our protocol in terms of expected time. In addition, we can also show the performance of our protocol in terms of *with high probability*. Specifically, the following two theorems hold.

Theorem 2. *$HT(\Xi(C,\Gamma), LE) \ge n^{-c}s^*e^{-s^*}$ holds with probability at least $1-n^{-c}$ for any integer $c \ge 1$ and any $C \in \mathcal{S}_{\mathrm{half}}$*

Proof. From Eq. (3), we have $HT(\Xi(C,\Gamma), LE) \ge 2ns^*d$ with probability at least $(1-2ne^{-s^*})^d > 1-2dne^{-s^*}$ for any integer d. We obtain the theorem by substituting d with $n^{-c-1}e^{s^*}/2$.

Theorem 3. *$CT(\Xi(C,\Gamma), \mathcal{S}_{\mathrm{half}}) = O((c+1)ns\log n\cdot\log s)$ holds with probability at least $1-n^{-c}$ for any integer $c \ge 1$ and any $C \in \mathcal{C}_{\mathrm{all}}$*

Proof Sketch. From Eq. (7) and Lemma 14, $CT(C', \mathcal{S}_{\mathrm{half}}) \le ns + r$ holds with probability at least $(1-o(1))(1/2-o(1)) = 1/2-o(1)$ for any integer c and any $C' \in \mathcal{L}$. The probability $1/2-o(1)$ becomes greater than $1/3$ for sufficiently large n, and from Lemma 1, $\Pr(CT(C', \mathcal{S}_{\mathrm{half}}) \le d(ns+r)) \ge 1-(2/3)^d$ holds for any integer $d \ge 1$. By substituting d with $(c+1)\log_{3/2} n$, we find that $CT(C', \mathcal{S}_{\mathrm{half}}) = O((c+1)ns\log n)$ holds with probability at least $1-n^{-c-1}$. (Remind that $r = O(ns)$.) From Eq. (6) and Lemma B.3, $CT(C'', \mathcal{L}) = O((c+1)ns\log n\cdot\log s)$ holds with probability at least $1-n^{-c-1}$ for any integer c and any $C'' \in \mathcal{C}_{\mathrm{all}}$. Hence, we get

$$\Pr(CT(C'', \mathcal{S}_{\mathrm{half}}) = O((c+1)ns\log n\cdot\log s)) \ge (1-n^{-c-1})^2 \ge 1-n^{-c}. \quad \square$$

5. Conclusion and discussion

In this paper, we introduced a novel concept of loose-stabilization and presented a probabilistic loosely-stabilizing leader election protocol in the PPP model of complete networks. The basic strategy of this protocol is described as follows: if two leaders interact each other then one of the two becomes a non-leader, and if two non-leaders with the timer value 0 interact each other then one of the two becomes a leader. The timer value of each agent is controlled so that the timer value of every agent keeps relatively high when at least one leader exists in the population, while the highest timer value among all agents is monotonically non-increasing when no leader exists. Thus, starting from any arbitrary configuration, the protocol reaches a configuration in $\mathcal{S}_{\mathrm{half}}$ within $O(nN\log n)$ expected steps, and then, it keeps the unique leader for $\Omega(Ne^N)$ expected steps where n is the actual network size, and N is a known upper bound of n. The proposed protocol has practical significance from the following reason: the protocol can be practically considered to attain self-stabilization because of exponentially long time of keeping a unique leader while the self-stabilizing leader election in the PPP model of complete networks is impossible without the knowledge of the exact network size [11].

In the PPP model, which we adopt in this paper, the probabilistic law of interactions is completely uniform; any pair of agents have the same probability of being selected at each time. However, this is not the case in reality. For example, when two agents have just met, they have a higher probability of meeting again at the next time because an interaction happens only when two agents get sufficiently close. This difference between the model and the reality may damage our result because the probabilistic law of interactions is crucial for evaluating holding time and convergence time. However, we currently do not know another probabilistic law that appropriately reflects the characteristic of the mobile sensor network. In addition, even if we find the appropriate law, the calculations of holding time and convergence time likely become too complicated. One of our future works is to find a probabilistic law that is appropriate to an acceptable level, and under which we can evaluate holding time and convergence time.

We will also aspire to apply the notion of loose-stabilization to other problems that are known unsolvable or too costly in a self-stabilizing fashion.

Acknowledgement

This work is supported in part by Global COE Program of MEXT, Grant-in-Aid for Scientific Research ((B)17300020, (B)22300009, (B)20300012) of JSPS, Grant-in-Aid for Young Scientists ((B)18700059) of JSPS, and the Kayamori Foundation of Informational Science Advancement.

Appendix A. Impossibility result

In this section, we show that the impossibility result of [11] holds even in the PPP model and even if we can use the infinite space of each agent. That is, without knowledge of the exact network size, there is no probabilistic stabilizing leader election protocol even if the infinite space of each agent is available.

Definition 3 (*Probabilistic Stabilization*)**.** A protocol $P(Q, Y, O, \delta)$ is probabilistic stabilizing for a specification $SP(Y)$ if a nonempty set $\mathcal{S}$ of configurations exists such that:

$$\forall C \in \mathcal{C}_{\mathrm{all}}(P),\ \Pr(CT_P(\Xi_P(C, \Gamma), \mathcal{S}) \leq t) \to 1 \quad (t \to \infty)$$
$$\forall C \in S,\ \forall \gamma,\ HT_P(\Xi_P(C, \gamma), SP) = \infty.$$

Probabilistic stabilization requires that any execution starting from any configuration eventually reaches a safe configuration (i.e. a configuration in $\mathcal{S}$) with probability 1, and after that, the execution must satisfy the specification forever.

Theorem 4. *Let n and n' be distinct integers ($n, n' \geq 2$). Then, there is no probabilistically stabilizing protocol for LE that works on the complete networks of size n and n'.*

Proof. We prove the theorem by contradiction assuming that such a protocol $P(Q, Y, O, \delta)$ exists. This proof is almost the same as the proof of [11]. We assume $n > n'$ without loss of generality. Consider that P runs on complete networks $G = (V, E), V = \{1, 2, \ldots, n\}$. Since P is probabilistically stabilizing on G, there exists a safe-configuration $C : V \to Q$, from which every execution satisfies the specification of leader election. Let v_l be the unique leader in C. Let $V' \subseteq V$ be a set of n' agents such that $v_l \notin V'$. We focus on the sub-configuration $C' : V' \to Q$ such that $C'(v) = C(v)$ for any agent $v \in V'$. (Note that C' does not have a leader.) Since P is also probabilistically stabilizing on complete network $G'(V', E')$, there exists some interaction sequence γ such that C'_t has a leader for some t where $\Xi_P(C', \gamma) = C'_0, C'_1, \ldots$. However, this means that C_t has two leaders where $\Xi_P(C, \gamma) = C_0, C_1, \ldots$, which contradicts the fact that C is a safe configuration. □

This proof does not use the constraint on the memory space of the agents. Hence, the theorem holds even if the infinite space of each agent is available.

Appendix B. Other lemmas

Lemma B.1. *Let n and m be integers ($0 \leq m \leq n$) and let ${}_0C_2 = {}_1C_2 = 0$. Then, the following inequality holds:*

$$\frac{m-1}{{}_mC_2 + {}_{n-m}C_2} \leq \frac{4}{n}. \tag{8}$$

Proof. If $m \geq \frac{n}{2}$, then Eq. (8) follows from $\frac{m-1}{{}_mC_2 + {}_{n-m}C_2} \leq \frac{m-1}{{}_mC_2} = \frac{2}{m} \leq \frac{4}{n}$. If $m < \frac{n}{2}$, then $n - m - 1 > m - 1$ holds. Thus, the following inequalities hold:

$$\begin{aligned}\frac{m-1}{{}_mC_2 + {}_{n-m}C_2} &= \frac{2(m-1)}{m(m-1) + (n-m)(n-m-1)} \\ &< \frac{2(m-1)}{m(m-1) + (n-m)(m-1)} \\ &= \frac{2}{n} < \frac{4}{n}.\end{aligned}$$

Therefore, Eq. (8) holds in all cases. □

Lemma B.2. *$(1-p)^r \geq 1 - rp$ holds for any real number $p \leq 1$ and any integer $r \geq 1$.*

Proof. We show this lemma by induction with respect to r. If $r = 1$, the equality $(1-p)^r = 1 - rp$ trivially holds. Assume that $(1-p)^r \geq 1 - rp$ holds when $r = k$. Then, we have

$$\begin{aligned}(1-p)^{k+1} = (1-p)(1-p)^k &\geq (1-p)(1-kp) \\ &\geq 1 - (k+1)p + kp^2 \geq 1 - (k+1)p.\end{aligned}$$

Therefore, $(1-p)^r \geq 1 - rp$ holds for any integer $r \geq 1$. □

Lemma B.3. *Let $X_1, X_2, \ldots, X_m$ be m random variables such that $\Pr(X_i > k) \leq (1-p_i)^k$ for any integers i and k. And, let $Y = \sum_{i=1}^{m} 1/p_i$. Then, we have*

$$\Pr(X_1 + \cdots + X_m > cY + m) \leq me^{-c}$$

for any real number c.

Proof.

$$\begin{aligned}\Pr(X_1 + \cdots + X_m > cY + m) &\leq \Pr\left(\bigvee\left(X_i > \frac{c}{p_i} + 1\right)\right)\\ &\leq \sum (1 - p_i)^{\lceil c/p_i \rceil}\\ &\leq me^{-c}. \quad \square\end{aligned}$$

References

[1] E.W. Dijkstra, Self-stabilizing systems in spite of distributed control, Communications of the ACM 17 (11) (1974) 643–644.
[2] S. Dolev, Self-stabilization, MIT Press, 2000.
[3] A. Israeli, M. Jalfon, Token management schemes and random walks yield self-stabilizing mutual exclusion, in: PODC, 1990, pp. 119–131.
[4] J.-C. Lin, T.C. Huang, C.Z. Yang, N. Mou, Quasi-self-stabilization of a distributed system assuming read/write atomicity, Computers and Mathematics with Applications 57 (2) (2009) 184–194.
[5] M.G. Gouda, The theory of weak stabilization, in: WSS, 2001, pp. 114–123.
[6] S. Devismes, S. Tixeuil, M. Yamashita, Weak vs. self vs. probabilistic stabilization, in: ICDCS, 2008, pp. 681–688.
[7] D. Angluin, J. Aspnes, Z. Diamadi, M.J. Fischer, R. Peralta, Computation in networks of passively mobile finite-state sensors, Distributed Computing 18 (4) (2006) 235–253.
[8] D. Angluin, J. Aspnes, M. Chan, M.J. Fischer, H. Jiang, R. Peralta, Stably computable properties of network graphs, in: DCOSS, 2005, pp. 63–74.
[9] D. Angluin, J. Aspnes, D. Eisenstat, Fast computation by population protocols with a leader, in: DISC, 2006, pp. 61–75.
[10] D. Angluin, J. Aspnes, M.J. Fischer, H. Jiang, Self-stabilizing population protocols, in: OPODIS, 2005, pp. 103–117.
[11] S. Cai, T. Izumi, K. Wada, Space complexity of self-stabilizing leader election in passively-mobile anonymous agents, in: SIROCCO, 2009, pp. 113–125.
[12] M. Fischer, H. Jiang, Self-stabilizing leader election in networks of finite-state anonymous agents, in: OPODIS, 2006, pp. 395–409.
[13] D. Angluin, J. Aspnes, D. Eisenstat, A simple population protocol for fast robust approximate majority, Distributed Computing 21 (2) (2008) 87–102.
[14] M. Mitzenmacher, E. Upfal, Probability and Computing: Randomized Algorithms and Probabilistic Analysis, Cambridge University Press, 2005.