

Time- and Space-Optimal Silent Self-Stabilizing Exact Majority in Population Protocols

Haruki Kanaya¹, Ryota Eguchi¹, Taisho Sasada¹, Fukuhito Ooshita², and Michiko Inoue¹

¹Nara Institute of Science and Technology, Nara, Japan

²Fukui University of Technology, Fukui, Japan

Abstract

We address the self-stabilizing exact majority problem in the population protocol model, introduced by Angluin, Aspnes, Diamadi, Fischer, and Peralta (2004). In this model, there are n state machines, called agents, which form a network. At each time step, only two agents interact with each other, and update their states. In the self-stabilizing exact majority problem, each agent has a fixed opinion, A or B, and stabilizes to a safe configuration in which all agents output the majority opinion from any initial configuration.

In this paper, we show the impossibility of solving the self-stabilizing exact majority problem without knowledge of n in any protocol. We propose a silent self-stabilizing exact majority protocol, which stabilizes within $O(n)$ parallel time in expectation and within $O(n \log n)$ parallel time with high probability, using $O(n)$ states, with knowledge of n . Here, a silent protocol means that, after stabilization, the state of each agent does not change. We establish lower bounds, proving that any silent protocol requires $\Omega(n)$ states, $\Omega(n)$ parallel time in expectation, and $\Omega(n \log n)$ parallel time with high probability to stabilize. Thus, the proposed protocol is time- and space-optimal.

1 Introduction

The population protocol model, introduced by Angluin, Aspnes, Diamadi, Fischer, and Peralta [3], is a model of a passively mobile sensor network. In this model, there are n state machines (called *agents*), and these agents form a network (called the *population*). Each agent lacks a unique identifier and updates its state through pairwise communication (called *interaction*). At each time step, only one pair of agents interacts, chosen by a uniform random scheduler. In this paper, we assume that the network is a complete graph, meaning that each agent can interact with all other agents. The time complexity is measured in *parallel time*, which is defined as the number of interactions divided by n .

The majority problem requires each agent to have a fixed opinion, either A or B, and to determine the majority opinion. The agents output the majority opinion, A or B, if there is no tie (i.e., the numbers of agents with input A and agents with input B are not equal); otherwise, they output T. There are two types of majority problems: approximate and exact. The approximate majority problem [5, 15] allows a small error; that is, a protocol may stabilize to an incorrect answer with low probability. On the other hand, the exact majority problem does not allow errors; that is, a protocol stabilizes to a correct answer with probability 1. The exact majority problem with designated initial states has been widely studied [1, 2, 4, 6, 7, 9, 10, 11, 16, 17, 19, 21, 22, 23], and a time- and space-optimal protocol [16], which converges within $O(\log n)$ parallel time and uses $O(\log n)$ states, has been proposed.

We address the self-stabilizing exact majority problem, which requires each agent to have a fixed opinion, either A or B, and to determine the exact majority opinion starting from any configuration. In prior research, a study [8] uses loose-stabilization [24], which relaxes the closure property of stabilization.

However, to the best of our knowledge, no study has solved the self-stabilizing exact majority problem. We solve this problem with initial knowledge of n .

With initial knowledge of n , there have been some studies on another problem: the self-stabilizing leader election [12, 13, 18], in which agents elect a unique leader. In [12], Burman, Chen, Chen, Doty, Nowak, Severson, and Xu solve the self-stabilizing ranking problem, in which agents are assigned unique ranks from $[1, n]$. They proposed two protocols: a silent protocol, which is both time- and space-optimal ($O(n)$ parallel time, $O(n)$ states), and a non-silent protocol, which is time optimal ($O(\log n)$ parallel time) but requires $\exp(O(n^{\log n} \cdot \log n))$ states. Here, a silent protocol refers to a protocol in which, after stabilization, the state of each agent does not change.

In the non-silent self-stabilizing ranking protocol [12], agents are ranked by assigning unique names in $[1, n^3]$. Upon stabilization, each agent holds a list of all agents' names and determines its rank in $[1, n]$ from the name list. This protocol works correctly even if the name space is reduced to half its size, while still maintaining the same time complexity and number of states, since Lemma 5.1 of [12] still holds, as the name space is $O(n^3)$. Thus, using this property, the exact majority problem can be solved by assigning agents with input A a name space in $[1, n^3/2]$ and agents with input B a name space in $[n^3/2 + 1, n^3]$. Also, each agent can determine its rank in $[1, n]$ from the name list. This protocol is time-optimal since, in self-stabilization, all agents must interact at least once, and by the coupon collector's problem, $\Omega(n \log n)$ interactions (which corresponds to $\Omega(\log n)$ parallel time) are required.

1.1 Our Contribution

In this paper, we address a silent protocol that solves the self-stabilizing exact majority problem. Our contribution is summarized in Table 1. First, we show the impossibility that any protocol without knowledge of n cannot solve the self-stabilizing exact majority. Second, we also show the lower bounds: any silent protocol with knowledge of n requires at least n states, and any silent protocol requires at least $\Omega(n)$ parallel time in expectation and $\Omega(n \log n)$ parallel time with high probability to stabilize. Here, the phrase "with high probability" refers to with a probability of $1 - O(1/n)$. Finally, we propose a silent protocol, \mathcal{P}_{EM} , which stabilizes to a silent configuration¹ within $O(n)$ parallel time in expectation and within $O(n \log n)$ parallel time with high probability using $O(n)$ states, with knowledge of n .

Table 1: Overview of our results. The variable n denotes the number of agents. W.H.P. means with high probability.

	knowledge	states	expected parallel time	parallel time W.H.P.
Impossibility	without n	-	-	-
Lower Bound (space)	n	n	-	-
Lower Bound (time)	n	-	$\Omega(n)$	$\Omega(n \log n)$
Protocol	n	$O(n)$	$O(n)$	$O(n \log n)$

1.2 Organization of This Paper

Section 2 defines the model and problem. Section 3 introduces existing protocols used in our protocol. In Section 4, we show that any protocol without knowledge of n cannot solve the self-stabilizing exact majority problem. Section 5 presents lower bounds for any silent protocol that solves the self-stabilizing exact majority problem and proposes a protocol that matches these lower bounds. We conclude and discuss future directions in Section 6.

¹A silent protocol stabilizes to a silent configuration; Thus, its time complexity is measured by silence time, where silence time is defined as parallel time until reaching a silent configuration. However, comparing the lower bound of stabilization time with the upper bound of silence time is not an issue, since stabilization time is no more than silence time.

2 Preliminaries

We denote the set of positive integers by \mathbb{N} , and the set of non-negative integers by \mathbb{N}_0 .

2.1 Population Protocols

A *population* is represented by a bidirectional complete graph $G = (V, E)$, where V denotes the set of agents, and $E = \{(u, v) \in V \times V \mid u \neq v\}$ denotes the set of ordered pairs of agents that can interact, and let $n = |V|$. Since G is a bidirectional complete graph in this paper, we simply represent the population as V , the set of agents. A protocol is defined as a 5-tuple $\mathcal{P} = (Q, X, Y, \delta, \pi_{out})$, where Q is the set of agents states, X is the set of input symbols, Y is the set of output symbols, $\delta : (Q \times X) \times (Q \times X) \rightarrow Q \times Q$ is the state transition function, and $\pi_{out} : Q \times X \rightarrow Y$ is the output function. The variable `var` of agent u is denoted as $u.\text{var}$. Similarly, the input of each agent u is denoted by $u.\text{input}$. Each agent outputs $\pi_{out}(s, x) \in Y$ at each time step when it is in state $s \in Q$ and its input is $x \in X$. In each interaction, the interacting agents u and v are assigned the roles of initiator and responder, respectively, breaking the symmetry of the interaction. Suppose that two agents u, v interact, with u as the initiator and v as the responder, while they are in states p and q , respectively, and their inputs are x and y respectively. They then update their states to p' and q' , respectively, where $\delta((p, x), (q, y)) = (p', q')$.

A *configuration* $C : V \rightarrow Q \times X$ represents the state and input of all agents. A configuration C changes to C' via an interaction $(u, v) \in E$, denoted by $C \xrightarrow{(u, v)} C'$ if $(C'(u), C'(v)) = \delta(C(u), C(v))$ and $\forall w \in V \setminus \{u, v\} : C'(w) = C(w)$. When a configuration C changes to C' via an interaction, we say that C can change to C' denoted by $C \rightarrow C'$. The set of all configurations by a protocol \mathcal{P} is denoted by $\mathcal{C}_{all}(\mathcal{P})$. A uniform random scheduler $\Gamma = \Gamma_0, \Gamma_1, \dots$ determines which ordered pair of agents interacts at each step, where $\Gamma_t \in E$ (for $t \geq 0$) is a random variable satisfying $\forall (u, v) \in E, \forall t \in \mathbb{N}_0 : \Pr(\Gamma_t = (u, v)) = \frac{1}{n(n-1)}$. Similarly, a deterministic scheduler $\gamma = \gamma_0, \gamma_1, \dots$ determines which ordered pair of agents interacts at each time step, where $\gamma_i \in E$ (for $i \geq 0$) is a predetermined value. Note that γ can be a possible value of Γ with positive probability.

An *execution* of \mathcal{P} is defined as an infinite sequence of configurations $\Xi_{\mathcal{P}}(C_0, g) = C_0, C_1, \dots$ that starts from an initial configuration $C_0 \in \mathcal{C}_{all}(\mathcal{P})$, and satisfies $\forall i \in \mathbb{N} : C_{i-1} \rightarrow C_i$, where g is a uniform random or deterministic scheduler. Note that if g is a uniform random scheduler, then C_i is a random variable. A configuration C' is reachable from C_0 if there exists an execution $\Xi_{\mathcal{P}}(C_0, \Gamma) = C_0, C_1, \dots$ such that $\exists i \in \mathbb{N} : C_i = C'$. In an execution under a uniform random scheduler, if a configuration C appears infinitely often, then every configuration reachable from C also appears infinitely often². An execution is said to reach a set of configurations \mathcal{C} if it reaches a configuration in \mathcal{C} . Similarly, an execution is said to belong to a set of configurations \mathcal{C} if its initial configuration is in \mathcal{C} . We assume that the input remains unchanged during execution, meaning that $\forall v \in V, \forall i \in \mathbb{N} : C_i(v).\text{input} = C_0(v).\text{input}$ holds for an execution $\Xi_{\mathcal{P}}(C_0, g) = C_0, C_1, \dots$, where g is a uniform random or deterministic scheduler. A configuration C is safe if and only if, for any execution starting from C , the outputs of all agents never change. Similarly, a configuration C is silent if and only if, for any execution starting from C , the states of all agents never change. A protocol is silent if and only if an execution reaches a silent configuration with probability 1.

The time complexity of a protocol is defined as the number of interactions divided by n . This is referred to as either *parallel time* or simply *time*. The stabilization time is defined as the time until an execution of the protocol under a uniform random scheduler reaches a safe configuration. Similarly, the silence time is defined as the time until an execution of the protocol under a uniform random scheduler reaches a silent configuration. The phrase “*with high probability*” refers to with a probability of $1 - O(1/n)$.

2.2 Self-Stabilizing Exact Majority

The self-stabilizing exact majority problem requires that each agent outputs the exact majority opinion from the two types of opinions, A and B, held by agents, starting from any initial configuration. We define

²This property holds since a uniform random scheduler is globally fair with probability 1, as shown in [14].

V_a as the set of agents with the opinion A, and V_b as the set of agents with the opinion B. Note that we assume the input of each agent does not change during execution; thus, V_a and V_b remain unchanged throughout execution. The exact majority opinion is as follows: If $|V_a| < |V_b|$, the exact majority opinion is B; if $|V_a| > |V_b|$, it is A; otherwise, it is T.

Definition 1. A protocol $\mathcal{P} = (Q, X, Y, \delta, \pi_{out})$ is a self-stabilizing exact majority protocol if and only if $X = \{\text{A}, \text{B}\}$, $Y = \{\text{A}, \text{B}, \text{T}\}$, and the following conditions hold for any initial configuration $C_0 \in \mathcal{C}_{all}(\mathcal{P})$:

- For any safe configuration C_{safe} reachable from C_0 , it holds that $\forall v \in V : \pi_{out}(C_{\text{safe}}(v)) = y$, where $y \in Y$ is the exact majority opinion of agents.
- Any execution starting from C_0 reaches safe configurations with probability 1.

3 Tools

3.1 Epidemic Protocol

The epidemic protocol proposed by Angluin, Aspnes, and Eisenstat [4] is used in PROPAGATE-RESET described later. In this protocol, each agent has a variable $x \in \{0, 1\}$, and when two agents u and v interact, $u.x$ and $v.x$ are both updated to $\max(u.x, v.x)$. Suppose that all agents have $x = 0$. Once one agent's x becomes 1, all agents' x will become 1 within $\Theta(\log n)$ time in expectation and with high probability.

3.2 Ranking Protocol

In this subsection, we explain the silent self-stabilizing ranking protocol proposed by Burman et al. [12] since we will use it in Section 5.2. A self-stabilizing ranking protocol assigns agents unique ranks in $[1, n]$ from any initial configuration, and the agents maintain their ranks forever. Their protocol is divided into two parts, PROPAGATE-RESET and OPTIMAL-SILENT-SSR. In their protocol, an agent has a **role** $\in \{\text{Resetting}, \text{Settled}, \text{Unsettled}\}$ to reduce the number of states, and they define variables for each role. The number of states becomes the sum of the states required for each **role**. We briefly explain their protocol, but refer to [12] for more details and proofs.

3.2.1 PROPAGATE-RESET

In this protocol, when some inconsistency is detected in the population, all agents are reset to a specific state with high probability. Each agent uses two variables, **role** and **resetcount**, which represent the agent's role and a timer used to determine whether a reset has been completed. When agents detect some inconsistency, they set **role** to **Resetting** and **resetcount** to $R_{max} (= 60 \log n)$. The role of **Resetting** propagates to all agents through the epidemic protocol, treating **Resetting** as 1 and all other roles as 0. Eventually, all agents are reset within $O(\log n)$ time with high probability. After resetting, all inconsistencies are eliminated. We have following lemmas from [12].

Lemma 1 (Lemma 3.2 in [12]). *If some inconsistency is detected, all agents' **role** become **Resetting** within $4 \log n$ time with high probability.*

Lemma 2 (Corollary 3.5 in [12]). *Starting from any configuration, an execution reaches a configuration in which no agent has **role** = **Resetting** within $O(n)$ time with high probability.*

3.2.2 OPTIMAL-SILENT-SSR

In this protocol, agents are assigned unique ranks in $[1, n]$ from any initial configuration. The functions of **role** for agents are as follows: **role** = **Settled** means the agent has been assigned a rank, **role** = **Unsettled** means the agent has not been assigned a rank, and **role** = **Resetting** means the agent is

executing PROPAGATE-RESET. When some rank conflict or some inconsistency is detected, PROPAGATE-RESET starts. At this point, as mentioned above, `role` is set to `Resetting`, `resetcount` is set to R_{max} , and additionally, `leader` is set to L . Here, $\text{leader} \in \{L, F\}$ is a variable of an agent with `role = Resetting` in this protocol, indicating whether the agent is a leader. While resetting, agents elect a unique leader through a process where, when leaders meet, one of them becomes a follower (F). After resetting, all agents are unranked, and a unique leader agent is elected. The leader's `role` becomes `Settled`, with its `rank` set to 1. Each agent with `rank = i` creates agents with `rank = 2i` and `rank = 2i + 1` if the newly assigned `rank` does not exceed n . Eventually, all agents are assigned unique ranks. Let s_{rank} be the expected silence time of OPTIMAL-SILENT-SSR. We have the following lemma from [12].

Lemma 3 (Theorem 4.3 in [12]). *OPTIMAL-SILENT-SSR is a silent self-stabilizing ranking protocol with $O(n)$ states and $s_{rank} = O(n)$ expected time.*

4 Impossibility

In this section, we provide the impossibility. We prove that no protocol can solve the self-stabilizing exact majority problem without knowledge of n , using the same approach as the proof of the impossibility of a self-stabilizing bipartition protocol by Yasumi, Ooshita, Yamaguchi, and Inoue [25].

Theorem 1. *There is no protocol that solves the self-stabilizing exact majority without knowledge of n .*

Proof. For contradiction, we assume there is a protocol \mathcal{P} that solves self-stabilizing exact majority without knowledge of n . Consider a population V , where $|V| \geq 4$, $|V_a| \geq 2$, $|V_b| \geq 2$, and $|V_a| > |V_b|$. From the assumption, \mathcal{P} solves the problem for V and V_b . Let C_0 be any configuration of V , and let $\gamma = \gamma_0, \gamma_1, \dots$ be a deterministic scheduler on V that makes C_0 eventually reach a safe configuration. For an execution $\Xi_{\mathcal{P}}(C_0, \gamma) = C_0, C_1, \dots$ on V , there exists a safe configuration C_t in which all agents output A.

Similarly, consider another population V' consisting of all agents in V_b , and let C'_0 be a configuration of V' satisfying $\forall v \in V' : C'_0(v) = C_t(v)$. Let $\gamma' = \gamma'_0, \gamma'_1, \dots$ be a deterministic scheduler on V' that makes C'_0 eventually reach a safe configuration. For an execution $\Xi_{\mathcal{P}}(C'_0, \gamma') = C'_0, C'_1, \dots$ on V' , there exists a safe configuration $C'_{t'}$ in which all agents output B.

From these executions, let $\gamma'' = \gamma''_0, \gamma''_1, \dots$ be a deterministic scheduler on V satisfying $\forall i \in [0, t-1] : \gamma''_i = \gamma_i \wedge \forall j \in [0, t'-1] : \gamma''_{t+j} = \gamma'_j$. Then, let $C''_0 = C_0$, and consider the execution $\Xi_{\mathcal{P}}(C''_0, \gamma'')$. In this execution, C''_t is a safe configuration in which all agents output A. However, in $C''_{t+t'}$, all agents that belong to V_b output B. This contradicts the assumption that C''_t is a safe configuration. \square

Note that a deterministic scheduler can appear as a prefix of a uniform random scheduler with nonzero probability. Therefore, this theorem is also applicable to the impossibility result under a uniform random scheduler.

5 Lower Bound and Upper Bound

5.1 Lower Bounds

In this subsection, we provide the lower bounds.

Lemma 4. *Consider a population V where $|V| \geq 4$, $|V_a| \leq |V_b|$. In a silent configuration of a silent self-stabilizing exact majority protocol, the states of agents with input A are all distinct from one another.*

Proof. We prove this lemma by contradiction. We assume that there exists a silent self-stabilizing exact majority protocol \mathcal{P} that uses $|V_a| - 1$ states for agents with input A in a silent configuration. Suppose some execution reaches a silent configuration C . From the assumption, there exist two agents with input A that are in the same state s_A in C . Since C is a silent configuration, the state satisfies

$\delta((s_A, \mathbf{A}), (s_A, \mathbf{A})) \rightarrow (s_A, s_A)$ and $\pi_{out}(s_A, \mathbf{A}) \in \{\mathbf{B}, \mathbf{T}\}$. Next, consider another population with n agents, where all agents have `input` = \mathbf{A} , and an execution starting from the configuration in which all agents' states are s_A . Such an execution cannot reach a safe configuration since all agents eternally output \mathbf{B} or \mathbf{T} , while the exact majority opinion is \mathbf{A} . This is a contradiction. \square

Theorem 2. *A silent self-stabilizing exact majority protocol requires n states.*

Proof. Let n be an even number no less than 4. Consider a population V with n agents, where $|V_a| = n/2 - 1$. From Lemma 4, there exist $n/2 - 1$ distinct states of agents with `input` = \mathbf{A} that output \mathbf{B} . Next, consider another population V' with n agents, where $|V'_a| = n/2$. From Lemma 4, there exist $n/2$ distinct states of agents with `input` = \mathbf{A} that output \mathbf{T} . Finally, consider a population V'' with n agents, where $|V''_a| = n/2 + 1$. It is clear that there exists at least one state of agents with `input` = \mathbf{A} such that they output \mathbf{A} . Thus, the number of states is at least $(n/2 - 1) + (n/2) + 1 = n$. \square

We prove a lower bound for the stabilization time of a silent self-stabilizing exact majority protocol in the same way as the proof of the lower bound for the stabilization time of a silent self-stabilizing leader election protocol by Burman et al. [12].

Theorem 3. *Any silent self-stabilizing exact majority protocol requires $\Omega(n)$ time in expectation and $\Omega(n \log n)$ time with high probability to reach a safe configuration.*

Proof. Consider a population V where $|V| \geq 5 \wedge |V| \bmod 2 = 1$, $|V_a| = \lfloor n/2 \rfloor$, and $|V_b| = \lceil n/2 \rceil$. From Lemma 4, in a silent configuration C , the states of agents whose input are \mathbf{A} are distinct from one another. Let s_A be the state of an agent w whose input is \mathbf{A} in C .

For an agent $u \in V$ whose input is \mathbf{B} , we consider another population $V' = V$ and a configuration C' satisfying $\forall v \in V' \setminus \{u\} : C'(v) = C(v)$ and $C'(u) = (s_A, \mathbf{A})$. In C' , all agents output \mathbf{B} , and the exact majority opinion of V' is \mathbf{A} . Thus, C' is not safe. Since C is a silent configuration, in C' , unless u and w interact with each other, their states will not be changed. The probability that u and w interact is $\frac{2}{n(n-1)}$. Thus, the expected number of interactions until the interaction occurs is $n(n-1)/2 = \Omega(n^2)$. Therefore, divided by n , the expected time is $\Omega(n)$.

The probability that u and w do not interact in $\alpha \frac{n(n-1)}{2} \log_e n - 1$ interactions is $\left(1 - \frac{2}{n(n-1)}\right)^{\alpha \frac{n(n-1)}{2} \log_e n - 1} > e^{-\alpha \log_e n} = n^{-\alpha}$ for some constant value $\alpha > 0$. Thus, the number of interactions until u and w interact with each other is $\frac{n(n-1)}{2} \log_e n - 1 = \Omega(n^2 \log n)$ with probability $1 - O(1/n)$ ($\alpha = 1$). Therefore, divided by n , the time with high probability is $\Omega(n \log n)$. \square

5.2 Matching Upper Bound

In this subsection, we propose a silent self-stabilizing exact majority protocol, \mathcal{P}_{EM} , using $O(n)$ states and stabilizing within $O(n)$ time in expectation and within $O(n \log n)$ time with high probability, with knowledge of n . In \mathcal{P}_{EM} , each agent has an input and a state. The input of an agent a is $a.\text{input} \in \{\mathbf{A}, \mathbf{B}\}$ (read-only). The state of each agent is represented by separate variables. Since we use the silent self-stabilizing ranking protocol OPTIMAL-SILENT-SSR [12], each agent has the variables used in OPTIMAL-SILENT-SSR. Additionally, each agent a has two other variables: $a.\text{answer} \in \{\phi, \mathbf{T}, \mathbf{A}, \mathbf{B}\}$ and $a.\text{timer} \in [0, 7(t_{rank} + 4)]$. Here, t_{rank} is a constant satisfying $s_{rank} \leq t_{rank} \cdot n$ and $t_{rank} = O(1)$, where $t_{rank} \cdot n$ denotes a sufficient amount of time for OPTIMAL-SILENT-SSR to stabilize. The number of states of \mathcal{P}_{EM} is $O(n)$ since the number of states of `answer` and `timer` is $O(1)$, and the number of states of the variables used in OPTIMAL-SILENT-SSR is $O(n)$. Each agent a outputs \mathbf{T} if $a.\text{answer} = \phi$; otherwise, it outputs $a.\text{answer}$.

Our protocol \mathcal{P}_{EM} , described in Algorithm 1, can be summarized as follows:

- **Ranking.** Execute OPTIMAL-SILENT-SSR (line 1).

Algorithm 1: Silent Self-Stabilizing Exact Majority Protocol \mathcal{P}_{EM}

```

when an agent  $a_0$  interacts with an agent  $a_1$  do
1   Execute OPTIMAL-SILENT-SSR [12].
2   for  $i \in \{0, 1\}$  do
3     if  $a_i.\text{role}$  becomes Resetting in this interaction then
4        $a_i.\text{answer} \leftarrow \phi$ 
5     if  $a_i.\text{role}$  becomes Settled in this interaction  $\wedge a_i.\text{rank} = \lceil n/2 \rceil$  then
6        $a_i.\text{timer} \leftarrow 7(t_{\text{rank}} + 4)$ 
7     if  $\exists i \in \{0, 1\} : a_0.\text{role} = a_1.\text{role} = \text{Resetting} \wedge a_i.\text{answer} = \phi \wedge a_{1-i}.\text{answer} \neq \phi$  then
8        $a_i.\text{answer} \leftarrow a_{1-i}.\text{answer}$ 
9     if  $a_0.\text{role} = a_1.\text{role} = \text{Settled}$  then
10      if  $a_0.\text{rank} < a_1.\text{rank} \wedge a_0.\text{input} = B \wedge a_1.\text{input} = A$  then
11        Swap the states between  $a_0$  and  $a_1$ .
12      if  $n \bmod 2 = 0 \wedge \exists i \in \{0, 1\} : a_i.\text{rank} = a_{1-i}.\text{rank} - 1 = n/2$  then
13        if  $a_i.\text{input} = a_{1-i}.\text{input}$  then
14           $(a_i.\text{answer}, a_{1-i}.\text{answer}) \leftarrow (a_i.\text{input}, a_i.\text{input})$ 
15        else
16           $(a_i.\text{answer}, a_{1-i}.\text{answer}) \leftarrow (T, T)$ 
17      else if  $n \bmod 2 = 1 \wedge \exists i \in \{0, 1\} : a_i.\text{rank} = \lceil n/2 \rceil$  then
18         $a_i.\text{answer} \leftarrow a_i.\text{input}$ 
19      if  $\exists i \in \{0, 1\} : a_i.\text{rank} = \lceil n/2 \rceil$  then
20        if  $a_{1-i}.\text{rank} = n$  then  $a_i.\text{timer} \leftarrow \max(0, a_i.\text{timer} - 1)$ 
21        if  $a_i.\text{timer} = 0 \wedge a_i.\text{answer} \neq a_{1-i}.\text{answer}$  then
22           $a_{1-i}.\text{answer} \leftarrow a_i.\text{answer}$ 
23          for  $j \in \{0, 1\}$  do
24             $(a_j.\text{role}, a_j.\text{leader}, a_j.\text{resetcount}) \leftarrow (\text{Resetting}, L, R_{\text{max}})$ 

```

- **Swapping.** Swap the values of all variables between interacting agents, except for `input`, if the `rank` of an agent with input `B` is less than the `rank` of an agent with input `A` (lines 10–11). This eventually places agents with input `A` in the lower ranks and agents with input `B` in the higher ranks. Note that even if the states of pairs of agents are swapped, this does not affect the correctness or the silence time of the ranking protocol, due to the symmetry among agents.
- **Decision.** If n is even, the agent with `rank` = $n/2$ decides the exact majority opinion based on the inputs of both itself and the agent with `rank` = $n/2 + 1$ (lines 12–16). Otherwise, the agent with `rank` = $\lceil n/2 \rceil$ decides the exact majority opinion based on its own input (lines 17–18).
- **Propagation.** After completing the above steps, the agent with `rank` = $\lceil n/2 \rceil$ checks the opinions of the other agents. If there is an agent with a different opinion, `PROPAGATE-RESET` is triggered, and the correct opinion is propagated during resetting. (lines 2–8 and 19–24).

In what follows, we explain the details of the Decision part and the Propagation part.

In the Decision part, agents decide the exact majority opinion. We consider the situation where the Ranking part and the Swapping part are completed. If n is even, the exact majority opinion is determined by the agents with `rank` = $n/2$ and `rank` = $n/2 + 1$. If the inputs of both agents are the same, then the agents with that input are in the majority, making that input the exact majority opinion. If not, the number of agents with input `A` and input `B` is the same, so the exact majority opinion is `T`.

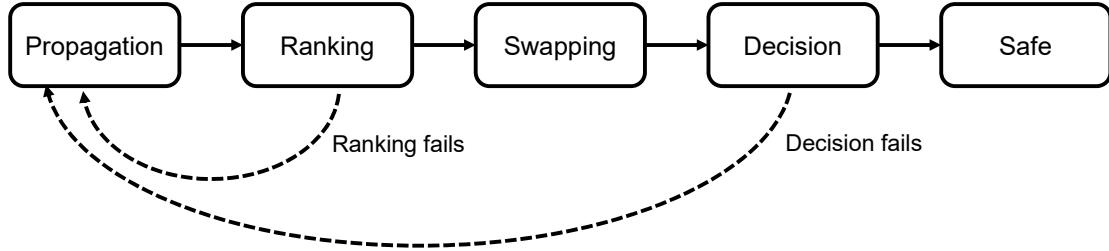


Figure 1: Flow of \mathcal{P}_{EM}

If n is odd, the exact majority opinion is the input of the agent with $\text{rank} = \lceil n/2 \rceil$ since the number of agents with that input constitutes a majority.

In the Propagation part, the agent with $\text{rank} = \lceil n/2 \rceil$ propagates its `answer` to all agents. However, if the agent with $\text{rank} = \lceil n/2 \rceil$ shares its `answer` to every agent by a direct interaction, it would take $\Theta(n \log n)$ time, since it is equivalent to the coupon collector's problem. To achieve propagation within $O(n)$ time in expectation, we use the following method. The agent with $\text{rank} = \lceil n/2 \rceil$ checks the opinions of the other agents (line 22). It detects that ranking, swapping, and decision are completed when its `timer` reaches 0. To detect this, the agent with $\text{rank} = \lceil n/2 \rceil$ sets its `timer` to $7(t_{\text{rank}} + 4)$ when it is ranked (lines 5–6) and decreases it by 1 when it interacts with the agent with $\text{rank} = n$ (line 20). If there is an agent with a different opinion, it can be detected within $O(n)$ time in expectation and triggers `PROPAGATE-RESET` by setting its `role` to `Resetting` (lines 21–24). Note that the variables `leader` and `resetcount` in the pseudocode are used in the ranking protocol (see Section 3.2). During `PROPAGATE-RESET`, that is, while the `role` of agents is `Resetting`, the correct opinion is propagated to all agents. Specifically, any agent with `role` = `Resetting` other than the agent with $\text{rank} = \lceil n/2 \rceil$ sets its `answer` to ϕ when its `role` becomes `Resetting` (lines 3–4). Then, the exact majority opinion overwrites ϕ via the epidemic protocol³ (lines 7–8). After the succeeding ranking, swapping, and decision, the agent with $\text{rank} = \lceil n/2 \rceil$ again decides the correct opinion (lines 12–18). At this time, all agents have the correct opinion; thus, `PROPAGATE-RESET` will not occur. These process takes $O(n)$ time in expectation.

To summarize the above, \mathcal{P}_{EM} is as shown in Figure 1. First, agents execute ranking, swapping, and decision. After that, if `answer` of some agent is incorrect, the output of the agent with $\text{rank} = \lceil n/2 \rceil$ is propagated to all agents by restarting the ranking. After re-ranking, re-swapping, and re-decision, the outputs of all agents are identical and correct. Note that these transitions in the figure are probabilistic. Thus, if ranking or decision fails, agents also restart the protocol from the beginning by resetting via `PROPAGATE-RESET`.

5.2.1 Correctness and Analysis

In this part, we prove that an execution of \mathcal{P}_{EM} under a uniform random scheduler reaches a silent configuration within $O(n)$ time in expectation and $O(n \log n)$ time with high probability. To prove it,

³In the epidemic protocol, the exact majority opinion is treated as 1, and ϕ is treated as 0.

we define the sets of configurations as follows:

$$\begin{aligned}\mathcal{S}_{rank} &= \{C \in \mathcal{C}_{all}(\mathcal{P}_{EM}) \mid \forall v \in V : C(v).\text{role} = \text{Settled} \wedge \forall u, \forall v \in V : u \neq v \Rightarrow C(u).\text{rank} \neq C(v).\text{rank}\}, \\ \mathcal{S}_{swap} &= \{C \in \mathcal{S}_{rank} \mid \forall u, \forall v \in V : C(u).\text{input} = A \wedge C(v).\text{input} = B \Rightarrow C(u).\text{rank} < C(v).\text{rank}\}, \\ \mathcal{T}_{swap} &= \{C \in \mathcal{S}_{rank} \mid \exists v \in V : C(v).\text{rank} = \lceil n/2 \rceil \wedge C(v).\text{timer} \geq 28\}, \\ \mathcal{S}_{dec} &= \{C \in \mathcal{S}_{swap} \mid \forall v \in V : C(v).\text{rank} = \lceil n/2 \rceil \Rightarrow C(v).\text{answer} \text{ is the exact majority opinion}\}, \\ \mathcal{S}_{out} &= \{C \in \mathcal{S}_{rank} \mid \forall v \in V : C(v).\text{answer} \text{ is the exact majority opinion}\}, \\ \mathcal{S}_{tim} &= \mathcal{S}_{swap} \cap \mathcal{S}_{out}, \text{ and} \\ \mathcal{S}_{em} &= \{C \in \mathcal{S}_{tim} \mid \forall v \in V : C(v).\text{rank} = \lceil n/2 \rceil \Rightarrow C(v).\text{timer} = 0\}.\end{aligned}$$

Lemma 5. \mathcal{S}_{em} is a set of silent configurations.

Proof. In a configuration $C_0 \in \mathcal{S}_{em}$, ranking, swapping, and decision are completed, the **timer** of the agent with **rank** = $\lceil n/2 \rceil$ is 0, and all agents' **answer** are the same. Thus, throughout a execution from C_0 , no agent changes its state. \square

Table 2: Stabilization Steps of \mathcal{P}_{EM}

Step	Time	Success Probability	Lemmas
$\mathcal{C}_{all}(\mathcal{P}_{EM}) \rightarrow \mathcal{S}_{rank}$	$O(n)$	1/10	Lemma 6
$\mathcal{S}_{rank} \rightarrow \mathcal{T}_{swap} \cup \mathcal{S}_{tim}$	$O(n)$	1/20	Lemma 8
$\mathcal{T}_{swap} \rightarrow \mathcal{S}_{dec}$	$O(n)$	1/8	Lemma 9
$\mathcal{S}_{dec} \rightarrow \mathcal{S}_{tim}$	$O(n)$	1/1280	Lemma 11
$\mathcal{S}_{tim} \rightarrow \mathcal{S}_{em}$	$O(n)$	1/2	Lemma 12
$\mathcal{C}_{all}(\mathcal{P}_{EM}) \rightarrow \mathcal{S}_{em}$	$O(n)$	1/4096000	Lemma 13

We analyze \mathcal{P}_{EM} by dividing it into several steps, as shown in Table 2.

Lemma 6. Let $C_0 \in \overline{\mathcal{S}_{rank}}$ and $\Xi_{\mathcal{P}_{EM}}(C_0, \Gamma)$. The probability that the execution reaches \mathcal{S}_{rank} within $O(n)$ time is at least 1/10.

Proof. First, we consider the case that the agent with **rank** = $\lceil n/2 \rceil$ does not start PROPAGATE-RESET by interacting with an agent that has a different **answer** until the execution reaches \mathcal{S}_{rank} . In this case, from Lemma 3, agents complete the ranking within s_{rank} expected time. From Markov's inequality, the probability that ranking has completed within $2 \cdot s_{rank}$ time is at least 1/2.

Second, we consider the case where the agent with **rank** = $\lceil n/2 \rceil$ starts PROPAGATE-RESET by interacting with an agent that has a different **answer** until the execution reaches \mathcal{S}_{rank} . In this case, after PROPAGATE-RESET, an agent will be assigned the rank $\lceil n/2 \rceil$ within $2 \cdot s_{rank}$ time with at least 1/2 probability from Markov's inequality. At the time, the agent sets **timer** to $7(t_{rank} + 4)$. We analyze the probability that the **timer** does not reach 0 before the ranking is completed. Since the silence time of the ranking is $s_{rank} \leq t_{rank} \cdot n$, we analyze the probability that **timer** does not reach 0 during $t_{rank} \cdot n^2$ interactions. Let $Z \sim B(n^2 \cdot t_{rank}, 2/(n(n-1)))$ be a binomial random variable representing the number of times the agent with **rank** = $\lceil n/2 \rceil$ interacts with the agent with **rank** = n during $n^2 \cdot t_{rank}$ interactions. From the Chernoff bound (Eq. (4.2) in [20], with $\delta = 2/3$), $\Pr(Z \geq (1 + 2/3)E[Z]) \leq e^{-2/3(2/3)^2} < 4/5$. Since $(1 + 2/3)E[Z] \leq \frac{20}{3}t_{rank} < 7t_{rank}$, that probability is at last 1/5. Thus, the execution reaches \mathcal{S}_{rank} with probability at least 1/10. \square

Lemma 7. Let $C_0 \in \mathcal{S}_{rank}$ and $\Xi_{\mathcal{P}_{EM}}(C_0, \Gamma)$. If PROPAGATE-RESET does not occur, the expected time for the execution to reach \mathcal{S}_{swap} is at most n .

Proof. Let m be the number of agents with input A. Let V_B be the set of agents with input B and **rank** at most m , and let V_A be the set of agents with input A and **rank** greater than m at initial

configuration. Note that $|V_A| = |V_B|$. The sizes of V_A and V_B each decrease by 1 if and only if an interaction occurs between $a \in V_A$ and $b \in V_B$. Additionally, in any other interactions, the sizes of V_A and V_B remain unchanged. The probability that an interaction occurs between $a \in V_A$ and $b \in V_B$ for each interaction is $\frac{2|V_A||V_B|}{n(n-1)}$. Thus, the number of interactions until $|V_A|$ and $|V_B|$ become 0 is $\sum_{i=1}^{|V_A|} \frac{n(n-1)}{2i^2} = \frac{n(n-1)}{2} \sum_{i=1}^{|V_A|} \frac{1}{i^2} < \frac{n(n-1)}{2} \cdot \frac{\pi^2}{6} < n^2$. Therefore, after dividing by n , the expected time for the execution to reach \mathcal{S}_{swap} is at most n . \square

Lemma 8. *Let $C_0 \in \mathcal{S}_{rank} \cap \overline{\mathcal{T}_{swap}}$ and $\Xi_{\mathcal{P}_{EM}}(C_0, \Gamma)$. The probability that the execution reaches $\mathcal{T}_{swap} \cup \mathcal{S}_{tim}$ within $O(n)$ time is at least $1/20$.*

Proof. First, we show that the expected time until either PROPAGATE-RESET occurs or the execution reaches \mathcal{S}_{dec} is at most $1/2 + (n - 1)/2$. We will derive this by analyzing the time to reach \mathcal{S}_{dec} without PROPAGATE-RESET occurring. From Lemma 7, the execution reaches \mathcal{S}_{swap} within n expected time. If n is even, for the execution starting from a configuration belonging to \mathcal{S}_{swap} to reach \mathcal{S}_{dec} , it is sufficient for the agent with $\text{rank} = n/2$ and the agent with $\text{rank} = n/2 + 1$ to interact. Since the probability that those two agents interact in each interaction is $2/(n(n-1))$, the expected time for this to happen is $(n-1)/2$. If n is odd, for the execution starting from a configuration belonging to \mathcal{S}_{swap} to reach \mathcal{S}_{dec} , it is sufficient for the agent with $\text{rank} = \lceil n/2 \rceil$ to participate in an interaction. Since the probability of that agent participating in an interaction is $2/n$, the expected time is $1/2$. Thus, the execution starting from C_0 reaches \mathcal{S}_{dec} within $1/2 + (n-1)/2$ in expectation.

After the execution reaches \mathcal{S}_{dec} , if there are agents with an **answer** that is not the exact majority opinion, an interaction between the agent with $\text{rank} = \lceil n/2 \rceil$ and such an agent will trigger PROPAGATE-RESET. The probability of this happening is at least $2/(n(n-1))$, so the expected time is $(n-1)/2$. If the **answer** of all agents are the exact majority opinion, the execution has reached to \mathcal{S}_{tim} . Therefore, the expected time the execution starting from C_0 reaches \mathcal{S}_{tim} is bounded by $1/2 + (n-1)/2 + (n-1)/2 = n - 1/2$. From Markov's inequality, the probability that the execution reaches \mathcal{S}_{tim} within $2n - 1$ time is at least $1/2$.

Now, we analyze the time and probability for the execution to reach \mathcal{T}_{swap} after PROPAGATE-RESET occurs. From Lemma 3 and Markov's inequality, if the agent with $\text{rank} = \lceil n/2 \rceil$ does not trigger PROPAGATE-RESET again, the execution reaches \mathcal{S}_{rank} within $2 \cdot s_{rank}$ time with probability at least $1/2$. During the ranking, when the agent with $\text{rank} = \lceil n/2 \rceil$ is created, its **timer** is set to $7(t_{rank} + 4)$. Thus, we analyze the probability that its **timer** ≥ 28 when the execution reaches \mathcal{S}_{rank} . Since $s_{rank} \leq t_{rank} \cdot n$, we analyze the probability that **timer** ≥ 28 after $n^2 \cdot t_{rank}$ interactions. Let $Z \sim B(n^2 \cdot t_{rank}, 2/(n(n-1)))$ be a binomial random variable representing the number of times the agent with $\text{rank} = \lceil n/2 \rceil$ interacts with the agent with $\text{rank} = n$ during $n^2 \cdot t_{rank}$ interactions. From the Chernoff bound (Eq. (4.2) in [20], with $\delta = 2/3$), $\Pr(Z \geq (1 + 2/3)E[Z]) \leq e^{-2/3(2/3)^2} < 4/5$. Since $(1 + 2/3)E[Z] \leq \frac{20}{3}t_{rank} < 7t_{rank}$, that probability is at last $1/5$. Therefore, after PROPAGATE-RESET occurs, the execution reaches \mathcal{T}_{swap} within $O(n)$ time with probability at least $1/10$.

From the above, the execution reaches $\mathcal{T}_{swap} \cup \mathcal{S}_{tim}$ within $O(n)$ time with probability at least $1/20$. \square

Lemma 9. *Let $C_0 \in \mathcal{T}_{swap}$ and $\Xi_{\mathcal{P}_{EM}}(C_0, \Gamma)$. The probability that the execution reaches \mathcal{S}_{dec} within $3n - 1$ time is at least $1/8$.*

Proof. We analyze the case where the execution reaches \mathcal{S}_{dec} without PROPAGATE-RESET occurring.

First, we analyze the time and probability for the execution to reach \mathcal{S}_{dec} if PROPAGATE-RESET does not occur. From Lemma 7 and Markov's inequality, agents finish swapping within $2n$ time with probability at least $1/2$. After swapping, decision is finished within $(n - 1)/2$ time in expectation as shown in the proof of Lemma 8. From Markov's inequality, agents finish decision within $n - 1$ time with probability at least $1/2$. Therefore, agents finish swapping and decision within $3n - 1$ time with probability at least $1/4$.

Next, we analyze the probability that PROPAGATE-RESET does not occur within $4n$ time. Let $Z \sim B(4n^2, 2/(n(n-1)))$ be a binomial random variable representing the number of times the agent with

$\text{rank} = \lceil n/2 \rceil$ interacts with the agent with $\text{rank} = n$ during $4n^2$ interactions. From the Chernoff bound (Eq. (4.2) in [20], with $\delta = 2/3$) and $E[Z] \geq 8$, $\Pr(Z \geq (1 + 2/3)E[Z]) \leq e^{-8/3(2/3)^2} \leq 1/2$. Since $(1 + 2/3)E[Z] \leq 80/3 < 27$, the **timer** of the agent with $\text{rank} = \lceil n/2 \rceil$ does not reach 0 during $4n$ time with probability at least $1/2$.

From the above, the probability for the execution to reach \mathcal{S}_{dec} within $3n - 1$ time is at least $1/8$. \square

Lemma 10. *Let $C_0 \in \mathcal{S}_{rank}$ and $\Xi_{\mathcal{P}_{EM}}(C_0, \Gamma)$. The variable **timer** of the agent with $\text{rank} = \lceil n/2 \rceil$ reaches 0 within $O(n)$ time with probability at least $1/2$.*

Proof. Let x be the value of the **timer** of the agent with $\text{rank} = \lceil n/2 \rceil$. We analyze the expected time for the agent with $\text{rank} = \lceil n/2 \rceil$ to interact with the agent with $\text{rank} = n$ exactly x times. Since the probability that those two agents interact is $2/(n(n-2))$, the expected time to interact once is $(n-1)/2$. By the linearity of expectation, the expected time for the **timer** to reach 0 is $x(n-1)/2$. From Markov's inequality, the **timer** reaches 0 within $x(n-1) \leq 7(t_{rank} + 4)(n-1) = O(n)$ time with probability at least $1/2$. \square

Lemma 11. *Let $C_0 \in \mathcal{S}_{dec}$ and $\Xi_{\mathcal{P}_{EM}}(C_0, \Gamma)$. The probability that the execution reaches \mathcal{S}_{tim} within $O(n)$ time is at least $1/1280$.*

Proof. If there is no agent whose **answer** is not the exact majority opinion in C_0 , then $C_0 \in \mathcal{S}_{tim}$.

Now, we consider the case where there are agents whose **answer** is not the exact majority opinion. In this case, after the **timer** of the agent with $\text{rank} = \lceil n/2 \rceil$ reaches 0, when the agent interacts with an agent whose **answer** is not the exact majority opinion, PROPAGATE-RESET occurs, and the exact majority opinion propagates to all agents. The variable **timer** of the agent with $\text{rank} = \lceil n/2 \rceil$ reaches 0 within $O(n)$ time with probability at least $1/2$, from Lemma 10. Since the probability that agent with $\text{rank} = \lceil n/2 \rceil$ interacts with the agent whose **answer** is not the exact majority opinion is at least $2/(n(n-1))$, the expected time is $(n-1)/2$. From Markov's inequality, such an interaction occurs within $n-1$ time with probability at least $1/2$. Thus, PROPAGATE-RESET occurs within $O(n)$ time with probability at least $1/4$.

During PROPAGATE-RESET, the exact majority opinion propagates to all agents within $O(\log n)$ time with high probability from Lemma 1. Since PROPAGATE-RESET occurs, agents also execute OPTIMAL-SILENT-SSR. Note that PROPAGATE-RESET finishes within $O(n)$ time with high probability, and OPTIMAL-SILENT-SSR finishes within $2 \cdot O(n)$ time with probability at least $1/2$ from Lemma 2 and Markov's inequality. After both protocols finished correctly, the **timer** of the agent with $\text{rank} = \lceil n/2 \rceil$ is no less than 28 with probability at least $1/10$ from the latter part of the proof of Lemma 8. Thus, the execution reaches $\mathcal{T}_{swap} \cap \mathcal{S}_{out}$ within $O(n)$ time with probability at least $1/4 \cdot 1/10 \cdot (1 - 1/n)^2 \geq 1/160$.

From Lemma 9, the execution belonged \mathcal{T}_{swap} reaches \mathcal{S}_{dec} within $3n - 1$ time with probability $1/8$. Thus, since there is no agent whose **answer** is not the exact majority opinion, the execution reaches $\mathcal{S}_{dec} \cap \mathcal{S}_{out} \subseteq \mathcal{S}_{tim}$. Therefore, the probability that the execution reaches \mathcal{S}_{tim} within $O(n)$ time is at least $1/160 \cdot 1/8 = 1/1280$. \square

Lemma 12. *Let $C_0 \in \mathcal{S}_{tim}$ and $\Xi_{\mathcal{P}_{EM}}(C_0, \Gamma)$. The probability that the execution reaches \mathcal{S}_{em} within $O(n)$ time is at least $1/2$.*

Proof. For the execution to reach \mathcal{S}_{em} , it is necessary for the **timer** of the agent with $\text{rank} = \lceil n/2 \rceil$ to reach 0. From Lemma 10, the **timer** of the agent with $\text{rank} = \lceil n/2 \rceil$ reaches 0 within $O(n)$ time with probability at least $1/2$. Thus, the execution reaches \mathcal{S}_{em} within $O(n)$ time with probability at least $1/2$. \square

Lemma 13. *\mathcal{P}_{EM} reaches \mathcal{S}_{em} with probability 1, and the silence time of \mathcal{P}_{EM} is $O(n)$ in expectation, and $O(n \log n)$ with high probability.*

Proof. Let $C_0 \in \mathcal{C}_{all}(\mathcal{P}_{EM})$ and $\Xi_{\mathcal{P}_{EM}}(C_0, \Gamma)$. From Lemma 6, 8, 9, 11, and 12, the execution reaches \mathcal{S}_{em} with probability at least $1/4096000$. Since an execution reaches \mathcal{S}_{em} with nonzero probability, the

execution eventually reaches \mathcal{S}_{em} if it runs long enough. Thus, there exists an execution that reaches a silent configuration from any initial configuration; therefore, \mathcal{P}_{EM} reaches \mathcal{S}_{em} with probability 1.

Let $E_{\mathcal{P}_{EM}}$ be the expected time for the execution to reach \mathcal{S}_{em} . From the above, $E_{\mathcal{P}_{EM}} \leq O(n) + (1 - 1/4096000)E_{\mathcal{P}_{EM}}$ holds. Thus, $E_{\mathcal{P}_{EM}} = O(n)$. From Markov's inequality, the execution reaches \mathcal{S}_{em} within $2 \cdot O(n)$ time with probability at least $1/2$. Thus, the execution reaches \mathcal{S}_{em} within $2 \cdot O(n) \log n = O(n \log n)$ time with probability at least $1 - (1/2)^{\log n} = 1 - 1/n$. \square

From Theorem 2, 3, and Lemma 13, we derive the following theorem.

Theorem 4. \mathcal{P}_{EM} is a time- and space-optimal silent self-stabilizing exact majority protocol that uses $O(n)$ states and reaches a silent configuration within $O(n)$ time in expectation and within $O(n \log n)$ time with high probability.

6 Conclusion and Discussions

We addressed a silent protocol that solves the self-stabilizing exact majority problem. We showed that no protocol can solve this problem without knowledge of n . We proposed a silent protocol within $O(n)$ time in expectation and $O(n \log n)$ time with high probability, using $O(n)$ states, with knowledge of n . We established lower bounds, proving that any protocol requires $\Omega(n)$ states, $\Omega(n)$ time in expectation, and $\Omega(n \log n)$ time with high probability to reach a safe configuration. Thus, the proposed protocol is time- and space-optimal.

We propose the following open problems: An extension of the self-stabilizing majority problem to the exact plurality consensus problem [6], in which agents have k different opinions, can also be considered. Whether this problem can achieve the lower bound shown in this paper remains an intriguing open question. Additionally, it is worth considering a non-silent self-stabilizing exact majority protocol with fewer states.

Acknowledgments. This work was supported by JSPS KAKENHI Grant Numbers JP22H03569, JP23K28037, and JP25K03101.

References

- [1] Dan Alistarh, James Aspnes, and Rati Gelashvili. Space-optimal majority in population protocols. In *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms*, page 2221–2239, 2018.
- [2] Dan Alistarh, Rati Gelashvili, and Milan Vojnović. Fast and exact majority in population protocols. In *Proceedings of the 2015 ACM Symposium on Principles of Distributed Computing*, page 47–56, 2015.
- [3] Dana Angluin, James Aspnes, Zoë Diamadi, Michael J. Fischer, and René Peralta. Computation in networks of passively mobile finite-state sensors. *Distributed Computing*, 18(4):235–253, 2006.
- [4] Dana Angluin, James Aspnes, and David Eisenstat. Fast computation by population protocols with a leader. In *Distributed Computing*, pages 61–75, 2006.
- [5] Dana Angluin, James Aspnes, and David Eisenstat. A simple population protocol for fast robust approximate majority. *Distributed Computing*, 21(2):87–102, 2008.
- [6] Gregor Bankhamer, Petra Berenbrink, Felix Biermeier, Robert Elsässer, Hamed Hosseinpour, Dominik Kaaser, and Peter Kling. Population protocols for exact plurality consensus: How a small chance of failure helps to eliminate insignificant opinions. In *Proceedings of the 2022 ACM Symposium on Principles of Distributed Computing*, page 224–234, 2022.

- [7] Stav Ben-Nun, Tsvi Kopelowitz, Matan Kraus, and Ely Porat. An $O(\log 3/2 n)$ parallel time population protocol for majority with $O(\log n)$ states. In *Proceedings of the 39th Symposium on Principles of Distributed Computing*, PODC '20, page 191–199, New York, NY, USA, 2020.
- [8] Petra Berenbrink, Felix Biermeier, Christopher Hahn, and Dominik Kaaser. Loosely-stabilizing phase clocks and the adaptive majority problem. In *1st Symposium on Algorithmic Foundations of Dynamic Networks (SAND 2022)*, volume 221, pages 7:1–7:17, 2022.
- [9] Petra Berenbrink, Robert Elsässer, Tom Friedetzky, Dominik Kaaser, Peter Kling, and Tomasz Radzik. A population protocol for exact majority with $O(\log 5/3 n)$ stabilization time and $\Theta(\log n)$ states. In *32nd International Symposium on Distributed Computing (DISC 2018)*, volume 121, pages 10:1–10:18, 2018.
- [10] Petra Berenbrink, Robert Elsässer, Tom Friedetzky, Dominik Kaaser, Peter Kling, and Tomasz Radzik. Time-space trade-offs in population protocols for the majority problem. *Distributed Computing*, 34(2):91–111, 2021.
- [11] Andreas Bilke, Colin Cooper, Robert Elsässer, and Tomasz Radzik. Brief announcement: Population protocols for leader election and exact majority with $O(\log 2 n)$ states and $O(\log 2 n)$ convergence time. In *Proceedings of the ACM Symposium on Principles of Distributed Computing*, page 451–453, 2017.
- [12] Janna Burman, Ho-Lin Chen, Hsueh-Ping Chen, David Doty, Thomas Nowak, Eric Severson, and Chuan Xu. Time-optimal self-stabilizing leader election in population protocols. In *Proceedings of the 2021 ACM Symposium on Principles of Distributed Computing*, PODC'21, page 33–44, 2021.
- [13] Shukai Cai, Taisuke Izumi, and Koichi Wada. How to prove impossibility under global fairness: On space complexity of self-stabilizing leader election on a population protocol model. *Theory of Computing Systems*, 50(3):433–445, 2012.
- [14] Ioannis Chatzigiannakis, Shlomi Dolev, Sándor Fekete, Othon Michail, and Paul Spirakis. On the fairness of probabilistic schedulers for population protocols. In *Algorithmic Methods for Distributed Cooperative Systems*, volume 9371, pages 1–23, 2010.
- [15] Anne Condon, Monir Hajiaghayi, David Kirkpatrick, and Ján Maňuch. Approximate majority analyses using tri-molecular chemical reaction networks. *Natural Computing*, 19(1):249–270, 2020.
- [16] David Doty, Mahsa Eftekhari, Leszek Gąsieniec, Eric Severson, Przemysław Uznański, and Grzegorz Stachowiak. A time and space optimal stable population protocol solving exact majority. In *2021 IEEE 62nd Annual Symposium on Foundations of Computer Science (FOCS)*, pages 1044–1055, 2022.
- [17] Moez Draief and Milan Vojnovic. Convergence speed of binary interval consensus. In *2010 Proceedings IEEE INFOCOM*, pages 1–9, 2010.
- [18] Leszek Gąsieniec, Tytus Grodzicki, and Grzegorz Stachowiak. Near-state and state-optimal self-stabilising leader election population protocols, 2025.
- [19] Adrian Kosowski and Przemysław Uznanski. Brief announcement: Population protocols are fast. In *Proceedings of the 2018 ACM Symposium on Principles of Distributed Computing*, PODC '18, page 475–477, 2018.
- [20] Michael Mitzenmacher and Eli Upfal. *Probability and computing: randomized algorithms and probabilistic analysis*. Cambridge University Press, 2005.
- [21] Yves Mocquard, Emmanuelle Anceaume, James Aspnes, Yann Busnel, and Bruno Sericola. Counting with population protocols. In *2015 IEEE 14th International Symposium on Network Computing and Applications*, 2015.

- [22] Yves Mocquard, Emmanuelle Anceaume, and Bruno Sericola. Optimal proportion computation with population protocols. In *2016 IEEE 15th International Symposium on Network Computing and Applications (NCA)*, pages 216–223, 2016.
- [23] E. Perron, D. Vasudevan, and M. Vojnovic. Using three states for binary consensus on complete graphs. In *IEEE INFOCOM 2009*, pages 2527–2535, 2009.
- [24] Yuichi Sudo, Junya Nakamura, Yukiko Yamauchi, Fukuhito Ooshita, Hirotsugu Kakugawa, and Toshimitsu Masuzawa. Loosely-stabilizing leader election in a population protocol model. *Theoretical Computer Science*, 444:100–112, 2012.
- [25] Hiroto Yasumi, Fukuhito Ooshita, Ken’ichi Yamaguchi, and Michiko Inoue. Space-optimal population protocols for uniform bipartition under global fairness. *IEICE Transactions on Information and Systems*, E102.D(3):454–463, 2019.