Constructing Self-Stabilizing Oscillators in Population Protocols

Colin Cooper[1], Anissa Lamani[2], Giovanni Viglietta[3], Masafumi Yamashita[2], and Yukiko Yamauchi[2]

[1] Department of Informatics, Kings College, United Kingdom
[2] Department of Informatics, Graduate School of ISEE, Kyushu University, Japan
[3] School of Electrical Engineering and Computer Science, University of Ottawa, Canada

Abstract. Population protocols (PPs) are a model of passive distributed systems in which a collection of finite-state mobile agents interact with each other to accomplish a common task. Unlike other works, which investigate their computation power, this paper throws light on an aspect of PPs as a model of chemical reactions. Motivated by the well-known *BZ reaction* that provides an autonomous chemical oscillator, we address the problem of autonomously generating an oscillatory execution from any initial configuration (i.e., in a self-stabilizing manner). For deterministic PPs, we show that the self-stabilizing leader election (SS-LE) and the self-stabilizing oscillator problem (SS-OSC) are equivalent, in the sense that an SS-OSC protocol is constructible from a given SS-LE protocol and vice versa, which unfortunately implies that (1) resorting to a leader is inevitable (although we seek a decentralized solution) and (2) n states are necessary to create an oscillation of amplitude n, where n is the number of agents (although we seek a memory-efficient solution). Aiming at reducing the space complexity, we present and analyze some randomized oscillatory PPs.

Keywords: Population protocol, Self-stabilization, Oscillatory behavior, Leader election, Distributed algorithm

1 Introduction

The motivation of our study is to understand how autonomy emerges in biological systems, and to apply such understanding in giving artificial distributed systems autonomous properties. Specifically, we focus on self-oscillations, which play fundamental roles in autonomous biological reactions, and investigate them as a phenomenon in distributed computing. Self-oscillations are often understood as a chemical oscillator provided by certain reactions, such as the Belousov–Zhabotinsky reaction. We use the population protocol model for our investigation, since it was introduced in part to model chemical reactions.

The *population protocol* (PP) model introduced by Angluin et al. [2] is a model of passive distributed systems. It is used as a theoretical model of

a collection of finite-state mobile agents that interact with each other in order to solve a given problem in a cooperative fashion. Computations are done through pairwise interactions, i.e., when two agents interact, they exchange their information and update their states accordingly. The interaction pattern, however, is unpredictable, and the agents have no control over which agent they interact with. We thus assume the presence of an abstract mechanism called *scheduler* that chooses at any time instant the pair of agents that interact with each other. PPs can represent not only artificial distributed systems such as sensor networks and mobile agent systems, but also natural distributed systems such as chemical reactions and biological systems.

In the past few years, many problems have been investigated on PPs, including the problems of computing a function, electing a leader, counting, coloring, synchronizing and naming [1–4, 7, 8]. Most of the problems consider the computational power of the population and hence are *static*; the agents are requested to eventually reach a configuration that represents the answer to the given computation problem.

The notion of termination is typically intended in the Noetherian sense (in the context of abstract rewriting systems); however the agents are not requested to eventually terminate, but the execution is requested to repeat the same configuration forever.

Unlike most of the past works in PPs, we throw light on an aspect of PPs as a model of chemical reactions. Specifically, we investigate the problem of designing a PP that stabilizes to an oscillatory execution, no matter from which initial configuration it starts; that is, we explore a *self-stabilizing* PP that generates an oscillatory execution. The problem emerges in the project of designing molecular robots [9], and is directly motivated by the Belousov–Zhabotinsky reaction, which is an example of non-equilibrium thermodynamics providing a non-linear chemical oscillator. We show that under a deterministic scheduler governed by an adversary, the self-stabilizing leader election problem and the self-stabilizing oscillator problem are equivalent, and hence costly in term of space complexity. Aiming at space reduction, we then propose and analyze some approximate solutions assuming a randomized scheduler.

In biological systems, the oscillatory behavior is used as a natural clock to transmit signals and hence transfer information. In artificial distributed systems, PPs that exhibit an oscillatory behavior could be used to distributely and autonomously implement a clock.

Apart from the difference of motivation, a few works on *dynamic* problems are related to our work. Angluin et al. [4] provided a self-stabilizing token circulation protocol in a ring with a pre-selected leader. Beauquier and Burman investigated the self-stabilizing mutual exclusion, group mutual exclusion problems [6] and also the self-stabilizing synchronization

problem [5]. In the latter work, they have shown that the synchronization problem in the PP model under a deterministic scheduler is impossible to solve, and hence they proposed a solution assuming the presence of an unlimited-resource agent called *Base Station*. Our problem also belongs to the class of dynamic problems.

After introducing some concepts and notions in Section 2, we consider a *deterministic scheduler* governed by an adversary in Section 3. Under this scheduler, we focus on the self-stabilizing oscillator (SS-OSC) problem. We show that the self-stabilizing leader election (SS-LE) problem and the SS-OSC problem are equivalent; that is, an SS-OSC protocol is constructible from a given SS-LE protocol, and vice versa. In Section 4, we consider a *probabilistic scheduler*, i.e., the interacting agents are chosen uniformly at random. Under the probabilistic scheduler, we present and analyze some oscillatory PPs, mainly aiming to reduce space complexity. Section 5 is devoted to the conclusions.

In this paper, we use results from [7] that concern the SS-LE problem. In [7], it has been shown that the SS-LE is impossible to solve with less than n states where n is the size of the population. Also, a PP that solves the SS-LE was proposed. The protocol ensures that eventually each agent has unique state.

2 Preliminaries

In this paper, we consider a population of n anonymous finite-state agents that update their state by interacting with other agents. We consider only pairwise interactions, i.e., each interaction involves exactly two agents. When two agents interact, they update their state according to a common protocol. We denote by $A = \{0, 1, \ldots, n-1\}$, the set of agents in the population, that is, $|A| = n$. Indices are used for notation purposes only; in fact, the agents are anonymous, i.e., they have no identity and cannot be distinguished from each other. Any pair of agents a_1 and a_2 ($a_1 \neq a_2$) in the population are susceptible to interact and the interactions are undirected.

A *protocol* $P = (Q, \delta)$ is a pair of a finite set of states Q and a transition function $\delta : Q \times Q \to Q \times Q$. When two agents interact with each other, δ determines the next state of both agents. Let p and q be the states of agents a_1 and a_2, respectively. $\delta(p, q) = (p', q')$ indicates that the states of agents a_1 and a_2, after interacting with each other, are p' and q', respectively. We assume that if $\delta(p, q) = (p', q')$ then $\delta(q, p) = (q', p')$.

A *configuration* C is a mapping $A \to Q$ that specifies the state of all the agents in the population. By $C(i)$, we refer to the state of agent i in configuration C. By $\mathcal{C}$ we refer to the set of all possible configurations of the system. Given a configuration $C \in \mathcal{C}$ and an interaction between the two agents a_1 and a_2, $r = (a_1, a_2)$, we say that

C' is obtained from C via the interaction r, denoted by $C \xrightarrow{r} C'$, if $(C'(a_1), C'(a_2)) = \delta(C(a_1), C(a_2))$.

Let C_t be the configuration at time t and let r_t be the interaction on C_t at time t. An *execution* E of a protocol P is a sequence of configurations and transitions $(C_0, r_0, C_1, r_1, \dots)$ such that $\forall\ i \geq 0$, r_i is a transition of δ and $C_i \xrightarrow{r_i} C_{i+1}$. When a configuration C' is reachable from C after a finite number of transitions we note $C \xrightarrow{*} C'$.

A *scheduler* chooses a pair of agents to interact at each time $t \geq 0$. In this paper, we consider two types of schedulers: (i) A deterministic but globally fair scheduler that guarantees that if there is a configuration that is reachable infinitely often, then the configuration is eventually reached. (ii) A uniform random scheduler, i.e., the pair of agents that are chosen for the interaction are selected at random, independently and uniformly from the set of all the agents in the system.

In the sequel, we define some important notions that will be used in the paper.

Definition 1. ***(Oscillation)*** *Let $f\colon [a,b] \subset \mathbb{N} \to \mathbb{R}$ be a function. We say that f is an oscillation if there exists $c \in \mathbb{N}$ such that:*

1. $a < c < b$,
2. $f(a) < f(c) > f(b)$,
3. *f is weakly increasing in $[a,c]$ and weakly decreasing in $[c,b]$.*

The value $f(c) - (f(a)+f(b))/2$ is called the amplitude *of the oscillation and is denoted by ι_a, whereas $b-a$ is called the* period *of the oscillation and is denoted by ι_p. The* increasing phase *(respectively,* decreasing phase*) of the oscillation is the interval in which f is weakly increasing (respectively, weakly decreasing).*

Definition 2. ***(Oscillatory behavior)*** *Given an execution E of a population protocol $\mathcal{P}$ on n agents and a set of states S, let $f_S : \mathbb{N} \to [0,n] \subset \mathbb{N}$ be the function mapping a time instant t into the number of agents whose state is in S at time t. Let $\{t_0, t_1, \dots\}$ be a strictly increasing sequence of time instants. We say that E exhibits an oscillatory behavior for the set of states S, if for every $i \geq 0$, the restriction of f_S to $[t_i, t_{i+1}]$ is an oscillation.*

Note that, according to the previous definitions, any execution exhibits oscillatory behavior, unless the number of agents whose state is in S eventually stabilizes. However, we are also interested in evaluating the "quality" of the oscillations, in terms of their amplitude and period.

3 Deterministic scheduler

We investigate in this section the problem of generating oscillatory executions under a global fair deterministic scheduler and starting from an arbitrary configuration. We show that the SS-LE problem and SS-OSC problem are equivalent. By using the results in [7], we can deduce then, that the SS-OSC problem is impossible to solve if $|Q| < n$ or if n is arbitrary.

Let us first define the deterministic self-stabilizing oscillator.

Definition 3. ***(Deterministic oscillator)*** *A population of agents executing a deterministic protocol $\mathcal{P}$, under a globally fair scheduler, is a (C, S, ι_a, ι_p)-oscillator if any execution $E = (C, r, C', r', \dots)$ of $\mathcal{P}$ exhibits an oscillatory behavior for the set of states S, with amplitude ι_a and period ι_p.*

Definition 4. ***(Deterministic self-stabilizing oscillator)*** *A population of agents executing a deterministic protocol $\mathcal{P}$, under a globally fair scheduler, is a self-stabilizing oscillator for the set of states S if, starting from an arbitrary configuration $C_0 \in \mathcal{C}$, every execution $E = (C_0, r_0, C_1, r_1, \dots)$ of Protocol $\mathcal{P}$, reaches a configuration $C \in \mathcal{C}$ such that (C, S, ι_a, ι_p) is a deterministic oscillator.*

Observe that since the deterministic globally fair scheduler can delay any particular transition of the system for an arbitrary amount of time, it is not possible to bound the period using the classical definition of an interaction. Hence, we use the notion of active interactions that was introduced in [7]. Basically, an interaction r is said to be active in a given configuration $C \in \mathcal{C}$, if it updates the state of at least one of the two agents that have participated in r. More precisely, an interaction $r = (a, b)$, is said to be active in $C \in \mathcal{C}$, if $C \xrightarrow{r} C'$ and either $C(a) \neq C'(a)$ or $C(b) \neq C('b)$.

When ι_p is omitted, it means that the period of the oscillator is not specified, or that the oscillator is not periodic (i.e., not all oscillations have the same period). We consider in the following (C, S, n, ι_p)-oscillators.

Starting from an arbitrary initial configuration $C_0 \in \mathcal{C}$, we show the following two results:

1. If the SS-LE problem is solvable using M_{LE} states, then it is possible to solve the SS-OCS problem using $M_{LE} + O(n)$ states.
2. If the SS-OSC problem is solvable using M_{OSC} states, then it is possible to solve the SS-LE problem using $M_{OSC} + O(1)$ states.

(1) SS-LE $\Rightarrow$ SS-OSC. We show that a deterministic population protocol $\mathcal{P}_{OSC}$ exists using $M_{LE} + 2^n$ states per agent (M_{LE} being the

number of states necessary to solve the self-stabilizing leader election problem). A solution with $M_{LE} + O(n)$ states is also presented later in Discussion (1). The idea of the solution is as follows: we combine our SS-OSC protocol $\mathcal{P}_{OSC}$ with the SS-LE protocol $\mathcal{P}_{LE}$ proposed in [7] and that uses n distinct states per agent. When an interaction occurs between two agents, the two agents execute the enabled actions of both $\mathcal{P}_{OSC}$ and $\mathcal{P}_{LE}$. Protocol $\mathcal{P}_{LE}$ ensures that eventually one leader is elected and all the agents have a unique state [7]. Our solution takes advantage of this "identification" to create an oscillatory behavior. Indeed, using the identification created by Protocol $\mathcal{P}_{LE}$, the leader can somehow remember the agents it has already interacted with.

The state of each agent a_i consists of a triplet of variables $(id_{a_i}, p_{a_i}, T_{a_i})$. Variable id_{a_i} is used by Protocol $\mathcal{P}_{LE}$ ($id_{a_i} \in \{0, 1, 2, \ldots, n-1\}$), where 0 is the leader's state. According to [7], eventually each agent has a unique value for id_{a_i}. Variable $p_{a_i} \in \{0, 1\}$, indicates the phase of the oscillation Agent a_i is part of (increasing or decreasing phase). Variable T_{a_i} is an array of n entries such that $\forall\ j \in \{1, \ldots, n-1\}$, $T_{a_i}[j] \in \{0, 1\}$. The array is used only by the leader to keep track of the agents the leader has already interacted with, hence, in the sequel, the state of a non-leader agent a_j is only represented by the pair (id_{a_j}, p_{a_j}). Let a_i and a_j be the two interacting agents at time t. Without loss of generality, assume that $id_{a_i} < id_{a_j}$. Protocol 1 describes how agents a_i and a_j update their state.

Protocol 1 Self-stabilizing deterministic oscillator with central control ($\mathcal{P}_{OSC}$)

```
 1: if (id_{a_i} = 0) then
 2:     if (∀ k ∈ [1, n−1], T_{a_i}[k] = 1) then
 3:         ∀ k ∈ [1, n−1], T_{a_i}[k] := 0
 4:         if (p_{a_i} = 0) then p_{a_i} := 1
 5:         else p_{a_i} := 0
 6:         end if
 7:     else
 8:         if (p_{a_i} = p_{a_j}) then
 9:             if (T_{a_i}[id_{a_j}] = 0) then T_{a_i}[id_{a_j}] := 1
10:             end if
11:         else
12:             p_{a_j} := p_{a_i}
13:             if (T_{a_i}[id_{a_j}] = 0) then T_{a_i}[id_{a_j}] := 1
14:             end if
15:         end if
16:     end if
17: end if
```

By executing Protocol $\mathcal{P}_{LE}$, eventually a unique leader is elected and each agent $a_i \in A$ has a unique value for its variable id_{a_i} (refer to [7]). Hence, id_{a_i} can be used to identify Agent a_i. Let us consider the population after the stabilization of Protocol $\mathcal{P}_{LE}$, i.e., $\exists!\ a_i \in A$ such that

$id_{a_i} = 0$ and $\forall\ a_j, a_{j'} \in A$, $a_j \neq a_{j'} \Rightarrow id_{a_j} \neq id_{a_{j'}}$. Let us refer to the elected leader by a_i. Recall that the array T is only used by a_i. Each entry $k \in \{1, \ldots, n-1\}$ of Array T_{a_i} corresponds to the entry of Agent a_j such that $id_{a_j} = k$. Every time a_i interacts with another agent, say a_j, Agent a_j updates its variable p_{a_j} to be in the same phase as the leader (refer to Line 12 in Protocol 1) and the leader updates the entry of a_j to 1 to indicate that it has already interacted with a_j (Lines 9 and 13). When all the entries of T_{a_i} are set to 1, the leader toggles its phase and re-initializes its array (Lines 3-5). Since the initial configuration is arbitrary, some of the entries in the leader's array might be equal to 1 even if the leader did not interact with the corresponding agents. However, since the leader's array is eventually re-initialized, we are sure that after the first re-initialization of T, if an entry of T is equal to 1, then the leader has indeed interacted with the corresponding agent and thus, all agents update their phase to be in the same phase as the leader. Let S be the set of state such that $p = 1$ (Phase 1). Hence, starting from any arbitrary configuration $C_0 \in \mathcal{C}$, every execution $E = (C_0, r_0, C_1, r_1, \ldots)$ of $\mathcal{P}_{LE} \circ \mathcal{P}_{OSC}$ reaches a configuration $C \in \mathcal{C}$ such that (C, S, n, ι_p) is a deterministic oscillator with $\iota_p = O(n)$ active interactions.

Discussion (1). The number of states per agent can be reduced to $M_{LE} + O(n)$ by using the same idea as in Protocol 1, but instead of using an array, the leader uses a counter that we denote by N_{ext} ($N_{ext} \in \{1, \ldots, n-1\}$). The counter is used to indicate the next agent the leader needs to interact with in order to update its state, i.e., the agents update their state in a given order so that the leader is sure to have interacted with everyone. While interacting with the leader, the agents update their phase to be in the same phase as the leader.

Protocol 2 Self-stabilizing deterministic oscillator with central control (second approach)

(C(Leader), C(¬ Leader)) → δ(C(Leader), C(¬ Leader))	
1. $(0,0,i),(i,0) \to (0,0,i+1),(i,1)$	if $i < n-1$
2. $(0,0,i),(i,1) \to (0,0,i+1),(i,1)$	if $i < n-1$
3. $(0,0,n-1),? \to (0,1,1),\ ?$	
4. $(0,1,i),(i,0) \to (0,1,i+1),\ (i,0)$	if $i < n-1$
5. $(0,1,i),(i,1) \to (0,1,i+1),(i,0)$	if $i < n-1$
6. $(0,1,n-1),? \to (0,0,1),?$	

The formal description of the solution is given in Protocol 2. Character '?' indicates any state of a non leader agent. If '?' is used then, the corresponding non-leader agent does not update its state in the interaction. The state of a leader agent a_i consists of a triplet of variables $(id_{a_i}, p_{a_i}, N_{ext})$ where id_{a_i} and p_{a_i} have the same role as in Protocol 1. The

state of a non leader agent a_j is only represented by the couple (id_{a_j}, p_{a_j}). Protocol 1 was introduced to get rid of the state update order induced while using the counter in Protocol 2. We state the following result:

Theorem 1. *Under the global fair scheduler, if there exists a population protocol that solves the SS-LE problem using M_{LE} states then, there exists a population protocol that solves the SS-OSC problem using $M_{LE} + O(n)$ states.*

Discussion (2). From Discussion (1) we know that the number of states per agent can be reduced to $M_{LE} + O(n)$ to create oscillations with amplitude $\iota_a = n$. In fact, the result can be generalized to $M_{LE} + O(\iota_a)$ states where ι_a is the desired amplitude of the oscillator. By using the same strategy as in Discussion (1), it is sufficient to set the maximum value of N_{ext} to $\iota_a - 1$. In addition, when the leader interacts with a non leader agent a_j such that $id_{a_j} > N_{ext}$, Agent a_j updates its phase p_{a_j} to 0. Since the scheduler is globally fair, $\forall\ a_j \in A$ such that $id_{a_j} > N_{ext}$, a_j eventually interacts with the leader and hence $p_{a_j} = 0$. Thus, only $(\iota_a - 1)$ agents toggle their phase with the leader.

(2) SS-OSC $\Rightarrow$ SS-LE. We show that if the deterministic SS-OSC problem is solvable using M_{OSC} states, then it is also possible to solve the deterministic SS-LE problem using $M_{OSC} + O(1)$ states. To show this result, we build our self-stabilizing SS-LE protocol $\mathcal{P}'_{LE}$ on the top of the SS-OSC protocol $\mathcal{P}'_{OSC}$. By executing Protocol $\mathcal{P}'_{OSC}$, the system eventually exhibits an oscillatory behavior with respect to a given set of state S. Let us consider the population after the stabilization of $\mathcal{P}'_{OSC}$. We first show some important properties of a population that exhibits an oscillatory behavior. We assume that $\iota_a = n$. Given a configuration $C \in \mathcal{C}$, let $N_{C(S)}$ be the set of agents such that $\forall\ a_i \in A$, $a_i \in N_{C(S)}$ if $C(a_i) \in S$. The number of agents part of $N_{C(S)}$ is denoted by $|N_{C(S)}|$. By C^+, we denote the set of configurations that can appear during the increasing phase of any oscillation before reaching the amplitude, that is, $\forall\ C \in C^+$, $|N_{C(S)}| < n$. By C^*, we refer to the set of configurations such that $\forall\ C \in C^*$, $|N_{C(S)}| = n$ (configurations in which all the agents have their states part of S, i.e., the amplitude is reached). The first step is to show that there is a non-empty subset of states that can only appear when the amplitude of the oscillation is reached. More precisely, in any configuration $C \in C^+$, the transition $\delta(C(a_i), C(a_j)) = (C'(a_i), C'(a_j))$ such that $|N_{C(S)}| > |N_{C'(S)}|$ is never enabled when the system is stabilized. Let Q' be the set of states that enable such a transition then, $\forall\ C \in C^+$, $\forall\ a_i \in A$, $C'(a_i) \notin Q'$ and $\forall\ C' \in C^*$, $\exists\ a_i \in A$, $C'(a_i) \in Q'$ (States in Q' indicates that the next phase of the oscillation can be initiated). Next, we define a subset of special configurations that we denote by $C_{sp} \subset C^+$. A configuration $C \in C_{sp}$ satisfies the two following conditions:

(1) $\exists!\ a_j \in A$ such that $C(a_j) \notin S$ and (2) $\forall\ a_i \in A$, $C(a_i) \notin Q'$. Observe that Condition (1) implies that $\forall\ a_i \in A \setminus \{a_j\}$, $C(a_i) \in S$. We show that a configuration $C \in C_{sp}$ is eventually reached and $\exists\ a_i, a_j \in A$ such that $\delta(C_{sp}(a_i), C_{sp}(a_j)) = (C'(a_i), C'(a_j))$ with $C(a_j) \notin S$ and $C'(a_j) \in S$ and either $(C'(a_i) \in Q')$ or $(C'(a_j) \in Q')$. That is, the amplitude is reached and at least one of the two interacting agents has a state part of Q'. We refer to such an interaction by r_{sp}. Finally, we prove that from a configuration $C \in \mathcal{C}^*$, if $\exists\ a_i \in A$ such that $C(a_i) \notin Q'$ and $\exists a_j \in A$ such that $\delta(C(a_i), C(a_j)) = (C'(a_i), C'(a_j))$ with $C'(a_i) \in Q'$ then $C(a_j) \in Q'$, that is, when the amplitude is reached, a given agent can change its state to a state in Q' only if it interacts with an agent already in a state part of Q'.

Protocol. In order to elect a leader starting from an arbitrary configuration $C_0 \in C$ using the SS-OSC population protocol $\mathcal{P}'_{OSC}$, we add to the state of each agent one bit of memory to indicate whether the agent is a leader or not ($l \in \{0, 1\}$). When r_{sp} is executed, if C' is the resulting configuration, then $\exists\ a_i \in A$ such that $C'(a_i) \in Q'$ (recall that r_{sp}:$\delta(C_{sp}(a_i), C_{sp}(a_j)) = (C'(a_i), C'(a_j))$ such that (i) $C(a_j) \notin S$. (ii) $C'(a_j) \in S$. (iii) $((C'(a_i) \in Q') \vee (C'(a_j)) \in Q'))$. Assume that after the execution of r_{sp}, $\exists!\ a_i \in A$ such that $C'(a_i) \in Q'$ (let us refer to this agent by a_{sp}). The idea of the protocol is as follows: when r_{sp} is executed, Agent a_{sp} becomes a leader. In addition, when a given agent a_i interacts with a leader then, a_i does not update its state (keep the same state). Observe that since we assume an arbitrary initial configuration, such a transition can be executed even if the population is not yet stabilized with respect to $\mathcal{P}'_{OSC}$. To be sure to create only one leader, if in a given configuration $C \in \mathcal{C}$, a_i is a leader then a_i becomes a non-leader in the next interaction if $C(a_i) \notin Q'$ or $C(a_i) \notin S$. In the same manner, a_i becomes a non leader if it interacts either with another leader or with an agent a_j such that $C(a_j) \notin S$. Observe that if $\exists\ a_i \in A$ such that a_i is a leader, then a_i can only be enabled to become a non-leader, We show that:

Theorem 2. *Under the global fair scheduler, if there exists a population protocol $\mathcal{P}'_{OSC}$ that solves the SS-OSC problem with amplitude n using M_{OSC} states, then the SS-LE problem is also possible to solve using $M_{OSC} + O(1)$ states.*

Remark. Theorem 2 can be generalized to any amplitude $\iota_a \leq n$. Indeed, the main idea of the solution is to make any leader becomes a non leader infinitely often during the stabilization time to ensure the convergence of Protocol $\mathcal{P}'_{OSC}$. Once the population converges to an oscillatory behavior, the properties of the oscillatory behavior ensure that only one leader is created. The leader then prevents the second phase of the oscillation to be initiated thus, no more leaders are created.

Recall that it has been proved in [7] that the SS-LE problem is not solvable when the number of states is less than the size of the population n and hence impossible to solve in the case where n is arbitrary. Using Theorems 1 and 2, we can deduce the following corollary:

Corollary 1. *There exists no deterministic self-stabilizing oscillator if the number of states by agent is less than n, or if the size of the population is arbitrary.*

4 Stochastic scheduler

Aiming at the reduction of the space complexity, we investigate in this section the SS-OSC problem under a uniform random scheduler, i.e., the pair of agents that are selected for the interaction are chosen at random, independently and uniformly from the set of all the agents of the population. Let us first define the notion of the self-stabilizing *stochastic oscillator*.

Definition 5. ***(Self-stabilizing stochastic oscillator)*** *A (sufficient large) population of agents executing a deterministic protocol $\mathcal{P}$, under a uniform random scheduler, is a (C, S, ι_a, ι_p)-oscillator, if starting from any arbitrary configuration $C_0 \in \mathcal{C}$, any execution $E = (C_0, r_0, C_1, r_1, \dots)$ of $\mathcal{P}$, reaches a configuration $C \in \mathcal{C}$ such that (C, S, ι_a, ι_p) exhibits an oscillatory behavior for the set of states S with an expected average amplitude ι_a and an expected average period ι_p.*

We present in this section, three deterministic protocols. Each of them assumes an arbitrary initial configuration and also the presence of a leader, that is, the agents need first to elect a leader in order to achieve the oscillatory behavior. Recall that, without a leader detector oracle, the SS-LE problem is impossible to solve with less than n states. That is, $\Omega(n)$ states are necessary to achieve the election [7]. We aim in the following at the reduction of the extra-cost used to create the oscillatory behavior.

In the sequel, the state of the leader is represented by the couple (L_p, c) where L_p indicates that the agent is a leader in Phase p ($p \in \{0, 1\}$) and $c \in \{0, \dots k\}$ represents the current value of the leader's counter where $k \in \mathbb{N}$, is the maximum value that the counter can reach. The state of a non-leader agent consists of only one variable p such that $p \in \{0, 1\}$. Variable p indicates which phase of the oscillation the agent is part of. An agent is said to be *a follower* (respectively *a non-follower*) if it is not a leader and if the value of its variable p is the same as (respectively different from) the leader's.

First approach. The idea of the solution is as follows: at each time the leader interacts with a follower agent, the leader increments its counter.

If it interacts with a non-follower agent, the leader re-initializes its counter and the non-follower agent part of the interaction updates its phase to become a follower. When the leader's counter reaches its maximum value, it toggles its phase and re-initializes its counter. The formal description of the protocol is given in Protocol 3.

Protocol 3 Self-stabilizing stochastic oscillator with central control

(C(Leader), C(¬ Leader)) → δ(C(Leader), C(¬ Leader))	
1. (L_0,i), 0 → (L_0,i+1), 0	$i < k$
2. (L_0,i),1 → (L_0,0), 0	$i < k$
3. (L_0,k), 0 → (L_1,0), 0	
4. (L_1,i), 0 → (L_1,0), 1	$i < k$
5. (L_1,i), 1 → (L_1,i+1), 1	$i < k$
6. (L_1,k), 1 → (L_0,0), 1	

We show in the following that for any counter size $k \gg \log n$, the remaining non-follower agents at the end of a phase is $O((n/k)\ \log n)$ with high probability (provided that n is sufficiently large).

Suppose without loss of generality that the leader's phase is 0 (L_0), and there are initially $B_0 \leq n$ non-follower agents. Let $X(k) = X(k, B_0)$ be the number of non-follower agents at the end of the phase (when the leader toggles its phase from 0 to 1). Let $P(J)$ be the probability the switch of phase by the leader occurs when J non-follower agents remain. Then:

$$P(J) = \prod_{j=B_0}^{J+1}(1-(1-j/n)^k)(1-J/n)^k$$

Note that only interactions with the leader matter in this calculation. Let E be the expected value of $X(k)$ then: $E = \sum_{J=0}^{B_0} JP(J)$.

Let $\omega = \omega(n)$ be some slowly growing function of n and assume $B_0 = n$ (worst case), and $J \geq \omega n/k$. We have: $P(J) \leq (1-J/n)^k \leq e^{-kJ/n} \leq e^{-\omega}$. Let $\omega = 2\log n$, the contribution to E from $\omega n/k \leq J \leq n$ is then:

$$\sum_{J=\omega n/k}^{n} JP(J) \leq n^2 e^{-2\log n} = 1$$

Similarly, if $\omega = 3\log n$ then, with high probability the number of non-follower agents is never bigger than $(3\log n)\ n/k$ when the leader toggles its phase. Thus for $\omega = (2\log n)\ n/k$, we have:

$$E \leq 1 + [(2\log n)\ n/k] \sum_{J \leq (2\log n)\ n/k} P(J) = O((\log n)\ n/k)$$

As for the expected period, for any k a phase completes in $O(kn)$ interactions with the leader. So the length of the phase is $O(n^2 k)$ with high probability.

Second approach. To reduce even more the space complexity, we propose in the sequel, two population protocols that solve the SS-OSC problem and that use, in addition to the leader, another agent that we call *marked agent* and that we denote by M. Recall that the SS-LE protocol proposed in [7] not only elects a leader, but also provides a kind of identification, i.e., each agent a_i has unique state $C(i)$ such that $C(i) \in \{0, 1, \ldots, n-1\}$. Hence, we can assume that the leader is the agent $a_i \in A$ such that $C(a_i) = 0$ and the marked agent is the agent $a_j \in A$ such that $C(a_j) = 1$. Thus, no other run of the leader election protocol is performed.

First solution. The idea of the first solution is as follows: at each time the leader interacts with the marked agent, the leader's counter is incremented. On another hand, when the leader interacts with a non-follower agent, the non-follower agent updates its state to become a follower. When the leader's counter value reaches its maximum, i.e., after $(k+1)$ interactions with the marked agent, the leader toggles its phase and re-initializes its counter value. The formal description of the first solution is given in Protocol 4.

Protocol 4 Self-stabilizing stochastic oscillator with central control using the marked agent trick without re-initialization

(C(Leader), C(¬ Leader)) → δ(C(Leader), C(¬ Leader))	
1. (L_0,i), 0 → (L_0,i), 0	if $i < k$
2. (L_0,i),1 → (L_0,i), 0	if $i < k$
3. (L_0,i), M → (L_0,i+1), M	if $i < k$
4. (L_0,k), M → (L_1,0), M	
5. (L_1,i), 0 → (L_1,i), 1	if $i < k$
6. (L_1,i), 1 → (L_1,i), 1	if $i < k$
7. (L_1,i), M → (L_1,i+1), M	if $i < k$
8. (L_1,k), M → (L_0,0), M	

Let $\mathcal{X}$ (respectively $\mathcal{X}(i, i+1)$) be the number of interactions to reach the maximum value of the leader's counter (respectively to the number of interactions in order for the leader's counter to be incremented from i to $i+1$), we have: $\mathcal{X}=\sum_{i=0}^{k} \mathcal{X}(i, i+1)$. Recall that the leader's counter is only incremented when the leader interacts with the marked agent, that is, $\mathcal{X}(i, i+1)$ has a geometric distribution of parameter p: $P(\mathcal{X}(i, i+1) = m) = (1-p)^{m-1}p$, where p is the probability to get an interaction between the leader and the marked agent. Note that $p = \frac{2}{n(n-1)}$. The expected number of interactions to increment the leader's counter from i to $i+1$ is

$$E[\mathcal{X}(i, i+1)] = \frac{1}{p} = \frac{n(n-1)}{2}.$$

Using the linearity of the expectations, we obtain $E[\mathcal{X}] = (k+1)\ \frac{n(n-1)}{2}$. So after $E[\mathcal{X}]$ average interactions, the leader updates its phase to initiate the next phase of the oscillation.

Assume without loss of generality that the leader's phase is equal to 0. Let us now determine the expected amplitude ι_a which represents the expected number of non-follower agents that become followers before the leader's counter reaches its maximum value. Let $A_b(t)$ refers to the number of non-follower agents at time t. The expected number of non-follower agents at time $t+1$ ($A_b(t+1)$) is given by: $A_b(t+1) = A_b(t) - \frac{2.A_b(t)}{n(n-1)}$. That is, at time $(t+1)$, the number of non-follower agents either remains the same or decreases when there is an interaction between the leader and a non-follower agent. Approximately we have:

$$\frac{dA_b(t)}{dt} = -\frac{2}{n(n-1)}A_b(t), \text{ hence, } A_b(t) = A_b(0)\ e^{-(\frac{2t}{n(n-1)})}$$

Recall that we know the expected number of interactions before reaching the amplitude. By replacing t by $E[\mathcal{X}]$, we obtain the expected number of non-follower agents when the amplitude is reached. That is:

$$A_a(E[\mathcal{X}]) = n - A_b(0)\ e^{-(k+1)}$$

We observe that if $k = log(n)$, $A_a(E[\mathcal{X}]) = n$.

Second solution. We present in the sequel, a variation of Protocol 4 aiming at solving the SS-OSC problem with $k \in O(1)$ (recall that k is the maximum value of the leader's counter). The idea of the solution is similar to the one used in Protocol 4 except that, when the leader interacts with either a follower or a non-follower agent, it re-initializes its counter, that is, the leader, needs to interact $(k+1)$ consecutive times with the marked agent in order to toggle its phase. The system can be represented by a Markov chain as shown in Figure 1, where p is the probability to get an interaction between the leader and the marked agent, q is the probability to get an interaction between the leader and either a follower or a non-follower and $M = 1-(p+q)$ (interactions that do not include the leader).

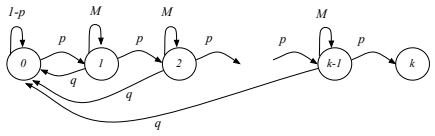

Fig. 1. Corresponding Markov Chain.

The average number of interactions I_k, for the leader to toggle its phase, can be computed using the first step analysis. We obtain:

$$I_k = \frac{n(n-1)}{2}(\frac{(n-1)^k-1}{n-2}) \quad (k > 1).$$

Assume that $(i-1)$ non-follower agents have already updated their phase to become followers. Let compute the probability, P_{Next-i}, that a new non-follower agent changes its phase to become a follower before the switch of phase is performed by the leader i.e., before the $(k+1)$ consecutive interactions between the leader and the marked agent. We have:

$$P_{Next-i} = P_{b_i} \cdot (\sum_{j=0}^{k-1}(P_M^j)) \cdot \sum_{g\geq 0}(P_{w_i} \cdot \sum_{j=0}^{k-1}(P_M^j))^g$$

Probability	Probability of an interaction	Value
P_M	(Leader, Marked agent)	$\frac{1}{(n-1)}$
P_{b_i}	(Leader, non-follower agent)	$\frac{B-i+1}{(n-1)}$
P_{w_i}	(Leader, follower agent)	$\frac{n-B+i-3}{(n-1)}$

Let $Z = \sum_{j=0}^{k-1} P_M^j$ and $Z' = \sum_{g\geq 0}(Z \cdot P_{w_i})^g$. Both Z and Z' are geometric series of common ration (P_M) and $(Z \cdot P_{w_i})$ respectively. Hence:

$$Z = (1 - P_M^k)/(1 - P_M) \text{ and } Z' = (1 - (Z \cdot P_{w_i})^m)/(1 - Z \cdot P_{w_i})$$

For a large population of agents, $P_M^k \simeq 0$ even if $k \in O(1)$. In the same manner, since, $(Z \cdot P_{w_i}) < 0$ and $m \to \infty$, $(Z \cdot P_{w_i})^m = 0$. Thus, $P_{Next-i} \simeq 1$. That is, all the non-follower agents become followers before reaching the maximum value of the leader's counter. Hence, $\iota_a \simeq n$

5 Conclusion

In this paper, we have considered the PPs model and have addressed the problem of autonomously generating oscillatory executions. We have considered the problem using deterministic protocols and have shown that, under a deterministic scheduler, $\Omega(n)$ states are necessary to solve the SS-OSC problem. This result emphasizes somehow the impact and the importance of randomization in biological systems and chemical reactions in creating self-oscillations. We have then proposed some protocols that solve the problem assuming a probabilistic scheduler. This is a preliminary work as several open questions arise: (i) All the proposed solutions in this paper, assume a central control, that is, the agents first need to elect a leader in order to create the desired oscillatory behavior. This is really costly especially for these kinds of systems, since the number of agents is usually huge. Thus, the problem of designing protocols that solve the SS-OSC problem in a decentralized way remains open. The main challenge is to achieve the self-stabilizing oscillatory behavior using a number of states that is independent from any global parameter of the system. Observe that when decentralized solutions are considered, it is impossible to achieve the oscillatory behavior as defined in this paper as, during the increasing phase (respectively, the decreasing phase) of an oscillation, the

number of agents whose state is in the set S can decrease (respectively, increase) before (respectively, after) reaching the amplitude. However, there is a scaling effect that ensures that if we consider the global behavior of the population, by zooming out and ignoring the small fluctuations due to the agents that may toggle their phase before (respectively, after) reaching the amplitude of the oscillation, the population could display an oscillatory behavior. (*ii*) We have recently addressed the SS-OSC problem in a slightly different setting in which we assume that the population is synchronous i.e., each agent is part of an interaction at each instant t. We were able to implement a self-synchronized clock and use it to design primitive oscillators. The number of states used to solve the problem does not depend on the size of the population however, it does depends on the period of the oscillator. Hence, it would be also interesting to investigate the impact of the degree of synchrony on the SS-OSC problem. Finally, (*iii*) it would be challenging to simulate, as for the *Fourier Transform*, in a self-stabilizing way, any periodic behavior of a given population using a finite number of deterministic oscillators. We were able to do so in a recent investigation assuming synchronous populations. Extending the investigation taking in account different level of synchrony seems to be interesting direction to investigate.

References

1. Dana Angluin, James Aspnes, Melody Chan, Michael J. Fischer, Hong Jiang, and René Peralta. Stably computable properties of network graphs. In *DCOSS*, volume 3560, pages 63–74, 2005.
2. Dana Angluin, James Aspnes, Zoë Diamadi, Michael J. Fischer, and René Peralta. Computation in networks of passively mobile finite-state sensors. In *PODC*, pages 290–299, 2004.
3. Dana Angluin, James Aspnes, and David Eisenstat. Fast computation by population protocols with a leader. *Distributed Computing*, 21(3):183–199, 2008.
4. Dana Angluin, James Aspnes, Michael J. Fischer, and Hong Jiang. Self-stabilizing population protocols. *TAAS*, 3(4), 2008.
5. Joffroy Beauquier and Janna Burman. Self-stabilizing synchronization in mobile sensor networks with covering. In *Distributed Computing in Sensor Systems (DCOSS)*, volume 6131, pages 362–378, 2010.
6. Joffroy Beauquier and Janna Burman. Self-stabilizing mutual exclusion and group mutual exclusion for population protocols with covering. In *OPODIS*, volume 7109, pages 235–250, 2011.
7. 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 Comput. Syst.*, 50(3):433–445, 2012.
8. Keigo Kinpara, Tomoko Izumi, Taisuke Izumi, and Koichi Wada. Improving space complexity of self-stabilizing counting on mobile sensor networks. In *OPODIS*, volume 6490, pages 504–515, 2010.
9. Satoshi Murata, Akihiko Konagaya, Satoshi Kobayashi, Hirohide Saito, and Masami Hagiya. Molecular robotics: A new paradigm for artifacts. *New Generation Computing*, 31(1):27–45, 2013.

A Proof of correctness of Theorem 1

Let us first show that by executing $\mathcal{P}_{LE} \circ \mathcal{P}_{OSC}$, one leader is eventually elected and each agent $a_i \in A$ has a unique value of id_{a_i} after $\Theta(n^2)$ active interactions (recall that for any agent $a_i \in A$, $id_{a_i} \in \{0, 1, \ldots, n-1\}$).

Theorem 3. *Starting from an arbitrary initial configuration $C_0 \in C$, for any execution $E = (C_0, r_0, C_1, r_1 \ldots)$ of Protocol $\mathcal{P}_{LE} \circ \mathcal{P}_{OSC}$, a configuration $C \in \mathcal{C}$ in which $\exists!\ a_i \in A$, $id_{a_i} = 0$ and $\forall\ a_j, a_{j'} \in A$, $id_{a_j} \neq id_{a_{j'}}$ is eventually reached.*

Proof. First recall that during an interaction between two agents, the two agents execute the actions of both Protocol $\mathcal{P}_{LE}$ and $\mathcal{P}_{OSC}$. Observe also that for any agents $a_i \in A$, Protocol $\mathcal{P}_{OSC}$ can only read Variable id_{a_i}, that is, id_{a_i} is never modified by $\mathcal{P}_{OSC}$. On another hand, Protocol $\mathcal{P}_{LE}$ does not use any variable introduced by Protocol $\mathcal{P}_{OSC}$, that is, Protocol $\mathcal{P}_{OSC}$ has not effect on the convergence and stabilization of Protocol $\mathcal{P}_{LE}$. It has been shown in [7] that $\forall\ C_0 \in C$, for any execution $E = (C_0, r_0, C_1, r_1 \ldots)$ of Protocol $\mathcal{P}_{LE}$, a configuration $C \in \mathcal{C}$ in which $\exists! a_i \in A$, $id_{a_i} = 0$ and $\forall\ a_j, a_{j'} \in A$, $id_{a_j} \neq id_{a_{j'}}$ is eventually reached. Thus, the lemma holds.

□

Let us now consider the population after the convergence of Protocol $\mathcal{P}_{LE}$ i.e., $\forall\ a_i, a_j \in A$, $id_{a_i} \neq id_{a_j}$ and $\exists!\ a_i \in A$ such that $id_{a_i} = 0$ (the leader). From [7], we know that when $\mathcal{P}_{LE}$ converges, the value of id_{a_i} of each agent a_i never changes. We prove in the sequel that in $O(n)$ active interactions, the leader is sure to have interacted with all the other agents in the population. A configuration in which all the agents have the same phase is then reached.

Lemma 1. *Starting from an arbitrary initial configuration $C_0 \in \mathcal{C}$, for any execution $E = (C_0, r_0, C_1, r_1, \ldots)$ of Protocol $\mathcal{P}_{LE} \circ \mathcal{P}_{OSC}$, a configuration $C \in \mathcal{C}$ in which $\forall\ i \in \{1, \ldots, n-1\}$, $T[i] = 0$ is reached in $O(n)$ active interactions.*

Proof. According to Theorem 3, the system elects a leader, that is, there exists a single agent a_i such that $id_{a_i} = 0$ and all the agents that are not leaders have a unique value of variable id. We show that in $O(n)$ active interactions, $\forall\ j \in \{1, \ldots, n-1\}$, $T[j] = 0$. Where T is the leader's array. Assume by contradiction that for any execution of our solution, $\forall\ t > t_0$ (t_0 refers to the system in the initial configuration), $\exists\ j \in \{1, \ldots, n-1\}$ such that $T[j] \neq 0$. Let $\mathcal{N}_{E_0}$ be the number of entries j in the leader's array such that $T[j] = 0$.

After the stabilization of Protocol 1 ([7]), all active interactions include the leader (refer to Protocol 1). Assume, without loss of generality, that the phase of the leader is equal to 1. Let $\mathcal{N}(A_0)$ be the number of agents that are in phase 0. Note that if $\mathcal{N}_{E_0} = n - 1$ then the lemma holds. Otherwise, the cases below are possible (let $a_i, a_j \in A$ be the two agents that are interacting such that a_i is the leader):

1. $p_{a_i} \neq p_{a_j}$ and $T[id_{a_j}] = 0$. In this case, the leader updates the entry related to the agent with whom it has interacted to 1 (refer to Line 13 in Protocol 1). Thus, $\mathcal{N}_{E_0}$ has decreased by 1. In addition, the phase of the non-leader agent is updated to be equal to the leader's phase (refer to Line 12 in Protocol 1), that is $\mathcal{N}_{A_0}$ has also decreased by 1.
2. $p_{a_i} \neq p_{a_j}$ and $T[id_{a_j}] = 1$. In this case the corresponding entry in the leader's array is already equal to 1. Such an interaction does not modify it. That is, $\mathcal{N}_{E_0}$ does not change. However, the phase of the non-leader agent is updated to be equal to the leader's phase (refer to Line 12 in Protocol 1). Thus, $\mathcal{N}_{A_0}$ decreases by 1.
3. $p_{a_i} = p_{a_j}$ and $T[id_{a_j}] = 0$. In this case, the leader only updates the entry related to the agent with whom it has interacted to 1 (refer to Line 9 in Protocol 1). Thus, $\mathcal{N}_{E_0}$ has decreased by 1. Note that the value of $\mathcal{N}_{A_0}$ has not changed.

Case 2 cannot happen infinitely often since the number of agents is finite and $\mathcal{N}_{A_0}$ is never increased when the leader's phase is equal to 1. Thus, after $O(n)$ active interactions, Case 2 never occurs. On the other hand, in Case 1 and Case 3, $\mathcal{N}_{E_0}$ decreases by 1 at each time. Since the number of entries in T is equal n and since $\mathcal{N}_{E_0}$ never increases (Recall that the phase of the leader never change unless all the entries in T are equal to 1), in $O(n)$ active interactions, $n - \mathcal{N}_{E_0}$ reaches 0. Action on Line 3 in Protocol 1 becomes the only active action enabled on the leader a_i. When the action is executed, the array of the leader verifies: $\forall\ j \in \{1, \ldots, n-1\}$, $T[j] = 0$. Contradiction.

We can conclude that starting from an arbitrary initial configuration $C_0 \in \mathcal{C}$, a configuration in which $\forall\ j \in \{1, \ldots, n-1\}$, $T[j] = 0$ is reached in $O(n)$ active interactions. Hence, the lemma holds.

□

Lemma 2. *Starting from an arbitrary initial configuration $C_0 \in C$, For any execution $E = (C_0, r_0, \ldots)$ of Protocol $\mathcal{P}_{LE} \circ \mathcal{P}_{OSC}$, a configuration in which all agents except the leader are in the same phase and $\forall\ i \in \{1, \ldots, n-1\}$, $T[i] = 0$, is reached in $O(n)$ active interactions.*

Proof. According to Lemma 1, after the election of a leader by Protocol $\mathcal{P}_{LE}$, a configuration in which $\forall\ i \in \{1, \ldots, n-1\}$, $T[i] = 0$ is

reached in $O(n)$ active interactions. Let us consider the system at that time. Without loss of generality assume that the leader's phase is equal to 1. Observe that at each time Action on Line 12 of Protocol 1 is executed, Action on Line 13 of Protocol 1 is also executed, the non leader agent updates its phase to the leader's phase (in the case where the agent has a different phase from the leader). Let $\mathcal{N}_{E_1}$ be the number of entries i in the leader's array such that $T[i] = 1$ and let $\mathcal{N}_{\neg L}$ be the number of agents that are in a different phase from the leader. Note that $\mathcal{N}_{E_1} = 0$ and $\mathcal{N}_{\neg L} \leq n-1$. Either Lines 12,13 or Line 9 in Protocol 1 are enabled. Once one of these set of actions is executed, $\mathcal{N}_{E_1}$ increases by 1 and, $\mathcal{N}_{\neg L}$ decreases by 1 in the case where the agent part of the interaction has a different phase from the leader. That is, after $O(n)$ active interactions, a configuration in which $\mathcal{N}_{E_1} = n-1$ and $\mathcal{N}_{\neg L} = 0$ is reached. Action on Line 3 is the only one enabled on the leader. When the leader executes the action, it updates its phase to 0 and re-initializes all the entries of its array. That is, $\forall\ i \in \{1, \ldots, n-1\}$, $T[i] = 0$. Observe that, now $\mathcal{N}_{\neg L} = n-1$. Hence the lemma holds. □

Theorem 4. *Given a population of n agents, Protocol $\mathcal{P}_{LE} \circ \mathcal{P}_{LE}$ is a self-stabilizing oscillator of amplitude n and period $O(n)$ active interactions.*

Proof.
Clear from Theorem 3 and Lemmas 2.

□

B SS-OSC ⇒ SS-LE

We show in the following that when the SS-OSC protocol stabilizes, some properties are verified. We first show that there is a non empty set of states $Q' \subset Q$ that only appear when the amplitude of a given oscillation is reached. Recall that since we assume that there exists a population protocol $\mathcal{P}'_{OSC}$ that solves the SS-OSC problem, we know that $\forall\ C_0 \in \mathcal{C}$, a configuration $C \in \mathcal{C}$ such that (C, S, ι_a) is an oscillator with $\iota_a = n$ is eventually reached. In the sequel, all the lemmas are stated after the stabilization of $\mathcal{P}'_{OSC}$ i.e., the population exhibits an oscillatory behavior with amplitude $\iota_a = n$.

Lemma 3. *$\exists\ Q' \subset Q$ such that: $\forall\ C \in C^+$, $\forall\ a_i \in A$, $C(a_i) \notin Q'$ and $\forall\ C' \in C^*$, $\exists\ a_j \in A$, $C'(a_j) \in Q'$.*

Proof. Recall that we assume that the behavior of the population is stabilized i.e., it exhibits an oscillatory behavior of amplitude n. Hence, the system eventually reaches a configuration C such that $C \in C^*$

($\forall\ a_i \in A,\ C(a_i) \in S$). From configuration C, since the system is an oscillator, $\exists\ a_i, a_j \in A$ such that $\delta(C(a_i), C(a_j)) = (C'(a_i), C'(a_j))$ and ($C'(a_i) \notin S$ or $C'(a_j) \notin S$) (the next phase of the oscillator has been initiated). Hence $|N_{C(S)}| > |N_{C'(S)}|$. Let us denote by Q', the set of states that enable such a rule. Now assume that given a configuration $C \in C^+$, $\exists\ a_i \in A$ such that $C(a_i) \in Q'$. This means that there exists a transition $\delta(C(a_i), C(a_j)) = (C'(a_i), C'(a_j))$ such that $|N_{C(S)}| < |N_{C'(S)}|$. Assume that the scheduler selects a_i and a_j for such an interaction. Hence, the number of agents $N_{C(S)}$ decreases before reaching the amplitude $\iota_a = n$ i.e., the system is not yet stabilized. Contradiction.

We can deduce that $\forall\, C \in C^+$, the following transition $\delta(C(a_i), C(a_j)) = (C'(a_i), C'(a_j))$ such that $|N_{C(S)}| > |N_{C'(S)}|$ is never enabled when the system is stabilized. Let Q' be the set of states that enable such a transition. We can deduce that $\forall\ C \in C^+$, $\forall\ a_i \in A$ and $\forall\ C' \in C^*$, $\exists\ a_j \in A$, $C'(a_j) \in Q'$ and the lemma holds.

□

We show in the following that by executing Protocol $\mathcal{P}_{OSC}$, Configuration C_{sp} is eventually reached.

Lemma 4. *$\forall\ C_0 \in C$, for any execution $E = (C_0, r_0, C_1, r_1, \dots)$ of Protocol $\mathcal{P}_{OSC}$, a configuration $C \in C_{sp}$ is eventually reached.*

Proof. Recall that the scheduler activates only one pair of agents at each instant $t \geq 0$. Since $\mathcal{P}_{OSC}$ is a self-stabilizing oscillator with amplitude n, when the system stabilizes, during the increasing phase of a given oscillation, a configuration $C \in \mathcal{C}$ in which $|N_{C(S)}| = n - 1$ is eventually reached. Hence, $\exists!\ a_j \in A$ such that $C(a_j) \notin S$. Since $C \in C^+$, according to Lemma 3, $\forall\ C \in C^+$, $\forall\ a_i \in A$, $C(a_i) \notin Q'$. Thus, the lemma holds. □

Let $Q'' \in Q$ be the subset of states that an agent can have in any configuration $C \in C^+$. Note that $Q'' \cap Q' = \emptyset$ (refer to Lemma 3).

In a given configuration $C \in C_{sp}$, $\exists!\ a_i \in A$ such that, $C(a_i) \notin S$. We show that when $r_{sp} = \delta(C(a_i), C(a_j)) = (C'(a_i), C'(a_j))$ such that $C'(a_j) \in S$ (when the amplitude is reached), one of the two interacting agents (a_i or a_j) has a state in Q'.

Lemma 5. *Let $a_i \in A$ be the agent such that $C_{sp}(a_i) \notin S$. $\forall\ C \in C_{sp}$, $\exists\ a_j \in A$ such that if $\delta(C(a_i), C(a_j)) = (C'(a_i), C'(a_j))$ then $C'(a_i) \in S$ and either $C'(a_i) \in Q'$ or $C'(a_j) \in Q'$.*

Proof. The proof is by contradiction. Assume first that $\forall\ a_j \in A \setminus \{a_i\}$, if $\delta(C(a_i), C(a_j)) = (C'(a_i), C'(a_j))$ then $C'(a_i) \notin S$ or ($C'(a_i)$

$\notin Q'$ and $C'(a_j) \notin Q'$). Let assume first that $C'(a_i) \notin S$. That is, the system never reaches the amplitude n. Contradiction. Now suppose that $C'(a_i) \notin Q'$ and $C'(a_j) \notin Q'$, we know that $C'(a_i) \in S$. Now, since $C'(a_i)$ $\notin Q'$ and $C'(a_j) \notin Q'$ the transition r: $\delta(C(a), C(a)) = (C'(a), C'(b))$ such that $|N_{C(S)|} > |N_{C'(S)}|$ is never enabled. Hence, the decreasing phase of an oscillation is never initiated (the system is not an oscillator). Contradiction.

□

We show now that for a given configuration $C \in C^*$, $\forall\ a_i \in A$ such that $C(a_i) \notin Q'$, $\exists\ a_j \in A$ such that $if \delta(C(a_i), C(a_j)) = (C'(a_i), C'(a_j))$ and $C'(a_i) \in Q'$ then $C(a_j) \in Q'$.

Given a configuration $C \in \mathcal{C}$, let us refer by $N_{C_{Q'}}$ (respectively $N_{C_{Q''}}$) to the set of of agents such that $a_i \in N_{C_{Q'}}$ (respectively $a_i \in N_{C_{Q''}}$) if $C(a_i) \in Q'$ (respectively $C(a_i) \in Q'$). The number of agents in Set $N_{C_{Q'}}$ (respectively $N_{C_{Q''}}$) is denoted by $|N_{C_{Q'}}|$ (respectively $|N_{C_{Q''}}|$).

Lemma 6. *$\forall\ C \in C^*$ such that $0 < |N_{C_{Q'}}| \leq 2$, if for any $a_i, a_j \in A$, $\delta(C(a_i), C(a_j)) = (C'(a_i), C'(a_j))$ and $|N_{C'_{Q'}}| > |N_{C_{Q'}}|$ then either ($C(a_i) \in Q'$ and $C(a_j) \in Q''$) or ($C(a_i) \in Q''$ and $C(a_j) \in Q'$).*

Proof. Assume by contradiction that starting from a configuration $C \in C^*$ such that $0 < |N_{C_{Q'}}| \leq 2$, if $\delta(C(a_i), C(a_j)) = (C'(a_i), C'(a_j))$ and $|N_{C'_{Q'}}| > |N_{C_{Q'}}|$ then ($C(a_i) \in Q'$ or $C(a_j) \in Q''$) and ($C(a_i) \in Q''$ or $C(a_j) \in Q'$). (i) if $C(a_i) \in Q'$ and $C(a_j) \in Q'$ then $|N_{C'_{Q'}}| \leq |N_{C_{Q'}}|$. Contradiction (recall that $|N_{C'_{Q'}}| > |N_{C_{Q'}}|$ by assumption). (ii) if $C(a_i)$ $\in Q''$ and $C(a_j) \in Q''$ then according to lemma 3, $\delta(C(a_i), C(a_j)) =$ $(C'(a_i), C'(a_j))$ never generates a Q' state. Contradiction. From (i) and (ii) we can deduce that $C(a_j) \in Q''$) or ($C(a_i) \in Q''$ and $C(a_j) \in Q'$) and the lemma holds. □

C Proof of correctness of Theorem 2

We denote by $\mathcal{C}_{legit} \in C^*$ the set of configurations that verify the following condition: $\forall\ a_i \in A$, $\mathcal{C}_{legit}(a_i) \in S$ and $\exists!\ a_j \in A$ such that $\mathcal{C}_{legit}(a_j) \in Q'$ and a_j is a leader.

Observe that if in a given configuration C there is a non empty set of agents $A' \subseteq A$ such that $\forall\ a_i \in A'$, a_i is a leader and ($C(a_i) \notin Q'$ or $C(a_i) \notin S$), then since the scheduler is globally fair, $\forall\ a_i \in A'$, a_i is eventually activated by the scheduler. Hence, a_i becomes a non-leader agent. On another hand, in a given configuration C, an agent a_i becomes

a leader only when the agent updates its state to be part Q' i.e., if a_i is a leader in a given configuration C then $C(a_i) \in Q'$. Hence, the number of agents in A' never increases. Therefore, eventually $A' = \emptyset$ i.e., $\forall\ a_i \in A$, if a_i is a leader then $C(a_i) \in Q'$ (recall that since $Q' \subset S$ then $C(a_i) \in S$). In the sequel, we consider any configuration C in which $A' = \emptyset$. Let refer to the set of agents that are leaders in a given configuration by L_C. We show that $\exists\ C \in \mathcal{C}$ such that $|L_C| = 1$ is eventually reached and that $\forall\ C' \in \mathcal{C}$ such that $C \rightarrow C'$, $|L_{C'}| = 1$.

Observe that in any configuration $C \in \mathcal{C}$ as long as $|L_C| > 1$, there exist two agents $a_i, a_j \in A$ that are both leaders such that if $\delta(C(a_i), C(a_j)) = (C'(a_i), C'(a_j))$ then, both a_i and a'_j become non leaders in C' and $C'(a_i)$, $C'(a'_j) \in Q'$ (recall that $A' = \emptyset$). In the same manner, if there exists an agent a_i in a given configuration C such that $C(a_i) \in Q'$ and $\exists\ a_j \in C$ such that $C(a_j) \in S$ then if $\delta(C(a_i), C(a_j)) = (C'(a_i), C'(a_j))$ then $(C'(a_i) \notin S$ or $C'(a_j) \notin S)$. Finally, if in a given configuration C, $\exists\ a_i \in A$ such that a_i is a leader and $\exists\ a_j \in A$ such that $C(a_j) \notin S$ then if $\delta(C(a_i), C(a_j)) = (C'(a_i), C'(a_j))$, then a_i becomes a non leader in C'.

We can deduce that as long as the configuration is different from $\mathcal{C}_{legit}$, each agent $a_i \in A$ that is leader is persistently enabled to become a non leader. Since the scheduler is globally fair, a_i is eventually activated for an interaction. Hence, as long as the configuration is different from $\mathcal{C}_{legit}$, each agent is persistently non leader. Thus, Protocol $\mathcal{P}_{OSC} \circ \mathcal{P}_{LE}$ behaves as $\mathcal{P}_{OSC}$ as long as the configuration is different from $\mathcal{C}_{legit}$ since $\mathcal{P}_{OSC} \circ \mathcal{P}_{LE}$ just halts the execution of some agents for a finite time. However, since each agent is persistently non leader and since the scheduler is globally fair, Protocol $\mathcal{P}_{OSC} \circ \mathcal{P}_{LE}$ eventually stabilizes. Thus, according to Lemmas 4, 5 and 6, there exists a configuration C in which there is exactly one agent $a_i \in A$ such that $C(a_i) \in Q'$ is reached. Thus, one leader is elected and the theorem holds.

D Some simulation results of Protocol 3

We present in Figure 2 some simulation results of a population of 1000 agents executing Protocol 3 for different values of k.

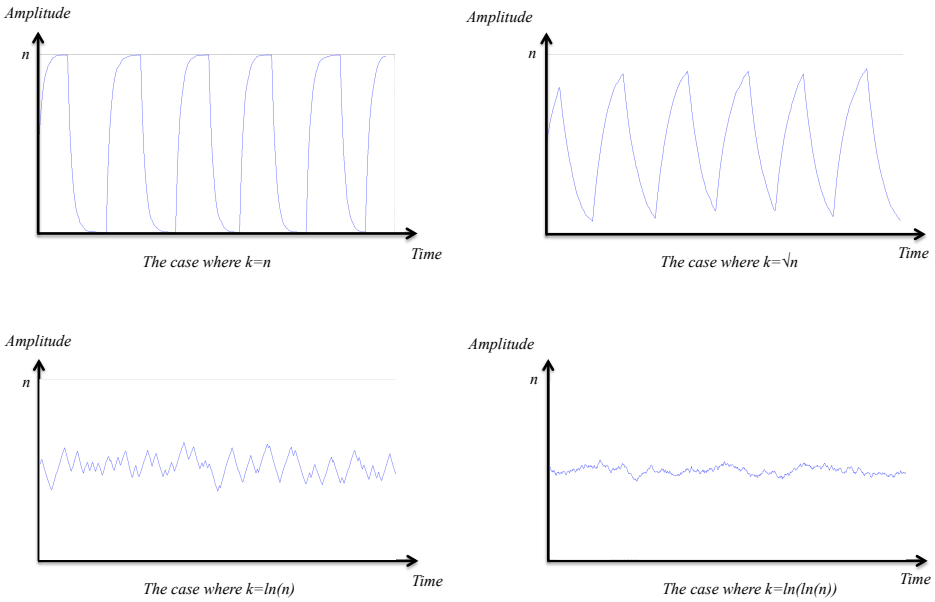

Fig. 2. Oscillatory behaviors by a population of 1000 agents executing Protocol 3 for different values of k

As we can observe that the population indeed exhibits an oscillatory behavior for the values of $k \gg log(n)$, which validates the theoretical results obtained.

E First step analysis to compute the average number of interactions of the second solution of the second approach (stochastic scheduler)

The average number of interactions to reach node i in the Markov chain presented in Figure 1, starting from node 0, is denoted by A_i. To compute the average number of interactions to reach node k, starting from node 0, A_k can be given using a recursion equation as follows: we know that after A_{k-1} interactions, the leader's counter is equal to $k-1$. (i) If the next interaction is an interaction between the leader and the market agent then we are done. Hence, A_k=A_{k-1}+1. (ii) if the next interaction is an interaction between the leader and either a white or a black agent, then $(1+A_{k-1})$ interactions have been already performed and since the leader's counter is re-initialized (we go back to node 0), A_k interactions remain to

get to node k. Finally, (iii) if the interaction is an interaction that does not include the leader then, we stay on the same node $(k-1)$ and $T_{k,1}$ interactions remain to get to node k where $T_{k,A}$ is the average number of interactions to get from node $k-1$ to node k. Thus:

$$A_k = p(A_{k-1}+1) + M(A_{k-1}+1+T_{k,1}) + q(A_{k-1}+1+A_k)$$

We need now to determine the expression of $T_{k,1}$. As we did for A_k, we obtain:

$$T_{k,1} = p + M(1+T_{k,1}) + q(1+A_k)$$

$$T_{k,1} = \frac{p+M+q(1+A_k)}{1-M}$$

hence:

$$A_k = p(A_{k-1}+1) + M(A_{k-1}+1+\frac{p+M+q(1+A_k)}{1-M}) + q(A_{k-1}+1+A_k)$$

By solving this equation we obtain (recall that $p+M+q=1$ and that $1-M-q=p$):

$$A_k = \frac{(1-M)A_{k-1}+1}{p}$$

Observe that:

$$A_1 = \frac{1}{p}$$

We prove by induction that :

$$A_k = \frac{1}{p^k}.(1-M)^{k-1}(\frac{1-[\frac{p}{1-M}]^k}{1-\frac{p}{1-M}}) \quad \text{for } k>1 \quad (i)$$

To do so let us verify that

$$A_2 = \frac{1}{p^2}.(1-M)(\frac{1-[\frac{p}{1-M}]^2}{1-\frac{p}{1-M}})$$

$$A_2 = \frac{1}{p^2}.(\frac{(1-M)^2-p^2}{1-M}.\frac{1-M}{1-M-P})$$

$$A_2 = \frac{(1-M+p)}{p^2}$$

$$A_2 = \frac{1}{p}.[\frac{1-M+p}{p}]$$

$$A_2 = \frac{(1-M)A_1+1}{p}$$

Suppose now that (i) holds for A_k and let us prove that (i) holds also for A_{k+1}. Observe that the expression of A_k is a geometric series of parameter $\frac{p}{1-M}$. Thus, we can write A_k in the following way:

$$A_k = \frac{1}{p^k} \cdot \sum_{i=0}^{k-1} (1-M)^{k-1} \cdot [\frac{p}{1-M}]^i$$

We have:

$$A_{k+1} = \frac{(1-M)A_k+1}{p}$$

By replacing A_k by its value, we obtain:

$$A_{k+1} = \frac{(1-M)}{p} \cdot (\frac{1}{p^k} \cdot \sum_{i=0}^{k-1} (1-M)^{k-1} \cdot [\frac{p}{1-M}]^i) + \frac{1}{p}$$

$$A_{k+1} = (\frac{1}{p^{k+1}} \cdot \sum_{i=0}^{k-1} (1-M)^{k} \cdot [\frac{p}{1-M}]^i) + \frac{1}{p}$$

$$A_{k+1} = \frac{1}{p^{k+1}} \cdot (\sum_{i=0}^{k-1} (1-M)^{k} \cdot [\frac{p}{1-M}]^i + p^k)$$

$$A_{k+1} = \frac{1}{p^{k+1}} \cdot (\sum_{i=0}^{k} (1-M)^{k} \cdot [\frac{p}{1-M}]^i)$$

Thus, A_{k+1} is also a geometric series of parameter $\frac{p}{1-M}$. Hence:

$$A_{k+1} = \frac{1}{p^{k+1}} \cdot (1-M)^k (\frac{1-[\frac{p}{1-M}]^{k+1}}{1-\frac{p}{1-M}})$$

Let compute p, M and q.

Probability	Interaction	Value
p	(Leader, Marked agent)	$\frac{2}{n(n-1)}$
q	(Leader, (Black or white) agent)	$\frac{2(n-2)}{n(n-1)}$
M	(non-leader, non-leader)	$\frac{(n-2)}{n}$

We can hence deduce that:

$$A_k = \frac{n(n-1)}{2} \cdot (\frac{(n-1)^k-1}{n-2}) \quad \text{for } k > 1$$