

Realization of Periodic Functions by Self-stabilizing Population Protocols with Synchronous Handshakes

Anissa Lamani^(✉) and Masafumi Yamashita

Department of Informatics, Kyushu University, Fukuoka, Japan
anissa.lamani@gmail.com

Abstract. We consider in the following the problem of realizing periodic functions by a collection of finite state-agents that cooperate by interacting with each other. More formally, given a periodic non-negative integer function f that maps the set of non-negative integers \mathbf{N} to itself, we aim in this paper at designing a distributed protocol with a state set Q and a subset $S \subseteq Q$, such that, for any initial configuration C_0 , with probability 1, there are a time instant t_0 and a constant $d \in \mathbf{N}$ satisfying $f(t+d) = \nu_S(C_t)$ for all $t \geq t_0$, where $\nu_S(C)$ is the number of agents with a state in S in a configuration C . The model that we consider is a variant of the population protocol (PP) model in which we assume that each agent is involved in an interaction at each time instant t , hence the notion of synchronous handshakes. These additional assumptions on the model are necessary to solve the considered problem. We also assume that the interacting pairs are matched uniformly at random.

Keywords: Self-organizing systems · Self-stabilization · Clock synchronization · Oscillators · Group construction · Population protocol · Uniform scheduler

1 Introduction

The food hunting of army ants, the synchronized flashing of fireflies..., these remarkable self-organized behaviors observed in biological systems have intrigued a lot of researchers and a lot of investigations have been initiated to understand these autonomous and self-organized behaviors to implement them in artificial distributed systems.

In a typical artificial distributed system, its global behavior is controlled by its distributed elements, each of which determines its behavior depending only on its local view (i.e., local snapshot of the system). The wider views the elements have, the more delicately the global behavior can be controlled, needless to say; or equivalently, the narrower views they have, the more disorderly the distributed system would behave (at least in the worst case). Note that if no communication (direct or indirect) is allowed then obviously coordination is impossible to achieve. The extreme case is then when a distributed element

is completely isolated from the other elements, and changes its (local) state when it happens to interact with another element. In this paper, provided this extreme case and assuming an arbitrary initial state of the system (i.e., in a self-stabilizing manner), we examine what is necessary to control the distributed system and emerge a global behavior. Specifically, we examine the contribution of synchrony to have the system behave orderly. Self-stabilization is an important feature in distributed computing. It is more challenging to achieve as the system must retrieve, by itself, a correct behavior starting from any possible initial configuration. This permits to tolerate transient faults.

The closest model to our setting is the population protocol (PP) model that was introduced by Angluin et al. [1] to model a collection of finite-state agents that interact with each other in order to accomplish a common task [6]. Computations in PPs are performed through pairwise interactions i.e., when two agents interact, they exchange their local information and update their state according to a common protocol. The interaction pattern is assumed to be unpredictable i.e., each agent has no control on the agents it interact with. Hence, it is assumed that there is an external entity, called scheduler, which is in charge of selecting the pairs of agents for an interaction. Remark that the definition of PP is general enough to represent not only artificial distributed systems as sensor networks but also natural distributed systems as animal populations and chemical reactions.

Let \mathbf{N} be the set of non-negative integers. Given a periodic non-negative integer function $f : \mathbf{N} \rightarrow \mathbf{N}$ with a period τ i.e., $f(t + \tau) = f(t)$ for all $t \in \mathbf{N}$. We present in this paper, a population protocol with a state set Q and a subset $S \subseteq Q$, such that, for any initial configuration C_0 , with probability 1, there are a time instant t_0 and a constant $d \in \mathbf{N}$ satisfying $f(t + d) = \nu_S(C_t)$ for all $t \geq t_0$, where $\nu_S(C)$ is the number of agents with a state in S in a configuration C . That is, we aim at realizing the function f by counting the number of agents in a given set of states S . In other words, our target is to design a protocol to completely control, in a distributed manner, the number of agents with a state in S to represent f . Remark that if we allow a scheduler that may not invite some agents to interact at some time t , the problem becomes unsolvable even for a simplest problem instance. For example, define a function f_τ by $f_\tau(t) = n$ if $t \bmod \tau = 0$ (where n is the size of the population), and $f_\tau(t) = 0$, otherwise. Since the agents can only update their states when they are involved in an interaction, provided that there are at least three agents in the population, there is no PP that realizes f_τ , since all agents must participate in an interaction to update their states and reach a configuration C such that $\nu_S(C) = n$ from a configuration C' in which $\nu_S(C') = 0$ (after convergence). We thus consider a variant of PP in which we assume that every agent is part of an interaction at each instant.

To solve our problem, we also need to consider the self-stabilizing clock synchronization problem. Clock synchronization is one of the important features in distributed computing as it provides a form of logical clock that could be used to solve several distributed problems. The problem has been widely investigated in the context of distributed computing [3, 6, 9]. Under the PP model, the problem was recently addressed in [4] where it has been shown that in the case where

the agents have only a constant number of states, the self-stabilizing synchronization problem is impossible to solve, even if the agents know their covering time, which is the minimum number of transitions for them to have met with each other agent with certainty. A self-stabilizing population protocol is then proposed to solve the phase synchronization problem. In the proposed solution, an unlimited resource station called *Base Station* is used to provide an infinite repetition of phases, $0, 1, \dots, \tau - 1$ and ensure that any agent i does not execute phase $x + 1 \pmod{\tau}$ before all the agents have executed phase $x \pmod{\tau}$ i.e., there are time instants in which the clock values of the agents are not all the same. In this paper, provided our scheduler, we propose a solution for the self-stabilizing synchronization problem using a constant number of states.

Most of the problems investigated in PPs consider the computational power of the model and hence are static. So far, only few investigations have considered *dynamic* problems: in [2], a self-stabilizing token circulation protocol on rings with a pre-selected leader was proposed. In [5], both the self-stabilizing mutual exclusion and the group mutual exclusion problems were considered. In [10], the authors investigated the convergence time and the threshold effects exhibited by Lotka-Volterra-type protocols. In a companion paper [8], the authors investigated the problem of realizing a (digitized) sine curve in a self-stabilizing manner. Under a deterministic scheduler which is adversarial, they showed that the problem is at least as difficult as the self-stabilizing leader election problem, and hence (by [7]) at least n states are necessary to solve the problem.

2 Preliminaries

We consider a collection $A = \{0, 1, \dots, n - 1\}$ of identical (anonymous) finite-state agents that interact with each other in order to cooperate and solve a given problem. Computations are performed through pairwise interactions i.e., when two agents interact, they exchange their information and update their states according to a common deterministic protocol $P = (Q, \delta)$, where Q is a finite set of states (of an agent) and δ is a transition function $(Q \times Q) \rightarrow (Q \times Q)$ that specifies the result of each interaction; if two agents in A with states p and p' interact at time t , their states at $t + 1$ will be q and q' , where $(q, q') = \delta(p, p')$. We assume that each agent is able to interact with any other agent in the population.

Let $p_i \in Q$ be the state of an agent $i \in A$. The n -tuple $C = (p_1, p_2, \dots, p_n)$ of states where each entry p_i corresponds to the state of the agent i , describes the global state of the population, and is called a *configuration*. We frequently regard C as a mapping $A \rightarrow Q$ such that $C(i) = p_i$ for all $i \in A$. Once an interaction pattern has been given to a configuration C , the configuration C' at the next time instant is naturally constructed from C as follows: for each pair (i, j) of agents in the interaction pattern, we replace p_i and p_j in C with q_i and q_j , where $(q_i, q_j) = \delta(p_i, p_j)$. In this paper, we distinguish the *initiator* and the *responder* in δ , so that $\delta(p_i, p_j) = (q_i, q_j)$ may not imply $\delta(p_j, p_i) = (q_j, q_i)$.

Unlike the classical model of PP, we assume that at each instant t , each agent is part of an interaction, we hence assume that n is even. The scheduler, in charge

of creating a maximum matching on all the agents, is assumed to be probabilistic and uniform i.e., the pairs of agents are matched uniformly at random.

Let $P = (Q, \delta)$ and \mathcal{C} be the set of all possible configurations of P , respectively. Given a configuration $C \in \mathcal{C}$ and an interaction pattern, i.e., a set of interactions (i.e., disjoint pairs of agents) $R = \{(i_1, j_1), (i_2, j_2), \dots, (i_m, j_m)\}$, we say that C' yields from C via R , denoted by $C \xrightarrow{R} C'$, if for all $k = 1, 2, \dots, m$, $(C'(i_k), C'(j_k)) = \delta(C(i_k), C(j_k))$, and otherwise, if $i \notin \{i_k, j_k : 1 \leq k \leq m\}$, $C'(i) = C(i)$ holds. Let C_t and R_t be respectively the configuration and the interaction pattern on C_t at time t . An execution \mathcal{E} of a protocol P can be represented by a sequence of configurations and interaction patterns $(C_0, R_0, C_1, R_1, \dots)$ such that for all $t \geq 0$, $C_t \xrightarrow{R_t} C_{t+1}$. Recall that R_t is produced by the scheduler. Since this paper assumes that each agent is part of an interaction and n is even, $|R_t| = n/2$ for any t . By $C \xrightarrow{*} C'$, we denote the fact that a configuration C' is reachable from C after a finite number of transitions.

Let \mathbf{N} be the set of non-negative integers. For any configuration C and a subset of states S , $\nu_S(C)$ denotes the number of agents with a state $p \in S$ in C . Suppose that f is a periodic non-negative integer function $f : \mathbf{N} \rightarrow \mathbf{N}$ with a period τ , i.e., $f(t + \tau) = f(t)$ for all $t \in \mathbf{N}$. This paper investigates the problem of designing a protocol P that *realizes* f (in a self-stabilizing manner). Formally, a protocol $P = (Q, \delta)$ and a subset S of Q are looked for, that eventually realizes f , regardless of its initial configuration C_0 , in the following sense: for any execution $\mathcal{E} : C_0, C_1, \dots$, there are a time instant t_0 and a constant d ($0 \leq d \leq \tau$) satisfying $f(t + d) = \nu_S(C_t)$ for all $t \geq t_0$ with probability 1.

To achieve our goal, we first present a protocol τ -CLK that implements a self-synchronized clock using τ states and then use it to design a protocol PO_h , called a primitive oscillator that realizes a primitive periodic function h whose period is τ and range is $\{0, n\}$. We next explain how to execute a set of primitive oscillators in parallel in such a way as to satisfy prescribed offsets between them, assuming that the agents that participate in different oscillators are disjoint, although they interact to synchronize their executions. Since a periodic function f is represented by the sum of a set of primitive periodic functions h_i with range $\{0, n_i\}$, the problem of designing a protocol that realizes f is reduced to the group construction problem to assign each agent to a group g_i in such a way that each group g_i eventually accommodates n_i agents (to run primitive oscillators PO_{h_i}). The group construction problem for one group is trivial. The general group construction problem can be reduced to the problem for n groups, each consisting of one agent. We present first a protocol for solving the general group construction problem that requires $\Omega(n)$ states. Aiming at the reduction of the space complexity, we next present a heuristic protocol to approximately construct m groups of size n_0, \dots, n_{m-1} respectively, using $m \cdot c \cdot \max\{b_1, \dots, b_{m-1}\}$ states, where c is a small constant larger than 5 and $b_k = n_k/n_0 \in \mathbf{N}$ for $k = 1, 2, \dots, m-1$.

3 Clock Synchronizers and Primitive Oscillators

3.1 Self-stabilizing Clock Synchronization

In this section, we first formulate the self-stabilizing clock synchronization problem taking care of the fact that the scheduler is probabilistic, and show that there is a self-stabilizing clock synchronization protocol that solves the problem. A protocol $P = (T_\tau, \delta_\tau)$, where $T_\tau = \{0, 1, \dots, \tau - 1\}$, is said to be a self-stabilizing modulo τ clock synchronization protocol if for any execution $\mathcal{E} : C_0, C_1, \dots$ starting from any initial configuration $C_0 \in \mathcal{C}$, with probability 1, there are a time instant $t_0 \in \mathbb{N}$ and an integer $d (\geq -t_0)$ satisfying $\nu_{\{(t+d) \bmod \tau\}}(C_t) = n$ for all $t (\geq t_0)$. Consider the following protocol τ -CLK.

Protocol 1. τ -CLK: Self-stabilizing Modulo τ Clock Synchronizer

$$\delta(x, y) = ((\min(x, y) + 1) \bmod \tau, (\min(x, y) + 1) \bmod \tau)$$

Theorem 1. *Protocol τ -CLK is a self-stabilizing modulo τ clock synchronization protocol.*

Proof. Let $\mathcal{E} : C_0, C_1, \dots$ be any execution starting from any initial configuration C_0 . Once it reaches a configuration C_t such that $\nu_{\{(t+d) \bmod \tau\}}(C_t) = n$, by the assumption on the scheduler, no matter what maximum matching is proposed as the interaction pattern, $\nu_{\{(t+d+1) \bmod \tau\}}(C_{t+1}) = n$ holds. Thus it suffices to show that, with probability 1, there is a time instant t such that $\nu_{\{t'\}}(C_t) = n$ holds for some $t' \in T_\tau$.

Since the case of $\tau = 1$ is trivial, we assume that $\tau \geq 2$. Consider a directed graph $\mathcal{G} = (\mathcal{C}, E)$ representing the transition relation between configurations, i.e., $(C, C') \in E$ if and only if $C \xrightarrow{R} C'$ via some interaction pattern R , which implies that when the current configuration is C , the next configuration is C' with a positive probability. Let $\mathcal{C}^* = \{C : \nu_{\{t\}}(C) = n \text{ for some } t \in T_\tau\}$. Thanks to the theory of finite state Markov chain, in order to show that, with probability 1, there is a time instant t such that $\nu_{\{t'\}}(C_t) = n$ holds for some t' , it is sufficient to show that for any $C \in \mathcal{C}$, there is a $C' \in \mathcal{C}^*$ that is reachable from C in \mathcal{G} . We show this fact in the following.

For any $C \in \mathcal{C}$, let $Z_C = \{t \in T_\tau : \nu_{\{t\}}(C) = 0\}$, which is the set of states $t \in T_\tau$ that does not appear in C . By definition, $C \in \mathcal{C}^*$ if and only if $|Z_C| = \tau - 1$. We show that for any $C \notin \mathcal{C}^*$ there is a configuration C' reachable from C in \mathcal{G} such that $|Z_C| < |Z_{C'}|$.

First consider the case in which $|Z_C| = 0$. Recall that n is even and thus $n/2$ is a natural number. If $\nu_{\{\tau-1\}}(C) \leq n/2$, there is a maximum matching R that matches every agent with state $\tau - 1$ to an agent with a different state, and by definition $0 \in Z_{C'}$, where $C \xrightarrow{R} C'$, since an agent with 0 emerges in C' if and only if two agents with state $\tau - 1$ interact in C .

Suppose otherwise that $\nu_{\{\tau-1\}}(C) \geq n/2+1$, which implies that $\nu_{\{\tau-2\}}(C) \leq n/2-1$. Consider a maximum matching R that matches every agent with state $q_{\tau-1}$ to an agent with the same state; if $\nu_{\{\tau-1\}}(C)$ is odd, then R matches the remaining one to an agent with state 0. Then $\nu_{\{\tau-1\}}(C') \leq n/2$, where $C \xrightarrow{R} C'$, since $\nu_{\{\tau-1\}}(C') \leq \nu_{\{\tau-2\}}(C) + 1 \leq n/2$. A configuration C'' such that $0 \in Z_{C''}$ is reachable from C in \mathcal{G} .

Next, consider the case in which $|Z_C| > 0$. Let R be any interaction pattern in C and C' the configuration such that $C \xrightarrow{R} C'$. Observe that if $\nu_{\{t\}}(C) = 0$ then $\nu_{\{(t+1) \bmod \tau\}}(C') = 0$, and hence $|Z_C| \leq |Z_{C'}|$ holds. Assume first that $\nu_{\tau-1}(C) > 0$, if $\nu_{\{\tau-1\}}(C) \leq n/2$, then we can easily find C' such that $|Z_C| < |Z_{C'}|$ holds. Otherwise if $\nu_{\{\tau-1\}}(C) \geq n/2+1$, by a similar argument as above, we can show that there is a path from C to C' in G , such that $0 < \nu_{\{\tau-1\}}(C') \leq n/2$ holds.

By contrast, assume that $\nu_{\tau-1}(C) = 0$. We show that after at most τ steps, $\nu_{\tau-1}(C') > 0$, where $C \rightarrow C'$. Let consider the smallest $t \in [0, \tau-2]$ such that $\nu_t(C) > 0$ (for all $t' = 0, 1, \dots, t-1$, $\nu_{t'}(C) = 0$). Note that $\nu_{t+1}(C') > 0$. If $\nu_{\tau-1}(C') = 0$ then again $\nu_{t+2}(C'') > 0$ where $C' \rightarrow C''$. By repeating the same reasoning, after at most τ steps, $\nu_{\tau-1}(C^-) > 0$ where $C \xrightarrow{*} C^-$. We retrieve the cases discussed above.

3.2 Primitive Oscillators

Let $\alpha = (\iota_a, \iota_p, \iota_d, \iota_\ell)$ be a tuple of four parameters representing respectively *amplitude*, *period*, *delay* and *length* where $0 \leq \iota_d \leq \iota_p$. By f_α , we denote a function with period ι_p defined as follows: $f_\alpha(t) = \iota_a$ if $\iota_d \leq t \pmod{\iota_p} \leq \iota_d + \iota_\ell - 1$, and $f_\alpha(t) = 0$, otherwise.

Figure 1 illustrates a part of f_α , where $\alpha = (5, 10, 6, 4)$. That is, for example, $f_\alpha(x) = 0$ when $x = 0, 1, \dots, 5$, and $f_\alpha(x) = 5$ when $x = 6, 7, 8, 9$. We call such a function (f_α) *primitive (periodic) function*. Note that a constant function is also a primitive function.

In Fig. 1, f_α is presented as a continuous trapezoidal function to emphasize the shape of the oscillations, however, recall that, in reality, f_α is a discontinuous step function.

A pair of protocol $P = (Q, \delta)$ and a subset S of Q is said to be a *primitive oscillator* if it realizes a primitive function f_α : For any execution $\mathcal{E} : C_0, C_1, \dots$ starting from any initial configuration $C_0 \in \mathcal{C}$, with probability 1, there are a time instant $t_0 \in \mathbb{N}$ and an integer $d (\geq -t_0)$ satisfying $\nu_S(C_{t+d}) = f(t)$ for all $t (\geq t_0)$.

Given $\alpha = (n, \tau, \iota_d, \iota_\ell)$, let $T_{\iota_d, \iota_\ell} = \{\iota_d, \iota_d+1, \dots, \iota_d+\iota_\ell-1\}$. Then, the pair $PO_\alpha = (\tau\text{-CLK}, T_{\iota_d, \iota_\ell})$ is a primitive oscillator for f_α , by the next corollary.

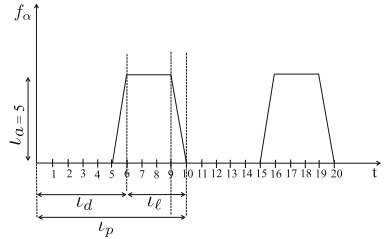


Fig. 1. A primitive (periodic) function f_α , where $\alpha = (5, 10, 6, 4)$

Corollary 2. *Let f be any periodic function with period τ that takes a value in $\{0, n\}$. Then there is a state set $S_f \subseteq Q_\tau$ such that τ -CLK with S_f realizes f .*

Proof. Letting $S_f = \{t \bmod \tau : f(t) = n\}$, τ -CLK with S_f realizes f by definition.

4 Synchronization of Primitive Oscillators

In this section, we investigate how to synchronize several primitive oscillators, in order to satisfy prescribed offsets between them, assuming that the agents that participate in different oscillators are disjoint (i.e., they know to which group they belong), although they do interact to synchronize their executions. The *offset* is the difference, expressed in time, between the delays of two primitive oscillators referenced to the same point in time (see Fig. 2). In our case, the offset is expressed as the difference between the agents' clock values. We assume that the primitive oscillators have the same period, however, they might have different delays. Note that we can assume that the offset is positive without loss of generality, since we can order the primitive oscillators according to the predefined offsets.

To simplify the explanation of our protocol, we first focus on how to synchronize two primitive oscillators. Let PO_{α_k} ($k = 0, 1$) be two primitive oscillators for the set of agents A_k , where $\alpha_k = (n_k, \tau, \iota_{d_k}, \iota_{\ell_k})$ and $n_k = |A_k|$. Let o be the offset. Provided that $A_0 \cap A_1 = \emptyset$ and $|A_0 \cup A_1|$ is even, a protocol that works on $A = A_0 \cup A_1$ is looked for, such that for any execution $\mathcal{E} : C_0, C_1, \dots$ starting from any initial configuration $C_0 \in \mathcal{C}$, with probability 1, there are a time instant $t_0 \in \mathbf{N}$

and integers d_0, d_1 ($\geq -t_0$) satisfying the following conditions: (1) $o = d_1 - d_0$, and (2) for $k = 0, 1$, $\nu_{S_k}(C_{t+d_k}) = f_{\alpha_k}(t)$ for all $t(\geq t_0)$, where S_k is the state set of PO_{α_k} , i.e., $PO_{\alpha_k} = (\tau\text{-CLK}, S_k)$. We call a protocol satisfying these conditions $\langle PO_{\alpha_0}, o, PO_{\alpha_1} \rangle$.

In the following, we present a protocol $P = (Q, \delta)$ and a subset S of Q that satisfy these conditions. First, the set of agents is $A = A_0 \cup A_1$, and the state set is $Q = G_2 \times Q_\tau$, where $G_2 = 0, 1$. An agent $i \in A$ is in state $(k, t) \in Q$ means that it belongs to group A_k and its local time is t . Note that in this section, agents do not change their group A_k , and we concentrate on the synchronization of their local times so that all members in the same group have the same time, while the difference of the local times in A_0 and A_1 is exactly o , provided that initially all members in A_k are in a state in $\{k\} \times T_\tau$, i.e., initially each agent correctly recognizes its group. Let $x = (k, r)$ and $x' = (k', r')$ be two states. The following protocol 2-CPO defines $(y, y') = \delta(x, x')$, where $y = (\ell, s)$ and $y' = (\ell', s')$. In the 2-CPO, δ_τ is the transition function of τ -CLK. Note that the

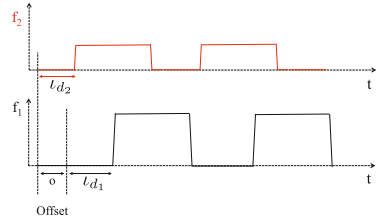


Fig. 2. Offset between two primitive oscillators

description of 2-CPO presents the case in which $k = 0$ and $k' = 1$ holds (and the case in which $k = k'$ holds). Although omitted in 2-CPO, the other case in which $k = 1$ and $k' = 0$ should be defined symmetrically.

Protocol 2. 2-CPO: Coupling Two Primitive Oscillators

```

if  $k = k'$  then /* The agents are part of the same group. */
   $\ell = \ell' = k$  and  $(s, s') = \delta_\tau(r, r')$ 
else /* The agents are part of different groups. */ ( $k = 0$  and  $k' = 1$ ) */
  if  $(r + o) \bmod \tau = r'$  then
     $\ell = k = 0, \ell' = k' = 1, s = (r + 1) \bmod \tau$  and  $s' = (r' + 1) \bmod \tau$ 
  else
     $\ell = k = 0, \ell' = k' = 1, s = (r + 1) \bmod \tau$  and  $s' = (r + o + 1) \bmod \tau$ 

```

Theorem 3. *The protocol defined by 2-CPO is indeed $\langle PO_{\alpha_0}, o, PO_{\alpha_1} \rangle$, provided that initially all agents in A_k have a state in $\{k\} \times T_\tau$ for $k = 0, 1$.*

Proof. Let $\mathcal{E} : C_0, C_1, \dots$ be any execution starting from any initial configuration C_0 such that, for all $i \in A$, $C_0(i) = (0, t)$ for some $t \in T_\tau$ if and only if $i \in A_0$. By the definition of 2-CPO, the agents do not change their groups, and thus through out the execution, all members originally in A_0 (resp. A_1) are in A_0 (resp. A_1).

Observe that agents in A_1 cannot affect agents in A_0 (except to increment time), and the clocks (i.e., the second element t of state $(0, t)$) of agents in A_0 are eventually synchronized by executing τ -CLK.¹ That is, there is a time instant t_0 and a constant d such that for any $t \geq t_0$, all of their clocks show $r = (t + d) \bmod \tau$ at C_t , with probability 1, by Theorem 1. All agents in A_1 also execute τ -CLK, but their clocks are not synchronized, unless all of their clocks show r' satisfying $r' = (r + o) \bmod \tau$ at C_t . On the other hand, if there is a time instant $t > t_0$ at which the clock of each of the agents in A_1 at C_t shows time r' satisfying $r' = (r + o) \bmod \tau$, then the conditions for $\langle PO_{\alpha_0}, o, PO_{\alpha_1} \rangle$ hold.

Thus it suffices to show that there is a time instant $t \geq t_0$ at which all of their clocks are synchronized and show time r' satisfying $r' = (r + o) \bmod \tau$, with probability 1. Thanks again to the theory of finite state Markov chain, we follow the same argument as in the proof of Theorem 1. We define the graph \mathcal{G} on the configuration space \mathcal{C} of 2-CPO representing the transition of this protocol. Let \mathcal{C}^* be the set of configurations corresponding to the desired configurations, and show that for all configuration $C \in \mathcal{C}$, there is a path to a configuration in \mathcal{C}^* . Let C be any configuration. Without loss of generality, we can assume that the clocks of all agents in A_0 are synchronized in C . If $n_0 \geq n_1$, then obviously there

¹ In spite that the size of A_0 may be odd, the clocks of agents in A_0 are eventually synchronized by executing τ -CLK, since every agent that interacts with an agent in different group increments its time, even if it does not interact with an agent in the same group.

is an interaction pattern R that matches all members in A_1 to members in A_0 , and $C' \in \mathcal{C}^*$ is reached from C via R , by the definition of 2-CPO.

Otherwise if $n_0 < n_1$, let Z_C be the set of agents i in A_1 whose clock in C satisfies $r' = (r + o) \bmod \tau$. Consider an interaction pattern R defined as follows:

- M1:** Take a maximum matching among Z_C , so that every agent (except at most one, say i) matches to an agent in Z_C .
- M2:** Take a maximum matching between agents in A_0 and those in $A_1 \setminus Z_C \cup \{i\}$ if there is such an i in M1.
- M3:** Take a maximum matching among the agents whose partners have not been determined in M1 and M2.

Let C' be a configuration reached from C via R . If all agents in A_1 can be matched by M1 and M2 (without using M3), then $C' \in \mathcal{C}^*$.

Otherwise, by M1, $Z_C \setminus \{i\} \subseteq Z_{C'}$. Set $Z_{C'}$ also contains agents in A_1 that are matched with agents in A_0 by M2. Since $n_0 \geq 1$ and $|A_1| + |A_2|$ is even, $|Z_C| < |Z_{C'}|$. Thus there is a path from C to a configuration C'' at which all agents in A_2 can be matched by M1 and M2.

We extend this protocol to synchronize m primitive oscillators. Let PO_{α_k} ($k = 0, 1, \dots, m-1$) be m primitive oscillators for the set of agents A_k , where $\alpha_k = (n_k, \tau, \iota_{d_k}, \iota_{\ell_k})$ and $n_k = |A_k|$. Let o_k ($k = 0, 1, \dots, m-1$) be the offset between PO_{α_k} and $PO_{\alpha_{k+1}}$. Provided that A_k 's are disjoint and $A = \cup_{0 \leq k \leq m-1} A_k$ is of even size, a protocol that works on A is looked for, such that for any execution $\mathcal{E} : C_0, C_1, \dots$ starting from any initial configuration $C_0 \in \mathcal{C}$, with probability 1, there are a time instant $t_0 \in \mathbf{N}$ and integers d_k ($\geq -t_0$) for $k = 0, 1, \dots, m-1$ satisfying the following conditions: (1) $o_k = d_{k+1} - d_k$ for $k = 0, 1, \dots, m-2$, and (2) for $k = 0, 1, \dots, m-1$, $\nu_{S_k}(C_{t+d_k}) = f_{\alpha_k}(t)$ for all $t(\geq t_0)$, where S_k is the state set of PO_{α_k} , i.e., $PO_{\alpha_k} = (\tau\text{-CLK}, S_k)$. We call a protocol satisfying these conditions $\langle PO_{\alpha_0}, o_0, PO_{\alpha_1}, o_1, \dots, o_{m-2}, PO_{\alpha_{m-1}} \rangle$.

We present a protocol $P = (Q, \delta)$ and a subset S of Q that satisfies this condition. First, the set of agents is $A = \cup_{1 \leq k \leq m} A_k$, and the state set is $Q = G_m \times T_\tau$, where $G_m = \{0, 1, \dots, m-1\}$. An agent $i \in A$ is in state $(k, t) \in Q$ means that it belongs to group A_k and its local time is t . We are concerned with synchronization so that all members in the same group have the same time, while the difference of the local times in A_k and A_h is exactly o_k if $h = k+1$, provided that initially all members in A_k are in a state in $\{k\} \times T_\tau$. Let $x = (k, r)$ and $x' = (k', r')$ be two states. The following protocol m -CPO defines $(y, y') = \delta(x, x')$, where $y = (\ell, s)$ and $y' = (\ell', s')$. In the m -CPO, δ_τ is the transition function of τ -CLK. Note that the description of m -CPO presents the case in which $k \leq k'$ holds. Although omitted in m -CPO, the other case in which $k > k'$ should be defined symmetrically.

Protocol 3. *m*-CPO: Coupling *m* Primitive Oscillators

```

if  $k = k'$  then /* The agents are part of the same group. */
   $\ell = \ell' = k$  and  $(s, s') = \delta_\tau(r, r')$ 
else /* The agents are part of different groups. */ ( $k < k'$ ) /*
  if  $((r + o_k) \bmod \tau = r') \vee (k + 1 < k')$  then
     $\ell = k, \ell' = k', s = (r + 1) \bmod \tau$  and  $s' = (r' + 1) \bmod \tau$ 
  else
     $\ell = k, \ell' = k', s = (r + 1) \bmod \tau$  and  $s' = (r + o_k + 1) \bmod \tau$ 

```

By induction on *m* and using Theorem 3 the following theorem holds.

Theorem 4. *The protocol defined by m-CPO is indeed $\langle PO_{\alpha_0}, o_0, PO_{\alpha_1}, o_1, \dots, o_{m-2}, PO_{\alpha_{m-1}} \rangle$, provided that initially all agents in A_k have a state in $\{k\} \times T_\tau$ for $k = 0, 1, \dots, m - 1$.*

For $k = 0, 1, \dots, \tau - 1$, by defining f_k as $f_k(t) = f(t)$ if $t \bmod \tau = k$, we deduce the following:

Proposition 5. *Any periodic function f with period τ can be decomposed into τ primitive periodic functions f_k ($k = 0, 1, \dots, \tau - 1$) such that $f(t) = \sum_{0 \leq k \leq \tau-1} f_k(t)$ for all t .*

Let PO_k be the primitive oscillator for f_k in the decomposition of f in the proof of Proposition 5, for $k = 0, 1, \dots, \tau - 1$. Define $P = (Q, \delta)$ as $\langle PO_0, o_0, PO_1, o_1, \dots, o_{\tau-2}, PO_{\tau-1} \rangle$, where $o_k = 0$ for $k = 0, 1, \dots, \tau - 2$, and δ is τ -CPO. Let $S = \{(k, k) : k = 0, 1, \dots, \tau - 1\} \subseteq Q$.

Corollary 6. *Let C_0 be any configuration such that for any $k = 0, 1, \dots, \tau - 1$, $|\{i : C_0(i) = (k, t) \text{ for some } t \in T_\tau\}| = f(k)$ holds. Then P with S realizes f from C_0 with probability 1.*

Now, the remaining task is to reach a configuration C , starting from any configuration C_0 , that satisfies the initial condition in Corollary 6, which is the theme of the next section.

5 Self-stabilizing Group Composition

Given a number of groups $G_m = \{0, 1, \dots, m - 1\}$ along with their respective sizes $\{n_0, n_1, \dots, n_{m-1}\}$, a protocol is said to solve the self-stabilizing group construction problem if it ensures that given a population of size $n = \sum_{i=0}^{m-1} n_i$, starting from an arbitrary configuration $C_0 \in \mathcal{C}$, the population is eventually divided into m groups of sizes n_0, n_1, \dots, n_{m-1} , respectively.

• **Accurate Solution.** We first present a protocol $P = (G_n, \delta_n)$, denoted n -GC, with the state set $G_n = \{0, 1, \dots, n - 1\}$ that solves the group construction problem for n groups, each of size 1. The idea behind n -GC is the same as the

one used in the protocol proposed in [7], which is used for the leader election and ensures that starting from an arbitrary configuration $C_0 \in \mathcal{C}$, in any execution $\mathcal{E} : C_0, C_1, \dots$, there is a time instant $t_0 \in \mathbf{N}$ from which each agent has a unique state. Transition function δ_n is given below.

Protocol 4. *n*-GC: Self-stabilizing *n*-Group Constructor

if $(r \neq r')$ then $\delta_n(r, r') = (r, r')$
 else $\delta_n(r, r') = (r, (r + 1) \bmod n)$

By the definition of the *n*-GC, we can state the following proposition:

Proposition 7. *Protocol n-GC is a self-stabilizing group construction protocol.*

Corollary 8. *The (general) self-stabilizing group construction problem is solvable by using n-GC.*

Proof. Let n_k ($k = 0, 1, \dots, m - 1$) be the size of Group k . We assume that an agent i is in Group k if its state ℓ satisfies $\sum_{j=0}^{k-1} n_j \leq \ell < \sum_{j=0}^k n_j$. Then by Proposition 7, the corollary holds.

Note that *n*-GC correctly works under a deterministic scheduler, and after the convergence, agents never change the group to which they belong. The (only) drawback however is the size n of state set G_n .

By Theorem 4 and Corollaries 6 and 8, we can construct a protocol that realizes f with probability 1, by applying the fair merge technique. Hence,

Theorem 9. *Any periodic function f is realizable using $(n + \tau)$ states.*

As a note, in the special case where $m = 2$ and $n_0 = n_1$, it is possible to solve accurately the group construction problem using only two states per agent. However, agents change their groups infinitely often and hence, the solution cannot be used directly in our case since the agents need to stay long enough in their group to insure the synchronization of their clocks.

• **Heuristic Protocol.** Aiming at solving the self-stabilizing group construction problem using a smaller number of states, we propose in the following a heuristic protocol $P = (Q, \delta)$, denoted HGC, that approximately solves the group construction problem. It is a heuristic in that we do not have a formal proof that it indeed approximately solves the problem, with probability 1. Some simulations have been performed that show the possible convergence of the protocol, however, the stabilization time is extremely long.

Let n_0, n_1, \dots, n_{m-1} be the sizes of the groups, and assume that they are sorted in the increasing order without loss of generality, i.e., $n_k \leq n_{k+1}$ for $k = 0, 1, \dots, m-2$. Protocol HGC assumes $n_0 | n_j$ for all j . Let $B = \{b_1, b_2, \dots, b_{m-1}\}$, where $n_k = b_k \cdot n_0$. Let $c \in \mathbf{N}$ be a small constant (compared with n) greater

than 5. The state of agent $i \in A$ is (k, t) , where $k \in \{0, 1, \dots, m-1\}$ is the current group the agent belongs to, and $t \in \{0, 1, \dots, c \cdot b_k - 1\}$ indicates its *local progress*. Protocol HGC defines $(y, y') = \delta(x, x')$, where $x = (k, r)$, $x' = (k', r')$, $y = (\ell, s)$ and $y' = (\ell', s')$.

Protocol 5. HGC: Heuristic Group Constructor

```

if  $k = k'$  then /* The agents are part of the same group. */
  if  $r = r' = c \cdot b_k$  then  $\ell = (k + 1) \bmod m$ ,  $s = 0$  and  $y' = y$ 
  else
    if  $r < c \cdot b_k$  then  $\ell = k$ ,  $s = r + 1$  and  $y' = x'$ 
    if  $r' < c \cdot b_{k'}$  then  $y = x$ ,  $\ell' = k'$  and  $s' = r' + 1$ 
  else /* The agents are part of different groups. */
     $\ell = k$ ,  $s = 0$ ,  $\ell' = k'$  and  $s' = 0$ 

```

To understand the idea of Protocol HGC, assume that $n_0 = n_1 = \dots = n_{m-1}$ and thus $b_k = 1$ for all $k = 1, 2, \dots, m-1$. In order for an agent i to change its group k to $((k+1) \bmod m)$, agent i needs to interact $c \cdot b_k = c$ consecutive times with an agent in the same group. That is, for any sufficiently large population, as long as the difference in the group sizes is too large, there would be agents who change their groups, which eventually leads a configuration to an equilibrium state in which the size of each group is approximately the same.

As a final note, since the agents may change their group during the execution of HGC, in order to realize a given function f using the fair merge of m -CPO and HGC, we need to make sure that when an agent updates its group, it also updates its clock value to adjust it to the new offset. That is, when an agent interacts, it first executes the actions of m -CPO (assume that the new computed clock value is s) and then executes the actions of HGC. If by executing HGC, it changes the group it belongs to, it also updates its clock value to respect the offset o (if any). That is, its new clock value is updated to $(s + o) \bmod \tau$.

6 Conclusion

In this paper, we have investigated the problem of realizing a periodic function f assuming a variant of the PP model in which each agent is part of an interaction at each time instant and the interacting pairs are chosen uniformly at random.

Several open problems arise from this work: (1) The problem of designing a memory efficient self-stabilizing protocol that solves the group construction problem under our setting remains open. That is, our group construction protocol is based on n -GC which requires n states, independently of the actual specification. Provided that the number of groups is $m = o(n)$, is it possible to solve the problem in $O(m)$ space?. Observe that by using less than n states, agents need to change their group infinitely often (since initially, all agents might have the same state). Hence, for our implementation, we also need to ensure that

agents stay in a given group long enough to allow the stabilization of the primitive oscillators. (2) While protocol HGC seems to achieve the desired goal from the different simulations performed, its formal analysis remains open. More precisely, is it possible to estimate the “approximation ratio” of HGC?. (3) We have assumed that every agent is part of an interaction at each time instant, this assumption is necessary to represent accurately a given periodic function f . If we relax the specification and allow an approximation of the periodic function f , we may choose a weaker scheduler that chooses $1 \leq p < n/2$ pairs of agents for an interaction. It will be then challenging to investigate the impact of the degree of synchrony (number of interactions at each instant), on the considered problem.

References

1. Angluin, D., Aspnes, J., Diamadi, Z., Fischer, M.J., Peralta, R.: Computation in networks of passively mobile finite-state sensors. In: PODC, pp. 290–299 (2004)
2. Angluin, D., Aspnes, J., Fischer, M.J., Jiang, H.: Self-stabilizing population protocols. TAAS, **3**(4), 13:1–13:28 (2008)
3. Awerbuch, B., Kutten, S., Mansour, Y., Patt-Shamir, B., Varghese, G.: A time-optimal self-stabilizing synchronizer using A phase clock. IEEE Trans. Dependable Secure Comput. **4**(3), 180–190 (2007)
4. Beauquier, J., Burman, J.: Self-stabilizing synchronization in mobile sensor networks with covering. In: Rajaraman, R., Moscibroda, T., Dunkels, A., Scaglione, A. (eds.) DCOSS 2010. LNCS, vol. 6131, pp. 362–378. Springer, Heidelberg (2010). doi:[10.1007/978-3-642-13651-1_26](https://doi.org/10.1007/978-3-642-13651-1_26)
5. Beauquier, J., Burman, J.: Self-stabilizing mutual exclusion and group mutual exclusion for population protocols with covering. In: Fernández Anta, A., Lipari, G., Roy, M. (eds.) OPODIS 2011. LNCS, vol. 7109, pp. 235–250. Springer, Heidelberg (2011). doi:[10.1007/978-3-642-25873-2_17](https://doi.org/10.1007/978-3-642-25873-2_17)
6. Boulmier, C., Petit, F., Villain, V.: When graph theory helps self-stabilization. In: PODC, pp. 150–159 (2004)
7. Cai, S., Izumi, T., Wada, K.: 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. Cooper, C., Lamani, A., Viglietta, G., Yamashita, M., Yamauchi, Y.: Constructing self-stabilizing oscillators in population protocols. In: Pelc, A., Schwarzmann, A.A. (eds.) SSS 2015. LNCS, vol. 9212, pp. 187–200. Springer, Heidelberg (2015). doi:[10.1007/978-3-319-21741-3_13](https://doi.org/10.1007/978-3-319-21741-3_13)
9. Couvreur, J., Francez, N., Gouda, M.G.: Asynchronous unison (extended abstract). In: ICDCS, pp. 486–493 (1992)
10. Czyżowicz, J., Gąsieniec, L., Kosowski, A., Kranakis, E., Spirakis, P.G., Uznański, P.: On convergence and threshold properties of discrete Lotka-Volterra population protocols. In: Halldórsson, M.M., Iwama, K., Kobayashi, N., Speckmann, B. (eds.) ICALP 2015. LNCS, vol. 9134, pp. 393–405. Springer, Heidelberg (2015). doi:[10.1007/978-3-662-47672-7_32](https://doi.org/10.1007/978-3-662-47672-7_32)

Theory and Practice of Natural Computing
5th International Conference, TPNC 2016, Sendai,
Japan, December 12-13, 2016, Proceedings
Martín-Vide, C.; Mizuki, T.; Vega-Rodríguez, M.A. (Eds.)
2016, XIV, 221 p. 88 illus., Softcover
ISBN: 978-3-319-49000-7