Bio-Inspired Algorithms for Decentralized Round-Robin and Proportional Fair Scheduling

Roberto Pagliari, Y.-W. Peter Hong, and Anna Scaglione

Abstract

In recent years, several models introduced in mathematical biology and natural science have been used as the foundation of networking algorithms. These bio-inspired algorithms often solve complex problems by means of simple and local interactions of individuals. In this work, we consider the development of decentralized scheduling in a small network of self-organizing devices using the model of pulse-coupled oscillators (PCO). Firstly, by following Peskin’s PCO model with inhibitory coupling, we show that round-robin scheduling can be achieved with weak convergence, i.e., the nodes transmission times remain separated by a constant of equal amount, but their clocks continue to drift at unison. Then, we introduce two ways to achieve strict desynchronization: one by restricting the pulse coupling to only a subset of neighboring nodes and the other by imposing a more deliberate coupling rule where a node’s pulsing time is only affected by its immediate neighbors. More interestingly, by having each node maintaining two local clocks, we show that it is possible to achieve a proportional fair schedule in a decentralized way. The convergence of these algorithms is studied both analytically and numerically.

I. INTRODUCTION

Long before man made communication network existed, natural phenomena had ways of creating what we can call order. One of the first scientific observers of these phenomena was the dutch physicist Christian Huygens, who in the 17th century studied the synchronization of two penduli mounted on the same beam, which he attributed to the “imperceptible motion of the
air” [1]. Centuries later, in 1975, Charles Peskin [2] modeled the sinoatrial node cells that are responsible for the pumping of the hearth as leaky integrators (a common resistor-capacitor series) followed by a threshold element; as the voltage across the capacitor reached a threshold each cell would broadcast to its neighbors a current, perturbing the voltage at the other capacitors. The Pulse Coupled Oscillator (PCO) model was born. Since then, the model has become extremely popular in basic science and has offered insights on a number of phenomena, as is beautifully narrated in the book by Steven Strogatz [3]. Less obvious to the engineering arena was the idea that the PCO model offered a remarkably simple architecture suitable for self-organization in small networks of very simple low-cost devices. It is not only the physical simplicity of the mechanisms that are striking - it is also how in PCO networks the communications and the nodes computation of the schedule are intertwined and inseparable. The elegant idea of getting two things at the price of one defies the modularity of the engineering approach, which consists in designing the information channel and computations as two separate and interacting units. Even more technical aspects, such as duplexing transmission and reception are elegantly addressed by the PCO algorithm. This paper shows how to expand the functionality of the PCO to provide not only common frame synchronization, but also a decentralized multiple access scheduling.

Motivation and Main Contribution

Wireless Sensor Networks (WSNs) have been exploited today by literally hundreds of applications, ranging from habitat, infrastructure, and environmental monitoring, to localization and tracking, to cooperation and coordination of mobile agents. However, networking among sensors is often constrained by limitations in energy, bandwidth, and computational power. Most popular communication protocols, such as IEEE 802.15.4/ZigBee [4] and Bluetooth [5], inherit the layered architecture of classical wireless networks, with the aim of enabling standardization of commercial products. Yet, these schemes have been shown to be orders of magnitude above the limits imposed by many applications, such as Wireless Body Area Networks (WBANs) [6]. Many sensor MAC protocols have been proposed in the past (see, e.g., [7]), focusing predominantly
on two extremes of the spectrum, namely, random access, such as variants of slotted ALOHA or carrier sense medium access (CSMA), on one side and centralized scheduling, such as TDMA or CDMA, on the other. Random access is not efficient when the pattern of transmissions is periodic. In this scenario, an appealing option would be a decentralized MAC protocol able to exploit the regularity of the transmission, thereby mitigating the energy consumption, without the need for a master node in charge of handling the process.

In the literature, a number of papers has already been devoted to promoting the use of PCO as a synchronization primitive [8]–[10]. By inverting the polarity of the coupling signal, it has been observed in [3] that the PCO network can be led to a pulsing pattern that yields constant spacing between neighboring pulses. This emergent behavior, called “desynchronization”, has been utilized to achieve equal-share round-robin scheduling in [13]. In this work, we first show that the original PCO model with inhibitory coupling yields only weak desynchronization, which refers to the case where each node’s pulsing does not occur exactly once every period (according to its local clock), even though a constant spacing is maintained. However, we show that, by restricting each coupling to only a subset of nodes, strict desynchronization can be attained. Moreover, by having each node maintain two local clocks, we show that it is also possible to achieve proportional fair scheduling according to the different requests made by the nodes. The proposed decentralized scheduling algorithms, which are based on bio-inspired PCO primitives for desynchronization, provide a scalable and robust solution for medium access control (MAC) in sensor networks.

II. BACKGROUND ON SYNCHRONIZATION OF PCOs

Pulse coupled oscillators (PCOs) are elements that pulse individually in a periodic manner if separated, but may alter their pulsing patterns in response to the signals heard from other elements if interconnected (hence the adjective “coupled”). In particular, the recipient of a pulse will move earlier or later its own firing by altering a local clock that regulates the pulse emission.
of the node. By following simple local coupling rules, networks of PCOs are known to produce a variety of pulsing patterns, which have been used to model the spiking of neurons in the brain [12] or the flashing of fireflies [3]. By emulating the PCO dynamics that lead to synchronous pulsing patterns, a scalable network synchronization algorithm has been proposed in [8] for sensor networks. We consider a network of \( n \) sensors inter-connected through direct transmission links.\(^1\)

When in isolation, each node emits a pulse periodically every \( T_f \), which is referred to as the firing cycle. The evolution of the local time at node \( i \) is characterized by a phase variable

\[
\Phi_i(t) = \frac{t}{T_f} + \phi_i(0) \mod 1, \tag{1}
\]

which increases linearly from 0 to 1 in each cycle. A pulse is emitted when \( \Phi_i(t) = \frac{t}{T_f} + \phi_i(0) = 1 \). Using an analogy with an alarm clock, \( \Phi_i(t) \) is, basically, the local time at node \( i \) normalized by the firing cycle \( T_f \), and \( \phi_i(0) \) is an initial offset at absolute time \( t = 0 \). The local phase variable at each node, say node \( i \), is transformed into the local state variable \( X_i(t) \) by the function \( f \), i.e., we have \( X_i(t) = f(\Phi_i(t)) \), where \( f \) is referred to as the PCO dynamics, and it is the solution of the partial differential equation that describes the uncoupled PCO (see e.g. [12] for more details).

When the nodes are interconnected, the reception of a pulse at any given node adds \( \varepsilon \) (called coupling) to the local state variable \( X_i(t) \), altering the phase and, thus, the next pulsing time of the nodes receiving the coupling signal. Specifically, if node \( i \) receives a pulse at time \( t \)

\[
\Phi_i(t^+) = \lim_{\eta \to 0} \Phi_i(t + \eta) = f^{-1}(X_i(t) + \varepsilon) = f^{-1}(f(\Phi_i(t)) + \varepsilon) \tag{2}
\]

upon the reception of the pulse. In the seminal paper by Mirollo and Strogatz, i.e., [3], conditions have been given for the coupling strength \( \varepsilon \) and the dynamics \( f \) that guarantee convergence to a synchronous state:

\(^1\)As common in many works on PCOs, the transmissions and receiver processing for coupling signals is highly idealized. The pulses emitted by each node are received and detected instantaneously and without error by every other node in the network. We shall adopt these assumptions in our analytical studies.
Theorem 1 (SYNCHRONIZATION [3]): For any positive coupling, i.e. $\varepsilon > 0$, the set of initial states \( \{\phi_1, \phi_2, \ldots, \phi_n\} \) that never result in synchrony has measure zero if the function \( f \) is smooth, monotonically increasing, and concave down.

A well known example of the dynamics mentioned above is the one given by Peskin in [2] where \( f(\Phi_i(t)) = (1 - e^{-\gamma \Phi_i(t)})/(1 - e^{-\gamma}) \). More interestingly, it has also been shown in [3] that, with the dynamics \( f \) mentioned above and a negative coupling (i.e., $\varepsilon < 0$), the nodes will separate their firing times asymptotically by a constant amount. The same result occurs if the function \( f \) is concave-up and the coupling strength is positive $\varepsilon > 0$ (see Figure 1). Thus, it is evident that by tuning local parameters, such as the updating function or the coupling strength, the PCO dynamics can produce different global behaviors that can be used as the basis of decentralized scheduling algorithms. Examples are given in the following sections.

III. ROUND-ROBIN SCHEDULING WITH PCO DESYNCHRONIZATION

The problem of desynchronization can be considered the dual of the consensus problem, where the states of the nodes coalesce under the action of the network dynamics. Instead, nodes that are desynchronized will alternate their pulses in order with constant spacing between each other. The constant spacing produced by the PCO dynamics can be utilized as the basis of round-robin scheduling [11]. Specifically, suppose that the initial phases of the nodes are ordered such that $\phi_1(0) < \phi_2(0) < \cdots < \phi_n(0)$, i.e., the nodes pulse in the inverse order of their indices. The initial order of firing is maintained, as we will clarify next; we call the \( R \)th round the \( R \)th time we had \( n \) firing events. Between the \( m \)th and the \( m+1 \)-th firing, each node is an oscillator with the same period \( T_f \), whose phase evolves as in (1) with an offset $\phi_i(m)$ which is updated at every firing event. We can imagine the set of \( n \) nodes as balls placed on a circle of circumference equal to 1 (the normalized period) moving clockwise at the same speed in decreasing order of index. When a node, say node \( i \), crosses the finish line (i.e., $\Phi_i(t) = 1$), the node emits a pulse and, at the same time, other nodes phases will jump based on (2). To study the evolution of the
system it is sufficient to track how the phase distances between the nodes change. Let
\[
\Delta_i(t) = \Phi_i(t) - \Phi_{i-1}(t) \pmod{1}, \quad \text{for } i = 2, \ldots, n,
\]
\[
\Delta_1(t) = \Phi_1(t) - \Phi_n(t) \pmod{1}
\]
be the phase difference between nodes \(i\) and \(i-1\) at time \(t\). Following our previous analogy, where we view the nodes as balls placed on a circle of circumference equal to 1, the parameter \(\Delta_i(t)\) refers to the distance around the circumference between neighboring balls \(i\) and \(i-1\).

**Remark 1** The pulsing of node \(i\) at time \(t_i\) triggers progress in the algorithm, as well as informs the other nodes of the phase difference between their clocks and that of the pulsing node, via the readings of \(\Phi_j(t_i)\). Notice that the difference \(\Phi_i(t) - \Phi_{i-1}(t)\) changes sign whenever node \(i\) fires, becoming negative. For \(-1 \leq \Phi_i(t) - \Phi_{i-1}(t) = -\xi \leq 0\) we have \(\Phi_i(t) - \Phi_{i-1}(t) \pmod{1} = -\xi - \lfloor -\xi \rfloor = 1 - \xi = 1 + \Phi_i(t) - \Phi_{i-1}(t) \geq 0\). Considering this fact, in the following, the firing of node \(i\) at time \(t_i\) will produce the updates:
\[
\Delta_i(t_{i}^+) = 1 - f^{-1}(f(\Phi_{i-1}(t_i)) + \varepsilon) \quad (4)
\]
\[
\Delta_j(t_{i}^+) = f^{-1}(f(\Phi_j(t_i)) + \varepsilon) - f^{-1}(f(\Phi_{j-1}(t_i)) + \varepsilon) \quad j \neq i \quad (5)
\]
where the change in \(\Delta_i(t_{i}^+)\) is due to the update of node \(i-1\).

Each firing modifies the distances between the nodes and, thus, the state of the system can be described by the vector \(\mathbf{\Delta}(t) = (\Delta_n(t), \ldots, \Delta_1(t))^T\), which follows a trajectory over the \(n\)-dimensional plane defined by \(\sum_{i=1}^{n} \Delta_i(t) = 1\), subject to the constraints \(\Delta_i(t) \geq 0, \forall i\). Furthermore, since the nodes turn around the circle at the same speed when no firing occurs, the state of the network \(\mathbf{\Delta}(t)\) will not change in between firings. Then, by indicating with \(t_j[k]\) the firing event of node \(j\) at round \(k\), we have that \(\Delta_i(t) = \text{constant}, \forall i\) and \(\forall t \in (t_j[k], t_{j-1}[k])\). Therefore, we can define \(\Delta_i[k, j] \triangleq \Phi_i(t) - \Phi_{i-1}(t) \pmod{1}\) for \(t \in (t_j[k], t_{j-1}[k])\). The phase differences at the end of round \(R\) are denoted by \(\mathbf{\Delta}[R] \triangleq \mathbf{\Delta}[R, 1] = (\Delta_n[R, 1], \ldots, \Delta_1[R, 1])^T\). The evolution of the system is described fully by the vector \(\mathbf{\Delta}[R]\). We can achieve two types
of convergence, namely, strict and weak desynchronization, as described below.

**Definition 1:** The network is strictly desynchronized if the phase differences converge to a constant value as \( t \to \infty \). This is equivalent to saying that:

\[
\lim_{t \to \infty} \Delta_i(t) = \lim_{t \to \infty} \Phi_i(t) - \Phi_{i-1}(t) \mod 1 = \text{constant}, \quad \text{for } i = 2, \ldots, n \tag{6}
\]

and

\[
\lim_{t \to \infty} \Delta_1(t) = \lim_{t \to \infty} \Phi_1(t) - \Phi_n(t) \mod 1 = \text{constant}.
\]

**Definition 2:** The network is weakly desynchronized if the phase difference of nodes in each round, i.e. \( \Delta[R] \), converges to a fixed point \( \Delta^* \), i.e.: ²:

\[
\lim_{R \to \infty} ||\Delta[R] - \Delta^*||_1 = 0 \tag{7}
\]

When strict desynchronization is reached, the nodes will fire at fixed time instants in each period. When weak desynchronization is reached, instead, the time between two consecutive firings in each round converges to a constant, but the phase differences are not fixed for all \( t \).

Clearly, strict desynchronization implies weak desynchronization, but not the other way around.

We will make also use of the following:

**Definition 3** [13]: The network is said to be \( \epsilon \)-desynchronized if \( ||\Delta[R] - \Delta^*||_1 < \epsilon \), where \( \Delta^* \) is the fixed point (equal to \( \frac{1}{n} \cdot 1 \) in the case of round-robin scheduling).³

**Definition 4:** The convergence time is the minimum number of rounds needed to achieve \( \epsilon \)-desynchronization, i.e., \( R^* = \inf \{ R \} \), with \( R = \{ r : ||\Delta[r] - \Delta^*||_1 < \epsilon \} \).

A. Weak Desynchronization

The protocol we describe first is inspired by the PCO model, introduced by Peskin [2], with inhibitory coupling. Rather than changing the polarity of the coupling, we maintain \( \epsilon > 0 \) and introduce a convex function \( f \) (see Figure 1). This system produces the same global effects as described in the following. Specifically, by taking the dynamics as \( f(\Phi_j(t)) = -\log(\Phi_j(t)) \) and

²Note that norm-1 will be used throughout this paper unless stated otherwise, i.e., \( || \cdot || = || \cdot ||_1 \).

³In the rest of the paper we will use norm 1, i.e., \( || \cdot || = || \cdot ||_1 \).
positive coupling $\varepsilon = -\log(1 - \alpha)$, where $\alpha \in (0, 1)$, the firing of node $i$ at time $t_i$ will induce a jump in the phase of node $j$ to

$$\Phi_j(t_i^+) = f^{-1}(f(\Phi_j(t_i)) - \log(1 - \alpha)) = (1 - \alpha)\Phi_j(t_i).$$  \hspace{1cm} (8)

We can now state the following result (see Appendix A for the proof):

**Theorem 2 (Weak Desynchronization):** For any $\alpha \in (0, 1)$, the dynamics in (8) will converge to weak desynchronization in $R^* \leq -\frac{1}{n_0 \log(1-\alpha)}$ rounds, except over a set of initial conditions of measure zero. The fixed point is $\Delta^* = (\Delta^*_n, \ldots, \Delta^*_1)^T$, where $\Delta^*_i = \frac{(1-\alpha)(1-\varepsilon)}{1-(1-\alpha)n_0} \forall i$.

An illustrative example of weak desynchronization for a 2-node network is given in Fig. 3.

**B. Strict Desynchronization**

Previously, any firing causes all the other nodes to step back in phase and, thus, it is difficult to settle into a stable configuration (i.e., strict desynchronization). Intuitively, a firing node should only push the nodes whose phase is too close, suggesting the introduction of a threshold mechanism to rule out the nodes that are not close enough to constitute a conflict. Based on this intuition, we introduce the following updating rule (assuming node $i$ fires at time $t_i$)

$$\Phi_j(t_i^+) = \begin{cases} f^{-1}(f(\Phi_j(t_i)) + \varepsilon), & \text{if } \Phi_j(t_i) \in \left(1 - \frac{1}{n_0}, 1\right) \\ \Phi_j(t), & \text{otherwise.} \end{cases}$$ \hspace{1cm} (9)

$\forall j \neq i$, where the parameter $n_0 \geq 1$ controls the spacing between nodes, and $f(\Phi_j(t)) = -\log(1 - n_0(1 - \Phi_j(t)))$, which is similar to the convex function in Fig. 2 but defined only over the range $\left(1 - \frac{1}{n_0}, 1\right)$. We can see that the phase of node $j$ is pushed toward the point $1 - 1/n_0$ through a convex combination with parameter $\alpha = 1 - e^{-\varepsilon} \in (0, 1)$. In fact,

$$\Phi_j(t^+) = f^{-1}(f(\Phi_j(t)) + \varepsilon) = (1 - \alpha)\Phi_j(t) + \alpha \left(1 - \frac{1}{n_0}\right) \forall j \neq i. \hspace{1cm} (10)$$

In particular, when $n_0 = 1$, (9) reduces to (8). We can now state the following results (see Appendix B for proof):
Theorem 3 (Strict Desynchronization): For any \( \varepsilon > 0 \), the dynamics in (9), with \( n_0 = n \), achieve to strict desynchronization for all initial conditions, except over a set of measure zero.

An additional result is given in the following Lemma (see Appendix C for proof):

Lemma 1: An achievable convergence time is \( R^* \leq \frac{\log\left(\frac{2n}{\varepsilon}\right)}{\log(1-\alpha)} \).

The algorithm we described reaches strict desynchronization in a way that depends on the number of active nodes. Therefore, extra complexity needs to be introduced to set the parameter \( n_0 \) appropriately.

In [13], an alternative algorithm called DESYNCH that does not rely on the knowledge of \( n \) has been proposed. The key idea is that, each node performs only one update per round, upon hearing the pulse from the node firing after it. In the update node \( i \) will deliberately move its phase toward the middle-point between the estimated phases of node \( i+1 \) and \( i-1 \). It seems reasonable that the nodes will eventually settle to having \( \Delta_i(t) = \frac{1}{n} \). More specifically, node \( i \) at round \( k-1 \) records the value \( \zeta = 1 - \Phi_i(t_{i+1}[k-1]) \). At round \( k \) after \( t_{i-1}[k] \) it performs the update: \( \Phi_i(t_{i-1}[k]) = (1-\eta)\Phi_i(t_{i-1}[k]) + \frac{\eta}{2}\hat{\Phi}_{i+1}(t_{i-1}[k]) \), where \( \hat{\Phi}_{i+1}(t_{i-1}[k]) = \zeta + \Phi_i(t_{i-1}[k]) \) is the phase of node \( i+1 \) estimated by node \( i \) at time \( t_{i-1}[k] \). A conjecture on the convergence of the DESYNCH protocol was given in [11] and is restated below.

Conjecture 1 ([11]): \( n \) nodes governed by DESYNCH will always be driven to strict desynchronization (for \( n < 500 \)) in \( R^* = \frac{1}{\alpha}n^2\log\left(\frac{1}{\varepsilon}\right) \) rounds.

Note that, in essence, the DESYNCH protocol also restricts coupling to a subset of nodes (i.e., its immediate phase neighbors) and thus is able to achieve strict desynchronization. In the following section, we show how proportional fair scheduling can also be achieved by adopting a similar concept and by maintaining two phase variables at each node.

IV. PCO PRIMITIVES FOR PROPORTIONAL FAIR SCHEDULING

In general, nodes may have different bandwidth demands since they could, in fact, be monitoring different phenomena. Therefore, the assignment of the same amount of time to each sensor...
may not be the best choice. In this setup, we assume that each node has a specific request,
\( K_i \), where \( i \) is the node index.\(^4\) The value \( K_i \) lies between 1 and a maximum value \( K_{\text{max}} \), which is fixed \textit{a priori} and equal for all the nodes, that is, \( K_i \in [1, K_{\text{max}}] \) for \( K_{\text{max}} < \infty \). Basically, \( K_i \) stands for the amount of the resource node \( i \) is hoping to obtain by negotiation.

On the other hand, since our objective is to derive a negotiation scheme not based on explicit exchange of information, the nodes may be requesting more than the actual available resource. More formally, we would like each node to obtain, by negotiation, a portion of time equal to \( \frac{K_i}{K} T_f \), where \( K = \sum_{i=1}^{n} K_i \) is the total demand of all nodes.

To achieve this goal, we make use of two phase variables for each node, indicated by \( \Phi_i(t) \) and \( \Psi_i(t) \), where \( i \) is the index of the node. Initially, \( \Phi_i(0) = \Psi_i(0) = \Omega_i(0), \forall i \), with \( \Omega_i(0) \) a random number in \([0,1)\). When nobody is firing, the two phases will evolve as

\[
\begin{align*}
\Phi_i(t) &= (\Omega_i(0) + t/T) \mod 1, \\
\Psi_i(t) &= (\Omega_i(0) + t/T) \mod 1,
\end{align*}
\]

and fire when \( \Phi_i(t) = 1 \) and when \( \Psi_i(t) = 1 \). We will call clocks \textit{adjacent} if they fire one after the other. In the scheme, a \( \Phi \)-type clock is always before a \( \Psi \)-type clock of the node with lower index. As done in the previous section, we will indicate as \( t_j[k] \) the firing time of the clock \( \Psi_j(t) \) at round \( k \) and define similarly \( \tau_j[k] \) as the firing of the clock \( \Phi_j(t) \). In this case as well the nodes fire in descending order of their indices, from \( n \) to 1. In the proposed updating rule, these two phase variables expand their distance, until they are pushed back by the firing of neighboring nodes. As in the DESYNC protocol, the updates of \( \Phi_i(t) \) and \( \Psi_i(t) \) occur only once in a round, and the values recorded for the update at round \( k \) are \( \Psi_i(t_{i+1}[k]) \) and \( \Phi_i(\tau_{i-1}[k]) \).

\(^4\) An example of proportionally fair schedule is in the protocol IS-895, aimed at unleashing multi-user diversity taking into consideration fairness issues in the channel assignment.
created between their elbows (i.e. the two firing times) as their own collision free interval for transmission. More specifically, we shall have the firing of $\Phi_i(t)$ mark the beginning of the transmission interval of node $i$ and have the firing of $\Psi_i(t)$ mark its end.

In preparation for its update in round $k$ each node $i$ keeps a record of the distance $\Psi_{i+1}(t) - \Phi_i(t) \mod 1$ by buffering the value $1 - \Phi_i(t_{i+1}[k-1])$ when clock $\Psi_{i+1}(t_{i+1}[k-1]) = 1$. The update of node $i$ in round $k$ is performed after the firing time $\tau_{i-1}[k]$, when the node reads also the value $\Psi_i(\tau_{i-1}[k]) = \Psi_i(t) - \Phi_{i-1}(t) \mod 1$. Ideally, the node would want to divide the interval between $\Psi_{i+1}(t)$ and $\Phi_{i-1}(t)$ in $K_i + 1$ parts and place its own clocks so that they take over $K_i$ units of this interval. The ideal rule is: (see Figure 4):

$$\begin{align*}
\Psi_i(\tau_{i-1}^+[k]) &= \eta \Psi_i^{\text{target}} + (1 - \eta) \Psi_i(\tau_{i-1}[k]) \\
\Phi_i(\tau_{i-1}^+[k]) &= \eta \Phi_i^{\text{target}} + (1 - \eta) \Phi_i(\tau_{i-1}[k])
\end{align*}$$

(12)

where $\eta \in (0,1)$, and $\Psi_i^{\text{target}} = \max\left(1/2 + K_i, \Psi_{i+1}(\tau_{i-1}[k]), \Psi_i(\tau_{i-1}[k])/2\right)$ and similarly $\Phi_i^{\text{target}} = \min\left(1/2 + K_i, \Psi_{i+1}(\tau_{i-1}[k]), \Psi_i(\tau_{i-1}[k])/2\right)$. In essence, the nodes are trying to stretch the space in between clocks to make the difference $\Phi_i(\tau_{i+1}[k]) - \Psi_i(\tau_{i+1}[k])$ look more like $\Phi_i^{\text{target}} - \Psi_i^{\text{target}}$, which, if the first arguments are picked by max and min, is $\Phi_i^{\text{target}} - \Psi_i^{\text{target}} = K_i/2$. This may not be too aggressive, and the max and min are needed to avoid the possibility that the clock interleave their phases. When the second argument of max or min are picked, the clocks just stretch their distance going in the middle point of the gaps between them and their adjacent clocks.

The only real problem in using this rule is that $\Psi_{i+1}(\tau_{i-1}[k])$ is not really available, and all that the node $i$ has is the estimate it made of this value in round $k - 1$, namely $\hat{\Psi}_{i+1}(\tau_{i-1}[k]) = 1 - \Phi_i(t_{i+1}[k-1]) + \Phi_i(\tau_{i-1}[k])$. Hence, the actual rule uses this value in computing the target points $\hat{\Psi}_i^{\text{target}}$ and $\hat{\Phi}_i^{\text{target}}$, in lieu of $\Psi_{i+1}(\tau_{i-1}[k])$. As discussed in [14], the use of this outdated rule uses this value in computing the target points $\hat{\Psi}_i^{\text{target}}$ and $\hat{\Phi}_i^{\text{target}}$, in lieu of $\Psi_{i+1}(\tau_{i-1}[k])$. As discussed in [14], the use of this outdated

---

5In the trivial case where there is only one node, the reverse will happen, instead of using the zero space between $\Phi_i(t)$ and $\Psi_i(t)$ the node will use the entire cycle for itself.
estimate does not appear to affect the convergence in numerical simulations. Also in this case, the evolution of the system depends uniquely on a state vector \( \Delta(t) \) that includes the following quantities in its odd and even entries, respectively:

\[
\Theta_i(t) = \Psi_i(t) - \Phi_{i-1}(t) \pmod{1}, \quad \Gamma_i(t) = \Phi_i(t) - \Psi_i(t) \pmod{1} \quad (13)
\]

In [14] we studied the convergence of a simplified system, where \( \hat{\Psi}_{i+1}(\tau_{i-1}[k]) = \Psi_{i+1}(\tau_{i-1}[k]) \) and when we assumed that the initial condition and parameters rendered always \( \Psi_i^{\text{target}} = \frac{1/2}{K_{i+1}} \Psi_{i+1}(\tau_{i-1}[k]) \) and \( \Phi_i^{\text{target}} = \frac{1/2+K_i}{K_{i+1}} \Psi_{i+1}(\tau_{i-1}[k]) \), so as to obtain linear updates in the state \( \Delta(t) \). The convergence of the algorithm is established in the following theorem, whose proof is omitted but is available in [14]:

**Theorem 4:** [Proportional Fairness] The algorithm in (12) converges for all initial conditions \( (\Omega_1, \Omega_2, \ldots, \Omega_n) \), except over a set of measure zero. Furthermore, by letting \( \Gamma_i(t) = \Phi_i(t) - \Psi_i(t) \pmod{1} \), and \( \Theta_i(t) = \Psi_i(t) - \Phi_{i-1}(t) \pmod{1} \), we have

\[
\lim_{t \to \infty} \Gamma_i(t) = \frac{\beta K_i}{K} \quad \text{and} \quad \lim_{t \to \infty} \Theta_i(t) = \frac{\beta}{2K},
\]

for all \( i \), where \( \beta = \frac{1}{1+\frac{1}{K}} \) and \( K = \sum_{i=1}^{n} K_i \).

The theorem indicates that the portion of time node \( i \) obtains goes towards \( \beta K_i / K \), as \( t \to \infty \). Ideally, we would like to have \( \beta = 1 \) to efficiently use the frame duration, which corresponds to leaving no gaps between different adjacent nodes’ clocks; this happens when \( n/(2K) \to 0 \). However, in order to allow external nodes to join the network, \( \beta \) must should be chosen to be less than 1; note that the space left diminishes as the sum \( K \) grows. For any \( \beta \), the ratios between the requests are preserved, i.e., the ratio between \( \beta K_i / K \) and \( \beta K_j / K \) is equal to \( K_i / K_j, \forall (i,j) \) with \( i \neq j \). That is, each node obtains, by negotiation, an amount of time proportional to its demand, namely, \( K_i \), such that the ratios of the requests are preserved.
V. COMPUTER SIMULATIONS

Round-Robin Scheduling

In Figure 5, we show the evolution of the network state according to the dynamics specified in (9) with \( n = 5 \) for different values of the coupling strength \( \alpha \). As expected, increasing the coupling strength results in a faster convergence to the fixed point \( \frac{1}{n} \). In the same figure we also report the evolution of the firing times, that converge to a constant value (modulo 1), showing that the system achieves strict desynchronization.

The parameter \( n_0 \) controls the spacing between the nodes. In order for the algorithm in (9) to converge, the condition \( n_0 \geq n \) must be satisfied, otherwise the nodes will not settle into a stable configuration. In Figure 6 we have \( n = 5 \) nodes with \( \alpha = 0.4 \), but \( n_0 = 3 < n \). We can see that, the spacing between neighbors’ firings converge to a constant value, which is not equal to \( T_f/n \). From the same figure we can observe that the time between consecutive firings, indicated by \( \Delta_f \), converges to a fixed value, even if the firing times always change, as shown on top-right.

In Figures 7, we reported the number of rounds \( R \) required to achieve \( \epsilon \)-desynchronization, with \( \epsilon = 10^{-2} \) in log-scale. As we can see from Figure 7, the DESYNCH protocol exhibits a complexity of \( n^2 \), as conjectured in [11]. The PCO-based dynamics in (9) exhibits a logarithmic complexity, while in the case of weak desynchronization it is of the order of \( \frac{1}{n} \). However the advantage of DESYNCH over the PCO-based model is that it does not require parameters, other than the coupling strength. Thus, there is a trade-off between the complexity required to obtain an estimate of the network size and the required time of convergence.

Proportional Fairness

In Figure 8, we show the performance of the proportional fair scheduling algorithm based on two local clocks, described in Section IV. In the scenario we simulated the nodes change their request \( K_i \) as time goes on. At the beginning the vector of requests is \( K = [5, 5, 5, 5, 5] \). At
iteration 200 node 1 changes its demand to $K_1 = 20$. At time 400 node 5 changes its demand to $K_5 = 20$. Finally, at iteration 700, the requests vector becomes $K = [10, 10, 10, 10, 10]$. We can observe that the network adaptively changes the portion of time for each sensor, according to the actual requests.

In Figure 9 nodes are allowed to leave or enter the system. The network is initially composed of 5 nodes, whose requests are $K = [5, 5, 5, 20, 20]$. At round 200, nodes 4 and 5 leave the network, while nodes 1, 2 and 3 remain. We can see that the period $T_f$ is shared equally here, since $K_1 = K_2 = K_3 = 5$. At round 500 node 6 joins the network, and its demand is $K_6 = 20$. The final state of the network is then given by 4 nodes: node 6 obtains the biggest amount of time, while the rest is equally divided by the remaining nodes, node 1, 2 and 3. Finally, at round 800 the requests become all equal, i.e., $K_1 = K_2 = K_3 = K_6 = 20$ and the nodes converge toward a state where they all get the same portion of time.

Proportional fairness can also be achieved using the DESYNCH protocol, adding virtual nodes in the network and replacing each node request $K_i$ with $K_i$ virtual DESYNCH clocks. Hence, the total number of virtual nodes in the network would be $\sum_{i=1}^{n} K_i$. Given that the converge time grows with $(\sum_{i=1}^{n} K_i)^2$ it is of interest to compared the DESYNCH with virtual nodes, with our proportionally fair protocol in terms of convergence time. To this end, in Figure 10, we show the number of rounds required to reach $\epsilon$-desynchronization for both DESYNCH with virtual nodes and our protocol in Section IV. In the first case we set all $K_i = 1, \forall i$ and the second case all $K_i = n, \forall i$, respectively. We can see that the PCO-based scheme is faster than DESYNCH in both cases. Moreover, from an implementation point of view, when handling widely different demands, our scheme is simpler since each node only needs to handle two clocks (instead of $K_i$ in the case of DESYNCH).

VI. CONCLUSIONS

In this paper we presented novel bio-inspired algorithms for multiplexing the transmission activities in a small network of low-cost, low-complexity devices. The emergent properties of
these protocols, inspired by the PCO model, allow a network to self-organize the activities of the nodes without the need of a master node nor global synchronization. Our results indicate that variants of the PCO algorithm can be used to achieve a TDMA scheme in a decentralized manner. Future work will focus on the implementation of such primitives on sensor platforms.

APPENDIX A

PROOF OF THEOREM 2

In this case, the firing of a node causes all the others to perform the update. We shown next that the evolution of the state vector, from round to round, can be described by an affine transformation. Consider for example, the case with three nodes, and assume that the initial offsets, at round 0, are $\phi_3(0) > \phi_2(0) > \phi_1(0)$. Note that, $\Phi_3(t_3) = 1$, $\Delta_3(t_3) = 1 - \Phi_2(t_3)$ while $\Delta_2(t_3) = \Phi_2(t_3) - \Phi_1(t_3)$ and $\Delta_1(t_3) = \Phi_1(t_3) - \Phi_3(t_3) \mod 1 = \Phi_1(t_3)$. Considering both Remark 1 and equation (8), we obtain from (4) that $\Delta_3(t_3^+) = 1 - (1 - \alpha)\Phi_2(t_3) = 1 - (1 - \alpha)(1 - \Delta_3(t_3))$ and from (5) $\Delta_2(t_3^+) = (1 - \alpha)\Phi_2(t_3) - (1 - \alpha)\Phi_1(t_3)$. Hence, when node 3 fires at time $t_3$, the state of the system becomes linear:

$$\Delta_3(t_3^+) = (1 - \alpha)\Delta_3(t_3) + \alpha$$

$$\Delta_2(t_3^+) = (1 - \alpha)\Delta_2(t_3)$$

$$\Delta_1(t_3^+) = (1 - \alpha)\Delta_1(t_3)$$

At time $t_2 > t_3$ node 2 fires, and, thus, the intervals between the nodes change as $\Delta_3(t_2^+) = (1 - \alpha)\Delta_3(t_2)$, $\Delta_2(t_2^+) = (1 - \alpha)\Delta_2(t_2) + \alpha$ and $\Delta_1(t_2^+) = (1 - \alpha)\Delta_1(t_2)$. At time $t_1 > t_2$ node 1 fires, and the state of the system becomes

$$\Delta_3(t_1^+) = (1 - \alpha)\Delta_3(t_1) = (1 - \alpha)^3\Delta_3(t_3) + (1 - \alpha)^2\alpha$$

$$\Delta_2(t_1^+) = (1 - \alpha)\Delta_2(t_1) = (1 - \alpha)^3\Delta_2(t_3) + (1 - \alpha)\alpha$$

$$\Delta_1(t_1^+) = (1 - \alpha)\Delta_1(t_1) + \alpha = (1 - \alpha)^3\Delta_1(t_3) + \alpha.$$
The evolution of the system from round $R$ to round $R+1$ is then $\Delta[R+1] = M_3\Delta[R] + v_3$, where $M_3 = (1 - \alpha)^3I$ and $v_3 = [(1 - \alpha)^2\alpha, (1 - \alpha)\alpha, \alpha]^T$. More generally, if we have $n$ nodes the system evolves as $\Delta[R+1] = M\Delta[R] + v$ where $M = (1 - \alpha)^nI$ and $v = \alpha[(1 - \alpha)^{n-1}, (1 - \alpha)^{n-2}, \ldots, (1 - \alpha), 1]^T$. Therefore, at round $R+1$

$$\Delta[R+1] = M^R\Delta[0] + \sum_{i=0}^{R-1} M^i v = M^R\Delta[0] + (I - M)^{-1} (I + M^R) v. \quad (14)$$

$M$ is a contraction, and, thus, from Eq. (14) we have $\Delta^* = \lim_{R \to \infty} \Delta[R] = (I - M)^{-1} v$. By expanding (14) we obtain

$$||\Delta[R] - \Delta^*|| = ||M^R\Delta[0] + (1 - (1 - \alpha)^n)^{-1} M^R v||$$

$$= (1 - \alpha)^n ||\Delta[0] + (1 - (1 - \alpha)^n)^{-1} v|| \leq 2(1 - \alpha)^n < \epsilon. \quad (15)$$

Condition (15) implies $R^* \leq -\frac{\log(\frac{\epsilon}{2})}{n \log(1 - \alpha)}$ (notice that this bound improves the results found in [13]). The spacing between two consecutive firings is given by $\Delta_1^* T_f = ((I - M)^{-1} v)_1 T_f = \frac{\alpha}{1 - (1 - \alpha)^n} T_f$ (we can see that only for $\alpha \to 0$, $\Delta_1^*$ converges to $1/n$).

**APPENDIX B**

**PROOF OF THEOREM 3**

In order to prove the convergence of (9), we need to show the following facts: i) there exists a unique fixed point with probability 1, given by $\Delta^* = \frac{1}{n} I$; ii) it is not possible to have $\Delta[k, i] = \Delta[k', i]$, with $k' > k$ and iii) the algorithm makes progress.

i) Suppose that $\Delta[k, i] = \Delta^*$ for some $k$ and $i$. Any pair of consecutive nodes is, thus, separated by a quantity equal to $1/n$. At the next iteration, node $i - 1$ fires. Node $i - 2$ is at a distance $1/n$ from node $i - 1$, i.e., $\Delta_{i-1}[k, i] = 1/n$. Hence, from (9) node $i - 2$ does not perform the update. On the other hand, all the remaining nodes are at a distance $k/n > 1/n$, $(k > 1)$ w.r.t. node $i - 1$. Recalling that nodes are firing in reverse order of index, we have that $\Delta[k, i - 1] = \Delta^*$. Now, other fixed points of the system exist. In fact, if we divide the unit circle into $n$ ticks, and place each node randomly onto any of those, the state of the system will not change for the
same reason. Thus, a total of $n^n$ fixed points exist. Define $F$ as the set of all fixed points of the system. $F - \{\frac{1}{n}\}$ is the set of all fixed points we do not wish to have. Suppose the network state converges an element in $F - \frac{1}{n}$. We must have that at least two consecutive nodes reach a condition where their phase difference goes to zero. So, $\lim_{k \to \infty} \Delta_i[k,j] = 0$, which implies $\lim_{t \to \infty} \Phi_{i-1}(t) = \Phi_i(t)$. But, from (9), when $\Phi_{i-1}(t) \to 1$, we have $\Phi_{i-1}(t^+) \to 1 - \frac{\alpha}{n} \neq 0$, leading to a contradiction. Therefore, with probability 1, $\Delta^* = \frac{1}{n}$ is the (unique) fixed point of the system.

ii) Suppose that, for $k$ and $i$, $\Delta_i[k,i] < 1/n$ for some $l \in L$ and $\Delta_m[k,i] > 1/n$ for some $m \in M$, where $L \cap M = \emptyset$ and $L \cup M = \{1, 2, \ldots, n\}$. At least one $\Delta_i[k,i]$ must be adjacent to $\Delta_m[k,i] = \Delta_{l-1}[k,i] > 1/n$ for some $l$ (otherwise they would be all less than $1/n$, which is impossible since $\sum_{i=1}^n \Delta_i[k,j] = 1, \forall (k,j)$). Consider the first firing node, say $l_1$, in round $k$ s.t. $\Delta_{l_1}[k,l_1 + 1] < 1/n$ and $\Delta_{l_1-1}[k,l_1 + 1] > 1/n$. Right after the firing of node $l_1$, we will have that $\Delta_{l_1}[k,l_1] > \Delta_{l_1-1}[k,l_1 + 1]$ and $\Delta_{l_1-1}[k,l_1] = \Delta_{l_1-1}[k,l_1 + 1] + \Delta_{l_1}[k,l_1 + 1] - \Delta_{l_1}[k,l_1] < \Delta_{l_1-1}[k,l_1 + 1]$. So long as $\Delta_i[k,\cdot] < 1/n$ and $\Delta_i[k,\cdot] > 1/n$, $\Delta_i[k,\cdot]$ either decreases to $1/n$ or becomes less than $1/n$, depending on the phases of the other nodes. In both cases, it is not possible for node $l_1 - 1$ to get back to a distance $\Delta_{l_1-1}[k,l_1 + 1]$ and, thus, $\Delta[k,i] \neq \Delta[k',i]$ for any $i$ and $k' > k$.

iii) Since a node cannot push its neighbors apart for more than $1/n$, if $\Delta_j[0,0] < 1/n$ for some $j$, then $\Delta_j[k,i] < 1/n, \forall (k,i)$. On the other hand, if $\Delta_j[0,0] > 1/n$, either $\Delta_j[\cdot,\cdot]$ decreases monotonically to $1/n$ or becomes less than $1/n$. We can conclude that the number of intervals bigger than $1/n$ (which is a discrete sequence of positive numbers) decreases monotonically as the algorithm proceeds, and its minimum is given by $\lim_{k \to \infty} |M||[k] = s^*$ for some $s^* \geq 1$.

Consider the first round at which this event happens, i.e., $s^*$. We can define a chain, denoted as $C_l[k]$, as a biggest sequence of consecutive intervals less than $1/n$, and $\{C_l\}$ denotes the set of chains in the network. We have that $|C_l[k]| = \text{constant}, \forall l$ and $\forall k > s^*$ Moreover, the distance between the first and last node within the same chain increases (i.e. the chain expands), from
Therefore, from round $s^\ast$ on, each chain spreads out to $|C|/n$ and, thus, the state of the
network converges to the fixed point $\frac{1}{n}1$.

APPENDIX C

PROOF OF LEMMA 1

Recall that $n_0 = n$. Consider the case where, at a given round $R$ (we shall omit the firing and
round indices and time, occasionally, for simplicity), between firings $\Delta_i < 1/n, \forall i = 2, \ldots, n$ and
$\Delta_1 > 1/n$. From Appendix B, at every round $\Delta_1$ decreases monotonically to $1/n$ and, therefore,
$1 - \sum_{i=2}^{n} \Delta_i$ increases to $1 - 1/n$. In particular, we assume that the distances $\Delta_i, (i = 2, \ldots, n)$
are such that each firing of the nodes $\{n, n-1, \ldots, 2\}$ causes only one node at a time to perform
the update. This is the case if, for example, $\alpha$ is sufficiently large. Because $\Delta(t)$ is constrained
to be on the hyperplane $\sum_{i=2}^{n} \Delta_i(t) = 1$ we can reduce the state of the network $\Delta[R]$ to a
$n - 1$ dimensional vector $\hat{\Delta}[R] = (\Delta_n, \Delta_{n-1}, \ldots, \Delta_2)^T$. The following equalities are a result of the
norm one, and constraints on $\Delta_i(t)$:

$$
\left\| \Delta - \frac{1}{n} \right\| = \left\| \hat{\Delta} - \frac{1}{n}1_{n-1} \right\| + \left( \Delta_1 - \frac{1}{n} \right) = \left\| \hat{\Delta} - \frac{1}{n}1_{n-1} \right\| + \left( 1 - \sum_{i=2}^{n} \Delta_i - \frac{1}{n} \right)
$$

$$
= \left\| \hat{\Delta} - \frac{1}{n}1_{n-1} \right\| + \left( \frac{n-1}{n} - \sum_{i=2}^{n} \Delta_i \right) = 2 \left\| \hat{\Delta} - \frac{1}{n}1_{n-1} \right\|
$$

where $1_{n-1}$ is the $n - 1$ dimensional vector of ones. Therefore, the convergence time is the
minimum number of rounds required to satisfy the condition $2 \left\| \hat{\Delta} - \frac{1}{n}1_{n-1} \right\| < \epsilon$.

Consider the network at the end of round $R$. We would like to derive the state of the network
at the end of round $R + 1$, i.e., $\hat{\Delta}[R + 1]$, as a function of $\hat{\Delta}[R]$. The same methodology used
in Appendix A can be used here to prove that the updates on the $\Delta_j(t_i)$ are affine. In particular,
Let $\Delta_i'$ denote $\Delta_i$ after an update. From (10), the first firing (node $n$) modifies variables $\Delta_n$ and
$\Delta_{n-1}$ as follows:

$$
\Delta_n' = (1 - \alpha)\Delta_n + \frac{\alpha}{n} \tag{16}
$$

$$
\Delta_{n-1}' = \Delta_n + \Delta_{n-1} - \Delta_n' = \alpha\Delta_n + \Delta_{n-1} - \frac{\alpha}{n}. \tag{17}
$$
This is because, similarly to what observed for node 3, 2, and 1 in Appendix A, $\Delta_n = 1 - \Phi_{n-1}$ and, from Remark 1, (4) $\Delta'_n = (\Phi'_n - \Phi'_{n-1}) \mod 1 = 1 - \Phi'_n - 1 - (1 - \alpha)\Phi_{n-1} - \alpha(1 - 1/n_0) = 1 - (1 - \alpha)(1 - \Delta_n) - \alpha(1 - 1/n_0)$, which proves the first equation (16). To prove the second equation, because of the initial condition $\Phi_{n-2}$ does not change; $\Delta_{n-1} = \Phi_{n-1} - \Phi_{n-2} > 0$; based on Remark 1, (5), $\Delta'_{n-1} = \Phi'_{n-1} - \Phi'_{n-2}$, because $\Delta'_n = 1 - \Phi'_n$ and also $\Delta_n = 1 - \Phi_{n-1}$, it follows that $\Delta'_{n-1} = 1 - \Delta_n - \Phi_{n-2} = 1 - \Delta_n - \Phi_{n-2} + \Phi_{n-1} = \Delta_n + \Delta_{n-1} - \Delta'_n$ and substituting (16) in it we get (17). As node $n-1$ fires, again, only node $n-2$ is going to make an update. The new $\Delta''_{n-1}$ and $\Delta'_{n-2}$ can be computed as a recursion of the previous relationships, as follows

$$\Delta''_{n-1} = (1 - \alpha)\Delta_n + (1 - \alpha)\Delta_{n-1} + \frac{\alpha^2}{n}$$

$$\Delta'_{n-2} = \Delta_n + \Delta_{n-1} + \Delta_{n-2} - \Delta'_n - \Delta''_{n-1} = \Delta_{n-2} + \alpha\Delta_{n-1} + \alpha^2\Delta_n - \frac{\alpha}{n} - \frac{\alpha^2}{n}$$

Node $n-2$ will then modify $\Delta'_{n-2}$ and $\Delta_{n-3}$ and so on and so forth, until node 1, whose firing will have no effect on the others, since $\Delta_1 > 1/n$. By algebraic manipulation, the state of the network at round $R + 1$ can be expressed as

$$\hat{\Delta}[R + 1] = M^R \hat{\Delta}[0] + \sum_{i=0}^{R-1} M^i v = M^R \hat{\Delta}[0] + (I - M)^{-1}(I + M^R)v$$

where $\{M\}_{ij} = (1 - \alpha)\alpha^{i-j}U(i-j)$ and $v = \frac{1}{\alpha}(\alpha, \alpha^2, \ldots, \alpha^{n-1})^T$. Therefore,

$$2\left\|M^R \hat{\Delta}[0] + (I - M)^{-1}(I + M^R)v - \frac{1}{n}1_{n-1}\right\| < \epsilon.$$  

Since $\{(I - M)^{-1}\}_{ij} = \frac{1}{\alpha}((1 - \alpha)\delta[i > j] + \delta[i = j])$, then $(I - M)^{-1}v = \frac{1}{\alpha}1_{n-1}$ (it can be easily verified), and condition (19) can be simplified as $2\|M^R\|\left(\|\hat{\Delta}[0]\| + \frac{1}{1 - \|M\|}\|v\|\right) < \epsilon$. Moreover, $\frac{1}{1 - \|M\|} = \alpha^{-n}$ and $\|M^R\| < (1 - \alpha)^Rn$ (we do not show these facts for the lack of space). Therefore, the minimum number of rounds to achieve $\epsilon$-desynchronization satisfies the condition $2(1 - \alpha)^Rn = \epsilon$, which implies $R^* \leq -\frac{\log 2n}{\log(1 - \alpha)}$.

REFERENCES

Fig. 1. Left: Strogatz model with concave-down $f$ and $\varepsilon < 0$. Right: PCO-based model with concave-up $f$ and $\varepsilon > 0$.

Fig. 2. Update in PCO-Based Round-Robin, with $n = 5$ nodes. Left: $n_0 = n$. Right: $n_0 = 1$. 
Fig. 3. Example of weak desynchronization with \( n = 2 \) nodes. Even if the number of rounds required to achieve a given accuracy goes to zero as the number of nodes increases, the firing times always shift in time because each firing event causes the other nodes to push their phase backward. Weak desynchronization is illustrated in Figure 3 where it is shown a system composed of two weakly desynchronized nodes in steady-state. The fixed point is given by \( \Delta^* = (\Delta^*_2, \Delta^*_1)^T = (1-(1-\alpha)^2)^{-1}((1-\alpha)\alpha, \alpha) \).

At time \( t_1 \) (right before node 1 fires) the distances between nodes 1 and 2 are \( \Delta_1(t_1) = \Delta^*_2 \) and \( \Delta_2(t_1) = 1 - \Delta_1(t_1) = \Delta^*_1 \), respectively. Node 1, then, fires, causing node 2 to step back in phase. Thus, at time \( t_1^+ \) the distance between node 1 and 2 is \( \Delta_1(t_1^+) = \Delta^*_1 \), while \( \Delta_2(t_1^+) = 1 - \Delta_1(t_1^+) = \Delta^*_2 \). At time \( t = t_2 \) node 2 reaches \( \Phi_2(t) = 1 \). Right after its firing, node 1 jumps back in phase, and, thus, at time \( t_2^+ \) we have that \( \Delta_2(t_2^+) = \Delta^*_1 \) and, therefore, \( \Delta_1(t_2^+) = 1 - \Delta_2(t_2^+) = \Delta^*_2 \). Weakly desynchronization is a state where right after the firing of a node, the state of the network (i.e., the distances between the nodes) is the same with respect to the firing node (note that only for \( \alpha \to 0 \) one can obtain a constant separation of the pulses equal to \( T_f/n \)).
Fig. 4. Update of node $i$, caused by the firing of node $i - 1$. Node $i$ pushes its phase variables towards $\Phi_{i}^{\text{target}}$ and $\Psi_{i}^{\text{target}}$.

Fig. 5. Evolution of a network with $n = 5$ nodes and $n_0 = n$ ($\alpha = 0.1$ [orange], $\alpha = 0.5$ [red], $\alpha = 0.9$ [blue]). On top-right are also shown the firing times (mod 1).
Fig. 6. Center: Evolution of a network with $n = 5$ nodes, $n_0 = 3$ and $\alpha = 0.4$. Top left: $\Delta_f$ over the time. Top right: firing times.

Fig. 7. Number of rounds $R$ required to achieve an accuracy $\epsilon = 10^{-2}$ for (i) Desynch with $\alpha = 0.9$, (ii) PCO with $n = n_0$ and $\alpha = 0.75$, and (iii) PCO-based with $n_0 = 1$ and $\alpha = 0.1$. 
Fig. 8. Evolution of $\Phi_i$ and $\Psi_i$, with respect to $\Psi_1$, when the objective vector $K$ changes over the time.

Fig. 9. Evolution of $\Psi_i(t)$ and $\Phi_i(t)$ with respect to $\Psi_1(t)$ when nodes leave or join the network, and the requests change over the time.
Fig. 10. Number of rounds required to reach $\epsilon$-desynchronization for DESYNCH and our PCO-based protocol, with $\epsilon = 10^{-2}$ and $\alpha = 0.9$ in both cases.