On the Improvement of a Self-Organized MAC Protocol for Multi-Hop Wireless Sensor Networks

Clemens Mühlberger
Department of Computer Science
University of Würzburg
97074 Würzburg, Germany
Email: muehlberger@informatik.uni-wuerzburg.de

***Abstract*—Biologically inspired self-organization methods can help to manage the access control to the shared communication medium of wireless ad-hoc networks. One lightweight method is the primitive of *desynchronization*, which has already been implemented as MAC protocol for single-hop topologies successfully: periodically transmitting nodes establish a collision-free TDMA schedule autonomously. However, multi-hop topologies are more realistic, but each node now requires knowledge about its two-hop neighborhood to solve the *hidden terminal problem*.**

In this paper we describe our experience using an extended version of a MAC protocol for multi-hop topologies based on the primitive of desynchronization. We identified *stale information* as pitfall of desynchronization when using phase shift propagation to solve the hidden terminal problem in multi-hop topologies. As a result, we included a refractory threshold to manage stale information at the self-organized MAC protocol based on the primitive of desynchronization for multi-hop topologies.

***Index Terms*—desynchronization; refractory threshold; self-organization; wireless sensor network; multi-hop topology**

I. Introduction

This section introduces the primitive of desynchronization as MAC protocol for single-hop topologies, which will be the core for the following sections. Based on the first mathematical model of pulse-coupled oscillators from Mirollo and Strogatz [5], Degesys et al. [4] introduced the biologically inspired primitive of *desynchronization*: it implies that each node "oscillates" at the same frequency $f = 1/T$. Applied to the domain of wireless sensor networks, each node tries to transmit a so called *firing packet* after every period T. Such periodical data transmissions are common, e.g., in biomedical sensor networks due to periodic sensor sampling (cf. [9]). For single-hop topologies, desynchronization results in the temporally equidistant transmission of firing packets: if such a network consists of a set N of nodes, the time span between successively transmitting nodes equals $T/|N|$.

Using the primitive of desynchronization, Degesys et al. implemented the self-organized MAC protocol DESYNC [4] for single-hop topologies. Each participating node can estimate its time of transmission within a single-hop network autonomously. Therefore, each node possesses a unique identifier[1] i, and – as already mentioned before – each node periodically transmits its firing packet.

[1]For the sake of simplicity, we do not further distinguish between the identifier itself and the node's ordinal in N. Moreover, without loss of generality let $1 \leq i \leq |N|$.

Let t_i be the current time of firing of node $i \in N$, and let t_i^+ be its next time of firing. When node i finishes its period, it broadcasts its firing packet, resets its phase, and updates t_i^+. The phase shift $\phi_i(t) \in [0, T)$ of a node i denotes the elapsed time since its current firing t_i and the given point in time t, normalized to the period T as

$$\phi_i(t) = (t - t_i) \bmod T. \tag{1}$$

Let $N_1(i)$ be the set of *one-hop neighbors*, and let $N_2(i)$ be the set of *two-hop neighbors* of node i. Please note that $\{i\}$, $N_1(i)$, and $N_2(i)$ are pairwise disjoint. When node i is receiving a firing packet of its one-hop neighbor $j \in N_1(i)$, node i is able to calculate the phase shift $\phi_i(t_j)$ towards this one-hop neighbor j using its time of reception t_j in (1). For example, $\phi_i(t_j) = 0.5 \cdot T$ means that node i has finished half of its current period when it received the firning packet from node j at time t_j. Two neighbors of node i are of special interest: the previous phase neighbor $p(i) \in N$ (*predecessor*) broadcasts its firing packet just before node i, whereas the successive phase neighbor $s(i) \in N$ (*successor*) broadcasts its firing packet just after node i.

The primitive of desynchronization forces each node to transmit its firing packet at a maximum temporal distance towards both phase neighbors, i.e., each node attempts to achieve the midpoint of its phase neighbors. Therefore, observing the firing packets of its phase neighbors, each node i is able to calculate the corresponding phase shifts $\phi_i(t_{s(i)})$ and $\phi_{p(i)}(t_i)$ as well as its *adjustment factor* ε_i as

$$\varepsilon_i = \frac{\phi_i(t_{s(i)}) - \phi_{p(i)}(t_i)}{2}. \tag{2}$$

Finally, node i sets its next time of firing t_i^+ immediately after transmitting its own firing packet at time t_i as

$$\begin{aligned} t_i^+ &= t_i + T + \alpha \cdot \varepsilon_i \\ &= t_i + (1 - \alpha) \cdot T + \alpha \cdot (\varepsilon_i + T). \end{aligned} \tag{3}$$

The *jump size parameter* $\alpha \in (0, 1)$ regulates how fast[2] node i moves toward the assumed midpoint between its phase neighbors $p(i)$ and $s(i)$. The last expression of (3) shows its similarity to the exponentially weighted moving average, which smooths out short-term fluctuations but highlights long-term trends.

[2]Using $\alpha = 0$ means no movement, $\alpha = 1$ means no damping at all.

II. Stale Information Problem

The primitive of desynchronization aims for a self-organized but collision-free arrangement of time slots. Hence, the nodes are able to rely on just locally available information. Indeed, received data from adjacent nodes sometimes is ”stale”, i.e., information obtained from received firing packets is obsolete at the time of their application, and thus unreliable or even invalid. This problem already exists in single-hop topologies but it is intensified in multi-hop topologies.

A. Single-Hop Topologies

While node i uses (3) to calculate its next time of firing t_i^+ immediately after transmitting its firing packet at time t_i, both its phase neighbors may already have adjusted their individual next time of firing autonomously. Therefore, the formerly measured phase shifts $\phi_i\left(t_{s(i)}\right)$ and $\phi_{p(i)}\left(t_i\right)$ might be stale. This problem is inherent in single-hop (and multi-hop) topologies, and was first mentioned by Degesys et al. [4].

Patel et al. [8] further examined this problem for single-hop topologies, and – with regard to the jump size parameter α – they also proved the convergence of their more robust variant of the DESYNC algorithm for single-hop topologies: The use of more recent data for the phase shift $\phi_i\left(t_{s(i)}\right)$ in (3) will omit one unreliable information. For this purpose, node i has to calculate its next time of firing t_i^+ not immediately after the transmission of its own firing packet, but immediately after the reception of the first subsequent firing packet of its successor $s(i)$. As a result, node i uses more recent data, but the equation to compute the next time of firing t_i^+ remains the same. Just the time when the next time of firing is calculated changes from t_i to $t_{s(i)}$.

B. Multi-Hop Topologies

According to Degesys and Nagpal [3], a node additionally requires knowledge about its two-hop neighborhood to enable a collision-free communication within multi-hop topologies, still being consistent with the primitive of desynchronization. One approach is the phase shift propagation as proposed and implemented by Mühlberger and Kolla [7] for the EXTENDED-DESYNC algorithm. Here, each node broadcasts the knowledge about its one-hop neighborhood along with every firing packet. Therefore, each node i additionally is able to take care of its two-hop neighbors. That means, node i arranges itself according to the firings of all known nodes $j \in N_1(i) \cup N_2(i)$. This is the reason why phase neighbors can be two-hop neighbors as well. Due to the phase shift propagation, node i gains information about its two-hop neighbor $k \in N_2(i) \cap N_1(j)$ just in cooperation with a one-hop neighbor $j \in N_1(i)$. This data flow is additionally delayed by at least the phase shift $\phi_k\left(t_j\right)$ between nodes j and k.

To exemplify the impact of stale information in multi-hop topologies using phase shift propagation, we simulated the EXTENDED-DESYNC algorithm on a small but manageable scenario. For this purpose, we used a self-developed simulator on an Intel Core i5-2540M CPU with 2.60 GHz and 8.00 GB main memory under Windows 7 Professional 64 Bit.

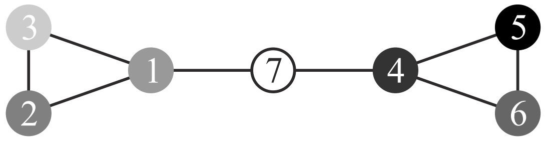

Fig. 1. Topology M_7 consists of the set $N = \{1, \ldots, 7\}$ of nodes.

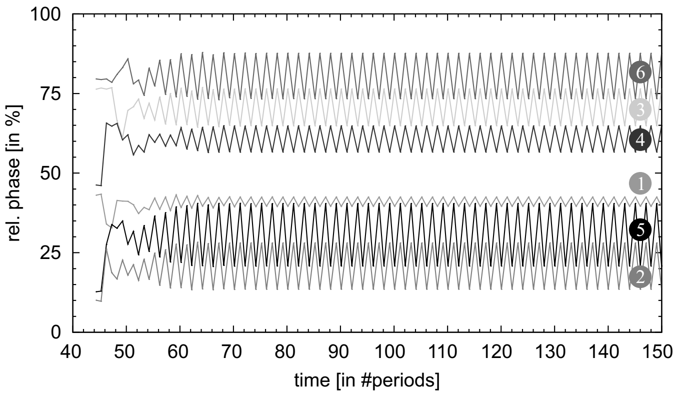

Fig. 2. Simulation of M_7 (about 110 periods since the start up of node 7 at period 45), $\alpha = 0.95$, ($\rho = 0$), point of view: node 7.

First, we assume idealized conditions, i.e., all communication links are symmetrical and reliable, not any node will fail, and there is no clock drift. According to Degesys et al. [4], we set $\alpha = 0.95$ as damping factor. The observed topology M_7 consists of the set $N = \{1, \ldots, 7\}$ of nodes as shown in Fig. 1. This topology contains two cyclic (and complete) sub-graphs $C_3 = \{1, 2, 3\}$ and $C_3' = \{4, 5, 6\}$. Let the nodes of both disjoint single-hop topologies C_3 and C_3' start first. Therefore, both sub-graphs will desynchronize, but are unaware of each other. When node 7 joins the network, it instantly gains knowledge of both topologies C_3 and C_3'. With its first firing packet containing its one-hop neighborhood (i.e., nodes 1 and 4), node 7 connects C_3 with C_3' and thus completes the topology M_7. Figure 2 shows the first 100 periods since the start up of node 7 at period 45 from its point of view. Due to the stale information in this multi-hop topology, the one-hop and two-hop neighbors of node 7 (i.e, all nodes of C_3 and C_3') rather diverge than converge, as intended by the primitive of desynchronization. In fact, approximately 20 periods after the start up of node 7, the time of transmission of each node oscillates with a constant but individual amplitude. Besides, the phase neighbors of node 7 are its two-hop neighbors node 6 and node 2. Therefore, to diminish the impact of stale information in multi-hop topologies, it is not sufficient to calculate the next time of firing t_i^+ after the reception of successor’s firing packet.

III. Solution for Stale Information Problem

In Section II, we analyzed the problem of stale information. As already mentioned, this problem is inherent in the primitive of desynchronization. For single-hop topologies it is sufficient for a node i to calculate its next time of transmission t_i^+ only after receiving the firing packet of its successor $s(i)$ (cf.

Section II-A). Therefore, we will concentrate on multi-hop topologies in this section. However, we can not avoid stale information at all, but with our new approach we want to take control of its evolution and reduce its impact.

A. Refractory Threshold

In multi-hop topologies, the effect of stale information is intensified due to the delayed propagation of information about two-hop neighbors (cf. Section II-B). To some extent, our approach follows the *law of similars*, because we suggest to intentionally delay the adjustment of a node's next time of firing. Therefore, we introduce an additional *refractory threshold* $\rho \in [0,1]$ along with a continuous random variable $X_i \in [0,1]$ following the continuous uniform distribution $\mathcal{U}(0,1)$. According to the random variable X_i and the arbitrary refractory threshold ρ, the adjustment factor ε_i will be considered, or not. Therefore, node i will set its next time of firing t_i^+ as

$$t_i^+ = \begin{cases} t_i + T + \alpha \cdot \varepsilon_i & \rho < X_i \qquad \text{(4a)} \\ t_i + T & \text{otherwise.} \qquad \text{(4b)} \end{cases}$$

Obviously, choosing $\rho = 0$ lets the nodes always adjust their time of firing, which results in the same behavior as observed in Section II-B without any refractory threshold. In contrast, choosing $\rho = 1$ is useless, since a node will not use its adjustment factor according to (2) for its next time of firing anymore.

In some sense, the refractory threshold ρ contradicts the primitive of desynchronization, because it "skips" the adjustment of the next time of firing using (2). However, it allows a node to keep its phase (and thus its time of firing) with a probability of ρ. This behavior helps the system to converge: Let node i be phase neighbor of another node $j \in N_1(i) \cup N_2(i)$. First of all, node j in return needs not to be phase neighbor of node i in multi-hop topologies (cf. [6]). Moreover, if node i skips the adjustment of its phase using (4b), node j's estimation of its next time of firing remains valid regarding the phase shift towards the skipping node i. The information about node i is still reliable.

B. Simulation Results

We will exemplify the impact of our new threshold on the simple scenario from Section II-B, where the two disjoint single-hop topologies C_3 and C_3' are combined by node 7 (cf. Fig. 1). Again, we use the self-developed simulator on the same computer as mentioned in Section II-B. Once more, we set $\alpha = 0.95$, but now each node calculates its next time of firing according to (4) using $\rho = 0.25$. That means, on average each node keeps its phase at every fourth period. In contrast to the oscillating time of transmission of each node (cf. Fig. 2), this rather low refractory threshold helps the network to converge after about 25 periods since the start up of node 7 (cf. Fig. 3). Again, the phase neighbors of node 7 are its two-hop neighbors node 6 and node 2 (cf. II-B).

However, a higher refractory threshold slows down the convergence rate of the whole system: in comparison to

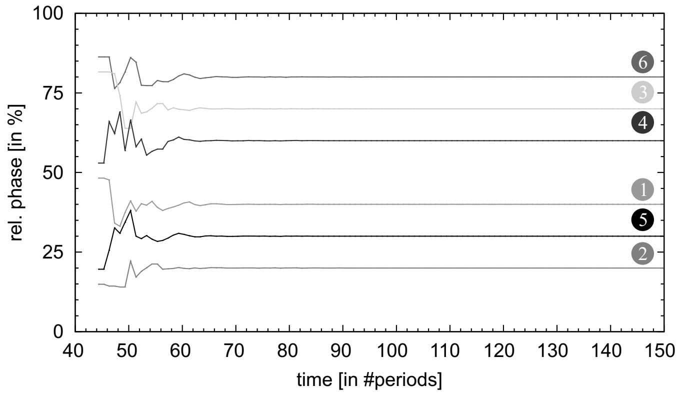

Fig. 3. Simulation of M_7 (about 110 periods since the start up of node 7 at period 45), $\alpha = 0.95$, $\rho = 0.25$, point of view: node 7.

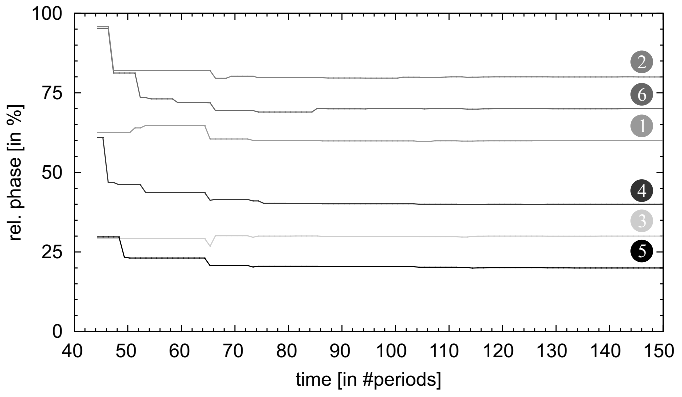

Fig. 4. Simulation of M_7 (about 110 periods since the start up of node 7 at period 45), $\alpha = 0.95$, $\rho = 0.9$, point of view: node 7.

the scenario described above, we just raised the refractory threshold to $\rho = 0.9$. That means, adjustments of the nodes will be rarely skipped. The simulation result is shown in Fig. 4: the network is approximately desynchronized after about 50 periods since the start up of node 7. In comparison to the previous simulation results, the phase neighbors of node 7 have changed in its two-hop neighbors node 2 and 5.

However, if the refractory threshold is too low, the system may still rather diverge than converge. For instance, if we set $\rho = 0.1$ at the same scenario from above, the time of transmission of each node again fluctuates, but with a smaller amplitude (cf. Fig. 5). The simulation results so far exemplify the capability of our refractory threshold. However, the refractory threshold ρ must be set carefully in combination with the jump size parameter α (cf. Section V).

IV. Related Work

In the previous sections, we already referred to some work regarding the primitive of desynchronization. Therefore, this section describes further work dealing with stale information.

A. Refractory Period

In Section III, we introduced our refractory threshold ρ to handle obsolete and thus unreliable data from neighbor nodes.

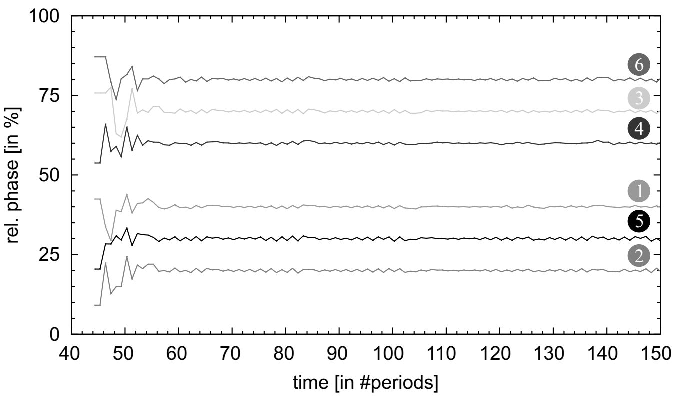

Fig. 5. Simulation of M_7 (about 110 periods since the start up of node 7 at period 45), $\alpha = 0.95$, $\rho = 0.1$, point of view: node 7.

Therefore, a node probabilistically skips the adjustment of its next time of transmission to obtain more reliable data. Similar to our approach is the so called *refractory period*, suggested by Degesys, Basu, and Redi [2] to synchronize (not to desynchronize, as we do) strongly pulse-coupled oscillators: if an oscillator receives the firing of a neighbor within the refractory period, the receiving oscillator does not process this incoming firing. That means, the phase shift between sender and receiver is too short and thus, the receiver temporarily does not adjust its next time of firing. Moreover, the phase shift between two oscillators specifies, whether an oscillator skips the adjustment of its next time of firing, or not. In contrast, our refractory threshold is probabilistic and independent from the phase shift between two nodes.

B. Artificial Force Field

Another approach to desynchronize a single-hop network is presented by Choochaisri et al. [1]. Their DWARF algorithm reduces the impact of erroneous information from phase neighbors: The next time of firing of a node i does not only depend on the firings of its phase neighbors, instead, the next time of firing is specified by an artificial force field which is defined by all other nodes. Each force is weighted by the phase shift of the corresponding neighbor node towards the adjusting node i. This approach circumvents stale information and is very efficient for single-hop topologies. It also results in the equal time span $T/|N|$ between successively transmitting nodes. However, to the best of our knowledge, an extension for multi-hop topologies is currently missing.

V. Conclusion and Outlook

In this paper we introduced the biologically inspired primitive of desynchronization as MAC protocol for wireless sensor networks. The resulting self-organized protocol for single-hop as well as for multi-hop topologies has to manage the inherent problem of stale information. Due to this problem, the periodical transmission times of nodes may fluctuate in a multi-hop topology. Therefore, we introduced the refractory threshold ρ. According to this threshold, and contrary to the primitive of desynchronization, each node is now able to probabilistically skip the adjustment of its next time of firing. Based on some sample scenarios, we demonstrated the impact of our approach for a small but yet manageable multi-hop topology. As a result, our approach helps to damp this fluctuation: the time of transmission of each node will converge and thus the whole system will desynchronize.

Our future work will be mainly dedicated to the refractory threshold: first, we want to discover an optimal combination of the probabilistic refractory threshold ρ and the jump size parameter α. Next, we want to analyze the convergence behavior of several scenarios if our threshold depends on certain topological factors, e.g., the nodes' degree. Moreover, we are currently implementing our new algorithm on wireless sensor nodes, however an analysis under real-world conditions of this implementation is yet missing. In particular, these real-world conditions include asymmetrical as well as unreliable links, clock drifts, and erroneous nodes.

References

[1] S. Choochaisri, K. Apicharttrisorn, K. Korprasertthaworn, P. Taechalert-paisarn, and C. Intanagonwiwat, “Desynchronization with an Artificial Force Field for Wireless Networks,” *SIGCOMM Comput. Commun. Rev.*, vol. 42, no. 2, pp. 7–15, Apr. 2012.

[2] J. Degesys, P. Basu, and J. Redi, “Synchronization of strongly pulse-coupled oscillators with refractory periods and random medium access,” in *Proceedings of the 2008 ACM symposium on Applied computing*, ser. SAC ’08. New York, NY, USA: ACM, 2008, pp. 1976–1980.

[3] J. Degesys and R. Nagpal, “Towards Desynchronization of Multi-hop Topologies,” in *SASO*. Venice, Italy: IEEE Computer Society, 2008, pp. 129–138.

[4] J. Degesys, I. Rose, A. Patel, and R. Nagpal, “DESYNC: Self-Organizing Desynchronization and TDMA on Wireless Sensor Networks,” in *IPSN*, Tarek F. Abdelzaher, Leonidas J. Guibas, and Matt Welsh, Eds. Cambridge, MA, USA: ACM, 2007, pp. 11–20.

[5] R. E. Mirollo and S. H. Strogatz, “Synchronization of Pulse-Coupled Biological Oscillators,” *SIAM Journal on Applied Mathematics*, vol. 50, no. 6, pp. 1645–1662, 1990.

[6] C. Mühlberger, “Desynchronization in Multi-Hop Topologies: A Challenge,” in *9. GI/ITG KuVS Fachgespräch Drahtlose Sensornetze*, R. Kolla, Ed. Würzburg, Germany: Universität Würzburg, Institut für Informatik, Sep. 2010, pp. 21–24.

[7] C. Mühlberger and R. Kolla, “Extended Desynchronization for Multi-Hop Topologies,” Institut für Informatik, Universität Würzburg, Tech. Rep. 460, Jul. 2009.

[8] A. Patel, J. Degesys, and R. Nagpal, “Desynchronization: The Theory of Self-Organizing Algorithms for Round-Robin Scheduling,” in *SASO*. Cambridge, MA, USA: IEEE Computer Society, 2007, pp. 87–96.

[9] S. Støa and I. Balasingham, “Periodic-MAC: Improving MAC Protocols for Biomedical Sensor Networks Through Implicit Synchronization,” in *Biomedical Engineering Trends in Electronics, Communications and Software*, A. N. Laskovski, Ed. Janeza Trdine 9, 51000 Rijeka (Croatia): InTech, Jan. 2011, ch. 26, pp. 507 – 522.