A resilient synchronization algorithm for wireless ad hoc and sensor networks based on distributed discrete-time PLLs

O. Simeone[1](*), U. Spagnolini[2] and Y. Bar-Ness[1]
[1]CWCSPR, Newark, New Jersey 07102-1982, USA
[2]DEI, Politecnico di Milano, Milan, I-20133 Italy
e-mail: osvaldo.simeone@njit.edu, spagnoli@elet.polimi.it barness@yegal.njit.edu

***Abstract*—Time synchronization is a critical feature of wireless ad hoc/ sensor networks in a variety of application. Distributed time synchronization via pulse-coupled oscillators is currently being investigated as a valid alternative to traditional packet-based techniques. This paper addresses the issue of resilience of such schemes in an adversarial or emergency situation with malicious or malfunctioning nodes, by focusing on a system of distributed discrete-time phase locked loops (PLLs). After showing the sensitivity of the basic system to attacks or faults, a resilient algorithm is proposed that is shown by numerical simulations to be robust to the activity of malicious nodes.**

I. Introduction

An increasing number of applications in wireless ad hoc and sensor networks require the participating nodes to share a common time scale. Different synchronization conditions can be of interest, ranging from the availability of a common notion of absolute time (e.g., for distributed tracking applications) to the establishment of period oscillations with the same frequency and, possibly, phase (e.g., for medium access control) [4]. Here we focus on the latter case by recognizing that, if clocks with same frequency and phase can be attained, a common absolute time can be achieved as well, by appropriately initializing the system through, e.g., the transmission of a beacon signal.

Two main distributed approaches to synchronization have been considered. Traditional packet-based methods prescribe the exchange of packets between different nodes using either point-to-point [2] or broadcast [3] connections. The main source of errors for packet-based techniques is the non-determinism of network dynamics due to the random delays associated with: (*i*) construction of a packet, (*ii*) queuing at the MAC layer, (*iii*) propagation and (*iv*) processing of the packet at the receiver side. Different techniques are designed to mitigate the effects of these random factors according to diverse principles (see [4] for an overview). The state of the art reports synchronization accuracies of the order of milliseconds to microseconds [2] [3]. However, common to all packet-based methods is the need for the exchange of a large number of packets, which in turns entails large computational complexity, energy expenditure and poor scalability.

In order to obviate to the drawbacks of packet-based solutions, more recently, there has been some interest in physical layer-based schemes, where synchronization takes place through the exchange of pulses either in an overlay system such as UWB or in a dedicated bandwidth [5] [6] [1]. The methods are scalable, since the operations performed at each node are independent on the number of nodes available in the network, and they have limited complexity, requiring only simple processing at the baseband level.

As discussed in [7], distributed synchronization schemes assume a benign environment and cannot survive the presence of malfunctioning nodes in an emergency situation or malicious attacks in hostile scenarios (see [8] for an overview of security issues in wireless networks). The recent work [7] (see also references therein) addresses the problem of security and resilience for packet-based synchronization schemes. The goal of this paper is to tackle the same problem for physical layer-based synchronization schemes. In particular, we focus on the system of distributed discrete-time phase locked loops (PLLs) presented in [1], that extends the synchronization algorithms of [11] [12] (which in turn can be seen as special cases of general consensus algorithms, see, e.g., [13]). An important reference is also [14], where stability of a system of distributed analog (and continuously coupled) PLLs is studied. A novel approach is herein proposed that modifies the basic system considered in [1] towards the goal of achieving resilience to the activity of malfunctioning or malicious nodes. Effectiveness of the proposed technique is validated by numerical simulations.

II. Distributed discrete-time PLLs with malicious nodes

Let us consider a set $\mathcal{K}$ of $K = |\mathcal{K}|$ wireless nodes located in a given area on the plane. A subset $\mathcal{K}_c \subseteq \mathcal{K}$ of K_c nodes collaborate with each other through the exchange of pulses towards the goal of achieving synchronization [5] [1], whereas the remaining set $\mathcal{K}_m$ of $K-K_c$ nodes is malicious or malfunctioning (see fig. 1-a). Each node has its own local free-oscillation frequency $1/T_k$ for $k = 1, ..., K$. The local clocks are defined by a discrete-time function $t_k(n)$, that evolve as[1]

$$t_k(n) = nT_k + \theta_k(n), \tag{1}$$

[1]Clock inaccuracies other than frequency offset are not considered here. We refer to [14] for the analysis of distributed analog PLLs in presence of a more realistic clock model.

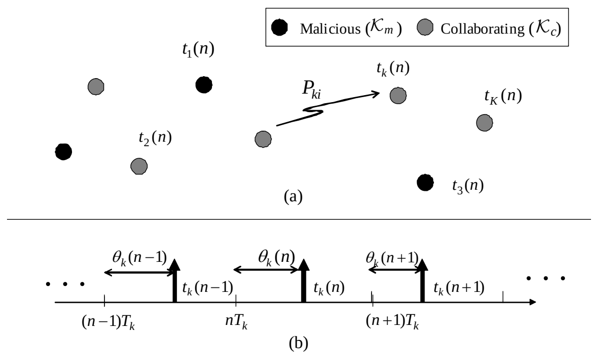

Fig. 1. (a) Network of pulse-coupled clocks: $\mathcal{K}_m$ is the set of malicious nodes and $\mathcal{K}_c$ the set of node collaborating towards the goal of synchronization; (b) Illustration of the clock of node k.

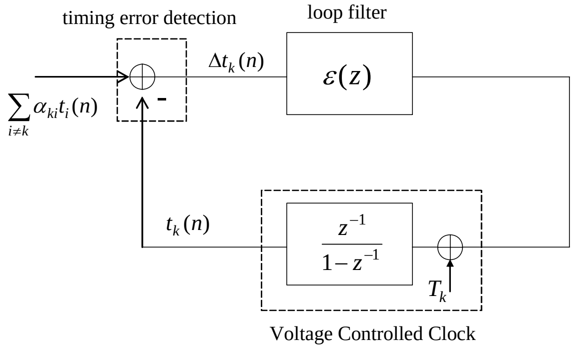

Fig. 2. Discrete-time PLL run by each node.

where index $n = 0, 1, 2, ...$ runs over the periods of the clock ($t_k(0)$ or equivalently $\theta_k(0)$ represents the initial condition) and $0 \leq \theta_k(n) < T_k$ is the instantaneous phase with respect to the local frequency $1/T_k$ (see fig. 1-b). In case the collaborating nodes are isolated $\theta_k(n) = \theta_k(0)$ and the nodes are not synchronized unless $T_k = T$ and $\theta_k(0) = \theta$ for every $k \in \mathcal{K}_c$. Notice that model (1) assumes that all the nodes are active at time $n = 0$. However, in case a given number of nodes is added at a later stage to the network, it is enough to redefine accordingly the initial time instant in order to study the convergence properties of the new network configuration.

Two synchronization conditions are of interest. We say the K_c cooperating clocks are *frequency* synchronized if

$$t_k(n+1) - t_k(n) = T \tag{2}$$

for each $k \in \mathcal{K}_c$ and for sufficiently large n, where $1/T$ is the common frequency. A more strict condition requires full *frequency and phase* synchronization, i.e.,

$$t_i(n) = t_j(n) \tag{3}$$

for $i, j \in \mathcal{K}_c$ and for n sufficiently large.

In order to couple the clocks towards the aim of achieving synchronization, each node transmits pulses at times $t_k(n)$ in (1). The topology of the network determines the power received by any kth node from the ith as

$$P_{ki} = \frac{C_{ki}}{d_{ki}^{\gamma}}, \tag{4}$$

where C_{ki} is an appropriate constant (accounting for possible fading and shadowing), $d_{ki} = d_{ik}$ is the distance between the nodes and γ is the path loss exponent ($\gamma = 2 \div 4$). Notice that distances d_{ki} and channels C_{ki} are assumed to be constant throughout the synchronization process. While malicious nodes in set $\mathcal{K}_m$ select their instantaneous phases $\theta_k(n)$ in (1) either randomly or according to some jamming rule, the collaborating nodes in set $\mathcal{K}_c$ update their instantaneous phases $\theta_k(n)$ according to the discrete-time PLL in fig. 2.

In the rest of this section, we first give a brief review of the synchronization technique based on distributed discrete-time PLLs first studied in [1], along with a discussion on the effect of propagation delays that was not included in [1] (Sec. II-A and II-B). We then recall a convergence result proved in [1] for the case where all the nodes collaborate ($K = K_c$), that shows to be a natural extension of well-known results in the context of point-to-point PLLs [9] (Sec. II-C).

A. Basic discrete-time PLLs

As shown in fig. 2, at the kth collaborating node ($k \in \mathcal{K}_c$) and for each nth period, a timing error detector calculates the time differences between the local clock and the timing of the received pulses from the other nodes (see below on how to handle propagation delays):

$$\delta t_{ki}(n) = t_i(n) - t_k(n) \text{ for } i \neq k, \tag{5}$$

and then computes a convex weighted sum of the time differences $\{\delta t_{ki}(n)\}_{i \neq k}$:

$$\Delta t_k(n) = \sum_{i=1, i\neq k}^{K} \alpha_{ki} \cdot \delta t_{ki}(n), \tag{6}$$

with $\alpha_{ki} \geq 0$ and $\sum_{i=1,i\neq k}^{K} \alpha_{ki} = 1$. Notice that the timing error detector defined by (5)-(6) does not require each node to be able to detect the pulses received from all the other nodes (i.e., does not require a fully connected network). On the contrary, by appropriately choosing the parameters α_{ki}, one can give non-zero weight only to the detected pulses. A specific choice that mimic this constraint is the one proposed by [11] [12], whereby we have:

$$\alpha_{ki} = \frac{P_{ki}}{\sum_{j=1,j\neq k}^{K} P_{kj}}, \tag{7}$$

so that each time difference $\delta t_{ki}(n)$ is weighted proportionally to the received power of the corresponding pulse. In practice, it is shown in [1] that a simple instantaneous estimators of the powers P_{ki} based on the square of the discrete-time received samples is enough to achieve satisfactory performance over a bandlimited Gaussian channel. In the following we will use the model (7), referring to [1] for implementation details.

The output of the timing error detector $\Delta t_k(n)$ is fed to a first-order loop filter $\varepsilon(z) = \varepsilon_0/(1-\mu z^{-1})$, whose output drives the local Voltage Control Clock (VCC)

$$t_k(n+1) = t_k(n)+\varepsilon_0\cdot\Delta t_k(n)+\mu(t_k(n)-t_k(n-1))+(1-\mu)T_k. \tag{8}$$

Assuming first-order loop filters $\varepsilon(z)$ amounts to considering second-order discrete-time PLLs, according to the conventional denomination. It should be also noticed that the update rule (8) with $T_k = 0$ and $\mu = 0$ was studied in the literature on consensus of network of agents [13] [15].

B. Synchronization in presence of propagation delays

Let q_{ij} be the propagation delay between the ith and the jth node (by symmetry, we have $q_{ij} = q_{ji}$). The time at which the nth pulse emitted by node i (at time $t_i(n)$) is recorded by the kth is $t_i(n)+q_{ki}$. It follows that the timing detection error of the kth PLL in fig. 2 measures

$$\sum_{i=1,i\neq k}^{K} \alpha_{ki}\cdot(t_i(n)+q_{ki}-t_k(n)) = \Delta t_k(n) + \sum_{i=1,i\neq k}^{K} \alpha_{ki}q_{ki}, \tag{9}$$

which differs with the basic scheme discussed above by the constant term $\sum_{i=1,i\neq k}^{K} \alpha_{ki}q_{ki}$. Therefore, defining

$$\tilde{T}_k = T_k + \frac{\varepsilon_0}{1-\mu}\sum_{i=1,i\neq k}^{K} \alpha_{ki}q_{ki}, \tag{10}$$

it is easy to see that the update rule (8) holds even in the case of propagation delays by substituting T_k with $\tilde{T}_k$ in (10). Therefore, the presence of propagation delays has the equivalent effect of an additional term in the period of the local clocks.

C. Convergence of distributed discrete-time PLLs in absence of malicious nodes [1]

Reference [1] studies the convergence of the system of discrete-time PLLs (8) in case there are no malicious nodes ($K = K_c$). In the following, we briefly review this result for reference. Let us denote a possible common value for the frequency of all nodes as $1/T$ (to be determined), i.e., $t_k(n) - t_k(n-1) = T$ for sufficiently large n, so that the clock of the kth sensor can be written (for large n) as

$$t_k(n) = nT + \tau_k(n), \tag{11}$$

where $\tau_k(n)$ denotes the relative phase with respect to the common frequency $1/T$. Moreover, let us define the vector $\boldsymbol{\tau}(n) = [\tau_1(n) \ \cdots \tau_K(n)]^T$.

As discussed in, e.g., [6] [13], the convergence of various synchronization algorithms can be related to the properties of the graph that describe the interconnections among different clocks. In our scenario, a weighted directed graph $\mathcal{G}(V, E, \mathcal{A})$ can be constructed with vertices V given by the K nodes and set of weights $\mathcal{A}$ of the edges in E given by the parameters α_{ij}. In particular, the weight of the edge between node j and node i is given by α_{ij}. Related algebraic quantities are the $K\times K$ Laplacian matrix of the network $\mathbf{L}$, defined as $[\mathbf{L}]_{ii} = \sum_{j\neq i}\alpha_{ij}$ and $[\mathbf{L}]_{ij} = -\alpha_{ij}$, $i\neq j$ and the system matrix $\mathbf{A} = \mathbf{I}-\varepsilon_0\mathbf{L}$.

It is shown in [1] that, if the gain ε_0 and the pole μ are small enough, and the graph $\mathcal{G}$ is strongly connected[2], then the system (8) synchronizes the clocks of the K nodes to the common period

$$T = \mathbf{v}^T\mathbf{T}, \tag{12}$$

where $\mathbf{v}$ is the normalized left eigenvector of matrix $\mathbf{A}$ corresponding to eigenvalue 1 ($\mathbf{A}^T\mathbf{v} = \mathbf{v}$ with $\mathbf{1}^T\mathbf{v} = 1$). However, under the same assumptions, the timing phases $\boldsymbol{\tau}(n)$ remain generally mismatched and given for $n\to\infty$ by

$$\boldsymbol{\tau}(n)\to\boldsymbol{\tau}^* = \mathbf{1}\cdot\eta+(1-\mu)\frac{\mathbf{L}^\dagger}{\varepsilon_0}\boldsymbol{\Delta}\mathbf{T}, \tag{13}$$

with $(\cdot)^\dagger$ denoting the pseudoinverse, η being a given constant (see [1]) and vector $[\boldsymbol{\Delta}\mathbf{T}]_k = T_k - T$. Notice that [1] (see also [13] for the case $\mu = 0$) showed that, in absence of frequency mismatch among the clocks ($\boldsymbol{\Delta}\mathbf{T} = \mathbf{0}$), the network achieves full frequency and phase synchronization to the value

$$\boldsymbol{\tau}(n)\to\boldsymbol{\tau}^* = \mathbf{1}\cdot\mathbf{v}^T\boldsymbol{\tau}(0). \tag{14}$$

The results of [1] draw a correspondence between distributed discrete-time PLLs and the corresponding theory of point-to-point PLLs [9]. In fact, it is stated that, in absence of a frequency mismatch ($\boldsymbol{\Delta}\mathbf{T} = \mathbf{0}$), first and second-order PLLs are able to attain perfect phase recovery (and in particular the asymptotic common phase is (14)). However, with a frequency mismatch $\boldsymbol{\Delta}\mathbf{T}\neq\mathbf{0}$, only perfect frequency recovery is feasible (to the common value (12)), but with an asymptotic phase error (see (13)). Moreover, second-order PLLs ($\mu\neq 0$) help reducing the asymptotic error according to (13), but increasing $\mu\to 1$ leads to an instable system. We notice that similar results in the context of continuously-coupled distributed PLLs have been derived in [14]. Further discussion is provided by some examples in the following.

III. Resilience of distributed discrete-time PLLs

Here we evaluate the impact of malicious or faulty nodes on the basic system of distributed discrete-time PLLs discussed in the previous section via numerical simulation. Consider a network of $K = 20$ randomly distributed nodes in a square region of unit area. Among the K nodes, $K_m = 4$ are malicious or malfunctioning and select at each step n an independent random phase $\theta_k(n)$ (recall (1)), uniformly distributed in the set $(0, 1)$. The other $K_c = 16$ collaborating nodes run the discrete-time PLL illustrated in fig. 2 and discussed in the previous section. Constants C_{ki} are selected as $C_{ki} = 1$, thus neglecting channel randomness due to fading or shadowing. Fig. 3 shows the standard deviation of the clocks of collaborating nodes $\nu_t(n)$, with

$$\nu_t^2(n) = \frac{1}{K_c}\cdot\sum_{k\in\mathcal{K}_c}(t_k(n)-\frac{1}{K_c}\sum_{k\in\mathcal{K}_c}t_k(n))^2, \tag{15}$$

[2]A graph is strongly connected if there exists at least one path linking every pair of nodes. This condition is equivalent to requiring that matrix $\mathbf{A}$ is irreducible [16].

versus time n for the basic scheme described in the previous sections. Parameters are set as $\varepsilon_0 = 0.6$ and $\mu = 0$. Moreover, the initial phases $\tau_k(0) = t_k(0)$ are randomly and uniformly selected in the set $(0, 1)$, while for simplicity the local frequencies are set to $1/T_k = 1/T = 1$, $k = 1, ..., K$. The dashed line correspond to the performance of the system with no malicious nodes ($K = K_c = 20$), which as expected from the previous section leads to an asymptotically vanishing error $\nu_t(n)$. On the contrary, in a scenario with malicious nodes the timing error increases linearly, thus showing that the network is not able to reach full synchronization. This example demonstrates the sensitivity of the basic synchronization scheme (8) to attacks or faulty behavior.

IV. A RESILIENT ALGORITHM FOR DISTRIBUTED DISCRETE-TIME PLLS

In this section, we propose a solution to the issue of attack resilience of the synchronization scheme (8) illustrated in the previous section. The basic scheme (8) prescribes the evaluation by each collaborating node in $\mathcal{K}_c$, say the kth, of the weighted average $\Delta t_k(n)$ of the clock errors $\{\delta t_{ki}(n)\}_{i=1,i\neq k}^{K}$ in (6). Using only this measure, it is not possible for the nodes to recognize possible suspicious outliers that attempt to disrupt the synchronization process. Toward this goal, a possible solution is to evaluate the dispersion of the clock errors $\{\delta t_{ki}(n)\}_{i=1,i\neq k}^{K}$ around the mean $\Delta t_k(n)$, by, e.g., computing the variance

$$\sigma_k^2(n) = \sum_{i=1,i\neq k}^{K} \alpha_{ki} \cdot (\delta t_{ki}(n) - \Delta t_k(n))^2, \qquad (16)$$

and then consider as outliers all the clock differences $\delta t_{ki}(n)$ that satisfy

$$|\delta t_{ki}(n) - \Delta t_k(n)| > \beta \sigma_k(n), \qquad (17)$$

where β is some constant. The update (8) is performed by considering only the set of clock differences $\{\delta t_{ki}(n)\}_{i=1,i\neq k}^{K}$ such that index i belongs to the set $\mathcal{I}_k(n) = \{i \neq k$: $|\delta t_{ki}(n) - \Delta t_k(n)| \leq \beta\sigma_k(n)\}$. In other words, the algorithm (8) is modified by substituting $\Delta t_k(n)$ with

$$\widetilde{\Delta t}_k(n) = \sum_{i\in\mathcal{I}_k(n+1)} \tilde{\alpha}_{ki} \cdot \delta t_{ki}(n) \qquad (18a)$$

$$\tilde{\alpha}_{ki} = \frac{\alpha_{ki}}{\sum_{j\in\mathcal{I}_k(n+1)} \alpha_{kj}}. \qquad (18b)$$

Notice that normalization of the coefficients $\tilde{\alpha}_{ki}$ in (18b) is needed to guarantee that (18a) is a convex combination, i.e., $\sum_{i\in\mathcal{I}_k(n+1)} \tilde{\alpha}_{ki} = 1$.

A. Numerical results

Here we follow on the example presented in Sec. III in order to demonstrate the benefits of the secure algorithm proposed above. Fig. 3 shows the standard deviation $\nu_t(n)$ of the secure scheme for different values of the system parameter β in (17). For β small enough, the error $\nu_t(n)$ remains constant over n, thus showing that the secure scheme is able to achieve full synchronization within a limited (here 5%) timing error, as opposed to the basic scheme of Sec. II-A. More insights are provided by fig. 4, which shows the error $\nu_t(n)$ after $n = 100$ iterations versus parameter β for different fraction of malicious nodes K_m/K over the total $K = 20$. It can be seen that a sufficiently small β leads to a timing error that increases with K_m/K in a significantly less severe way with respect to the basic scheme. Notice that results similar to fig. 4 can be obtained for different total number of nodes K.

In the previous example, we considered a frequency-synchronous network. Here we discuss a scenario where the local free-oscillation frequencies of the K nodes $1/T_k$ are selected independently and uniformly in the set $1 \pm 1\%$ ($k =$

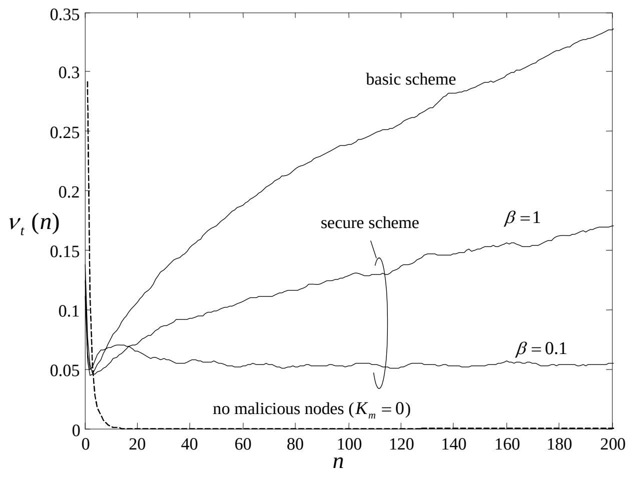

Fig. 3. Standard deviation $\nu_t(n)$ of the clocks of the collaborating nodes versus n for the basic scheme of Sec. II-A and the secure scheme of Sec. IV ($K = 20$, $K_m = 4$). Also shown as a reference is the performance with no malicious nodes ($K_m = 0$).

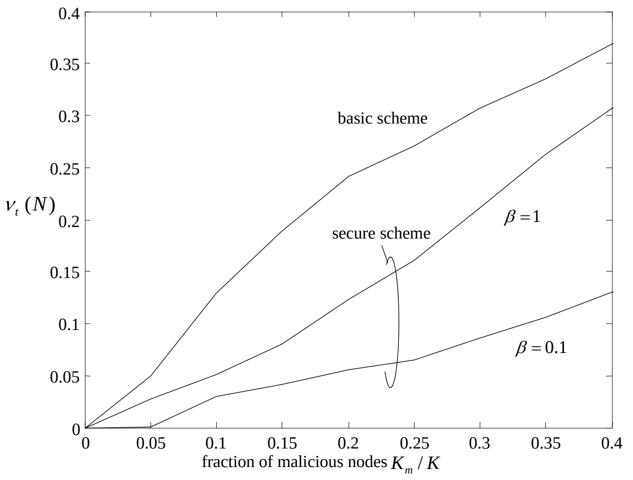

Fig. 4. Standard deviation $\nu_t(N)$ of the clocks of the collaborating nodes versus the fraction of malicious nodes K_m/K for the basic scheme of Sec. II-A and the secure scheme of Sec. IV ($K = 20$, $N = 100$).

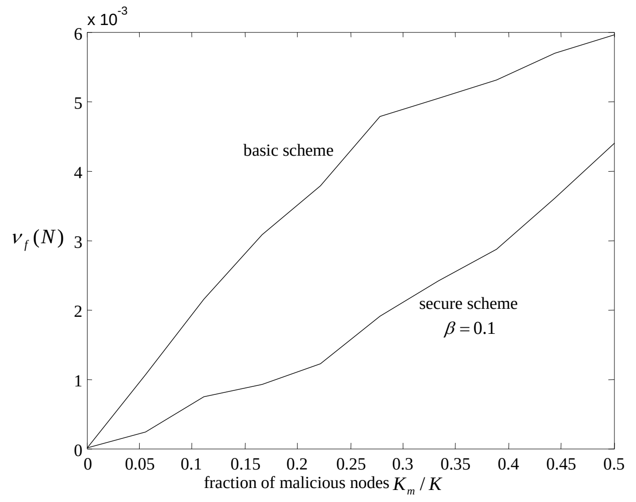

Fig. 5. Standard deviation $\nu_f(N)$ of the instantaneous frequencies of the clocks of the collaborating nodes versus the fraction of malicious nodes K_m/K for the basic scheme of Sec. II-A and the secure scheme of Sec. IV ($K = 20$, $N = 100$).

$1, ..., K$). Notice that in case of a frequency-asynchronous network, only perfect frequency synchronization is achievable with an asymptotic phase error given by (13). The simulation parameters are selected as in the previous example except the pole, set to $\mu = 0.6$ in order to reduce the phase error (13). Fig. 5 shows the instantaneous frequency error

$$\nu_f^2(n) = \frac{1}{K_c} \cdot \sum_{k \in \mathcal{K}_c} (T_k(n) - \frac{1}{K_c} \sum_{k \in \mathcal{K}_c} T_k(n))^2, \qquad (19)$$

where the instantaneous period $T_k(n)$ reads $T_k(n) = t_k(n) - t_k(n-1)$ for $n = 100$, versus the fraction of malicious nodes K_m/K ($K = 20$). Similarly to fig. 4, the secure scheme is able to guarantee a smaller (here frequency) error for any value of K_m/K.

V. Concluding remarks

In this paper, we addressed the problem of designing a secure distributed synchronization algorithm that is resilient to attacks by malicious nodes or to malfunctioning nodes in emergency situations. We focused on distributed discrete-time PLLs and proposed a modification of the basic scheme that has been shown by simulations to provide satisfactory performance. The method is based on the detection of ”suspicious” clocks as outliers with respect to the received signal statistics. A drawback of the technique is that it does not discriminate between malicious or faulty nodes and benign nodes that joined the network at a later stage where synchronization state had already been achieved. To mitigate such a problem, the system should be periodically re-initialized in order to allow the newcomers to synchronize with the rest of the network. Notice that this drawback is not shared by the basic scheme where new nodes eventually modify the synchronized state of the entire network. Therefore, it can be concluded that, for the proposed scheme, resilience to attacks comes at the expense of a loss in capability of adaptively synchronizing nodes added at a later stage to the network.

References

[1] O. Simeone and U. Spagnolini, ”Distributed time synchronization in wireless sensor networks with coupled discrete-time oscillators,” submitted to *Eurasip Journ. on Wireless Commun. and Networking*, special issue on Wireless Sensor Networks (invited).
[2] S. Ganeriwal, R. Kumar, ond M. Srivartava, ’Timing Sync Protocol for Sensor Networks,” in *Proc. ACM SenSyr*, Nov. 2003.
[3] J. Elson, L. Girod and D. Estrin, ”Fine-grained network time synchronization using reference broadcast,” in *Proc. 5th Symp. Operating Syst. Design, Implement.*, 2002.
[4] F. Sivrikaya and B. F. Yener, ”Time synchronization in sensor networks: a survey,” *IEEE Network*, vol. 18, no. 4, pp. 45-50, July-Aug. 2004.
[5] Y.-W. Hong, A. Scaglione, ”A scalable synchronization protocol for large scale sensor networks and its applications,” *IEEE Journal Selected Areas Commun.*, vol. 23, no. 5, pp. 1085-1099, May 2005.
[6] D. Lucarelli and I-J. Wang, ”Decentralized synchronization protocols with nearest neighbor communication,” *in Proc. ACM SenSys 2004*, Baltimore, Nov. 2004.
[7] Kun Sun, Peng Ning and Cliff Wang, ”Secure and resilient clock synchronization in wireless sensor networks,” *IEEE Journ. Selected Areas Commun.*, vol. 24, no. 2, pp. 395-408, Feb. 2006.
[8] H. Yang, F. Riccato and L. Zhang, ”Securing a wireless world,” *Proc. of the IEEE*, vol. 94, no. 2, pp. 442-454, Feb. 2006.
[9] F. M. Gardner, *Phaselock Techniques*, John Wiley & Sons, Inc., 1966.
[10] H. Meyr, M. Moeneclaey and S. A. Fechtel, *Digital communication receivers*, John Wiley & Sons, Inc., 1998.
[11] F. Tong and Y. Akaiwa, ”Theoretical analysis of interbase-station synchronization systems,” *IEEE Trans. Commun.*, vol. 46, no. 5, pp. 590-594, 1998.
[12] E. Sourour and M. Nakagawa, ”Mutual decentralized synchronization for intervehicle communications.” *IEEE Trans. Veh. Technol.*, vol. 48, no. 6, pp. 2015-2027, Nov. 1999.
[13] R. Olfati-Saber and R. Murray, ”Consensus problems in networks of agents with switching topology and time-delays,” *IEEE Trans. Automatic Control*, vol. 49, no. 9, pp. 1520–1533, Sept. 2004.
[14] W. C. Lindsey, F. Ghazvinian, W. C. Hagmann and K. Desseouky, ”Network synchronization,” *Proc. of the IEEE*, vol. 73, no. 10, pp. 1445-1467, Oct. 1985.
[15] D. P. Bertsekas and J. N. Tsitsiklis, *Parallel and distributed computation: numerical methods*, Athena scientific, 1997.
[16] C. Godsil and G. Royle, *Algebraic Graph Theory*, Springer-Verlag, 2001.
[17] An-Swol Hu and S. D. Servetto, ”On the scalability of cooperative time synchronization in pulse-connected networks,” *IEEE Trans. Inform. Theory*, vol. 52, no. 6, pp. 2725-2748, June 2006.
[18] S. N. Elayadi, *An introduction to difference equations*, Springer 1999.