Lifetime and Coverage Maximization in Wireless Sensor Networks [⋆]

**Daniele Fontanelli, Luigi Palopoli, Roberto Passerone,
David Macii, Dario Petri**

Dipartimento di Ingegneria e Scienza dell'Informazione, University of Trento - Trento, Italy (e-mail: {fontanelli,palopoli,passerone,macii,petri}@disi.unitn.it)

Abstract In the considered scenario, a wireless sensor network (WSN) operates in a difficult to reach (or even hostile) environment, and is therefore required to autonomously configure itself and tune its parameters. We investigate techniques that guarantee a good compromise between conflicting requirements: 1) a good coverage of the area, 2) a long lifetime, 3) a good temporal accuracy in classifying the events. Our solution is based on the combination of two complementary techniques. The first one, the wakeup scattering algorithm, iteratively identifies a periodic schedule that keeps the node awake for a limited time, so saving power. In this paper, we consider the global convergence of this algorithm. The second technique is a small overhead distributed algorithm for the synchronization of the nodes, which takes into account the communication delays.

Keywords: Sensor Systems, Distributed Control, WSN coverage, Synchronization

1. INTRODUCTION

In this paper, we consider the use of a wireless sensor network (WSN) to monitor an area. Our goal is threefold: 1) detecting events with a good probability, 2) ensuring a long and controllable lifetime for the network, 3) achieving a good accuracy in associating an event with the occurrence instant.

An effective way to extend the lifetime (which is determined by the battery duration) is by duty-cycling: a periodic alternation of intervals of time in which nodes are active and intervals of time in which nodes are in stand-by mode. The potential risk of duty-cycling is that some areas might be uncovered for a long time, with a detrimental effect on the probability of detecting events in those areas. This risk can be mitigated by choosing an appropriate schedule of the wake-up intervals that reduces the overlap between the nodes that cover the same area, the problem of wake-up scattering. In the absence of a precise knowledge of the position and of the sensing range of each node, which would enable an off-line computation of the optimal schedule (Palopoli et al. [2009]), it is possible to use decentralized heuristic algorithms, which use information from neighboring nodes (Cao et al. [2005], Cărbunar et al. [2006]). Important issues are then the algorithm convergence to a solution, how far the solution is from optimal, and how long the transient of the computation lasts.

The first important contribution of this paper is reporting global convergence results for the *wake–up scattering* algorithm, first proposed in Giusti et al. [2007]. The algorithm operates through a series of iterations, in which the network gets progressively closers toward the maximum coverage. In view of our result, the algorithm is guaranteed to converge to a global periodic schedule with a good average coverage of the area. The results presented in this paper (Section 2) extend prior work of the authors, in which the convergence problem was studied for particular topologies (see Fontanelli et al. [2009]).

Clearly, in order to implement this schedule, as well as to associate a meaningful temporal tag with each detected event, nodes need to synchronize their clocks. The problem of time synchronization is of paramount interest for the scientific communities dealing with networking and measurement science (IEEE [2008]), and several synchronization protocols has been proposed in recent years (Yoon et al. [2007], Schenato and Gamba [2007]). As a second contribution of this paper (Section 4), we propose a solution to the clock synchronization problem in which each node computes and updates individually its own local time using information collected from nearby nodes. Local clocks exchange their time data using a mechanism similar to the Reference Broadcast Synchronization (RBS) protocol and apply an enhanced version of the proportional and integral (PI) consensus controller (see Olfati-Saber et al. [2007] and Carli et al. [2008]). The specific contribution of our paper is that we consider time-varying communication delays and, more importantly, we adapt the sampling period to the variance of the clocks so as to keep in check the communication overhead.

Finally, we show how the concurrent application of the two techniques enhances the performance of the system in terms of area coverage, with respect to the application of each of the two techniques separately (Section 5).

[⋆] The research leading to these results has received funding from the European Community's Seventh Framework Programme FP7 under grant agreement n° IST-2008-224428 "CHAT - Control of Heterogeneous Automation Systems".

2. BACKGROUND AND MODEL DEFINITION

We consider a WSN consisting of n nodes $N_1, \ldots, N_n$. Nodes are deployed on an area and are endowed with a sensing mechanism operating within a range. An effective heuristic solution to optimize the coverage, as required by such applications as surveillance against intrusion detection, is to “scatter” the activation times of the nodes (see Giusti et al. [2007]). We denote by $w_i \in [0, E]$ the wake-up time of node N_i and by $\mathcal{V}_i$ the set of nodes visible from node N_i ($i \notin \mathcal{V}_i$). The wake-up time of node N_i at step k is updated as follows:

$$w_i^{k+1} = (1-\alpha)\, w_i^k + \tfrac{\alpha}{2}\left(\min_{j\in\mathcal{V}_i}\{w_j^k : w_j^k \geq w_i^k\} + \max_{j\in\mathcal{V}_i}\{w_j^k : w_j^k \leq w_i^k\}\right) \mod E. \tag{1}$$

The rationale behind this algorithm is the assumption that neighboring nodes are more likely to cover the same area. This assumption is strictly verified only when the sensing and the communication ranges are comparable, but the resulting technique is simple and relies solely on connectivity, instead of requiring exact position information.

In order for the outcome of the computation of the wake-up times to be applicable, we need that nodes have synchronized clocks. The relative time of each node can be measured by means of a local timer clocked by a local (preferably crystal) oscillator of nominal frequency f_0. In general, clocks may be dis-aligned for two reasons: 1) each node can with a different initial offset, 2) clocks can drift for the differences between the oscillator frequencies.

In the rest of the paper, we will discuss the convergence of the wakeup scattering algorithm and show our specific solution for clock synchronization. In this section, we briefly discuss a state space model for the two systems.

2.1 Modelling the Wakeup Scattering Algorithm

Let us start with an example. Suppose that the epoch is equivalent to one minute and that the granularity of the wakeup times is the second. In such a case, each w_i corresponds to a position of the second hand, that is invariant to the minute and/or hour chosen. The epoch E defines a ring symmetry, visually equal to the clock dial.

For each pair of nodes (N_i, N_j), we define two distances w.r.t. the wake up times w_i and w_j: one, denoted $E \geq \overrightarrow{d}_{i,j} \geq 0$ that goes forward in time, the other, denoted $E \geq \overleftarrow{d}_{i,j} \geq 0$ that goes backward. Using the update Equation (1), it is possible to derive the dynamics of $\overrightarrow{d}_{i,j}$ and $\overleftarrow{d}_{i,j}$. Given the ring invariance $\overleftarrow{d}_{i,j} = E - \overrightarrow{d}_{i,j}, \forall i, j$, consider a state vector $\mathbf{x}$ whose entries are the distances $\overrightarrow{d}_{i,l}$, $\forall l \in \mathcal{V}_i$ and for $i = 1, \ldots, n$. The discrete time evolution of distances is $\mathbf{x}^{k+1} = A\mathbf{x}^k$, where A has, at least, one eigenvalue equals to one.

In Fontanelli et al. [2009], the reader can find additional details on this construction. In particular, we showed that if two nodes do not see each other, their distance can change in sign. This behavior, along with the non-linearity of Equation (1), makes the overall system dynamics switching. Defining $\sigma(k)$ as the switching signal, that takes values $1, \ldots, S$, the switching system $\mathbf{x}^{k+1} = A_{\sigma(k)}\mathbf{x}^k$ is thus derived, with system matrices $\{A_1, A_2, \ldots, A_S\}$. The region of the state space in which the system evolves using a dynamic A_i is a convex polyhedron delimited by a set of subspaces of the type $x_i < x_j$, for appropriate choices of i and j. Hence, in the general case, the number S of linear dynamics is upper bounded by the number of pairs of nodes that do not see each other.

On the other hand, the distances between visible nodes never change in sign. Intuitively, by Equation (1), w_i^{k+1} will be placed in between the wakeup times of the preceding and following nodes that it sees. Therefore, if all nodes see their nearest neighbors, the system evolves with a linear and time–invariant dynamics. The algorithm in this case asymptotically converges to an equilibrium in which the wake–up times are equally spaced in the epoch E. The convergence analysis for the general case is reported in the next section.

2.2 Modelling the Clock Synchronization Problem

Let us consider a generic WSN consisting of n identical nodes, each one with a unique identifier in the set $ID = \{1, \ldots, n\}$. Formally, the WSN timers are a set of discrete–time linear integrators, i.e.,

$$\mathbf{x}_\tau(t+1) = \mathbf{x}_\tau(t) + \mathbf{d}(t) + \underline{\mathbf{q}}(t), \tag{2}$$

where $t \in \mathbb{N}_0$ represents the number of clock ticks on an ideal timescale, $\mathbf{x}_\tau \in \mathbb{R}^n$ is the column vector containing the time values of all WSN nodes and formally correspond to the state of the system, $\underline{\mathbf{q}} \in \mathbb{R}^n$ is the random vector (whose elements are uniformly distributed in the interval $[0, \frac{1}{f_0}]$) modelling the quantization noise due to the resolution of the timer[1]. $\mathbf{d} \in \mathbb{R}^n$ is the vector containing the actual time increments of each clock, i.e., the clock drifts, during the $t-$th tick of the timer. In particular, $\mathbf{d}$ is given by: $\mathbf{d}(t) = \frac{1}{f_0}\mathbf{1} + \boldsymbol{\Delta}(t) + \underline{\boldsymbol{\nu}}(t)$ where $\mathbf{1}$ is the unit vector, $\boldsymbol{\Delta} \in \mathbb{R}^n$ is the vector of the systematic offsets between the ideal and the real clock periods of the various nodes and $\underline{\boldsymbol{\nu}} \in \mathbb{R}^n$ is a random vector modelling the jitter resulting from the superimposition of different types of power–law noise (Bregni [1997]). Of course, if $x_{\tau_i}(0) = x_{\tau_j}(0)$ and if $d_i(0) = 1/f_0$, for all $i, j = 1, \ldots, n$, then the clocks will remain synchronized from the very beginning. However, in a realistic scenario initial clock offsets and clock frequency skews differ, thus the corresponding time values drift away from one another.

In this context, the term *time synchronization* should be intended as the compensation of the inter–node time differences, regardless of the Coordinated Universal Time (UTC), here indicated by t. Hence, the nodes can be considered as *synchronized*, if there exists a finite time $\bar{t}$ and two values $b_1, b_2 \in \mathbb{R}$ such that $\|x_{\tau_i}(t) - (b_1 t + b_2)\| \leq \varepsilon$ for $t \geq \bar{t}$ and for $\forall i = 1, \ldots, n$, with fixed $\epsilon \geq 1/f_0$.

3. CONVERGENCE OF WAKEUP SCATTERING

For lack of space, we cannot report the entire analysis of the global convergence. In this paper, we only offer an intuitive description of the most important results, omitting the proofs of the theorems.

[1] In this paper random variables are denoted by underlined symbols.

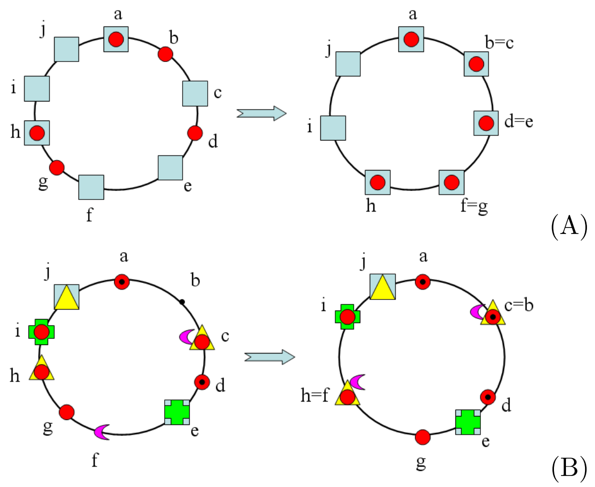

Figure 1. On the left the initial configuration or the networks is depicted, while on the right the reached equilibrium is shown. The circumferences are of length E. (A) Two connected sub–networks topology. Nearest neighbors are the closest nodes in clockwise and counter–clockwise direction, i.e. the nearest neighbor to c are d and a respectively. Visibility is depicted with different symbols, squares and circles in this case ($\{b, d, g\} \notin \mathcal{V}_{c,e,f}$ and viceversa). (B) Generic network topology. Notice that if a node has m associated symbols, it means that m different nodes can be the nearest for some instants.

Instrumental to the analysis of global convergence is the fact that nodes that see each other never overtake each other, regardless of the switchings that may occur (see Fontanelli et al. [2009]).

3.1 Global Stability for Connected Sub–Networks

We start our analysis, by considering a particular topology, defined as follows.

Definition 1. A network with n nodes is called *ordered* if for any pairs of ordered indices $i < j$ implies $w_i < w_j$, $\forall i, j = 1, \dots, n$.

Definition 2. Given an ordered network, a *sub–network* Θ is a set of $n_\theta + 2$ nodes, with ordered indices $\{\theta_0, \theta_1, \dots, \theta_{n_\theta}, \theta_{n_\theta+1}\}$ ($\theta_i < \theta_j$ if $i < j$), whose wake–up times are ordered $w_{\theta_0} < w_{\theta_1} < \cdots < w_{\theta_{n_\theta}} < w_{\theta_{n_\theta+1}}$ and such that node θ_i sees nodes θ_{i-1} and θ_{i+1}. Nodes θ_0 and $\theta_{n_\theta+1}$ are defined as the *end–points* of the sub–network.

The previous definitions are instrumental for the subsequent definition of partial visibility network by means of sub–networks.

Definition 3. Given an ordered network and two sub–networks Θ and Σ whose end–points are coincident and equal to $\theta_0 \equiv \sigma_0 \equiv \nu_i$ and $\theta_{n_\theta+1} \equiv \sigma_{n_\sigma+1} \equiv \nu_e$ respectively, Θ and Σ are named *connected sub–networks* $\Theta \rightsquigarrow \Sigma$ if any element of Θ does not see any element of Σ, except the end–points ν_i and ν_e.

Example 1. The network of figure 1-A has $\Theta = \{a, b, d, g, h\}$ and $\Sigma = \{a, c, e, f, h\}$ with $\Theta \rightsquigarrow \Sigma$.

Let us consider the case of two connected sub–networks with $n_\theta > n_\sigma$ We can construct a state vector $\mathbf{z} = [\overrightarrow{d}_{\theta_1,\sigma_1}, \dots, \overrightarrow{d}_{\theta_{n_\sigma},\sigma_{n_\sigma}}]^T$, composed of n_σ distances between nodes that do not see each other in the connected sub–networks. Applying Equation (1) and considering that the distance between two nodes can be expressed as difference of distances between nodes that see each other (e.g., $\overrightarrow{d}_{\theta_i,\sigma_i} = \overrightarrow{d}_{\nu_i,\sigma_i} - \overrightarrow{d}_{\nu_i,\theta_i}$) we come up with an update expression for the state $\mathbf{z}^+ = A_{cn}\mathbf{z} + \mathbf{c}$, where $\mathbf{c} = [0, \dots, 0, \alpha/2\overrightarrow{d}_{\theta_{i+1},\nu_e}]^T$, represents the dynamics of a linear system with A_{cn} be a tridiagonal Toepliz matrix with eigenvalues λ_i. Since $-1 < \lambda_i < 1$ if and only if $0 < \alpha < 1$, the unforced linear system dynamics is asymptotically stable with equilibrium $\mathbf{z} = 0$. Since $\mathbf{c}$ converges to a constant value independently of $\mathbf{z}$, the system with bounded input $\mathbf{c}$ is asymptotically stable with a number of switchings that depends on the eigenvalues of the matrix A_{cn} and on the initial conditions $\mathbf{z}(0)$. In the case of Figure 1-(A) with $n_\theta = n_\sigma$, the analysis follows trivially since $\mathbf{c} = 0$.

In light of the discussion above, the following summarizing results hold.

Theorem 1. Given a network with n nodes and with two connected sub–networks with $n_\theta \geq n_\sigma$, the distances between nodes that see each other will eventually converge to $E/(n - n_\sigma)$.

Corollary 1. The equilibrium of the distances between the nodes of the shorter sub–network Σ is given by $E(n_\theta + 1)/[(n - n_\sigma)(n_\sigma + 1)]$

Remark 1. The Theorem 1 still holds if: 1) $\nu_i \equiv \nu_e$ 2) there exists any number of connected sub–networks between nodes ν_i and ν_e; 3) one or more connected sub–networks comprise additional connected sub–networks.

3.2 Global Stability for a Generic Network Topology

To prove the convergence in the general case, we will firstly construct all the possible sub–networks for a generic network topology (see figure 1-(B), left). Consider an ordered network Δ with n nodes and with associated ordered indices $\{\delta_1, \delta_2, \dots, \delta_n\}$. For each $i = 1, \dots, n$, consider δ_i and construct the set, ordered with respect to the wake–up times in the clockwise direction, of all the nodes that it sees. In this set, remove the nodes that see each other, obtaining $\mathcal{C}_i = \{\delta_{i,1}, \dots, \delta_{i,m_i}\}$. Notice that $m_i \geq 1$, $\forall i$. Indeed, $m_i = 0$ for blind nodes (not considered). Furthermore, $m_i = 1$ for all i in the case of nearest neighbor visibility.

Starting from node i, construct a tree with root δ_i and first level nodes given by the set $\mathcal{C}_i$. For each node $\delta_{i,j}$, $j = 1, \dots, m_i$, construct the second level using the sets $\mathcal{C}_j$. If a node $\delta_{j,k} \in \mathcal{C}_j$ is equal to δ_i, the construction procedure stops for $\delta_{j,k}$ with leaf δ_i. If the sum of the wake–up times distances from the root to a node $\delta_{j,k} \in \mathcal{C}_j$ is greater than E, the construction procedure stops for that node and the branch to $\delta_{j,k}$ is removed. In all the other cases, the procedure continues recursively. This way, all the possible paths π from the root δ_i to a leaf δ_i are available. The set of all the paths generated from each node δ_i is dubbed $\mathcal{P}$.

Let us consider the i-th path $\pi^i \in \mathcal{P}$. The path π^i, of length l_i, is represented by an ordered set of nodes $\{\delta_{\pi^i_1}, \delta_{\pi^i_2}, \dots, \delta_{\pi^i_{l_i}}\}$, with $\delta_{\pi^i_j} \neq \delta_{\pi^i_k}$ except for $j = 1$

and $k = l_i$. Furthermore, it is straightforward that all paths obtained by π^i by permutations that preserve the order represents the same path, i.e., $\{\delta_{\pi^i_1}, \delta_{\pi^i_2}, \ldots, \delta_{\pi^i_{l_i}}\}$ is equivalent to $\{\delta_{\pi^i_j}, \ldots, \delta_{\pi^i_{l_i}}, \delta_{\pi^i_2}, \ldots, \delta_{\pi^i_j}\}$. Hence, let us call $\bar{\pi}^i$ the set of all the possible node sequences regardless of the path π^i node permutations. Moreover, let us define $\bar{\mathcal{P}}$ the set of paths ordered with respect to the length of the paths, in descending order, such that $\bar{\pi}^i \not\subset \bar{\pi}^j$, for all $\bar{\pi}^i$, $\bar{\pi}^j \in \bar{\mathcal{P}}$.

Example 2. Let us consider the case of connected sub–networks, represented in figure 1-(A). The procedure produces $\bar{\pi}^1 = \{a, c, e, f, h, i, j, a\}$, $\bar{\pi}^2 = \{a, b, d, g, h, i, j, a\}$ that comprises the connected sub–networks of the figure. For the generic network topology of figure 1-(B), the $\bar{\mathcal{P}}$ has 6 strings, i.e., $\bar{\pi}^1 = \{a, c, d, g, h, i, a\}, \bar{\pi}^2 = \{a, b, d, g, h, i, a\}, \bar{\pi}^3 = \{c, d, g, h, j, c\}, \bar{\pi}^4 = \{e, j, e\}, \bar{\pi}^5 = \{e, i, e\}, \bar{\pi}^6 = \{c, f, c\}$.

If $\bar{\pi}^i \cap \bar{\pi}^j = \{\delta_k\}$, they generate two connected sub–networks with start and end points in δ_k. In general, if $\bar{\pi}^i \cap \bar{\pi}^j = \{\delta_k, \ldots, \delta_{k+m}\}$ is a sequence of consecutive nodes, then the two sub–network $\Sigma \in \bar{\pi}^i$ and $\Theta \in \bar{\pi}^j$ have start–point δ_{k+m} and end–point δ_k (see Example 2). Such results can be generalized to networks with general topology.

Example 3. Consider two paths $\bar{\pi}^i$, $\bar{\pi}^j \in \bar{\mathcal{P}}$, with $l_i \leq l_j$. Let $\bar{\pi}^i \cap \bar{\pi}^j = \{\delta_k, \ldots, \delta_{k+m}\}$, where $\Sigma \in \bar{\pi}^i$ and $\Theta \in \bar{\pi}^j$ are the two connected sub–networks. In light of Theorem 1 and Corollary 1, the equilibrium distances of the nodes in $\bar{\pi}^j$ is E/l_j, while the equilibrium distances of the nodes in $\bar{\pi}^i \backslash (\bar{\pi}^i \cap \bar{\pi}^j)$ is given by $E(l_j - m + 1)/[l_j(l_i - m + 1)]$.

Without loss of generality, consider that the undirected graph representing the network visibility is strongly connected, i.e., from each node in $\bar{\mathcal{P}}$ it is possible to reach any other node of the network. If the graph is not strongly connected, the two (or more) disjoint sub–networks can be analyzed separately.

Let us summarize the previous results in matrix terms, noticing that each path $\bar{\pi}^i$ *is* a network with nearest neighbor visibility. With reference to the longest path $\bar{\pi}^1$, the dynamic of the distances between nearest neighbor nodes $\mathbf{x}_{\bar{\pi}^1}$ can be expressed by a single matrix $A_{\bar{\pi}^1} \in \mathbb{R}^{(l_1-1)\times(l_1-1)}$. The dynamic of the distances of the network $\bar{\pi}^1 \cup \bar{\pi}^2$ is determined by the switching matrix set $\mathcal{A}_{\bar{\pi}^1 \cup \bar{\pi}^2}$. Arranging the new distances in a vector $\mathbf{x}_{\bar{\pi}^1 \cup \bar{\pi}^2} = [\mathbf{x}_{\bar{\pi}^1}, \mathbf{x}_{\bar{\pi}^2/\bar{\pi}^1}]^T$, by means of Theorem 1 and Corollary 1 follows that the steady state matrix governing the dynamic of the distances is given by

$$A_{\bar{\pi}^1 \cup \bar{\pi}^2} = \begin{bmatrix} A_{\bar{\pi}^1} & 0 \\ B_{\bar{\pi}^1 \cup \bar{\pi}^2} & A_{\bar{\pi}^2} \end{bmatrix}.$$

By Remark 1, the same holds if there are two (or more) sequences of common nodes in $\bar{\pi}^1 \cap \bar{\pi}^2$. Adding an additional path $\bar{\pi}^3$, with $l_3 \leq l_2 \leq l_1$, the steady state matrix of the overall distances would be given by

$$A_{\bar{\pi}^1 \cup \bar{\pi}^2 \cup \bar{\pi}^2} = \begin{bmatrix} A_{\bar{\pi}^1 \cup \bar{\pi}^2} & 0 \\ B_{\bar{\pi}^1 \cup \bar{\pi}^2 \cup \bar{\pi}^2} & A_{\bar{\pi}^3} \end{bmatrix}.$$

Therefore:

Theorem 2. Given a generic network, the wake–up scattering algorithm asymptotically converges towards an equilibrium where the node distances are given by the path lengths l_i.

4. CLOCK SYNCHRONIZATION ALGORITHM

In principle, we can achieve time synchronization by suitably controlling the timers of the various nodes. Accordingly, the discrete–time linear system (2) can be modified as follows:

$$\mathbf{x}_\tau(t+1) = \mathbf{x}_\tau(t) + \mathbf{d}(t) + \underline{\mathbf{q}}(t) + \mathbf{u}(t), \qquad (3)$$

with $\mathbf{u} \in \mathbb{R}^n$ being the vector of the control inputs to each timer. In Carli et al. [2008] it is proved that the problem can be solved by using a *Proportional Integral* (PI) controller on each node. In matrix notation, a general expression for this controller is:

$$\mathbf{y}(t+1) = \mathbf{y}(t) - \alpha \mathbf{K} \underline{\hat{\mathbf{x}}_\tau}(t) \qquad (4)$$

$$\mathbf{u}(t) = \mathbf{y}(t) - \mathbf{K} \underline{\hat{\mathbf{x}}_\tau}(t) \qquad (5)$$

where $\mathbf{y} \in \mathbb{R}^n$ is the state vector of the controller, the feedback matrix $\mathbf{K}$ and the coefficient α result from the consensus–related theory and $\underline{\hat{\mathbf{x}}_\tau} = \mathbf{x}_\tau - \underline{\boldsymbol{l}}$ is the vector of the time values measured by the various nodes. Notice that in general the elements of $\underline{\hat{\mathbf{x}}_\tau}$ are different from those of $\mathbf{x}_\tau$ because both collecting the time values from all nodes and computing the next controller output requires an additional time, represented by the latency random vector $\underline{\boldsymbol{l}} \in \mathbb{R}^n$, usually in the order of several tens of ms. It is worthwhile to note that the communication latencies may increase considerably as a function of the number of nodes, since they are related to the number of messages for transferring the time values and to the probability of having packet collisions (see Ageev et al. [2008]).

4.1 Controller Design

The main consequence of the presence of the latencies is that the input values to the controller can be updated at a rate which is much smaller than the frequency of the oscillator clocking the timer, thus generating a switching behavior between two different controllers.

The first one is represented by equations (4) and (5) and occurs as soon as a new complete set of local time values for all nodes is available. In this case, plugging the equation (5) into clock dynamic equation (3), we have

$$\begin{aligned} \begin{bmatrix} \mathbf{x}_\tau(t+1) \\ \mathbf{y}(t+1) \end{bmatrix} &= \begin{bmatrix} I_n - \mathbf{K} & I_n \\ -\alpha\mathbf{K} & I_n \end{bmatrix} \begin{bmatrix} \mathbf{x}_\tau(t) \\ \mathbf{y}(t) \end{bmatrix} + \xi^1(t) \\ &= A^1_{cl} \begin{bmatrix} \mathbf{x}_\tau(t) \\ \mathbf{y}(t) \end{bmatrix} + \xi^1(t), \end{aligned} \qquad (6)$$

where I_n is a $n \times n$ identity matrix and

$$\xi^1(t) = \begin{bmatrix} \mathbf{d}(t) + \underline{\mathbf{q}}(t) + \mathbf{K}\underline{\boldsymbol{l}} \\ \alpha\mathbf{K}\underline{\boldsymbol{l}} \end{bmatrix}, \qquad (7)$$

is the vector containing all the described nuisances.

The second configuration, when new local time values $\underline{\hat{\mathbf{x}}_\tau}$ are not available, hold constant the output $\mathbf{u}$ of the controller. Hence, the corresponding closed–loop dynamics is given by (6), where the matrix A^0_{cl} and the vector $\xi^0(t)$ are obtained imposing $\mathbf{K} = 0$ in (6) and (7).

The system switches between the two different dynamics depend on the time interval between two subsequent synchronizations. The lower bound $\underline{\gamma}$ of this interval is equal to the (random) time spent to run each iteration. On the contrary, the time interval $\gamma_k \geq \underline{\gamma}$ after the k–th synchronizations can be, in principle, arbitrarily large. In more strict theoretic terms, the overall closed loop is governed by the closed–loop matrix $A_{cl}^{\sigma(t,\gamma_k)}$, determined by the *switching signal* $\sigma(t, \gamma_k)$, that is equal to 1 if local time values are available and 0 otherwise. Therefore, the dynamics of the system is given by

$$A_{cl}^{0\;\gamma_k-1}A_{cl}^{1} = A_{cl_{\gamma_k}} = \begin{bmatrix} I_n - [1+\alpha(\gamma_k-1)]\mathbf{K} & \gamma_k I_n \\ -\alpha\mathbf{K} & I_n \end{bmatrix} \qquad (8)$$

Similarly to Carli et al. [2008], the asymptotic stability of the system governed by the dynamic matrix (8) for a fixed γ_k can be obtained by choosing the eigenvalues λ_i of $\mathbf{K}$, with $i = 1, \ldots, n$, in the set $(0, 4/[2 + \alpha(\gamma_k - 2)])$, where $\alpha \in (0, 1)$. Notice that the feedback matrix $\mathbf{K}$ computation can be distributed in each node, since it is obtained by scaling the Laplacian $\mathbf{L}$ of the visibility matrix. This way, the system can be made stable and

$$x_{\tau_i}(t) \to \frac{1}{n}\sum_{j=1}^{n}\left[d_j t + x_{\tau_j}(0)\right],\ \forall i,\ \ t \to +\infty. \qquad (9)$$

In other words, every timer converges to the mean of the time values measured by all network nodes. The effect of $\xi^1(t)$ and $\xi^0(t)$ is to decrease the steady state accuracy of the algorithm.

In the case of complete visibility, i.e., when all the WSN nodes of a cluster are able to communicate within a single–hop link, the feedback matrix $\mathbf{K}$ and the parameter α can be computed by each node on the basis of the planned synchronization period γ_k. In particular, by choosing

$$\alpha = \frac{1}{\gamma_k + 1} \ \text{ and } \ \mathbf{K} = \frac{\gamma_k + 1}{n\gamma_k}\mathbf{L}, \qquad (10)$$

two of the eigenvalues of the closed–loop matrix in (8) are equal to 1 and all the others are 0, which guarantees the fastest convergence (dead–beat controller).

Notice that if the time interval γ_k between the $k-$th and the $(k{+}1)-$th synchronization changes, then also (8) must change according to (10). This leads to a set of switching matrices, namely one for each γ_k. Since switching between stable controllers may disrupt stability, unstable behaviors are avoided assuming an appropriate *dwell time* between two successive switches. Hence, define $\sigma_x^2(t)$ as the variance of the time differences for the first time t in which the period γ_k is selected. If $\gamma_{k+1} \neq \gamma_k$ at a certain time $\bar{t} > t$ and $\sigma_x^2(t) \leq \sigma_x^2(\bar{t})$, then the next period has to be kept equal to γ_k. Conversely, if $\sigma_x^2(t) > \sigma_x^2(\bar{t})$, the next period can be safely set to γ_{k+1}. This way, the asymptotic stability of the switching system is assured and the synchronization uncertainty still globally decreases, although some local divergence during the dwell time may exist.

4.2 Synchronization Procedure

The synchronization procedure based on the control algorithm described in the previous Section consists of four iterative steps:

Step 1: When a generic synchronization interval γ_k (for any $k \geq 1$) expires, the node with identifier equals to

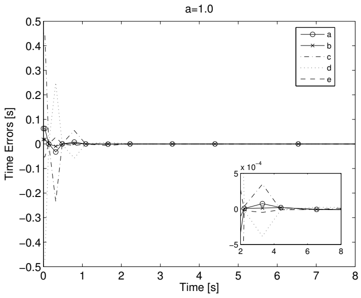

Figure 2. Time differences measured by 5 WSN nodes obtained with the synchronization algorithm with $a = 1$. Each line style corresponds to a different node.

mod (k, n) is elected as the *synchronization master* (SM) and it broadcasts a MAC time–stamped *beacon packet*. We will refer to t_{s_m} as the sending time–stamp of the synchronization beacon. Each other node receiving the synchronization beacon, time–stamps the incoming packet using its local clock at time $t^i_{r_m}$ and modifies the visibility matrix accordingly ($t_{s_m} \neq t^i_{r_m}$ due to latencies). Noticing that the communication broadcasting times are approximately the same for all nodes and that the interrupt service routines are smaller than $\underline{\boldsymbol{l}}$, the differences between the values of $t^i_{r_m}$ are mainly caused by unequal offsets and skews of the local clocks;

Step 2: The i–th receiving node broadcasts the value of its own $t^i_{r_m}$, which is time–stamped at time t_{s_i}. Any other receiving node j time–stamps the incoming packet at time $t^j_{r_i}$ and updates the visibility matrix;

Step 3: The i-th node is now able to estimate the mean communication latency, used to compensate the differences between the reception/sending time–stamps applied by the SM, and to compute its feedback u_i using (10) to apply to (3);

Step 4: Immediately after correcting the timers the next synchronization interval γ_{k+1} is set to $\gamma_k(1 + a)$ if the dwell–time time condition on σ_x^2 is met, γ_k otherwise. Since $a \geq 0$ the duration of subsequent synchronization intervals may only increase.

5. SIMULATIONS

5.1 Synchronization Algorithm

The effectiveness of the proposed approach has been validated through several simulations in Matlab™. All simulations take advantage of the synchronization uncertainty models developed in previous research works (Ageev et al. [2008]). Fig. 2 shows the performance of the algorithm when the interval expansion coefficient $a = 1$. The simulated WSN consists of 5 nodes, whose timers are clocked by crystal oscillators (XO) running at $f_0 = 32768$ Hz, as in XBow TelosB™ or Tmote-Sky™ platforms. Nodes' timers are randomly initialized in the range [0, 1] seconds. Clock relative systematic frequency skews are randomly chosen between ±100 ppm, while the timing jitter caused by phase noise is in the order of 2 ns rms over 1 s. Communication latencies depend on the offered network traffic and they are in the order of several milliseconds.

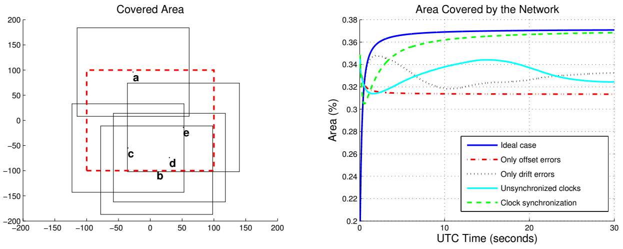

Figure 3. Node deployment in the area of interest (left) and results of the scattering algorithm w.r.t. to different clock synchronizations (right).

The steady state accuracy of the proposed clock synchronization algorithm obtained in simulation has been shown to be of ±1 timer tick.

5.2 Synchronization and Scattering Algorithms

The coexistence of the scattering and the clock synchronization algorithms is straightforward since each algorithm is distributed and works on a different subset of information exchanged in the network. Indeed, the clock synchronization algorithm needs the timer information of each clock, collected in the state vector $\mathbf{x}_\tau$. On the other hand, the scattering algorithm defines the node wake–up schedules w based on the visibility and on the chosen epoch E, regardless of the actual clock timers.

Nevertheless, the clock synchronizations affects the overall performance of the scattering algorithm. For example, suppose $E = 3$ time units and 3 nodes (a, b and c) with complete visibility and with a wake-up interval for the nodes, i.e., the time in which a node is awake, equals to 1. In this case, the scattering algorithm will converge to a solution in which the node a is active if $0 \leq \mod (t_a, E) < 1$, b for $1 \leq \mod (t_b, E) < 2$ and c for $2 \leq \mod (t_c, E) < 3$, where t_i is the clock timer of node i. If all the clock nodes are synchronized, $t_a = t_b = t_c = t$ and during the epoch at least one node is awake. Unfortunately, in the worst case of unsynchronized clocks $t_a = t$, $t_b = t+1$ and $t_c = t + 2$ and, hence, the node will be all active only in the same time unit.

In order to show the performance of the cooperation of both the algorithms, we present the coverage problem with 5 nodes, randomly deployed over a bi-dimensional area (fig. 3, left). For the sake of simplicity, we consider a rectangular sensing range and a complete visibility among the nodes of the WSN. The wake-up interval for the nodes equals to $E/5$ s, i.e., each node is awake for 20% of the total time. The application of the wake-up scattering algorithm produces the result shown in fig. 3, right, from which it is evident how the clock synchronization algorithm correct the skews among the clock timers and allow the scattering algorithm to converge towards the ideal solution.

6. CONCLUSIONS

In this paper, we have considered a realistic application of a WSN to monitor an area of interest. Firstly, a suboptimal coverage of the area, with a controlled lifetime for the WSN, is obtained by means of the wake-up scattering algorithm, for which global convergence has been proved. Then, the problem of node clock synchronization for a realistic WSN has been solved using a consensus–based switching controller. By simulations, the benefits of the combination of both the techniques has been shown. As a future work, we plan to apply the wake-up scattering to target tracking problems and to study the robustness issues of the clock synchronization protocol in case of unstable radio links.

REFERENCES

A. Ageev, D. Macii, and D. Petri. Synchronization uncertainty contributions in wireless sensor networks. In *Proc. of Int. Instrumentation and Measurement Technology Conference*, pages 1986–1991, 2008.

S. Bregni. Clock stability characterization and measurement in telecommunications. *IEEE Trans. on Instrumentation and Measurement*, 46(6):1284–1294, December 1997.

Q. Cao, T. Abdelzaher, T. He, and J. Stankovic. Towards optimal sleep scheduling in sensor networks for rare-event detection. In *Proc. of the 4^{th} Int. Symp. on Information Processing in Sensor Networks (IPSN)*, April 2005.

R. Carli, A. Chiuso, L. Schenato, and S. Zampieri. A PI consensus controller for networked clocks synchronization. In *Proc. of 17th IFAC World Congress*, Seoul (Korea), July 2008.

B. Cărbunar, A. Grama, J. Vitek, and O. Cărbunar. Redundancy and coverage detection in sensor networks. *ACM Transaction on Sensor Networks*, 2(1):94–128, February 2006.

D. Fontanelli, L. Palopoli, and R. Passerone. Convergence of distributed wsn algorithms: The wake-up scattering problem. In R. Majumdar and P. Tabuada, editors, *Proc. of Hybrid Systems: Computation and Control*, pages 180–193, San Francisco, April 2009. Springer-Verlag Berlin Heidelberg.

A. Giusti, A.L. Murphy, and G.P. Picco. Decentralized Scattering of Wake-up Times in Wireless Sensor Networks. In *Proc. of the 4^{th} European Conf. on Wireless Sensor Networks (EWSN)*, LNCS 4373, pages 245–260. Springer, January 2007.

IEEE. *IEEE 1588:2008, Precision clock synchronization protocol for networked measurement and control systems.* New York, USA, July 2008.

R. Olfati-Saber, J.A. Fax, and R.M. Murray. Consensus and cooperation in networked multi–agent systems. *Proc. of IEEE*, 95(1):215–233, January 2007.

L. Palopoli, R. Passerone, A.L. Murphy, G.P. Picco, and A. Giusti. Solving the wake-up scattering problem optimally. In Utz Roedig and Cormac J. Sreenan, editors, *6th European Conf. Wireless Sensor Networks (EWSN)*, volume 5432 of *Lecture Notes in Computer Science*, pages 166–182. Springer, 2009.

L. Schenato and G. Gamba. A distributed consensus protocol for clock synchronization in wireless sensor network. In *Proc. of IEEE Conf. on Decision and Control*, pages 2289–2294, New Orleans, LA, USA, 12-14 December 2007.

S. Yoon, C. Veerarittiphan, and M.L. Sichitiu. Tiny-sync: Tight time synchronization for wireless sensor networks. *ACM Trans. on Sensor Networks (TOSN)*, 3(2):1–33, June 2007.