

RECURSIVE TIME-SYNCHRONIZATION IN SENSOR NETWORKS

Prabir Barooah and Ananthram Swami

Abstract—Clock synchronization is a critical challenge in wireless sensor networks. Noisy measurements of relative offset and skew between pairs of nodes can be obtained by an exchange of time-stamped packets. The problem then can be cast as a linear estimation problem, and many distributed algorithms have been proposed recently. These algorithms are iterative, involving exchange of estimates with 1-hop neighbors. These message exchanges could be time-stamped to yield additional measurements of clock offsets and skews. We propose a distributed algorithm to fuse the new measurements with the current estimates. We show that the estimates produced by the proposed protocol converge in mean square to the true parameter values. More importantly, the error variance of the estimates even after a small number of iterations are far lower than what the earlier protocols achieved upon convergence. The performance of the algorithm is illustrated via simulations. Simulations also indicate that the algorithm is robust to intermittent link and node failures.

I. INTRODUCTION

Accurate time or clock synchronization is critical in applications such as range finding for target tracking and localization, intrusion detection, time correlation of telemetry data, sensor fusion, slot assignment in TDMA, duty cycling protocols, and so on. Consequently, there has been a large amount of research on synchronization of wireless sensor networks; see [1] and references therein, for a survey of challenges, applications, and protocols. At a given “global” time t , the local clock time at node u can be approximately written as $t_u = \alpha_u t + \beta_u$, where α is the skew and β the offset [2]. In a network, time synchronization consists of estimating the skews and offsets of all the nodes relative to a global reference. In practice, these parameters drift slowly with time, and clocks must be periodically synchronized.

In a typical sensor network, only a handful of nodes may have access to GPS (recall that GPS performance indoors is problematic, and GPS is easily jammed). In principle, a designated source node can send time-stamped messages which are received with some delay at secondary nodes. There is a fixed delay due to propagation; but in addition, one must also cope with delay jitter, and randomness due to issues related to the MAC and internal processors. These in turn can cause asymmetric delays. Time-stamping at the PHY layer, as in the IEEE 1588 Precise Time Protocol (PTP) [3] can eliminate some of these variabilities. The 1588 standard has the notion of a master node, of a hierarchy to distribute PTP messages, and also the concept of (possibly) multiple nodes with high-quality clocks (the so-called boundary and transparent clocks). But the 1588 standard does not readily address the synchronization problem in an ad hoc network. Duty-cycling and RF propagation conditions can cause intermittent node

disconnects and link failures. Our interest is in synchronization of a wireless network in such an environment.

We assume PHY layer time-stamping of packets as in the 1588 standard, at the transmitter and at the receiver. Estimates of the relative offset between a pair of nodes u and v can be obtained by exchanging time-stamped packets. Algorithms for doing this are described in detail in [4], [5], [1], [6, Ch. 2]. In particular, when the skew is zero, a noisy measurement of the relative offset, $\beta_u - \beta_v$, can be obtained as

$$\eta_{u,v} = \beta_u - \beta_v + \varepsilon_{u,v}, \quad (1)$$

where $\varepsilon_{u,v}$ is a zero-mean measurement error. Similarly, noisy ratios of their skews, α_u/α_v can also be obtained. Upon taking logarithm, we get

$$\xi_{u,v} = \log(\alpha_u) - \log(\alpha_v) + \varepsilon'_{u,v}, \quad (2)$$

where $\xi_{u,v}$ is the logarithm of the measured ratio of clock skews, and $\varepsilon'_{u,v}$ is a zero-mean error. The details of obtaining such measurements are explained in [1] [6, Chapter 2]. Equations (1) and (2) are special cases of

$$\zeta_{u,v} = x_u - x_v + \epsilon_{u,v}, \quad (3)$$

where $\zeta_{u,v}$ is a noisy measurement of the “relative difference” between the unknown variables x_u and x_v , and $\epsilon_{u,v}$ is a zero-mean measurement error. For this reason, we restrict our attention to the problem of estimating scalar-valued variables from relative measurements of the type (3). For the sake of concreteness, we will frequently refer to offset estimation as a specific example of this problem. However, the problem considered here is also relevant to applications such as location estimation from relative position measurements [7].

Given a set of relative measurements of the form in (3), we seek to obtain estimates of the node variables x_u . One could consider sending the $\zeta_{u,v}$ measurements to a fusion center for joint estimation of the x_u 's. It is rare to have a star topology where nodes are within one hop of a fusion center. Typically, these data would have to be transmitted on a multi-hop route. Distributed schemes have been proposed to cope with failure of the fusion center, as well as intermittent loss of nodes and links. Given an initial set of relative measurements, several iterative algorithms have been proposed in [5], [8], [9], [10], [4]: At every iteration, a node receives the current estimates of its neighbors which are used to update its own estimate. All of these algorithms are essentially variants of the Jacobi algorithm for solving a linear system of equations [11].

These algorithms have several beneficial properties; if no errors are introduced in the message exchanges during the iterations, these algorithms converge to the best linear unbiased estimates that are possible with the initially gathered measurements; further they are robust to temporary node and link failures [8], [10], [12].

P. Barooah is with the Dept. of Mechanical and Aerospace Engineering, Univ. of Florida, Gainesville, FL; pbarooah@ufl.edu, and A. Swami is with the Army Research Lab, Adelphi, MD; a.swami@ieee.org.

During the iterative process, one could time stamp the packets used to communicate estimates; these potentially provide new measurements of relative offsets. How should one incorporate these new measurements? How do they affect convergence rate? How do they affect the final variance? Sensor networks are typically energy-constrained; thus increasing the rate of convergence, i.e., reducing the number of iterations, can prolong network lifetime. It is reasonable to assume that the measurement errors $\epsilon_{u,v}$ in (3) are uncorrelated from measurement to measurement. If all the data could be sent to a fusion center, one would obviously use a recursive least-squares algorithm. The challenge here is that a node has access to neither all of the new measurements, nor the current estimates of all of the nodes.

The problem formulation is stated precisely in Section II. Our proposed distributed algorithm to incorporate the new measurements is discussed in Section III. This algorithm is also motivated by the Jacobi algorithm; it updates a node estimate as a weighted average of the current estimates of its neighbors' states, the newly available relative measurements, and its old estimate. In Section IV, we show that under fairly mild assumptions, the resulting estimates are unbiased at every iteration and that they converge in the mean square sense (as the number of iterations increases) to the true parameter values. Simulation results in Section V will verify that for a fixed number of iterations, the error variances of the proposed estimator are orders of magnitude smaller than those produced by the earlier protocols in [5], [8], [10], [4]. We will also see that the proposed algorithm is robust to link and node failures.

II. PROBLEM STATEMENT

For the sake of clarity, we will consider only the scalar offset estimation problem. We refer to the unknown offsets, x_u , as the *node variables*. The network has a total of n_{total} nodes which we associate with the nodes $\mathcal{V} = \{1, 2, \dots, n_{total}\}$ of a directed *measurement graph* $\mathcal{G} = (\mathcal{V}, \mathcal{E})$. The measurements, $\zeta_{u,v}$ correspond to the edges in \mathcal{E} , where each edge consists of an ordered pair (u, v) such that a noisy relative measurement of the form (3) is available.

It is clear that the problem is under-determined (the set of equations in (3) has rank $n_{total} - 1$). Hence, one of the offsets must be known; equivalently, offsets are estimated relative to some specific node. Such a node could be a cluster-head or have access to GPS or some other accurate time source. It is possible that several nodes in the network have access to GPS; the corresponding node variables are all zero¹. We will refer to these known node variables as *reference variables*; we assume that there are r reference nodes, and hence $n = n_{total} - r$ unknown node variables. Without loss of generality, we number the unknown node variable as $1, \dots, n$, and the reference variables as $n + 1, \dots, n + r$; we denote the vertex set of the reference variables by \mathcal{V}_r . We will assume that communication is symmetric, i.e., if u can receive messages from v , v can also receive messages from u .

¹We assume that the reference nodes are mutually consistent.

A. Notation

Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ denote a directed graph, with of $n_{total} = |\mathcal{V}|$ nodes and $m = |\mathcal{E}|$ edges. The node-edge *incidence matrix* \mathbf{A} is a $n_{total} \times m$ matrix with entries 0, 1 and -1 defined as $A(u, e) = a_{u,e}$, where $a_{u,e} = 0$ if edge e is not incident on node u ; $a_{u,e} = +1$ if edge e is directed away from node u and $a_{u,e} = -1$ if edge e is directed towards node u . The *basis incidence matrix* A_b is the $n \times m$ sub-matrix of A that is obtained by removing the rows corresponding to the reference nodes. The sub-matrix of A consisting of those rows of A that correspond to the reference nodes is termed the *reference incidence matrix* A_r . Consider the *weighted graph* specified by the couple (\mathcal{G}, w) where \mathcal{G} is a graph and $w : \mathcal{V} \times \mathcal{V} \rightarrow \mathbb{R}^+$ is a weight function that assigns positive weights to those node pairs that have an edge and 0 to those that don't. Specifically, $w_{u,v} \geq 0$ for every pair $u, v \in \mathcal{V}$, and $w_{u,v} = w_{v,u} > 0$ if and only if $(u, v) \in \mathcal{E}$. If there are m edges, there are exactly m positive weights, which are denoted by w_1, \dots, w_m . The matrix $L_b := A_b W A_b^T$, where $W = \text{diag}(w_1, \dots, w_m)$, is called the *Dirichlet Laplacian matrix* of the weighted graph (\mathcal{G}, w) . The diagonal matrix M constructed from the diagonal entries of L_b is the *basis weighted degree matrix* and $N := M - L_b$ is a sub-matrix of the (weighted) adjacency matrix. Explicitly, $M(u, u) = \sum_v w_{u,v}$, and $N(u, u) = 0, N(u, v) = w_{u,v}$ for $u, v = 1, \dots, n$. If a node pair (u, v) is connected by an edge e , we will find it useful to denote x_v by $x_{e \setminus u}$ and x_u by $x_{e \setminus v}$.

B. The Jacobi algorithm

We noted earlier that many of the iterative algorithms are special cases of the *Jacobi algorithm* described in [8]. In this algorithm, a node obtains estimates from its one-hop neighbors and then updates its own estimate. With $\hat{x}_u^{(i)}$ denoting node u 's estimate of x_u at the i^{th} iteration, the algorithm can be written as

$$\hat{x}_u^{(i)} = \left(\sum_{v \in \mathcal{N}_u} \frac{1}{\sigma_{u,v}^2} \right)^{-1} \sum_{v \in \mathcal{N}_u} \frac{1}{\sigma_{u,v}^2} \left(\hat{x}_v^{(i-1)} + \zeta_{uv} \right), \quad (4)$$

where \mathcal{N}_u denotes the (one-hop) neighbors of u ; $\sigma_{u,v}^2$ is the variance of the measurement error $\epsilon_{u,v}$. The Jacobi algorithm has a simple interpretation. A node's initial naive estimate is to average its relative offset with all of its one-hop neighbors. It updates this estimate by a weighted sum of the estimates of its neighbors at each iteration. The algorithm can also be interpreted as a maximal ratio combiner. For simplicity we have assumed that the measurement error variances are independent of the iteration; but this is easily relaxed.

C. Beyond the Jacobi algorithm

At each iteration, a message exchange takes place between one-hop neighbors; in this process, neighboring nodes u and v could obtain a new measurement of the relative offset $\zeta_{u,v}^{(i)}$. We argued earlier that such measurements are uncorrelated across the index i . Thus at the end of iteration i , a node u has \mathcal{N}_u additional measurements of the form,

$$\zeta_{u,v}^{(i)} = x_u - x_v + \epsilon_{u,v}^{(i)}, \quad \forall v \in \mathcal{N}_u, \quad (5)$$

The measurement errors $\epsilon_{u,v}^{(i)}$ are uncorrelated at different index values, i.e., $E[\epsilon_{u,v}^{(i)}\epsilon_{u,v}^{(j)}] = 0$ if $i \neq j$. Under the assumption of symmetric communications, every node $v \in \mathcal{N}_u$ also has a new measurement, $\zeta_{u,v}^{(i)} = -\zeta_{u,v}^{(i)}$. Let $m := |\mathcal{E}|$ be the number of edges in \mathcal{G} ; then at each iteration we have m additional offset measurements. We will use symbol e to denote an edge as well as the index of that edge, i.e., $e = (u, v) \in \mathcal{E}$ as well as $e = 1, \dots, m$. We assume that the measurement errors are uncorrelated $E[\epsilon_e^{(i)}\epsilon_{\bar{e}}^{(j)}] = 0$ if $e \neq \bar{e}$, for all i and j . and that the variance $(\sigma_e^2)^{(i)} = E[(\epsilon_e^{(i)})^2]$ is known for every edge $e \in \mathcal{E}$,

The problem considered in this paper is to obtain an estimate $\hat{x}_u^{(i)}$, by linearly combining the sequence of its own past estimates, the sequences of the estimates of its one-hop neighbors, and the sequence of relative measurements obtained from its one-hop neighbors. We assume that there is a finite memory constraint so that all these sequences cannot be stored. We recall the assumptions of uncorrelated measurement errors and bidirectional communications.

III. ALGORITHM DESCRIPTION

1) *Recursive Jacobi Algorithm (RJU)*: The basic idea is to combine all available relative measurements and average them to obtain a single (reduced variance) relative measurement. We can then use the previously discussed Jacobi algorithm to update estimates. Thus, the nodes compute the time-average of the relative measurements

$$\xi_{u,v}(i) = \frac{1}{i} \sum_{k=1}^i \zeta_{u,v}(i) \quad (6)$$

which is a zero mean measurement of $x_u - x_v$, with variance $\frac{1}{i}\sigma_e^2$. Note that nodes only need to store the previous averaged measurement $\xi_{u,v}(i-1)$. Now define

$$\hat{x}_u^{(i)}(v) := \hat{x}_v^{(i)} + \xi_{u,v}(i) \quad \forall v \in \mathcal{N}(u) \quad (7)$$

which is an estimate of x_u that is obtained by combining the averaged measurement $\xi_{u,v}^{(i)}$ computed at time i with the current estimate of x_v .

The following recursive scheme to combine all these estimates is proposed:

$$\hat{x}_u^{(i)} = (1 - \beta_i)\hat{x}_u^{(i-1)} + \beta_i \frac{1}{d_u} \sum_{v \in \mathcal{N}_u} w_{u,v} \hat{x}_u^{(i-1)}(v), \quad (8)$$

where, $0 < \beta_i \leq 1$, w_e is a positive weight associated with the edge e and $d_u = \sum_{e \in \mathcal{E}_u} w_e$ is the *weighted degree* of node u . The weights $w_{u,v}$ are design variables. Note that two neighbors u and v must use the same weights $w_{u,v} = w_{v,u}$. It should be understood that when one of the neighboring nodes of u is a reference node, the reference value is used for the neighbor's previous estimate. Note that in the original Jacobi algorithm w_e corresponds to $1/\sigma_e^2$.

We can also write equation (8) in terms of the incidence matrix as

$$\hat{x}_u^{(i)} = (1 - \beta_i)\hat{x}_u^{(i-1)} + \frac{\beta_i}{d_u} \sum_{\substack{e \in \mathcal{E} \\ e \in \mathcal{E}_u}} w_e \left(\hat{x}_{e \setminus u}^{(i-1)} + a_{u,e} \xi_e^{(i-1)} \right) \quad (9)$$

The scalar β and the weights $w_e, e \in \mathcal{E}$ are design variables. In the Jacobi algorithm (4), the weight w_e on an edge e is set equal to the inverse of the variance of the measurement error σ_e^2 associated with that edge. In the proposed algorithm, a similar rule can be followed by the nodes. Specifically, the variance of the filtered measurement $\xi_e(i)$ at iteration i is $\frac{1}{i}\sigma_e^2$, assuming the errors $\{\epsilon_e(i)\}_{i=0}^{\infty}$ on the edge e are wide sense stationary with variance σ_e^2 . In that case, a node can set weight w_e of an edge e incident on it as $w_e = i/\sigma_e^2$. Due to the form of (9), this is equivalent to setting $w_e = 1/\sigma_e^2$, just as in the Jacobi algorithm.

A. Asynchronous implementation

The description above assumes that every node shares the same iteration counter i , thereby making the algorithm synchronous. In practice, an asynchronous iteration is preferable where nodes do not have to share the same iteration counter but can incorporate new estimates arriving from different neighbors in different times without any regard to their arrival times. The most important benefit of an asynchronous operation is that random communication faults can be tolerated.

To implement the algorithm in an asynchronous mode, a ‘‘time out’’ period can be introduced so that every node waits for this time to receive messages from its neighbors. If messages from a neighbor are not received within the time-out period, the most resent estimate received from that neighbor is used in the updates. Note that when no messages are received from a neighbor within a time-out period, no additional measurements are gathered, either. Therefore, every node u keeps track of the number of measurements of $x_u - x_v$ gathered until the current time, which determines the variance of the averaged measurement $\xi_{u,v}(i)$ gathered by u until iteration counter i , though the counter i is local to u in the asynchronous case. Accordingly, the nodes also have to change the weights w_e 's at every iteration as a function of the number of measurements gathered on that edge till that iteration.

IV. ANALYSIS OF THE ALGORITHM

We analyze the algorithm proposed in the previous section. We compactly express the update law (9) for the vector of all node estimates $\hat{\mathbf{x}}^{(i)} = [\hat{x}_1^{(i)}, \dots, \hat{x}_n^{(i)}]^T$. The temporal evolution of this estimate is expressed by

$$\hat{\mathbf{x}}^{(i)} = J\hat{\mathbf{x}}^{(i-1)} + B\bar{\boldsymbol{\xi}}^{(i-1)} \quad (10)$$

where

$$J = I - \beta M^{-1} L_b, \quad B(i) = \beta M^{-1} A_b W, \quad (11)$$

$$\bar{\boldsymbol{\xi}}^{(i)} := \boldsymbol{\xi}^{(i)} - A_r^T \mathbf{x}_r, \quad \boldsymbol{\xi}^{(i)} = [\xi_1^{(i)}, \dots, \xi_m^{(i)}]^T,$$

and M, N, A_b and A_r are the basis degree, basis adjacency, basis incidence, and reference incidence matrices for the weighted graph (\mathcal{G}, w) , while $\mathbf{x}_r := [x_{n+1}, \dots, x_{n_{total}}]^T$ is the vector of reference variables. (See Section II-A for definitions.)

We state our main result next (proof is given in the Appendix)

Theorem 1: Consider the recursive algorithm (8) carried out by

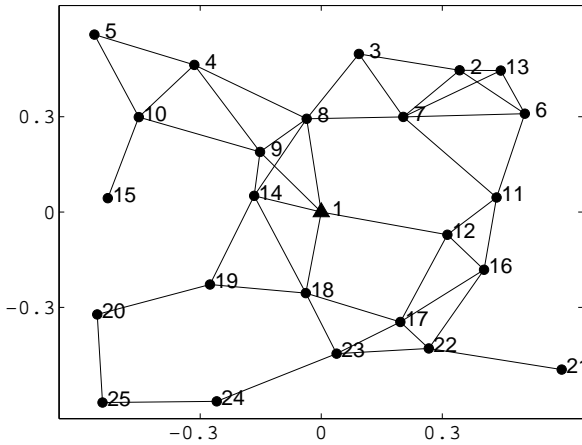


Fig. 1. The measurement graph for the 25 node simulation. All communications are bidirectional.

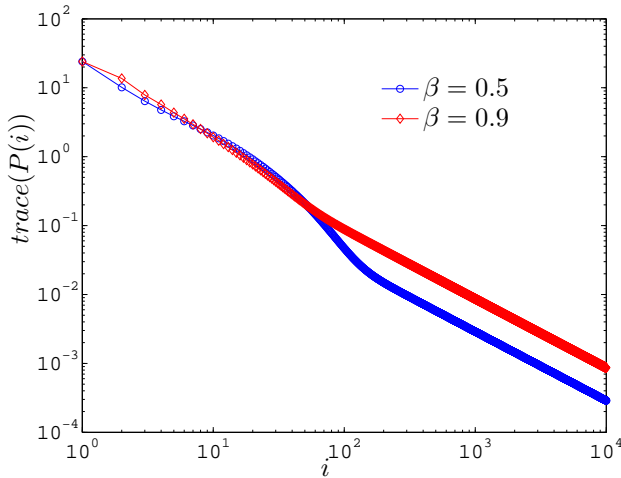


Fig. 2. The trace of the estimation error covariance matrix $P(i)$ as a function of the iteration counter i for the network shown in Figure 1. The covariance $P(i)$ is computed from the recursive relationship (12), with initial condition chosen as the identity matrix. Time evolution of the variances for two values of β are shown.

the nodes of a connected graph \mathcal{G} in a synchronous manner. The estimate $\hat{\mathbf{x}}(i)$ is unbiased at every iteration i , as long as the initial condition $\hat{\mathbf{x}}(0)$ is an unbiased estimate of \mathbf{x} . If the measurement errors $\epsilon(i)$ are wide sense stationary with $E[\epsilon^{(i)}\epsilon^{(j)T}] = \delta(i-j)R_0$, the estimate $\hat{\mathbf{x}}(i)$ converges to the true value \mathbf{x} in the mean square sense. \square .

V. NUMERICAL INVESTIGATION

Figure 1 shows a measurement graph consisting of a 25 node network. There is one reference node, at $(0, 0)$. Figure 2 shows the sum of the variances of the offset estimation errors (for the network shown in 1) as a function of the iteration index i . This sum is computed from the trace of the covariance matrix $P(i)$, where $P(i)$ is evaluated from the recursive relation (12). This plot provides numerical evidence in support of theorem 1.

In conducting simulations, skew rates of all the clocks are set at 1 for the sake of simplicity. Time offsets for the 24

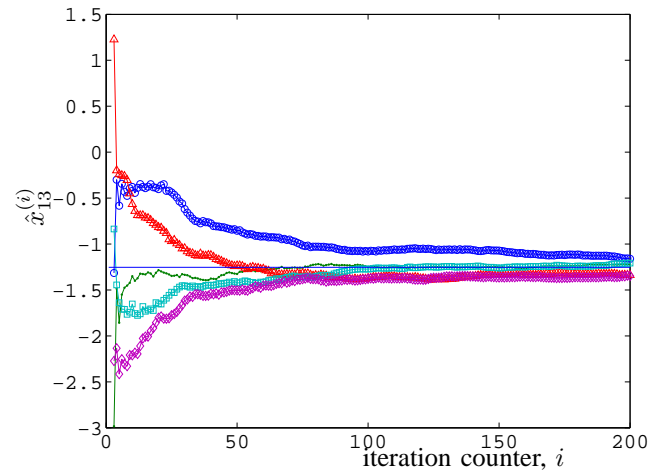


Fig. 3. Five sample runs of the estimates at node 13 (upper right hand corner node in Figure 1) as a function of time, with $\beta = 0.9$. The solid line shows the true value of node 13's time offset with respect to the reference.

nodes with respect to the reference node are fixed arbitrarily, and the algorithm described in Section III is simulated in MATLAB in a synchronous manner, i.e., all nodes shared a common iteration counter. The “flagged initialization” scheme described in [9] was used for initializing the estimates. Due to lack of space, we do not describe this scheme, but merely note that it is a method of providing good initial estimates to the nodes without requiring additional communication or computation, and refer the interested reader to [7], [9] for details. Simulations are done with a wide sense stationary, Gaussian distributed, measurement error sequence with zero mean and variance $\sigma_e^2(i) = \sigma_0^2 = 1$ for every $e \in \mathcal{E}$ and every $i = 1, \dots$. The edge weights w_e are chosen as the inverse of the measurement error variance on e (as it is done in the Jacobi algorithm [7]).

Figure 3 shows five distinct sample runs of the estimate of one of the nodes in the network shown in Figure 1 (node #13). Each sample run is obtained by running the algorithm with a distinct sequence of pseudo-random numbers generated in MATLAB. The convergence of the estimates in each run to the true value is visible in the plot.

Next, we empirically evaluate the error covariances from Monte-Carlo simulations. Figure 4 shows the empirically estimated variance of node 13 (see Figure 1 for the node's location). It also shows the minimum variance that can be achieved by the centralized optimal estimator. We see that the asymptotic variance of the proposed distributed estimators is quite close to the best possible, i.e., that of the centralized estimator. In addition, the figure shows the variance that can be obtained by the Jacobi algorithm in [8] (and its variants described in [5], [9], [10], [4]) upon convergence. We see from Figure 4 that the error variance $\sigma_u^2(i)$ of the proposed estimator becomes, within a few iterations, much smaller than the limiting variance that the Jacobi algorithm can achieve with a single set of measurements. We conclude that significant improvement in accuracy is possible by employing the proposed method.

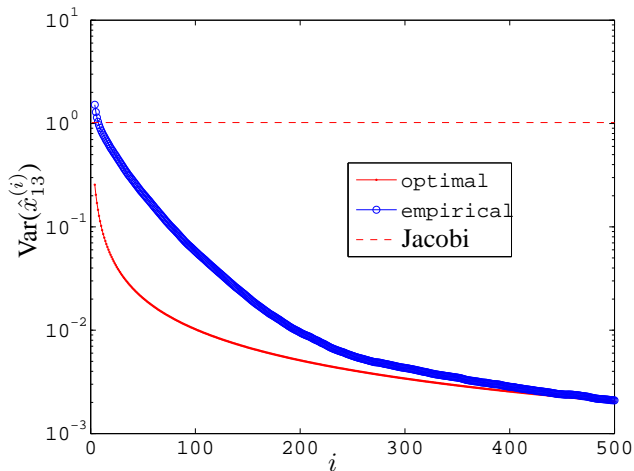


Fig. 4. The temporal evolution of the variance of the estimation at node #13 (shown in Figure 1). The legend “empirical” refers to the empirically estimated error variance of the node’s estimates obtained by the proposed algorithm, with $\beta = 0.3$. The estimates were averaged over 100 sample runs. “Jacobi” refers to the steady-state variance the estimates produced by the Jacobi algorithm in [8] and its variants in [5], [10]

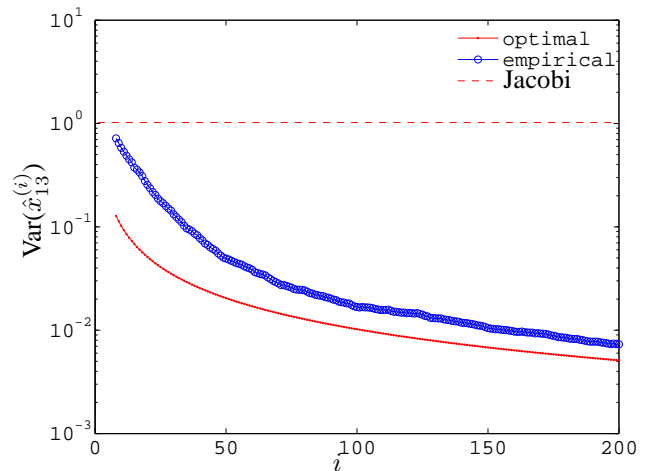


Fig. 6. Evolution of node 13’s estimation error variance with random communication faults. At every iteration, every communication fails independently of all other edges, with a probability of 0.3. The legend “empirical” refers to the empirically estimated (from 100 sample runs) error variance of the node’s estimates produced by the proposed algorithm. “Jacobi” refers to the variance the estimates produced by the Jacobi algorithm in [8] and its variants in [5], [10] achieve upon convergence without communication faults.

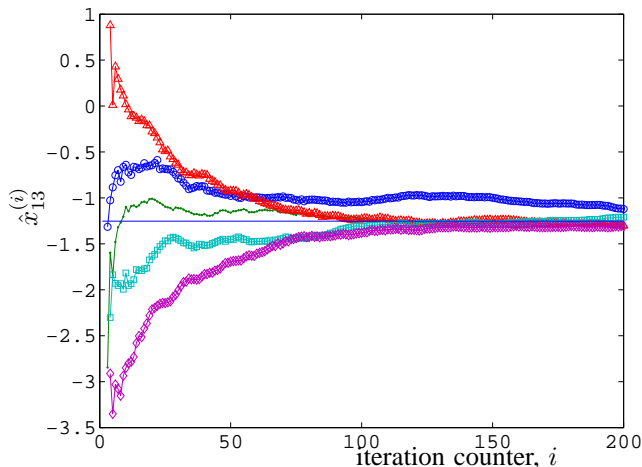


Fig. 5. Five sample runs of the estimates at node 13 (upper right hand corner node in Figure 1) as a function of time in the presence of random communication faults (probability of failure is 0.3), with $\beta = 0.9$. The solid line shows the true value of node 13’s time offset with respect to the reference.

Next, we consider the scenario when communication between two arbitrary nodes u and v (that have an edge between them) fails with a probability p , independent of all other node pairs. Figure 5 shows five sample runs of node #13’s estimate when the simulation is done in an asynchronous manner in the presence of random communication failures, with $p = 0.3$. The evolution of the estimation error variance of node 13’s offset estimate, evaluated empirically from Monte Carlo simulations (100 sample runs) is shown in Figure 6. As expected, the convergence of the error variance toward 0 is slower with random communication failures than in the case with no communication failure; cf. Figure 4.

VI. SUMMARY

We proposed a distributed algorithm to estimate clock offsets and skews in wireless sensor networks for accurate time synchronization. It takes advantage of noisy measurements of the difference between time offsets (or skews) that can be obtained while exchanging messages between neighboring pairs of nodes. We showed that the estimator is unbiased and converges in mean-square. Numerical simulations show the proposed algorithm produces estimates with much lower variance within a few iterations than algorithms proposed in [5], [8], [9], [10], [4].

There are several aspects of the algorithm that needs further study. We proved that the estimates converges in the mean square sense when there are no communication faults. Although the algorithm seems to perform well (error variance decays monotonically) in simulations, its convergence property in the presence of random node and link failures, and asynchronous mode of operation, remains to be examined. In addition, our analysis assumed that communication among node pairs is symmetric. In wireless communication, asymmetric communication – such that u can send data to v but cannot receive from v – is quite possible. The behavior of the proposed algorithm to such asymmetry needs to be studied, which can perhaps be done along the lines of our earlier work [12].

REFERENCES

- [1] B. M. Sadler and A. Swami. Synchronization in sensor networks: an overview. In *IEEE MILCOM*. 2006.
- [2] S. Ganeriwal, R. Kumar and M. B. Srivastava. Timing-sync protocol for sensor networks. In *ACM Conference on Embedded Networked Sensor Systems (SenSys)*. 2003.
- [3] IEEE I&M Society, TC9: “Precision clock synchronization protocol for networked measurement and control systems”, IEEE 1588, IEC 615888, Sept 2004.

- [4] R. Solis, V. S. Borkar and P. R. Kumar. A new distributed time synchronization protocol for multihop wireless networks. In *Proc. of the 45th IEEE Conference on Decision and Control*. 2006.
- [5] R. Karp, J. Elson, D. Estrin and S. Shenker. Optimal and global time synchronization in sensor networks. Tech. rep., Center for Embedded Networked Sensing, Univ. of California, Los Angeles, 2003.
- [6] P. Barooah. *Estimation and Control with Relative Measurements: Algorithms and Scaling Laws*. Ph.D. thesis, University of California, Santa Barbara, 2007.
- [7] P. Barooah and J. P. Hespanha. Estimation from relative measurements : Algorithms and scaling laws. *IEEE Control Systems Magazine*, vol. 27, no. 4: 57 – 74, 2007.
- [8] P. Barooah and J. P. Hespanha. Distributed optimal estimation from relative measurements. In *Proc. 3rd Int'l Conf. Intelligent Sensing and Information Processing (ICISIP)*, pp. 226–231. 2005.
- [9] P. Barooah, N. M. da Silva and J. P. Hespanha. Distributed optimal estimation from relative measurements for localization and time synchronization. In P. B. Gibbons, T. Abdelzaher, J. Aspnes and R. Rao (editors), *Distributed Computing in Sensor Systems DCOSS*, vol. 4026 of *LNCS*, pp. 266 – 281. Springer, 2006.
- [10] A. Giridhar and P. R. . Kumar. Distributed time synchronization in wireless networks: Algorithms and analysis (I). In *45th IEEE Conference on Decision and Control*. 2006.
- [11] G. H. Golub and C. F. van Loan. *Matrix Computations*. The John Hopkins University Press, 3rd edn., 1996.
- [12] P. Barooah, J. P. Hespanha and A. Swami. On the effect of asymmetric communication on distributed time synchronization. In *46th IEEE Conference on Decision and Control*, pp. 5465 – 5471. 2007.
- [13] A. Berman and R. J. Plemmons. *Nonnegative Matrices in the Mathematical Sciences*. Computer Science and Applied Mathematics. Academic Press, 1979.
- [14] W.-K. Chen. *Applied Graph Theory*. North Holland Publishing Company, 1971.

APPENDIX

We state a result that will be useful in proving Theorem 1:
Proposition 1: For a connected undirected graph \mathcal{G} , and J defined by (11), we have

$$\rho(J) < 1,$$

where $\rho(\cdot)$ denotes the spectral radius. \square

We will need the following terminology. For a matrix A , we write $A \succeq (\succ) 0$ to mean that A is entry-wise non-negative (positive). For a vector x , we write $x \succeq 0$ to mean x is entry-wise non-negative, and $x \succ 0$ to mean x is entry-wise non-negative and at least one entry is positive. We will need the following result to prove Proposition 1:

Proposition 2 [Theorem (5.2), pp. 181 of [13]] Let $Z = X - Y$ with Z and X non-singular, and suppose $X^{-1}Y \succeq 0$. Then $\rho(X^{-1}Z) < 1$ if and only if $Z^{-1}Y \succeq 0$, in which case,

$$\rho(X^{-1}Y) = 1 - \frac{1}{1 + \rho(Z^{-1}Y)} \quad \square.$$

Proof of Proposition 1: It is straightforward to see that by construction, $L_b \mathbf{1} \succ 0$, where $\mathbf{1}$ is the vector of all 1's with appropriate dimension (all entries of the vector $L_b \mathbf{1}$ are zero except those corresponding to those nodes that have edges connecting them to at least one reference). $L_b = M - N = A_b W A_b^T$ is positive definite since $W = W^T > 0$ and A_b is full column rank for a connected graph [14]. From [13, condition C₁₃, pp. 135], we get that L_b is a non-singular M -matrix, which also ensures that $L_b^{-1} \succ 0$ and therefore $L_b^{-1}N \succeq 0$ [13, condition N₃₈, pp. 137]. Now

applying Proposition VI we see that $\rho(J) = \rho(M^{-1}N) < 1$, which proves the proposition. \square

Proof of Theorem 1: The statement about unbiasedness of the resulting estimates follows immediately upon taking expectation of both sides of (10). The covariance of the state estimation error at iteration i is $P(i) := \mathbb{E}[(\mathbf{x} - \hat{\mathbf{x}}^{(i)})(\mathbf{x} - \hat{\mathbf{x}}^{(i)})^T]$. The iteration (10) is a discrete time LTI system driven by white noise whose covariance decreases linearly with time. It follows from straightforward calculation that, as long as the measurement noise process $\{\epsilon^{(i)}, i = 1, \dots\}$ is white with $\mathbb{E}[\epsilon^{(i)} \epsilon^{(j)T}] = \delta(i - j)R_0$, the covariance $P(i)$ evolves according to:

$$P(i) = JP(i-1)J^T + BR_i B^T, \quad R_i = \frac{1}{i}R_0 \quad (12)$$

with initial condition $P(0) := \mathbb{E}[(\mathbf{x} - \hat{\mathbf{x}}^{(0)})(\mathbf{x} - \hat{\mathbf{x}}^{(0)})^T]$.

Define $\mathbf{p}_i := \text{vec}(P(i))$, where $\text{vec}(A)$ of a matrix A is the tall column vector obtained by stacking all columns of the matrix vertically. Similarly, define $\mathbf{r}_i = \text{vec}(R(i))$. Using the $\text{vec}(\cdot)$ operation, the estimation error covariance dynamics (12) can be written as

$$\mathbf{p}_i = (J \otimes J)\mathbf{p}_{i-1} + (B \otimes B)\mathbf{r}_{i-1},$$

where \otimes denotes the Kronecker product. Define $F := J \otimes J$, $G := B \otimes B$, and $\mathbf{y} := e_u^T \mathbf{p}$, where $e_u = [0 \dots 1 \dots 0]$ is an unit vector with all zeros and a single 1 at the u^{th} location. Now consider the discrete time linear time invariant system

$$\begin{aligned} \mathbf{p}_{i+1} &= F\mathbf{p}_i + G\mathbf{r}_i, \\ \mathbf{y}_i &= e_u^T \mathbf{p}_i \end{aligned} \quad (13)$$

whose output is the variance of node u 's estimation error, and is driven by an input sequence $\{\mathbf{r}_i\}$. Since $F = J \otimes J$, $\rho(F) = \rho(J)^2$, and therefore $\rho(F) < 1$ by Proposition 1. Since the matrix F has all eigenvalues strictly inside the unit circle, the linear time invariant system (13) has a finite \mathcal{H}_∞ norm from the input \mathbf{r} to output \mathbf{y} , which implies that there exists a positive constant $c < \infty$ such that

$$\sup_{\mathbf{r}} \frac{\|\mathbf{y}\|_{L_2}}{\|\mathbf{r}\|_{L_2}} \leq c,$$

where $\|\mathbf{y}\|_{L_2} = \sum_i |y_i|^2$ is the L_2 -norm of the signal $\{y_i\}_{i=0}^\infty$ and the supremum is evaluated over all input sequences of finite L_2 -norm. The sequence \mathbf{r}_i is square-summable, since $\mathbf{r}_i = \frac{1}{i} \text{vec}(R_0)$, which follows immediately from (6). The above implies that the sequence $\mathbf{y}(i)$ is square-summable too, therefore $\mathbf{y}(i) \rightarrow 0$ as $i \rightarrow \infty$. However, $\mathbf{y}(i) = \mathbb{E}[(x_u - \hat{x}_u(i))^2]$ is the variance of the estimation error in $\hat{x}_u(i)$. This proves that $\hat{x}_u(i) \rightarrow x_u$ in m.s. The same argument can be repeated for every node by choosing y appropriately. This proves the theorem. \square