DiSync: Accurate Distributed Clock Synchronization in Mobile Ad-hoc Networks from Noisy Difference Measurements

Chenda Liao and Prabir Barooah

***Abstract*— To perform clock synchronization in mobile ad-hoc networks, nodes need to estimate current time of a global clock based on the readings of their local clocks along with time-stamped messages from their neighbors. We formulate the problem as nodes simultaneously estimating skews and offsets of their own clocks with respect to a global clock from noisy difference measurements of logarithm of skews and that of offsets. These measurements can be obtained by exchanging time stamped messages. A leader-following consensus-based algorithm is proposed to estimate these parameters in a distributed manner. Ideas from stochastic approximation are used to ensure mean square convergence of estimation error under certain conditions. A sequence of scheduled update times is used to meet the requirement of specific decreasing time-varying gains that need to be synchronized across nodes with unsynchronized clocks. Simulations indicate that high accuracy of global time estimation can be maintained for long time duration with the proposed algorithm. Performance of the proposed algorithm is compared through simulations with the virtual clock synchronization algorithm ATS [1]. It is seen that the proposed algorithm is more robust than ATS to measurement noise resulting from random delays in message exchange.**

I. Introduction

Clock synchronization is extremely important for the functionality and performance of wireless ad-hoc networks and sensor networks. In TDMA-based communication schemes, accurate time synchronization ensures that each node communicates with others in their own time slots without interfering with others. Operation on a pre-scheduled sleep-wake cycle for energy conservation in sensor networks also requires all nodes to share a common notion of time. However, clocks run at different speeds due to imperfectness of quartz crystal oscillators. Even if they were to have the same speed, each clock may start at different time instants which leads to a difference in their local times.

The local time of node u when the global time is t, $\tau_u(t)$, is usually modeled as:

$$\tau_u(t) = \alpha_u t + \beta_u, \tag{1}$$

where the scalar parameters α_u, β_u are called its *skew* (speed of clock) and *offset*, respectively [2]. In practice, skews are time-varying due to temperature change, aging etc. However, it is common to model the skew of a clock as a constant since its variation is negligible during time intervals of interest [3]. The global time is the Coordinated Universal Time (UTC), or the local time of one of the clocks that is elected as a reference when none of the clocks can access the UTC. A clock can determine the global time t from its local time by using the relationship $t = (\tau_u(t) - \beta_u)/\alpha_u$ as long as it knows its skew and offset. Hence the problem of the clock synchronization can be alternatively posed as the problem of nodes estimating their skews and offsets. A node u can use a *pairwise synchronization* method, such as those in [4]–[6], to estimate α_u and β_u if it directly communicates to the reference node. However, this is not the case for most of the nodes due to limited communication range. It is therefore not possible for all nodes to measure their skews and offsets directly.

Network-wide clock synchronization in ad-hoc networks has been intensely studied in recent years. Work in this area can be grouped into three categories: cluster-based protocols, tree-based protocols and distributed protocols. In cluster-based [7] and tree-based protocols [8], [9], synchronization relies on establishing a pre-specified network structure. In mobile networks, however, network topology continually changes, which results in frequent re-computation of a cluster and spanning tree, or re-election of a root node. This introduces considerable communication overhead to the networks, therefore the above cluster-based and tree-based protocols are primarily targeted to networks of static or quasi-static nodes.

Recently, a number of fully distributed algorithms that do not require the establishment of clusters or trees have been proposed. These typically perform synchronization by estimating skews and/or offsets and then computing the global time from them. The algorithms proposed in [6], [10]–[15] belong to this category. These distributed protocols are more readily applicable to mobile networks than the previous two. However, little is known about how such algorithms will perform in mobile networks. In [16], an algorithm similar to that in [10]–[12] is proposed, and its performance is analyzed for mobile networks. It was shown that the variances of estimation errors of skews and offsets converge to positive values when suffering from measurement noise. However, even a small error in the skew estimates is likely to cause large error in the estimate of global time t for large values of t. Thus, frequent restarting of the synchronization process may be needed with such an algorithm.

In this paper, similar to [16], we formulate the clock synchronization problem as the estimation of skews and offsets using noisy difference measurements of log-skews and offsets. The two types of measurements can be computed by employing existing pairwise synchronization protocols via exchanging time-stamped messages. We propose a distributed algorithm (DiSync) which ensures the variances of

C. Liao and P. Barooah are with the Dept. of Mechanical and Aerospace Engineering, Univ. of Florida, Gainesville, FL; {cdliao,pbarooah}@ufl.edu.
This work has been supported by the National Science Foundation by Grants CNS-0931885 and ECCS-0955023.

the estimation errors in skews and offsets converge to zero under mild assumption on node mobility, etc. In addition, we provide a formula of the limiting bias of the estimates, which requires further information on the switching sequence of the graphs. Time varying gains in the algorithm that make the variances converge to 0 are adopted from stochastic approximation, used in consensus to attenuate noise [17], [18]. This leads much more accurate estimates of global times compared to the algorithms mentioned earlier. The gains need to vary in a specific manner with time, which poses challenges to implement in a network of unsynchronized clocks. This is addressed by using an iteration schedule so that nodes can effectively perform a synchronous update without having synchronized clocks.

A new type of *virtual* time-synchronization protocols has been proposed recently. They let nodes estimate a common virtual global time that may not be related to the time of any clock [1], [19]. These algorithms are potentially applicable to mobile networks. However, these approaches are not useful when nodes want to know a true global time, not just a virtual one.

We evaluate the accuracy of the DiSync algorithm when applying to global time estimation through Monte Carlo simulations. Simulations indicate the error of the global time estimate stays close to 0 for long time intervals. We also compare the result with ATS algorithm proposed in [1]. Although ATS does not provide a true global time, we compare the two algorithms in terms of the maximum synchronization error - the maximum deviation in the estimates of (virtual or true) global time over two arbitrary nodes. It turns out that the proposed DiSync algorithm outperforms ATS under this metric.

II. Problem formulation

The clock synchronization problem is formulated as nodes estimating their skews and offsets. Most of the nodes cannot estimate their skews and offsets directly from the reference node(s) due to limited range of communication. However, it is possible for a pair of nodes u, v, who can communicate with each other, to estimate their *relative skew* $\alpha_{u,v} := \frac{\alpha_u}{\alpha_v}$ and *relative offset* $\beta_{u,v} := \beta_u - \beta_v \frac{\alpha_u}{\alpha_v}$. The reason for this terminology is the following relationship $\tau_u(t) = \frac{\alpha_u}{\alpha_v}\tau_v(t) + \beta_u - \beta_v \frac{\alpha_u}{\alpha_v}$, derived from (1). The estimation of relative skews and offsets is called "pairwise synchronization". Several protocols for pairwise synchronization exist using time-stamped messages [4]–[6]. We assume nodes can estimate relative skews and offsets by using one of these existing protocols.

Suppose between a pair u and v, node u obtains noisy estimates $\hat{\alpha}_{u,v}, \hat{\beta}_{u,v}$ of the parameters $\alpha_{u,v}, \beta_{u,v}$ by using a pairwise synchronization protocol. We model the noisy estimate as $\hat{\alpha}_{u,v} = \alpha_{u,v} + e^s_{u,v}$, where $e^s_{u,v}$ is the estimation error. Therefore, by $\alpha_{u,v} = \frac{\alpha_u}{\alpha_v}$,

$$\log \hat{\alpha}_{u,v} = \log \alpha_u - \log \alpha_v + \xi^s_{u,v}, \tag{2}$$

where $\xi^s_{u,v} = \log(1 + e^s_{u,v}\frac{\alpha_v}{\alpha_u})$. The quantity obtained from pairwise synchronization is therefore a noisy difference measurement of log-skews. If $\alpha_v/\alpha_u \approx 1$, which is usually the case, and $e^s_{u,v}$ is small, then the measurement noise $\xi^s_{u,v}$ is small. Similarly, the noisy estimate of relative offset is modeled as $\hat{\beta}_{u,v} = \beta_{u,v} + e^o_{u,v}$, where $e^o_{u,v}$ is the error. Again, by $\beta_{u,v} = \beta_u - \beta_v \frac{\alpha_u}{\alpha_v}$,

$$\hat{\beta}_{u,v} = \beta_u - \beta_v + \xi^o_{u,v}, \tag{3}$$

which is a noisy difference measurement of the offsets between the two nodes, with measurement noise $\xi^o_{u,v} = \beta_v(1 - \frac{\alpha_u}{\alpha_v}) + e^o_{u,v}$. Due to the term $\beta_v(1 - \frac{\alpha_u}{\alpha_v})$, the measurement error is biased even if $e^o_{u,v}$ is zero mean. Since $\frac{\alpha_u}{\alpha_v}$ is close to 1 for most clocks, the bias is usually small.

We see from (2) and (3) that $\log \hat{\alpha}_{u,v}$ and $\hat{\beta}_{u,v}$ are the noisy measurements of log-skew difference $\log \alpha_u - \log \alpha_v$ and offset difference $\beta_u - \beta_v$, respectively. We now seek to estimate the log-skews and offsets of all the nodes in a distributed manner from these noisy pairwise difference measurements. Note that once a node estimate its log-skew, it can recover the skew. Once an estimate of skew and offset is obtained, it can compute the global time from its local time.

To facilitate further discussion, we only consider the estimation of scalar valued node variables from noisy difference measurements. If an algorithm of solving this problem is available, two copies of the algorithm can be executed in parallel to obtain both skews and offsets. Let u-th node in a n-node network have an associated scalar *node variable* $x_u \in \mathbb{R}$, $u \in \mathcal{V} = \mathcal{V}_b \cup \mathcal{V}_r = \{1, \ldots, n\}$. Nodes in $\mathcal{V}_b = \{1, \ldots, n_b\}$ do not know their node variables, while the reference nodes are the remaining n_r nodes in $\mathcal{V}_r = \{n_b + 1, \ldots, n\}$. Here x_u represents $\log(\alpha_u)$ for skew estimation and β_u for offset estimation. Time is measured by a discrete time-index $k = 0, 1, \ldots$. The mobile nodes define a time-varying undirected *measurement graph* $\mathcal{G}(k) = (\mathcal{V}, \mathcal{E}(k))$, where $(u, v) \in \mathcal{E}(k)$ if and only if u and v can obtain a difference measurement of the form

$$\zeta_{u,v}(k) = x_u - x_v + \xi_{u,v}(k), \tag{4}$$

during the time interval between k and $k + 1$, where $\xi_u(k)$ is measurement error. We assume that between u and v, whoever obtains the measurement first shares it with the other so that it is available to both u and v. We also follow the convention that the difference measurement between u and v that is obtained by the node u is always of $x_u - x_v$ while that used by v is always of $x_v - x_u$. Since the same measurement is shared by a pair of neighboring nodes, if v receives the measurement $\zeta_{u,v}(k)$ from u, then it converts the measurement to $\zeta_{v,u}(k)$ by assigning $\zeta_{v,u}(k) := -\zeta_{u,v}(k)$. For similar reasons, between a pair u and v, the node who computes $\zeta_{u,v}(k)$ in node pair u and v is fixed for all time k. This can be achieved by comparing the magnitude of the index of nodes. For example, if $u > v$, then u computes $\zeta_{u,v}(k)$ first and then sends it to v. The *neighbors* of u at k, denoted by $\mathcal{N}_u(k)$, is the set of nodes that u has an edge with in the measurement graph $\mathcal{G}(k)$. We assume that if $v \in \mathcal{N}_u(k)$, then u and v can also exchange information through wireless communication at time k.

Now the reformulated problem is to estimate the node variables x_u, $u \in \mathcal{V}_b$, by using the difference measurements $\zeta_{u,v}(k), (u,v) \in \mathcal{E}(k)$ that become available over time. We assume $n_r \geq 1$ (i.e., there exists a least one reference node), otherwise the problem is indeterminate up to a constant.

III. The DiSync algorithm

We first present an iterative algorithm that nodes can use to solve the problem of node variable estimation from noisy difference measurements in a distributed manner. Since nodes do not have synchronized clocks, iterative updates have to be performed asynchronously. Each node $u \in \mathcal{V}_b$ keeps its local *iteration index* k_u and maintains an estimate $\hat{x}_u(k_u) \in \mathbb{R}$ of its node variable x_u in its local memory. The estimates can be initialized to arbitrary values. In executing the algorithm, node u starts its i-th iteration at a pre-specified local time $\tau^{(i)}$, for $i = 0, 1, \ldots$, which will be described in Section III-.1. Then, node u obtains current estimates $\hat{x}_v(k_u)$ along with the measurements $\zeta_{u,v}(k_u)$ from its current neighbors $v \in \mathcal{N}_u(k_u)$. After a fixed time length δt (measured in local time), node u updates its new estimate based on current measurements and neighbors' estimates by using the following update law:

$$\hat{x}_u(k_u+1) = \hat{x}_u(k_u) + m(k_u) \sum_{v \in \mathcal{N}_u(k_u)} a_{uv}(k_u)(\hat{x}_v(k_u) + \zeta_{u,v}(k_u) - \hat{x}_u(k_u)); \tag{5}$$

where the time varying gain $m(\cdot) : \mathbb{Z}^+ \to \mathbb{R}^+$ has to be specified to all nodes a-priori. Note that when $\mathcal{N}_u(k_u) = \emptyset$, $\hat{x}_u(k_u+1) = \hat{x}_u(k_u)$. The choice of $m(\cdot)$ will play a crucial role in the convergence of the algorithm and will be described in Section IV. In this paper, we let weight $a_{uv}(k_u) = 1$ if $(u,v) \in \mathcal{E}(k_u)$. The reference nodes take part by helping their neighbors obtain difference measurements, and keep their variables at 0 for all the time. After the update, node u increments its local iteration index k_u by 1. The algorithm is summarized in Algorithm 1. Note that since obtaining difference measurements requires exchanging time-stamped messages, current estimates can be easily exchanged during the process of obtaining new measurements.

1) Iteration schedule and synchronous view: We will later describe that the gains $m(\cdot)$ is chosen to be a decreasing function of time, which helps reduce the effect of measurement noise. This is a well-known idea in stochastic approximation. However, using this idea in a network of unsynchronized clocks presents an unique challenge since no node has a notion of a common time index, at least in the initial phase when they do not have good estimates. If nodes waits for a constant length of time (measured in their local clocks) before starting a new iteration, a node with faster skew might finish the $(i+1)$-th iteration while a node with slower skew hasn't even finished the i-th iteration. Therefore, specifying a function $m(\cdot)$ to all the nodes does not ensure that nodes use the same gain at the same (global) interval, which is required for the theoretical guarantees of stochastic approximation to hold.

Algorithm 1 DiSync algorithm at node u

```
1:  while u is performing time synchronization do
2:    if Local time τ_u = τ^(i), i = 0, 1, ... then
3:      u collects current local indices k_v from neighbors v ∈ N_u(k_u).
4:      for all v ∈ N_u(k_u) do
5:        if k_u = k_v and u does not have ζ_{u,v}(k_u) then
6:          1.u and v perform pairwise communication;
7:          2.u saves ζ_{u,v}(k_u) and x̂_v(k_u); v saves ζ_{v,u}(k_v) and
            x̂_u(k_v);
8:        else
9:          u and v stop the communication;
10:       end if
11:     end for
12:   end if
13:   if τ_u = τ^(i) + δt, i = 0, 1, ... then
14:     if N_u(k_u) ≠ ∅ then
15:       u updates x̂_u(k_u + 1) using (5);
16:     else
17:       x̂_u(k_u + 1) = x̂_u(k_u);
18:     end if
19:     u updates, k_u=k_u+1;
20:   end if
21: end while
```

This problem is ameliorated by providing the nodes a priori the sequence of local time instants $\tau^{(i)}$, $i = 0, 1 \ldots$ mentioned earlier. This sequence is called an *iteration schedule*, and the formula for computing it is described below. Let the skews and offsets of all clocks be lower and upper bounded by those in two fictitious clocks c_L and c_H, such that $\alpha_{c_L} \leq \alpha_u \leq \alpha_{c_H}$, $\beta_{c_L} \leq \beta_u \leq \beta_{c_H}$. Therefore $\tau_{c_L}(t) \leq \tau_u(t) \leq \tau_{c_H}(t)$ for all $u \in \mathcal{V}$. The formula for calculating $\tau^{(i)}$ is

$$\tau^{(i+1)} = \frac{\alpha_{c_H}}{\alpha_{c_L}}(\tau^{(i)} + \delta t - \beta_{c_L}) + \beta_{c_H}, \tag{6}$$

where $\tau^{(0)}$ has to be chosen such that $\tau^{(0)} > \beta_{c_H}$. This schedule ensures that nodes operating on their unsynchronized local clocks still perform updates in an effectively synchronous manner. To see this, define $\mathcal{I}^{(i)} := (\frac{\tau^{(i)} - \beta_{c_H}}{\alpha_{c_H}}, \frac{\tau^{(i+1)} - \beta_{c_H}}{\alpha_{c_H}})$ as a global interval and $\mathcal{I}_u^{(i)} := (\frac{\tau^{(i)} - \beta_u}{\alpha_u}, \frac{\tau^{(i)} + \delta t - \beta_u}{\alpha_u})$ as the global time interval with respect to i-th local iteration of node u. Eq. (6) guarantees that, at each i, $\mathcal{I}_u^{(i)} \subset \mathcal{I}^{(i)}$ for all $u \in \mathcal{V}$. In other words, there exists a sequence of global time intervals such that the i-th global interval contains, and only contains, the i-th local iteration (in global time) of all $u \in \mathcal{V}$. We emphasize that $\tau^{(i)}$ is the same for all nodes and every node u starts and ends its i-th iteration at the same local time instants $\tau^{(i)}$ and $\tau^{(i)} + \delta t$. Although we don't know real skews and offsets, we can still pick fairly safe values for $\frac{\alpha_{c_H}}{\alpha_{c_L}}$, β_{c_L} and β_{c_H}, which is described in the technical report [20].

Due to the use of the iteration schedule, the DiSync algorithm can be now analyzed as if it is executed by all nodes synchronously.

IV. Convergence analysis

In this section we consider only the synchronous version of the algorithm using global index k. We rewrite (5) as

$$\hat{x}_u(k+1) = \hat{x}_u(k) + m(k) \sum_{v \in \mathcal{N}_u(k)} a_{uv}(k)(\hat{x}_v(k) + \zeta_{u,v}(k) - \hat{x}_u(k)). \quad (7)$$

Now define the estimation error as $e_u(k) := \hat{x}_u(k) - x_u$. Eq. (7) reduces to the following using (4):

$$e_u(k+1) = e_u(k) + m(k) \sum_{v \in \mathcal{N}_u(k)} a_{uv}(k)(e_v(k) - e_u(k) + \epsilon_{u,v}(k)). \quad (8)$$

In order to pursue further analysis, we introduce some stipulations and notations. First, we let $a_{uv}(k) = 0$ for $v \notin \mathcal{N}_u(k)$. Then, the $n \times n$ Laplacian matrix $L(k)$ of the graph $\mathcal{G}(k)$ is defined as $L_{uv}(k) = \sum_{v=1}^{n} a_{uv}(k)$ if $u = v$, and $L_{uv}(k) = -a_{uv}(k)$ if $u \neq v$. By removing the rows and columns of $L(k)$ with respect to reference nodes, we get the $n_b \times n_b$ principle submatrix $L_b(k)$ (so called grounded or Dirichlet Laplacian matrix [21]). Let $e(k) := [e_1(k), \ldots, e_{n_b}(k)]^T$, the corresponding state space form of the estimation error is

$$e(k+1) = (I - m(k)L_b(k))e(k) + m(k)D(k)\epsilon(k), \quad (9)$$

where

$$\epsilon(k) := [\bar{\epsilon}_1(k)^T, \ldots, \bar{\epsilon}_{n_b}(k)^T]^T,\ \bar{\epsilon}_u(k) := [\epsilon_{u,1}(k), \ldots \epsilon_{u,n}(k)]^T,$$
$$D(k) := diag(\bar{a}_1(k), \ldots, \bar{a}_{n_b}(k)),\ \bar{a}_u(k) := [a_{u,1}(k), \ldots, a_{u,n}(k)],$$

where $\epsilon_{u,u}(k) \notin \bar{\epsilon}_u(k)$ and $a_{u,u}(k) \notin \bar{a}_u(k)$. Note that when $a_{uv}(k) = 0$, $\epsilon_{u,v}(k)$ is a random variable with the same mean and variance as the measurement noise on any existing edge. Moreover, recall that once a node u computes measurement $\zeta_{u,v}(k)$, it sends this measurement to v. Thus $\epsilon_{u,v}(k) = -\epsilon_{v,u}(k)$.

Assumption 1: Measurement noise vector $\epsilon(k)$ is with mean $E[\epsilon(k)] = \gamma$ and bounded second moment, i.e. $E[\|\epsilon(k)\|^2] < \infty$, where $\|\cdot\|$ denotes 2-norm. Furthermore, $\epsilon(k)$ and $\epsilon(j)$ are independent for $k \neq j$. In addition, $\{\epsilon(k)\}$ is independent of $e(0)$, where $E\|e(0)\|^2 < \infty$.

Assumption 2: The non-increasing positive sequence $\{m(k)\}$ (step size of the stochastic approximation) is chosen as $m(k) = \frac{c_1}{k+c_2}$, where c_1, c_2 are constant real numbers. Therefore, $m(k)$ satisfies $\sum_{k=0}^{\infty} m(k) = \infty$ and $\sum_{k=0}^{\infty} m^2(k) < \infty$.

Assumption 3: There exists $d \in \mathbb{N}$ s.t. for any $t \geq 0$,$\hat{\mathcal{G}}_t^d := \bigcup_{k=t}^{t+d-1} \mathcal{G}(k) = (\mathcal{V}, \bigcup_{k=t}^{t+d-1} \mathcal{E}(k))$ is connected, where $\mathcal{E}(k)$ is set of edges in $\mathcal{G}(k)$.

Assumption 4: The limits $\bar{L}, \bar{L}_b, \bar{D}$ defined below exist: $\bar{L} := \lim_{t\to\infty} \frac{1}{t} \sum_{k=0}^{t} L(k)$, $\bar{L}_b := \lim_{t\to\infty} \frac{1}{t} \sum_{k=0}^{t} L_b(k)$, $\bar{D} := \lim_{t\to\infty} \frac{1}{t} \sum_{k=0}^{t} D(k)$.

Remark 1:

1) Assumption 3 implies that information can go from any node to the rest of the nodes within a fixed bounded length of time. In other words, nodes are connected for an infinite number of times. Furthermore, as $\mathcal{G}(k)$ is bidirectional, another equivalent assumption is that $\hat{G}_t^d$ contains a spanning tree. The proposed algorithm is also robust to permanently adding or deleting nodes in case the new resulting graph satisfies the assumption.
2) To understand the meaning of Assumption 4, define the finite state space $\mathbb{G} = \{\mathcal{G}_1, \ldots, \mathcal{G}_N\}$ as the set of graphs that can occur over time. If the sequence of $\mathcal{G}(k)$ can be divided into a sequence of finite intervals I_j, $j = 1, 2, \ldots$, such that the percentage of times that each state $\mathcal{G}_k$ occurs is fixed in all except finitely many such intervals I_j, then $\bar{L}$, $\bar{L}_b$ and $\bar{D}$ exist. Another example is that the state $\mathcal{G}_i$ occurs according to a sample path of a stationary ergodic process. In the end, denote sets of matrices $\mathbb{L}_b = \{L_{b1}, \ldots, L_{bN}\}$ and $\mathbb{D} = \{D_1, \ldots, D_N\}$, where L_{bi} and D_i correspond to $\mathcal{G}_i \in \mathbb{G}$. If the percentage of all states occurring is $\pi = \{\pi_1, \pi_2, \ldots, \pi_N\}$, then

$$\bar{L}_b := \sum_{i=1}^{N} \pi_i L_{bi},\ \bar{D} := \sum_{i=1}^{N} \pi_i D_i \quad (10)$$

Theorem 1: Under Assumption 1-4, the Algorithm 1 ensures that $e(k)$ in (9) converges to $\bar{L}_b^{-1}\bar{D}\gamma$ in mean square, i.e., $\lim_{k\to\infty} E(\|e(k) - \bar{L}_b^{-1}\bar{D}\gamma\|^2) = 0$. □

The theorem states that under the assumptions, the variance of the estimation error decays to 0. If additionally all the difference measurements are unbiased ($\gamma = 0$), then the bias of the estimates converge to 0 as well. Due to limited space, we include the proof in the technical report [20].

V. Numerical evaluation of time synchronization accuracy with DiSync

We now examine the performance of the DiSync algorithm of performing clock synchronization. To simulate a realistic scenario, we use a pairwise synchronization algorithm to generate difference measurements. Random delays during the exchange of time stamped message directly determine the level of noise in the resulting difference measurements for the pairwise synchronization phase, which ultimately determines the clock synchronization accuracy. Since we directly employ a pairwise synchronization protocol, we do not know if the noise in the difference measurements of log-skew and offsets are unbiased or not, and what the noise variance is. Therefore, the communication delays are chosen to be realistic based on available information so that the noise levels are close to what are expected to occur in practice. The setting of the simulations is such that the assumptions of Theorem 1 are either unlikely to be satisfied or not possible to check. Simulations are performed in a 50-node mobile network within a $100m \times 100m$ square field. Nodes' motions are generated according to the widely used Random Direction (RD) mobility model [22]. A pair of nodes can communicate when distance between them is less than $15m$. When within communication range, they exchange time stamped messages to perform pairwise synchronization

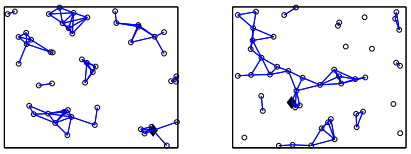


Fig. 1. Two graphs that occur during one simulation with 50 nodes moving according to the random direction mobility model.

to obtain difference measurements of log skews and offsets. Due to random motion, the connectivity assumption of Theorem 1 is violated.

The true skews and offsets of 49 nodes are picked uniformly from $[1 - 2 \times 10^{-5}, 1 + 2 \times 10^{-5}]$ and $[-10^{-2}, 10^{-2}]sec$ respectively according to [9]. The single reference node (50th) has skew 1 and offset 0. The update interval (also called synchronization period) is chosen as 1 sec, i.e., $t_{k+1} - t_k = 1$. For the sake of convenience, simulations are carried out in a synchronous fashion.

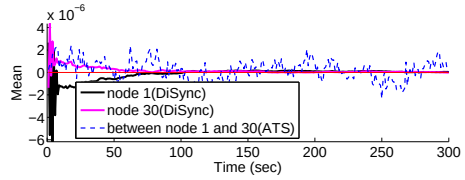
A. More details on pairwise synchronization

In this evaluation, we select the pairwise synchronization algorithm proposed in [4] to compute the relative skew $\alpha_{u,v}$ and relative offset $\beta_{u,v}$. The difference measurements $\beta_u - \beta_v$ and $\log(\alpha_u) - \log(\alpha_v)$ are then obtained from these as described in Section II. According to [4], at the beginning of the kth interval, node u sends a message to v that contains the value of the local time at u when the message is sent: $\tau_u^{(1)}$. When node v receives this message, it records the local time of reception: $\tau_v^{(1)}$. After a waiting period, node v sends a message back to u that contains both $\tau_v^{(2)}$ and $\tau_v^{(1)}$. When it arrives at u, node u again records the local time of reception: $\tau_u^{(2)}$. Two nodes u and v in communication range performs this procedure, called two-way time-stamped message exchange, twice - at the beginning and in the middle of each synchronization period. At the end of the k-th synchronization period, node u uses the obtained eight time stamps $\{\tau_u^{(i)}, \tau_v^{(i)}\}$ for $i = 1, \ldots 4$ to estimate $\alpha_{u,v}(k)$ and $\beta_{u,v}(k)$ via the formula provided in [4]. Finally, node u sends back to v these estimates.

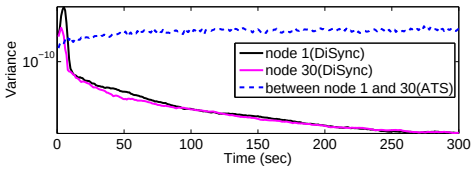
The random errors in the estimated $\alpha_{u,v}(k)$ and $\beta_{u,v}(k)$ come from the random delay during the message exchange. We assume the random delay is Gaussian distributed with mean $150\mu sec$ and standard deviation $10\mu sec$ [9]. The relation between the statistics of the delays and that of the measurement noise defined in Section II are complex. In simulations, we simply use the pairwise synchronization algorithm and the resulting relative skew and offset estimates, whatever the noise is, to estimate absolute skews and offsets.

B. DiSync performance in estimating global time

We conduct 500 Monte Carlo simulations. Figure 1 shows two snapshots of the network during a simulation. As we can see, only a limited number of nodes can communicate with each other. In the following plots, the x-axis is discrete in time, i.e., t_k for $k = 1, 2, \ldots$. Recall that $a_{uv}(k) = 1$ if $(u, v) \in \mathcal{E}(k)$ for all k. The step size is chosen as $m(k) = \frac{1.5}{k+3}$. Figure 2(a) and 2(b) show the mean and variance of estimation error of skews for two nodes. The variance is seen to converge to 0. Figure 3 shows the global time estimation error, $\hat{t}_u - t$ as a function of t for two nodes. After an initial transient period, the estimation error of the global time is quite small. The extremely accurate skew estimates obtained by DiSync is crucial in getting good global time estimates, since even a tiny error in the skew estimate leads to a large error in the prediction of global time t over time.



(a) Mean



(b) Variance

Fig. 2. Empirically estimated mean and variance of the estimation error of skews: for DiSync algorithm, the error is defined as $\hat{\alpha}_u(t) - \alpha_u$; for ATS, the error is defined as $\alpha_u \hat{\varrho}_u(t) - \alpha_v \hat{\varrho}_v(t)$, i.e., the difference of estimated skew of virtual clock between node u and v. Note that in (b), y-axis is in logarithm scale.

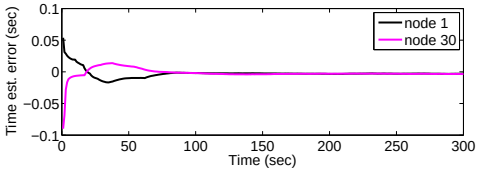


Fig. 3. The estimation error of global time of two nodes in DiSync.

C. Comparison between DiSync and ATS

In ATS [1], each node u estimates the virtual global time using $\hat{t}_u^r(t) = \hat{\varrho}_u(k)\tau_u(t) + \hat{o}_u(k)$ for $t_k \le t \le t_{k+1}$, where variables $\hat{\varrho}_u(k)$ and $\hat{o}_u(k)$ can be thought of as the skew and offset of a virtual global time respect to the local time of u during interval k. Each node u updates its $\hat{\varrho}_u(k)$ and $\hat{o}_u(k)$ using $\hat{\varrho}_v(k)$, $\hat{t}_v^r(k)$, $\hat{\alpha}_{uv}(k)$ from its neighbors, where $\hat{\alpha}_{uv}(k)$ is the estimated relative skew. $\hat{\alpha}_{uv}(k)$ is obtained by pairwise communication between u and v during k-th update interval, as part of the ATS algorithm. This is done as follows. Two time-stamped messages are sent from node v to node u: one at the beginning of k-th interval and the other one in the middle of the interval. Note that no return messages from u to v is required. The computation of $\hat{\alpha}_{uv}(k)$ is performed by a low-pass filter as provided in ATS: $\hat{\alpha}_{u,v}(k) = \rho\hat{\alpha}_{u,v}(k-1) + (1-\rho)\frac{\tau_v^{(1)} - \tau_v^{(2)}}{\tau_u^{(1)} - \tau_u^{(2)}}$, where ρ is a tuning parameter and chosen as 0.2 (same value used in ATS). It has been shown that $\lim_{k\to\infty} \alpha_u \hat{\varrho}_u(k) = \bar{\alpha}$, $\lim_{k\to\infty} \hat{o}_u(k) + \beta_u \hat{\varrho}(k) = \bar{\beta}$, where $\bar{\alpha}$ and $\bar{\beta}$ is the skew and offset of the virtual clock with respect to t. The ATS algorithm ensures that the estimated

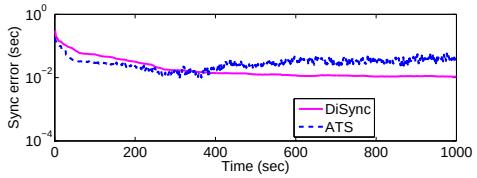


Fig. 4. Maximum synchronization error along time in one experiment.

virtual global times in all nodes are eventually equal, i.e., $\lim_{t\to\infty} \hat{t}_u^r(t) = \hat{t}_v^r(t)$ for all u and v, under the assumption that the time stamps are exchanged without random delay.

To compare with the proposed DiSync algorithm under identical conditions, we add random delay to $\tau_u^{(i)}$ for $i = 1, 2$. The delay parameters are the same as those used during the simulation of the DiSync algorithm. In addition, since ATS does not estimate the clock time at any of the nodes, we use the metric "maximum synchronization error" to compare ATS with DiSync. They are defined as $\max_{u,v} |\hat{t}_u(t_k) - \hat{t}_v(t_k)|$ and $\max_{u,v} |\hat{t}_u^r(t_k) - \hat{t}_v^r(t_k)|$ for all u and v, in DiSync and ATS respectively.

Figure 4 compares maximum synchronization error for both the algorithms. The maximum synchronization error decreases faster in ATS at the beginning. However, the superior robustness to the measurement noise of the DiSync algorithm helps it outperform ATS after about 300 sec. This higher robustness is also seen in Figure 2(a) and 2(b). While the variance of the estimation errors of skews and offsets converge to zero for DiSync, they converge to non-zero constants in case of ATS.

VI. Conclusion

We proposed DiSync, a distributed asynchronous protocol for clock synchronization in mobile ad-hoc networks. Clock synchronization is performed in two stages, first by estimating the skews and offsets (with respect to a global clock) of the nodes and then using them to estimate the global time. The estimation error of skews and offsets is proved to be mean square convergent. Numerical evaluation demonstrates that nodes can estimate the time of global clock accurately. Simulations also showed that DiSync outperforms ATS in terms of maximum synchronization error.

The iteration schedule proposed here to impose implicitly synchronized updates requires some knowledge of the bounds on skews and offsets of all the nodes. With poor bounds, especially on the skews, the time interval required to perform one update step grows without bound as time increases. We believe this potential pitfall can be fixed by using the estimated skews after some time has passed (instead of pre-specified bounds on skews). Since the DiSync algorithm provides highly accurate skew estimates, doing so will ensure synchronous updates while keeping the time interval between updates from growing. The effectiveness of this strategy will be studied in future work.

References

[1] L. Schenato and F. Fiorentin, "Average timesynch: A consensus-based protocol for clock synchronization in wireless sensor network," *Automatica*, vol. 47, no. 9, pp. 1878 – 1886, 2011.

[2] B. M. Sadler and A. Swami, "synchronization in sensor networks: an overview," in *IEEE MILCOM*, October 2006, pp. 1–6.

[3] J. R. Vig, "Introduction to quartz frequency standards," Army Research Laboratory, Tech. Rep., 1992.

[4] K.-L. Noh, Q. M. Chaudhari, E. Serpedin, and B. W. Suter, "Novel clock phase offset and skew estimation using two-way timing message exchanges for wireless sensor networks," *IEEE Transactions on Communications*, vol. 55, no. 4, pp. 766–777, Apr 2007.

[5] S. Yoon, C. Veerarittiphan, and M. L. Sichitiu, "Tiny-sync: Tight time synchronization for wireless sensor networks," *ACM Transactions on Sensor Networks*, vol. 3, no. 2, pp. 1–34, Jun 2007.

[6] M. Leng and Y.-C. Wu, "On clock synchronization algorithms for wireless sensor networks under unknown delay," *IEEE Transactions on Vehicular Technology*, vol. 59, no. 1, pp. 182–190, Jan 2010.

[7] J. Elson, L. Girod, and D. Estrin, "Fine-grained network time synchronization using reference broadcasts," in *the Fifth Symposium on Operating Systems Design and Implementation (OSDI)*, 2002.

[8] S. Ganeriwal, R. Kumar, and M. B. Srivastava, "Timing-sync protocol for sensor networks," in *ACM Conference on Embedded Networked Sensor Systems (SenSys)*, 2003.

[9] M. Maróti, B. Kusy, G. Simon, and Á. Lédeczi, "The flooding time synchronization protocol," in *ACM Conference on Embedded Networked Sensor Systems (SenSys)*, 2004.

[10] P. Barooah, N. M. da Silva, and J. P. Hespanha, "Distributed optimal estimation from relative measurements for localization and time synchronization," in *International Conference on Distributed Computing in Sensor Systems DCOSS'06*, San Francisco, June 2006.

[11] A. Giridhar and P. R. Kumar, "Distributed clock synchronization in wireless networks: Algorithms and analysis (I)," in *45th IEEE Conference on Decison and Control*, December 2006, pp. 4915 – 4920.

[12] 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 Decison and Control*, December 2006, pp. 2734–2739.

[13] N. Freris, V. Borkar, and P. Kumar, "A model-based approach to clock synchronization," in *Decision and Control, 2009 held jointly with the 2009 28th Chinese Control Conference. CDC/CCC 2009. Proceedings of the 48th IEEE Conference on*, dec. 2009, pp. 5744 –5749.

[14] N. Freris, S. Graham, and P. Kumar, "Fundamental limits on synchronizing clocks over networks," *Automatic Control, IEEE Transactions on*, vol. 56, no. 6, pp. 1352 –1364, june 2011.

[15] M. Leng and Y.-C. Wu, "Distributed clock synchronization for wireless sensor networks using belief propagation," *Signal Processing, IEEE Transactions on*, vol. 59, no. 11, pp. 5404 –5414, nov. 2011.

[16] C. Liao and P. Barooah, "Time synchronization in mobile sensor networks from difference measurements," in *In proceedings of the 49th IEEE Conference on Decision and Control*, December 2010, pp. 2118 – 2123.

[17] M. Huang, S. Dey, G. N. Nair, and J. H. Manton, "Stochastic consensus over noisy networks with Markovian and arbitrary switches," *Automatica*, vol. 46, no. 10, pp. 1571–1583, Oct. 2010.

[18] T. Li and J. Zhang, "Consensus conditions of multi-agent systems with time-varying topologies and stochastic communication noises," *Automatic Control, IEEE Transactions on*, vol. 55, no. 9, pp. 2043–2057, 2010.

[19] R. Carli, E. D'Elia, and S. Zampieri, "A pi controller based on asymmetric gossip communications for clocks synchronization in wireless sensors networks," in *Decision and Control and European Control Conference (CDC-ECC), 2011 50th IEEE Conference on*, dec. 2011, pp. 7512 –7517.

[20] C. Liao and P. Barooah, "Disync: Accurate distributed clock synchronization in mobile ad-hoc networks from noisy relative measurements [technical report]," Mechanical and Aerospace Engineering, University of Florida, Tech. Rep., September 2012. [Online]. Available: http://plaza.ufl.edu/cdliao/

[21] P. Barooah and J. P. Hespanha, "Graph effective resistances and distributed control: Spectral properties and applications," in *Proc. of the 45th IEEE Conference on Decision and Control*, December 2006, pp. 3479–3485.

[22] T. Camp, J. Boleng, and V. Davies, "A survey of mobility models for ad hoc network research," *Wireless Communications and Mobile Computing*, vol. 2, no. 5, pp. 483–502, Aug 2002.