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

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 durations 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:

$$\tau_u(t) = \alpha_u t + \beta_u,$$

where the scalar parameters $\alpha_u, \beta_u$ 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 $\alpha_u$ and $\beta_u$ 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 and tree-based protocols, synchronization relies on establishing a pre-specified network structure. Reference Broadcast Synchronization (RBS) [7] is one of the protocols performing cluster-based synchronization. Each cluster contains an elected reference node, which broadcasts beacons and other nodes in this cluster compare the arrival times of a beacon to figure out clock offset between each other. This is used to correct the local clocks. Nodes in different clusters use gateway nodes to get synchronized as well. The tree-based protocols is to first elect a root node and construct a spanning tree of the network with the root node being the “level 0” node. Then, every node thereafter synchronizes itself to a node of lower level (higher up in the hierarchy) by correcting its local clock. Examples of such spanning-tree based protocols include the widely used Timing-Sync Protocol for Sensor Networks (TPSN) [8] and Flooding Time Synchronization Protocol (FTSP) [9]. 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. The estimates of a variable (skew or offset) are obtained from noisy measurements of the difference between the variable’s values at two neighboring nodes. These measurements can be obtained using time-stamped messages. The time-varying topology of a network of mobile nodes is modeled in [16] as the states of a Markov chain. Under certain conditions, it was shown that the variance of estimation error converges to a positive value. However, even a small error in the skew estimates is likely to cause large error in the estimate of global time 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 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 $\alpha_{u,v} := \frac{s_{u,v}}{o_{u,v}}$ and relative offset $\beta_{u,v} := \beta_u - \beta_v - 2\alpha_{u,v}$. The reason for this terminology is the following relationship $\tau_u(t) = \frac{\beta_u}{\alpha_u} \tau_v(t) + \beta_u - \beta_v - 2\alpha_{u,v}$, 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 $\hat{\alpha}_{u,v} = \alpha_{u,v} + e_{u,v}^s$, where $e_{u,v}^s$ is the estimation error. Therefore, by $\alpha_{u,v} = \frac{e_{u,v}^s}{e_{u,v}^o}$,

$$\log \hat{\alpha}_{u,v} = \log \alpha_u - \log \alpha_v + \xi_{u,v}^s,$$

where $\xi_{u,v}^s = \log(1 + e_{u,v}^o \frac{\alpha_v}{\alpha_u})$. 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 $e_{u,v}^o$ is small, then the measurement noise $\xi_{u,v}^s$ is small. Similarly, the noisy estimate of relative offset is modeled as $\hat{\beta}_{u,v} = \beta_u + e_{u,v}^o$, where $e_{u,v}^o$ is the error. Again, by $\beta_{u,v} = \beta_u - \beta_v - \xi_{u,v}^s$,

$$\hat{\beta}_{u,v} = \beta_u - \beta_v + \xi_{u,v}^s,$$

which is a noisy difference measurement of the offsets between the two nodes, with measurement noise $\xi_{u,v}^s = \beta_u(1 - \frac{1}{\alpha_u}) + e_{u,v}^o$. Due to the term $\beta_v(1 - \frac{1}{\alpha_u})$, the measurement error is biased even if $e_{u,v}^o$ is zero mean. Since $\frac{\alpha_v}{\alpha_u}$ 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 $\beta_u$ 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 $G(k) = (\mathcal{V}', \mathcal{E}(k))$, where $(u, v) \in \mathcal{E}(k)$ if and only if $u$ and $v$ can obtain a difference measurement of the form

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

(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 \( \zeta_{v,u}(k) := -\zeta_{u,v}(k) \).

The neighbors of \( u \) at \( k \), denoted by \( N_u(k) \), is the set of nodes that \( u \) has an edge with in the measurement graph \( G(k) \). We assume that if \( v \in N_u(k) \), 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 \( n_p \geq 1 \) (i.e., there exists at 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 \( \hat{x}_u(k_u) \) along with the measurements \( \zeta_{u,v}(k_u) \) from its current neighbors \( v \in N_u(k_u) \). After a fixed time length \( \delta t \) (measured in local time), node \( u \) updates its new estimate based on current measurements and neighbors’ estimates by using the following update law:

\[
x_u(k_u+1) = \hat{x}_u(k_u) + \sum_{v \in \mathcal{N}_u(k_u)} a_{uv}(k_u) (\hat{x}_v(k_u) + \zeta_{u,v}(k_u) - \hat{x}_u(k_u));
\]

(5)

where the time varying gain \( m(\cdot) : \mathbb{Z}^+ \rightarrow \mathbb{R}^+ \) 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 \( \delta t \) (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.

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 skew 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 \( \tau_{c_L}(t) \leq \tau_{c_H}(t) \) for all \( u \in \mathcal{V} \). The formula for calculating \( \tau^{(i)} \) is

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

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 \( \tau_u^{(i)} := \tau^{(i)} - \beta_u = \tau^{(i)} - \beta_{c_L} + \delta t - (\alpha_{c_H} - \alpha_u) \) as a global index and \( \tau_u^{(i)} := \tau_u^{(i)} + \delta t - \beta\cdot \) as the global time interval with respect to \( i \)-th local iteration of node \( u \). Eq. (6) guarantees that,
at each $i$, $I_u^{(i)} \subset I^{(i)}$ for all $u \in \mathcal{U}$. 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 $u \in \mathcal{U}$. Figure 1(a) shows the relationship between intervals of local iterations and the corresponding global intervals. In Figure 1(b), we pick the 3rd global interval from Figure 1(a), and show the global time intervals when local iteration updates occur. 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 $\alpha_{cl}, \beta_{cl}$ and $\beta_{clH}$ as follows. In wireless sensor nodes, a pair of clocks in sensor nodes usually drift apart up to 40 $\mu$sec/sec [3]. Therefore, we can pick $\alpha_{clH}/\alpha_{cl} \approx 1 + 4 \times 10^{-5}$. Such a choice ensures that the time interval between two successive iterations, $\tau^{(i+1)} - \tau^{(i)}$, will only increase from 1 second to 60 second after about $10^5$ intervals, which takes more than 400 hours. To help with picking reasonable values of the offset bounds, the following procedure should be used to initialize the procedure of the synchronization. The reference node first broadcasts a message (to indicate the beginning of synchronization) and sets its local clock time $t$ to zero simultaneously. A node that receives this message sets its own clock to zero and broadcast such message again. The nodes that hear this message also set their local clocks to $t$ and so forth. As nodes (except the reference node) start their local clocks after – but close to – the instant of $t = 0$ their offsets are negative and small. Therefore, $\beta_{clH}$ can be chosen as zero and $\beta_{cl}$ can be picked as an estimate of the time it takes for all active nodes to receive the “synchronization start” signal. For a node who was out of communication range at the beginning but joins the networks later, it can set the local time to the current local time of a neighbor, who has already started the time synchronization, and record neighbor’s iteration index as well. In this way, the newly joined node can take part in the synchronization process as if it started at the very beginning.

IV. CONVERGENCE ANALYSIS

As described in the previous section, due to the use of the iteration schedule, there exists a common iteration counter $k$ that can be used to describe the local updates in the nodes even though none of the nodes has access to it. In this section we consider only the synchronous version of the algorithm using global index $k$. We rewrite (5) as

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

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

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

In order to pursue further analysis, we introduce some stipulations and notations. First, we let $a_{uv}(k) = 0$ for $u \neq \mathcal{N}_u(k)$. Then, the $n \times n$ Laplacian matrix $L(k)$ of the graph $G(k)$ is defined as $L(k) := \sum_{u=1}^{n} a_{uv}(k)$ 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 [20]). Let $e(k) := [e_1(k), \ldots, e_{n_b}(k)]^T$, the corresponding state space form of the estimation error is

$$e(k + 1) = (I - m(k)L_b(k))e(k) + m(k)D(k)e(k),$$

where

$$e(k) := [e_1(k), \ldots, e_{n_b}(k)]^T, \quad e_u(k) := [e_{u,1}(k), \ldots, e_{u,n_b}(k)]^T, \quad D(k) := \text{diag}(\bar{a}_1(k), \ldots, \bar{a}_{n_b}(k)), \quad \bar{a}_u(k) := [a_{u,1}(k), \ldots, a_{n_b}(k)],$$

and $e_u(k) \notin e_u(k)$ and $a_{u,n_b}(k) \notin \bar{a}_u(k)$. Note that when $a_{uv}(k) = 0$, $e_{u,v}(k)$ 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 $\hat{z}_{u,v}(k)$, it sends this measurement to $v$. Thus $e_{u,v}(k) = -e_{v,u}(k)$.

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

![Fig. 1](image-url)
Assumption 3: There exists \(d \in \mathbb{N}\) s.t. for any \(t \geq 0, \hat{G}_t^k := \bigcup_{t=k}^{t+d-1} G(k)\) is connected, where \(G(k)\) is set of edges in \(\hat{G}(k)\).

Assumption 4: The limits \(\bar{L}, \bar{L}_b, \bar{D}\) defined below exist:
\[
\bar{L} := \lim_{t \to \infty} \frac{1}{2} \sum_{k=0}^{t} L(k), \quad \bar{L}_b := \lim_{t \to \infty} \frac{1}{2} \sum_{k=0}^{t} L_b(k), \quad \bar{D} := \lim_{t \to \infty} \frac{1}{2} \sum_{k=0}^{t} D(k).
\]

Remark 1: 1) In practice, \(E[|e(k)|]\) may be time-varying, e.g. the bias in offset difference measurement computed from node \(u\) is different from that computed from node \(v\), as \(\beta_u(1 - \frac{a_u}{a_v}) \neq \beta_v(1 - \frac{a_u}{a_v})\). Therefore, \(E[|e(k)|]\) depends on which node initializes pairwise synchronization at time \(t\). To meet this requirement \(E[|e(k)|] = \gamma\) in Assumption 1, we can stipulate that the node who computes \(\zeta_{u,v}(k)\) between a pair \(u\) and \(v\) is fixed for all time \(t\). 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 \(v\). Indeed, the purpose of this requirement is to provide formula to compute the steady state value of estimation error, and the system may still achieve convergence without it.

2) 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 \(G(k)\) is bidirectional, another equivalent assumption is that \(\hat{G}_t^k\) 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.

3) To understand the meaning of Assumption 4, define the finite state space \(G = \{G_1, \ldots, G_N\}\) as the set of graphs that can occur over time. If the sequence of \(G(k)\) can be divided into a sequence of finite intervals \(I_j, j = 1, 2, \ldots\), such that the percentage of times that each state \(G_j\) 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 \(G_i\) 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 \(L_{bj}\) and \(D_i\) correspond to \(G_i \in G\). If the percentage of all states occurring is \(\pi = \{\pi_1, \pi_2, \ldots, \pi_N\}\), then
\[
\bar{L}_b := \sum_{i=1}^{N} \pi_i L_{bi}, \quad \bar{D} := \sum_{i=1}^{N} \pi_i D_i \quad (10)
\]

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

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. Proof of the theorem uses the following lemma.

**Lemma 1:** If difference measurement is unbiased, i.e., \(\gamma = 0\), under assumption 1-3, the Algorithm 1 ensures that \(e(k)\) in (9) converges to 0 in mean square, i.e., \(\lim_{k \to \infty} E[|e(k)|^2] = 0\) □

When \(\gamma = 0\), (9) can be regarded as a leader-following consensus problem with time-varying topology and zero-mean noisy input. The leaders are reference nodes \(u \in \mathcal{V}_r\), which hold their variable as zero. Then, \(e_u(k)\) for \(u \in \mathcal{V}_b\) is driven to zero by the reference nodes as \(k\) goes to \(\infty\) in mean square sense. The proof of the Lemma 1 is inspired from average-consensus algorithms as that in [17], [18]. The proof is given in Appendix I.

**Proof of Theorem 1.** Taking expectations on both sides of (9), we obtain
\[
\eta(k+1) = (I - m(k)L_b(k))\eta(k) + m(k)D(k)\gamma, \quad (11)
\]
where \(\eta(k) = E[e(k)]\). It follows from (9),
\[
\bar{e}(k+1) = (I - m(k)L_b(k))\bar{e}(k) + m(k)D(k)\xi(k), \quad (12)
\]
where \(\bar{e}(k) = e(k) - \eta(k)\) and \(\xi(k) = e(k) - \gamma\). Note that \(\xi(k)\) is zero mean and satisfies Assumption 1. By Lemma 1, \(\bar{e}(k)\) converges to 0 in mean square. Therefore, \(e(k)\) is mean square convergent to \(\eta(k)\). Now, we examine the convergence of \(\eta(k)\). Note that \(\bar{L}\) is a Laplacian matrix of a graph, therefore, \(\bar{L}_b\) is grounded Laplacian, which is the principal submatrix of \(\bar{L}\). In addition, by Assumption 3, \(\bigcup_{i=1}^{N} G_i \in G\) is connected. Thus, \(\bar{L}\) is a Laplacian matrix of a connected graph and all eigenvalues of \(L_b\) are positive by the Lemma 1 in [10]. Furthermore, since each \(L_b(k)\) is symmetric (as \(\alpha_{uv}(k) = \alpha_{v, u}(k)\) is guaranteed by the algorithm) and bounded, \(\bar{L}_b\) is symmetric and bounded. By Lemma 2 in Appendix II, \(\lim_{k \to \infty} \eta(k) = \bar{L}_b^{-1}\bar{D}\gamma\). Consequently, \(e(k)\) converges to \(\bar{L}_b^{-1}\bar{D}\gamma\) in mean square.

**A. Verification of Theorem 1**

We perform simulations on a made-up scenario that allows numerical verification of the predictions of Theorem 1. A network of 5 nodes is chosen. The topology \(G(k)\) switches periodically among a node set \(G = \{G_1, G_2, G_3\}\) shown in Figure 2 according to the rule: \(G(k) = G_1\) when \(k = 5(T - 1) + 1\) or \(5(T - 1) + 5\); \(G(k) = G_2\) when \(k = 5(T - 1) + 2\) or \(5(T - 1) + 3\); \(G(k) = G_3\) when \(k = 5(T - 1) + 4\), where \(T = 1, 2, \ldots\). Therefore, the percentage of time each graph occurs is \(\pi = [2/5, 2/5, 1/5]\). Note that the union of the graphs in \(G\) is connected, though none of the graphs is a connected graph. Node variables are picked randomly around 0. Node 5 is the single reference node, with the node variable 0. The variance of measurement noise is 1 and the bias in the measurement \(\gamma_{u,v}\) for \(1 \leq u < v \leq 5\) is assigned from set \(\{5, 3, -3, -2, 2, 1, 1, -1, 6, 7\}\) in order, e.g., \(\gamma_{1,2} = 5\) and \(\gamma_{3,5} = 6\). Theorem 1 predicts that the bias of estimation error is \(\bar{L}_b^{-1}\bar{D}\gamma = \{-3.49, -4.51, -3.95, -10.73\}\). The step size is chosen as \(m(k) = \frac{1}{k^{0.5}}\). Mean and variance of estimation error is computed from 1000 Monte Carlo
Fig. 2. All the graphs that occur in simulation #1.

Fig. 3. Empirically mean and variance of the estimation error for two nodes in the 5-node mobile network, which approach to −4.51 and 10.73 respectively as seen from the figure.

Fig. 4. Two graphs that occur during one simulation with 50 nodes moving according to the random direction mobility model.

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 $k$th 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 $\tau_u^{(1)}$. 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 $\{\tau_i^{(1)}, \tau_i^{(2)}\}$ 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 4 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 $\alpha_{uv}(k) = 1$ if $(u,v) \in E(k)$ for all $k$. The step size is chosen as $m(k) = ...
chosen as 0.2 (same value used in ATS). It has been shown that \( \lim_{k \to \infty} \alpha_u \beta_u(k) = \bar{\alpha}, \lim_{k \to \infty} \beta_u(k) + \beta_u \hat{\beta}(k) = \bar{\beta} \), 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 virtual global times in all nodes are eventually equal, i.e., \( \lim_{t \to \infty} \hat{t}_v(t) = \hat{t}_v(t) \) 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 \( \max_{u,v} |\hat{t}_u(t_k) - \hat{t}_v(t_k)| \) for all \( u \) and \( v \), in DiSync and ATS respectively.

Figure 7 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 5(a) and 5(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: estimating the skews and offsets (with respect to a global clock) of the nodes, and then using them to estimate the global time. The nodes measure log-skew differences and offset differences with nearby neighbors and fuse them with current estimates to iteratively update their estimates of skew and offset. DiSync is inspired by consensus algorithm, in particular the recent work on using stochastic approximation in consensus to ameliorate the effect of noise. The latter ensures that the variance of skew/offset estimation error asymptotically converges to zero under certain assumptions. Using estimated skews and offsets, nodes can estimate the time of global clock accurately which was demonstrated in numerical evaluation. The simulations also indicated that the convergence properties of the algorithm are robust to many of the assumptions made in the analysis. Simulations also showed that DiSync outperforms ATS.
Appendix I
Proof of Lemma 1

The following terminology will be needed for the subsequence analysis. Let \((\Sigma, \mathcal{F}, P)\) be a probability space, and \(\{\mathcal{F}(k)\}\) is a sequence of sub-\(\sigma\)-algebras of \(\mathcal{F}\) such that \(\mathcal{F}(k) \subset \mathcal{F}(k+1)\), for all \(k\). Random sequence \(\{\xi(k)\}\) is called martingale differences \(w.r.t.\) \(\{\mathcal{F}(k)\}\) if \(\mathcal{F}(k)\) \(\mathcal{F}(k)\)-measurable, \(E[\xi(k)|\mathcal{F}(k)] = 0\) and \(E[\xi(k)\xi(j)^T] = 0\), if \(k \neq j\). Therefore, by Assumption 1 along with the mean \(\gamma = 0\), the sequence of vectors \(\{e(k)\}\) is martingale differences \(w.r.t.\) \(\mathcal{F}(k)\), where \(\mathcal{F}(k)\) is a \(\sigma\)-algebra generated by \(\{e(0), e(1), \ldots, e(k)\}\).

Recall that \(e(k)\) is the vector of the estimation errors at index \(k\), and we define \(V(k) = e(k)^T e(k)\). To prove the theorem, we would like to show \(\lim_{k \to \infty} E[V(k)] = 0\), which implies \(E[e_0^2(k)] \to 0\) as \(k \to \infty\) for all \(u \in \mathcal{V}_b\).

Let \(\Phi_{rd}^{-1} = \prod_{j=rd}^{r+1}(I - m(j)L_b(j))\), where \(\prod_{j=1}^n A_t = A_nA_{n-1}\ldots A_1\) as the ordered products of matrices. Now, (9) can be rewritten as

\[
e((r+1)d) = \Phi_{rd}^{-1}e(rd) + \varphi_{rd}^{-1},
\]

where \(\varphi_{rd}^{-1} = \sum_{j=rd}^{r+1}(I - m(j)L_b(j))\) follows that,

\[
V((r+1)d) = e^T((r+1)d)e((r+1)d)
\]

\[
= V(rd) - 2e^T(rd) \sum_{j=rd}^{r+1}(m(j)L_b(j))e(rd) + 2e^T(rd)(\Phi_{rd}^{-1})^T\varphi_{rd}^{-1} + (\varphi_{rd}^{-1})^T\varphi_{rd}^{-1} + (r+1)d - I + 2 \sum_{j=rd}^{r+1}(m(i)L_b(i))e(rd).
\]

Use binomial expansion,

\[
C(2d, 3) \sum_{j_1, j_2, j_3 \in \mathcal{I}_{rd}^{d-1}} (m(j_1)m(j_2)m(j_3)L_b(j_1)L_b(j_3)L_b(j_3)) + H.O.T.
\]

(15)

where \(C(2d, 2) \sum_{j_1, j_2 \in \mathcal{I}_{rd}^{d-1}} (m(j_1)m(j_2)L_b(j_1)L_b(j_2))\) is the summation of the terms of all combinations of \(m(j_1)L_b(j_1)\) and \(m(j_2)L_b(j_2)\), where \(\mathcal{I}_{rd}^{d-1} = \{j \in \mathcal{I}_{rd}^{d-1} | j \leq (r+1)d\} \) and \(C(2d, 2)\) is the total number of terms inside the summation, which is equal to the number of 2-nd combination of \(2d\) elements set. \(C(2d, 3) \sum_{j_1, j_2, j_3 \in \mathcal{I}_{rd}^{d-1}} (\cdot)\) can be interpreted in the same fashion. Therefore,

\[
\|\Phi_{rd}^{-1}e(rd) + \varphi_{rd}^{-1} \| = 2 \sum_{j=rd}^{r+1}(m(i)L_b(i))
\]

\[
\leq m^2(rd)C(2d, 2) \max_{j \in \mathcal{I}_{rd}^{d-1}} \|L_b(j)\|^2 + m^3(rd)C(2d, 3) \max_{j \in \mathcal{I}_{rd}^{d-1}} \|L_b(j)\|^3 + \text{H.O.T.}
\]

(16)

where

\[
Q_d = (2d - 1) \left( \max_{0 \leq j \leq 2d} C(2d, j) \right) \max_{j \geq 0} \sup \|L_b(j)\|^2d, 1).
\]

(17)

For conciseness, let \(L_{bd}^{-1} = \sum_{j=bd}^{r+bd} L_b(j)\) and similarly \(L_{bd}^{-1} = \sum_{j=bd}^{r+bd} L_b(j)\). Since \(L_{bd}^{-1}\) is symmetric, \(\lambda_m(L_{bd}^{-1}) \leq x^T L_{bd}^{(r+bd)} x / (x^T x) \) for \(x \neq 0\) [Lemma 8.4.3 in [23]], where we recall that \(\lambda_m\) means the smallest eigenvalue. So,

\[
V((r+1)d) \leq \left(1 - \frac{2m_m(L_{bd}^{-1}))}{m((r+1)d - 1) + m^2(rd)Q_d} \right) V(rd) + 2e^T(rd)(\Phi_{rd}^{-1})^T\varphi_{rd}^{-1} + (\varphi_{rd}^{-1})^T\varphi_{rd}^{-1}.
\]

(18)

As the grounded Laplacian matrix \(L_{bd}^{-1}\) is the principle submatrices of \(L_{rd}^{-1}\), in which the corresponding undirected graph \(G_{rd}^{-1}\) is connected, by Lemma 1 in [10]. \(L_{bd}^{-1}\) is positive definite. Therefore, \(\lambda_m(L_{bd}^{-1}) > 0\). Since \(e(rd)\) is \(\mathcal{F}(rd - 1)\)-measurable and \(\{\xi(k)\}\) is martingale difference sequence \(w.r.t.\) \(\mathcal{F}(k)\), it follows that

\[
E[2e^T(rd)(\Phi_{rd}^{-1})^T\varphi_{rd}^{-1}|\mathcal{F}(rd - 1)] = 2e^T(rd)(\Phi_{rd}^{-1})^T E[\varphi_{rd}^{-1}|\mathcal{F}(rd - 1)] = 0,
\]

(19)

By the tower property of conditional expectation, \(E[2e^T(rd)(\Phi_{rd}^{-1})^T\varphi_{rd}^{-1}] = 0\). In addition, by property of martingale difference, \(E[e(i)e(j)^T] = 0\) for \(i \neq j\). Thus,

\[
E \left[ e(i)^T \varphi_{rd}^{-1} \right] = \sum_{j=rd}^{r+1} m^2(j)E \left[ e(j)^T D(j)^T \Phi_{j+1}^d(\Phi_{j+1}^d)^T e(j) \right] \leq F_d \sum_{j=rd}^{r+1} m^2(j),
\]

(20)
where \( \kappa(j) = (r + 1)d - j - 2 \) and \( F_d = \sup_{j \geq 0} \| D(j) \|^2 \sup_{j \geq 0} \| \Phi(j) \|^2 \sup_{j \geq 0} \| \epsilon(j) \|^2 \).

Note that \( \lim \sup_{k \to \infty} m(k)/m(k + 1) < \infty \), so there exists \( r > r_0 \) such that \( m(rd) \leq s_d m((r + 1)d) \), where scalar \( s_d \) is a function of \( d \), and \( m(rd) \leq 1 \). Consequently, \( m^2(rd) Q_d \leq m^2((r + 1)d) Q_d s_d \) for \( r > r_0 \). Let \( \lambda_{inf} = \inf_{t \geq 0}(\lambda_m(L_{bt}^{d-1})) \), then,

\[
E[V((r + 1)d)] \leq (1 - 2Q_{inf} d m((r + 1)d) + m^2((r + 1)d) Q_d s_d) E[V(rd)] + F_d \sum_{j = rd} m^2(j). \tag{21}
\]

Let \( q(r) = 2Q_{inf} d m((r + 1)d) - m^2((r + 1)d) Q_d s_d \). As \( \lambda_{inf} > 0 \), there exists \( r > r_1 \) such that \( \lambda_{inf} m((r + 1)d) \geq m^2((r + 1)d) Q_d s_d \). Hence there exists \( r > \max\{r_0, r_1\} \) such that

\[
\sum_{r = 0}^{\infty} q(r) \geq \sum_{r = 0}^{\lambda_{inf} d m((r + 1)d)} = \infty. \tag{22}
\]

and \( 0 < q(r) \leq 1 \). Along with \( \sum_{j = rd}^{(r + 1)d - 1} m^2(j) \to 0 \), it follows from Lemma A.1 in [18] that \( E[V(rd)] \to 0 \) as \( r \to \infty \). Consequently, for any \( \tau > 0 \), there exists \( r_2 \) such that when \( r > r_2 \), both \( E[V(rd)] < \tau \) and \( m^2(rd) < \tau \). Similar to the derivation for (21), we also obtain,

\[
E[V(rd + \ell)] \leq (1 - 2Q_{inf} d m((r + \ell - 1)d) + m^2((r + \ell - 1)d) Q_d) E[V(rd)] + F_d \sum_{j = rd} m^2(j), \tag{23}
\]

where \( 0 < \ell < d \) and \( \lambda_{inf} = \inf_{t \geq 0}(\lambda_m(L_{bt}^{d-1})) \). In this case, \( G_{inf}^d \) is not necessarily connected, therefore \( \lambda_{inf} = 0 \). Consequently, for any \( \tau > 0 \) and \( 0 < \ell < h \), there exists \( r > r_2 \) such that

\[
E[V(rd + \ell)] \leq (1 + m^2(0) Q_d) \tau + F_d \ell \tau = (1 + m^2(0) Q_d + F_d \ell) \tau. \tag{24}
\]

Together with \( E[V(rd)] < \tau \), we prove \( E[V(k)] \to 0 \) as \( k \to \infty \). Therefore, \( e_u(k) \) is mean square convergent to 0 for all \( u \in \Psi_b \).

**APPENDIX II**

**Lemma 2:** [24]. Denote by \( A \) an unknown linear, bounded, symmetric and positive operator on a real Hilbert space \( \mathbb{H} \), and we have to solve the equation \( Ax = y \) for an unknown \( y \in \mathbb{H} \). Assume that \( A^{-1} \) exists. We are given a sequence of linear, bounded operators \( A_k \) and a sequence \( y_k \in \mathbb{H} \), where \( k = 0, 1, \ldots \). In addition, suppose that \( \lim_{k \to \infty} \| \sum_{i=1}^{k} y_k - y \| = 0 \), \( \lim_{k \to \infty} \| \sum_{i=1}^{k} A_i - A \| = 0 \) and \( \lim_{k \to \infty} \| A_k \|^2 \) exists. Consider the sequence \( x_k : x_0 \) is arbitrary,

\[
x_{k+1} = x_k + \frac{c_1}{k + c_2} (y_k - A_k x_k), \tag{25}
\]

where \( c_1 \) and \( c_2 \) are constant real numbers. Then, \( \lim_{k \to \infty} x_k = A^{-1} y \).

**REFERENCES**


