Time Synchronization in WSNs: A Maximum-Value-Based Consensus Approach

Jianping He, Peng Cheng, *Member, IEEE*, Ling Shi, Jiming Chen, *Senior Member, IEEE*, and Youxian Sun

***Abstract*—This paper considers time synchronization in wireless sensor networks. When the communication delay is negligible, the maximum time synchronization (MTS) protocol is proposed by which the skew and offset of each node can be synchronized simultaneously. For a more practical case where the intercommunication delays between each connected node are positive random variables, we propose the weighted maximum time synchronization (WMTS), which is able to counteract the impact of random communication delays. Despite the clock offset that cannot be synchronized, WMTS can synchronize the clock skew completely in expectation and achieve acceptable synchronization accuracy. For both protocols, we provide rigorous proofs of global convergence as well as the upper bounds of their convergence time. Compared with existing consensus-based synchronization protocols, the main advantages of our protocols include: 1) a faster convergence speed so that the synchronization can be achieved in a finite time for MTS, and in a finite time in expectation for WMTS, respectively; 2) simultaneous synchronization of both skews and offsets; and 3) random communication delays can be handled effectively. Numerical examples are presented to demonstrate the effectiveness of the proposed protocols.**

***Index Terms*—Convergence, maximum consensus, time synchronization, wireless sensor network (WSN).**

I. Introduction

TIME synchronization is critical for many applications in wireless sensor networks (WSNs), such as mobile target tracking, event detection, speed estimation, environmental monitoring, etc. [1]–[3]. In these applications, it is crucial that all sensor nodes have a common time reference. Moreover, time synchronization also helps save energy in a WSN, since it provides the possibility of setting nodes into the sleeping mode [4].

Time synchronization has been studied extensively due to its importance and difficulty. Elson *et al.* [5] proposed a synchronization algorithm called reference-broadcast synchronization (RBS) for one-hop time synchronization, where a node is selected as the reference node and then broadcasts a reference message to all other nodes for synchronization. Ganeriwal *et al.* [6] proposed a timing-sync protocol for sensor networks (TPSN) for network-wide time synchronization. It first elects a root node and builds a spanning tree of the network, then the nodes are synchronized to their parents in the tree. However, the TPSN protocol can only compensate the relative clock offset, that is, the instantaneous clock difference, but not the clock skew, that is, the clock speed. Therefore, TPSN needs to send excessive messages for re-synchronization. In order to overcome these shortcomings, Maroti *et al.* [7] proposed the flooding-clock synchronization protocol (FTSP). The main idea is that the algorithm elects a root node and then the root node periodically floods its current time into the tree network. It should be noted that all three of these algorithms need a reference node or root node. Furthermore, TPSN, FTSP, and FBS are tree-based synchronization protocols, which are fragile to node failures. A detailed survey for time synchronization protocols in WSNs can be found in [2] and [8].

In order to enhance the robustness and scalability, various distributed protocols have been proposed for time synchronization in WSNs. For instance, Solis, *et al.* [9] proposed a fully asynchronous distributed time synchronization protocol (DTSC). Giridhar and Kumar [10] formulated DTSC as a coordinate-descent optimization problem. Recently, the consensus concept has been adopted to develop protocols for time synchronization in WSNs. A consensus-based time synchronization protocol has three main advantages. First, it can work in a distributed way. Therefore, it does not require a tree topology or a root node as a [11], [13], or [14]. Second, based on the consensus algorithm, more accurate synchronized clocks, especially for neighboring nodes, may be obtained for the entire network [15]. Third, it compensates the skew and offset differences among nodes. Existing consensus algorithms can be classified into two main categories: 1) synchronous algorithms, for example, [16], [17] and 2) asynchronous ones,for example, [18]. These two classes of consensus algorithms have been studied extensively for time synchronization in wired or wireless ad-hoc and peer-to-peer networks, for example, [4] and [12]–[15]. Usually, the node adjusts its clock by a mutually agreed consensus value after each node has learned the clock values of all its neighbors. For instance, Philipp and Roger [15] proposed a gradient time synchronization protocol (GTSP) which is designed to provide synchronized clocks between neighbors, and this protocol is mainly based on synchronous consensus algorithms [16]. Schenato and Fiorentin [14] proposed an Average TimeSynch (ATS) protocol, which is built

Manuscript received March 19, 2012; revised June 09, 2013; accepted October 08, 2013. Date of publication October 23, 2013; date of current version February 19, 2014. This paper was presented in part at the 50th IEEE Conference on Decision and Control, December 2011. This work was supported in part by NSFC61222305, 61290325-02, the SRFDP under Grant 20120101110139, and by an HKUST Grant RPC11EG34. Recommended by Associate Editor L. Schenato. *(Corresponding author: P. Cheng.)*

J. He, P. Cheng, J. Chen, and Y. Sun are with the State Key Laboratory of Industrial Control Technology, Zhejiang University, Hangzhou 310027, China (e-mail: jphe@iipc.zju.edu.cn; pcheng@iipc.zju.edu.cn; jmchen@iipc.zju.edu.cn; yxsun@iipc.zju.edu.cn).

L. Shi is with the Department of Electronic and Computer Engineering, Hong Kong University of Science and Technology, Kownloon 00852, Hong Kong, China (e-mail: eesling@ust.hk).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

on an asynchronous consensus algorithm. The main idea is to average local information to achieve a global agreement on a specific quantity of interest.

Communication delays, which are frequently seen in WSNs, are, however, omitted in GTSP and ATS. Taking the time delay into consideration, the distributed consensus timing synchronization (DCTS) algorithm may not be able to guarantee complete synchronization, for example, Xiong and Kishore [19] computed the asymptotic expectation and mean square of the global synchronization error of DCTS for WSNs assuming general Gaussian communication delays between nodes. Meanwhile, different estimation methods and algorithms are proposed to estimate the relative clocks between two neighbor nodes when the delay is considered [21]–[24]. For example, the relative clock was addressed from a statistical signal-processing viewpoint in [22] and Freris *et al.* [23] proposed a model-based approach to handle the estimation of the relative clock. Recently, based on the two-way message-exchange mechanism [6], [20], Mei and Wu [25] proposed three estimators for the joint estimation of the clock offset and skew without knowledge of the fixed delay. They also utilized belief propagation for achieving better synchronization accuracy than the average consensus-based algorithms [26]. However, only clock offset compensation was considered. Carli *et al.* in [27] and [28] regarded the different clock speeds as unknown constant disturbances and the different clock offsets as different initial conditions for the system dynamics, and developed a proportional-integral (PI) consensus-based controller to achieve time synchronization. Based on the second-order consensus algorithm [31], the proposed algorithms in [27] and [28] can compensate clock skews and offsets. They further consider time synchronization for WSNs under asymmetric gossip communication and optimal synchronization using the PI consensus-based controller in [29] and [30].

The convergence speed of time synchronization is also important in WSN applications. However, most existing average-consensus based protocols are time-consuming. Note, as pointed out by [14], that the exact value of the synchronized clock itself is not important as long as a consensus has been achieved. Hence it is of great interest to explore such characteristic to develop a protocol which provides much faster convergence time while maintaining the advantages brought by consensus algorithms. In this paper, we also consider time synchronization in WSNs. Being different from the aforementioned existing works, our main contributions are as follows.

1) We first consider the situation where the communication delays can be ignored, and propose a novel maximum time synchronization algorithm (MTS), using a maximum-value-based consensus algorithm, where the definition of max-consensus is first given in [32].
2) As, in many cases, the communication delay between nodes is critical and a fundamental limitation in time synchronization over WSNs [35], we consider the case where the random communication delay follows a normal distribution [7], and propose a weighted maximum time synchronization algorithm (WMTS) to solve the time synchronization problem.
3) Both protocols are completely asynchronous and distributed, hence robust to packet losses, node failure, replacement, or relocation. We provide rigorous proofs of the convergence. Furthermore, we prove that MTS will converge in a finite time and WMTS will converge in a finite time in expectation, and we present the upper bounds of the finite convergence time in a closed-form for the two protocols, respectively. Since the upper bounds grow linearly with the number of nodes, the convergence time is much smaller than that of the existing average-based consensus algorithms. Extensive simulations are conducted to demonstrate the effectiveness for static and dynamic communication topologies.

As the mean and the variance of the random communication delay are unknown to the nodes and there is no delay compensation in WMTS, the logical clock offset cannot be synchronized by WMTS completely. If there is prior knowledge of the random communication delay, for example, the mean of the random delay, we can exploit it to design delay compensation to increase synchronization accuracy.

The remainder of this paper is organized as follows. In Section II, the network and clock model for a WSN are introduced. The maximum-value-based protocol for time synchronization is proposed in Section III, where a proof for its convergence is provided as well as the analysis of the implementation of MTS. In Section IV, WMTS is presented for handling the random communication delay and a proof for its convergence is provided. Simulation results are shown in Section V. Some concluding remarks are given in Section VI.

II. Preliminaries and Problem Formulation

A. Network Model

We model a dynamic WSN as an undirected graph $\mathcal{G}(t) = (\mathcal{V}, \mathcal{E}(t))$ at each time t, where $\mathcal{V} = \{i : i = 1, ..., N\}$ is the set of nodes (sensors) and $\mathcal{E}(t) \subseteq \{(i, j) : i, j \in \mathcal{V}\}$ represents the set of available communication links (edges) at time t. The set of neighbors of node $i \in \mathcal{V}$ is denoted by $\mathcal{N}_i(t) = \{j \in \mathcal{V} : (i, j) \in \mathcal{E}(t)\}$. In this paper, we assume that $(i, j) \in \mathcal{E}(t)$ ($j \in \mathcal{N}_i(t)$) if and only if (iff) node i successfully receives information from node j at time t. We also assume the graph $\mathcal{G}(t)$ satisfies the following assumption.

Assumption 2.1: An integer $B \geq 1$ exists such that the graph $\left(\mathcal{V}, \cup_{t\in[0B)}\mathcal{E}(kB + t)\right)$ is connected for any non-negative integer k.

Assumption 2.1 is necessary as the time synchronization will be impossible for nodes in disconnected parts. The packet drop and collisions in communications will impact the connectivity of the communication graph, and a larger B is required to guarantee Assumption 2.1.

Throughout this paper, $\mathbf{N}^+$ denotes the set of positive integer, and $\mathbf{E}\{\cdot\}$ and $\mathbf{Var}\{\cdot\}$ denote the expectation and variance of a random variable, respectively.

B. Clock Model

In this section, we briefly introduce a linear clock model [14], [15]. Consider a WSN consisting of N sensors. Each sensor

node i is equipped with a hardware clock, whose clock reading $\tau_i(t)$ at time t is given by

$$\tau_i(t) = a_i t + b_i \tag{1}$$

where a_i is the hardware clock skew which determines the clock speed and b_i is the hardware clock offset. As pointed out in [15], the hardware clock skews may be slightly different from other nodes due to the imperfect crystal oscillators, ambient temperature, or battery voltage and oscillator aging. In order to facilitate the presentation, we assume that initially there is only one node with the maximum hardware clock skew throughout the network. We also assume that each hardware clock skew a_i satisfies

$$1 - p \le a_i \le 1 + p \tag{2}$$

where $0 \le p < 1$ is a constant and typically in the range of $p \in [10^{-5}, 10^{-4}]$ ($p \in [10, 100]$ ppm) [38], [39]. It has been pointed out that a_i and b_i cannot be computed as the absolute reference time t is not available to the sensor nodes [14], [35]. However, by comparing the local clock reading of one node i with another local clock reading of node j, we can obtain a relative clock between them, that is, $\tau_i(t) = \frac{a_i}{a_j}\tau_j(t) + (b_i - \frac{a_i}{a_j}b_j) = a_{ji}\tau_j(t) + b_{ji}$.

It is known that the value of the hardware clock cannot be adjusted manually, because other hardware components may depend on a continuously running hardware clock in WSNs [15]. Hence, generally, a logical clock is applied to represent the synchronized time. Specifically, a logical clock value $L_i(t)$ is a linear function of the hardware clock $\tau_i(t)$

$$L_i(t) = \hat{a}_i\tau_i(t) + \hat{b}_i = \hat{a}_i a_i t + \hat{a}_i b_i + \hat{b}_i \tag{3}$$

where $\hat{a}_i a_i$ and $\hat{a}_i b_i + \hat{b}_i$ are the logical clock skew and offset, respectively.

Initially, it is common to set a constant T for each node in advance so that each node broadcasts its message periodically with a period T based on its own hardware clock. To simplify the statements and proofs later on, the data collisions are not taken into account in this paper. In fact, the data collision will not affect the convergence of our protocols. It turns out that under a common broadcast period T, if all clocks are the same (or have the same skews and small offsets), collisions will occur constantly. One way to tackle this is by allowing communication asynchronously at random times, see, for instance, [11] and [16]. It should also be noted that there are many media-access control (MAC) protocols, for example, time-division multiple access (TDMA) and carrier-sense multiple access (CSMA), which can be easily incorporated for collision avoidance. The introduction of such a MAC protocol will not affect the main results of our paper. Moreover, Freris and Zouzias in [42] propose a randomized asynchronous approach which can be further adopted within each period T to avoid the collision.

Definition 2.2: Assume that every node broadcasts its message periodically with a common period T based on its own hardware clock. T is defined as the pseudoperiod [14].

For each node i, its real period T_i is given as $T_i = \frac{T}{a_i}, i \in \mathcal{V}$. Since the clock skew $a_i, i \in \mathcal{V}$ is slightly different in general, the real period $T_i \neq T_j$ for $i \neq j$. Under Assumption 2.1, we have

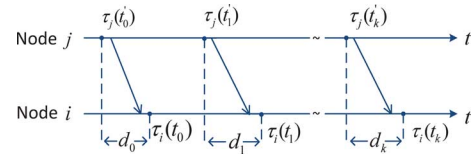


Fig. 1. Uncertainty time delay.

$T_i < B$ for $\forall i \in \mathcal{V}$ and $T < B$. Otherwise, the neighbor nodes may not successfully receive information from node i within the time interval $[kB, (k+1)B)$, which contradicts Assumption 2.1. The second key parameter is called relative skew, which is defined as follows.

Definition 2.3: The relative skew a_{ij} is defined as

$$a_{ij} = \frac{a_j}{a_i}, i, j \in \mathcal{V}. \tag{4}$$

Since the skew compensation is always based on relative skew a_{ij}, the relative skew plays an important role in a synchronization protocol. The value of a_{ij} cannot be computed by (4) directly as a_i and a_j are unavailable, but it can be estimated with their hardware clock readings. The details will be given later on.

C. Delay Model

Communication delay has a direct impact on the accuracy of synchronization. Delays in packet delivery constitute a fundamental limitation in synchronizing clocks over WSNs [35]. In this paper, we will consider two cases as follows:

1) Delay Free: There is no transmission or reception delay.

2) Random Delay: The communication delays at different times are identically and independently distributed (i.i.d) positive random variables with constant mean μ and variance σ^2.

Taking Fig. 1 as an example, node j periodically broadcasts its hardware clock readings $\tau_j(t_k')$ to its neighbor node i, and node i records its hardware clock readings as $\tau_i(t_k)$ for $k = 0, 1, ...$, where t_k' and t_k are the real broadcasting time and the receiving time for node j and node i, respectively. Thus, $d_k = t_k - t_k'$ is the communication delay of the $(k+1)$th communication from nodes j to i.

Let $\theta_{ij}(k) = \frac{d_{k-1} - d_k}{t_k - t_{k-1}}, i, j \in \mathcal{V}$ for $k \in \mathbf{N}^+$, where d_k and d_{k-1} denote the random delay of two consecutive communications, respectively. Then, it follows that the mean and variance of each random variable $\theta_{ij}(k)$ are equal to 0 and $\frac{2\sigma^2}{(t_k - t_{k-1})^2}$, respectively. Moreover, one infers that $\mathbf{E}\{\theta_{ij}(k)\theta_{ij}(k+1)\} = \frac{-\sigma^2}{(t_k - t_{k-1})(t_{k+1} - t_k)}$. Since d_k are i.i.d, the random variables $\theta_{ij}(k_1)$ and $\theta_{ij}(k_2)$ are also independent of each other for $|k_1 - k_2| > 1$. Hence, $\mathbf{E}\{\theta_{ij}(k_1)\theta_{ij}(k_2)\} = 0$.

D. Problems of Interest

The objective of time synchronization is to synchronize all of the nodes with respect to a common clock, given as

$$\tau_v(t) = a_v t + b_v. \tag{5}$$

There always exists $(\hat{a}_i, \hat{b}_i)$ for each logical clock $L_i(t)$ such that $L_i(t) = \tau_v(t)$ for $i \in \mathcal{V}$.

Our goal in this paper is to design a consensus-based clock synchronization algorithm to find $(\hat{a}_i(k), \hat{b}_i(k))$ for each node i, such that

$$\begin{cases} \lim\limits_{k\to\infty} \hat{a}_i(k)a_i = a_v, \\ \lim\limits_{k\to\infty} \hat{a}_i(k)b_i + \hat{b}_i(k) = b_v. \end{cases} \tag{6}$$

As pointed by [14], the real values of $(a_v; b_v)$ are not important; however, it is important that all clocks converge to one common virtual reference clock, and the final parameters $(a_v; b_v)$ only depend on the initial condition and the communication topology of the WSN.

Remark 2.4: The linear clock model is widely used for time synchronization in WSNs, for example, [12]–[14] and [35]. Note that a_i and b_i may be slowly time varying in practice. However, as long as the synchronization algorithm can achieve certain accuracy in a short time, the algorithm can be restarted once the synchronization error exceeds a given bound.

III. MTS Protocol: Delay-Free Case

A. Maximum Time Synchronization (MTS)

From (6), it is desirable to synchronize the clock skew and offset simultaneously. Since each hardware clock is a linear function of time t and the communication delay between two connected nodes is omitted in the *Delay Free* case, each relative skew a_{ij} can be computed by

$$a_{ij} = \frac{\tau_j(t_1) - \tau_j(t_0)}{\tau_i(t_1) - \tau_i(t_0)},\ i, j \in \mathcal{V} \tag{7}$$

where $(\tau_i(t_1), \tau_j(t_1))$ and $(\tau_i(t_0), \tau_j(t_0))$ are two pair of hardware clock readings of nodes i and j with $t_1 \neq t_0$. In fact, when node i receives time message $\tau_j(t_0)$ from node j, it reads its current hardware clock reading $\tau_i(t_0)$ and stores $(\tau_i(t_0), \tau_j(t_0))$; meanwhile, when node i receives the time message from node j at the second time, the relative skew a_{ij} can be obtained from (7) based on these two pairs of historical records. We summarize the aforementioned process in the following algorithm (Maximum Time Synchronization).

Algorithm 1 : *Maximum Time Synchronization (MTS)*

1:Given the initial conditions $\hat{a}_i = 1$ and $\hat{b}_i = 0$ for $i \in \mathcal{V}$, set the common broadcast period T to each node.

2:If the current hardware clock reading of node $i, i \in \mathcal{V}$, satisfies $\frac{\tau_i(t)}{T} \in \mathbf{N}^+$, then node i broadcasts its local hardware clock reading $\tau_i(t)$ and $(\hat{a}_i(t), \hat{b}_i(t))$ to its neighbors.

3:If node $i \in \mathcal{V}$ receives a packet from its neighbor node j at time t_1, then it records its local hardware clock reading $\tau_i(t_1)$ and records the packet information as $[\tau_j(t_1), (\hat{a}_j(t_1), \hat{b}_j(t_1))]$.

4:If node i has a historical record $[\tau_i(t_0), \tau_j(t_0)]$, then compute a_{ij} from according to **(7)** and q_{ij} by $q_{ij} = \frac{a_{ij}\hat{a}_j}{\hat{a}_i}$. After that, delete the record $[\tau_i(t_0), \tau_j(t_0)]$.

5:If $q_{ij} > 1$, then

$$\hat{a}_i \leftarrow a_{ij}\hat{a}_j,$$
$$\hat{b}_i \leftarrow \hat{a}_j(t_1)\tau_j(t_1) + \hat{b}_j - a_{ij}\hat{a}_j\tau_i(t_1).$$

If $q_{ij} = 1$, then

$$\hat{b}_i \leftarrow \max_{k=i,j}\{\hat{a}_k(t_1)\tau_k(t_1) + \hat{b}_k\} - \hat{a}_i\tau_i(t_1).$$

6:Store $[\tau_i(t_1), \tau_j(t_1)]$.

In step 4, q_{ij} satisfies $q_{ij} = \frac{\hat{a}_j a_j}{\hat{a}_i a_i}$, which is the ratio of two nodes' logical clock skew. Hence, node i can know from q_{ij} whether its neighbor node j has the larger logical clock skew. It follows from step 5 that for MTS, the logical clock skew and offset are adjusted simultaneously, and the fastest logical clock among neighbor nodes is selected as the reference clock.

Remark 3.1: Since the protocol drives all nodes throughout the network to approach the maximum hardware clock, that is, maximum-consensus, we name the protocol as Maximum Time Synchronization. According to the protocol, the skew and offset compensations are conducted simultaneously and completed at the same time.

Remark 3.2: For an arbitrary node i, its hardware clock reading satisfies $\tau_i(t_k^i) = kT$, where t_k^i denotes the reality time at the kth broadcast instant. Thus, we have $t_k^i = \frac{kT - b_i}{a_i}$ for $i \in \mathcal{V}$.

B. Convergence of MTS

Before proving the convergence of MTS, we first introduce some preliminarily results. Define $a_{max} = \max\limits_{i\in\mathcal{V}} a_i$. We use $\mathcal{V}_{max}$ to denote the set of these nodes whose clock skews are equal to a_{max}. Define $b_{max} = \max\limits_{i\in\mathcal{V}_{max}} b_i$. Let $a_v = a_{max}$ and $b_v = b_{max}$, then the common clock (5) can be rewritten as

$$\tau_v(t) = a_v t + b_v = a_{max}t + b_{max}. \tag{8}$$

Let $\mathcal{V}_v(t)$ be the set of nodes whose logical clock skew and offset are equal to a_{max} and b_{max} at time t, respectively. Clearly, the logical clock of nodes in $\mathcal{V}_v(t)$ are equivalent to the common clock $\tau_v(t)$ described in (8). Since the initial condition satisfies $\hat{a}_i = 1$ and $\hat{b}_i = 0$ for $i \in \mathcal{V}$ in MTS, according to the definition of $\mathcal{V}_v(t)$, it is clear that there exists at least one node whose hardware clock is equal to the common clock at the initial time, that is, $\mathcal{V}_v(0) \neq \varnothing$.

Let $N_v(t)$ be the number of nodes belonging to the set $\mathcal{V}_v(t)$ at time t. Define a function

$$V(t) = N - N_v(t). \tag{9}$$

Since $\mathcal{V}_v(0) \neq \varnothing$, $V(0) \leq N - 1$. Clearly, $V(t) = 0$ iff $L_i(t) = \tau_v(t)$ for $\forall\, i \in \mathcal{V}$, which is equivalent to that $V(t) = 0$ if and only if $\mathcal{V}_v(t) = \mathcal{V}$.

Theorem 3.3: Under Assumption 2.1 and using MTS, the skew and offset converge and satisfy

$$\begin{cases} \lim\limits_{k\to\infty} \hat{a}_i(k)a_i = a_v, \\ \lim\limits_{k\to\infty} \hat{a}_i(k)b_i + \hat{b}_i(k) = b_v \end{cases} \tag{10}$$

where $a_v = a_{max}$ and $b_v = b_{\max}$.

Proof: The proof is given in Appendix A. ■

In order to simplify the statements, we make the following assumptions:

Assumption 3.4: Assume that $(i,j) \in \cup_{t\in[0,B)}\mathcal{E}(kB+t)$ iff node i receives messages more than once successfully from node j within the time interval $[kB,(k+1)B)$ for all positive integers k.

Based on this assumption, node i is defined to be neighboring with node j in the time-interval $[kB,(k+1)B)$ iff node j can receive messages more than once successfully from node i within this interval.

Theorem 3.5: Under Assumptions 2.1 and 3.4, the convergence time T_{con} of MTS satisfies $T_{con} \leq B(N-1)$.

Proof: From the proof of Theorem 3.3, we know that $V(t)$ is a nonincreasing function. Assume $V(t) > 0$ at time t (where $t = mB, m \in \{0, \mathbf{N}^+\}$). Then, at each time interval $[t, t+B)$, Assumption 2.1 ensures that there is at least an edge $\{i,j\}$ which connects one node $i, i \in \mathcal{V} - \mathcal{V}_v(t)$ and another node $j, j \in \mathcal{V}_v(t)$. Assumption 3.4 ensures that node i can successfully receive messages from node j at least twice within the time interval $[t, t+B)$, which means that node i can set its logical clock equal to the logical clock of node j before $t+B$. Hence, we have $i \in \mathcal{V}_v(t+B)$, that is, $V(t+B) = V(t) - 1$. Since $V(0) \leq N-1$, $V((N-1)B) = V(0) - N + 1 \leq 0$. It follows from the definition of $V(t)$ that $V(t) \geq 0$, then $V((N-1)B) = 0$. Therefore, we have $T_{\rm con} \leq B(N-1)$. ■

Remark 3.6: For Theorems 3.3 and 3.5, Assumption 2.1 is used to ensure that there are nodes in the set $\mathcal{V} - \mathcal{V}_v(t)$ which can receive packets from the nodes in set $\mathcal{V}_v(t)$. Clearly, we can replace it by the following assumption, which is commonly adopted in related works, for example, [17] and [35].

Assumption 3.7: An integer $B \geq 1$ exists such that the graph $\left(\mathcal{V}, \cup_{t\in[0B)}\mathcal{E}(kB+t)\right)$ is strongly connected for all non-negative integers k.

Remark 3.8: From the aforementioned theorems, MTS makes all nodes synchronize their logical clocks as the hardware clock of node i with $a_i = a_{\max}$. Note that each a_i satisfies (2) with $p \in [10, 100]$ ppm, which means that each a_i is approximately equal to 1, and all nodes' logical clocks will be slightly faster or slower than the ideal time t. If there is a misbehaving node j with $a_j \gg 1$ in the network, its neighbor node i will observe $a_{ij} = \frac{a_j}{a_i} \gg 1$ from (7). Hence, node i will easily know that node j is an irregular node and isolate it. We can also set an upper bound, for example, $\frac{1+p}{1-p}$, for each node i, so that node i will not utilize the information from its neighbor node j if it detects $a_{ij} > \frac{1+p}{1-p}$. In this way, we can avoid the situation that the misbehaving node drives the clock speed much faster than the real time. Secure time synchronization under general malicious attacks, which is another interesting while challenging problem [40], [41], is beyond the scope of this paper, and will be pursued in our future research.

C. Analysis of MTS

Compared with the traditional tree-based or reference (root)-based time synchronization algorithms, for example, RBS [5], TPSN [6] and FTSP [7], etc., MTS has the advantage of being fully distributed, asynchronous, robust to packet drop, sensor node failure and new node joining, and it is adaptive to time-varying communication topology. Different from traditional time synchronization protocols, such as FTSP, MTS does not require setting a root node or building up a tree topology in advance, which largely reduces the implementing costs and enhances system robustness especially when the root node or nonleaf nodes are prone to various faults. Moreover, MTS can conduct both skew and offset compensation at the same time and complete them simultaneously in finite time.

Note that different from the existing average consensus algorithms, the basic idea of our approach is to drive the logical clocks to the maximum value among all nodes so that the network can achieve time synchronization. In the algorithm, each node i periodically broadcasts a packet containing its local hardware clock reading $\tau_i(t)$ and its current relative logical clock skew $\hat{a}_i(t)$ and the offset $\hat{b}_i(t)$, without requiring any feedback information from its neighbors. For each neighbor node j, a pair of hardware clock readings $[\tau_i(t), \tau_j(t)]$ needs to be stored by node i. Compared with the recently developed consensus-based time synchronization algorithms, that is, GTSP [15] and ATS [14], the messages for MTS broadcast and storage are the same as these algorithms. Recently, Carli *et al.* in [28] and [29] proposed a promising average consensus-based time synchronization algorithm, where a distributed PI control law, based on the standard linear average consensus algorithm, is developed to achieve time synchronization. Although this algorithm achieves time synchronization asymptotically as in ATS and GTSP, it requires less computation and communication capabilities as each node has to keep in memory only two variables. During the communication, each node only transmits the value of its own logical clock while in the ATS, GTSP, and MTS algorithms, and each node has to keep in memory a number of variables which are proportional to the number of its neighbors. Comparing these average consensus-based time synchronization algorithms, the main advantage of MTS is its much faster convergence speed, that is, the average consensus-based time synchronization algorithms converge to global synchronization asymptotically, while MTS converges to global synchronization in a finite time.

Energy cost is a major concern in WSNs. As the computational energy cost is comparably small due to the simple algorithm, we will focus on the communication energy cost, which can be measured by the broadcasting times. In the previous subsection, we have obtained an upper bound of the convergence time of MTS. Denote N_i^c as the broadcasting times of node i from the beginning to the time when the algorithm has just converged. Then based on Theorem 3.5, we obtain an upper bound of N_i^c, given by

$$N_i^c \leq \frac{B(N-1)}{T_i}, i \in \mathcal{V}. \tag{11}$$

Assume that every broadcast of all nodes costs the same amount of energy E and let E_c be the total energy cost for MTS to convergence, then

$$E_c \leq E\sum_{i\in\mathcal{V}}\frac{B(N-1)}{T_i} = \frac{EB(N-1)}{T}\sum_{i\in\mathcal{V}}a_i. \tag{12}$$

From (12), it is not difficult to see that enlarging the common period T can decrease the upper bound of E_c. However, a larger T will require a larger B in order to ensure the connectivity, which

may slow down the finite convergence time (seen from Theorem 3.5). Thus, it is a tradeoff between energy cost and convergence time. Obtaining an optimal T is an interesting problem which depends on the real applications, and will be our future work.

IV. WMTS Protocol: Random Delay Case

There are also many cases when the communication delay between nodes is considerably large, which may severely affect the performance of time synchronization if ignored.

Consider the random communication delay as modeled in Section II. Note that each relative skew $a_{ij}, i, j \in \mathcal{V}$, in MTS is computed based on two adjacent communications, i.e.,

$$\begin{aligned} a_{ij}(k) &= \frac{\tau_j(t'_k) - \tau_j(t'_{k-1})}{\tau_i(t_k) - \tau_i(t_{k-1})} \\ &= \frac{a_j(t_k - d_k - t_{k-1} + d_{k-1})}{a_i(t_k - t_{k-1})} \\ &= \frac{a_j}{a_i}(1 + \theta_{ij}(k)) \end{aligned} \tag{13}$$

where $\theta_{ij}(k) = \frac{d_{k-1} - d_k}{t_k - t_{k-1}}$. Since d_k is a random variable, $\theta_{ij}(k)$ is also a random variable. Thus, for each k, it may be true that $d_k \neq d_{k-1}$, which leads to $\theta_{ij}(k) \neq 0$. As a result, $a_{ij}(k) \neq \frac{a_j}{a_i}$. This phenomenon will directly impact the accuracy of the synchronization and even make the logical clock diverge, as illustrated by the following example.

1) Example 4.1: Consider a network with two nodes indexed, respectively, by 1 and 2. According to MTS, if node 1 receives information t times successfully from node 2 at time t, then node 1 will update its logical clock based on $q_{12}(t) = \frac{a_{12}(t)\hat{a}_2(t)}{\hat{a}_1(t)}$. Since $a_{12}(t)$ follows (13), that is, $a_{12}(k) = \frac{a_2}{a_1}(1 + \theta_{12}(t))$, we have $q_{12}(t) = \frac{a_2\hat{a}_2(t)}{a_1\hat{a}_1(t)}(1 + \theta_{12}(t))$. Then, from step 5 of MTS, we have $a_1\hat{a}_1(t^+) = a_2\hat{a}_2(t)(1 + \theta_{12}(t))$ for $q_{12}(t) \geq 1$ and $a_1\hat{a}_1(t^+) = a_1\hat{a}_1(t)$ for $q_{12}(t) < 1$, that is, $a_1\hat{a}_1(t^+)$ neither satisfies $a_1\hat{a}_1(t^+) = a_2\hat{a}_2(t)(1 + \theta_{12}(t)) \geq a_1\hat{a}_1(t)$ nor $a_1\hat{a}_1(t^+) = a_1\hat{a}_1(t) > a_2\hat{a}_2(t^+)(1 + \theta_{12}(t))$. Thus, node 1 is not able to fully synchronize its logical clock as node 2 since $\theta_{12}(t)$ is random and $\theta_{ij}(t) = 0$ with probability 0. Note that $a_1\hat{a}_1(t^+) \geq a_1\hat{a}_1(t)$ and $a_1\hat{a}_1(t^+) \geq a_2\hat{a}_2(t)(1 + \theta_{12}(t))$, which means $a_1\hat{a}_1(t)$ is nondecreasing and it will be updated to be equal to or larger than $a_2\hat{a}_2(t)(1 + \theta_{12}(t))$ after node 1 receives information from node 2 at time t. By a similar argument, for node 2, we also have $a_2\hat{a}_2(t)$ as nondecreasing and it will be updated to be equal to or larger than $a_1\hat{a}_1(t)(1 + \theta_{21}(t))$ after node 2 receives information from node 1 at time t. Therefore, with the increasing communication between nodes 1 and 2, the logical clock skew of them increases continually with positive probability. Note that if the communication delay is equal to a constant, then we will have $\theta_{ij}(t) = 0$ in (13), and then the entire network synchronization can still be guaranteed by MTS.

Therefore, when the random delay in communication of the networks cannot be ignored, we need to revise the MTS algorithm accordingly so that time synchronization can still be achieved. In this section, we adopt the random delay model in Section II, and extend the MTS algorithm to handle the random communication delays.

A. Weighted Maximum Time Synchronization (WMTS)

The accuracy of $a_{ij}, i, j \in \mathcal{V}$ is crucial for time synchronizationl; thus, we will introduce a new method to estimate these relative skews. For $i, j \in \mathcal{V}$, let $a_{ij}(0) = 1$. Assume that a node i successfully obtains the messages from node j at time $t, t = t_0, t_1, ..., t_k$, respectively. The following equation is used to estimate a_{ij}:

$$a_{ij}(k) = \frac{\frac{\tau_j(t'_k) - \tau_j(t'_{k-1})}{\tau_i(t_k) - \tau_i(t_{k-1})} + (k-1)a_{ij}(k-1)}{k},\ i, j \in \mathcal{V} \tag{14}$$

where $k \in \mathbf{N}^+$. Equation (14) is an averaging process which is a classical stochastic approximation approach [37]. With the increase of k, $a_{ij}(k) = \frac{a_j}{a_i}$ almost surely, which is the key to guarantee the convergence of WMTS.

For $i, j \in \mathcal{V}$, let $\theta^0_{ij} = 0$, and let $\theta^k_{ij} = \frac{\Sigma^k_{l=1}\theta_{ij}(l)}{k}$ for $k \in \mathbf{N}^+$. Since each $\theta_{ij}(l)$ satisfies $\mathbf{E}\{\theta_{ij}(l)\} = 0$, we have $\mathbf{E}\{\theta^k_{ij}\} = 0$ and

$$\mathbf{Var}\{\theta^k_{ij}\} = \frac{\Sigma^k_{l=1}\left(\frac{2\sigma^2}{(t_l - t_{l-1})^2}\right) - \Sigma^{k-1}_{l=1}\left(\frac{2\sigma^2}{(t_l - t_{l-1})(t_{l+1} - t_l)}\right)}{k^2}. \tag{15}$$

Lemma 4.2: Assume that $a_{ij},\ i, j \in \mathcal{V}$ is obtained by (14). Then

$$\mathbf{E}\{a_{ij}(k)\} = \frac{a_j}{a_i}, k \in \mathbf{N}^+ \tag{16}$$

and

$$\lim_{k\to\infty} \mathbf{Var}\{a_{ij}(k)\} = 0. \tag{17}$$

Proof: From (13), one notes that

$$\frac{\tau_j(t'_k) - \tau_j(t'_{k-1})}{\tau_i(t_k) - \tau_i(t_{k-1})} = \frac{a_j}{a_i}(1 + \theta_{ij}(k)), k \in \mathbf{N}^+.$$

Substituting the above equation into (14) yields

$$\begin{aligned} a_{ij}(k) &= \frac{\frac{a_j}{a_i}(1 + \theta_{ij}(k)) + (k-1)a_{ij}(k-1)}{k} \\ &= \frac{a_j}{a_i}\left(1 + \frac{\Sigma^k_{l=1}\theta_{ij}(l)}{k}\right) = \frac{a_j}{a_i}\left(1 + \theta^k_{ij}\right). \end{aligned} \tag{18}$$

Taking expectation on both sides of (18) yields (16). By computing the variance on both sides of (18), we have $\mathbf{Var}\{a_{ij}(k)\} \leq \frac{a_j^2}{k^2 a_i^2}\sum^k_{l=1}\left(\frac{2\sigma^2}{(t_l - t_{l-1})^2}\right)$. Note that $t_k - t_{k-1}$ is the time interval between two consecutive instances when a node successfully receives messages from one neighbor node, hence $t_k - t_{k-1} > (1-p)T$. Therefore

$$\mathbf{Var}\{a_{ij}(k)\} \leq \frac{1}{k}\frac{a_j^2}{a_i^2}\frac{2\sigma^2}{T^2(1-p)^2}, k \in \mathbf{N}^+. \tag{19}$$

Clearly, $\mathbf{Var}\{a_{ij}(k)\} \geq 0$ for $i, j \in \mathcal{V}$. Taking limitation on both sides of (19) yields (17). ■

In order to tackle the problems introduced by the communication delays, we adapt MTS to WMTS as follows.

Algorithm 2 : *Weighted Maximum Time Synchronization (WMTS)*

1:Given the initial conditions $\hat{a}_i = 1, \hat{b}_i = 0, w_i = 0$ and $r_i = i$ for $i \in \mathcal{V}$. Set a common broadcast period T to each node.

2:If $\tau_i(t) = lT, l \in \mathbf{N}^+$, then node i broadcasts packet including $\tau_i(t)$, w_i, r_i and $(\hat{a}_i, \hat{b}_i)$ to its neighbors.

3:If node $i \in \mathcal{V}$ receives the packet $(k+1)$-th time from node j at time t_k, then record its local hardware clock reading and received clock information as $\tau_i(t_k)$ and $\tau_j(t_k)$, respectively, for $k \in \{\mathbf{N}^+, 0\}$.

4:If $k \geq 1$, then compute $a_{ij}(k)$ according to **(14)**, and then compute $q_{ij}(k) = \frac{a_{ij}(k)\hat{a}_j}{\hat{a}_i}$.

5:If $r_i \neq r_j$ and $q_{ij}(k) > 1$ or if $r_i = r_j$ and $w_i > w_j$, then

$$w_i \leftarrow w_j + 1, r_i \leftarrow r_j,$$
$$\hat{a}_i \leftarrow a_{ij}(k)\hat{a}_j, \hat{b}_i \leftarrow \hat{a}_j\tau_j(t_k) + \hat{b}_j - \hat{a}_i\tau_i(t_k).$$

6:If $r_i \neq r_j$ and $q_{ij}(k) = 1$, then compare $L_i(t_k)$ with $L_j(t_k)$, where $L_i(t_k) = \hat{a}_i\tau_i(t_k) + \hat{b}_i$ and $L_j(t_k) = \hat{a}_j\tau_j(t_k) + \hat{b}_j$. If $L_i(t_k) < L_j(t_k)$, then

$$r_i \leftarrow r_j, w_i \leftarrow w_j + 1,$$
$$\hat{b}_i \leftarrow L_j(t_k) - \hat{a}_i\tau_i(t_k).$$

7:Store $a_{ij}(k)$ and $[\tau_i(t_k), \tau_j(t_k)]$.

In WMTS, there are two decision variables r_i and w_i for each node $i \in \mathcal{V}$, where r_i denotes the source reference node of node i, that is, node i has updated its logical clock based on the estimate of the logical clock information of node r_i, and w_i represents the number of hops that the logical clock information of node r_i is transmitted from the source reference node r_i to node i. Initially, we set the reference number $r_i = i$ and weight $w_i = 0$ for $i \in \mathcal{V}$, which means that node i has not yet updated its logical clock by referring to any other nodes. During the iterations of WMTS, if two neighbor nodes i and j obtain different reference numbers, that is, $r_i \neq r_j$, then the node with a larger logical clock will be selected as the reference node; if nodes i and j have the same reference number but different weight numbers, that is, $r_i = r_j$, $w_i \neq w_j$, then the node with a smaller weight number will be selected as the reference node. Meanwhile, after one node is selected as the reference node (assumed to be node j), node i will adjust its decision variables and logical clock, such that $r_i = r_j$, $w_i = w_j + 1$, and $L_i = L_j$, while node j will not update its decision variables and logical clock based on the messages received from node i. Hence, with the use of decision variables r_i and w_i, two neighbor nodes with the same reference numbers cannot be the reference node to each other, and the phenomenon mentioned in Example 4.1 will not occur under WMTS.

B. Convergence of WMTS

Before proving the convergence of WMTS, we introduce a few preliminary results.

Define $\Theta \triangleq \{\theta_{ij}^{k(t)} | i \neq j, i, j \in \mathcal{V}\}$, where $\theta_{ij}^0 = 0$ and $\theta_{ij}^{k(t)}$ ($k(t) \in \mathbf{N}^+$ is the number of successful times that node j receives information from node i before time t) is a random variable with $\mathbf{E}\{\theta_{ij}^{k(t)}\} = 0$ and

$$\mathbf{Var}\{\theta_{ij}^{k(t)}\} \leq \frac{1}{k(t)} \frac{2\sigma^2}{T^2(1-p)^2}. \quad (20)$$

Considering node i with $r_i(t)$ and $w_i(t) \geq 1$ at time t, one can infer that node i has obtained the logical clock information of node $r_i(t)$ from a node j which has $r_j(t') = r_i(t)$ and $w_j(t') = w_i(t) - 1$ at time $t', t' \in [0, t]$, and node i has updated its parameters $\hat{a}_i$ and $\hat{b}_i$ based on the logical clock information of node $r_i(t)$ obtained from node j. Thus, we have $\hat{a}_i(t)a_i = \hat{a}_i(t')a_i = \hat{a}_j(t')a_j\left(1 + \theta_{ij}^{k(t')}\right)$ and $L_i(t) = L_i(t') + (t - t')\hat{a}_i(t')a_i = L_j(t') - \hat{a}_j(t')a_j d(t') + (t - t')\hat{a}_i(t')a_i$, where $\theta_{ij}^{k(t')} \in \Theta$ and $k(t')$ is the number of times that node i successfully receives messages from node j within the time interval $[0, t']$. Similarly, for node j, one can infer that it has obtained the logical clock information of node $r_i(t)$ from a node l which has $r_l(t'') = r_i(t)$ and $w_l(t'') = w_j(t') - 1 = w_i(t) - 2$ at time $t'', t'' \in [0, t']$. Therefore, given a time t, for node i with $r_i(t)$ and $w_i(t)$, node sequence i_k and time sequence t_k exists with $k = 1, 2, ..., w_i(t)$, such that the information $r_i(t)$ is transmitted through the following path to node i

$$i_0 \rightarrow i_1 \rightarrow i_2 \rightarrow \cdots \rightarrow i_{w_i(t)}$$

where $i_0 = r_i(t)$ and $i_{w_i(t)} = i$, and each node i_k receives information of node $r_i(t)$ from node i_{k-1} at time t_{k-1}. Moreover, each node i_k in the path finishes the last change of its logical clock within time $[t_{k-1}, t_k]$ based on the information of node $r_i(t)$ obtained from node i_{k-1} at time t_{k-1}. Thus, for $k = 1, 2, ..., w_i(t)$, we have $r_{i_k}(\tau) = r_i(t)$ and $w_{i_k}(\tau) = k$ for $\tau \in [t_{k-1}, t_k]$, which means that these nodes have the same reference number but a different weight number. Hence, given a time t, for each node i with $r_i(t)$ and $w_i(t)$, we can find a node path $i_0 \rightarrow i_1 \rightarrow i_2 \rightarrow, ..., \rightarrow i_{w_i(t)}$ to transmit the logical clock information of reference node $r_i(t)$ to node i, where these nodes in the path have the same reference number and different weights. Note that the nodes in the path will be, in general, distinct. The reason is as follows. If there are two nodes (say i_k and i_l) in $i_0, i_1, i_2, ..., i_{w_i(t)}$, satisfying $i_k = i_l = j$, which means that node j has the same reference number but different weights at time t_l and t_k, we can infer the following two events:

1) Node j has changed its reference number at a time $t', t' \in (t_k, t_l)$, that is, $r_j(t_k) = r_j(t_l) \neq r_j(t')$.
2) There are nodes $i_{k+1}, i_{k+2}, ..., i_{l-1}$, in node sequence $i_0, i_1, i_2, ..., i_{w_i(t)}$, where $k + 1 \neq l - 1$, which means there are at least two nodes which have obtained the logical clock information of node $r_i(t)$ from node j.

Note that $\mathbf{E}\{\theta_{i_k i_{k-1}}^{k(t)}\} = 0$; thus, for $k = 1, 2, ..., w_i(t)$, we have $\mathbf{E}\{\hat{a}_{i_k}(t)a_{i_k}\} = \hat{a}_{i_{k-1}}(t)a_{i_{k-1}}$ for $t \in [t_{k-1}, t_k]$. Since the reference node i_0 satisfies $w_{i_0}(t) = 0$ for $t \in [t_0, t_1]$, it follows that $\hat{a}_{i_0}(t)a_{i_0} = a_{i_0}$ holds for $t \in [t_{k-1}, t_k]$. Meanwhile, for WMTS, if two nodes x and y with different reference numbers contact each other at time t, only when $q_{xy}(t) = \frac{a_y\hat{a}_y(t)}{a_x\hat{a}_x(t)}(1 + \theta_{xy}^{k(t)}) \geq 1$, node x will change its reference number and update its logical clock skew such that $a_x\hat{a}_x(t^+) = a_y\hat{a}_y(t)(1 + \theta_{xy}^{k(t)})$, which increases the logical clock skew of node x, that is, the logical clock skew of each node is increasing when it changes its reference number. Hence, for the first fact, one can infer that $a_{r_j(t_k)} \leq a_{r_j(t')} \leq a_{r_j(t_l)}$ in expectation, and then $a_{r_j(t')} = a_{r_j(t_k)}$ as $r_j(t_k) = r_j(t_l)$. However, the hardware clocks of nodes $r_j(t')$ and $r_j(t_k)$ are, in general, different from

each other as $r_j(t') \neq r_j(t_k)$. Thus, the first event will not occur in most cases. Furthermore, from WMTS, if two neighbor nodes u and v have the same reference number, then $w_u = w_v$ or $|w_u - w_v| = 1$ holds, which means that the second event cannot occur in static networks or networks having only new nodes joining and nodes failure. Note that only when the aforementioned events occur simultaneously, there will be repeated nodes in node sequence $i_0, i_1, i_2, ..., i_{w_i(t)}$. Therefore, without loss of generality, we assume that the nodes in node sequence $i_0, i_1, i_2, ..., i_{w_i(t)}$ are different from each other in the remainder of this paper.

Let $\theta_{i_0 i_{-1}}^{k(t_{-1})} = 0$, $i_0 = r_i(t)$ and $i_{w_i(t)} = i$, and recall that $d(t)$ is the random communication delay of two nodes at time t, which follows a normal distribution.

Theorem 4.3: Suppose that node i has $w_i(t)$ and $r_i(t)$ at time t. Then, for $x = 0, 1, ..., w_i(t)$, there are node sequence i_x and time sequence t_x, such that

$$\hat{a}_i(t)a_i = a_{r_i(t)} \prod_{x=0}^{w_i(t)} \left(1 + \theta_{i_x i_{x-1}}^{k(t_{x-1})}\right) \tag{21}$$

and

$$\begin{aligned} L_i(t) = &\sum_{l=1}^{w_i(t)} \left((t_l - t_{l-1}) a_{r_i(t)} \prod_{x=1}^{l} \left(1 + \theta_{i_x i_{x-1}}^{k(t_{x-1})}\right) \right) \\ &- \sum_{l=1}^{w_i(t)} \left(a_{r_i(t)} \prod_{x=0}^{l-1} \left(1 + \theta_{i_x i_{x-1}}^{k(t_{x-1})}\right) d(t_x) \right) \\ &+ a_{r_i(t)} t_0 + b_{r_i(t)} \end{aligned} \tag{22}$$

where the random variables $\theta_{i_x i_{x-1}}^{k(t_{x-1})} \in \Theta$ are independent of each other and $k(t_{x-1})$ is the number of successful times that node i_x receives information from node i_{x-1} before time t_{x-1}.

Proof: The proof is given in Appendix B. ■

Theorem 4.3 provides the relationship of the logical clock between node i and its reference node $r_i(t)$ at time t.

Theorem 4.4: Under Assumptions 2.1 and 3.4, WMTS guarantees that

$$\mathbf{E}\{\hat{a}_i(t)a_i\} = a_{max},\ i \in \mathcal{V}, \tag{23}$$

where $t \geq B(n-1)$. Moreover, the expected value of the logical clock satisfies

$$\mathbf{E}\{L_i(t)\} = a_{max}t + b_{max} - w_i(t)a_{max}\mu,\ i \in \mathcal{V} \tag{24}$$

where $t \geq B(n-1)$.

Proof: The proof is given in Appendix C. ■

Theorem 4.4 indicates that WMTS guarantees the finite-time convergence in expectation for time-variant networks. Note from the item $w_i(t)a_{max}\mu$ in (24) that the synchronization error is affected by the weights of nodes and the expectation of random delay; thus, nodes with different weights will have slightly different expected logical clocks. This is the limitation caused by the random communication delay, as pointed out by Freris *et al.* [35], [36]. Meanwhile, when it takes two-way communication, the offset estimation methods proposed in [23] and [35] can be utilized to alleviate the offset error, for example, in the case where $\sigma = 0$, the delay is known and equal to μ, from [35], the offset estimation is exact (just subtract the known delay in step 5 of WMTS); the same argument holds for $\sigma > 0$, where a similar approach for offset estimation can asymptotically eradicate the offset error. In addition, the variance is crucial for synchronization accuracy. Since $a_{r_i(t)} \leq a_{max}, i \in \mathcal{V}$, it follows from (21) that:

$$a_{\max} \Pi_{x=1}^{w_i(i)} \left(1 + \theta_{i_x i_{x-1}}^{k(t_{x-1})}\right) \geq \hat{a}_i(t)a_i,\ i \in \mathcal{V}.$$

Thus, for time $t \geq B(n-1)$, from (23)

$$\begin{aligned} &\mathbf{Var}\{\hat{a}_i(t)a_i\} = \mathbf{E}\{\hat{a}_i^2(t)a_i^2\} - a_{max}^2 \\ &\leq a_{max}^2 \mathbf{E}\left\{ \prod_{x=1}^{w_i(i)} \left(1 + \theta_{i_x i_{x-1}}^{k(t_{x-1})}\right)^2 - 1 \right\} \\ &\leq a_{max}^2 \left(\prod_{x=1}^{w_i(i)} \left(1 + \frac{1}{k(t_{x-1})} \frac{2\sigma^2}{T^2(1-p)^2}\right) - 1 \right) \end{aligned} \tag{25}$$

where $k(t_{x-1})$ is the communication time from node i_{x-1} to node i_x within the time interval $[0, t_{x-1}]$. From (25), it is clear that the variance of the error of each $\hat{a}_i(t)a_i$ is a decreasing function on the variables $k(t_{x-1}), x = 1, 2, ..., w_i(t)$. Thus, with more communication between nodes i_{x-1} to i_x, the variance of the error of each $\hat{a}_i(t)a_i$ will decrease. Especially, if each variable $k(t_{x-1}) \to \infty$ in (25), the variance of the error of each $\hat{a}_i(t)a_i$ converges to 0. However, Assumptions 2.1 and 3.4 do not guarantee that the upper bound provided in (25) is going to zero for $t \to \infty$. Thus, the WMTS algorithm does not guarantee that synchronization is reached completely. We will further discuss the detailed performance of WMTS in the simulation section.

Although it is really difficult to obtain the analytical expression for the variance of each logical clock in the general time-variant network, we can rigorously prove that for the time-invariant network, WMTS converges in the mean square sense, which is given in the following Theorem.

Theorem 4.5: Consider the time-invariant connected network. With WMTS, for $i, j \in \mathcal{V}$, we have

$$\lim_{t\to\infty} \mathbf{E}\left\{\hat{a}_i(t)a_i - \hat{a}_j(t)a_j\right\} = 0$$

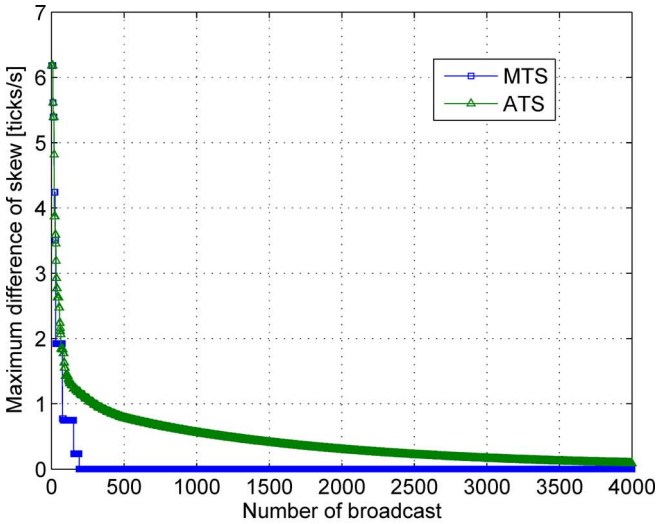
and

$$\lim_{t\to\infty} \mathbf{Var}\left\{\hat{a}_i(t)a_i - \hat{a}_j(t)a_j\right\} = 0.$$

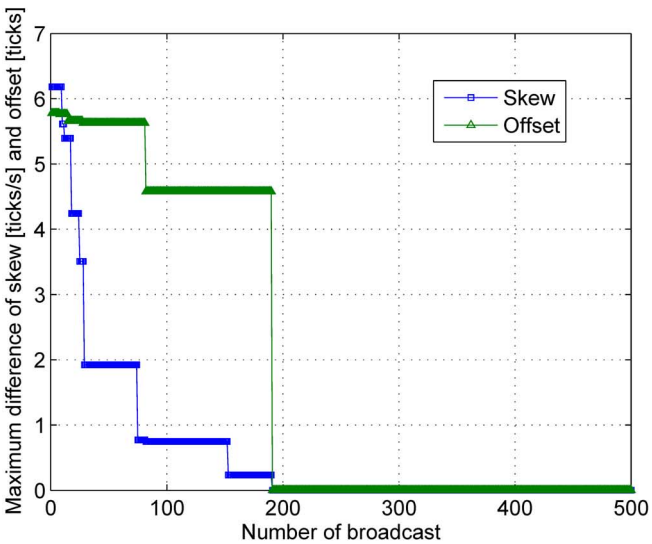
Proof: The proof is given in Appendix D. ■

With Theorem 4.5, it can be observed that under WMTS, the logical clock skew will converge asymptotically in the mean square sense in time-invariant connected networks. If the network is time variant, we can only guarantee that the expected value of each logical clock will converge in finite time, and the synchronization accuracy depends on the mean and the variance of the random delay. Although we cannot analytically provide the logical clock variance under time-variant networks, we conduct extensive large-scale network simulations which demonstrate that WMTS also provides good performance even for time-variant networks.

Remark 4.6: Comparing WMTS and MTS, the basic idea is to drive the logical clocks to the maximum value among all nodes. Meanwhile, besides $\tau_i(t)$, $\hat{a}_i(t)$, and $\hat{b}_i(t)$, that is, hardware clock reading and two parameters of logical clock, which are needed in MTS, the broadcasting message of node i in WMTS will additionally include $w_i(t)$ and $r_i(t)$, that is, weight number



(a)



(b)

Fig. 2. Performance comparison in the static network. (a) MTS and ATS. (b) Skew and offset.

and reference number. In order to implement WMTS, each node will also need to store a_{ij} in addition to the hardware clock reading pair $[\tau_i(t), \tau_j(t)]$. The detailed performance comparison between MTS and WMTS will be provided in the simulation section.

V. Simulation Results

For the simulation examples, we set $\hat{a}(0) = 1, \hat{b}(0) = 0$ and $T = 1$. Since the typical error for a quartz-crystal oscillator is $p \in [10, 100]$ ppm [39], which corresponds to a 10 to 100 microsecond (μs), we assume that each skew a_i of the hardware clock is randomly selected from the set $[0.9999, 1.0001]$, the offset b_i of each node i is randomly selected from the set $[0, 0.0002]$. We also define two functions as follows:

$$d_s(t) = \max_{i,j\in\mathcal{V}} \left(\hat{a}_i(t)a_i - \hat{a}_j(t)a_j\right),$$
$$d_o(t) = \max_{i,j\in\mathcal{V}} \left[\hat{a}_i(t)b_i + \hat{b}_i(t) - (\hat{a}_j(t)b_j + \hat{b}_j(t))\right] \quad (26)$$

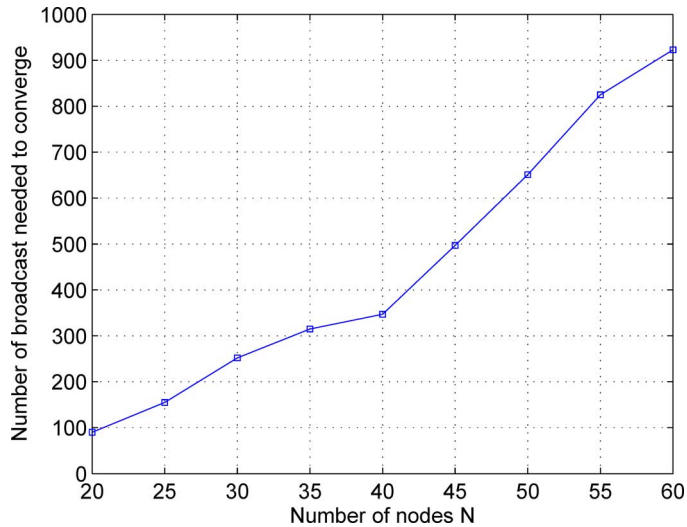


Fig. 3. Convergence time of MTS.

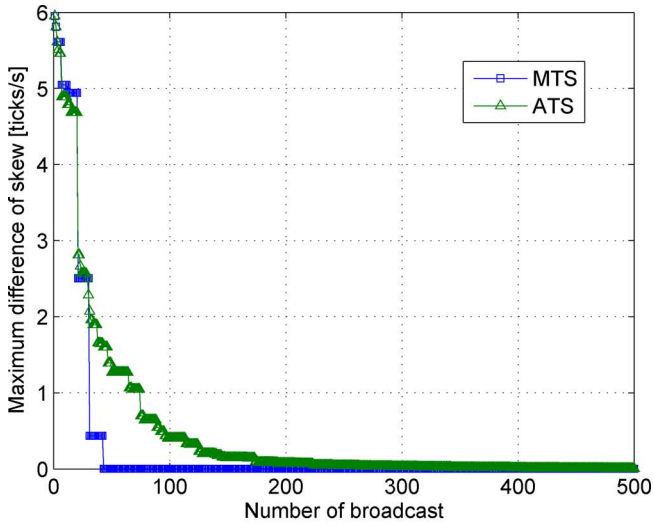
where $d_s(t)$ and $d_o(t)$ denote the maximum difference of the logical skew and of the logical offset between any two nodes, respectively. Clearly, the global time synchronization is reached iff $d_s(t) = 0$ and $d_o(t) = 0$.

A. Delay-Free Case

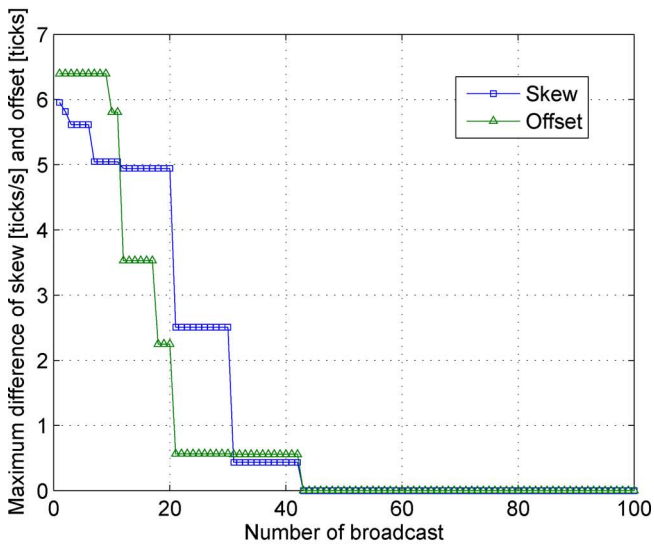
We compare our algorithm MTS with ATS in [14].

Consider a static ring graph with $N = 30$. It is well recognized that distributed consensus-based algorithms often converge slowly for a ring graph. In Fig. 2(a), it can be observed that the skew synchronization is reached after 192 broadcasts (the total number of broadcast of all nodes) under MTS, while it takes more than 4000 broadcast to obtain comparable accuracy under ATS. In addition, from Fig. 2(b), it can be seen that for MTS, the skew and offset synchronization can be achieved simultaneously. Note that in MTS, each iteration denotes one broadcast of nodes. Based on 500 tests, the average iteration convergence time for MTS in this static network is 208, while it is more than 4145 to ensure $d_s(t) \leq 10^{-4}$ ticks per second (where 1 tick $= 1/32768$ Hz $= 30.5$ μs) [14] under ATS. Fig. 3 shows how MTS converges with the number of sensors in the network, which shows the good scalability of MTS since the convergence time is almost linearly increasing with the number of sensors.

Consider a dynamic graph with $N = 50$. The nodes are randomly deployed in an 100×100-m area at initial time 0, and the maximum communication range of each node is 20 m. Assume that each node may randomly change its position once every $20T$. As shown in Fig. 4(a), $d_s(t) \leq 10^{-4}$ ticks per second for $t \geq 381$ by ATS; however, $d_s(t)$ of MTS decreases to zero when the iteration time $t \geq 43$. Thus, the convergence speed of MTS is much faster than that of ATS in a dynamical network. Furthermore, Fig. 4(b) shows that $d_o(t)$ and $d_s(t)$ of MTS do converge to 0 at the same time $t = 43$ in the dynamical network. The average time for MTS is around 47, and is about 545 for ATS to ensure $d_s(t) \leq 10^{-4}$ ticks per second.



(a)



(b)

Fig. 4. Performance comparison in the dynamical network. (a) MTS and ATS. (b) Skew and offset.

B. Random Delay Case

In the following simulations, we will consider the delay in the communication, and compare WMTS with MTS and ATS in [14]. Based on the empirical results in [7] and [34], without loss of generality, we assume that $d \sim N(2.5 \times 10^{-4}, 10^{-8})$ in the simulation, where d is random positive communication delay, which means that the delay is in the range of $[0, 550]$ μs with 99.97% confidence.

Consider a static ring graph with $N = 30$. From Fig. 5, it can be observed that the maximum logical skew of nodes diverges under MTS, which confirms Example 4.1. However, under WMTS, it is clear that the maximum logical skew of nodes converges to a fixed constant. We also compare the performance of MTS, ATS, and WMTS under the static and dynamic network defined in the previous subsection, which are shown in Figs. 6 and 7, respectively. It is observed that WMTS always has faster convergence speed than that of ATS, and with better

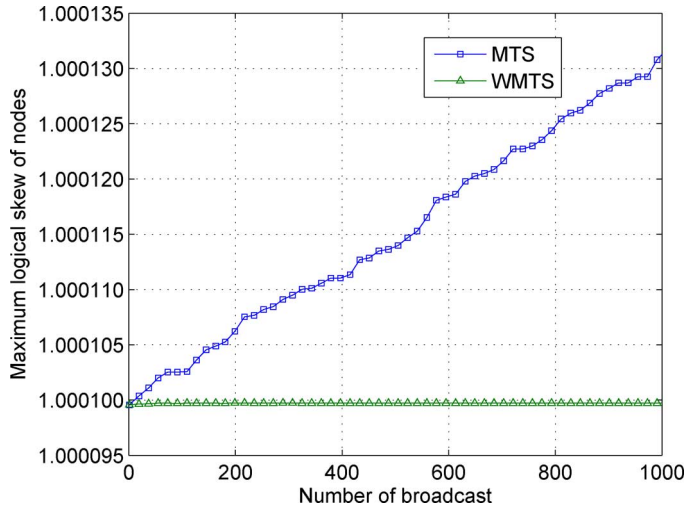
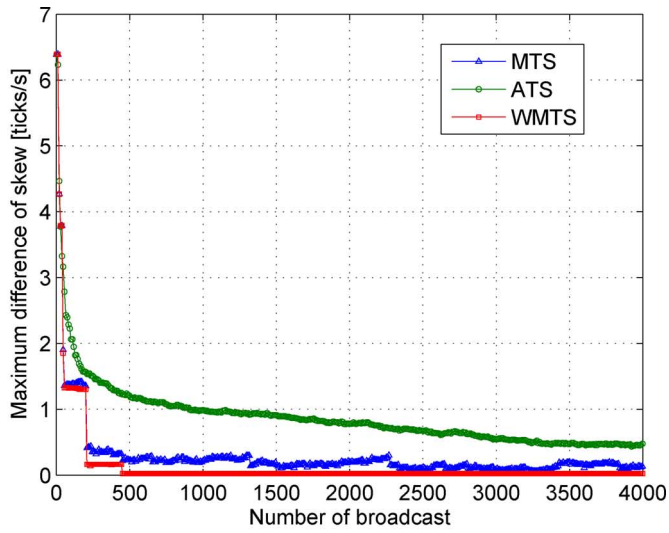
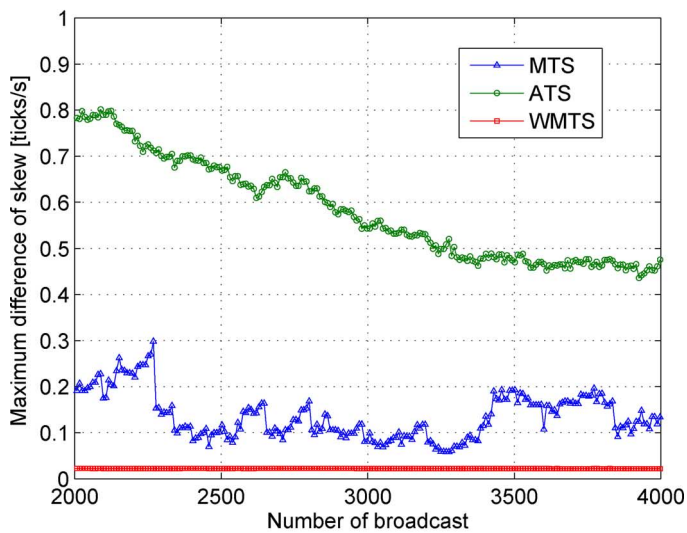


Fig. 5. Maximum skew among nodes changes under MTS and WMTS.



(a)



(b)

Fig. 6. Compare the performance of MTS, ATS, and WMTS under the static network. (a) 0 to 4000. (b) 2000 to 4000.

synchronization accuracy than those of MTS and ATS. From Figs. 6(b) and 7(b), it is clear that WMTS has high accuracy

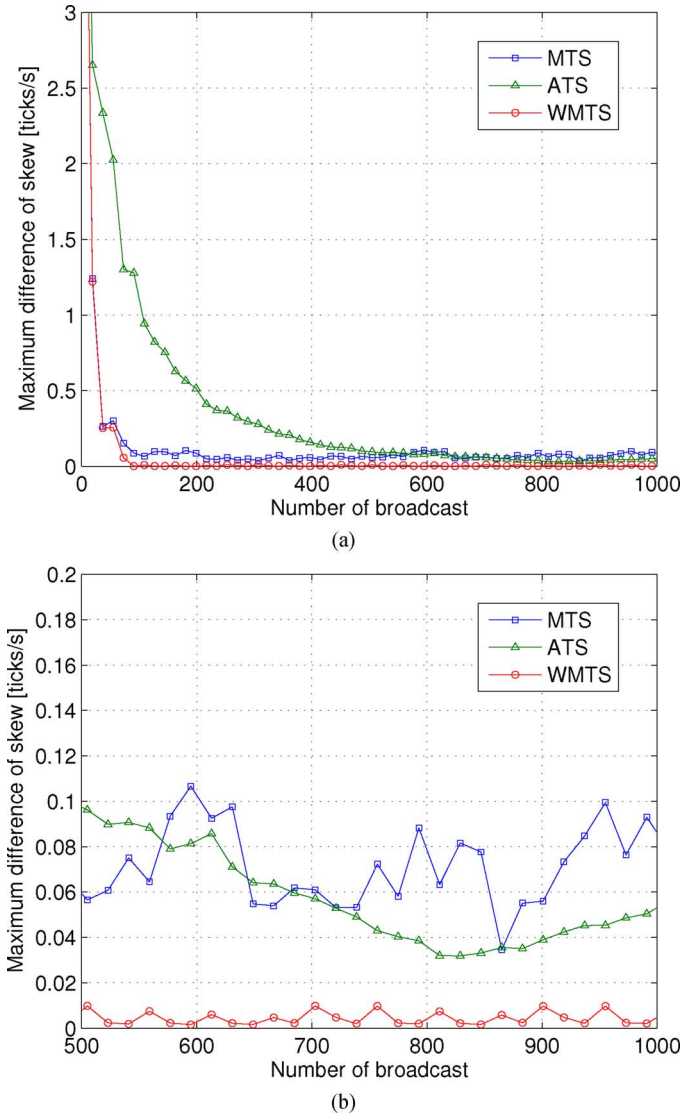


Fig. 7. Compare the performance of MTS, ATS, and WMTS under the dynamic network. (a) 0 to 1000. (b) 500 to 1000.

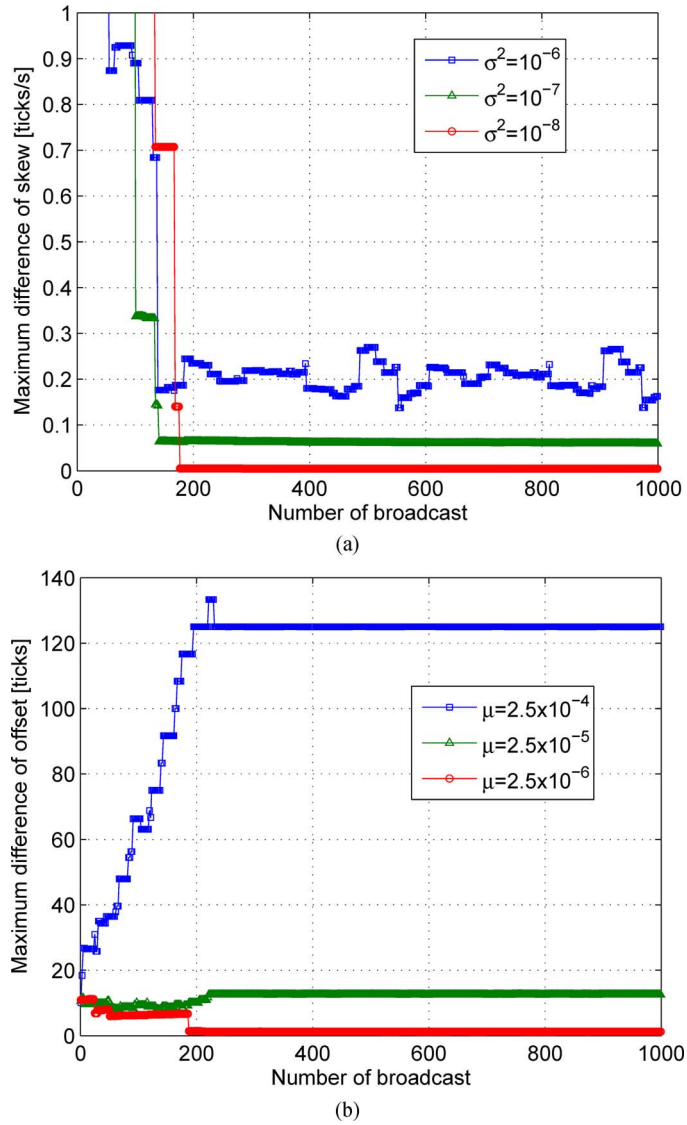
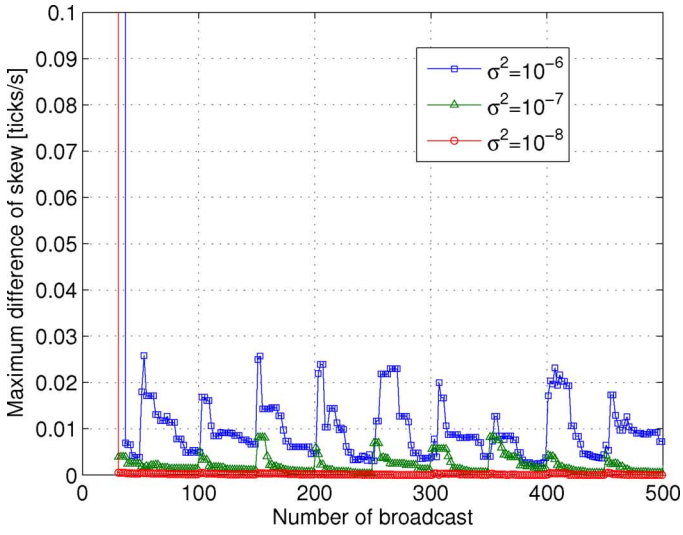


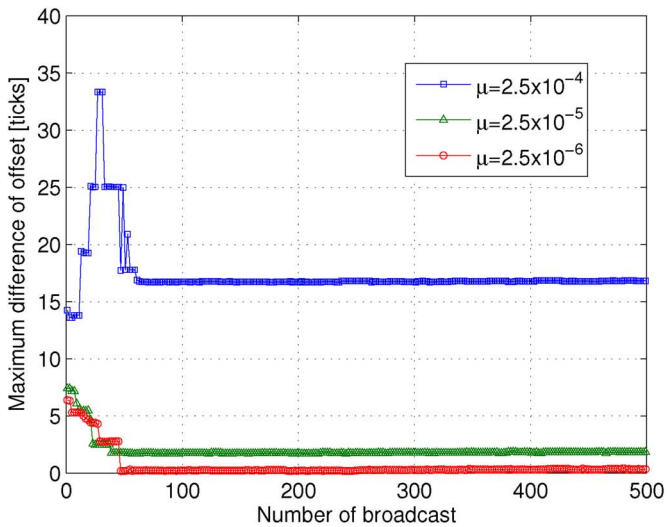
Fig. 8. Performance of WMTS with different mean and variance in the static network: (a) logical skew and (b) logical offset.

as the synchronization accuracy is less than 0.02 ticks/s (about 0.66 μs/s) for the static and dynamic network.

Next, we consider the relationship between synchronization error bound and the mean and variance of the random communication delay under WMTS. The error bound for skew compensation mainly depends on the variance but not on the mean of the delay, as the skew compensation depends on the estimate of relative skew [by (14)] which is only affected by the variance σ^2 (seen from Lemma 4.2). However, it follows from (24) that the error bound for the offset compensation is mainly affected by the mean μ. As shown in Fig. 8, the error bound will increase with the variance for skew compensation [see Fig. 8(a)] while with mean for offset compensation [see Fig. 8(b)]. Fortunately, for skew compensation, it follows from Fig. 8(a) that the error bounds are always less than 0.3 ticks/s (about 10 μs/s) for $\sigma^2 \leq 10^{-6}$ (which means that there is a delay in the range of $\mu \pm 3\sigma = \mu \pm 3$ ms with 99.97% confidence), that is, WMTS guarantees that the error bound is less than 10 μs even if the communication delay is in the order of microseconds. Note that skew compensation is more important since it can extend the re-synchronization period to save energy; thus, WMTS is able to well handle the random delay case. Especially, if $(\mu, \sigma^2) = (2.5 \times 10^{-6}, 10^{-12})$ (the delay in 2.5 ± 3 μs with a 99.97% confidence), WMTS will guarantee that the error bounds for skew compensation and offset compensation will be less than 10^{-5} μs and 5 μs, respectively, which means that with WMTS, the maximum logical clock difference between nodes will be less than 10 μs in the time interval $[0, 10^5]$ s after convergence of the algorithm. Finally, the performance of WMTS with different means and variances in a dynamic network is shown as in Fig. 9, where the relationship between synchronization error bound and the values of mean and variance are the same as those in the static network. Moreover, by comparing Figs. 9 with 8, it is observed that for both skew and offset compensation, the synchronization accuracy of WMTS in the dynamic network is improved with even faster converging speed. The reason is that the random dynamic network may help to improve network connectivity which accelerates the consensus process.



(a)



(b)

Fig. 9. Performance of WMTS with different mean and variance in the dynamic network: (a) logical skew and (b) logical offset.

VI. Conclusion

We investigate the time synchronization for WSNs in this paper. We present two new time synchronization algorithms—MTS and WMTS protocols for WSNs with *delay free* and *random delay* cases, respectively. These two algorithms conduct the skew and offset compensations simultaneously. The main idea is to drive all clocks to the maximum value among the network. It is proved that the MTS algorithm converges within a finite time and the expected convergence time of WMTS is also finite. Both algorithms are fully distributed, asynchronous, and robust to dynamic network topologies. Extensive simulations demonstrate the effectiveness of the proposed algorithms. Future directions include extending the idea to more complicated network models and experimental validation of the results.

Appendix A
Proof of Theorem 3.3

Proof: According to the definitions of $\tau_v(t)$ in (8) and $V(t)$ in (9), it is obvious that once $V(t)=0$, the skew and offset of all logical clocks are equal to $a_{\max}$ and $b_{\max}$ accordingly. Hence, it suffices to prove $\lim_{k\to\infty} V(k)=0$.

We first prove that $V(t)$ is nonincreasing. For any arbitrary node $i \in \mathcal{V}_v(t)$, its skew $\hat{a}_i(t)a_i$ and offset $\hat{a}_i(t)b_i+\hat{b}_i(t)$ are equal to a_{max} and b_{max}, respectively. Hence, $\hat{a}_i(t)a_i \geq \hat{a}_j(t)a_j$ holds for $\forall j \in \mathcal{V}$, which means $\frac{a_{ij}\hat{a}_j}{\hat{a}_i} \leq 1$ holds for $\forall j \in \mathcal{V}$. In other words, this node i will not update its logical clock skew any more as $q_{ij} \leq 1$ for $k \geq t$. Furthermore, $\hat{a}_i(t)b_i+\hat{b}_i(t)=b_{max}$ ensures that node i has the largest logical clock offset from the definition of $\mathcal{V}_v(t)$, which is equivalent to $L_i(t)=\tau_v(t), \forall i \in \mathcal{V}_v(t)$. Therefore, each node i in $\mathcal{V}_v(t)$ will keep its logical clock skew and offset, which means that $N_v(t)$ is nondecreasing which, in turn, implies that $V(t)$ is nonincreasing.

Assume $V(t)>0$ at time t (where $t=mB, m\in\{0,\mathcal{N}^+\}$). Then, there is at least one node in the network which is not in the set $\mathcal{V}_v(t)$. Assumption 2.1 guarantees that the network is connected during any time interval B, which means that there are edge sequences $e_{ij}(t_k), k\in\mathcal{N}^+$ which connect nodes i ($i\in\mathcal{V}_v(t)$) and j ($j\in\mathcal{V}-\mathcal{V}_v(t)$). Hence, k_0 exists such that node i in $\mathcal{V}-\mathcal{V}_v(t)$ receives packets from node j in $\mathcal{V}_v(t)$ at time $t+k_0B$ for the second time by two communication links existing between nodes i and j, where the two edges are in $e_{ij}(t_k), k\in\mathbf{N}^+$. According to MTS, node i will update its logical clock and then nodes i and j will have the same logical clock, which means that node i belongs to $\mathcal{V}_v(t+k_0B)$. Thus, $N_v(t+k_0B)=N_v(t)+1$, that is, $V(t+k_0B)$ satisfies $V(t+k_0B)=V(t)-1$. Hence, $V(t)$ will decrease strictly if $V(t)>0$. Notice that $V(0)\leq N-1$ is finite; therefore, $\lim_{k\to\infty}V(k)=0$. ■

Appendix B
Proof of Theorem 4.3

Proof: For WMTS, if node i has $w_i(t)=0$ and $r_i(t)=i$ at time t, the reference node of node i is itself, then node i has not adjusted its logical clock, that is, $\hat{a}_i(t)a_i=a_i$ and $L_i(t)=\tau_i(t)$. If node i has $w_i(t)>0$ and $r_i(t)\neq i$ at time t, node i makes the latest change of its logical clock, reference number, and weight when it communicates with a node j which has $r_j(t')=r_i(t)$ and $w_j(t')=w_i(t)-1$ at an earlier time $t'<t$. Thus, we have

$$\hat{a}_i(t)a_i=\hat{a}_i(t')a_i=\hat{a}_j(t')a_j\left(1+\theta_{ij}^{k(t')}\right) \tag{27}$$

where $\theta_{ij}^{k(t')}\in\Theta$ and $k(t')$ are equal to the number of successful times that node i received information from node j within $[0,t']$, and

$$\begin{aligned} L_i(t) &= L_i(t')+(t-t')\hat{a}_i(t')a_i \\ &= L_j(t')-\hat{a}_j(t')a_jd(t')+(t-t')\hat{a}_i(t')a_i \end{aligned} \tag{28}$$

where $d(t')$ is the random communication delay from node j to node i of the communication at time t'. For node j, similar to the previous discussion, one can obtain

$$\hat{a}_j(t')a_j=\hat{a}_j(t'')a_j=\hat{a}_l(t'')a_l\left(1+\theta_{jl}^{k(t'')}\right) \tag{29}$$

where $\theta_{jl}^{k(t'')} \in \Theta$ and $k_{jl}(t'')$ are equal to the number of successful times that node j receives information from node l within $[0, t'']$, and

$$\begin{aligned} L_j(t') &= L_j(t'') + (t' - t'')\hat{a}_j(t')a_j \\ &= L_l(t'') - \hat{a}_l(t'')a_l d(t'') + (t' - t'')\hat{a}_j(t')a_j \end{aligned} \quad (30)$$

where node l has $r_l(t'') = r_i(t)$ and $w_l(t'') = w_i(t) - 2$, and $t'' \in [0, t']$.

Substituting (29) and (30) into (27) and (28), respectively, yields

$$\hat{a}_i(t)a_i = \hat{a}_l(t'')a_l \left(1 + \theta_{jl}^{k(t'')}\right)\left(1 + \theta_{ij}^{k(t')}\right) \quad (31)$$

and

$$\begin{aligned} L_i(t) = L_l(t'') &- \hat{a}_l(t'')a_l \left[d(t'') + \left(1 + \theta_{jl}^{k(t'')}\right) d(t')\right] \\ &+ \hat{a}_l(t'')a_l(t' - t'')\left(1 + \theta_{jl}^{k(t'')}\right) \\ &+ \hat{a}_l(t'')a_l(t - t')\left(1 + \theta_{jl}^{k(t'')}\right)\left(1 + \theta_{ij}^{k(t')}\right), \end{aligned} \quad (32)$$

where $\theta_{ij}^{k(t')}, \theta_{jl}^{k(t'')} \in \Theta$. By repeating this procedure, until the weight of a node is equal to 0, one obtains (21) and (22) and the node sequence. Since the nodes in such a node sequence are different, the random variables in (21) and (22) are independent of each other. ■

APPENDIX C
PROOF OF THEOREM 4.4

Let $\theta_{r_i(t)}^{k_l(t)} \in \Theta, i, l = 1, 2, ..., N$ be series random variables which are independent of each other. Use $u, v = i, j$ to denote that $u = i, j$ and $v = i, j$.

Lemma C.1: Suppose that two neighbor nodes i and j have two successful communications within the time interval $[t, t^+]$. By WMTS, we have

$$\begin{aligned} &a_{r_u(t^+)} \prod_{x=1}^{w_u(t^+)+1} \left(1 + \theta_{r_u(t^+)}^{k_x(t^+)}\right) \\ &\geq a_{r_v(t)} \prod_{y=1}^{w_v(t)+1} \left(1 + \theta_{r_v(t)}^{k_y(t)}\right), u, v = i, j \end{aligned} \quad (33)$$

where t^+ is the finishing time of the two successful communications.

Proof: If $r_i(t) = r_j(t)$, (33) is obtained directly from step 5 and Theorem 4.3.

If $r_i(t) \neq r_j(t)$, from steps 5 and 6 of the WMTS algorithm, three cases may appear for nodes i and j after two successful communications between them.

Case 1: The reference numbers of them are updated as $r_i(t^+) = r_j(t^+) = r_i(t)$. In this case, $q_j = \frac{a_{ji}(k)\hat{a}_i}{\hat{a}_j} = \frac{\hat{a}_i a_i (1+\theta_{ji}^k)}{\hat{a}_j a_j} > 1$, where k is the communication times from nodes i to j within $[0, t]$. Hence

$$\hat{a}_i(t)a_i \left(1 + \theta_{ji}^k\right) > \hat{a}_j(t)a_j. \quad (34)$$

With WMTS, $\hat{a}_i(t^+)$ and $\hat{a}_i(t^+)$ satisfy

$$\hat{a}_i(t^+)a_i = \hat{a}_i(t)a_i,\ \hat{a}_j(t^+)a_j = \hat{a}_i(t)a_i \left(1 + \theta_{ji}^k\right) \quad (35)$$

and $w_i(t^+) = w_i(t)$ and $w_j(t^+) = w_i(t) + 1$. Combining (34) with (35) yields

$$\hat{a}_i(t^+)a_i \geq \hat{a}_i(t)a_i,\ \hat{a}_j(t^+)a_j \geq \hat{a}_j(t)a_j \quad (36)$$

and

$$\hat{a}_i(t^+)a_i \left(1 + \theta_{ji}^k\right) \geq \hat{a}_j(t)a_j,\ \hat{a}_j(t^+)a_j \geq \hat{a}_i(t)a_i \left(1 + \theta_{ij}^k\right). \quad (37)$$

From the above inequalities and using Theorem 4.3, one can conclude that (33) holds.

Case 2: The reference numbers of them are updated as $r_i(t^+) = r_j(t^+) = r_j(t)$. The remaining discussion is similar to that of Case 1.

Case 3: The reference numbers of them are updated as $r_i(t^+) = r_i(t)$ and $r_j(t^+) = r_j(t)$. In this case, $q_i \leq 1$ and $q_j \leq 1$ during the communication between nodes i and j. Hence

$$\hat{a}_i(t)a_i \left(1 + \theta_{ji}^k\right) \leq \hat{a}_j(t)a_j,\ \hat{a}_j(t)a_j \left(1 + \theta_{ij}^k\right) \leq \hat{a}_i(t)a_i. \quad (38)$$

With WMTS, $\hat{a}_i(t^+)$ and $\hat{a}_i(t^+)$ satisfy

$$\hat{a}_i(t^+)a_i = \hat{a}_i(t)a_i,\ \hat{a}_j(t^+)a_j = \hat{a}_j(t)a_j \quad (39)$$

and $w_i(t^+) = w_i(t)$ and $w_j(t^+) = w_j(t)$. Thus, with (38) and (39) and Theorem 4.3, the Lemma holds. ■

Taking expectation on both sides of (33), we have $\mathbf{E}\{a_{r_u(t^+)}\} \geq \mathbf{E}\{a_{r_v(t)}\}$, that is, after two successful communications between two nodes, the node with faster hardware clock speed will be selected as the source reference clock.

Lemma C.2: Under Assumptions 2.1 and 3.4, WMTS guarantees that

$$\mathbf{E}\{a_{r_i(t)}\} = a_{max},\ i \in \mathcal{V} \quad (40)$$

where $t \geq B(n-1)$.

Proof: Divide the time interval $[0, t]$ into $(n-1)$ small time intervals, which are given by $[0, B]$, $[B, 2B]$, ..., $[(N-2)B, (N-1)B]$. According to Assumption 2.1, the network is connected at each small time interval.

Assume that node i_1 with its hardware clock is equal to (8), that is, $a_{i_1} = a_{\max}$. Since the network is connected within each small time interval, there is at least one node, say i_2, which will be a neighbor of node i_1 within $[0, B]$. Assumption 3.4 ensures that node i_1 and i_2 have at least two successful communications within $[0, B]$. After two communications, from Lemma C.1

$$\begin{aligned} &a_{r_u(t_1)} \prod_{l=1}^{w_u(t_1)+1} \left(1 + \theta_{r_u(t_1)}^{k_l(t_1)}\right) \\ &\geq a_{r_v(t_0)} \prod_{l=1}^{w_v(t_0)+1} \left(1 + \theta_{r_v(t_0)}^{k_l(t_0)}\right),\ u, v = i_1, i_2 \end{aligned} \quad (41)$$

where $t_1, t_1 \in [0, B]$ is the finishing time of two successful communications between them. Note that $a_{r_{i_1}(t_0)} = a_{\max}$. Combining (41) with Theorem 4.3 yields that

$$\begin{aligned} &a_{r_u(t_1)} \prod_{l=1}^{n_{t_1}} \left(1 + \theta_{r_u(t_1)}^{k_l(t_1)}\right) \\ &\geq a_{max} \prod_{l=1}^{n_{t_1}} \left(1 + \theta_{r_{i_1}(t_0)}^{k_l(t_0)}\right),\ u \in \mathcal{V}_r(t_1) \end{aligned} \quad (42)$$

where n_{t_1} is the number of nodes in $\mathcal{V}_r(t_1)$ which is the node set satisfying $\mathcal{V}_r(t_1) = \{i | r_i = r_{i_1}(t_1)\ or\ r_i = r_{i_2}(t_1), i \in \mathcal{V}\}$.

In the time interval $[t_1, 2B]$, the network is also connected, and there is at least one node, say i_3, $i_3 \neq i_1, i_2$ in the network, which will be a neighbor node of i_1 or i_2 (assume it is node i_2 here). From Lemma 1.1, after two successful communications between nodes i_2 and i_3

$$a_{r_u(t_2)}\prod_{l=1}^{w_u(t_2)+1}\left(1+\theta_{r_u(t_2)}^{k_l(t_2)}\right) \geq a_{r_v(t_1)}\prod_{l=1}^{w_v(t_1)+1}\left(1+\theta_{r_v(t_1)}^{k_l(t_1)}\right),\ u,v=i_2,i_3 \quad (43)$$

where $t_2 \in [t_1, 2B]$ is the finishing time of two successful communications between them. We have $\{r_{i_2}(t_2), r_{i_3}(t_2)\} \subseteq \{r_{i_2}(t_1), r_{i_3}(t_1)\}$. Note that node i_1 may contact with i_2 only within $[t_1, t_2]$, that is, $r_{i_1}(t_2) \in \{r_{i_1}(t_1), r_{i_2}(t_1)\}$. Hence

$$\{r_{i_1}(t_2), r_{i_2}(t_2), r_{i_3}(t_2)\} \subseteq \{r_{i_1}(t_1), r_{i_2}(t_1), r_{i_2}(t_2), r_{i_3}(t_2)\}.$$

Combining (43) and (42) with Theorem 4.3, one can obtain

$$a_{r_u(t_2)}\prod_{l=1}^{n_{t_2}}\left(1+\theta_{r_u(t_2)}^{k_l(t_2)}\right) \geq a_{max}\prod_{l=1}^{n_{t_2}}\left(1+\theta_{r_{i_1}(t_0)}^{k_l(t_0)}\right),\ u\in\mathcal{V}_r(t_2) \quad (44)$$

where n_{t_2} is the number of nodes in $\mathcal{V}_r(t_2)$ which is defined as $\mathcal{V}_r(t_2)=\{i|r_i = r_{i_1}(t_1)$ *or* $r_i=r_{i_2}(t_2)$ *or* $r_i = r_{i_3}(t_2), i\in\mathcal{V}\}$.

By induction, assume that

$$\{r_{i_1}(t_{k-1}), r_{i_2}(t_{k-1}), ..., r_{i_k}(t_{k-1})\} \subseteq \{r_{i_1}(t_1), r_{i_2}(t_1), r_{i_2}(t_2), r_{i_3}(t_2), ..., r_{i_{k-1}}(t_{k-1}), r_{i_k}(t_{k-1})\} \quad (45)$$

and

$$a_{r_u(t_{k-1})}\prod_{l=1}^{n_{t_{k-1}}}\left(1+\theta_{r_u(t_{k-1})}^{k_l(t_{k-1})}\right) \geq a_{max}\prod_{l=1}^{n_{t_{k-1}}}\left(1+\theta_{r_{i_1}(t_0)}^{k_l(t_0)}\right), u\in\mathcal{V}_r(t_{k-1}) \quad (46)$$

holds for $N-1 \geq k \geq 2$, where $n_{t_{k-1}}$ is the number of nodes in $\mathcal{V}_r(t_{k-1})$ which satisfies

$$\mathcal{V}_r(t_{k-1}) = \{i|r_i = r_{i_1}(t_{k-1})\ or\ ...\ or\ r_i = r_{i_k}(t_{k-1}), i\in\mathcal{V}\}, t_{k-1}\in[t_{k-2}, (k-1)B].$$

Then, within $[t_{k-1}, kB]$, there is a node $i_{k+1}, i_{k+1} \neq i_1, i_2, ..., i_k,$, which becomes a neighbor node of one node in $i_1, i_2, ..., i_k$ (assume it is i_k). Similarly, from Lemma 1.1

$$a_{r_u(t_k)}\prod_{l=1}^{w_u(t_k)+1}\left(1+\theta_{r_u(t_k)}^{k_l(t_k)}\right) \geq a_{r_v(t_{k-1})}\prod_{l=1}^{w_v(t_{k-1})+1}\left(1+\theta_{r_v(t_{k-1})}^{k_l(t_{k-1})}\right),\ u,v=i_k,i_{k+1} \quad (47)$$

where $t_k \in [t_{k-1}, kB]$ is the finishing time of two successful communications between i_k and i_{k+1}. We have $\{r_{i_k}(t_k), r_{i_{k+1}}(t_k)\} \subseteq \{r_{i_k}(t_{k-1}), r_{i_{k+1}}(t_k)\}$. Note that nodes $i_1, i_2, ..., i_{k-1}$ may contact with $i_1, i_2, ..., i_k$ only in $[t_{k-1}, t_k]$, which guarantees that

$$\{r_{i_1}(t_k), r_{i_2}(t_k), ..., r_{i_{k-1}}(t_k)\} \subseteq \{r_{i_1}(t_{k-1}), r_{i_2}(t_{k-1}), ..., r_{i_k}(t_{k-1})\}.$$

From (45), we have

$$\{r_{i_1}(t_k), r_{i_2}(t_k), ..., r_{i_{k+1}}(t_k)\} \subseteq \{r_{i_1}(t_1), r_{i_2}(t_1), r_{i_2}(t_2), r_{i_3}(t_2), ..., r_{i_k}(t_k), r_{i_{k+1}}(t_k)\}. \quad (48)$$

Combining (47) and (46) with Theorem 4.3 gives

$$a_{r_u(t_k)}\prod_{l=1}^{n_{t_k}}\left(1+\theta_{r_u(t_k)}^{k_l(t_k)}\right) \geq a_{max}\prod_{l=1}^{n_{t_k}}\left(1+\theta_{r_{i_1}(t_0)}^{k_l(t_0)}\right),\ u\in\mathcal{V}_r(t_k) \quad (49)$$

where n_{t_k} is the number of nodes in $\mathcal{V}_r(t_k)$ which satisfies

$$\mathcal{V}_r(t_k) = \{i|r_i = r_{i_1}(t_k)\ or\ r_i = r_{i_2}(t_k)\ or\ ...\ or\ r_i = r_{i_{k+1}}(t_k),\ i\in\mathcal{V}\}.$$

From the aforementioned discussion, one concludes that (48) and (49) hold for $k = 1, 2, ..., N-1$. It follows that all nodes belong to $\mathcal{V}_r(t_{N-1})$ and satisfy (49).

Since $t_{N-1} \in [t_{N-2}, (N-1)B]$, we have

$$\{r_i(t), i = 1, 2, ..., N\} \subseteq \{r_{i_l}(t_{N-1}), l = 1, 2, ..., N\}.$$

Therefore, we obtain

$$a_{r_u(t)}\prod_{l=1}^{N}\left(1+\theta_{r_u(t)}^{k_l(t)}\right) \geq a_{max}\prod_{l=1}^{N}\left(1+\theta_{r_{i_1}(t_0)}^{k_l(t_0)}\right) u\in\mathcal{V}. \quad (50)$$

Taking expectation on both sides of (50) yields $\mathbf{E}\{a_{r_i(t)}\} \geq a_{max},\ i\in\mathcal{V}$. At the same time, note that $a_{r_i(t)} \leq a_{max}$ holds for $\forall i \in \mathcal{V}$. Thus, (40) holds. ■

From Lemma C.2, the expected value of the reference node is equal to that of the node whose hardware clock skew is $a_{\max}$. Note that the clock oscillators for sensor nodes are slightly different, which means that the hardware clock skew of nodes is also slightly different. Hence, by combining Lemma 1.2 and Theorem 4.3, we can obtain Theorem 4.4.

APPENDIX D
PROOF OF THEOREM 4.5

Proof: Since the network is time-invariant and connected, each $k(t)$ of the random variable $\theta_{ij}^{k(t)}$ in Theorem 4.3 satisfies $k(t) \to \infty$ when $t \to \infty$. From (20)

$$\lim_{t\to\infty}\mathbf{Var}\{\theta_{ij}^{k_{ij}(t)}\} = \lim_{k_{ij}(t)\to\infty}\mathbf{Var}\{\theta_{ij}^{k_{ij}(t)}\} = 0. \quad (51)$$

Hence, from Theorem 4.3, we have

$$\lim_{t\to\infty}\mathbf{E}\left\{\hat{a}_i(t)a_i - a_{r_i(t)}\right\} = 0$$

and

$$\lim_{t\to\infty}\mathbf{Var}\left\{\hat{a}_i(t)a_i - a_{r_i(t)}\right\} = 0, i\in\mathcal{V}. \quad (52)$$

Furthermore, for any neighboring nodes i and j, from Lemma 1.1, $\lim_{t\to\infty}\mathbf{E}\left\{a_{r_i(t)} - a_{r_j(t)}\right\} = 0$. Since the network is connected, it follows that (52) holds for $\forall i, j \in \mathcal{V}$. Therefore, for $i, j \in \mathcal{V}$, it follows from (52) that $\lim_{t\to\infty}\mathbf{E}\{\hat{a}_i(t)a_i - \hat{a}_j(t)a_j\} = 0$ and

$$\lim_{t\to\infty}\mathbf{Var}\{\hat{a}_i(t)a_i - \hat{a}_j(t)a_j\} = 0.$$

■

REFERENCES

[1] J. He, P. Cheng, L. Shi, and J. Chen, "Time synchronization in WSNs: A maximum value based consensus approach," in *Proc. CDC*, Dec. 12–15, 2011, pp. 7882–7887.

[2] B. Sundararaman, U. Buy, and A. D. Kshemkalyani, "Clock synchronization for wireless sensor networks: A survey," *Ad Hoc Netw.*, vol. 3, no. 3, pp. 281–323, 2005.

[3] J. Du and W. Shi, "APP-MAC: An application-aware event-oriented MAC protocol for multimodality wireless sensor networks," *IEEE Trans. Vehicular Technol.*, vol. 57, no. 6, pp. 3723–3731, Nov. 2008.

[4] Q. Li and D. Rus, "Global clock synchronization in sensor networks," presented at the Infocom, Hong Kong, China, Mar. 2004.

[5] J. Elson, L. Girod, and D. Estrin, "Fine-grained network time synchronization using reference broadcasts," presented at the Operating Syst. Design Implement., Boston, MA, USA, Dec. 9–11, 2002.

[6] S. Ganeriwal, R. Kumar, and M. B. Srivastava, "Timing-sync protocol for sensor networks," in *Proc.SenSys*, 2003, pp. 138–149.

[7] M. Maroti, B. Kusy, G. Simon, and A. Ledeczi, "The flooding time synchronization protocol," in *Proc. ACM SenSys*, 2004, pp. 39–49.

[8] Y. R. Faizulkhakov, "Time synchronization methods for wireless sensor networks: A survey," *Programm. Comput. Softw.*, vol. 33, no. 4, pp. 214–226, 2007.

[9] R. Solis, V. S. Borkar, and P. R. Kumar, "A new distributed time synchronization protocol for multihop wireless networks," in *Proc. IEEE CDC*, San Diego, CA, USA, Dec. 2006, pp. 2734–2739.

[10] A. Giridhar and P. R. Kumar, "Distributed clock synchronization over wireless networks: Algorithms and analysis," in *Proc. IEEE CDC*, 2006, pp. 4915–4920.

[11] M. Akar and R. Shorten, "Distributed probabilistic synchronization algorithm for communication networks," *IEEE Trans. Autom. Control*, vol. 53, no. 1, pp. 389–394, Feb. 2008.

[12] N. Marechal, J. Pierrot, and J. Gorce, "Fine synchronization for wireless sensor networks using gossip averaging algorithms," in *Proc. IEEE ICC*, 2008, pp. 4963–4967.

[13] C. Liao and P. Barooah, "Time-synchronization in mobile sensor networks from difference measurements," in *Proc. CDC*, 2010, pp. 2118–2123.

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

[15] S. Philipp and W. Roger, "Gradient clock synchronization in wireless sensor networks," in *Proc. IPSN*, 2009, pp. 37–48.

[16] R. Olfati-Saber, J. A. Fax, and R. M. Murray, "Consensus and cooperation in networked multi-agent systems," *Proc. IEEE*, vol. 95, no. 1, pp. 215–233, Jan. 2007.

[17] A. Nedic, A. Olshevsky, A. Ozdaglar, and J. N. Tsitsiklis, "On distributed averaging algorithms and quantization effects," *IEEE Trans. Autom. Control*, vol. 54, no. 11, pp. 2506–2517, Nov. 2009.

[18] S. Boyd, A. Ghosh, B. Prabhakar, and D. Shah, "Randomized gossip algorithms," *IEEE Trans. Inf. Theory*, vol. 52, no. 6, pp. 2508–2530, Jun. 2006.

[19] G. Xiong and S. Kishore, "Analysis of distributed consensus time synchronization with Gaussian delay over wireless sensor networks," *EURASIP J. Wireless Commun. Netw.*, 2009.

[20] Q. Chaudhari, E. Serpedin, and K. Qaraqe, "Some improved and generalized estimation schemes for clock synchronization of listening nodes in wireless sensor networks," *IEEE Trans. Commun.*, vol. 58, no. 1, pp. 63–67, Jan. 2010.

[21] J. Elson, R. M. Karp, C. H. Papadimitriou, and S. Shenker, "Global synchronization in sensornets," in *Proc. Latin Amer. Symp.*, 2004, pp. 609–624.

[22] P. Barooah and J. P. Hespanha, "Distributed estimation from relative measurements in sensor networks," in *Proc. ICISIP*, 2005, pp. 226–231.

[23] N. M. Freris, V. S. Borkar, and P. R. Kumar, "A model-based approach to clock synchronization," in *Proc. CDC*, 2009, pp. 5744–5749.

[24] D. Fontanelli and D. Macii, "Towards master-less WSN clock synchronization with a light communication protocol," in *Proc. IMTC*, 2010, pp. 105–110.

[25] L. Mei and Y. C. Wu, "On clock synchronization algorithms for wireless sensor networks under unknown delay," *IEEE Trans. Veh. Technol.*, vol. 59, no. 1, pp. 182–190, Jan. 2010.

[26] L. Mei and Y.-C. Wu, "Distributed clock synchronization for wireless sensor networks using belief propagation," *IEEE Trans. Signal Process.*, vol. 59, no. 11, pp. 5404–5414, Nov. 2011.

[27] R. Carli, A. Chiuso, S. Zampieri, and L. Schenato, "A PI consensus controller for networked clocks synchronization," in *Proc. 17th IFAC World Congr.*, 2008, pp. 10289–10294.

[28] R. Carli and S. Zampieri, "Networked clock synchronization based on second order linear consensus algorithms," in *Proc. CDC*, 2010, pp. 7259–7264.

[29] R. Carli, E. Elia, and S. Zampieri, "A PI controller based on asymmetric gossip communications for clocks synchronization in wireless sensors networks," in *Proc. CDC-ECC*, 2011, pp. 7512–7517.

[30] R. Carli, A. Chiuso, L. Schenato, and S. Zampieri, "Optimal synchronization for networks of noisy double integrators," *IEEE Trans. Autom. Control*, vol. 56, no. 5, pp. 1146–1152, May 2011.

[31] W. Ren, "On consensus algorithms for double-integrator dynamics," *IEEE Trans. Autom. Control*, vol. 53, no. 6, pp. 1503–1509, Jul. 2008.

[32] R. Olfati-Saber and R. M. Murray, "Consensus protocols for networks of dynamic agents," in *Proc. ACC*, 2003, pp. 951–956.

[33] H. Tijms, *Understanding Probability: Chance Rules in Everyday Life*. Cambridge, U.K.: Cambridge Univ. Press, 2004.

[34] Z. Zhong, P. P. Chen, and T. He, "On-demand time synchronization with predictable accuracy," in *Proc. IEEE Infocom*, 2009, pp. 2480–2488.

[35] N. M. Freris, S. R. Graham, and P. R. Kumar, "Fundamental limits on synchronizing clocks over networks," *IEEE Trans. Autom. Control*, vol. 56, no. 6, pp. 1352–1364, Jun. 2011.

[36] N. M. Freris, H. Kowshik, and P. R. Kumar, "Fundamentals of large sensor networks: Connectivity, capacity, clocks, and computation," *Proc. IEEE*, vol. 98, no. 11, pp. 1828–1846, Nov. 2010.

[37] J. R. Birge and F. Louveaux, *Introduction to Stochastic Programming*. New York, USA: Springer, 1997.

[38] D. Zhou and T. H. Lai, "An accurate and scalable clock synchronization protocol for IEEE 802.11-based multihop Ad Hoc networks," *IEEE Trans. Parallel Distrib. Syst.*, vol. 18, no. 12, pp. 1797–1808, Dec. 2007.

[39] B. Choi, H. Liang, X. Shen, and W. Zhuang, "DCS: Distributed asynchronous clock synchronization in delay tolerant networks," *IEEE Trans. Parallel Distrib. Syst.*, vol. 23, no. 3, pp. 491–504, Mar. 2012.

[40] S. Ganeriwal, C. Popper, S. Capkun, and M. B. Srivastava, "Secure time synchronization in sensor networks," *ACM Trans. Inf. Syst. Security*, vol. 11, no. 4, 2008.

[41] J. Chiang, J. Haas, Y.-C. Hu, P. R. Kumar, and J. Choi, "Fundamental limits on secure clock synchronization and man-in-the-middle detection in fixed wireless networks," in *Proc. Infocom*, 2009, pp. 1962–1970.

[42] N. M. Freris and A. Zouzias, "Fast distributed smoothing of relative measurements," in *Proc. CDC*, 2012, pp. 1411–1416.

Jianping He is currently pursuing the Ph.D. degree in control science and engineering at Zhejiang University, Hangzhou, China.

His research interests include time synchronization, consensus, and distributed security algorithm design problems in wireless sensor networks.

Dr. He is a member of the Group of Networked Sensing and Control (IIPC-nesC), State Key Laboratory of Industrial Control Technology, Zhejiang University.

Peng Cheng (M'10) received the B.E. degree in automation and the Ph.D. degree in control science and engineering from Zhejiang University, Hangzhou, China, in 2004 and 2009, respectively.

Currently, he is Associate Professor with the Department of Control Science and Engineering, Zhejiang University. His research interests include networked sensing and control, cyberphysical systems, and robust control.

Dr. Cheng serves as the publicity Co-Chair for IEEE MASS 2013.

Ling Shi received the B.S. degree in electrical and electronic engineering from the Hong Kong University of Science and Technology, Hong Kong, China, in 2002 and the Ph.D. degree in control and dynamical systems from the California Institute of Technology, Pasadena, CA, USA, in 2008.

Currently, he is an Assistant Professor at the Department of Electronic and Computer Engineering, Hong Kong University of Science and Technology. His research interests include networked control systems, wireless sensor networks, and distributed control.

Jiming Chen (M'08–SM'11) received the B.Sc. and Ph.D. degrees in control science and engineering from Zhejiang University, Hangzhou, China, in 2000 and 2005, respectively.

He was a Visiting Researcher at INRIA in 2006, National University of Singapore in 2007, and the University of Waterloo, Waterloo, ON, Canada, from 2008 to 2010. Currently, he is a Full Professor with the Department of Control Science and Engineering, and the coordinator of group of Networked Sensing and Control in the State Key Laboratory of Industrial Control Technology, and Vice Director of the Institute of Industrial Process Control at Zhejiang University, Hangzhou, China.

Dr. Chen also served/serves as Ad hoc and Sensor Network Symposium Co-Chair, IEEE Globecom 2011; General Symposia Co-Chair of ACM IWCMC 2009 and ACM IWCMC 2010, WiCON 2010 MAC track Co-Chair, IEEE MASS 2011 Publicity Co-Chair, IEEE DCOSS 2011 Publicity Co-Chair, IEEE ICDCS 2012 Publicity Co-Chair, IEEE ICCC 2012 Communications QoS and Reliability Symposium Co-Chair, IEEE SmartGridComm The Whole Picture Symposium Co-Chair, IEEE MASS 2013 Local Chair, Wireless Networking and Applications Symposium Co-Chair, IEEE ICCC 2013 and TPC member for IEEE ICDCS'10,'12,'13; IEEE MASS'10,'11,'13; IEEE SECON'11,'12; IEEE INFOCOM'11,'12,'13, etc. He is currently Associate Editors for several international journals, including IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS, IEEE TRANSACTIONS ON INDUSTRIAL ELECTRONICS, *IEEE Network*, *IET Communications*, etc. He was a guest editor of IEEE TRANSACTIONS ON AUTOMATIC CONTROL, *Computer Communication*, *Wireless Communication and Mobile Computer*, and *Journal of Network and Computer Applications*.

Youxian Sun joined the Department of Chemical Engineering, Zhejiang University, Hangzhou, China, in 1964. From 1984 to 1987, he was an Alexander Von Humboldt Research Fellow and Visiting Associate Professor at the University of Stuttgart, Stuttgart, Germany. He has been a Full Professor at Zhejiang University since 1988.

In 1995, he was elevated to an Academician of the Chinese Academy of Engineering, China. He is author/co-author of 450 journal and conference papers. Currently, he is the Director of the Institute of Industrial Process Control and the National Engineering Research Center of Industrial Automation, Zhejiang University. His current research interests include modeling, control, and optimization of complex systems, and robust control design and its applications.

Prof. Sun is President of the Chinese Association of Automation. He also served as Vice-Chairman of IFAC Pulp and Paper Committee and Vice-President of the China Instrument and Control Society.