Improving Multi-Hop Time Synchronization Performance in Wireless Sensor Networks Based on Packet-Relaying Gateways with Per-Hop Delay Compensation

Xintao Huan, *Graduate Student Member, IEEE*, Kyeong Soo Kim, *Senior Member, IEEE*, Sanghyuk Lee, *Senior Member, IEEE*, Eng Gee Lim, *Senior Member, IEEE*, and Alan Marshall, *Senior Member, IEEE*

***Abstract*—Based on the reverse asymmetric time synchronization framework, we have proposed several schemes with a major focus on the energy efficiency and computational complexity of a large number of battery-powered, low-cost sensor nodes in wireless sensor networks (WSNs). To address the cumulative end-to-end synchronization error, we have also introduced an idea of compensating for the processing delays at packet-relaying gateways as an energy-efficient way of multi-hop extension of WSN time synchronization schemes. In this paper, we present a comprehensive analysis of the multi-hop extension of WSN time synchronization schemes based on packet-relaying gateways with the per-hop delay compensation and the results of extensive experiments for the energy-efficient time synchronization schemes based on the reverse asymmetric time synchronization framework together with the flooding time synchronization protocol as a representative of existing schemes. Experimental results based on a real testbed demonstrate that the multi-hop extension based on packet-relaying gateways with the per-hop delay compensation greatly improves the performance of time synchronization of all the schemes considered compared to the multi-hop extension based on the conventional time-translating gateways.**

***Index Terms*—Multi-hop time synchronization, processing delay compensation, packet relaying, wireless sensor networks.**

I. Introduction

THE advent of the Internet of Things (IoT) ushers in large-scale monitoring and sensing [1], [2], which, combined with the artificial intelligence/machine learning (AI/ML), would bring a variety of intelligent applications enabling future smart homes to smart factories to smart cities [3], [4]. It is the *multi-hop extension* that enables a wireless sensor network (WSN)—i.e., one of the foundational components of IoT—to cover vast areas (e.g., farms [5], forests [6], or even metropolitan areas like Shanghai and New York City [3]) for such large-scale monitoring and sensing through long-range transmissions and flexible energy balancing among the sensor nodes. Providing time synchronization to WSN applications, which is indispensable to not only ordering the sensor data but also guaranteeing the collaboration of the sensor nodes, becomes more and more challenging as WSNs scale in size and coverage through multi-hop extension, especially considering the issues of the energy consumption and computational complexity of a large number of battery-powered, low-cost WSN sensor nodes as investigated in [7], [8].

Based on the way of transferring timing messages to sensor nodes, WSN time synchronization schemes could be classified into three major categories of *two-way message exchange*, *one-way message dissemination*, and *receiver–receiver synchronization* [9]. The former two categories could cover most WSN time synchronization schemes, which provide absolute timescales among sensor nodes with respect to the clock of a reference node (often called a head or a root node). Schemes based on two-way message exchange—e.g., timing-sync protocol for sensor networks (TPSN) [10] and recursive time synchronization protocol (RTSP) [11]—can compensate for propagation delay and, therefore, provide more accurate time synchronization. Though not being able to compensate for propagation delay, by the way, schemes based on one-way message dissemination can save the number of message exchanges and simplify the implementation at the expense of synchronization accuracy, which makes them popular for resource-constrained WSNs with a moderate coverage (e.g., 1 µs propagation delay for 300 meters); the flooding time synchronization protocol (FTSP) [12]—a representative of the one-way WSN time synchronization schemes—was the first to synchronize multi-hop WSNs through flooding synchronization messages, and many schemes leveraging the flooding time synchronization framework like flooding with clock speed agreement (FCSA) protocol [13], PulseSync [14] and external gradient time synchronization protocol (EGSync) [15] have been proposed to enhance the accuracy and coverage time of synchronization. Note that the introduction of media access control (MAC)-layer timestamping [11], [12] reduces the effect of random delays in timestamping and thereby greatly improves the synchronization accuracy of the schemes based on either two-way message exchange or one-way message dissemination.

The receiver-receiver synchronization, on the other hand, has been studied due to its distributed nature and reduc-

X. Huan is with the Department of Electrical Engineering and Electronics, University of Liverpool, Liverpool L69 3GJ, U.K., and also with the School of Advanced Technology, Xi'an Jiaotong-Liverpool University, Suzhou 215123, China (e-mail: Xintao.Huan@liverpool.ac.uk).

K. S. Kim, S. Lee and E. G. Lim are with the School of Advanced Technology, Xi'an Jiaotong-Liverpool University, Suzhou, P. R. China (e-mail: {Kyeongsoo.Kim, Sanghyuk.Lee, Enggee.Lim}@xjtlu.edu.cn).

A. Marshall is with the Department of Electrical Engineering and Electronics, University of Liverpool, Liverpool, UK (e-mail: Alan.Marshall@liverpool.ac.uk).

tion of the time-critical-path [16]: The reference broadcast synchronization (RBS) algorithm [17] exploits the broadcast channel through which messages from a sender are delivered to multiple receivers approximately at the same time. The receiver-to-receiver referenceless synchronization (termed in R4Syn) protocol proposed in [18] further distributes the role of the reference to all sensor nodes, which makes it completely decentralized. The reference broadcast infrastructure synchronization (RBIS) [19] investigates the applicability to industrial and home automation networks, while coefficient exchange synchronization protocol (CESP) [20] enhances the energy-efficiency.

With a major focus on the *energy efficiency and computational complexity* of a large number of battery-powered, low-cost WSN sensor nodes, we have also proposed a series of WSN time synchronization schemes based on the *reverse asymmetric time synchronization framework* [7], [8], [21]: First, we have proposed schemes based on the reverse two-way message exchange—i.e., energy-efficient time synchronization based on asynchronous source clock frequency recovery (EE-ASCFR) [7] and asymmetric high-precision time synchronization (AHTS) [21]—to reduce the energy consumption of the conventional two-way schemes while maintaining the synchronization accuracy through propagation delay compensation; initiating the two-way message exchange process from the head instead of sensor nodes, we could move most of the tasks related with time synchronization from sensor nodes to the head and thereby relieve the sensor nodes from the computational burden of time synchronization. In addition, bundling the upstream synchronization messages together with measurement data could reduce the energy consumption for message transmissions at the sensor nodes. As for one-way schemes, we have proposed beaconless asymmetric energy-efficient time synchronization scheme (BATS) [8] to reduce the energy consumption and computational burden at both gateway and leaf sensor nodes in multi-hop WSNs, where we can avoid broadcasting of beacons including time synchronization messages and their forwarding at each gateway and sensor nodes to achieve higher energy efficiency while maintaining microsecond-level time synchronization accuracy.

While extending the proposed time synchronization schemes to multi-hop WSNs based on the time translation (TT) method outlined in [7], we observed that multi-hop time synchronization faces a cumulative synchronization error caused by its per-hop synchronization strategy which results from the recursive TT at either gateways or the head [8], [21]. Note that the multi-hop extension based on TT is quite popular, especially among one-way schemes including FTSP. Since the sensor nodes in one-way flooding schemes are synchronized with the reference node through layer-by-layer TT, the closer the sensor node is to the reference node (i.e., the head), the better is its synchronization accuracy. In other words, the multi-hop synchronization accuracy in flooding schemes as well as BATS is curbed by TT method during the multi-hop extension through gateway nodes. The two-way schemes—i.e., the novel reverse two-way schemes like EE-ASCFR/AHTS as well as the conventional ones like TPSN—could also suffer from the cumulative errors in TT. Moreover, the multi-hop extension based on packet-relaying gateways described in [7] is no exception in this regard due to the cumulative errors caused again by the aforementioned processing delay during the packet-relaying process at the gateway nodes.

To address the cumulated multi-hop synchronization error induced by the processing delays at gateway nodes, therefore, we have recently proposed a novel per-hop delay compensation (PHDC) method [22] laying its foundation on the packet-relaying gateways [7], [23], where we demonstrate that PHDC is capable of alleviating the cumulative synchronization errors through a preliminary investigation with experimental results. To further extend the investigation based on coarse-level mathematical analysis and experiments only with one-way schemes in [22], in this paper we carry out an extensive comparative analysis of the performance of multi-hop extension based on TT and PHDC for both one-way and two-way WSN time synchronization schemes in the context of energy-efficient multi-hop WSN time synchronization with low computational complexity. The major contributions of our work in this paper are summarized as follows:

- First, we systematically and mathematically analyze the feasibility of compensating for the processing delay at gateway nodes with consideration on the effect of timestamping and clock skew compensation over multiple hops for both TT and PHDC methods.
- Second, we describe two implementation options for the multi-hop extension of WSN time synchronization schemes based on PHDC and discuss the details of PHDC implementation specific to the representative one-way schemes, i.e., BATS and FTSP, and the two-way scheme, i.e., EE-ASCFR.
- Third, we extend BATS, FTSP, and EE-ASCFR for the multi-hop time synchronization based on both TT and PHDC, and implement them for linear and tree topologies on a real WSN testbed consisting of TelosB sensor nodes [24] running TinyOS [25].
- Finally, we comprehensively demonstrate the experimental evaluation results, where the improvement brought by PHDC in multi-hop time synchronization performance is elucidated over the combination of three time synchronization schemes and two network topologies.

The rest of the paper is organized as follows: The related work are discussed in Section II. A systematic analysis of the effect of timestamping and clock skew compensation on PHDC in comparison to TT is carried out in Section III. The implementation of PHDC on both one-way and two-way schemes is exhibited in Section IV. The experimental results evaluated on a real testbed are demonstrated in Section V. Section VI concludes our work and outlines our future work.

II. Related Work

In this section, we review the existing work on the resolution of the per-hop cumulated synchronization error in multi-hop time synchronization in relation to our work.

A. *Per-Hop Synchronization Strategy*

In the per-hop synchronization strategy, the time synchronization between two adjacent sensor nodes is done through a

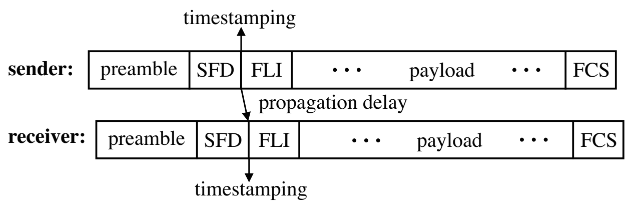


Fig. 1. MAC-layer timestamping in BATS [8], where SFD, FLI and FCS stand for start frame delimiter, frame length indicator, and frame check sequence, respectively.

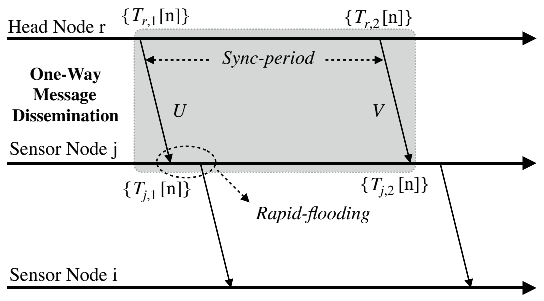


Fig. 2. Multiple-one-way-broadcast model introduced in [26].

pair of timestamps collected by the MAC-layer timestamping independent of the time synchronization between other nodes: As shown in Fig. 1, in the MAC-layer timestamping [21] the radio chip at a sender and a receiver triggers an interrupt immediately after the transmission and the reception of the start frame delimiter (SFD) byte of a frame to record timestamps, which could efficiently eliminate most of the MAC-layer processing delay. Hence, no processing delay related with timestamping in the per-hop synchronization strategy.

Note that the per-hop synchronization strategy is specific to the reverse asymmetric time synchronization framework aforementioned, where the head collects all the timestamps and establishes the time synchronization between the adjacent sensor nodes to reduce the effect of precision loss [8]. The end-to-end time synchronization over multiple hops in this strategy, however, is still achieved through a series of per-hop TT, which incurs the cumulative synchronization error. Sending all the timestamps to the head also takes the scarce payload space and thereby increases packet lengths.

B. Rapid-Flooding Multiple One-Way Broadcast

The rapid-flooding multiple one-way broadcast time synchronization (RMTS) is proposed to alleviate the synchronization error at the gateway nodes [26]. As illustrated in Fig. 2, through broadcasting N time information packets and collecting N timestamps at both sender and receiver during a single synchronization period, RMTS employs a timestamp set in estimating clock parameters unlike other time synchronization schemes employing a timestamp pair. The synchronization error, therefore, could be reduced due to the relatively more accurate estimation of the clock parameters.

Taking the estimation employing the downlinks U and V in Fig. 2 as an example, the observation timestamp sets—i.e., $u[n]$ and $v[n]$—used for clock skew estimation could be represented as follows [26]:

$$\{U, V\} = \{u[n], v[n]\}_{n=1}^{N}, \tag{1}$$

where

$$u[n] = T_{j,u}[n] - T_{r,u}[n],$$
$$v[n] = T_{j,v}[n] - T_{r,v}[n],$$

and $T_{r,u}[n]$ & $T_{r,v}[n]$ and $T_{j,u}[n]$ & $T_{j,v}[n]$ are the timestamps recorded at head node r and sensor node j during N broadcastings, respectively. Employing those timestamp sets and maximum likelihood estimation (MLE), RMTS estimates the clock parameters and thereby establishes the time synchronization. Note that the MAC-layer timestamping is also employed in RMTS.

In spite of its improved synchronization performance, RMTS is not suitable for energy-efficient time synchronization due to its innate tradeoff, i.e., using $N-1$ more message transmissions during a single synchronization interval compared to the one-way schemes based on a timestamp set like [12], which consumes more energy for packet transmission at resource-constrained sensor nodes.

C. Packet Relaying

Packet relaying, together with TT, is one of the two options discussed in [7] for the extension of EE-ASCFR to a hierarchical structure for network-wide, multi-hop time synchronization through gateway nodes. Unlike TT that is adopted for multi-hop extension by most time synchronization schemes (e.g., [8], [12], [26]), packet relaying is relatively simpler and provides a transparent end-to-end connection between the head and a leaf sensor node as far as time synchronization is concerned, which could reduce the problem of multi-hop time synchronization to that of single-hop time synchronization.

As pointed out in [7], [22], the performance of time synchronization based on packet relaying could be affected by rather large and random per-hop processing delay resulting from queueing/scheduling and MAC operations at each gateway. Due to its simplicity, however, packet relaying provides a more attractive option of multi-hop extension to time synchronization schemes highlighting energy efficiency and low computational complexity as their major advantages; especially for a large-scale deployment, the simplicity of packet relaying could relieve a large number of gateway nodes from the burden of multi-hop time synchronization in terms of energy and computation and thereby enable them to do other WSN-related tasks for offspring nodes as well as themselves for a longer period.

D. End-to-End Transparent Clock of IEEE 1588 Standard

In the context of wired networking, especially Ethernet, the idea of delay compensation at intermediate gateways has been investigated in time synchronization protocols like IEEE 1588 standard on precision time protocol (PTP) [27], where the idea of "end-to-end residence time correction" is described as the end-to-end transparent clock (E2E-TC).

The core idea of the end-to-end residence time correction of E2E-TC is somewhat similar to that of PHDC proposed in [22], but the actual implementation is more complicated and requires extra overhead, which would make it unsuitable for resource-constrained WSNs; specifically, accumulated residence times are stored in the correction field of the PTP event message or the associated follow-up message and carried all the way from a master to a slave.

Considering the bandwidth of typical WSNs (e.g., 250 kbit/s of Zigbee) and that of Ethernet (i.e., up to Gbit/s), we can see that PHDC is more tailored to WSNs as a simpler but more effective option for multi-hop extension of energy-efficient time synchronization schemes because the information on the processing delay is not carried all the way to the head due to its per-hop compensation of the processing delay, which does not incur any extra communication overhead compared to the end-to-end residence time correction of E2E-TC.

III. Effect of Timestamping and Clock Skew Compensation on Multi-Hop Extension Based on PHDC and TT

As discussed in Section I, PHDC is proposed to compensate for the processing delay at each packet-relaying gateway, which could address the issue of cumulative end-to-end synchronization error in multi-hop time synchronization [22]. Being based on packet relaying, PHDC inherits its major advantage of simple processing and reduces the problem of multi-hop time synchronization to that of single-hop time synchronization through transparent end-to-end connections between the head and leaf sensor nodes. Also, unlike the end-to-end residence time correction of E2E-TC, PHDC does not incur communication overhead. The preliminary analysis based on a single gateway in [22] shows that the clock skew could be selectively compensated for depending on the processing delay and relative skews of the sensor nodes, which is confirmed by the experimental results. In this section, we present a more comprehensive analysis of the effect of timestamping and clock skew compensation on PHDC in comparison with widely-employed TT.

A. Per-Hop Delay Compensation

We begin our analysis with the simplest case of the 2-hop WSN over a single gateway node shown in Fig. 3. During the kth (k=1, 2, ...) synchronization, the timestamp $\widehat{T1_2}^k$ of Fig. 3 (a), whose processing delay is compensated for at gateway node 1, can be described based on [22] as follows:

$$\widehat{T1_2}^k = T1_2^k + \left\lfloor R_{2,1}^k \times \Delta_1^k \right\rfloor, \tag{2}$$

where Δ_1^k is the processing delay at gateway node 1 estimated by $T1_1^k - T2_1^k$, and $R_{2,1}^k$ is the clock frequency ratio between gateway node 1 and sensor node 2 which could be estimated by either simple ratio-based method [28] or more advanced ones like linear regression [8]. Note that $T1_2^k$ is the timestamp recorded at sensor node 2 during the transmission of kth synchronization message and that $T2_1^k$ and $T1_1^k$ are the timestamps recorded at gateway node 1 during the reception and forwarding of that message, respectively. The *floor function* (i.e., $\lfloor\cdot\rfloor$) is used to convert the compensated delay to an integer number.[1]

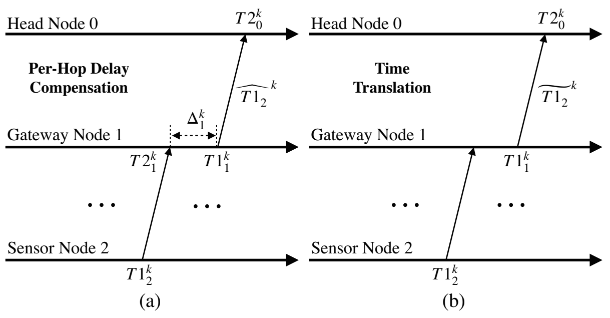


Fig. 3. A 2-hop WSN over a single gateway with multi-hop extension based on (a) per-hop delay compensation and (b) time translation.

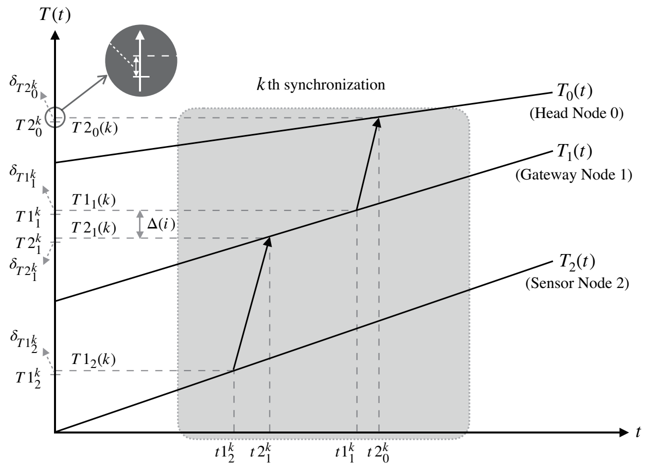


Fig. 4. Relationship between global reference times (e.g., $t1_2^k$) and corresponding continuous hardware clock times (e.g., $T1_2(k)$) and timestamps (e.g., $T1_2^k$) during the kth synchronization of the 2-hop WSN shown in Fig. 3.

Now we extend (2) by including the errors in timestamping of $T2_1^k$ and $T1_1^k$ and in clock skew compensation by $R_{2,1}^k$ at gateway node 1 to investigate their effect on PHDC[2]: First, we define a timestamping error of a timestamp as the differences between the continuous hardware clock time and the discrete timestamp as shown in Fig. 4, where we assume the first-order affine clock model [30] for nodes' hardware clocks; for instance, $\delta_{T2_1^k}$—i.e., the timestamping error of $T2_1^k$—is defined as $T2_1(k) - T2_1^k$ or frac($T2_1(k)$) where frac$(x) \triangleq x - \lfloor x \rfloor$ for $x \geq 0$; the timestamping errors could result from not only the integer conversion but also the remainder of interrupt delay compensation in MAC-layer timestamping [8], [11]. Second, we denote by $\delta_{\epsilon_{1,2}^k}$ the error in clock skew compensation including precision loss, where $\epsilon_{1,2}^k$ is a clock skew (i.e., $1+\epsilon_{1,2}^k = R_{1,2}^k$).

Let us consider the processing delay based on the discrete timestamps and compensated by the estimated value of clock

[1] Alternatively, the *ceiling function* could be used as in [29]

[2] We do not include the timestamping error for $T1_2^k$, which occurs at sensor node 2, in the following analyses as we focus on PHDC at gateway nodes.

skew in (2), i.e.,

$$\begin{aligned}\left\lfloor R_{2,1}^{k}\times\Delta_{1}^{k}\right\rfloor &= \left\lfloor\left(1+\epsilon_{1,2}^{k}+\delta_{\epsilon_{1,2}^{k}}\right)\times\left(T1_{1}^{k}-T2_{1}^{k}\right)\right\rfloor\\ &= T1_{1}^{k}-T2_{1}^{k}+\left\lfloor\left(\epsilon_{1,2}^{k}+\delta_{\epsilon_{1,2}^{k}}\right)\times\left(T1_{1}^{k}-T2_{1}^{k}\right)\right\rfloor,\end{aligned}\tag{3}$$

and that based on the continuous hardware clock times and compensated by the true value of clock skew, i.e.,

$$\begin{aligned}&R_{2,1}^{k}\times(T1_{1}(k)-T2_{1}(k))\\ &\quad=\left(1+\epsilon_{1,2}^{k}\right)\times\left(\left(T1_{1}^{k}+\delta_{T1_{1}^{k}}\right)-\left(T2_{1}^{k}+\delta_{T2_{1}^{k}}\right)\right).\end{aligned}\tag{4}$$

Subtracting (3) from (4) and rearranging it, we can obtain the *error in PHDC over a single gateway node* as follows:

$$\begin{aligned}&\left(\epsilon_{1,2}^{k}\Delta_{1}^{k}-\left\lfloor\left(\epsilon_{1,2}^{k}+\delta_{\epsilon_{1,2}^{k}}\right)\Delta_{1}^{k}\right\rfloor\right)+\left(\delta_{T1_{1}^{k}}-\delta_{T2_{1}^{k}}\right)\times\left(1+\epsilon_{1,2}^{k}\right)\\ &\quad\approx\left(\epsilon_{1,2}^{k}\Delta_{1}^{k}-\left\lfloor\epsilon_{1,2}^{k}\Delta_{1}^{k}\right\rfloor\right)+\left(\delta_{T1_{1}^{k}}-\delta_{T2_{1}^{k}}\right)\\ &\quad=\operatorname{frac}(\epsilon_{1,2}^{k}\Delta_{1}^{k})+\left(\delta_{T1_{1}^{k}}-\delta_{T2_{1}^{k}}\right),\end{aligned}\tag{5}$$

where the approximation is done on the basis of $\delta_{\epsilon_{1,2}^{k}}\ll\epsilon_{1,2}^{k}$ and $\epsilon_{1,2}^{k}\ll1$ because the clock skew compensation error (mainly precision loss) is less than 10^{-7} in 32-bit single-precision floating point arithmetic [21] and a typical frequency tolerance of a crystal over the manufacturing process (hence the clock skew) is ±100 ppm [31].

Now we can consider the effect of timestamping and clock skew compensation on PHDC over multiple gateway nodes based on (5). Let X_i and Y_i be random variables modeling the first and the second component of the error in PHDC in (5) at gateway node i. Then, we can model the *total error in PHDC for N-hop WSN* (i.e., over $N-1$ gateway nodes) as follows:

$$\sum_{i=1}^{N-1}(X_i+Y_i),\tag{6}$$

which gives the average of the total error in PHDC as follows:

$$\mathrm{E}\left[\sum_{i=1}^{N-1}(X_i+Y_i)\right]=\sum_{i=1}^{N-1}(\mathrm{E}[X_i]+\mathrm{E}[Y_i]).\tag{7}$$

Because timestamping errors (i.e., fractional part of continuous hardware clocks) are likely to be uniformly distributed in the range of $[0,1)$, Y_i modeling $\delta_{T1_i^k}-\delta_{T2_i^k}$ can be considered uniformly distributed as well in the range of $(-1,1)$ (i.e., $Y_i\sim U(-1,1)$), and $\mathrm{E}[Y_i]=0$. This means that the effects of timestamping on PHDC at multiple gateway nodes could be canceled one another. In such a case, the average of the total error in PHDC reduces to

$$\sum_{i=1}^{N-1}\mathrm{E}[X_i],\tag{8}$$

which is solely determined by X_i modeling $\operatorname{frac}(\epsilon_{i,i+1}^{k}\Delta_{i}^{k})$.

The variance of the total error can be obtained, too, on the condition that X_i's and Y_i's are independent of one another, i.e.,

$$\begin{aligned}\operatorname{Var}\left(\sum_{i=1}^{N-1}(X_i+Y_i)\right)&=\sum_{i=1}^{N-1}\operatorname{Var}(X_i)+\sum_{i=1}^{N-1}\operatorname{Var}(Y_i)\\ &=\sum_{i=1}^{N-1}\operatorname{Var}(X_i)+\frac{N-1}{3},\end{aligned}\tag{9}$$

because

$$\operatorname{Var}(Y_i)=\frac{(1-(-1))^2}{12}=\frac{1}{3}$$

for $Y_i\sim U(-1,1)$.

B. Time Translation

We also begin our analysis with the 2-hop WSN over a single gateway node shown in Fig. 3. During the kth time synchronization (k=1, 2, ...), the timestamp $\widetilde{T1_2}^k$ of Fig. 3 (b), which is translated at gateway node 1, can be expressed as follows:

$$\widetilde{T1_2}^k=\left\lfloor R_{2,1}^{k}\times T1_{2}^{k}+O_{1,2}^{k}\right\rfloor,\tag{10}$$

where $O_{1,2}^{k}$ is the clock offset between gateway node 1 and sensor node 2. Note that the translation of the received timestamp in (10) is not involved with timestamping unlike PHDC.

Let $\delta_{O_{1,2}^{k}}$ be the fractional part of the clock offset (i.e., $\operatorname{frac}(O_{1,2}^{k})$), which also takes into account the error in clock offset estimation including precision loss. Let us consider the translated timestamp based on the estimated value of clock skew, i.e.,

$$\begin{aligned}&\left\lfloor\left(1+\epsilon_{1,2}^{k}+\delta_{\epsilon_{1,2}^{k}}\right)T1_{2}^{k}+\left(\left\lfloor O_{1,2}^{k}\right\rfloor+\delta_{O_{1,2}^{k}}\right)\right\rfloor\\ &\quad=T1_{2}^{k}+\left\lfloor\left(\epsilon_{1,2}^{k}+\delta_{\epsilon_{1,2}^{k}}\right)T1_{2}^{k}+\delta_{O_{1,2}^{k}}\right\rfloor+\left\lfloor O_{1,2}^{k}\right\rfloor,\end{aligned}\tag{11}$$

and the translated time based on the true value of clock skew without integer conversion, i.e.,

$$(1+\epsilon_{1,2}^{k})T1_{2}^{k}+\left(\left\lfloor O_{1,2}^{k}\right\rfloor+\delta_{O_{1,2}^{k}}\right).\tag{12}$$

As in Section III-A, we can obtain the *error in TT over a single gateway node* by subtracting (11) from (12) as follows:

$$\begin{aligned}&\epsilon_{1,2}^{k}T1_{2}^{k}+\delta_{O_{1,2}^{k}}-\left\lfloor\left(\epsilon_{1,2}^{k}+\delta_{\epsilon_{1,2}^{k}}\right)T1_{2}^{k}+\delta_{O_{1,2}^{k}}\right\rfloor\\ &\quad\approx\epsilon_{1,2}^{k}T1_{2}^{k}+\delta_{O_{1,2}^{k}}-\left\lfloor\epsilon_{1,2}^{k}T1_{2}^{k}+\delta_{O_{1,2}^{k}}\right\rfloor\\ &\quad=\operatorname{frac}(\epsilon_{1,2}^{k}T1_{2}^{k}+\delta_{O_{1,2}^{k}}),\end{aligned}\tag{13}$$

where the approximation is done on the basis of $\delta_{\epsilon_{1,2}^{k}}\ll\epsilon_{1,2}^{k}$ as discussed in Section III-A.

As in Section III-A, we can consider the effect of timestamping and clock skew compensation on TT over multiple gateway nodes based on (13). Let Z_i be a random variable modeling the error in TT in (13) at gateway node i—i.e., $\operatorname{frac}(\epsilon_{i,i+1}^{k}T1_{i+1}^{k}+\delta_{O_{i,i+1}^{k}})$—that is in the range of $[0,1)$. Then, we can model the *total error in TT for N-hop WSN* as follows:

$$\sum_{i=1}^{N-1}Z_i,\tag{14}$$

which gives the average of the total error in TT as follows:

$$\mathrm{E}\left[\sum_{i=1}^{N-1}Z_i\right]=\sum_{i=1}^{N-1}\mathrm{E}[Z_i].\tag{15}$$

As in Section III-A, the variance of the total error in TT can be obtained on the condition that Z_i's are independent of one another, i.e.,

$$\mathrm{Var}\left(\sum_{i=1}^{N-1} Z_i\right) = \sum_{i=1}^{N-1} \mathrm{Var}\left(Z_i\right). \tag{16}$$

C. Comparison: PHDC vs. TT

Comparing the total error in PHDC and TT over multiple gateway nodes analyzed in the previous sections, we can find a couple of major differences between the two:

First, in case of PHDC, though the error over a single gateway node depends on both timestamping and clock skew compensation, the effect of timestamping can be canceled out, which leaves only the effect of clock skew compensation in the average of the total error over multiple gateway nodes in (8). In case of TT, on the other hand, the effect of both clock skew and offset compensation controls the average of the total error over multiple gateway nodes.

Second, the actual value of X_i in the total error in PHDC, which can be in the range of $[0, 1)$ by definition of frac($\cdot$), could be much smaller than one unless the traffic load of a gateway node and thereby its processing delay (i.e., Δ_i^k) is comparable to or larger than the inverse of the clock skew (i.e., $1/\epsilon_{i,i+1}^k$) because a typical value of clock skew is very small as discussed in Section III-A. This is not the case, however, for the two components of the error in TT, i.e., $\epsilon_{i,i+1}^k T1_{i+1}^k$ and $\delta_{O_{i,i+1}^k}$ in (13). As for the first component, unlike the processing delay in PHDC (i.e., $\Delta_i^k{=}T1_i^k{-}T2_i^k$), $T1_{i+1}^k$ in TT can take any integer value in the range of $[0, 2^M)$, where M is the size of a timestamp in bits, so its value can be large even after multiplied by the clock skew. Given that $\delta_{O_{i,i+1}^k}$ is the fractional part of the clock offset in the range of $[0, 1)$, it is likely that Z_i can take any value in the range of $[0, 1)$ unlike X_i.

In summary, we can conclude that the cumulative error in multi-hop time synchronization is well under control for PHDC in comparison to TT, because the major component in the error[3] in PHDC (i.e., timestamping error which is the fractional part of continuous hardware clock) can be canceled out over multiple gateway nodes, while that in the error in TT (i.e., the fractional part of the clock offset) cannot.

IV. On The Implementation of PHDC

In this section, we provide a general overview of PHDC implementation. We also discuss the details of the implementation specific to the two energy-efficient time synchronization schemes based on the reverse asymmetric framework (i.e., BATS and EE-ASCFR) as well as one of the most popular conventional schemes (i.e., FTSP).

A. Delay Estimation in PHDC

Consider two neighbor nodes—i.e., gateway nodes $i{-}1$ and i—in Fig. 5. During the kth synchronization, a pair

[3]On the condition that the error in clock skew compensation is negligible compared to the clock skew itself (i.e., $\delta_{\epsilon_{i,i+1}^k} {\ll} \epsilon_{i,i+1}^k$) as discussed in Section III-A.

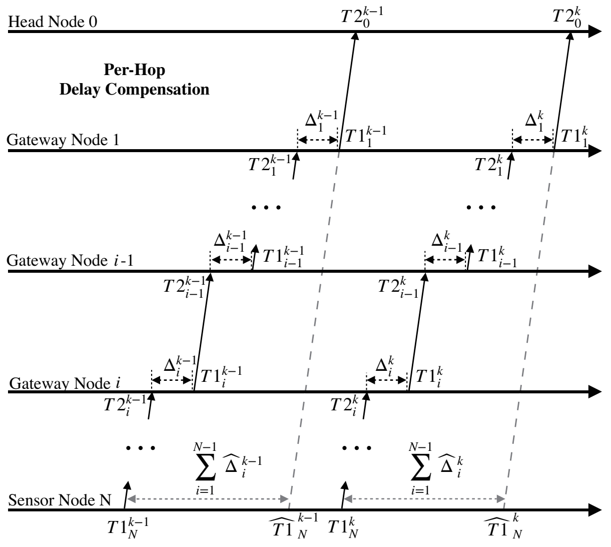


Fig. 5. PHDC implementation for one-way N-hop WSN time synchronization based on the reverse asymmetric framework.

of timestamps for the departure time $T1_i^k$ at gateway node i and the arrival $T2_{i-1}^k$ at gateway node $i{-}1$ are recorded through MAC-layer timestamping. The clock frequency ratio can be estimated based on a set of those timestamps using a simple ratio-based method or more advanced methods such as linear regression. Here, we consider the ratio-based estimation, where we can estimate the clock frequency ratio $R_{i,i-1}^k$ as follows[4]: For $k{>}l{\geq}1$,

$$R_{i,i-1}^k = \frac{T1_i^k - T1_i^l}{T2_{i-1}^k - T2_{i-1}^l}. \tag{17}$$

Note that, if we fix l to 1, (17) becomes the cumulative ratio (CR) method adopted in EE-ASCFR [7]; for simplicity, we set l to $k{-}1$ in the following. Then, the skew-compensated processing delay $\widehat{\Delta}_{i-1}^k$ based on (17) with $l{=}k{-}1$ is given by

$$\begin{aligned}\widehat{\Delta}_{i-1}^k &= R_{i,i-1}^k \times \Delta_{i-1}^k \\ &= \frac{T1_i^k - T1_i^{k-1}}{T2_{i-1}^k - T2_{i-1}^{k-1}} \times \left(T1_{i-1}^k - T2_{i-1}^k\right),\end{aligned} \tag{18}$$

where $\Delta_{i-1}^k{=}T1_{i-1}^k{-}T2_{i-1}^k$ as discussed in Section III-A.

One implementation option discussed in [22] is the centralization of PHDC in the head at the expense of the increased communication overhead, which can address the impact of limited precision floating-point arithmetic of gateway and sensor nodes on time synchronization. In this option, we could more accurately compensate for the processing delays by doing the following calculation at the head based on all the timestamps transferred from gateway and sensor nodes:

$$\widehat{T1}_N^k = T1_N^k + \sum_{i=1}^{N-1}\left(\left(\prod_{j=i+1}^{N} R_{j,j-1}^k\right) \times \Delta_i^k\right). \tag{19}$$

[4]The clock skew compensation could be ignored when the clock frequencies of all gateway and sensor nodes are synchronized to the head [7] or the processing delay is controlled within a certain bound [22].

An alternative, distributed implementation option for gateway and sensor nodes is to compensate for the processing delay and replace the original timestamp $T1_N^k$ from sensor node N with a new compensated timestamp *on the fly* through gateway nodes[5]: At gateway node i (i=1, . . ., N−1), we replace a received timestamp for the message departure time with a compensated timestamp $\widehat{T1}_{N,i}^k$ as follows:

$$\widehat{T1}_{N,i}^k = \begin{cases} \widehat{T1}_{N,i-1}^k + \widetilde{\Delta}_i^k & \text{if } i < N-1, \\ T1_N^k + \widehat{\Delta}_{N-1}^k & \text{if } i = N-1, \end{cases} \tag{20}$$

where

$$\widetilde{\Delta}_i^k = \frac{\widehat{T1}_{N,i+1}^k - \widehat{T1}_{N,i+1}^{k-1}}{T2_i^k - T2_i^{k-1}} \times (T1_i^k - T2_i^k). \tag{21}$$

Note that the estimation of the clock frequency ratio in (21) is now based on the updated timestamps. In this way, the compensation of processing delay based on clock skew can be done independently at each gateway node. This option could be readily implemented at the resource-constrained gateway and sensor nodes due to its simplicity and is our choice for PHDC implementation on the three representative schemes, which will be discussed in the next subsection.

With either of the PHDC implementation options, the head can finally estimate clock parameters based on the final pair of timestamps ($\widehat{T1}_N^k$, $T2_0^k$) during the kth synchronization as if the head and sensor node N are directly connected to each other.

B. Case Studies

Here we discuss the details of our own implementation of PHDC on three representative WSN time synchronization schemes for their performance analysis through real testbed experiments, whose results are reported in Section V.

1) BATS: Algorithm 1 describes the details of the lightweight PHDC implementation at gateway nodes for BATS, which was briefly sketched in [22], where a gateway node keeps track of timestamps $T1$ and $T2$ from the initial phase to the current and then estimate the most recent clock frequency ratio based on the current timestamps.

PHDC running on gateway node is formed with two parts locating respectively in the application and the MAC layer. The relatively complex processes, i.e., collecting timestamp pairs and calculating the frequency ratio, are done at the application layer. The frequency ratio is calculated following the illustration in Section IV-A, which is based on the simple ratio-based method leveraging the timestamp pairs of $T1$ and $T2$. Afterwards, the packet that contains the $T1$ together with the calculated frequency ratio and current $T2$ will be sent out to the MAC layer. The only operation which is done at the MAC layer besides the MAC-layer timestamping is the updating of the timestamp $T1$. Using the MAC-layer timestamp stored in $T1'$, we could update the $T1$ in the packet as exhibited in line 23 of the Algorithm 1. Note that, though this calculation is simple, it may delay the actual transmission of the packet due to its being done at the MAC

```
Data: The node maintains the following data and variables:
  • $e$: Event object including a timestamp;
  • $packet\_status$: Variable indicating the status of packet (i.e.,
    FIRST_PACKET or NON_FIRST_PACKET);
  • $d$: Measurement data;
  • $ts$: Measurement timestamp;
  • $p$: Packet object (optionally) including timestamps from
    MAC-layer timestamping;
  • $T1, T2, T1\_last, T2\_last, T1'$: Timestamps;
  • $R$: Frequency ratio;
  • $Q_M$: FIFO queue for measurement data;
  • $Q_P$: FIFO queue for packets;
  • $Q_{T2}$: FIFO queue for timestamp $T2$.
1  On detecting an event $e$:
2  switch $e.type$ do
3      case MEASUREMENT do // its own measurement
4          $d \leftarrow Q_M$.dequeue() // measurement data from the queue
5          $ts \leftarrow e$.getTimestamp()   // for measurement, not for synchronization
6          $p \leftarrow$ Packet($d, ts, T1$)   // create a packet object
7          sendToMAC($p$)   // send to MAC layer
8          $p.T1 \leftarrow$ getMACTimestamp()   // get MAC-layer timestamp
9          send($p$)   // send packet out
10     case PACKET do
11         if $p$.getDestAddress()$\neq HEAD$ then // packet received from other sensor nodes
12             $p \leftarrow Q_P$.dequeue()   // packet from the queue
13             $T2 \leftarrow Q_{T2}$.dequeue()   // from MAC-layer timestamping
14             $T1 \leftarrow p.T1$   // get T1 from the packet
15             if $packet\_status$==$FIRST\_PACKET$ then
16                 $R \leftarrow 1.0f$   // initialize frequency ratio variable
17             else
18                 $R \leftarrow (T1 - T1\_last)/(T2 - T2\_last)$ // calculate current frequency ratio
19             $T1\_last \leftarrow T1$   // update last T1
20             $T2\_last \leftarrow T2$   // update last T2
21             sendToMAC($p, R, T2$)   // send to MAC layer
22             $T1' \leftarrow$ getMACTimestamp()   // get MAC-layer timestamp
23             $p.T1 \leftarrow p.T1 + R * (T1' - T2)$   // update $T1$
24             send($p$)   // forward the packet
25         else
               // Process the packet from the head ...
26     otherwise do
           // Process other event ...
```

Algorithm 1: PHDC at a gateway node in BATS.

layer. This delay is system-specific, which could be counted in the interrupt delay and later handled by the receiver through interrupt delay compensation as illustrated in [11]. When the delay is relatively large, however, we could alternatively skip the $T1$ update at the MAC layer and send the corresponding frequency ratio and delay to the upper gateway node for post-updating the $T1$ at the application layer, which of course at the expense of payload overhead.

Compared to the original BATS based on TT, the multi-hop extension of BATS based on PHDC not only enhances the synchronization performance through processing delay compensation but also reduces the communication overhead, i.e., timestamps occupying the payload: For instance, $2N$ timestamps are required for synchronizing all the sensor nodes of a flat N-hop WSN in the case of the original BATS (e.g., refer to Fig. 8 (b) of [8]); for the same network, on the other hand, N is enough for BATS with PHDC as discussed

[5]Algorithm 1 in Section IV-B1 explains this option in more detail.

in Section IV-A. Also, we could directly establish the time synchronization between the head node and sensor node in BATS with PHDC due to the update procedure of $T1$ on the gateway nodes as illustrated in Fig. 3 unlike that between the sensor nodes in BATS's per-hop synchronization strategy. Thanks to the end-to-end time synchronization between the sensor node and the head, the time translation in the multi-hop scenario of BATS with PHDC becomes as straightforward as in the single-hop scenario [8], which could be established between sensor node i and the head as follows:

$$t = \frac{T_i(t) - \theta_i}{1 + \epsilon_i}, \tag{22}$$

where $T_i(t)$ and t denote the time of sensor node i and that of the head, and $1+\epsilon_i$ and θ_i are the estimated clock frequency ratio and offset between the two.

2) FTSP: Because the publicly-available implementation of FTSP is incomplete[6] [33], [34], we need to remold FTSP to achieve microsecond-level time synchronization accuracy: Considering that the performance of multi-hop extension of FTSP is limited by the accuracy of the calculation in linear regression, which could be affected by many factors such as the sample size and the precision of floating-point arithmetic. We first employ the MAC-layer timestamping suggested in [8] which is simpler and more prevalent now. We then adapt the specific linear regression method provided in public FTSP implementation [32], this method unlike other methods which do the linear regression directly using the pairs of reference and local timestamps [8], it carries out the linear regression from local timestamps to time offsets, i.e., time differences of reference time and local time.

Afterwards, we extend FTSP for multi-hop time synchronization based on PHDC following Algorithm 1 but with the direction from the head to the gateway and sensor nodes, which could provide microsecond-level time synchronization accuracy.

3) EE-ASCFR: We have investigated the multi-hop extension of EE-ASCFR based on TT in [21], where we identify the issue of precision loss in time synchronization due to the recursive nature of TT and propose AHTS to address it by moving all the time synchronization tasks except timestamping from gateway and sensor nodes to the head with higher computing and power resources. Note that, although AHTS could provide microsecond-level time synchronization accuracy, it is centralized implementation of EE-ASCFR with increased communication overhead.

Thanks to PHDC, now we can extend EE-ASCFR for multi-hop time synchronization while keeping its distributed nature, i.e., carrying out in parallel the synchronization of clock frequency at gateway and sensor nodes and clock offset at the head, respectively. Given the time t of a reference clock (i.e., the hardware clock of the head), we convert the logical clock time $\mathscr{T}_i\Big(T_i(t)\Big)$ of sensor node i based on its hardware

[6]For instance, the time synchronization accuracy of TinyOS public FTSP implementation [32] is limited to milliseconds.

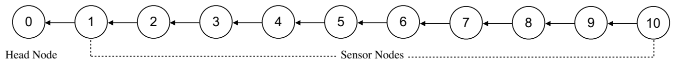


Fig. 6. 10-hop flat WSN testbed employed in the experiments.

clock time $T_i(t)$ presented in [7] as follows: For $t_k{<}t{\leq}t_{k+1}$ (k=0, 1, . . .),

$$\mathscr{T}_i\Big(T_i(t)\Big) = \mathscr{T}_i\Big(T_i(t_k)\Big) + R^k_{i,0} \times \Big(T_i(t) - T_i(t_k)\Big) \tag{23}$$

where t_k represents the time of a reference clock when a kth synchronization occurs, and $R^k_{i,0}$ is the clock frequency ratio between the head node 0 and sensor node i estimated as $(t_k{-}t_{k-1})/(T_i(t_k){-}T_i(t_{k-1}))$ based on the timestamp pairs from kth and $(k{-}1)$th synchronization, which is slightly different from and simpler than CR used in [21].

Note that in case of two-way time synchronization schemes like EE-ASCFR, PHDC is used for timestamps in both directions, i.e., from the head to sensor nodes or vice versa to establish the virtual two-way end-to-end connection between the head and sensor nodes.

V. Performance Evaluation

We have extended the three representative WSN time synchronization schemes discussed in Section IV-B—i.e., BATS, FTSP, and EE-ASCFR—for multi-hop time synchronization based on PHDC as well as TT and implemented them on a flat WSN testbed consisting of 11 TelosB motes running TinyOS as shown in Fig. 6 for a comparative analysis of their multi-hop time synchronization performance. The TelosB motes in our testbed embed with a 32-kHz crystal oscillator (CO) with a resolution of 30.5 µs and could provide a minimum resolution of 1 µs through running a digitally-controlled oscillator (DCO). Hence the time synchronization accuracy of the evaluated schemes is limited to microseconds, even though it has been shown that the schemes under reverse asymmetric framework could theoretically provide the possibility of sub-microsecond-level time synchronization accuracy [7].

In the following, we set the synchronization interval (SI) to 1 s for all the schemes and assume the self-data bundling option for BATS and EE-ASCFR for a fair comparison with FTSP.

A. BATS with TT and PHDC

Fig. 7 shows the mean absolute error (MAE) of measurement time estimation at each hop of BATS with TT and PHDC with its standard deviation (i.e., the errorbar). BATS with TT has a per-hop cumulative synchronization error of about 0.58 µs, which verifies the results of the analysis in Section III that the time translation process at gateway nodes of multi-hop extension based on TT could induce cumulative error. In contrast, BATS with PHDC demonstrates that the cumulative synchronization error over ten hops is 0.62 µs, which results in a much smaller per-hop cumulative synchronization error of 0.069 µs; as a result, the MAE of measurement time estimation of the farthest hop is maintained around 2 µs, i.e., much smaller than 7 µs for BATS with TT.

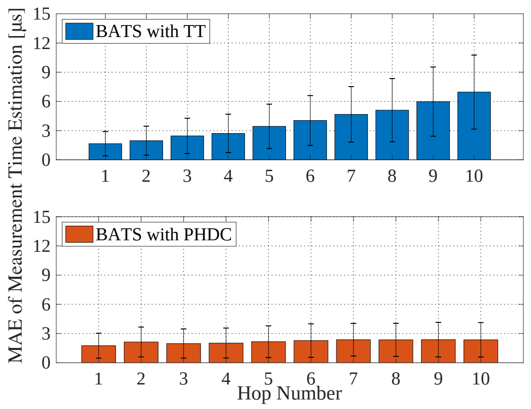
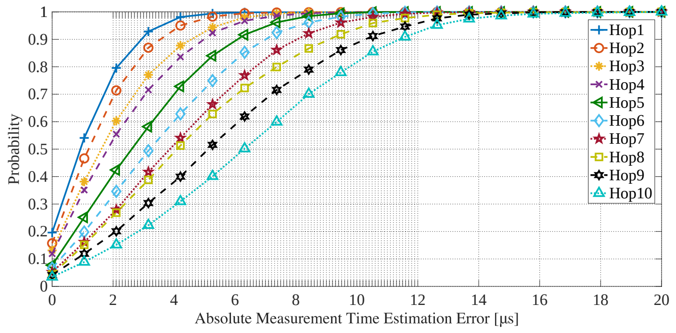
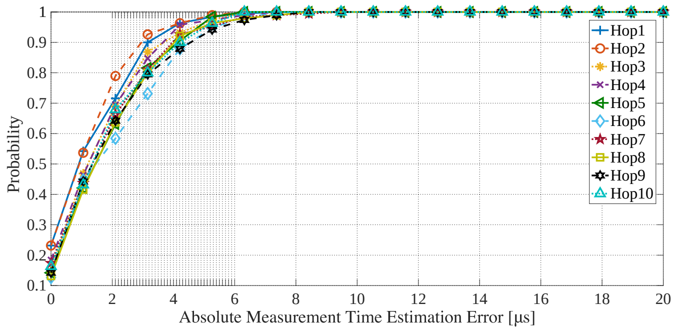


Fig. 7. MAE of measurement time estimation of BATS with TT and PHDC.



(a)



(b)

Fig. 8. Cumulative distribution function of absolute measurement time estimation error for BATS based on (a) TT and (b) PHDC.

The cumulative distribution functions (CDFs) of absolute measurement time estimation error shown in Fig. 8 provide a further evidence. For instance, the 90th-percentile absolute measurement time estimation error of BATS with TT is cumulatively increasing over ten hops from 2.9 µs to 11.4 µs, while that of BATS with PHDC hardly depends on the hop number.

An additional view of the measurement time estimation errors of BATS with PHDC over time is shown in Fig. 9 for a period of 3600 s; all sensor nodes across 10 hops could achieve approximately the same performance—i.e., the fluctuations of the measurement time estimation errors are similar from the first to the last hop.

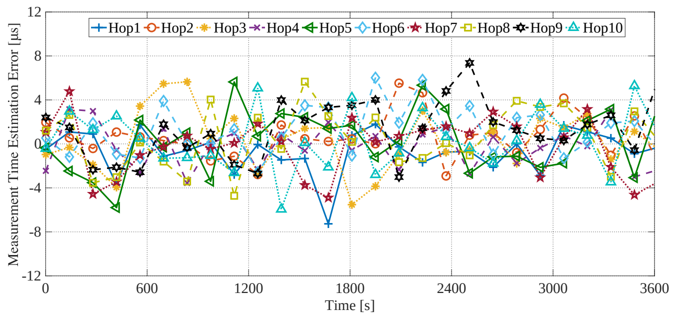


Fig. 9. Measurement time estimation errors of BATS with PHDC over 3600 s.

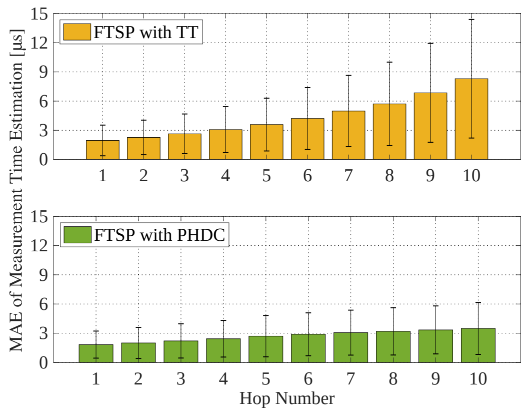


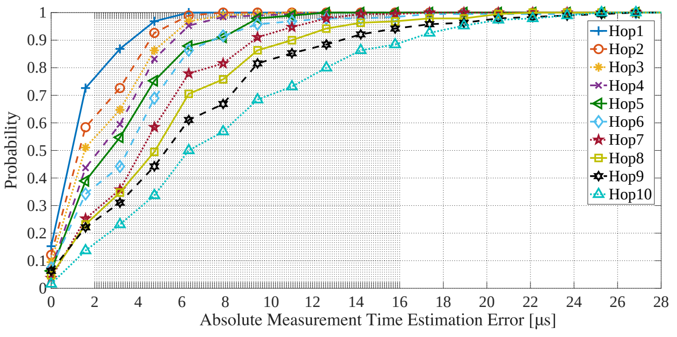
Fig. 10. MAE of the measurement time estimation of FTSP with TT and PHDC.

B. FTSP with TT and PHDC

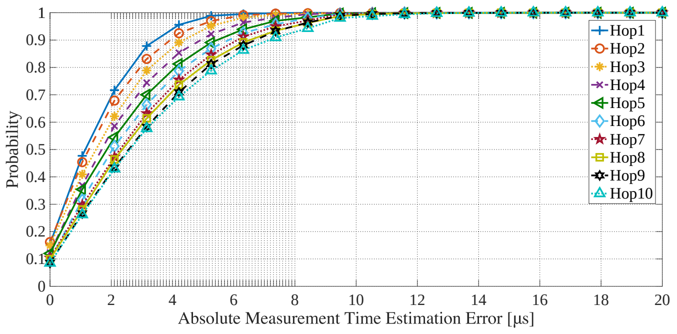
To demonstrate the wider applicability of PHDC to and its performance on conventional one-way schemes in addition to the reverse one-way schemes, we take FTSP—i.e., the representative conventional one-way flooding time synchronization scheme—as an example and extend it for multi-hop time synchronization based on both TT and PHDC. Fig. 10 shows the MAE of measurement time estimation and its standard deviation of FTSP with both TT and PHDC, where we can observe that the MAE of measurement time estimation at hop 10 with TT is more than double that with PHDC; FTSP with PHDC achieves 0.18 µs error per hop over 10 hops, which is more than 70% improvement over 0.7 µs error per hop for FTSP with TT.

The CDFs of absolute measurement time estimation error in Fig. 11 illustrate the nature of multi-hop time synchronization performance of FTSP with TT and PHDC in a clearer way. FTSP with TT has larger fluctuations in its absolute estimation errors where the maximum value exceeds 20 µs. In comparison, FTSP with PHDC has smaller fluctuations, and the 90%-percentile absolute estimation error at the last hop is smaller than 7.1 µs.

Though the multi-hop time synchronization performance of FTSP could be greatly enhanced by PHDC, it is still not as good as that of BATS with PHDC. This is because, unlike FTSP, the head in BATS receives timestamps from sensor nodes for synchronization due to its reverse asymmetric frame-



(a)



(b)

Fig. 11. Cumulative distribution function of absolute measurement time estimation error for FTSP based on (a) TT and (b) PHDC.

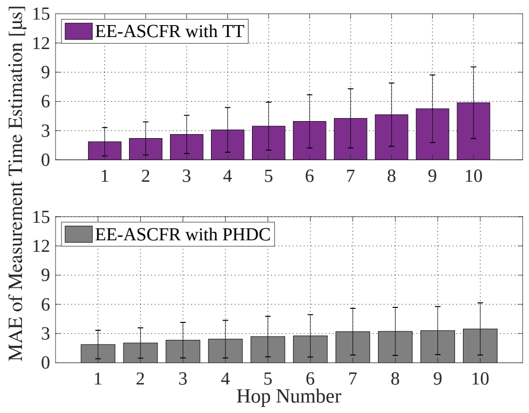
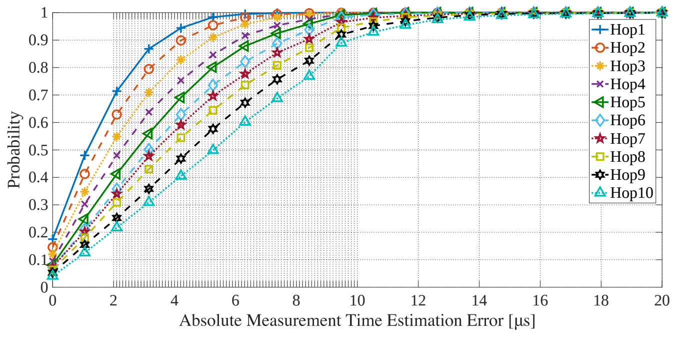
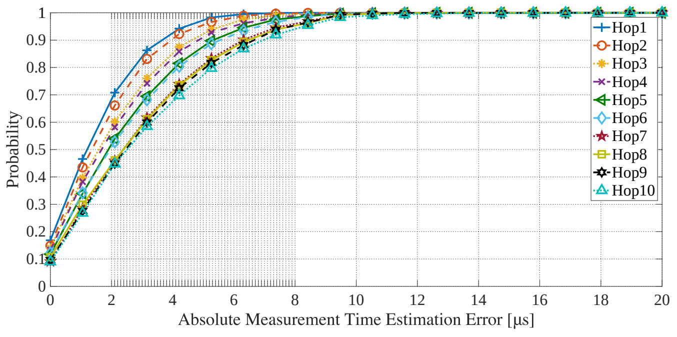


Fig. 12. MAE of the measurement time estimation of EE-ASCFR with TT and PHDC.



(a)



(b)

Fig. 13. Cumulative distribution function of absolute measurement time estimation error for EE-ASCFR based on (a) TT and (b) PHDC.

work and can increase the sample size and the complexity of the estimation algorithms due to its ample computational and power resources [8]. The experimental results in Sections V-A and V-B demonstrate that PHDC could effectively alleviate the cumulated synchronization error over multiple gateway nodes for both conventional and reverse one-way time synchronization schemes.

C. EE-ASCFR with TT and PHDC

In addition to the one-way time synchronization schemes, we also take EE-ASCFR as an example and demonstrate the effectiveness of PHDC on two-way schemes as well. Fig. 12 shows the MAE of measurement time estimation and its standard deviation of EE-ASCFR with TT and PHDC where the per-hop cumulative error is more clearly visible for TT, which is also confirmed by its more rapidly increasing standard deviation. Note that the MAE of measurement time estimation at the tenth hop of EE-ASCFR with TT is 5.87 µs, which is noticeably lower than those of the one-way schemes with TT, i.e., 6.96 µs in BATS with TT and 8.3 µs in FTSP with TT. This is because the reverse two-way message exchange procedure in EE-ASCFR can provide a better estimation of the clock offset due to its two-way nature and the clock skew estimation at each gateway or sensor node is not affected by TT. As for EE-ASCFR with PHDC, it also shows a slight increase in its MAE of measurement time estimation over the hops, which results from the clock skew estimation at each gateway and sensor node affected by PHDC. EE-ASCFR with PHDC, however, still performs 60% better than EE-ASCFR with TT in terms of per-hop error—i.e., 0.18 µs vs 0.45 µs, which interestingly is quite similar to that of FTSP with PHDC discussed in Section V-B.

Like BATS and FTSP with TT, the CDFs of absolute measurement time estimation error of EE-ASCFR with TT in Fig. 13 (a) show huge fluctuations, and the maximum absolute measurement time estimation error is close to 14 µs; only 65% of the errors at the tenth hop are distributed within ±7 µs. On the contrary, the CDFs of EE-ASCFR with PHDC in Fig. 13 (b) show that even the 90%-percentile at the last hop is less than 7 µs, which again is similar to that of FTSP with PHDC. Compared to BATS with PHDC, EE-ASCFR with PHDC has a slight per-hop error like FTSP with PHDC. This indicates that, as discussed in Section V-A, the centralized

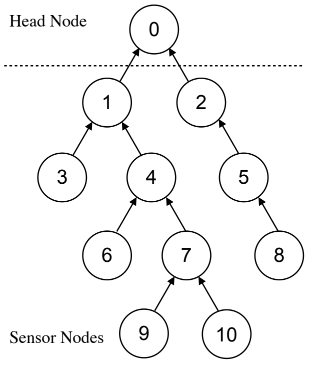


Fig. 14. 4-hop tree topology employed in the experiments.

clock parameter estimation in BATS has advantages over the distributed estimation of clock skew at gateway and sensor nodes in EE-ASCFR.

D. Impact of Network Topology on PHDC Performance

We have demonstrated so far the effectiveness of PHDC in multi-hop extension of various time synchronization schemes with a 10-hop flat WSN shown in Fig. 6. To investigate the impact of network topology on the performance of PHDC, we set up our testbed with the 4-hop tree topology consisting of 10 sensor nodes and 1 head shown in Fig. 14 for further evaluations. Of the three time synchronization schemes considered, we select BATS for the experiments with the 4-hop tree WSN, which provides the best performance for the flat WSN and thereby could better demonstrate the performance of PHDC.

TABLE I
MAE OF MEASUREMENT TIME ESTIMATION AND ITS STANDARD DEVIATION OF BATS-PHDC FOR THE MULTI-HOP TREE SCENARIO

Hop Number		MAE [1]	STD [1]
Hop	1	1.7018E-06	1.3445E-06
	2	1.9617E-06	1.4774E-06
	3	1.9266E-06	1.4781E-06
	4	1.7668E-06	1.3734E-06
	5	1.7664E-06	1.4023E-06
	6	2.0559E-06	1.5960E-06
	7	2.0270E-06	1.5804E-06
	8	1.8606E-06	1.4593E-06
	9	2.0569E-06	1.5954E-06
	10	2.0083E-06	1.5696E-06

[1] Based on the measurement time estimation obtained from 3600 s such that the actual performance in real deployment is represented.

Table I summarizes the MAE of measurement time estimation and its standard deviation for BATS with PHDC with the 4-hop WSN. From the results, we can observe that the maximum (i.e., that of node 9) and the minimum (i.e., that of node 1) MAE values are kept quite close to each other, i.e., with the difference of 0.3551 µs; a similar observation can be made in regard to the standard deviation of MAE, which shows the difference between the maximum and the minimum values of 0.2515 µs. Interestingly, the MAE of measurement time estimation with the tree topology does not show strict dependency on the hop count unlike that with the linear topology of Fig. 6: For instance, sensor node 8, which is 3 hops away from the head, achieves slightly better time synchronization performance than sensor nodes 2 and 3, which are one hop and two hops away from the head. These experimental results demonstrate the effectiveness of PHDC in multi-hop extension with tree topology as well as linear topology.

VI. CONCLUSIONS

We have investigated how to efficiently improve the performance of energy-efficient time synchronization schemes in multi-hop WSNs based on packet-relaying gateways with PHDC. Having identified that the cumulative errors over multiple gateways is the major cause of the multi-hop synchronization error, we carried out a preliminary study on multi-hop extension based on packet-relaying gateways and PHDC in our previous work [22]. Based on that preliminary study, we have systematically analyzed the feasibility of PHDC over single and multiple gateway nodes, especially the effect of timestamping and clock skew compensation including precision loss in the skew estimation. On the basis of this analysis, we have extended both reverse and conventional one-way schemes—i.e., BATS and FTSP—and one reverse two-way scheme—i.e., EE-ASCFR—for multi-hop time synchronization based on TT as well as PHDC, where, unlike the centralized multi-hop extension reported in [21], we provide the distributed multi-hop extension of EE-ASCFR for the first time in this paper.

The experimental results on a real 10-hop WSN testbed presented in Sections V-A to V-C clearly demonstrate the effectiveness of PHDC on both one-way and two-way schemes in alleviating the cumulative multi-hop time synchronization error. Specifically, BATS with PHDC can achieve nearly flat multi-hop synchronization accuracy; PHDC, compared to TT, also reduces more than 70% and 60% per-hop synchronization errors for FTSP and EE-ASCFR, respectively. From our observation of the results, we have also identified that, besides the multi-hop extension methods like PHDC and TT, there are other factors affecting the per-hop synchronization error in multi-hop time synchronization, including clock parameter estimation methods & sample sizes and the computational capability of the underlying platform (i.e., gateway and sensor nodes or the head).

Even though the evaluation results in this work demonstrate that the one-way schemes employing PHDC could achieve satisfactory multi-hop synchronization performance, it should be note that, when the propagation delay could not be ignored due to a larger communication range of a WSN, the use of two-way schemes would be essential for the compensation of the propagation delay. In this regard, the evaluation of PHDC on two-way schemes taking into account the propagation delay is essential to large-scale WSN deployments, which is an interesting topic for further investigation.

Also, note that our investigation of the centralized and distributed implementation options suggests that there is a research potential in centralized two-way schemes; while leveraging PHDC, more advanced clock parameter estimation methods could be employed in the centralized two-way time synchronization schemes to achieve better multi-hop time synchronization accuracy in the aforementioned large-scale deployments.

Acknowledgment

This work was supported in part by Research Development Fund (under Grant RDF-16-02-39), Key Programme Special Fund (under Grant KSF-E-25), the Centre for Smart Grid and Information Convergence (CeSGIC), and the AI University Research Centre (AI-URC) through Key Programme Special Fund (under Grant KSF-P-02) of Xi'an Jiaotong-Liverpool University.

References

[1] B. Maag, Z. Zhou, and L. Thiele, "A survey on sensor calibration in air pollution monitoring deployments," *IEEE Internet Things J.*, vol. 5, no. 6, pp. 4857–4870, Dec. 2018.

[2] P. Giri, K. Ng, and W. Phillips, "Wireless sensor network system for landslide monitoring and warning," *IEEE Trans. Instrum. Meas.*, vol. 68, no. 4, pp. 1210–1220, Apr. 2019.

[3] R. Du, P. Santi, M. Xiao, A. V. Vasilakos, and C. Fischione, "The sensable city: A survey on the deployment and management for smart city monitoring," *IEEE Commun. Surveys Tuts.*, vol. 21, no. 2, pp. 1533–1560, Secondquarter 2019.

[4] F. Cirillo, D. Gómez, L. Diez, I. Elicegui Maestro, T. B. J. Gilbert, and R. Akhavan, "Smart city IoT services creation through large-scale collaboration," *IEEE Internet Things J.*, vol. 7, no. 6, pp. 5267–5275, Jun. 2020.

[5] T. Ojha, S. Misra, and N. S. Raghuwanshib, "Wireless sensor networks for agriculture: The state-of-the-art in practice and future challenges," *Computers and Electronics in Agriculture*, vol. 118, pp. 66–84, Oct. 2015.

[6] Y. Liu, Y. He, M. Li, J. Wang, K. Liu, and X. Li, "Does wireless sensor network scale? a measurement study on greenorbs," *IEEE Trans. Parallel Distrib. Syst.*, vol. 24, no. 10, pp. 1983–1993, Oct. 2013.

[7] K. S. Kim, S. Lee, and E. G. Lim, "Energy-efficient time synchronization based on asynchronous source clock frequency recovery and reverse two-way message exchanges in wireless sensor networks," *IEEE Trans. Commun.*, vol. 65, no. 1, pp. 347–359, Jan. 2017.

[8] X. Huan, K. S. Kim, S. Lee, E. G. Lim, and A. Marshall, "A beaconless asymmetric energy-efficient time synchronization scheme for resource-constrained multi-hop wireless sensor networks," *IEEE Trans. Commun.*, vol. 68, no. 3, pp. 1716–1730, Mar. 2020.

[9] Y. Wu, Q. Chaudhari, and E. Serpedin, "Clock synchronization of wireless sensor networks," *IEEE Signal Process. Mag.*, vol. 28, no. 1, pp. 124–138, Jan. 2011.

[10] S. Ganeriwal, R. Kumar, and M. B. Srivastava, "Timing-sync protocol for sensor networks," in *Proc. SenSys'03*, Nov. 2003, pp. 138–149.

[11] M. Akhlaq and T. R. Sheltami, "RTSP: An accurate and energy-efficient protocol for clock synchronization in WSNs," *IEEE Trans. Instrum. Meas.*, vol. 62, no. 3, pp. 578–589, Mar. 2013.

[12] M. Maróti, B. Kusy, G. Simon, and Ákos Lédeczi, "The flooding time synchronization protocol," in *Proc. 2nd Int. Conf. SenSys*, Nov. 2004, pp. 39–49.

[13] K. S. Yildirim and A. Kantarci, "Time synchronization based on slow-flooding in wireless sensor networks," *IEEE Trans. Parallel Distrib. Syst.*, vol. 25, no. 1, pp. 244–253, Jan. 2014.

[14] C. Lenzen, P. Sommer, and R. Wattenhofer, "Pulsesync: An efficient and scalable clock synchronization protocol," *IEEE/ACM Trans. Netw.*, vol. 23, no. 3, pp. 717–727, Jun. 2015.

[15] K. S. Yildirim and A. Kantarci, "External gradient time synchronization in wireless sensor networks," *IEEE Trans. Parallel Distrib. Syst.*, vol. 25, no. 3, pp. 633–641, Mar. 2014.

[16] F. Sivrikaya and B. Yener, "Time synchronization in sensor networks: a survey," *IEEE Netw.*, vol. 18, no. 4, pp. 45–50, Jul. 2004.

[17] J. Elson, L. Girod, and D. Estrin, "Fine-grained network time synchronization using reference broadcasts," *SIGOPS Oper. Syst. Rev.*, vol. 36, no. SI, p. 147–163, Dec. 2003.

[18] D. Djenouri, N. Merabtine, F. Z. Mekahlia, and M. Doudou, "Fast distributed multi-hop relative time synchronization protocol and estimators for wireless sensor networks," *Ad Hoc Netw.*, vol. 11, no. 8, pp. 2329–2344, Jun. 2013.

[19] G. Cena, S. Scanzio, A. Valenzano, and C. Zunino, "Implementation and evaluation of the reference broadcast infrastructure synchronization protocol," *IEEE Trans. Ind. Informat.*, vol. 11, no. 3, pp. 801–811, Jun. 2015.

[20] F. Gong and M. L. Sichitiu, "CESP: A low-power high-accuracy time synchronization protocol," *IEEE Trans. Veh. Technol.*, vol. 65, no. 4, pp. 2387–2396, Apr. 2016.

[21] X. Huan and K. S. Kim, "On the practical implementation of propagation delay and clock skew compensated high-precision time synchronization schemes with resource-constrained sensor nodes in multi-hop wireless sensor networks," *Computer Networks*, vol. 166, p. 106959, Jan. 2020.

[22] X. Huan and K. S. Kim, "Per-hop delay compensation in time synchronization for multi-hop wireless sensor networks based on packet-relaying gateways," *IEEE Commun. Lett.*, vol. 24, no. 10, pp. 2300–2304, Oct. 2020.

[23] L. Xiao, X. Lu, D. Xu, Y. Tang, L. Wang, and W. Zhuang, "UAV relay in VANETs against smart jamming with reinforcement learning," *IEEE Trans. Veh. Technol.*, vol. 67, no. 5, pp. 4087–4097, May 2018.

[24] "TelosB datasheet," http://www2.ece.ohio-state.edu/~bibyk/ee582/telosMote.pdf, accessed: 2020-08-31.

[25] "Tinyos documentation wiki," http://tinyos.stanford.edu/tinyos-wiki/index.php/TinyOS_Documentation_Wiki, accessed: 2020-08-31.

[26] F. Shi, X. Tuo, S. X. Yang, J. Lu, and H. Li, "Rapid-flooding time synchronization for large-scale wireless sensor networks," *IEEE Trans. Ind. Informat.*, vol. 16, no. 3, pp. 1581–1590, Mar. 2020.

[27] *IEEE Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems*, Std., 2019.

[28] J.-P. Sheu, W.-K. Hu, and J.-C. Lin, "Ratio-based time synchronization protocol in wireless sensor networks," *Telecommunication Systems*, vol. 39, no. 1, pp. 25–35, Sep. 2008.

[29] B. Etzlinger, N. Palaoro, W. Haselmayr, B. Rudić, and A. Springer, "Timestamp free synchronization with sub-tick accuracy in the presence of discrete clocks," *IEEE Trans. Wireless Commun.*, vol. 16, no. 2, pp. 771–783, Feb. 2017.

[30] R. T. Rajan and A.-J. van der Veen, "Joint ranging and clock synchronization for a wireless network," in *Proc. CAMSAP 2011*, Dec. 2011, pp. 297–300.

[31] "Texas Instruments: Selection and specification of crystals for Texas Instruments USB 2.0 devices," https://www.ti.com/lit/an/slla122/slla122.pdf, accessed: 2020-08-31.

[32] "Public FTSP implementation in TinyOS," https://github.com/tinyos/tinyos-main/blob/master/tos/lib/ftsp, accessed: 2020-08-31.

[33] J. M. Castillo-Secilla, J. M. Palomares, and J. Olivares, "Temperature-compensated clock skew adjustment," *Sensors*, vol. 13, no. 8, pp. 10 981–11 006, Aug. 2013.

[34] D. Djenouri and M. Bagaa, "Implementation of high precision synchronization protocols in wireless sensor networks," in *Proc. WOCC 2014*, May 2014, pp. 1–6.

Xintao Huan (S'20) received the Ph.D. degree in electrical engineering and electronics from the University of Liverpool, U.K., in 2021, and the B.Sc. degree and the M.Sc. degree in computer engineering from the University of Duisburg-Essen, Germany, in 2013 and 2017. He was a Research Assistant with the Networked Embedded Systems Group, University of Duisburg-Essen, from 2012 to 2016. His research interests include wireless sensor networks and Internet of Things.

Kyeong Soo Kim (S'89-M'97-SM'19) received PhD degree in Electronics Engineering from Seoul National University, Korea, in 1995, and has been working as an associate professor of Electrical and Electronic Engineering at Xi'an Jiaotong-Liverpool University, China, since 2014. From 1996 to 1997, he was engaged in the development of multi-channel asynchronous transfer mode (ATM) switching systems as a Post-Doc researcher at Washington University in St. Louis, Missouri. From 1997 to 2000, he worked with the passive optical network (PON) Systems R&D organization of Lucent Technologies and was involved with development of ATM-PON systems, which won 1999 Bell Labs President's Silver Award. From 2001 to 2007, he was with STMicroelectronics, working as a Principal Engineer; during this period, he also took the position of STMicroelectronics Researcher-in-Residence at the Stanford Networking Research Center. From 2007 to 2014, he worked at Swansea University, U.K., as an associate professor. Dr. Kim is a senior member of IEEE and a member of IET.

Sanghyuk Lee (M'21-SM'21) received Doctorate degree from Seoul National University, Seoul, Korea, in Electrical Engineering in 1998. His main research interests include data evaluation with similarity measure, human signal analysis, high dimensional data analysis, controller design for linear/nonlinear system, and observer design for linear/nonlinear system. Dr. Lee is currently working as an Associate Professor at the Department of Electrical and Electronic Engineering of Xi'an Jiaotong-Liverpool University (XJTLU), Suzhou, China, which he joined in 2011. He has also been working as a founding director of the Centre for Smart Grid and Information Convergence (CeSGIC) in XJTLU since 2014. He has been serving as a Vice President of Korean Convergence Society (KCS) since 2012, and was appointed as an Adjunct Professor at Chiang Mai University, Chiang Mai, Thailand, in 2016. Dr. Lee organized several international conferences with KCS and was awarded multiple honors such as outstanding scholar/best paper award from KCS and Korean Fuzzy Society. Dr. Lee is a senior member of IEEE.

Eng Gee Lim (M'98-SM'12) received the BEng(Hons) and PhD degrees in Electrical and Electronic Engineering from the University of Northumbria, UK. Prof. Lim worked for Andrew Ltd, a leading communications systems company in the United Kingdom from 2002 to 2007. Since August 2007, Prof. Lim has been at Xian Jiaotong-Liverpool University, where he was formally the head of EEE department and University Dean of Research and Graduate studies. Now, he is School Dean of Advanced Technology, director of AI university research centre and also professor in department of Electrical and Electronic Engineering. He has published over 100 refereed international journals and conference papers. His research interests are Artificial Intelligence, robotics, AI+ Health care, international Standard (ISO/IEC) in Robotics, antennas, RF/microwave engineering, EM measurements/simulations, energy harvesting, power/energy transfer, smart-grid communication; wireless communication networks for smart and green cities. He is a charter engineer and Fellow of IET. In addition, he is also a senior member of IEEE and Senior Fellow of HEA.

Alan Marshall (M'88–SM'00) has spent more than 25 years working in the telecommunications and defense industries. He has been a Visiting Professor in network security with the University of Nice/CNRS, Nice, France, and an Adjunct Professor for Research with Sunway University, Malaysia, Subang Jaya, Malaysia. He is currently the Chair in Communications Networks with the University of Liverpool, Liverpool, U.K., where he is also the Director of the Advanced Networks Group. He has formed a successful spin-out company Traffic Observation and Management (TOM) Ltd., specializing in intrusion detection and prevention for wireless networks. He has authored more than 200 scientific papers and holds a number of joint patents in the areas of communications and network security. His research interests include network architectures and protocols, mobile and wireless networks, network security, high-speed packet switching, quality of service and experience architectures, and distributed haptics. He is a Fellow of The Institution of Engineering and Technology. He is a Section Editor (Section B: Computer and Communications Networks and Systems) for the Computer Journal of the British Computer Society, and sits on the program committees of a number of IEEE conferences.