A Robust Time Synchronization Scheme for Industrial Internet of Things

Tie Qiu, *Senior Member, IEEE*, Yushuang Zhang, *Student Member, IEEE*, Daji Qiao, *Senior Member, IEEE*, Xiaoyun Zhang, Mathew L. Wymore, and Arun Kumar Sangaiah

***Abstract*—Energy-efficient and robust-time synchronization is crucial for industrial Internet of things (IIoT). Some energy-efficient time synchronization schemes that achieve high accuracy have been proposed recently. However, some unsynchronized nodes namely isolated nodes exist in the schemes. To deal with the problem, this paper presents R-Sync, a robust time synchronization scheme for IIoT. We use a pulling timer to pull isolated nodes into synchronized networks whose initial value is set according to level of spanning tree. Then, another timer is set up to select backbone node and its initial value is related to the distance to parent node. Moreover, we do experiments based on simulation tool NS-2 and testbed based on wireless hardware nodes. The experimental results show that our approach makes all the nodes get synchronized and gets the better performance in terms of accuracy and energy consumption, compared with three existing time synchronization algorithms TPSN, GPA, STETS.**

***Index Terms*—Energy efficient, industrial Internet of things (IIoT), robust time synchronization, trust service.**

I. Introduction

THE industrial Internet of things (IIoT) applications have been widely studied and developed in recent years including wireless networks, mobile computing, and sensor devices [1]–[3]. A number of industrial IoT projects have been conducted in areas such as agriculture, food processing industry [4], [5]. The trust service, data processing, and information exchanging are required in the industrial wireless applications [6]–[8]. Consequently, wireless sensor nodes need a trust and robust time synchronization scheme, so that they could coordinate to finish the same task in IIoT [9]. Besides, much attention has been paid to energy consumption in IIoT [10], which is an important factor affecting network lifetime.

Time synchronization is a procedure offering a common notion of time for distributed system [11]. Network time protocol (NTP) [12] is a well-known time synchronization protocol for distributed networks, and it has many advantages for the complicated networks environment. When applied to IIoT, NTP brings many critical issues due to the limitations of energy consumption, adverse channel condition and the dynamic topology, etc. Therefore, many existing time synchronization protocols have been proposed for sensor networks [13].

As for distributed systems, program runs on more than one CPU so that it cannot guarantee the consistency of the start time. This leads to the initial time offset among CPUs. Moreover, the clock of CPU is driven by the internal crystal oscillators with the different frequencies. The light difference between CPUs, but nonignorable, might bring the clock [14]. In IIoT, sensor nodes that are deployed randomly, generate different local clock under the effect of clock drift and time offset. FTSP [15] and FCSA [16] were proposed to reduce the time error caused by clock drift as much as possible, but did not involve the energy-efficient improvement. In order to improve the accuracy of time synchronization, a majority of time synchronization algorithms have been presented, which focus on how to achieve high accuracy [17], [18]. Because of large amounts of packets exchanging and complex calculations, high accuracy is obtained at the sacrifice of too much energy consumption. The energy is valuable in IIoT, so it has become a crucial factor to evaluate a time synchronization algorithm. Pair broadcast synchronization protocol (PBS) [19] and Groupwise Pair Selection Algorithm (GPA) [20] for multicluster sensor networks reduce the energy consumption of the entire process, and achieve time synchronization in whole network.

The energy consumption and accuracy are two critical issues in designing time synchronization scheme due to the energy limitations of sensor nodes. Thus, energy consumption should be further reduced for a synchronization protocol without losing synchronization accuracy. In our previous work, a spanning tree-based energy-efficient time synchronization (STETS) [21] effectively decreases the energy consumption based on the combination of receiver-to-receiver protocol (RRP) and sender-to-receiver protocol (SRP). Especially in densely connected and large-scale networks, the energy-efficient performance is obvious. However, STETS cannot synchronize all the nodes in some cases through the theoretical analysis and simulation. The

Manuscript received May 4, 2017; revised July 27, 2017; accepted August 7, 2017. Date of publication August 11, 2017; date of current version August 1, 2018. This work was supported in part by the Natural Science Foundation of P.R. China under Grant 61672131 and the Fundamental Research Funds for the Central Universities (DUT16QY27). Paper no. TII-17-0952. *(Corresponding author: Tie Qiu.)*

T. Qiu, Y. Zhang are with the School of Software, Dalian University of Technology, Dalian 116620, China (e-mail: qiutie@ieee.org; Zhangyushuang@mail.dlut.edu.cn).

D. Qiao, X. Zhang, M. L. Wymore are with the Department of Electrical and Computer Engineering, Iowa State University, Ames, IA 50011 USA (e-mail: daji@iastate.edu; zxydut@iastate.edu; mlwymore@iastate.edu).

A. K. Sangaiah is with the School of Computing Science and Engineering, VIT University, Vellore 632014, India (e-mail: arunkumarsangaiah@gmail.com).

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

Digital Object Identifier 10.1109/TII.2017.2738842

main reason is that passive node (PN) in STETS cannot broadcast any message and fails to find some unsynchronized nodes (isolated nodes) in its communication range. Due to defect of algorithm design or missing message, some nodes do not join the synchronization process.

The main contributions of this paper are as follows.

1) We propose R-Sync, a robust time synchronization scheme for IIoT. Two timers are adopted in the scheme. One timer is for time synchronization using two-way message exchange. Each node sets up another timer at the beginning of synchronization process. The timer is used to pull isolated nodes to join the synchronized networks. So R-Sync is a trust time synchronization scheme.
2) A root node selection algorithm is given, which aims to balance energy consumption among sensor nodes. We choose a neighbor of last root node with the most residual energy as new root node. If hardware node cannot estimate the residual energy, the node with least broadcast messages is selected. The algorithm can also extend the lifetime of IIoT.
3) We do simulation using NS-2 and experiment based on wireless hardware nodes to evaluate the performance of R-Sync, and we compare R-Sync with other three existing time synchronization algorithms: TPSN [11], GPA [20], and STETS [21]. The results show that R-Sync makes all the nodes synchronized and is more energy-efficient than GPA and TPSN.

The rest of this paper is organized as follows. We give the related works and problem statement in Section II. The preliminaries are introduced in Section III. In Section IV, the algorithms are designed for topology construction and time synchronization. In Section V, we do the simulation using NS-2 to give the performance of R-Sync and compare with GPA, TPSN, and STETS. We implement R-Sync on the wireless hardware nodes to verify our proposed algorithm. Finally, the conclusion and future work are given in Section VI.

II. Related Works and Problem Statement

A. Related Works

There are many approaches for time synchronization, which have been extensively studied. Lim *et al.* recently presented TATS [22] that combines fast flooding and message delay compensation. TATS achieves submicrosecond synchronization error, but it is not involved in isolated nodes. Lenzen *et al.* employed a rapid-flooding time synchronization protocol PulseSync [23]. It also depends on the choice of a message delay calibration and on the distribution of propagation delays between sensor nodes. Each node uses a linear regression to estimate the clock of the root node. In FCSA [16], Yildirim and Kantarci *et al.* employed a clock speed agreement algorithm among sensor nodes to force all the nodes run at the same speed. FCSA is a slow-flooding-based time synchronization, but does not involve energy consumption affecting the entire network lifetime. In FTSP [15], sensor nodes broadcast the message using media access control (MAC)-layer timestamp and calculating skew based on linear regression. Also based on MAC-layer timestamp, a recursive time synchronization protocol was proposed by Akhlaq and Sheltami *et al.* in RTSP [24]. Unsynchronized node sends request recursively until synchronized node receives the request. It achieves low energy consumption owing to linear recursion between two nodes and infrequent broadcasting of reference node. By using MAC-layer timestamp, RTSP achieves high accuracy. MAC-layer timestamp can eliminate many source of errors, but it is not available in some real-hardware devices. It is assumed that any variations in processing delay is insensitive. In GTSP [25], Sommer and Wattenhofer *et al.* designed the algorithm to optimize the time error between neighbor nodes. The time synchronization is completely decentralized. At present, other time synchronization protocols are divided into two types typically, which are RRP [17], [26], [27] and SRP [11], [18], [28]. In SRP model, two sensor nodes synchronize with each other by a two-way message exchange. The messages carry the value of local timestamps. Nodes calculate the time offset through four collected timestamps. For RRP model, sensor nodes synchronize with reference node by listening the two-way message. RRP model is used to reduce the energy consumption.

RBS [17] is a typical approach based on RRP, which was proposed by Elson *et al.* The reference node broadcasts message, and other nodes receive the message to synchronize with each other. It reduces the timing critical path, which improves the accuracy, but energy saving cannot be guaranteed because of large amounts of exchanging timestamps to collect samples. The method limiting the transmission distance to reduce the power consumption has been proposed by Jain and Sharma *et al.* in OPRBS [27] based on RBS. But the number of broadcast messages does not decrease and the energy problem is still serious. Also based on RBS protocol, the reference node is changed periodically to avoid the over high consumption of the reference node in R^4syn [26]. A maximum likelihood method is used to improve synchronization accuracy [29] based on RRP, which was presented by Djenouri *et al.* The accuracy is related to the number of samples, which means message-exchange. Gong *et al.* proposed an energy-efficient coefficient exchange synchronization protocol (CESP) [30] based on RRP. It uses synchronization coefficient to reduce communication overhead and improves RBS. But CESP relies on a fixed reference node that consumes much energy. Ganeriwal *et al.* proposed TPSN [11], which established a spanning tree in the network and exchanged messages from the root to leaves. Then each node can synchronize with its parent node. In TSBST [28], He proposed a novel spanning tree synchronization algorithm. Each parent node selects only one child node to calculate the time offset. Then it broadcasts the offset and child nodes synchronize with it according to the offset. The approach has low energy consumption, but the accuracy of TSBST depends on selected child node, which is not stable.

Furthermore, Noh *et al.* presented the GPA [20], which is one of meaningful pair selection algorithms in the extension of PBS [19]. PBS proposed the combination of RRP and SRP. The entire network is divided into many groups. The nodes in the same group separately use SRP and RRP model to synchronize with the group leader. GPA largely reduces energy consumption after grouping and maintains high accuracy. But grouping

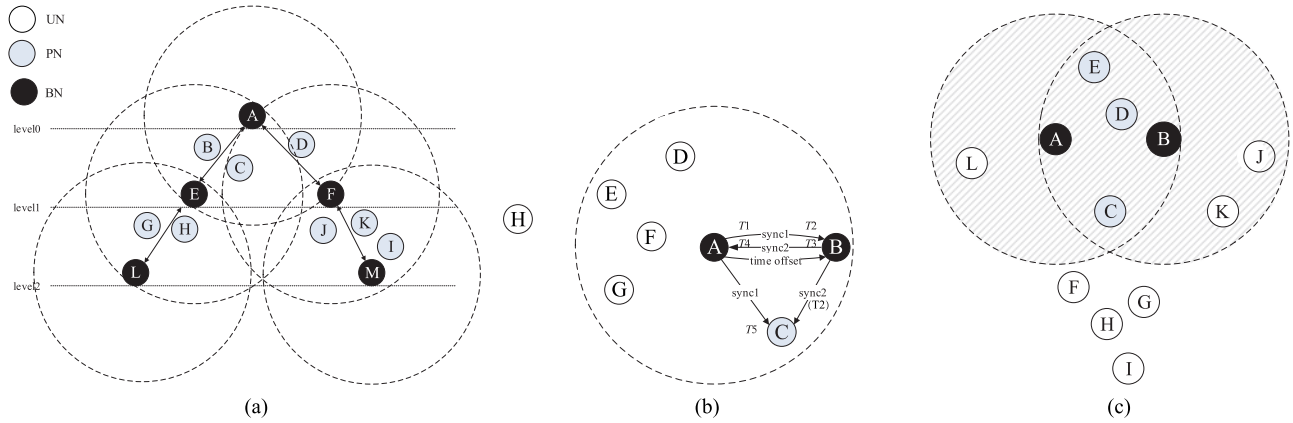


Fig. 1. Traditional time synchronization process based on spanning tree. (a) Spanning tree structure. (b) Two-way message exchange. (c) Isolated nodes.

process consumes much energy, especially in dynamic organizing topology that needs to be regrouped.

STETS [21], which is a spanning-tree-based time synchronization protocol, has been proposed. When applied to densely connected IIoT, low energy consumption is more evident. All nodes in the network are divided into three types. Undefined node (UN) is in the initial state, which is unsynchronized in IIoT. Backbone node (BN) is in the state, which forms the spanning tree, like black node in Fig. 1(a). The dashed circle means communication range of BNs in Fig. 1(a). And BNs synchronize with each other using SRP. By listening the information between two BNs, PNs could synchronize with the parent BN. PNs cannot send any broadcast message leading to some unsynchronized nodes. The type of node is only a flag in node. The transition just means that the value of flag is reassigned.

B. Problem Statement

There are many time synchronization algorithms [17], [18], which need to consume much energy. These methods get high accuracy at the sacrifice of energy consumption. Excessive energy consumption is likely to cause nodes failure, and further reduces the network lifetime.

After analysis, we find that GPA [20] consumes much energy when the topology is changed. The corresponding proof is given in Section V. STETS [21] gets a better performance for the volatile topology. A timer is used to avoid the collision and construct spanning tree. However, some isolated nodes exist in some algorithms based on two-way message exchange, which is described in Fig. 1(b). The dashed circle means communication range of node A in Fig. 1(b). Node A broadcasts a $sync1$ message at $T1$. Node B receives the $sync1$ message at $T2$ and replies a $sync2$ message at $T3$. Then node A receives the $sync2$ message at $T4$. Next, node A calculates the time offset between node A and node B according to four timestamps ($T1$ to $T4$) using SRP. Then node B receives the time offset to calibrate its local clock. Node C lies in the common communication range of node A and node B. So it is a PN node, which only listens to the overhead messages ($sync1$ and $sync2$). Node C records timestamp $T5$ at the moment of receiving $sync1$ message that carries the timestamp $T2$. Then it employs RRP to synchronize with node A. The two cases of isolated nodes are shown in Fig. 1(c). 1) Node F, G, H, I only lie in the communication range of a PN node (node C), but they are out of communication range of any BN. 2) Node K, J, L miss the message in two-way message exchange process due to poor signal or packet loss. In IIoT, sensor nodes are usually deployed in harsh outdoor environment and unstable channel condition, so packet loss is likely to happen. If any message is missed during exchanging timestamps, time synchronization is interrupted. They are all isolated nodes and do not join the networks. Since the topology is irregular and network environment is complex, the situations are likely to occur in IIoT. Besides, the phenomenon that only BN sends messages leads to unbalance energy among nodes in IIoT.

To provide a robust and energy-efficient time synchronization scheme, our approach is proposed.

III. Preliminaries

A. Assumptions

We assume that the distance between the node and its neighbors can be calculated. Classic distance estimation algorithm RSSI (received signal strength indicator) is widely used in wireless applications [31]. In this paper, we use RSSI to estimate the distance as

$$P_{\text{rcd}} = c * P_{tx}/(d^{\alpha}). \tag{1}$$

In (1), P_{rcd} stands for the received signal power and P_{tx} is the transmitted signal power. d is the distance we require, and c, α are parameters related to the propagation model.

The symbols are described in Table I.

B. Message Format

There are six types of messages appearing in different scenarios. The message format used in the algorithm is defined as follows.

1) DestAddr: destination address of the message.
2) SrcID: unique identifier of the sender.

TABLE I
SYMBOLS AND DESCRIPTION

Symbol	Description
T_i	ith timestamp
b_l	lth BN
$p_{l,i}$	ith PN whose parent node is lth BN
$\Delta(x, y)$	Measured time offset between node x and node y
$\mathrm{LE}(x)$	Local error of node x
$\mathrm{GE}(x)$	Global error of node x
$\mathrm{LT}(i)$	Local time of node i
$\mathrm{CT}(i)$	Calibration time of node i

DestAddr	SrcID	Type	ParentID	RTS	STS	Level

3) Type: the type of messages including ChRoot (for root selection), SetT (for setting up Pulling Timer), Init (for setting up Sync Timer), Sync (synchronization message), Ack (acknowledgement for receiving the Sync message), Pulling (for synchronizing isolated nodes).
4) ParentID: the unique identifier for the parent node. It is filled by NULL for root node.
5) RTS: the timestamp at the moment receiving the Sync message. Its default value is -1.
6) STS: the timestamp at the moment sending the Ack message. Its default value is -1.
7) Level: the level on the spanning tree.

C. Local Time Error and Global Time Error

The time error is an important factor to describe the difference among the values of calibration time for synchronized nodes. And calibration time of one node is the sum of local time and measured time offset. The local time is obtained by crystal oscillator within node and the measured time offset is calculated using SRP or RRP. Thus, time error can be used to represent the accuracy of time synchronization algorithm.

Definition 1: Local time error represents the difference of calibration time between a node and its parent node in spanning tree (i.e., one hop error). The local time error for a BN and a PN is shown in (2) and (3), respectively. LE means the local error. CT is the calibration time.

$$\mathrm{LE}(b_l) = |\mathrm{LT}(b_l) + \Delta(b_{l-1}, b_l) - \mathrm{CT}(b_{l-1})| \tag{2}$$

$$\mathrm{LE}(p_{l-1,l}) = |\mathrm{LT}(p_{l-1,l}) + \Delta(b_{l-1}, p_{l-1,l}) - \mathrm{CT}(b_{l-1})|\,. \tag{3}$$

For a BN and its parent node, the measured offset is gained by SRP using timestamps. PNs use RRP to calculate the measured offset.

Definition 2: Global time error represents the difference of calibration time between a node and the root node in spanning tree. The global time error of a BN and PN is shown in (4) and (5). GE is the global error and $root$ is the root node.

$$\mathrm{GE}(b_l) = |\mathrm{LT}(b_l) + \Delta(b_{l-1}, b_l) - \mathrm{LT}(\mathrm{root})| \tag{4}$$

$$\mathrm{GE}(p_{l-1,l}) = |\mathrm{LT}(p_{l-1,l}) + \Delta(b_{l-1}, p_{l-1,l}) - \mathrm{LT}(\mathrm{root})|\,. \tag{5}$$

Similarly to local time error, we may deduce global time error. All the nodes aim to get synchronized with root node, so root node does not need to calibrate time (i.e., $\mathrm{CT}(\mathrm{root}) = \mathrm{LT}(\mathrm{root})$).

IV. ALGORITHM DESIGN

A. Main Idea

As mentioned previously, TATS [22] and PulseSync [23] are proposed to significantly improve time synchronization accuracy. But both of them are not involved in isolated nodes. GPA [20] and TPSN [11] are two typical protocols based on spanning tree, which mainly focus on accuracy and energy and also do not include isolated nodes. Our approach R-Sync reduces energy consumption compared with GPA, and avoids the isolated nodes by setting up a pulling timer (PT) for every node. R-Sync consists of three parts.

1) Preparation for time synchronization is for root selection and the construction of spanning tree using flood protocol.
2) Initial time synchronization is for message-exchange algorithm by combining the SRP and RRP.
3) Pulling process is for avoiding the isolated nodes.

We simply describe the main idea. Once a node N does not synchronize in the certain time, its PT expires. It would broadcast a message requesting for time synchronization. Its neighbors whose node type is PN, response to node N. If there is no response due to missing message or without PN neighbors, node N resets up PT to wait. When more PNs receive the request, node N chooses first-reply PN to synchronize with and ignores other PNs.

Our main idea is shown in Fig. 2. The solid line without arrow indicates that two nodes lie in the communication range of each other. The dashed line with arrow indicates the sending message. The dashed circle indicates the communication range and solid circle represents the synchronized area. All nodes are UNs at the beginning. In the first round, root node is randomly selected based on node identifier. We select node A as the root. When initializing the topology, the root node A broadcasts a message to all neighbors. Any node that receives the message sets up the PT, and forward the message until all nodes set up the timer shown in Fig. 2(a). The initial value of PT is set according to level of spanning tree. When the node is synchronized, it cancels the PT. Fig. 2(b) shows the finished initial time synchronization. Then the PT of one unsynchronized node expires such as M in Fig. 2(b). It broadcasts a message to request for time synchronization. If there is no PN among its neighbors or the message is missed, M must reset up the PT. In contrast, the request message of node K and node R can be received by one PN in Fig. 2(c). Node G and node J receive the message and they convert themselves to BN. Thus, they begin to execute message-exchange synchronization algorithm using SRP and RRP, shown in Fig. 2(d). This process is repeated until all the nodes are synchronized, as shown in Fig. 2(e). Fig. 2(f) shows the process for root node selection. At the next round of the time synchronization, the node that lies in communication range of last root node and has the most residual energy, is selected as new root node such as B. Because it owns the most

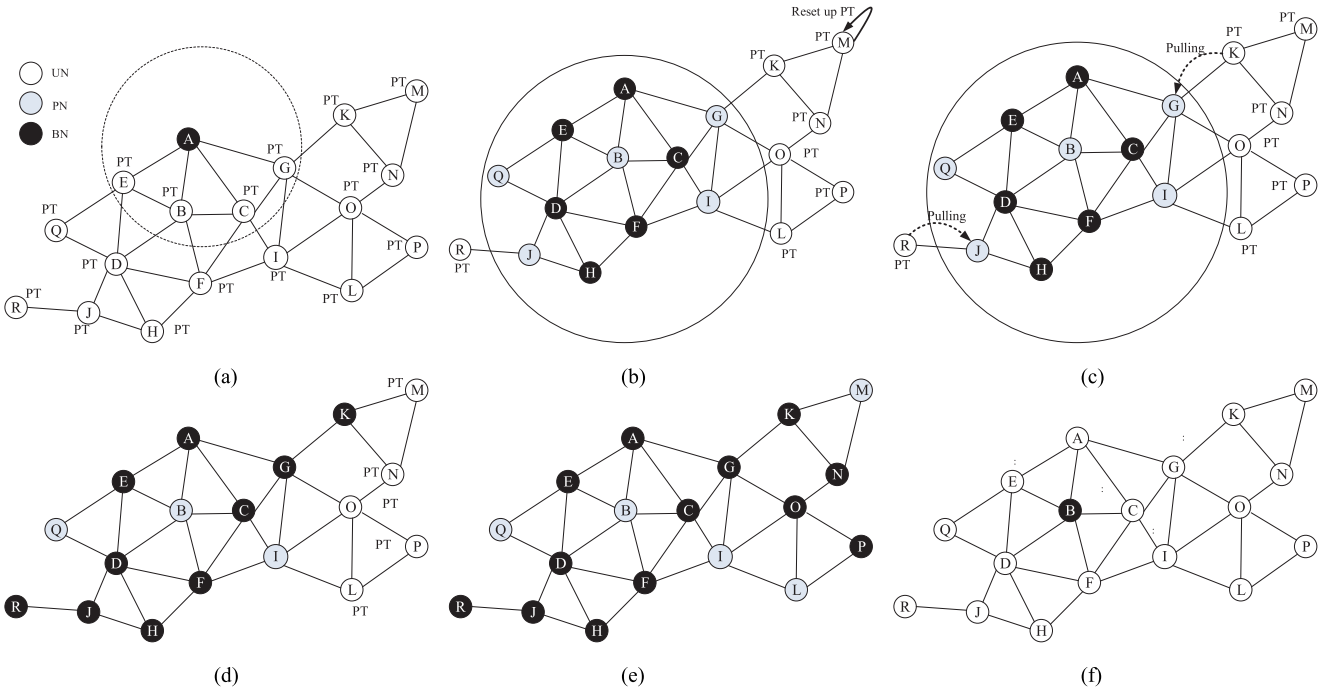


Fig. 2. Process of main idea. (a) Process of setting up PT. (b) One case of resetting up PT. (c) UNs broadcast a Pulling message. (d) PNs change to BNs. (e) Finished time synchronization. (f) Root selection strategy.

energy in all neighbors of node A. Therefore, the more the time synchronization rounds are, the better energy-balance is.

B. Time Synchronization Process

In this section, we present pulling mechanism with two timers. The sensor nodes are deployed in a certain area as shown in Fig. 2.

1) Preparation for Time Synchronization: In the beginning of each round of time synchronization, all nodes are UNs. Each UN has to set up a PT. Node A is selected randomly as root node in the first round. At first, it becomes a BN and its level is 0. Then the root node A broadcasts a SetT message. Other nodes that receive the message, set their levels to $L+1$ (L indicates sender's level), set up the PT, and forward the messages continuously. If a node has already set up a PT, it ignores the message. The SetT message of node A is filled as follow. The 0xFFFF in DestAddr means that the message is a broadcast message.

DestAddr	SrcID	Type	ParentID	RTS	STS	Level
0xFFFF	A	SetT	Null	-1	-1	0

2) Initial Time Synchronization: In this process, there are four phases to achieve the initial time synchronization using two-way message exchange.

Phase 1: After all nodes have set up the PT, the root node A broadcasts an Init message. The Init message of node A is filled as follow.

DestAddr	SrcID	Type	ParentID	RTS	STS	Level
0xFFFF	A	Init	Null	-1	-1	0

Phase 2: For each UN that receives an Init message (e.g. node C, B, G): If PT of the node is working, it is cancelled. The sender node A is marked as ParentID. Then it sets up a Sync timer (ST), which is used to select BN. It is different from PT. After ST of node C expires first, the node type of C converts from UN to BN. Then node C sends back a Sync message recording the local time $T1$.

Phase 3: For the node that receives a Sync message:

Case 1: If the node is an UN, DestAddr in the message is not equal to UN's ID (e.g., node B, G). It records the local timestamp $T5$ at receiving the message and cancels the ST. Then it changes its node type to PN.

Case 2: DestAddr in the Sync message is equal to the node's ID (e.g., node A). It records the local timestamp $T2$ when receiving the message and replies an Ack message at the moment $T3$. The Ack message carries the timestamps $T2$ and $T3$ in the field of RTS and STS.

Phase 4: For the node that receives an Ack message:

Case 1: DestAddr in the Sync message is equal to the node's ID (e.g., node C). The local timestamp $T4$ is recorded at receiving the message. The offset can be calculated by four timestamps $T1$, $T4$, RTS, and STS using SRP. So that it can synchronize with its parent node A by local clock and the offset. Then it broadcasts an Init message.

Case 2: If the node is a PN, DestAddr in the message is not equal to node's ID (e.g., node B, G). According to timestamp

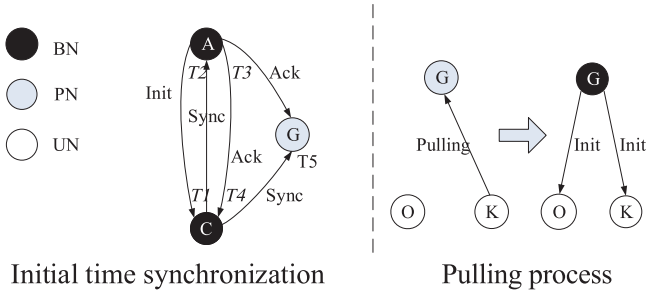


Fig. 3. Process of time synchronization.

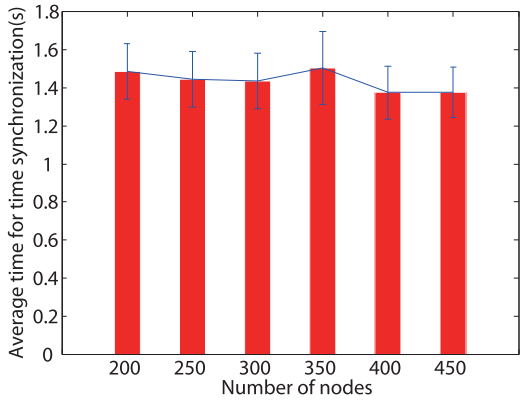


Fig. 4. Average running time of each hop.

$T5$ and RTS carried in the message, the time offset to its parent node can be calculated using RRP.

Phases 2–4 are repeated in the topology continuously. When BN at the leaves of spanning tree broadcasts an Init message without any reply, the initial time synchronization phases finish. But some UNs are not involved in the initial time synchronization. The UNs, which only lie in the communication range of PNs or miss message, are isolated (e.g., node K, O). In order to save these unsynchronized UNs, the PT starts to work.

3) Pulling Process: When node K's PT expires, it must be an UN. Then it broadcasts a Pulling message. If it does not receive any response (i.e., Init message) or misses the message, it resets up the PT immediately.

For the node that receives a Pulling message (e.g., G): Node G converts its node type from PN to BN. Then it broadcasts an Init message. Thus, phase 2-phase 4 in initial time synchronization can be repeated. One time synchronization process and corresponding timing chart are shown in Fig. 3.

4) Root Selection for Next Round: When neighbors of root node A get synchronized, they send their residual energy to node A. The root node selects one neighbor (e.g., node B) with the most residual energy. Then the root node sends a ChRoot message to node B. The node B is changed to the new root node, when it receives the message.

The pseudocode of above four processes is shown in Appendix. As for algorithm complexity of R-Sync, in addition to root node, each PN or BN only stores the offset in space complexity. The calculations for offset are not involved in loops. Thus, time complexity of PN is $\mathcal{O}(1)$ at best or worst case. BN's time complexity depends on the number of its child nodes. There is extra time cost for nodes that have to send Pulling message. It depends on the number of hops to the recent PN. In densely connected and large-scale network, the number of Pulling message is limited. So R-Sync is a lightweight algorithm.

C. Timers Initialization

For PT, we acquire the average time each hop of synchronization process through simulation, named AT_Period. Average time each hop refers to time cost of two-way message exchange process. One process starts when father node broadcasts initial message and ends until son nodes finish time synchronization. The results of AT_Period in simulation is shown in Fig. 4. And we get the specific value of AT_Period according to

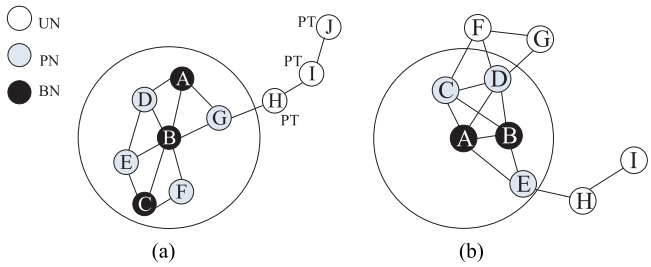


Fig. 5. Initial value for two timers. (a) Case of setting up PT. (b) Case of BN selection.

network size. Moreover, we name the period of time $Init\_Time$ for topology initialization at the beginning. And the root node starts time synchronization after topology initialization. The initial value of PT is set according to nodes level of spanning tree. And initial value of PT is $L * AT\_Period$+$Init\_Time$. In Fig. 5(a), node A is the root node without setting up PT. The initial value of node G is AT_Period+$Init\_Time$ and node H is 2* AT_Period+$Init\_Time$. Setting up PT according to level is used to reduce the situation of PT resetting up.

For ST, if the initial value is randomly selected, two BNs may be closer to each other. It is likely to slow the propagation of time synchronization process, such as node A and B in Fig. 5(b). Therefore, we should choose a BN as far as possible. Two BNs can cover more nodes when they are farther from each other. And the period of time synchronization can be reduced. The node can calculate the distance to the sender according to (1). So the initial value of ST is determined by the distance. The farther the distance is, the smaller the initial value is. The initial value of ST is $\alpha + 1/distance$, where α is a constant.

V. Simulation and Performance Evaluation

A. Simulation Setup

We use network simulation tools NS-2 to compare and analyze R-Sync with TPSN [11], GPA [20], and STETS [21]. Stochastic topology method is used in simulation, and all sensor nodes are randomly deployed in area of 1000 m $\times$ 1000 m. The model of radio propagation is two-ray ground that is suitable for long distance transmission. We use energy model in NS-2

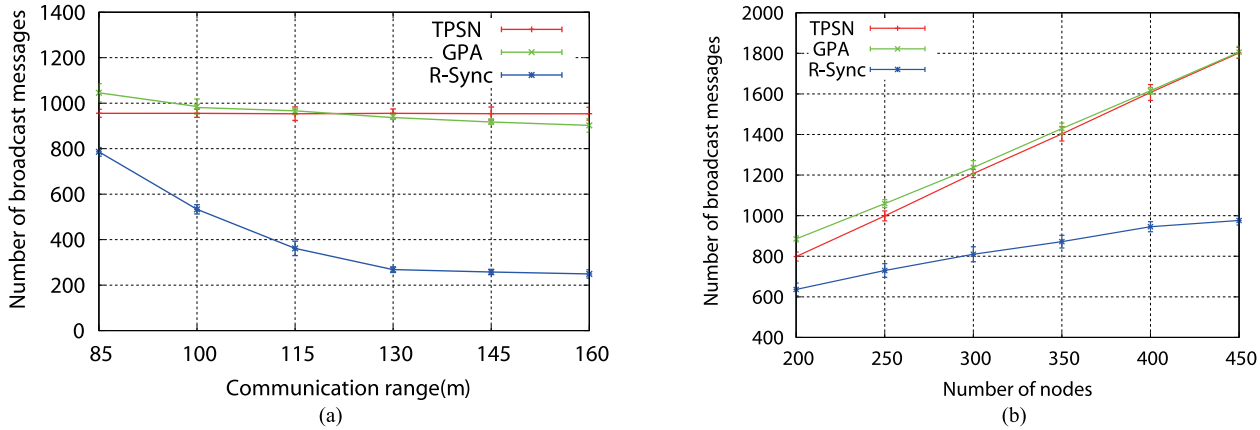


Fig. 6. Required number of messages. (a) Different communication ranges. (b) Different number of nodes.

to evaluate the energy consumption. The transmission power is 0.6 w, receiving power is 0.3 w, and idle power is 0.15 w. We focus on the number of broadcast messages, the percentage of synchronized nodes, time synchronization error, and energy balance to show the performance of R-Sync. In order to reduce the impact of random errors, we run 10 000 cycles for each experiment and get the average value of them.

B. Number of Messages

Here all the messages required broadcast are used to exchange timestamps and construct initial network architecture.

Lemma 1: Let M_{TPSN} be the number of required broadcast messages in TPSN, then $M_{TPSN} = 4L - 3$, where L is the number of overall nodes in IIoT.

Proof: For TPSN, the tree is constructed in the beginning. L messages are required. Since each node is connected to its parent node except the root node, there are $L - 1$ edges in a tree. In addition, three messages are required in every edge [11]. Thus, the number of required broadcast messages in TPSN is $M_{TPSN} = L + 3(L - 1) = 4L - 3$. ■

Lemma 2: Let M_{GPA} be the required number of broadcast messages in GPA. Then $M_{GPA} = 3L + 3B - 5$, where B is the number of BN (the BN in each group) in the network.

Proof: All the nodes are divided into several groups, which need L messages. Also, the group leader has to master the connection information among all the child nodes in a group to calculate the maximum connection child node. For every group, each child node needs to broadcast one connection discovery message to other child nodes and then sends another message carrying the number of connection child nodes to group leader, so $2 * (L - 1)$ messages are required. Besides, three messages are used for each nonroot node in every pairwise synchronization and the number of BN node is B [20]. Therefore, the number of required broadcast messages in GPA is the sum of above three parts: $M_{GPA} = L + 2(L - 1) + 3(B - 1) = 3L + 3B - 5$.

Lemma 3: Let $M_{R\text{-}Sync}$ be the number of required broadcast messages in R-Sync. Then MR-Sync $= L + 3(B - 1) + R$, where R is the number of Pulling messages in pulling process.

Proof: The spanning tree is constructed and the PT is set up at the beginning of R-Sync. Each node needs to broadcast one message. In the initial time synchronization phases, the message number is three times as many as the number of BN link in the network. Additionally, extra required message is Pulling message. Thus, the number of required broadcast messages in R-Sync is $M_{R\text{-}Sync} = L + 3(B - 1) + R$. ■

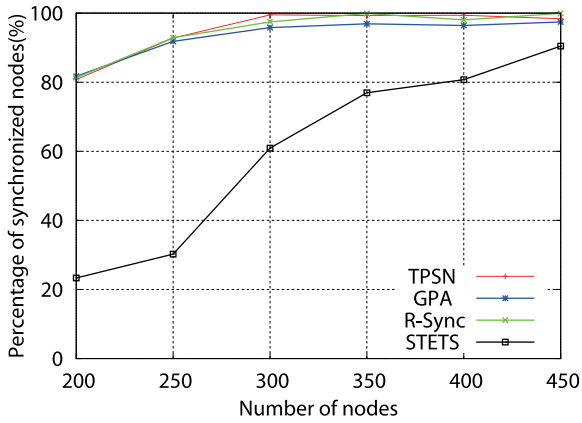


Fig. 7. Time synchronization percentage.

Compared with GPA, the nodes are not grouped in R-Sync. Some Pulling messages exist in pulling process, but the number of these messages is quite limited. Furthermore, in densely connected network, B and R decrease due to the increasing number of covered nodes. L is fixed for a certain topology. Thus, $M_{R\text{-}Sync}$ decreases through Lemma 3.

In Fig. 6(a), the broadcast number of R-Sync is compared with that of TPSN and GPA under different communication range in the entire network. In the simulation, the sensor nodes are randomly deployed in the area 1000 m $\times$ 1000 m. The number of sensor nodes is set to 240. It can be seen that R-Sync requires a much lower number of broadcast messages than TPSN and GPA with the communication range increasing. From the Lemma 1, the number of broadcast messages in TPSN is only related to the number of sensor nodes, so the numbers of broadcast messages in different communication ranges are almost same. Besides, the proposed R-Sync decreases faster than GPA due to that the Pulling messages decrease more sharply according to Lemma 2 and Lemma 3.

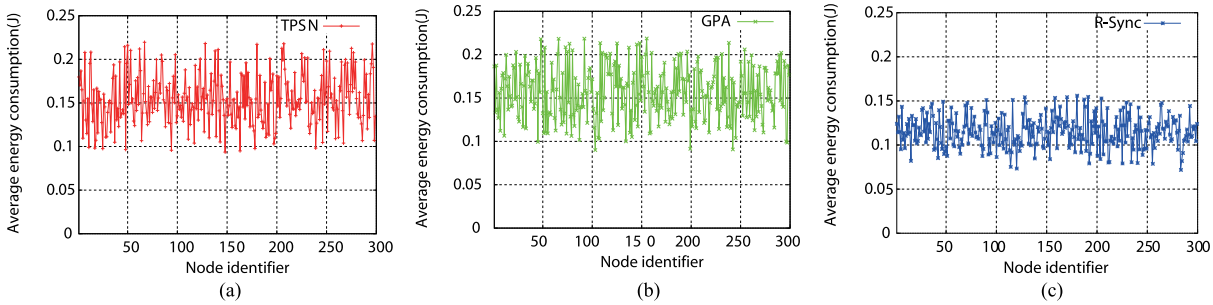


Fig. 8. Energy consumption distribution. (a) TPSN. (b) GPA. (c) R-Sync.

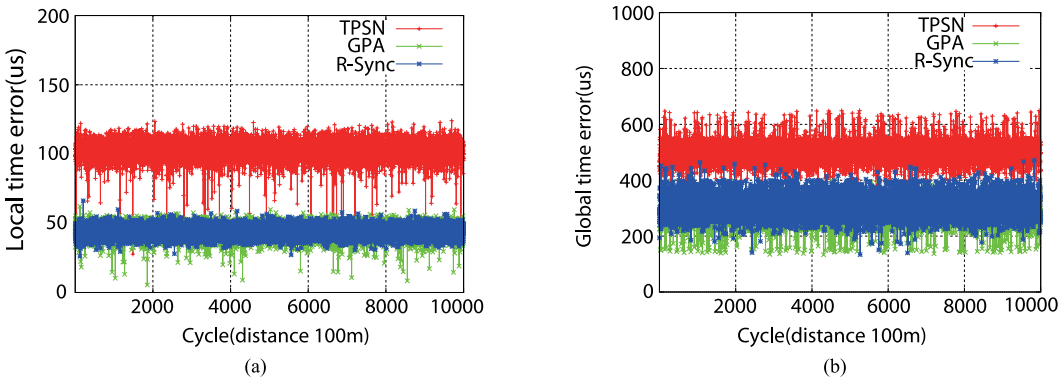


Fig. 9. Average time error. (a) Local time error. (b) Global time error.

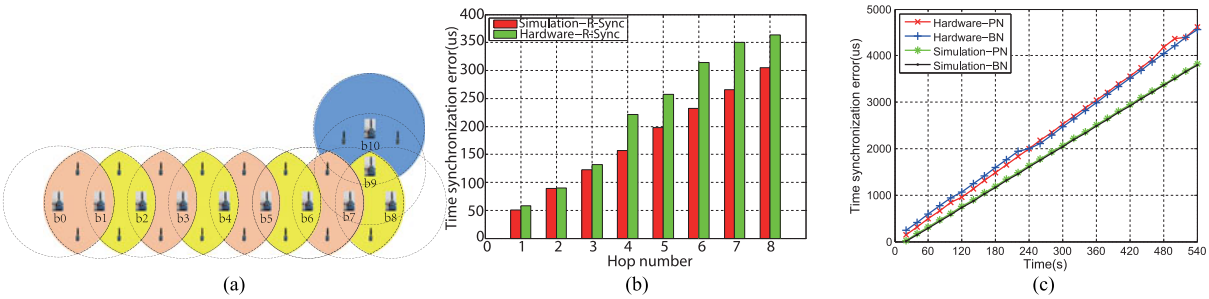


Fig. 10. Hardware experiment. (a) Topology of experiment. (b) Global time error. (c) Local PN error versus local BN error.

In Fig. 6(b), we evaluate the performance of R-Sync, TPSN, and GPA with different number of sensor nodes assuming the same simulation settings as in Fig. 6(a). The communication range of each node is set to 85 m. It can be seen that R-Sync increases slower than GPA and TPSN. Besides, for R-Sync, the number of broadcast messages is the least among the algorithms. Therefore, the energy consumption of R-Sync is lower than GPA and TPSN especially in densely connected and large-scale IIoT.

The number of broadcast messages is a crucial metric to evaluate whether an algorithm is energy-efficient because transmitting messages require maximum energy consumption during the lifetime of a node.

C. Synchronization Percentage

We compare the synchronization percentage (the proportion of synchronized nodes in all nodes) of each algorithm with different number of nodes in Fig. 7. The communication range is set to 85 m. The results show that R-Sync, GPA, and TPSN almost make all the nodes synchronized, but STETS has a poor performance despite of the synchronization percentage increasing. In addition, when the number of nodes is in the range of 200 to 300, it is inevitable that some nodes that lie in the unconnected area cannot be synchronized.

D. Energy Balance

In Fig. 8, the performance of nodes energy consumption distribution is illustrated. The number of nodes is set to 300 and the communication range is 100 m. The standard deviation of energy consumption in R-Sync is 0.017. And standard deviation of energy consumption in GPA and TPSN are 0.030 and 0.031, respectively. It can be seen that energy consumption is less and energy distribution is more balanced in R-Sync than

GPA and TPSN. For TPSN, the access time in MAC layer is varying, so that the energy distribution in TPSN is less unbalanced. In GPA, the distribution of energy consumption is more unbalanced, because no specific method of energy balance is given.

E. Local Time Error and Global Time Error

As an effective time synchronization algorithm, the synchronization accuracy is an essential factor. The average local time error of R-Sync, GPA and TPSN is shown in Fig. 9(a). The calculations of local time error follow (2) and (3). The communication range is set to 100 m. It can be observed that the local time error of R-Sync and GPA are lower than TPSN. In TPSN, only SRP is used to calculate the time error in overall the network, so the channel is more busy, especially in dense and large-scale networks. In order to avoid the packet loss, the packet must wait until it can access the channel in the MAC layer. It increases the access time, so the sender-side error increases. Although MAC-layer timestamping can reduce the sender-side error, it needs sophisticated implementation. Without MAC-layer timestamping, it has been proved that the accuracy of RRP is better than SRP in TPSN. So the time error in TPSN is larger. In Fig. 9(b), global time error is described and compared among the algorithms following (4) and (5). The standard deviation of global time error in R-Sync and GPA are 90.88 and 106.25, respectively. The accuracy of R-Sync is more stable than GPA. As global time error represents the difference of calibration time between the node and root node, the global error is larger than the local error normally. Therefore, the time error in R-Sync is smaller.

F. Implementation on Sensor Modules

The performance of R-Sync is tested on wireless hardware nodes based on ARM Cortex-M3 (88MZ100) with 32 MHz high precision crystal oscillator. The topology is shown in Fig. 10(a). The blue region indicates the pulling process. The dashed circle indicates the communication range. The mixed region of pink and yellow indicates the initial time synchronization. There are eight hops BNs in initial time synchronization (i.e., $b0$ to $b8$). The module $b9$ changes from PN to BN in the pulling process.

In order to calculate the time error, we connect a general-purpose input/output (GPIO) interface from wireless sensor devices to an external measuring-module. The measuring-module is programmed to emit a pulse in a interval. The sensor modules record the timestamps as soon as external interrupt is triggered by a pulse from measuring-module. The timestamps of each sensor module are sent to PC. And the difference between the timestamps represents the time error in PC. The experiments are repeated many times to reduce the impact of random errors. Both sensor and measuring module support the clock at the micro second level. Thus, the measured time error among sensor modules is reliable and accurate.

In Fig. 10(a), the experiment shows that the average global time error almost increases with hop number. Due to the various uncertain factors in real experiment such as unstable channel condition, hardware delay, or some interferences, the time error in experiment is a little greater than that in the simulation. With time changing, the local errors for PN and BN in simulation and hardware are compared in Fig. 10(b). Because the topology in experiment is not a densely connected network, the PN error and BN error are close to each other. It shows the clock drift in simulation and hardware. Due to temperature, the local error of hardware is almost 0 at the beginning. The drift of simulation is a little lower than that of hardware.

VI. Conclusion

In this paper, a robust time synchronization scheme for IIoT has been proposed, namely R-Sync to greatly reduce energy consumption on entire synchronization process. R-Sync eliminates the isolated nodes and makes all the nodes in networks synchronized. R-Sync also combines the SRP and RRP, so low energy consumption is required in the time synchronization. Besides, the nodes are not grouped in R-Sync. Thus, no extra energy consumption is required.

R-Sync is a trust and robust approach and synchronizes overall nodes by setting up a PT on each node at the beginning of the algorithm. The spanning tree is constructed. At the same time, PT is setting up. The order of the started PT is from the root to leaves in the spanning tree to avoid the reset of PT. After PT expires, one PN changes to BN, so that the isolated nodes are recovered. We also propose a root selection algorithm, which makes the energy balanced among nodes and extends the lifetime of sensor networks. In the simulation, we evaluate the performance of R-Sync, TPSN, GPA, and STETS on NS-2. Simulation results show that the energy consumption of R-Sync is obviously lower than GPA and TPSN, especially in densely connected and large-scale network. The percentage of synchronized node in R-Sync is greater than that in STETS. Therefore, the low energy consumption is achieved. The energy of R-Sync is more balanced than TPSN and GPA. Moreover, we use 28 sensor modules to evaluate the global error over multihops and clock drift on the ARM Cortex-M3-based testbed. The experimental results are close to that in simulation.

The major disadvantage of R-Sync is that the period of synchronization process might be prolonged due to the extra pulling process. The further research work will be focused on how to improve time synchronization accuracy with low energy consumption and shorten the synchronization period in the future. In order to achieve high accuracy, the MAC-layer timestamping will be applied in the scheme.

Appendix

Algorithm 1: Root Selection For Next Round.

1: □**Upon receiving** $< DestAddr, SrcID, Type, PrntID, RTS, STS, Level >$
2: **if** $Type = ChRoot$ and $DestAddr = ID$ **then**
3: $\quad isRootNode \leftarrow True$
4: $\quad nodeType \leftarrow BN$
5: $\quad myLevel \leftarrow 0$
6: **end if**

Algorithm 2: Preparation For Time Synchronization.

Initialization
if $isRootNode = True$ **then**
  $nodeType \leftarrow BN$
  $broadcast < 0xFFFF, ID, SetT, Null, -1, -1, 0 >$
end if
□**Upon receiving** $< DestAddr, SrcID, Type, PrntID, RTS, STS, Level >$
if $Type = SetT$ and PT is idle **then**
  set up timer PT
  $myLevel \leftarrow Level + 1$
  $broadcast < 0xFFFF, ID, SetT, Null, -1, -1, myLevel >$
end if

Algorithm 3: Initial Time Synchronization.

if $isRootNode = True$ **then**
  $broadcast < 0xFFFF, ID, Init, Null, -1, -1, 0 >$
end if
□**Upon ST expires**
$nodeType \leftarrow BN$
$T1 \leftarrow ReadCurrentTime()$
$broadcast < MyPrntID, ID, Sync, MyPrntID, -1, -1, myLevel >$
□**Upon receiving** $< DestAddr, SrcID, Type, PrntID, RTS, STS, Level >$
if $Type = Init$ and $nodeType = UN$ **then**
  cancel PT
  set up timer ST
  $MyPrntID \leftarrow SrcID$
else if $Type = Sync$ **then**
  $T2 \leftarrow ReadCurrentTime()$
  if $PrntID = MyPrntID$ and $nodeType = UN$ **then**
    $nodeType \leftarrow PN$
    cancel ST
  else if $DestAdrr = ID$ **then**
    $T3 \leftarrow ReadCurrentTime()$
    $broadcast < SrcID, ID, Ack, MyPrntID, T2, T3, myLevel >$
  end if
else if $Type = Ack$ and $SrcID = MyPrntID$ **then**
  $T4 \leftarrow ReadCurrentTime()$
  if $DestAddr = ID$ **then**
    $offset \leftarrow SRP(T1, RTS, STS, T4)$
    $localtime \leftarrow localtime + offset$
    $broadcast < DestAddr, ID, Init, MyPrntID, -1, -1, myLevel >$
  else if $nodeType = PN$ **then**
    $offset \leftarrow RRP(T2, RTS)$
    $localtime \leftarrow localtme + offset$
  end if
end if

Algorithm 4: Pulling Process.

□**Upon PT expires**
$broadcast < 0xFFFF, ID, Pulling, Null, -1, -1, myLevel >$
set up PT
□**Upon receiving** $< DestAddr, SrcID, Type, PrntID, RTS, STS, Level >$
if $Type = Pulling$ **then**
  $nodeType \leftarrow BN$
  $broadcast < 0xFFFF, ID, Init, MyPrntID, -1, -1, myLevel >$
end if

REFERENCES

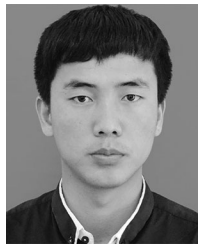
[1] L. Da Xu, W. He, and S. Li, "Internet of things in industries: A survey," *IEEE Trans. Ind. Inf.*, vol. 10, no. 4, pp. 2233–2243, Nov. 2014.
[2] C. Tsai, T. Hong, and G. Shiu, "Metaheuristics for the lifetime of WSN: A review," *IEEE Sensors J.*, vol. 16, no. 9, pp. 2812–2831, May 2016.
[3] G. Han, J. Jiang, C. Zhang, Q. T. Duong, M. Guizani, and G. Karagiannidis, "A survey on mobile anchors assisted localization in wireless sensor networks," *IEEE Commun. Surveys Tuts.*, vol. 18, no. 3, pp. 2220–2243, Jul.–Sep. 2016.
[4] S. Ji, R. Beyah, and Z. Cai, "Snapshot and continuous data collection in probabilistic wireless sensor networks," *IEEE Trans. Mobile Comput.*, vol. 13, no. 3, pp. 626–637, Mar. 2014.
[5] G. Han, C. Zhang, L. Shu, and J. J. Rodrigues, "Impacts of deployment strategies on localization performances in underwater acoustic sensor networks," *IEEE Trans. Ind. Inf.*, vol. 62, no. 3, pp. 1725–1733, Mar. 2015.
[6] T. Qiu, A. Zhao, F. Xia, W. Si, and D. O. Wu, "Rose: Robustness strategy for scale-free wireless sensor networks," *IEEE/ACM Trans. Netw.*, 2017, doi: 10.1109/TNET.2017.2713530.
[7] S. Cheng, Z. Cai, J. Li, and H. Gao, "Extracting kernel dataset from big sensory data in wireless sensor networks," *IEEE Trans. Knowl. Data Eng.*, vol. 29, no. 4, pp. 813–827, Apr. 2017.
[8] C. Tsai, C. Kang, K. Hu, and M. Chiang, "A quantum-inspired evolutionary clustering algorithm for the lifetime problem of wireless sensor network," *Int. J. Internet Technol. Secured Trans.*, vol. 6, no. 4, pp. 259–290, 2016.
[9] T. Vollmer, M. Manic, and O. Linda, "Autonomic intelligent cyber-sensor to support industrial control network awareness," *IEEE Trans. Ind. Inf.*, vol. 10, no. 2, pp. 1647–1658, May 2014.
[10] T. Qiu, K. Zheng, H. Song, M. Han, and K. Burak, "A local-optimization emergency scheduling scheme with self-recovery for smart grid," *IEEE Trans. Ind. Inf.*, 2017, doi: 10.1109/TII.2017.2715844.
[11] S. Ganeriwal, R. Kumar, and M. B. Srivastava, "Timing-sync protocol for sensor networks," in *Proc. 1st Int. Conf. Embedded Netw. Sens. Syst.*, Los Angeles, CA, USA, Nov. 5–7, 2003, pp. 138–149.
[12] D. L. Mills, "Internet time synchronization: The network time protocol," *IEEE Trans. Commun.*, vol. 39, no. 10, pp. 1482–1493, Oct. 1991.
[13] A. Mahmood, R. Exel, H. Trsek, and T. Sauter, "Clock synchronization over IEEE 802.11, a survey of methodologies and protocols," *IEEE Trans. Ind. Inf.*, vol. 13, no. 2, pp. 907–922, Apr. 2017.
[14] M. Lévesque and D. Tipper, "A survey of clock synchronization over packet-switched networks," *IEEE Commun. Surveys Tuts.*, vol. 18, no. 4, pp. 2926–2947, Oct.–Dec. 2016.
[15] M. Maróti, B. Kusy, G. Simon, and Á. Lédeczi, "The flooding time synchronization protocol," in *Proc. Int. Conf. Embedded Netw. Sens. Syst.*, Baltimore, MA, USA, Nov. 3–5, 2004, pp. 39–49.
[16] 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.
[17] J. Elson, L. Girod, and D. Estrin, "Fine-grained network time synchronization using reference broadcasts," in *Proc. Symp. Oper. Syst. Des. Implementation*, Boston, MA, USA, Dec. 9–11, 2002, pp. 147–163.
[18] I. Skog and P. Händel, "Synchronization by two-way message exchanges: Cramer-Rao bounds, approximate maximum likelihood, and offshore submarine positioning," *IEEE Trans. Signal Process.*, vol. 58, no. 4, pp. 2351–2362, Apr. 2010.

[19] K. Noh and E. Serpedin, "Pairwise broadcast clock synchronization for wireless sensor networks," in *Proc. IEEE Int. Symp. World Wireless, Mobile Multimedia Netw.*, Espoo, Finland, Jun. 18–21, 2007, pp. 1–6.
[20] K. Noh, Y. Wu, K. Qaraqe, and B. W. Suter, "Extension of pairwise broadcast clock synchronization for multicluster sensor networks," *EURASIP J. Adv. Signal Process.*, vol. 2008, no. 1, pp. 71–80, 2008.
[21] T. Qiu, L. Chi, W. Guo, and Y. Zhang, "STETS: A novel energy-efficient time synchronization scheme based on embedded networking devices," *Microprocessors Microsyst.*, vol. 39, no. 8, pp. 1285–1295, 2015.
[22] R. Lim, B. Maag, and L. Thiele, "Time-of-flight aware time synchronization for wireless embedded systems," in *Proc. Int. Conf. Embedded Wireless Syst. Netw.*, Feb. 15–17, 2016, pp. 149–158.
[23] 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.
[24] 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.
[25] P. Sommer and R. Wattenhofer, "Gradient clock synchronization in wireless sensor networks," in *Proc. Int. Conf. Inf. Process. Sens. Netw.*, San Francisco, CA, USA, Apr. 13–16, 2009, pp. 37–48.
[26] D. Djenouri, "R4Syn: Relative referenceless receiver/receiver time synchronization in wireless sensor networks," *IEEE Signal Process. Lett.*, vol. 19, no. 4, pp. 175–178, Apr. 2012.
[27] S. Jain and Y. Sharma, "Optimal performance reference broadcast synchronization (OPRBS) for time synchronization in wireless sensor networks," in *Proc. Int. Conf. Comput., Commun. Electr. Technol.*, Maruthakulam, India, Mar. 18–19, 2011, pp. 171–175.
[28] L. He, "Time synchronization based on spanning tree for wireless sensor networks," in *Proc. Int. Conf. Wireless Commun., Netw. Mobile Comput.*, Dalian, China, Oct. 12–14, 2008, pp. 1–4.
[29] 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, 2013.
[30] 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.
[31] P. Abouzar, D. G. Michelson, and M. Hamdi, "RSSI-based distributed self-localization for wireless sensor networks used in precision agriculture," *IEEE Trans. Wireless Commun.*, vol. 15, no. 10, pp. 6638–6650, Oct. 2016.

Tie Qiu (M'12–SM'16) received the M.Sc. and Ph.D. degrees in computer science from Dalian University of Technology (DUT), Dalian, China, in 2005 and 2012, respectively.

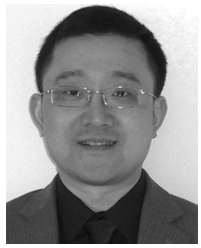
He is currently an Associate Professor in the School of Software, DUT. He was a Visiting Professor in electrical and computer engineering at Iowa State University in USA (January 2014–January 2015). He has authored/coauthored 8 books, and more than 60 scientific papers in international journals and conference proceedings, including ToN, TMC, TII, IEEE Communications, and Computer Networks. He has contributed to the development of 4 copyrighted software systems and holds 15 patents.

Dr. Qiu is an Associate Editor of IEEE ACCESS journal, *Computers and Electrical Engineering* (Elsevier), and *Human-centric Computing and Information Sciences* (Springer), an Editorial Board Member of *Ad Hoc Networks* (Elsevier) and *International Journal on AdHoc Networking Systems*, a Guest Editor of *Future Generation Computer Systems* (Elsevier). He has been a General Chair, PC Chair, Workshop Chair, Publicity Chair, Publication Chair, and TPC member of several conferences. He is a Senior Member of China Computer Federation and a Senior Member of ACM.



Yushuang Zhang (S'16) received B.E. degree in software engineering from Dalian University of Technology (DUT), Dalian, China, in 2014. He is currently working toward the Master's degree in software engineering at Dalian University of Technology.

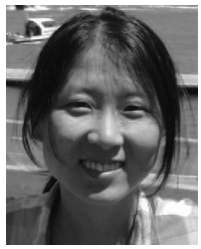
He has authored one scientific paper in an international journal. His research interests include embedded system and internet of things. He participated in the mathematical contest in modeling and won the Honorable Mention. He is an Outstanding Graduate Student of DUT and has received several scholarships in academic excellence and technology innovation.



Daji Qiao (M'04–SM'17) received the Ph.D. degree in electrical engineering systems from the University of Michigan, Ann Arbor, MI, USA, in February 2004.

He is currently an Associate Professor in the Department of Electrical and Computer Engineering, Iowa State University, Ames, IA, USA. His current research interests include protocol and algorithm innovation and implementation for wireless and sensor networks, cyber security and cloud computing, and pervasive computing applications.

Dr. Qiao is a member of ACM.



Xiaoyun Zhang received the B.E. and M.S. degree in electrical engineering from Dalian University of Technology, in 2009 and 2012, respectively. She is currently working toward the Ph.D. degree majoring in computer engineering at Iowa State University, Ames, IA, USA.

Her research interests include coverage in sensor networks and sensor network protocols.

Ms. Zhang is a student member of the IEEE Communications Society.

Mathew L. Wymore received the B.S. degree in computer engineering and the B.A. degree in performing arts from Iowa State University, Ames, IA, USA, in 2011, where he is currently working toward the Ph.D. degree in wind energy science, engineering and policy, with a co-major in computer engineering.

His research interests include sensor network protocols and applications, particularly to renewable energy.

Mr. Wymore is a student member of the IEEE Communications Society.

Arun Kumar Sangaiah received the Master of Engineering (M.E.) degree in computer science and engineering from the Government College of Engineering, Anna University, Chennai, India, and the Ph.D. degree in computer science and engineering from the VIT University, Vellore, India.

He is currently an Associate Professor in the School of Computer Science and Engineering, VIT University. His research interest includes software engineering, computational intelligence, wireless networks, bioinformatics, and embedded systems. He has authored more than 100 publications in different journals and conferences of national and international repute. His current research includes global software development, wireless ad hoc and sensor networks, machine learning, cognitive networks and advances in mobile computing and communications. Also, he has registered one Indian patent in the area of computational intelligence.

Prof. Sangaiah is an Editorial Board Member/Associate Editor of several international journals.