International Journal of Research
Available at https://edupediapublications.org/journals
e-ISSN: 2348-6848
p-ISSN: 2348-795X
Volume 05 Issue-01
January 2018

SRT Synchronization Protocol for Healthcare IoT

S.Ramakoteswararao[1], P.Srinivas[2], M.Ranjith[3]
[1]assistant Professor, Asist, Paritala,
[2]assistant Professor, Asist, Paritala, [3]assistant Professor, Asist, Paritala.

***Abstract*— *there is a lot of development in Sensor networks for healthcare IoT in recent years. For health care monitoring and proper medication sensors are attached to human bodies. Body sensors require accurate time synchronization in order to monitor health conditions. and medication usage. Nodes life time decreases due to in adequate power supply. This influences the efficiency and robustness of time synchronization protocols. The time synchronization process cannot be completed if a root node fails. To solve this problem, we present a Self-Recoverable Time Synchronization (SRTS) scheme for healthcare IoT. Here a recovery timer is set up for candidate nodes, which are dynamically elected. The candidate node whose timer expires first takes charge of selecting a new root node.*

Keywords: Healthcare IoT sensor networks, Time synchronization, Self-recovery, Two-point least-squares

1. Introduction

With the extension of body sensor technology, a number of applications based on Internet of Things (IoT) sensor networks have been developed for healthcare . These applications range from care for the elderly, patient monitoring, and ~~evaluating the condition of athletes~~. These capabilities are realized by coordinating sensor nodes using a unified time. Body sensors with limited bandwidth and power constraints are generally deployed to monitor certain body events. Hardware failures and power shortages are inevitable in healthcare environments. Owing to cost constraints, it is not feasible to repair failed nodes. Current battery technology cannot provide body sensors with sufficient energy to maintain long running times .Body sensors can efficiently harvest sunlight, radio frequency signals, thermal variations, etc. However, the harvesting of energy is typically difficult to predict, and somebody sensors fail owing to lack of power. Time synchronization efficiency is decreased by node failures. A root node failure will terminate the time synchronization process for tree-based protocols. Thus, self-recovery is a deterministic factor in evaluating the practicability of a time synchronization protocol in different areas

Furthermore, energy consumption is a crucial factor limiting network lifetime. Energy capacity is limited even in cases in which power harvesting techniques are used. A large number of applications require that sensor nodes work together to finish the same task with high accuracy. Thus, under certain energy consumption conditions, accuracy should be as high as possible. Consequently, healthcare IoT sensor networks are expected to have energy-efficient time synchronization protocols with high accuracy and effective self-recovery. The group pair selection algorithm (GPA) [is an extension for multi-cluster protocols based on the pair broadcast synchronization (PBS) protocol . First, the algorithm constructs a spanning tree, then it searches the connection status among a set of child nodes to form groups. However, GPA requires a root node in every cycle, which might not always be accessible in sensor networks. This paper pro-poses SRTS for recovering time synchronization in cases of root node failure, and improving the accuracy of PBS. The main contributions of this work can be summarized as follows.

- SRTS achieves self-recovery by dynamically employing candidate nodes that take charge of electing a root node. A recovery timer is adopted in SRTS to identify whether a root node has failed.

The electing of a root node depends on residual energy.

- Our approach applies the two-points least-squares method to PBS without changing the PBS timing chart. A MAC layer timestamp is used to further improve accuracy by eliminating error sources such as sending time delay, accessing time delay, and reception time delay. Propagation delay is the main component of the error.

- We conduct simulations using NS-2 network tools to evaluate the performance of SRTS. The simulation results show that SRTS is an energy-efficient and self-recoverable solution to the node failure problem, and provides better results compared with STETS and GPA. Furthermore, we describe the performance of SRTS integrated with PBS and TPSN in terms of accuracy and energy consumption.

The remainder of the paper is organized as follows. We describe related work in Section 2. An overview of SRTS is provided in Section 3. In Section 4, we explain the SRTS algorithm in detail. In Section 5, we conduct a simulation using NS-2 network tools to evaluate the performance of SRTS in terms of self-recovery, accuracy, and energy consumption. Finally, we discuss our conclusions.

2. Related work and problem statement

2.1. Related work

As innovative IoT applications have emerged, numerous methods have been proposed to im-prove IoT performance . Time synchronization protocols aim to provide a common timescale for sensor nodes in distributed systems. Network Time Protocol (NTP) is a well-known time synchronization protocol for distributed networks, and is widely applied in complex network environments. However, NTP is not suitable for wireless sensor networks, owing to their harsh channel conditions, complex topologies, and energy efficiency requirements. Therefore, nu-merous time synchronization protocols with high accuracy and low energy consumption have been proposed.

At present, the sender-to-receiver protocol (SRP) model and receiver-to-receiver protocol (RRP) model are widely used in time synchronization protocols. In the SRP model, a sensor node is synchronized by exchanging messages with a synchronized node. In the RRP model, a sensor node synchronizes its clock by over-hearing messages, which significantly reduces energy consumption compared with the SRP model. A classic protocol known as Timing-sync Protocol for Sensor Networks (TPSN) uses the SRP model. It constructs a spanning tree structure in the net-work and exchanges messages along each edge. PBS achieves time synchronization over the entire network by combining the SRP model and RRP model for multi-cluster sensor networks. Some of the nodes synchronize the clock by over-hearing the exchanged messages of a pair of nodes. It achieves low energy consumption by reducing the need to send messages. GPA is an extension of PBS for multi-cluster sensor networks. The whole network is divided into multiple groups by GPA. In each group, the nodes are synchronized with the group leader by combing the SRP and RRP models. Compared with PBS, the construction of groups consumes a significant amount of energy. TSBST and AMLE are two time synchronization protocols based on SRP. In RBS a selected reference node broadcasts messages, and its neighbors receive the messages in order to synchronize with each other according to the RRP model. The accuracy of RBS is high because it reduces the critical time path. OPRBS and R4Sync are also designed based on RRP. A recursive time synchronization protocol is proposed in RTSP . It periodically mea-sures the synchronization error. When drift and offset deviations are too large, nodes recursively make requests for synchronization. To improve accuracy, RTSP uses linear regression with two points to calculate the deviation. FTSP is a commonly used protocol based on flooding in time synchronization. It dynamically elects a root node and then floods its current timestamp into the network to form a tree structure. A node must wait for sufficient data to estimate the offset and skew by using least square linear regression. In general, the energy consumption is high when flooding is performed multiple times. FCSA achieves time synchronization with high accuracy by using slow-flooding. It calculates a common clock frequency for all the nodes and forces all nodes to follow the frequency. In AVTS , the authors employ an adaptive value tracking algorithm to synchronize the clock rates to a reference node. This method does not need to collect information about the reference node or keep track of the neighboring nodes. Thus, it achieves low computation and memory overhead.

Some other time synchronization protocols employ distributed time synchronization, which mainly utilizes clock information from neighboring nodes to achieve error compensation and time

synchronization. In this case, the reference nodes are not needed. Thus, the robustness and scala-bility of distributed time synchronization are better than those of protocols with reference nodes. In , the authors employ an event-based scheme based on multi-agent consensus algorithms. The time-varying threshold protocol and hold drift protocol have also been proposed to achieve time synchronization. Both of them can arrive at an error level determined by a threshold. ATS is based on a cascade of two consensus algorithms, and the main idea is to average local information to achieve a global agreement. These protocols do not focus on synchronizing to a reference node, and are therefore not suitable for networks that require a stable time.

To address the problem of node failure, some network recovery protocols have been proposed. In , relay nodes (RNs) are deployed to recover network connectivity. The 2C-SpiderWed algorithm is executed to rebuild the network with the lowest possible number of RNs. LeDiR is a localized and distributed algorithm. It utilizes existing route discovery activities to recover a network without increasing communication overhead. It achieves high recovery performance, especially in densely connected and large-scale networks. In our previous work, we proposed GMSW and STETS . GMSW achieves high robustness in centralized networks, while STETS is not suitable for the unstable sensor networks used in healthcare IoT. Thus, some improvements were made to increase the self-recovery ability and accuracy of STETS and GMSW.

2.2. Problem statement

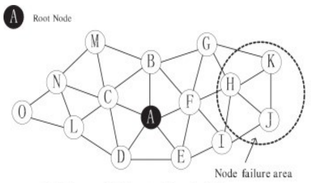
The accuracy of time synchronization protocols is a critical issue. Further, energy-efficient performance is crucial to the lifetime of networks. FCSA and FTSP achieve high accuracy by sacrificing energy efficiency. However, excessive energy consumption is not suitable for sensor networks, whose energy cannot be predicted. PBS employs a combination of SRP and RRP models in order to save energy. It calculates the offset between nodes to achieve error compensation without considering clock drift; therefore, the error increases as a result of the drift. To address the problem, SRTS utilizes the two-points least-squares method to calculate the clock drift and offset. It requires the same number of timing messages as PBS, while its synchronization accuracy is much higher than that of PBS.

Clearly, spanning-tree-based time synchronization protocols need the root node as a time ref-erence. However, the root node would fail frequently owing to unstable energy supplies and harsh environments. Thus, designing an optimal self-recoverable strategy is an important problem to address. In this paper, we divide the distribution of failed nodes into three cases.

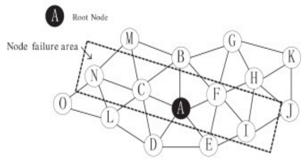
- **Case 1**: The root node is not located in the failed node area. As shown in Fig. 1a, the connectivity of the network decreases, but the synchronization protocols still work if the area is limited.

- **Case 2**: The failed node area spreads widely in the network topology and is shaped like a rectangle, in which the root node locates; an example is shown in Fig. 1b.
- **Case 3**: The root node is located in the center of the failed node area, whose radius is limited; an example is shown in Fig. 1c.

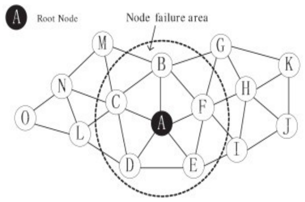
In **Case 1**, numerous nodes cannot communicate with any synchronized nodes if the failed node area is large; this seriously affects time synchronization efficiency. However, the influence caused by a few failed nodes can be ignored. For tree-based time synchronization protocols that rely on one or more certain nodes, the time synchronization process will be terminated owing to root node failure in **Case 2** and **Case 3**. Thus, a self-recoverable strategy is needed to solve the problem. In SRTS, horizontal and vertical branches in the topology are used as candidate nodes. These nodes select a new root node to achieve self-recovery. The horizontal branch that surrounds the root node is used to solve **Case 2**. The vertical branch extends from the root node to the network edge. Thus, it can successfully address **Case 3**. In addition, dynamic candidate election enhances the robustness of SRTS.



(a) Root node is out of node failure area



(b) Root node is in a rectangular node failure area



(c) Root node is in the center of a circular node failure area

3. SRTS overview

The main idea of SRTS is described in this section. SRTS employs a candidate node election strategy to achieve self-recovery. Each candidate node sets up a recovery timer (RT) to identify the failure of the root node. The candidate node election strategy of SRTS has proven to be more effective than other similar strategies. Moreover, SRTS improves accuracy by combining two-points least-squares and a MAC layer timestamp.

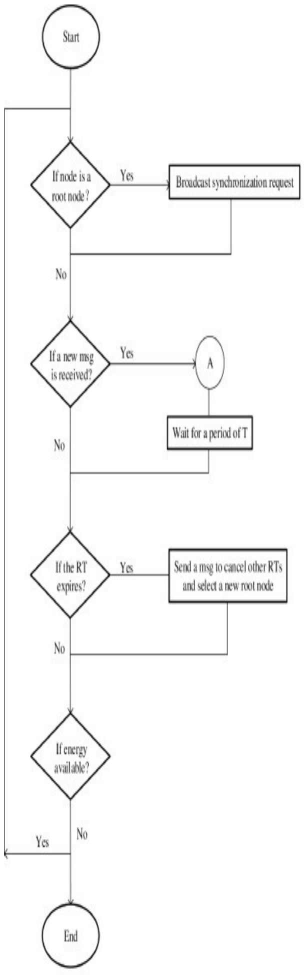
Main idea

As mentioned previously, most spanning-tree-based time synchronization protocols cannot re-cover a synchronization process that has terminated because of a failed root node. Our approach, SRTS, selects candidate nodes from horizontal and vertical branches during every time synchro-nization cycle. Nodes that are one hop away from the root node will be used to construct the horizontal branch. The vertical branch is composed of nodes whose *Seq* is equal to 1. Each can-didate node has an RT with a different initial value. By setting the RT?s minimum value to the time of the synchronization period, the expiration of a certain node's RT can reflect the failure of the root node. This node floods a *CancelT* message to cancel other RTs, and then it chooses the child node with the highest residual energy as the root node. The candidate election strategy of SRTS is more effective than the strategies used in other schemes. We start with a definition and a proposition to prove it.

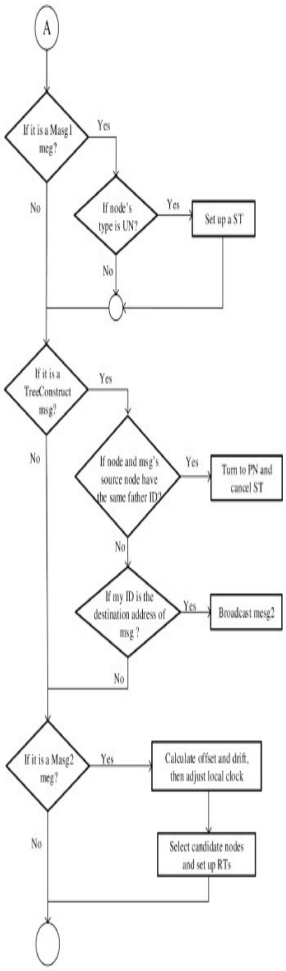
Definition : Let P denote the failure probability of all candidate nodes. p_1 represents the failure probability of a node in healthcare IoT sensor networks. If a certain candidate node has

4. Algorithm design

The self-recovery process and the SRTS time synchronization strategy are introduced in this section. Here, we use the spanning tree method described in our previous work for STETS. A synchronization timer (ST) is used to select the backbone node (BN). Some nodes are passive nodes (PNs), which only need to overhear messages without sending messages. Undefined node (UN) is the initial state of each node. Further, we elect candidate nodes from BNs and set up an RT to achieve self-recovery. We employ BNs whose level is 1 as the horizontal branch. The vertical branch is formed by BNs whose *Seq* is equal to 1.



(a) Time synchronization process of SRTS.



(b) Segment-A of the SRTS algorithm.

4.1. SRTS Process

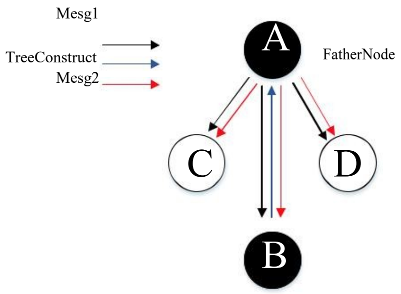
In this section, we describe our approach and use Fig. 4 as an example to describe the time synchronization process. The black nodes (node *A*, *B*) represent BNs. The white nodes (node *C*, *D*) are PNs.

In each round of time synchronization, all nodes are set to UN. Variables *Level* and *Seq* are initialized to 0. The root node is selected randomly in the first round and then *Seq* is set to 1. The process of time synchronization and candidate node election is divided into the following five phases.

Phase 1: The root node changes to a BN. It broadcasts the Mesg1 message and records the MAC layer timestamp $T1$.

Phase 2: After receiving the Mesg1 message (node *B*,*C*,*D*) from the father node (node *A*), the nodes record the MAC layer timestamp $T2$ and set node *A* as *FatherID*. They then set their *Level* to $L + 1$ (L indicates sender's level). They also set up an ST. The initial value for the timer is a random value between 5s and 10s to avoid collisions. If the ST of any UN expires (we assume that the ST of node *B* expires first), it becomes a BN and sends back a *TreeConstruct* message. The timestamp $T3$ is recorded. $T2$ (timestamp for the receipt of Mesg1) and $T3$ are carried in the *TreeConstruct* message.

Phase 3: If any UN receives the *TreeConstruct* message, and its *FatherID* is equal to *DestAddr* in the message (node *C*,*D*), it cancels the ST. When *DestAddr* in the *TreeConstruct* message is equal to the node's *ID* (node *A*), it records the timestamp $T4$. It then records the arrival sequence of the current *TreeConstruct* message if *Seq* is equal to 1. According to timestamps $T1, T4, T2$, and $T3$, we calculate the propagation delay *d* between node *A* and its child node *B* following Eq. 9, which represents the propagation delay between node *A* and its child nodes. The propagationdelay indicates the time that the message propagates wirelessly from one node to another node. Node *A* then replies with a *Mesg*2 message, which includes *d* on *PropD* filed at the moment $T5$. If the *Seq* of node *A* is equal to 1, *Mesg*2 carries the latest arrival sequence of *TreeConstruct* in the *SetSeq* field. Otherwise, *SetSeq* is set to 0



(a) Process of SRTS.

.

Phase 4: After receiving *Mesg*2 (node *B*,*C*,*D*), the nodes record the MAC layer timestamp T6.

Case 1: The *DestAddr* in the *Mesg*2 message is equal to the node's ID (node *B*). The node updates *Seq* with the *SetSeq* carried in *Mesg*2. If the current node belongs to the horizontal branch (*Level* is equal to 1) or vertical branch (*Level* > 0 and *Seq* = 1), it sets up an RT. Then the drift and offset between itself and node *A* are calculated according to two-points least-squares. Subse-quently, the node uses them to adjust the logical clock. Finally, this node forwards *Mesg*1 and the spanning tree continues to stretch.

Case 2. The *DestAddr* in the *Mesg*2 message is not equal to the node's ID (node *C*,*D*). It becomes a PN and sets *Seq* to 0, and then it adjusts the logical clock using the same method.

Phases 2 through 4 are repeated until all the nodes in the network are synchronized.

5. SIMULATION RESULTS

5.1. Simulation Setup

We used NS-2 network tools to conduct the SRTS simulation. To show the performance of SRTS, we conducted the simulation using different scales of networks and compared the re- reults with related time synchronization protocols. We analyzed the performance in terms of self-recovery, accuracy, and energy consumption. The related simulation parameters are described in Table .

5.2. Self-recovery Performance

We compared SRTS with STETS and GPA, which are both spanning-tree-based time synchronization protocols. We evaluated the self-recovery process of SRTS by running the simulation thousands of times with different parameters.

self-recovery performance is illustrated under three different scales of networks.

Self-recovery performance can be measured in terms of the percentage of synchronized nodes

Parameter	Description
Topology	Random topology
Channel	Wireless channel
Broadcast model	Two Ray Ground
MAC Layer	IEEE 802.11
Communicate radium	100*m*
-txPower	0.6*w*
-rxPower	0.3*w*
-idlePower	0.0006*w*

Simulation configuration.

For STETS and GPA, we ran the first cycle 1000 times. However, SRTS must set candidate nodes in the first cycle. Therefore, we ran the later cycles 1000 times and calculated the percentage of synchronized nodes. The related simulation parameters are shown in Table . It can be seen in Fig. that STETS and GPA linearly decrease with the increase in the percentage of failed nodes. This occurred because STETS and GPA have no strategy to handle root node failures. Obviously, SRTS can synchronize most of the nodes when the percentage of failed nodes is less than 40%. However, self-recovery performance is not satisfactory when the percentage of failed nodes is greater than 40%. This is the case because the decreased connectivity of the network greatly influences synchronization efficiency. However, SRTS can prevent time synchronization process stoppages caused by root node failure. Thus, SRTS has a better self-recovery process than

STETS and GPA. When the percentage of failed nodes is over.

5.3. Accuracy

Both PBS and TPSN are message-exchange synchronization protocols. The simulation area is set to 500*m**500*m*. The number of nodes is set to 200. We analyze the average error, error according to hop distance, and drift with time.

5.4. Energy Efficiency

The number of exchanged messages in SRTS is less than that in TPSN and similar to that in PBS. In TPSN, only SRP is used, and therefore the number of exchanged messages is large. Because SRTS combines the SRP model and RRP model in the same man-ner as PBS, the number of exchanged messages is similar. Energy consumption depends on the number of exchanged messages. Compared with TPSN, SRTS significantly reduces the number

The connectivity of a certain area increases with an increase in the number of nodes. Therefore, the number of isolated nodes that cannot communicate with any BNs decreases when the density of the network increases. As a result, SRTS achieves better performance when the number of nodes increases.

CONCLUSION

In this paper, we propose a time synchronization protocol with self-recovery and high accuracy for healthcare IoT sensor networks. Self-recovery is necessary for time synchronization protocols. When some of the body sensors in a healthcare IoT sensor network fail, the network still needs to maintain time synchronization. SRTS selects candidate nodes from the horizontal and vertical branches of the network topology. A timer is adopted to identify whether the time synchronization process is complete. A candidate node whose timer has expired takes charge of selecting a new root node, according to the residual energy. In the simulation, we evaluate self-recovery performance using NS-2 network tools by setting up different percentages of failed nodes. Simulation results show that the synchronization percentage of SRTS is clearly higher than those of GPA and TPSN.

Further, SRTS combines the two-points least-squares method and a MAC layer timestamp to improve the accuracy of PBS. Owing to the MAC layer timestamp, the main source of error is from propagation delay. Because SRTS combines SRP and RRP models in the same manner as PBS, its energy consumption is similar to that of PBS. By estimating the drift and offset according to two least squares, the accuracy of the protocol is much higher. Because PNs do not broadcast any messages, we use the propagation delay between two BNs to represent the delay between a BN and PNs. Simulation results show that SRTS achieves a high level on accuracy and balanced energy consumption.

FUTURESCOPE:

In future work, we will focus on developing a method to enhance the recovery strategy and robustness. In real-time applications, time synchronization must be performed rapidly. Therefore, we will also consider how to accelerate the time synchronization process

References

[1] D. Zhang, Z. Zhou, S. Mumtaz, J. Rodriguez, T. Sato, One integrated energy efficiency proposal for 5g iot communications, IEEE Internet of Things Journal 3 (6) (2016)

[2] J. Gubbi, R. Buyya, S. Marusic, M. Palaniswami, Internet of things (iot): A vision, architectural elements, and future directions, Future Generation Computer Systems 29 (7) (2013) 1645–1660.

[3] S. R. Islam, D. Kwak, M. H. Kabir, M. Hossain, K.-S. Kwak, The internet of things for health care: a comprehensive survey, IEEE Access 3 (2015) 678–708.

[4] A. Solanas, C. Patsakis, M. Conti, I. S. Vlachos, V. Ramos, F. Falcone, O. Postolache, P. A. Perez-martinez, R. D. Pietro, D. N. Perrea, A. Martinez-Balleste, Smart health: A context-aware health paradigm within smart cities, IEEE Communications Magazine 52 (8) (2014) 74–81.

[5] A. Zanella, N. Bui, A. Castellani, L. Vangelista, M. Zorzi, Internet of things for smart cities, IEEE Internet of Things Journal 1 (1) (2014) 22–32.
[6] Q. Li, D. Ding, M. Conti, Brain-computer interface applications: Security and privacy challenges, in: 2015 IEEE Conference on Communications and Network Security (CNS), 2015,
[7]S. Amendola, R. Lodato, S. Manzari, C. Occhiuzziet, G. Marrocc, RFID technology for IoT-based personal healthcare in smart spaces, IEEE Internet of things journal. 1(2) (2014) 144–152.
[8]A. Seyedi, B. Sikdar, Energy efficient transmission strategies for body sensor networks with energy harvesting, IEEE Trans. Commun. 58(7) (2010) 2116–2126.
[9]T. Qiu, N. Chen, K. Li, D. Qiao, Z. Fu, Heterogeneous Ad Hoc Networks: Architectures, Advances and Chal-lenges, Ad Hoc Networks. 55 (2017) 143–152.
[10]G. Han, J. Jiang, N. Bao, L. Wan, M. Guizani, Routing Protocols for Underwater Wireless Sensor Networks, IEEE Commun. Mag. 53 (11) (2015) 72–78.

Authors:

S.RAMAKOTESWARARAO
ASSISTANT PROFESSOR
AMRITASAI INSTITUTE OF SCIENCE &TECH.
PARITALA
ANDHRA PRADESH.
Email:koram419@gmail.com

P.SRINIVAS
ASSISTANT PROFESSOR
AMRITASAI INSTITUTE OF SCIENCE &TECH.
PARITALA
ANDHRA PRADESH.
Email:padalasrinu@gmail.com

M.RANJITH KUMAR
ASSISTANT PROFESSOR
AMRITASAI INSTITUTE OF SCIENCE &TECH.
PARITALA
ANDHRA PRADESH
Email:ranji469@gmail.com