Reliable Time Synchronization Protocol for Wireless Sensor Networks

Soyoung Hwang and Yunju Baek

Department of Computer Science and Engineering
Pusan National University, Busan 609-735, South Korea
{youngox,yunju}@pnu.edu

Abstract. Sensor network applications need synchronized time to the highest degree such as object tracking, consistent state updates, duplicate detection, and temporal order delivery. In addition to these domain-specific requirements, sensor network applications often rely on synchronization as typical distributed system do: for secure cryptographic schemes, coordination of future action, ordering logged events during system debugging, and so forth. This paper proposes a Reliable Time Synchronization Protocol (RTSP) for wireless sensor networks. In the proposed method, synchronization error is decreased by creating hierarchical tree with lower depth and reliability is improved by maintaining and updating information of candidate parent nodes. The RTSP reduces recovery time and communication overheads comparing to TPSN (Timing-sync Protocol for Sensor Networks) when there are topology changes owing to moving of nodes, running out of energy and physical crashes. Simulation results show that RTSP has about 10% better performance than TPSN in synchronization accuracy and the number of message in the RTSP is 10%∼35% lower than that in the TPSN when nodes are failed in the network. In case of different transmission range of nodes, the communication overhead in the RTSP is reduced up to 50% than that in the TPSN at the maximum.

1 Introduction

Recent advances in sensors, MEMS (Micro-Electro-Mechanical Systems), low power and highly integrated digital electronics and low power RF technology have allowed the construction of low-cost small sensors nodes. Such sensor nodes are generally equipped with computation and wireless communication capabilities that can form distributed wireless sensor network systems. The sensing circuitry of sensor nodes measures ambient conditions related to the environment and transforms them into an electric signal. Processing such a signal reveals some properties about objects located and/or events happening in the vicinity of the sensor. The sensor node sends such collected data, usually via radio transmitter, to a command center (sink) either directly or through a data concentration center (a gateway). A natural architecture for such collaborative distributed sensor nodes is a network with wireless links that can be formed among the sensor nodes

in an ad hoc manner [1, 2]. The most important characteristic of these sensor networks is the crucial need for energy efficiency. To facilitate easy deployment without an infrastructure, many nodes will necessarily be untethered, having only finite energy reserves from a battery [3]. These sensor networks can be used for various application areas such as health, military, home network, managing inventory, monitoring disaster areas and so on.

The main technology in wireless sensor networks includes hardware platforms and OS, low-energy consumption network protocols, time synchronization, localization, middleware, security, and applications. Especially, distributed wireless sensor networks make particularly extensive use of synchronized time: for example, to integrate a time-series of proximity detection into a velocity estimate; to measure the time-of-flight of sound for localizing its source; to distribute a beamforming array; or to suppress redundant messages by recognizing that they describe duplicate detections of the same event by different sensors. In addition to these domain-specific requirements, sensor network applications often rely on synchronization as typical distributed system do: for secure cryptographic schemes, coordination of future action, ordering logged events during system debugging, and so forth [4, 5]. Since the characteristic of sensor nodes with limited computation and energy, traditional time synchronization protocols in distributed systems can not be applied to the sensor networks directly. So existing synchronization methods are revised or new approaches are proposed to synchronize the sensor networks.

In this paper we propose reliable time synchronization protocol for wireless sensor networks. It constructs hierarchical topology in the first phase, and performs pair-wise synchronization in the second phase. In the proposed method, synchronization error is decreased by creating hierarchical tree with lower depth and reliability is improved by maintaining and updating information of candidate parent nodes. The RTSP reduces recovery time and costs - communication overhead - comparing to TPSN [6] when there are topology changes owing to moving of nodes, running out of energy and physical crashes.

The rest of this paper is organized as follows. Section 2 discusses motivation and related research in the area. In section 3, we describe proposed time synchronization protocol for wireless sensor networks. Section 4 includes the performance evaluation of the proposed method. Finally, we conclude this paper in section 5.

2 Motivation and Related Work

Since the characteristic of sensor nodes with limited computation and energy, traditional time synchronization protocols in distributed systems can not be applied to the sensor networks directly. So existing synchronization methods are revised or new approaches are proposed to synchronize the sensor networks. In the first stage of research on time synchronization in sensor networks, most approaches are based on the synchronization model such as event ordering or relative clock. These methods do not synchronize the sensor node clocks but

generate a right chronology of events or maintains relative clock of nodes. From a viewpoint of network topology, synchronization coverage is limited in a single broadcast domain; however, typical wireless sensor networks operate in areas larger than the broadcast range of a single node, so network-wide time synchronization is needed essentially. Besides, adjusting the local clock has better efficiency than maintaining relative clock since it requires more memory capacity and communication overheads. TPSN and FTSP are the representative ones which meet these requirements [7].

TPSN works in two phases: level discovery and synchronization. The aim of the first phase is to create a hierarchical topology in the network, where each node is assigned a level. Only one node is assigned level 0, the root node. In the second phase, a node of level i synchronizes to a node of level i-1. At the end of the synchronization phase, all nodes are synchronized to the root node, and network-wide synchronization achieved [6].

The goal of the FTSP is to achieve a network wide synchronization of the local clocks of the participating nodes. The assumptions in FTSP are that each node has a local clock exhibiting the typical timing errors of crystals and can communicate over an unreliable but error corrected wireless link to its neighbors. The FTSP synchronizes the time of a sender to possibly multiple receivers utilizing a single radio message time-stamped at both the sender and the receiver sides. It compensates for the relevant error sources by utilizing the concepts of MAC layer time-stamping and skew compensation with linear regression [8].

FTSP achieves robustness against node and link failures by utilizing periodic flooding of synchronization message and implicit dynamic topology update. On the other hand, TPSN does not handle dynamic topology changes; however, FTSP can not be applied generally since the synchronization accuracy in FTSP is seriously affected by the analyzed source of delays and uncertainties which are varied according to changes of the systems. The synchronization accuracy of network-wide multi-hop synchronization is a function of the construction and depth of the tree. The synchronization error is propagated hop by hop. Therefore new approaches are required to reduce the synchronization error and to manage dynamic topology changes.

3 Reliable Time Synchronization Protocol

In the following we present our scheme called *Reliable Time Synchronization Protocol* (RTSP) for wireless sensor networks. As mentioned before, the synchronization accuracy is a function of the construction and depth of the tree in network-wide multi-hop time synchronization. So it is necessary that every node is assigned a level with the shortest path from the root node to reduce synchronization error. Besides, sensor nodes can fail easily such as nodes may move, may run out of energy and may be destroyed physically. Hence a scheme is needed to manage nodes failure for synchronization accuracy and energy efficiency. We designed the protocol considering these issues.

3.1 Basic Concept

The proposed reliable time synchronization protocol works in two phases. It is assumed that nodes in the network have unique ID. But it does not need that each node is aware of the neighbor set as in the TPSN. The management of neighbor nodes is included in the operations of the protocol.

In *the first phase – hierarchical topology setup* – a hierarchical topology is created in the network. Root node with level *0* initiates topology setup. A node receives topology setup messages and assigns its level by selecting a parent with lowest level to reduce the depth of tree in the network. Other parent information is stored in candidate parent list for node failure management. Eventually every node is assigned level and a tree structure is constructed.

In *the second phase – synchronization and handling topology changes* – a node belonging to level *i* synchronizes to its parent node which is belonging to level *i-1* by exchanging time-stamp messages. When a node can not communicate with its parent, it selects another parent in the candidate list and performs synchronization. If the candidate list is empty, it request level setup to its neighbors and assigns new level, new parent and candidate parents. Candidate list is updated periodically by listening to communications of neighbors.

Following figure 1 and formula show how to obtain clock offset of a node which is used widely in many time synchronization protocols. Clock offset represents the amount to adjust the local clock to bring it into correspondence with the reference clock.

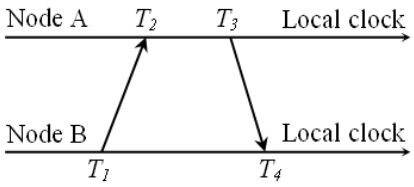


Fig. 1. Measuring delay and offset

As in the NTP, the roundtrip delay and clock offset between two nodes A and B are determined by a procedure in which timestamps are exchanged via wireless communication links between them. The procedure involves the four most recent timestamps numbered as show in figure 1. The T_1 and T_4 timestamps are determined relative to the node A clock, while the T_2 and T_3 timestamps are determined relative to the node B clock. The measured roundtrip delay δ and clock offset θ of B relative to A are given by [9]

$$\delta = (T_4 - T_1) - (T_3 - T_2)\,,\ \theta = \frac{(T_2 - T_1) + (T_3 - T_4)}{2}.$$

3.2 Protocol Description

The proposed reliable time synchronization protocol works in two phases. Operations of the protocol are detailed as follows.

The first phase: Hierarchical topology setup
In the first phase, a hierarchical topology is created in the network. This phase enforces to create a tree structure with lower depth and candidate parent list is generated to manage failure of nodes in the network.

Step 1: The root node initiates topology setup phase. Level *0* is assigned to the root node. It broadcasts topology setup message with its ID and its level.

Step 2: A node receives topology setup message during pre-defined time interval. (Root node discards this message.) It selects a parent with the lowest level number from received messages and stores other information to the candidate parent list according to the level number. Then it broadcasts topology setup message with its ID and its level.

Step 3: Each node in the network performs step 2 and eventually every node is assigned level.

Step 4: When a node does not receive topology setup message or a new node joins the network, it waits for some time to be assigned a level. If it is not assigned a level within that period, it broadcasts topology setup request message and then performs step 2 with reply of its neighbors.

The second phase: Synchronization and handling topology changes
In the second phase, a node belonging to level i synchronizes to its parent node which is belonging to level *i-1* by exchanging time-stamp messages. When a node can not communicate with its parent, it selects another parent in the candidate list and performs synchronization.

Step 1: The root node initiates synchronization phase by broadcasting synchronization message.

Step 2: On receiving synchronization message, nodes belonging to level *1* exchange time-stamp message with the root node and adjust the local clock and then broadcast synchronization message.

Step 3: On receiving synchronization message, each node belonging to level i exchanges time-stamp message with its parent and performs step 2. Eventually every node is synchronized. Once it receives a synchronization message, it discards additional messages from other upper level nodes.

Step 4: When a node can not communicate with its parent, it selects another parent in the candidate list, updates own level - if it is needed - and performs step 3. The level of its child nodes will be updated when they execute synchronization. If the candidate list is empty, it performs step 4 of the topology setup phase ahead. Candidate list can be updated periodically by listening to communications of neighbors.

When the root node fails, a node which has the lowest ID in the next level takes over it. The synchronization accuracy may be improved by utilizing the concepts of MAC layer time-stamping as in the TPSN, and the random back-off mechanism can be adapted to avoid the collision of wireless links.

4 Performance Evaluation

In order to evaluate the performance of the proposed method, we established a simulation model in the NESLsim based on the PARSEC platform. PARSEC (PARallel Simulation Environment for Complex systems) is a C-based discrete-event simulation language. It adopts the process interaction approach to discrete-event simulation [10]. In NESLsim, a sensor network is modeled as a collection of sensor nodes, a channel, and a supervising entity to create the nodes [11].

4.1 Simulation Model

N nodes are deployed in a uniformly random fashion over a sensor terrain of size 100x100. Each node has a transmission range of 28. The number of nodes, N, is varied from 100 to 300 with each increase of 50. All other parameters are arranged with the same value in the TPSN simulation. The setup includes a CSMA MAC. The radio speed is 19.2kb/s, similar to the UC Berkeley MICA Motes, and every packet has a fixed size of 128bits. A node is chosen randomly to act as the root node. The granularity of the node clocks, which is the minimum accuracy attainable, is 10s. The clock model used in simulations has been derived from the characteristics of the oscillators used in sensor nodes. The frequency drift is varied randomly with time, within the specified range, to model the temporal variations in temperature. All sensor node clocks drift independently of each other. There is an initial random offset uniformly distributed over 2 seconds among the sensor node clocks to capture the initial spatial temperature variations and the difference in the boot up times [12].

4.2 Simulation Results

All results are averaged over hundred simulation runs. The performance is compared to TPSN. The depth of a tree means the length of the path from the root node to a node with the highest level number. The synchronization error is defined as the difference between the clocks of the sensor nodes and the root node.

Figure 2 shows average depth of the tree per nodes. The synchronization accuracy of multi-hop synchronization is a function of the construction and depth of the tree. So the lower depth of tree has the better synchronization accuracy. Usually RTSP has 1∼2 lower depth than TPSN.

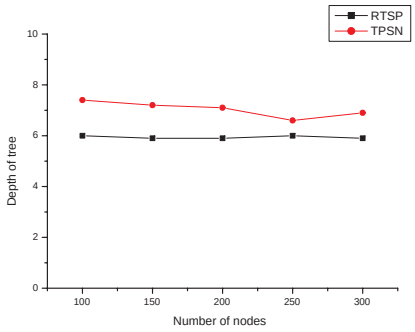
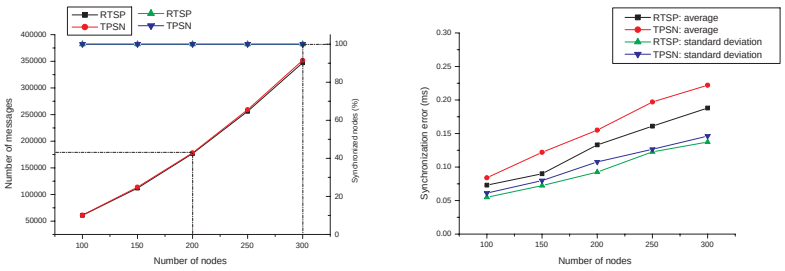


Fig. 2. Depth of tree

In figure 3, the number of messages processed during the simulation and the synchronization accuracy are presented when there is no failure of nodes. In the almost same number of messages, the RTSP has better performance in synchronization accuracy. This is the effect of the tree depth.

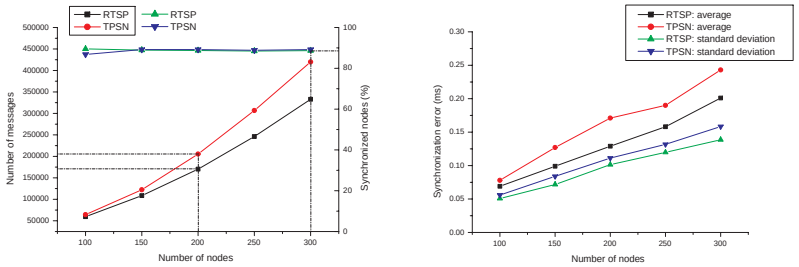


(a) Number of messages (b) Synchronization error

Fig. 3. Without failure of nodes

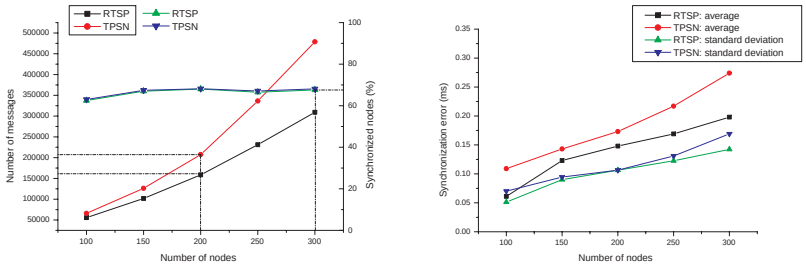
Figure 4 and figure 5 show the number of messages processed during the simulation, synchronized proportion of nodes and synchronization accuracy when

there are 10% and 30% failure of nodes respectively. In sensor networks, sensor nodes can fail easily such as nodes may move, may run out of energy and may be destroyed physically. And this failure of nodes leads to topology changes. In the simulation, node failure means that there are topology changes. In a similar proportion of synchronized nodes to the entire nodes, RTSP reduces the number of messages and shows better performance in synchronization accuracy. In wireless sensor networks, communication is one of the dominant factors in energy efficiency. Therefore communication overheads must be reduced to save energy. The RTSP reduces the number of messages and improves the synchronization accuracy by handing dynamic topology changes through the candidate parent list.



(a) Number of messages (b) Synchronization error

Fig. 4. 10% failure of nodes

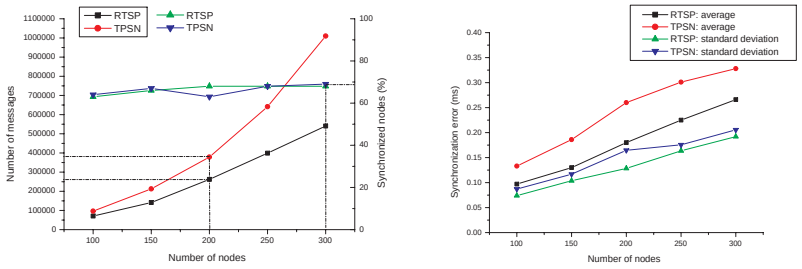


(a) Number of messages (b) Synchronization error

Fig. 5. 30% failure of nodes

As can be seen in the results, the performance of RTSP gets better than TPSN as the failure rate (topology changes) is increased. At 10% failure out of 300 nodes, the number of messages in the RTSP is 20% lower than that in the TPSN. At 30% failure out of 300 nodes, the number of messages in the RTSP is decreased by 35% comparing to that in the TPSN.

Additionally, we varied transmission range of nodes. Except transmission range, all other parameters are arranged with the same value as described in the section 4.1 Simulation Model. Each node has different transmission range from 20 to 40. Figure 6 depicts the number of messages processed during the simulation, synchronized proportion of nodes and synchronization accuracy when there is 30% failure of nodes in different transmission range. In case of different transmission range, the number of messages in the RTSP is 25%∼50% lower than that in the TPSN when there are topology changes.



(a) Number of messages (b) Synchronization error

Fig. 6. 30% failure of nodes in different transmission range

5 Conclusions

As in any distributed computer system, time synchronization is a critical issue in sensor networks. Time synchronization is a prerequisite for sensor network applications such as object tracking, consistent state updates, duplicate detection, and temporal order delivery. In addition to these domain-specific requirements, sensor network applications often rely on synchronization as typical distributed system do: for secure cryptographic schemes, coordination of future action, ordering logged events during system debugging, and so forth. But traditional time synchronization methods in distributed systems can not be applied to the sensor networks directly because of the characteristic of sensor networks with limited computation and energy.

In this paper we proposed reliable time synchronization protocol for wireless sensor networks. It constructs hierarchical topology in the first phase, and performs pair-wise synchronization and handling topology changes in the second phase. In the proposed method, synchronization error is decreased by creating hierarchical tree with lower depth and reliability is improved by maintaining and updating information of candidate parent nodes. The RTSP reduces recovery time and costs - communication overheads - comparing to TPSN when there are changes of topology. In order to evaluate the performance of the proposed method, we established a simulation model in the NESLsim based on the PARSEC platform. Simulation results shows that RTSP has about 10% better performance than TPSN in synchronization accuracy. And the number of message in the RTSP is 10%∼35% lower than that in the TPSN when there are topology changes. In case of different transmission range of nodes, the communication overhead in the RTSP is reduced up to 50% than that in the TPSN at the maximum.

Acknowledgment. This work was supported by the Regional Research Centers Program (Research Center for Logistics Information Technology), granted by the Korean Ministry of Education and Human Resources Development.

References

1. Kim, D.:Ubiquitous sensor networks, Real-Time Embedded World 19, pp.34-43, 2004.
2. Akkaya, K. Younis, M.: A survey on routing protocols for wireless sensor networks, Elsvier Ad Hoc Networks Journal 3(3), pp.325-349, 2005.
3. Elson, J., Estrin, D.: Time synchronization for wireless sensor networks, Proceedings of the IEEE International Symposium on Parallel and Distributed Processings, pp.1965-1970, 2001.
4. Elson, J., Romer, K.: Wireless Sensor Networks: A new regime for time synchronization, ACM Computer Communication Review 33(1), pp.149-154, 2003.
5. Elson, J.: Time synchronization in wireless sensor networks, Ph.D. Thesis, UCLA, 2003.
6. Ganeriwal, S. Kumar, R., Srivastava, M.B.: Timing-sync protocol for sensor networks, Proceedings of the ACM SenSys, pp.138-149, 2003.
7. Hwang, S.Y, Baek, Y.J.: A survey on time synchronization for wireless sensor networks, ESLAB Technical Report, 2004.
8. Maroti, M., Kusy, B., Simon, G., Ledeczi, A.: The flooding time synchronization protocol, Proceedings of ACM SenSys, pp.39-49, 2004.
9. Mills, D.L: Network Time Protocol (Version 3) Specification, Implementation and Analysis, RFC1305, 1992.
10. PARSEC User Manual, http://pcl.cs.ucla.edu/projects/parsec, 1999.
11. Ganeriwal, S., Tsiatsis, V., Schurgers, C., Srivastava, M.B.: NESLsim: A parsec based simulation platform for sensor networks, NESL, 2002.
12. Ganeriwal, S., Kumar, R., Adlakha, S., Srivastava, M.B.: Network-wide time synchronization in sensor networks, NESL Technical Report, 2003.