A Low Overhead Multi-hop Time-Sync Protocol for Wireless Sensor Networks

Zhijun Shang and Haibin Yu

Abstract—Time synchronization is a fundamental service for both design and application of wireless sensor networks (WSN). This paper concerns more on the aspect of overhead caused by time synchronization and presents a low overhead multi-hop time synchronization protocol (LOTS). In single hop domain, only one-way synchronization beacon is transmitted. MAC layer time stamping and linear regression methods are used to improve the accuracy of the synchronization between pair of nodes. Single hop synchronization spreads hop by hop from the time master node to the whole network. A synchronized node counts the beacons it heard to calculate a wireless connectivity metric. Only the node whose connectivity metric is higher than a certain threshold can broadcast synchronization beacon. Therefore the overhead and chances of packets collision are reduced. Simulation results show that our idea is effective when nodes are deployed densely. In addition, we argue that the possibility of synchronization error blowing up with hop distance is very small. This point is verified by experimenting on real sensor networks.

Index Terms—Multi-hop, Overhead, Sensor Networks, Time Synchronization

I. INTRODUCTION

Wireless Sensor Networks (WSN) are fast emerging as a new sensing paradigm based on the collaborative effort of large number of sensors deployed close to or inside the phenomenon to be observed. It has large potential in a wide range of applications such as military surveillance and habitat monitoring [1]-[3].

Time synchronization is a fundamental service for WSN. It involves both design and application of WSN, e.g. duty schedule among nodes, accurate time stamping of events, and collaborative signal and information processing etc.

In addition to the accuracy of the synchronization, we should also take into account the following aspects due to the unique characteristics of WSN. First, most of the application will span multi-hop distance in the field WSN deployed, so multi-hop synchronization is needed. Second, energy efficient should not be ignored by time synchronization protocol because time synchronization may last for a long period. Third, the communication traffic required by synchronization should be reduced as much as possible to avoid packet collision because nodes are generally deployed densely.

To our knowledge, most of the present time synchronization protocols or algorithms concern the accurate single hop time synchronization and how to extend to multi-hop networks. Their performances are verified on real platforms e.g. Berkeley mote using several nodes and manually arranged topology. However, the problem of overhead brought by time synchronization was not studied in detail. Since nodes are usually deployed densely in a field, how to reduce the overhead is not a trivial problem in time synchronization protocols.

LOTS reduce the overhead both in single and multi-hop synchronization phases. It uses one-way beacon message to synchronize nodes in single hop instead of round trip beacons using by NTP. In fact, it takes the advantage of integrating time synchronization with MAC layer and using liner regression to estimate the draft of local clock. The single hop synchronization spreads out level by level similar to water ripples. A synchronized node counts the beacons it heard to calculate a wireless connectivity metric. Only the node whose connectivity metric is higher than a certain threshold can broadcast synchronization beacon. Therefore the overhead and chances of packets collision are reduced. We analyze and simulate the overhead of multi-hop synchronization based on IEEE 802.11 MAC layer. In addition, we argue that the possibility of multi-hop synchronization error blowing up with hop distance is very small in practical applications. This point is verified by experimenting on real sensor networks.

This paper is organized as following: Section II reviews the exiting synchronization algorithms for WSN as well as traditional approaches. The details of single hop synchronization used by LOTS are described in section III. Section IV presents multi hop synchronization protocol and how we reduce the overhead. We analyze the simulation results in section V and give experimental results in section VI. Finally, we conclude in Section VII.

II. RELATED WORK

A. General Synchronization Techniques

We regard GPS (or other similar systems) and Network Time Protocol (NTP) as general time synchronization techniques, since they are widely used in the internet and kinds of

Manuscript received December 15, 2004. This work was supported by the National Science Foundation of China (NSFC) under Grant 60434030.
Zhijun Shang is with Shenyang Institute of Automation, China Academy of Sciences, Shenyang, 110016, P.R.China and Graduate School of the Chinese Academy of Sciences, Beijing, 100039, P.R.China (phone: 86-24-2397-0395; fax: 86-24-2397-0013; e-mail: shangzj@sia.cn).
Haibin Yu is with Shenyang Institute of Automation, China Academy of Sciences, Shenyang, 110016, P.R.China (e-mail: ybh@sia.cn).

0-7803-8812-7/05/$20.00 © 2005 IEEE
application [4][5]. However, it is claimed that they are not well suited for WSN [6].

WSN can not highly depend on GPS for two main reasons: a) the price of GPS equipment is relative high; b) GPS signals require the line-of-sight.

NTP takes the advantage of hierarchical time servers and round trip time delay measurement. A workstation can obtain high synchronization accuracy if it is connected to LAN with a time server. Or, it can synchronize with a remote time server in the internet. But the accuracy may be affected by current traffic conditions. NTP is considered as an application layer protocol. Because the strict layer structure of the protocol stack, the application layer can not know the actual time at which the packet is sent by physical layer. Furthermore, there are too much stochastic factors to one-way time delay. Therefore, NTP use round trip time delay measurement based on the assumption that the path between the client and server is symmetric. But studies show that this assumption is not true in wireless sensor networks [11].

B. Time Synchronization in Sensor Networks

Several time synchronization protocols have been proposed for WSN. We divide them into receiver-receiver style and sender-receiver style.

Reference Broadcast Synchronization (RBS) [7] proposed by Elson belongs to the receive-receiver style and is the only one. In RBS, a node periodically broadcasts wireless beacon messages to its neighbors. All the receivers use the broadcast beacon’s arrival time as the reference point to compare their local time stamp. The local timestamps are exchanged between neighboring peers in order to calculate the time offset. RBS removes non-deterministic sources from sending side. A disadvantage of this approach is the overhead caused by message exchanging between neighbors to achieve pair-wise synchronization. This overhead increases with network density [8]. An extension to RBS for time synchronization over multiple hops has also been proposed. But it is not studied in detail.

There are five synchronization approach based on sender-receiver style. They share the same idea of integrating time synchronization with MAC layer, i.e. time stamping in MAC layer. Ganeriwal et al proposed TPSN [9]. TPSN is implemented in level discovery phase and synchronization phase. In level discovery phase, each node in the sensor network is assigned a level. The node that initiates the synchronization is called the root node and is assigned level 0. The level field on each node reflects the hop count from itself to the root node. In the synchronization phase, every node exchanges time stamps with its parent in a manner similar to the NTP [5]. LTS [18] is also a lightweight protocol. It simulates centralized and distributed schemes. The former scheme is similar to TPSN. The latter means every node can ask to be synchronized on its own initiative. It is a good idea, but we only discuss the synchronization style which a time master node initiated the synchronization process. FTSP [10] also synchronized nodes over multi-hops level by level. But it uses only one-way synchronization beacon and employs linear regression function to estimate the global time. It achieves a good robustness when the only reference node failure. Su proposed DMTS that estimate the delay between pairs of node [11]. Although DMTS and FTSP is both implement in mica mote [14], there are two differences between them. First, DMTS does not use linear regression method. Second, the position that DMTS takes the timestamp is little different to that of FTSP.

In fact, LOTS proposed in this paper can be regard as an improvement version of FTSP due to taking into account the overhead.

III. SINGLE-HOP SYNCHRONIZATION

Single hop synchronization aims to synchronize nodes within the broadcast domain. It is the foundation of multi-hop synchronization. If a node synchronizes itself with a remote node directly, the uncertainty of time delay over multi-hop will be very large due to the low data transfer rate. As for MICA2 mote, the valid data rate is 19.2 bits/s when use the default configuration. The time nondeterministic in one hop caused by CSMA/CA will be in the magnitude of tens milliseconds. Additionally, the topology is not as fixed as internet and the channel is easy to be interfered comparing with wired networks.

The nondeterministic sources in sync-critical path has been decomposed well in literatures [7][12]. The biggest nondeterministic is the media access time caused by CSMA/CA mechanism in MAC layer. For the low data rate of typical WSN radio stack, the time slot used by MAC layer is in the magnitude of millisecond. Thus, it is the popular approach to time stamp the synchronization beacon in MAC layer (except for RBS). The basic concept of CSMA/CA is that the stack monitors the physical channel constantly, if the channel is idle, the stack send the packet after a random back off time. If the MAC layer can record the “actual” time when the packet is transmitted, it writes this time stamp in a field of the packet before the packet is sent. In this way, the delays caused by preparing and transmitting the beacon are excluded in the sync-critical path. The only factor will affect the time delay is the propagation time. But it can be ignored due the speed of radio propagation in the air.

A single time stamp would be sufficient to eliminate the two nodes’ clock offset if their clock crystals had exactly the same frequency during the running time. Unfortunately, this is not the case of the real crystal used by the inexpensive node. Fortunately, the offset between the two clocks changes in a linear fashion provided the short term stability of the clocks. We refer to the linear regression approach to estimate the clock offset between two nodes [10]. The data pair \((f_{\text{prior}}, \Delta t)\) was used to find a regression function, where \(\Delta t\) is defined to be the difference between local recorded time stamp, \(f_{\text{local}}\), and global time stamp written in the received beacon. In case node A synchronizes with node B, when node A collected enough pairs of time stamp, node A could the following function:
\[ T_{\text{local}} = \rho \times T_{\text{local}} + \Delta T, \]  

(1)

where \( \rho \) indicate the drift of node A’s clock compared with node B’s clock; \( \Delta T \) is the local time offset between node A and B. We should note that both \( \rho \) and \( \Delta T \) will be update when a new sync-beacon is receive by node A. When a global time stamp is required by applications, node A may convert the local time stamp to the global time stamp using function (1). Since the concern of this paper is the overhead, we have not analyzed the linear regression in detail. The interested reader can read reference [10] and [16].

The advantage of the above single hop synchronization is the low overhead comparing with RBS and TPSN. After collecting enough pairs of time stamp, the node uses only unidirectional sync-beacon to update the regression table. So, it reduces the energy consumption of transmitting round trip packets and the chances of packet collision.

IV. MULTI-HOP SYNCHRONIZATION

A. Definitions

Although some localization experiments only require pairwise synchronization [13], we envision that most applications will need time synchronization beyond single hop distance. So we extend the single hop synchronization approach in the above section to multi-hop distance. All the nodes synchronize with one or limited number of “reference node” which are called time master node (TM). A typical TM is the base station installed with GPS equipment. We consider the network to have just one TM in this paper.

We use the following definitions.

1) Local level: It is the hop distance from a TM measured by the node itself.

2) Local sequence: The sequence number of last received synchronization beacon. Only TM can increment its local sequence number.

3) Sync-beacon: It is a short packet that synchronizes the receiver and sender. It has the field of local sequence, local level and local time stamp, which is record in MAC layer (also see Fig.7).

B. Synchronizing Nodes Level by Level

LOTS does not depend on additional service to create and keep the hierarchical structure. The globe time spreads out as water ripples. We share the same idea of TPSN and FTSN. TM broadcasts the sync-beacon in the period \( P \). The level of TM is assigned to 0. For any ordinary node, when it receives a sync-beacon, it records the time of arrival. After receiving the whole packet, it compares the received level with its local level. If the received level is smaller than the local level it will set its local level to received level plus 1 and put the data pair to its regression table.

In order to spread synchronization process over multi-hops, the receiver will generate and broadcast sync-beacon again \( \Delta T \) after wait for a period of time \( T \) which is derived from the following formula,

\[ T = 0.5T + T_{\text{local}}, \]  

(2)

where \( T_{\text{local}} \) is uniformly chosen from \((0,0.5T)\) to avoid the contention in medium access. The receiver records the local sequence in the packet and updates the local sequence number. If it receives other beacons with the same sequence, it discards the packets. When a node collected enough time-stamp pairs, it can calculate a regression function.

C. Analysis of Multi-hop Synchronization Error

Since the sync process spreads level by level, does the synchronization error increase with the hop distance linearly? We think it is possible that synchronization error accumulates and blows up only if all the nodes’ clock drifts in the same direction. But we argue that the possibility is small in practice. Generally, the drift of crystal’s frequency can be modeled by Gaussian distribution. Notice that in LOTS protocol, a node adds the time stamp pair which it heard firstly to the regression table. The broadcasting order of upstream nodes is decided by the back off time chosen from the uniform distribution, i.e., the parent with which a node synchronizes is not fixed. Therefore, the positive and negative error will cancel each other as the synchronization spreading in ripple style. We will verify our assumption by experiments later in section VI.

D. Reducing the Overhead

Although the sequence number prevents nodes from adding more time stamp pairs to its regression table, redundant beacon are still broadcasted. Let us consider the simple topology in Fig.1-(a). All nodes lie uniformly in a line. The distance between them is equal to the valid commutation radius \( r \). If the link quality is good, the synchronization process can spread out to the node with level 2 and no redundant beacons be broadcasted. But the density in Fig.1-(a) is too sparse. In actual applications, the density of nodes will be higher than that in Fig.1-(a). One node has neighbor nodes whose distance to its upstream nodes is less than \( r \). This situation is illustrated in Fig.1-(b). Redundant beacons appear when synchronization beacons spread out from TM, i.e., node E in level 2 can hear beacons broadcasted by node D and node B. Because node D, B, and E lie within the same broadcast domain, node E can still receive the beacon if node B broadcast beacons whereas node D not. We call node D is the “redundant” node. We envision that there are many “redundant” nodes when the scale of network is larger. It is required by the sparse network that every node in same level broadcast synchronization beacons to ensure that all the nodes in the networks are synchronized. We call this “All Beast” scheme. It is used by FTSP. As for the densely deployed networks, the overhead caused by “All Beast” scheme would be large. So, we proposed an alternative scheme -- “Selected Beast” to depress the “redundant” nodes.

Intuitively, we select the nodes lie at edge of every level. The idea is similar to that of TPSN. But this problem was neglected by FTSP.
V. SIMULATION

We call LOTS a overhead protocol for two reasons. First, it only uses one-way sync-beacon to do pair wise synchronization. If the number of nodes in a broadcast domain is $n$, RBS required at least $n$ message be broadcasted. The reference node have to transmit 1 message as the reference point. The other nodes will each transmit a message contained local time stamp to exchange with other nodes except for the reference node. The other approaches summarized in section II, such as TPSN and LTS, measuring round trip time so they need 2 messages to be transmitted for pair wise synchronization. We think the approaches which only use one-way message for pair wise synchronization take the lower overhead, such as PESN and DMTS. Second, LOTS reduces the overhead when this pair wise synchronization is extended to multi hop. We do not give a quantized analysis here because it is obvious, see above section. In stead, we give the results of simulations in this section.

A. Simulation Setup

We use the standard IEEE 802.11 as the MAC protocol to simulate the overhead of time synchronization. Because our main goal is to study the overhead aspect, we have not implemented the details of single hop synchronization in our model. In stead, we just simulate the sync-beacon propagation in multi-hop. A node is assumed to be synchronized once if it received a sync-beacon. The true multi-hop synchronization accuracy will be testified on MICA2 platform in section VI.

We use 100 nodes and adjust deployment area to change the node density. In our simulation TM initially broadcast the sync-beacon every 30 second. The “advanced” node re-generates a sync-beacon and broadcast it after 10 seconds plus a random number of second chosen from 1 to 10 seconds. The metrics we used to analysis the results are as following.

1) Percentage of synchronized nodes: This metric tells us how many nodes are synchronized in any synchronization period $P$. It is record every 30 seconds. During a polling period, if a node received a valid beacon in the past 30 seconds, it is regarded as a synchronized node.

2) Packet throughput: The total packets sent and received in the whole network during a polling period. It is the indicator of the overhead caused by time synchronization because no other type of packets is used in the simulation.

B. Results and Analysis

**Result 1:** Initially, we use the networks with density approximate to 10 and put the TM in the center and a corner respectively. The simulation runs 30 minutes. We observed that the throughput is similar but the percentage is different in Fig.3. If TM is located at the center, the average percentage is 91%. However, the values decrease to 82% if we put TM at the corner. Although more nodes could be synchronized when TM is at the center of the area, we envision that TM used to be a
base station located at the corner. So we just use the corner style topology in the subsequent scenarios.

![Fig. 3 Comparing the percentage of synchronized nodes in case TM at center and corner of the area. The nodes density is 10.](image)

**Result 2:** We run simulation at density of 10 using all Bcast scheme and selected Bcast scheme respectively. The results see Fig. 4 and Fig. 5. In Fig. 5, the throughput is reduced remarkably by selected Bcast scheme. The average of throughput of all broadcast is 26.4 packets/second and that of selected Bcast scheme is 11.3 packets/second. The throughput is reduced by 57.2%. Notice that the peak before 500 seconds is caused by the stage of all Bcast when nodes collect the beacons to calculate the connectivity metric. In Fig. 5, the percentage of synchronized node in all Bcast scheme is 82.6% and that of selected Bcast scheme is 73.2%.

**Result 3:** In order to compare the performance of LOTS, we run the simulation in three different densities: 10, 20, and 44. In Fig. 6, we compare the reduction of the throughput and the decrease of synced node percentage. We can see that LOTS, which employ the selected Bcast scheme, reduce the packet collision and energy consumption at the price of a little decrease of the percentage of synchronized nodes comparing with the all Bcast scheme.

![Fig. 4 The throughput of all Bcast and selected Bcast scheme. TM is at the corner and nodes density is 10.](image)

![Fig. 5 Percentage of all broadcast and selected broadcast style. TM is at the corner and nodes density is 10.](image)

![Fig. 6 The reduction of synchronized nodes and the decrease of the throughput in different densities. The notation d denotes to the density and th to the threshold for connectivity metric.](image)

**VI. EXPERIMENT**

In section V, we just analyze the change of synchronization accuracy with the hop distance. But the real single hop time synchronization is not implemented in the simulation. We did experiments to testify how the synchronization accuracy changes over multi-hop distance.

At present, the most widely used platform in WSN research community is Berkeley MOTE [14]. We choose MICA2 MOTE, which evolved from Berkeley MOTE and developed by Crossbow Company [15]. The operation system employed by MICA2 is Berkeley TinyOS. The external clock used by MICA2 mote a 32.768 KHz crystal, hence the time resolution is 30.5 microsecond, called jiffy in [10].

![Fig. 7 Time stamping in MAC layer](image)
We use time-stamping program contributed by Vanderbilt University [10]. The process of packet transmission in TinyOS MAC layer is illustrated in Fig.7. The radio stack constantly monitors the radio channel. If the channel state is idle, the stack can start to transmit packet. Firstly 18 bytes preamble and 2 bytes Sync code, then the active message formatted in TinyOS. In FTSP time-stamping program, the sender obtain the current global time after it sends the last Sync byte and updates the time synchronization message with the time stamp record. On the receiver side, the receiver records the local time stamp at the arrival of the last Sync byte and corrects the received time stamp by a constant depending on the bit offset.

We enforced a link topology in the program download. In the experiment, the node with id 0 initializes broadcast sync beacon in the period of 30 seconds. The node with ID n only receives sync-beacon from nodes with ID n - 1, i.e. every node is synchronized with its parent (the nodes with lower ID) directly. All nodes synced with the node 0. A polling node broadcast the polling message every 30 seconds. All nodes that receive the polling message will send it local timestamp and estimated global timestamp to the base station. The base station is connected to a PC, which can display the results.

We run the experiment 20 times and rearrange the nodes’ ID among the nodes we used every time. The results show that the accuracy of the synchronization does not decrease with the level distance. It seems to be within the range from 3 to 5 jiffies in our experiment. The results see Fig.8.

**Fig. 8. The average synchronization error does not blow up with the hop distance from TM.**

VII. CONCLUSION

Multi-hop time synchronization is a fundamental service for wireless sensor networks. But the overhead caused by time synchronization is a nontrivial problem for the energy constraints. In this paper we propose LOTS to keep the networks time synchronized with a single time master. In single hop domain, LOTS use only unidirectional transmission of sync-beacon; in multi-hop domain, we compare the overhead of “All Beast” and “Selected Beast” scheme by simulations. Results show that the later one can reduce the overhead significantly. The price is that the percentage of synchronized nodes in every synchronization round is reduced a little. In fact, it makes a worthy tradeoff between the overhead and the accuracy of time synchronization.

REFERENCES