2009 International Symposium on Nonlinear Theory and its Applications
NOLTA'09, Sapporo, Japan, October 18-21, 2009

Independent Two Skew Estimations Improves Accuracy and Robustness of Synchronization in Wireless Sensor Networks

Kazuhiro Yonekura†, Hisa-Aki Tanaka†, and Kenta Shinohara†

†Department of Electronic Engineering, The University of Electro-Communications (UEC)
1-5-1 Choufugaoka, Choufu-shi, Tokyo 182-8585 Japan
Email: yonekura@synchro2.ee.uec.ac.jp

Abstract—Among several timing synchronization protocols proposed so far, flooding time synchronization protocol (FTSP) is now accepted as one of the most energy-efficient and the highest-precision synchronization protocols for wireless sensor networks. FTSP is based on a periodic flooding of time synchronization messages in a tree-like network of sensor nodes. Therefore, when the root node is lost and reelected, a certain amount of degradation in synchronization precision is temporally observed. Also, in our experiments with Mica2Dot platform we found that the synchronization precision can still be improved by using less synchronization points than in the original FTSP. In this study, we analyze the underlying mechanism of the above observations, and as a result, the following two components are turned out to be problematic; (a) a long term frequency modulation of crystal oscillators in each sensor node, and (b) the resulting error in the estimation of the clock drift (i.e. skew) in FTSP. Based on this analysis, we propose an effective improvement of FTSP with two sets of independent estimations for synchronization offsets between sender and receiver nodes, for robust and better synchronization. Systematic experiments with Mica2Dot platform are carried out, where higher synchronization precision is constantly obtained and dynamical robustness in root reelection process is also enhanced by this improvement.

1. Introduction and motivation of this study

Wireless sensor networks (WSN) are generally consisted of small, cheap, and resource limited devices which communicate each other and sense the environment. They are now utilized for distributed sensing purpose; indoor, outdoor, and even in-body monitoring applications, mobile commerce, and so on. Researches are ongoing to realize thousands or even millions of sensor nodes communicating each other to accomplish certain sensing tasks in non ideal, harsh environments. In such WSN, robust and precise time synchronization often becomes essential because time synchronization is a basis for consistent distributed sensing and control.

Among several timing synchronization protocols proposed so far [1, 2], flooding time synchronization protocol (FTSP) [3] is now accepted as one of the highest-precision synchronization protocols for WSN. FTSP is, by its definition, based on the periodic flooding of time synchronization messages from the root node (i.e. the top of the tree-like network of sensor nodes) as shown in Fig. 1, where time synchronization messages are carried unidirectionally from a sender node to (possibly multiple) receiver nodes.

Although this tree-like network can be implicitly updated, temporal degradation of synchronization precision is inevitable in dynamic environments due to link or node failure. In particular, when the root node is lost and reelected, we observe synchronization precision temporary degrades considerably (in Sec 3.2). On the other hand, when the network is static and stable, we discover that the synchronization precision can still be improved by using less synchronization points, as opposed to the original FTSP (in Sec 4.1). To obtain higher synchronization precision even in harsh environments including root reelection, we then analyze the underlying mechanism of the above observations. As a result, the following two components in WSN are turned out to be problematic; (a) long term frequency modulation of crystal oscillators in each sensor node, and (b) the resulting error in the estimation of the clock drift (i.e. skew) estimation in FTSP. Based on this analysis, we propose an effective improvement of FTSP with two sets of independent estimations for synchronization offsets between sender and receiver nodes, for robust and better synchronization. Systematic experiments with Mica2Dot[4] platform are carried out, where higher synchronization precision is constantly obtained and dynamical robustness in root reelection process is also enhanced by this improvement.

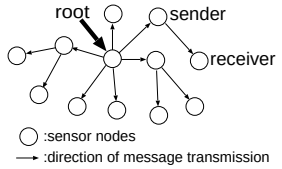


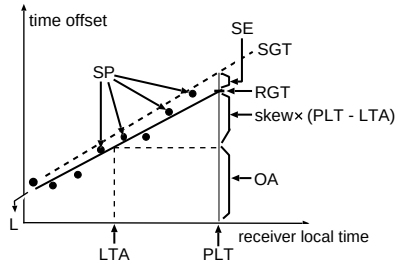
Figure 1: Network configuration in FTSP

2. Synchronization mechanism of FTSP

In FTSP, the root node has the global time which serves as a standard time in the network. This global time can be obtained from GPS or other reliable sources. On the other hand, all other nodes maintain the following two time informations; (a) their own local time, and (b) their global time,

respectively. The local time is constantly clocked by the crystal oscillators in each node. While, the global time for each node (except for the root node) is an virtual estimated global time from its local time. Therefore, a small error remains between this virtual global time in each node and the real global time in the root node, although experimental results in [3] show that this error can be as small as in a micro second order. In FTSP, estimation of the global time from the local time is based on a set of past time messages from a sender node. This estimation handles the following two components together; (a) compensation of static delays in message transmissions, and (b) compensation of dynamic clock drifts between sender and receiver nodes.

The compensation of delays is successfully handled thanks to a detailed analysis of message transmissions in [3]. On the contrary, the clock drift is due to the intrinsic frequency mismatch of clocks between sender and receiver nodes, and also due to the temporal instability of clocks. Therefore, compensation for the clock drift requires a bit harder dynamical estimation of the clock frequency. For this purpose, FTSP cautiously employs the linear regression with a set of previous eight synchronization points as shown in Fig. 2. At each synchronization point in Fig. 2, the receiver node gets a combination of the sender time stamp and the corresponding local time of the receiver node at message reception. The sender time stamp is the virtual global time estimated by the sender node at message transmission. If the time offset between the sender global time and the receiver local time is correctly estimated, receiver nodes can predict the sender's global time from their local times and synchronize to its global time. Therefore, estimation of this offset becomes a key to achieve a high-precision synchronization. In FTSP [3], this estimation is realized as follows. First, skew is assumed to be a constant in a short time span, and it is obtained by the increasing (or decreasing) rate of the linear regression line L best approximating a set of past time offsets in Fig.2. Also, by the average of these past offsets, delays in message transmissions can be implicitly compensated [3] and the receiver's global time is obtained as
receiver global time = receiver local time + offset average + skew × (receiver local time - local time average) as shown in Fig.2.



time offet = sender global time - receiver local time
RGT : receiver global time, PLT : present local time
SP : synchronization point, L : linear regression line
LTA : local time average, OA : offset average
SGT : sender global time, SE : synchronization error

Figure 2: Estimation of receiver global time from past time offsets

3. Analysis of synchronization precision

We analyze here the synchronization precision obtained by FTSP both for (i) static network environments and in (ii) transient, dynamic network environments due to root lost and root reelection. Systematic experiments with Mica2Dot platform are carried out both for the cases.

3.1. Static network environments

As explained in Section 2, skew estimation plays a central role in FTSP. Then, we investigated the actual variation of skew by the following experiments.

[Experiment 1] Eight Mica2Dot motes operate in a single-hop network, until the battery becomes empty at some node. Until then, the estimated skew variation at one of the node is recorded as in Fig. 3. Temperature and humidity is maintained to be constant throughout the experiment.

Then, we observe the skew continues to increase (or decrease) in a certain range of time. At first, we suspected that this observation is due to a possible failure of this particular node. However, it turned out not to be the case, because the same experiment using any other Mica2Dot motes leads to the same observation.

Besides this observations, the following experiment provides some information on the amount of skew variation between time synchronization points.

[Experiment 2] Eight Mica2Dot motes operate in a single-hop network for about two hours. The initial root node is removed from the network, 2451 seconds later from the beginning of this experiment. And, again, the secondary elected root is removed, 4830 seconds later from the beginning. Temperature and humidity is maintained to be constant throughout the experiment.

Figure 4 (a), (b), and (c) show temporal variation of the skew estimation in three receiver nodes (; node a, node b, node c), respectively. It is noted that other four receiver nodes show a similar temporal variation of the skew. From Fig. 4, we observe the skew keeps increasing (or decreasing) in any node for about 2,500 seconds until the root node is removed. From this observation and the observation in the above experiment 1, it is reasonable to assume the long term skew increase (or decrease) is inevitable, as far as Mica2Dot motes are employed in FTSP, possibly due to the characteristics of its clock.

From such long term skew increase (or decrease), one can infer the temporal variation of time offsets as shown in Fig. 5. By the definition of skew in FTSP, data set of time offsets should exhibit a convex curve (or concave curve) if the skew continues to decrease (or increase). For instance, in Fig. 4 (a), (b), and (c), node a, node b, and node c respectively around 0.02×10^{-4}, 0.02×10^{-4}, and 0.05×10^{-5} decrease in skew for 210 seconds. From this fact, we can roughly estimate the resulting synchronization error in node a (node b) and node c respectively by $(0.02 \times 10^{-4} \times 210^{-1})[s^{-1}] \times (105+30)[s] \times 30[s] \sim 39[\mu s]$,

and $(0.05 \times 10^{-5} \times 210^{-1})[s^{-1}] \times (105 + 30)[s] \times 30[s] \sim 9.6[\mu s]$.

It is noted that this estimation is a direct consequence from the linear regression line of past eight offsets (for 210[s]) on a convex time offset curve at every 30 seconds synchronization point, as shown in Fig. 5. Then around 30 micro seconds maximum synchronization error should emerge during this experiment.

Besides this prediction, as Fig. 5 schematically shows, we expect that smaller synchronization error is obtained with less time offsets in linear regression.

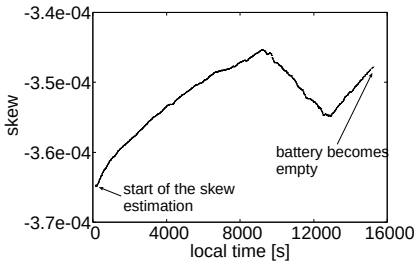


Figure 3: Long term skew variation (experiment 1)

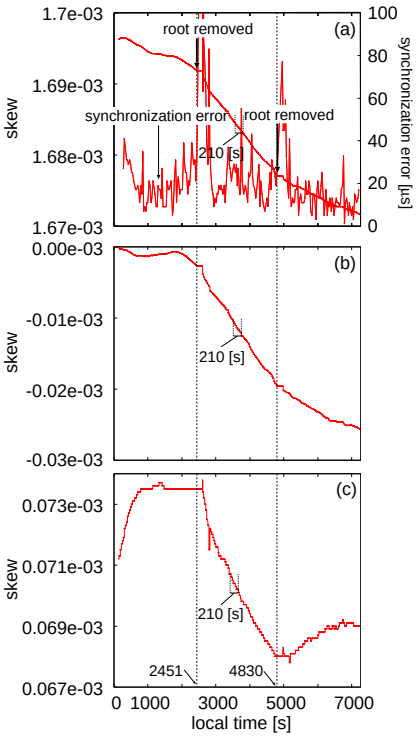


Figure 4: Simultaneous skew variation due to root reelection

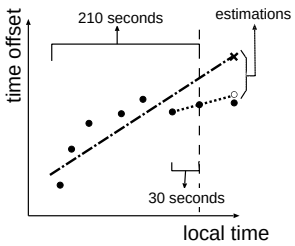


Figure 5: Synchronization error due to long term skew variation

3.2. Case of transient, dynamic network environments

In harsh, dynamical network environment, node or link failure is not inevitable. In FTSP, the most serious, such failure is lost of the root node and resulting root node reelection. Concerning this problem, FTSP already gives various scheme[3]. We approach this serious problem in terms of skew variation. In the previous subsection, we considered the case of stable environments. However, in the same experiment 2, a large amount of skew variation due to root node reelection is observed in a relatively short time span. Figure 6 shows a typical example of such large skew variation. Two data sets are plotted respectively for the case of linear regression with 8 time offsets, and the case of that with 2 time offsets. In both of them, after the root is removed, the skew remains constant until the next root is newly reelected. This is because in FTSP skew estimation stops and holds the previous value until the new root appears. Then, after this, skew estimation starts again using previous offsets and the current offsets together. Naturally, this mixed state lasts until the old offsets are lost one by one. Then, during these transients certain amount of skew variations emerge. As opposed to the case shown in Fig. 5, linear regression with 2 offsets leads to a much worse skew estimation, compared with the 8 offsets linear regression.

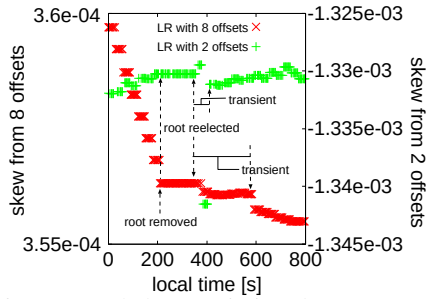


Figure 6: Temporal skew variation due to root reelection

4. Design of improved FTSP and its performance verification

A natural improvement of FTSP with Mica2Dot platform is proposed as follows, and its effectiveness is verified through systematic experiments both for stable network and for unstable network environments due to root reelection.

4.1. Basic design of improved FTSP

In section III, we analyzed how a certain amount of synchronization error emerges due to long term modulations the following of the Mica2Dot clock. Based on this insight, systematic experiments are carried out to measure the synchronization error obtained from different numbers of consecutive time offsets used for linear regression.

[Experiment 3] Twenty Mica2Dot motes operate in a single-hop network for 30 minutes. In each experiment, the number of time offsets in all nodes is set to 2, 4, 6, 8 and 10, respectively. The batteries are newly replaced and the same

motes are used for every measurement to keep conditions the same. Temperature and humidity is maintained to be constant throughout the experiment.

In this experiment setup, we also measured the temporal synchronization error due to root node reelection, for 2, 8, and 10 time offsets cases, respectively, and we measured the synchronization error from the proposed improvement of FTSP, which is explained below in detail.

Figure 7 and 8 show the synchronization error obtained in this experiment, respectively for stable networks and for unstable networks due to root reelection. Each data point (×) comes from independent trials, and for each trial, average of synchronization errors from all pairs of nodes and the maximum of them are plotted in Fig. 7 and 8, respectively.

From these results, it is clearly observed that (i) For stable networks, smaller (average) synchronization error is realized for less number of time offsets in skew estimation. (ii) In contrast, smaller (maximum) synchronization error is realized for more time offsets for unstable networks due to root reelection.

Then, why not using these two skew estimations in combination? Naturally, we expect that FTSP (with Mica2Dot) can be improved by the two independent skew estimations respectively using past two time offsets when the network is stable, and ten offsets only when the network is unstable. In other words, this improvement uses different these skew estimations according to network conditions, and always realizes better than the original FTSP. The algorithm of this mechanism (for receiver nodes) is shown in Fig. 9.

rootID contains the ID of the root known by the sender message, and myRootID contains the ID of the root known by the node. 'myRootID == rootID' implies the sender node to this particular receiver node is the same as before, and 'myRootID > rootID' implies there is some other sender node to this receiver node, in Fig. 9.

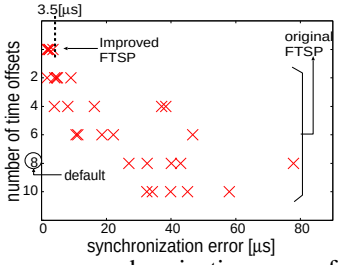


Figure 7: Average synchronization error from different skew estimations (stable network case)

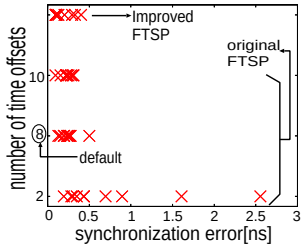


Figure 8: Maximum synchronization error from different skew estimations (unstable network case)

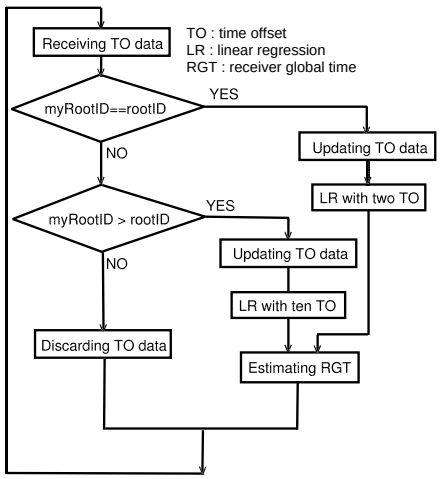


Figure 9: Proposed algorithm for skew estimation

4.2. Performance verification of the proposed method

To verify the performance of this improved FTSP, systematic experiments are carried out as described in the experiment 3. Figures 7 and 8 show the average synchronization error for the stable network case, and the maximum synchronization error for the unstable network case, respectively. Both results show that the improved FTSP out performs the original FTSP in its synchronization precision, which verifies the effectiveness of this improvement even in the unstable network.

5. Conclusion

Through systematic experiments, we measured and analyzed how synchronization error is controlled by tuning the skew estimation scheme in FTSP. The proposed improvement is simple and straightforward, and its effectiveness is explained and verified with consistent experimental results.

Although our improvement is motivated by particular FTSP environments of this study, the essential point of this improvement can be applied to other skew estimation purposes in distributed, harsh, dynamic environments.

More experiments including multi-hop networks and/or using other clock oscillators are required to explore further improvement and more applications.

References

[1] J. Elson, L. Girod, and D. Estrin, "Fine-Grained Network Time Synchronization using Reference Broadcasts," *Proc. 5th Symposium on Operating Systems Design and Implementation*, pp. 147 – 163, Boston, Massachusetts, 2002.

[2] S. Ganeriwal, R. Kumar, and M. B. Srivastava, "Timing-sync Protocol for Sensor Networks," *Proc. 1st ACM Conference on Embedded Network Sensor Systems*, pp. 138 – 149, Los Angeles, California, 2003.

[3] M. Maroti, B. Kusy, G. Simon, and A. Ledeczi, "The Flooding Time Synchronization Protocol," *Proc. 2nd ACM Conference on Embedded Networked Sensor Systems*, pp. 39 – 49, Baltimore, Maryland, 2004.

[4] Mica2Dot:http://www.xbow.com/Support/Support_pdf_files/MPR-MIB_Series_Users_Manual.pdf