NOLTA, IEICE

Paper

Better clock synchronization from simultaneous two skew estimations

Hisa-Aki Tanaka [1 a)], *Youjie Ouyang* [1], *Yoji Yabe* [1], *Isao Nishikawa* [1], *and Kazuki Nakada* [1,2]

[1] *Graduate School of Informatics and Engineering, The University of Electro-Communications, 1-5-1 Chofugaoka, Chofu, Tokyo 182-8585, Japan*

[2] *The author is currently with Graduate School of Information Sciences, Hiroshima City University, 3-4-1 Ozuka-Higashi, Asaminami-ku, Hiroshima 731-3194, Japan*

[a)] *htanaka@uec.ac.jp*

Received December 6, 2015; Revised June 6, 2016; Published October 1, 2016

Abstract: An improvement for clock synchronization in wireless sensor networks (WSNs) is presented, which is obtained by analyzing a temporal frequency variation observed in internal clock circuits. Clock synchronization is an essential building component in WSNs for distributed sensing. Flooding time synchronization protocol (FTSP) is one of the highest-precision synchronization protocols for WSNs, which has been implemented on certain WSN testbeds. We carry out systematic experiments of FTSP with a Mica2Dot testbed to understand how synchronization precision is affected by a dynamic frequency variation in the clock circuit with a button battery. Our observations clarify that the following two elements are essential for better clock synchronization; (i) a short-term frequency variation in the clock circuit, and (ii) the resulting error in clock drift (*i.e.*, skew) estimation from the linear regression in FTSP. Based on these findings, we propose a simplistic improvement for robust and more precise clock synchronization, utilizing two sets of simultaneous estimations of skew between sender and receiver nodes. Through systematic experiments and analysis, we confirm this improvement realizes a higher synchronization precision in a stable network environment, while it maintains robustness of time synchronization even in a worst case of unstable networks.

Key Words: clock synchronization, crystal oscillators, wireless sensor network, skew estimation, FTSP

1. Introduction and motivation of this study

Recent advances in electronics and wireless communication technology have made sensing and communicating devices smaller, cheaper, and more low-power than before. Such resource-limited devices (sensor nodes) construct wireless sensor networks (WSNs) for a wide range of distributed sensing purpose; indoor, outdoor, and even in-body monitoring applications. In such WSNs, precise clock synchronization often becomes essential because clock synchronization is a basis for consistent distributed sensing and control. An interesting application is shown in an early study [1] for instance.

Nonlinear Theory and Its Applications, IEICE, vol. 7, no. 4, pp. 548–556 DOI: 10.1587/nolta.7.548

Several distributed algorithms for clock synchronization have been proposed so far [2–4]. Moreover, a large survey on synchronization protocols in WSNs is now available [5]. Among them, flooding time synchronization protocol (FTSP) [6] and PulseSync [7] are one of the standard energy-efficient and high-precision synchronization algorithms for WSNs. Whereas ad-hoc mobile networks with a dense network topology are assumed in the IEEE 802.11 timing synchronization function (TSF) [4], both FTSP and PulseSync construct an adaptive tree-like network of sensor nodes, in which synchronization messages [6] (*i.e.*, time stamping packets) are periodically sent from the root node down to multiple receiver nodes, as exemplified in Fig. 1. As a result, undesirable frequent collisions [8] of synchronization messages (*i.e.*, timing beacons [4]) are avoided. From these synchronization messages each receiver node effectively estimates its clock skew (*i.e.,* the speed of clock drift, defined in Section 2) by using a linear regression algorithm in FTSP. A high-precision time synchronization is then realized even for resource-limited sensor nodes such as Mica2Dot motes [9].

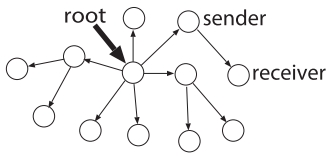

Fig. 1. Tree-like network topology in FTSP.

Incidentally, recent studies [10–12] have investigated an intrinsic characteristics of the clock circuit (*i.e.*, the crystal oscillator) in a dynamic environment. They have demonstrated that the clock skew is dynamically compensated according to the working temperature. In the present study, as opposed to such previous work [10–12], we focus on an intrinsic dynamic characteristics of the clock circuit under a static working temperature and humidity. We then clarify yet another mechanism that degrades the synchronization precision in FTSP even under a static working environment. The main contributions of this paper is twofold. First, we analyze the underlying mechanism of our experimental results for FTSP on Mica2Dot motes, which clarifies that the following two elements play essential roles for better clock synchronization; (i) a short-term frequency modulation in the clock circuit, and (ii) the associated error in clock skew estimation from the linear regression in FTSP. Second, based on these insights, we propose a simplistic improvement of FTSP by utilizing two simultaneous estimations of the clock skew. With this improvement, the synchronization precision is observed to increase by a factor of more than several times than the original FTSP, while robustness of time synchronization is maintained even under a nonstatic, unstable network environment. We note that our improvement can be useful even in a dynamic environment, since this can be incorporated into the results for such an environment [10–12].

The remainder of this paper is organized as follows. In Section 2, we review FTSP to explain how receiver nodes synchronize to the root node. In Section 3, we analyze an underlying mechanism for the synchronization precision degradation. In Section 4, we propose an effective improvement of FTSP, and provide systematic performance evaluations through comparative experiments. Finally, the discussion and conclusions are presented in Section 5.

2. Synchronization mechanism in FTSP

In FTSP, the root node provides a real global time which serves as the standard time in the network. This global time can be obtained from GPS or other reliable sources. On the other hand, all other receiver nodes independently maintain the following two sets of times; (i) their own local times, clocked by the crystal oscillators in each node, and (ii) their virtual global times. The virtual global time in each receiver node is estimated from its local time by referencing synchronization messages from a sender node. In the estimation of the virtual global time, delays in synchronization message transmissions and clock drift between sender and receiver nodes are involved. The compensation of delays is successfully handled through a detailed analysis of message transmissions for Mica2 motes [6]. Compensation of the rate of clock drift (*i.e.,* skew), on the other hand, is carried out by using linear

regression for intrinsic frequency mismatch of clocks in sender and receiver nodes, and for temporal instability of clocks, simultaneously.

Clock drift is estimated in FTSP in the following way. The original FTSP employs a linear regression (LR) in a time window of previous eight synchronization messages (SMs). These SMs are periodically sent at each synchronization point (SP) in a fixed resynchronization period denoted by T, as shown in Fig. 2. In usual, the value of T is set to 30 seconds [6]. At each SP, the receiver node obtains the combination of the sender's time stamp (carried on SMs) and the corresponding local time in the receiver node at message reception; the sender's time stamp is given as a virtual global time estimated in the sender node (or can be the absolute global time if the sender is the root node) at message transmission. Hence, if the time offset (TO) between the sender's (virtual) global time (SGT) and the receiver's present local time (PLT) is correctly estimated, a receiver node knows the sender's global time from its local time.

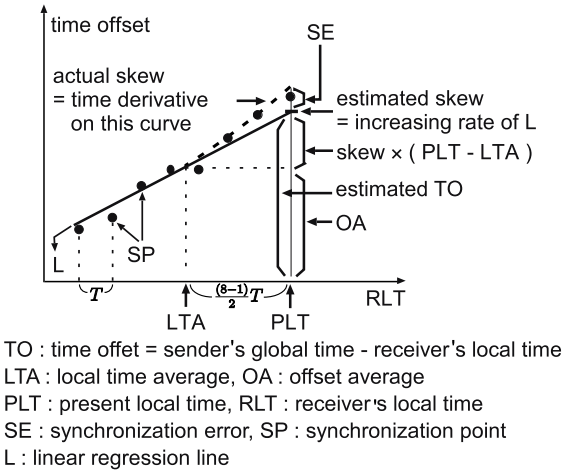

Fig. 2. Schematic of skew estimation in FTSP.

In the estimation scheme described above, FTSP implicitly sets the following two assumptions; (i) the skew is a constant, and it is estimated from the increasing (or decreasing) rate of the LR line L in Fig. 2, which best interpolates a set of past (eight) time offsets, and (ii) by using the average of these past offsets, delays in message transmissions can be compensated. Therefore, FTSP estimates the receiver's global time (RGT) as shown in Fig. 2 :

$$\begin{aligned} \mathrm{RGT} &= \text{present local time (PLT)} + \text{estimated time offset} \\ &= \mathrm{PLT} \ + \text{offset average (OA)} \ + \text{skew} \times (\mathrm{PLT} - \text{local time average (LTA)}). \end{aligned} \tag{1}$$

However, if the actual skew temporally increases or decreases in some nodes, the estimation (1) of RGT should involve a certain amount of estimation error. In fact, such a temporal variation of skew is a major factor of the synchronization error (SE) as observed in Section 3. To evaluate the amount of such SEs, here we assume that a skew increases at a constant speed in a short period. We then denote the time offset of node A's clock (*i.e.*, SGT) w.r.t. node B's clock (*i.e.*, RGT) as $\theta_B^A(t)$, in which t is the local time in node B [10, 12]. Similarly, the skew $\alpha_B^A(t)$ is defined as $\alpha_B^A(t) \equiv d\theta_B^A(t)/dt$ [10, 12]. Then, we consider the situation where the skew increases (or decreases) at a constant speed Δskew. Namely,

$$\frac{d\alpha_B^A(t)}{dt} = \Delta\mathrm{skew} \equiv C. \tag{2}$$

As shown in Fig. 2 the synchronization error (SE) is defined as $\mathrm{SE} \equiv \mathrm{SGT} - \mathrm{RGT}$. SGT and RGT at a local time t_1 (of node B) are respectively given by their definitions :

$$\mathrm{SGT}(t_1) \approx t_1 + \mathrm{OA} + \int_{t_0}^{t_0+T\ (=t_1)} \alpha_B^A(t)\ dt, \tag{3a}$$

$$\mathrm{RGT}(t_1) = t_1 + \mathrm{OA} + \alpha_B^A(t_0) \cdot (t_1 - t_0), \tag{3b}$$

in which t_0 and t_1 respectively correspond to LTA and PLT in Fig. 2. Subtracting Eq. (3b) from Eq. (3a), we obtain SE at t_1, as

$$\begin{aligned}\mathrm{SE}(t_1) &\approx \int_{t_0}^{t_1} \alpha_B^A(t)\ dt - \alpha_B^A(t_0)\cdot(t_1-t_0) = \int_{t_0}^{t_1} \alpha_B^A(t) - \alpha_B^A(t_0)\ dt \\ &= \int_{t_0}^{t_1} C\cdot(t-t_0)\ dt = \frac{C}{2}(t_1-t_0)^2 = \frac{C}{2}\left(\frac{n-1}{2}T\right)^2, \end{aligned} \tag{4}$$

since $\alpha_B^A(t) = \alpha_B^A(t_0) + C\cdot(t-t_0)$ from Eq. (2) and $t_1 - t_0 = \frac{(n-1)}{2}T$, where n is the number of data samples (from SMs) used in the linear regression, as shown in Fig. 2. This formula (4) turns out to be informative in analyzing SE in Section 3.

3. Analysis of synchronization error

To see how skew and SE behave in a real world environment, we observe and analyze them, firstly in a stable and static network environment (*i.e., Experiment 1*), and then in an unstable, transient network environment due to root lost and root reelection (*i.e., Experiment 2*); for both environments in a Mica2Dot testbed, we carry out systematic experiments using Mica2Dot motes which are shipped with two clock oscillators; a 32KHz crystal oscillator[1] and a more fine-grained 4MHz crystal oscillator. Here, we employ this 4MHz oscillator as the clock, because this 4MHz oscillator is used in the experiments in [6] as well as in the experiments of RBS [2] and TPSN [3].

As explained in Section 2, skew estimation plays an essential role in FTSP. Therefore, firstly, we observe temporal variations of skew in the following experiment.

Experiment 1: Eight sensor nodes (Mica2Dot motes) form a single-hop network in the high-performance incubator (Type KCL-2000A, Eyela Co.), in which external radio waves are naturally shielded, as shown in Figs. 3(a) and 3(b). Until the battery becomes empty in some node of the network, temporal variations of the estimated skews are recorded in all nodes; one of typical data from a randomly chosen node is shown in Fig. 4(a). To exclude undesired changes in the environments, temperature and humidity is maintained exactly at 25°C and 30%RH throughout the experiment.

In this experiment, we find the estimated skew alternatively increases and decreases at almost constant rates as shown in Fig. 4(a). At first, we suspected that this is caused by a possible failure in this particular node. Then, we repeated the same experiments by using other motes, but all experiments lead to similar observations. Next, we carried out a control experiment to *Experiment 1*, in which all button batteries on Mica2Dot motes are replaced with a common stable voltage source (PMM35-1.2DU, KIKUSUI), to exclude a possible voltage instability from the button battery. In this control experiment, as we expect, almost no temporal variation is observed in the estimated skew for all the nodes, and such an observed constant skew is of the same order to the time-averaged skew in the same node with a button battery. (Data is not shown due to space limit.) Therefore, the observed temporal variation of skew can be attributed to some internal mechanism of the battery and/or the clock circuit. In any case this temporal skew variation should affect SE for each node.

Then, to examine the relationship between SE and the window size of linear regression in FTSP, we repeated *Experiment 1* by increasing the resynchronization period T from its original 30 [s] upto 300 [s]. For each value of T, five trials are carried out for $30T$ [s]. The time-averaged SE in one randomly chosen node, which is further averaged over five trials, is shown in Fig. 4(b). A quadratic increase of SE is observed w.r.t. T. This observation is consistent to the result of $\mathrm{SE} \simeq T^2$ in Eq. (4)[2] under a constant rate of skew variations.

Based on the above observations, in the following experiment we further investigate a more direct relationship between temporal variations of skew and the resulting SEs.

Experiment 2: Eight Mica2Dot motes form a single-hop network for about two hours. The initial root node is removed from the network at 2451 seconds later from the beginning of the experiment.

[1] The experiments in [11] are carried out with a citizen CMR200T oscillator running at 32KHz in TelosB motes.
[2] Note C in Eq. (4) is naturally time-averaged during this observation.

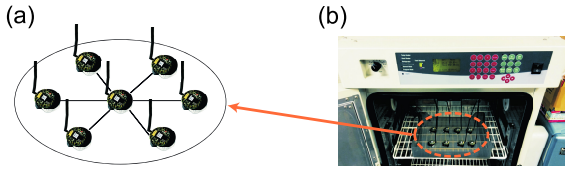

Fig. 3. Experimental setup. (a) A single-hop network in a Mica2Dot testbed. (b) All experiments are carried out at a constant temperature and humidity in the incubator.

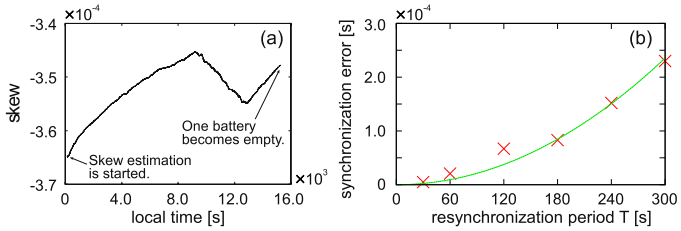

Fig. 4. Short-term skew variations observed in Experiment 1. (a) Temporal variation in an estimated skew in one node. (b) Averaged synchronization errors in one node for several different resynchronization periods. The curve represents the quadratic function of T fitting data points $\times$.

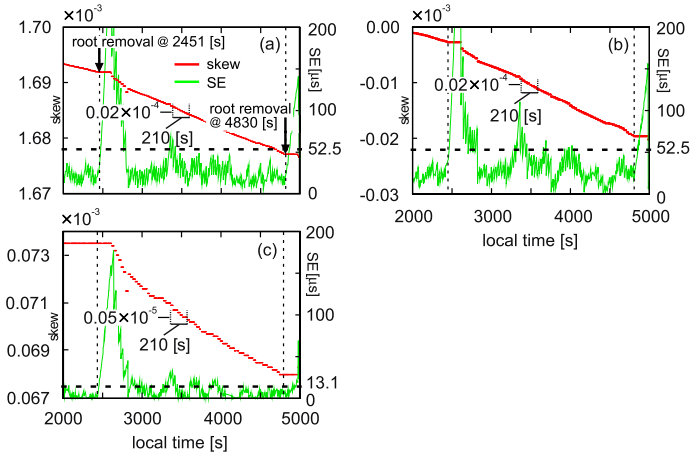

Fig. 5. Temporal skew variation and the associated synchronization error (SE). (a) Data from node A. (b) Data from node B. (c) Data from node C.

Then, the newly reelected root node is removed at 4830 seconds later from the beginning. Temperature and humidity is maintained exactly at 25°C and 30%RH throughout the experiment.

Figures 5(a), (b), and (c) respectively shows a temporal variation of the estimated skew and the associated SE in three randomly chosen receiver nodes; node A, node B, and node C. We note that four other receiver nodes show a similar temporal variation of the skew. In Fig. 5, we observe the skew keeps decreasing at an almost constant rate respectively in node A, B, and C, for about 2,400 [s] between two events of root removal at 2451 seconds and 4830 seconds respectively, and the rate of skew variation is altered before and after the root removal and reelection.

More precisely, we observe that node A, node B, and node C respectively shows about 0.02×10^{-4}, 0.02×10^{-4}, and 0.05×10^{-5} decrease in skew during a time laps of 210 [s], at around the local time = 3500 seconds. By using the data and Eq. (4), we can roughly predict the resulting SEs in node A; $\mathrm{SE}(t = 3500) = \frac{1}{2}\ (0.02 \times 10^{-4}/210)\ (210/2)^2 = 52.5$ [μs], since $C = 0.02 \times 10^{-4}/210$ and $t_1 - t_0 = 210/2$ in Eq. (4). In the same way, SEs in node B and node C are respectively obtained as 52.5 [μs] and 13.1 [μs]. Note that these predictions show a fair agreement to the observed SEs in Fig. 5. We also note that fine fluctuations of SEs in Fig. 5 are smaller than the associated time-averaged SEs, and this validates the meaning of the time-averaged SEs shown in Fig. 4(b).

From the above analysis of temporal skew variations observed in *Experiment 1* and *Experiment 2*, we infer that the estimated skew and the associated time offsets are influenced by the time window

size as follows[3]. Firstly, the instantaneous skew is the time derivative of the time offsets curve (*i.e.*, Eq. (2)). And, as shown in Fig. 2, data sets of time offsets should be on a concave curve (or a convex curve) if the skew continues to increase (or decrease). Therefore, as illustrated in Fig. 6, smaller SEs are expected for less number of data points (*i.e.*, shorter time window size) in linear regression (*i.e.*, Eq. (4)), on average, in the environment of *Experiment 1* and *Experiment 2*. This prediction is verified in control experiments for a stable static network environment in Section 4.

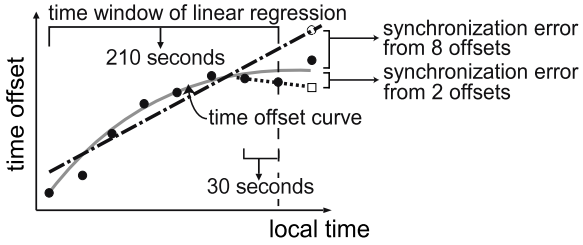

Fig. 6. Time offsets resulting from a constantly decreasing skew. The symbols •, ○, and □ respectively represent data points, the estimated time offset from eight data points, and the one from two data points.

In contrast to the static network environment considered above, in a harsh, dynamical network environment a failure in a node or a link is inevitable. One of the worst case scenarios of such a failure in FTSP is a lost of the root node and the resulting root node reelection. As observed in Fig. 5 of *Experiment 2*, right after root removal a large SE lasts for about several hundreds seconds. To ensure the quality of synchronization within a certain level, we analyze such large SEs in detail by varying the number of time offsets in linear regressions, in the next Section.

4. Design of improved FTSP and its performance evaluation

Based on the analysis of SE in Section 3, we now propose a simplistic improvement of FTSP in a Mica2Dot testbed. We evaluate its performance through systematic experiments in a stable network environment as well as in an unstable network environment due to root reelection.

First, to verify the prediction in Section 3, we investigate the relation between SE and the number of time offsets (*i.e.,* data samples) in linear regression (LR), in the following experiment.

Experiment 3: Twenty Mica2Dot motes form a single-hop network for 30 minutes. For each trial, the number of time offsets in LR is set to 2, 4, 6, 8, and 10, respectively. For the fairness in the conditions, the same motes and new button batteries are used. Temperature and humidity is maintained exactly at 25°C and 30%RH throughout the experiment. In this *Experiment 3*, we measured (i) skew variations and the associated SEs for each time offsets number, and (ii) SEs for our improved FTSP, both of which are obtained in the stable as well as in the unstable network environment as in *Experiment 1* and *Experiment 2*.

Firstly, we compare temporal skew variations for different time offsets numbers (*i.e.*, numbers of data samples) in detail. Figure 7 illustrates typical skew variations in *Experiment 3*; two data sets are respectively obtained for the case of eight time offsets (× in Fig. 7), and for the case of two time offsets (+ in Fig. 7). In both cases, right after the root removal at the local time (LT) = 1190 seconds, the observed skew becomes constant until another root is newly elected at LT = 1360 seconds. This is because in FTSP the value of skew estimation remains unchanged until a new root is reelected. After this root reelection, skew estimation starts again by joining previous time offsets (before root lost) to the current time offsets (after root reelection). This transient state lasts until the previous time offsets are completely lost from the moving time window of LR; it lasts for about $30 \times (8-1) = 210$ [s] in the eight offsets case, and for about $30 \times (2-1) = 30$ [s] in the two offsets case, as observed in Fig. 7. During these transients, larger instantaneous skew variations emerge as the number of time offsets becomes smaller; we observe big jumps in the estimated skew for the minimum two time offsets case (+) in Fig. 7. This induces large SE (not shown here, but similar to Fig. 5) according to

[3]In this particular testbed, the time offsets data itself was not directly available from the motes. Therefore, we observe the time offsets indirectly from the skew estimations available in each mote.

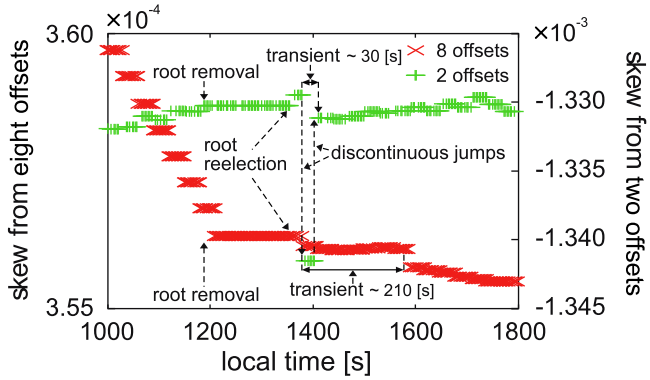

Fig. 7. Temporal skew variations due to root removal and root reelection. Data points × and + respectively show the case of eight time offsets and the case of two time offsets.

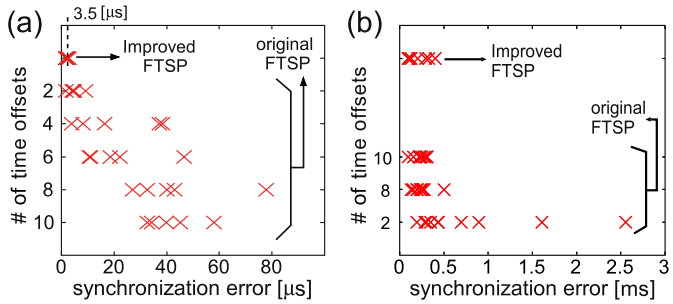

Fig. 8. Comparisons of synchronization errors from the original FTSP and the improved FTSP. (a) Average synchronization error in a stable network environment. (b) Maximum synchronization error in an unstable network environment.

Eq. (1). Namely, in an unstable network, LR with minimum two offsets no longer produces a better skew estimation compared with the original eight offsets LR, and this is consistent to the insight from Fig. 6.

Next, we consider how time offsets number influences the amount of SE on average. Figures 8(a) and 8(b) show the SEs for several numbers of time offsets, obtained in *Experiment 3*, respectively for stable networks and for unstable networks involving root reelection. Figure 8(a) shows a result of five trials in *Experiment 3* for the stable network environment. Each data point (×) represents the time-averaged SE over 30 minutes. On the other hand, Fig. 8(b) shows a result of ten trials in *Experiment 3* for the unstable network environment. Each data point represents the instantaneous, maximum SE observed in each trial. Note we have already observed this sort of large instantaneous SEs of a few milliseconds in Fig. 7. From these data, we observe; (i) for stable networks, the time-averaged SE is smaller for a less time offsets number in LR, and (ii) in contrast, the maximum, instantaneous SE is smaller for a larger time offsets number, for unstable networks involving root reelection. Note, these observations are consistent to the result shown in Fig. 4(b) and the observations in Fig. 7.

From these observations, we obtain a reasonable idea of simple improvement for better clock synchronization; FTSP (with Mica2Dot) is improved by using simultaneous two skew estimations. Namely, the minimum two time offsets in LR are used for reducing SEs on average in a stable environment. And, at the same time eight (or ten) time offsets are used for suppressing large, instantaneous SEs in an unstable environment, which are less than around 0.5 [ms] as observed in Fig. 8(b). As the results, from these independent two estimates of skew, our improved FTSP should provide less than around 0.5 [ms] SEs within around 30 [s] transients in the worst case scenario mentioned above (as far as the network instability does not occur so frequently).

This improvement in the skew estimation algorithm is realized on each receiver node. We illustrate the algorithm in Mica2Dot motes in Fig. 9. The algorithm of Fig. 9 involves two branches; in the first branch 'myrootID == rootID' means the sender node (to this particular receiver node) is the

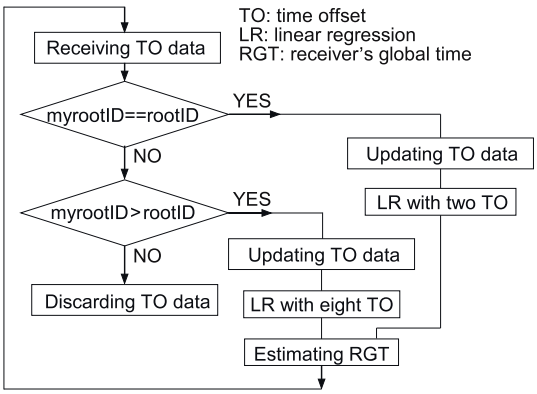

Fig. 9. Outline of the proposed algorithm for skew estimation.

same as before. Namely, it checks whether the network is the same as before (*i.e.*, stable). On the other hand, in the second branch 'myrootID > rootID' means the network is NOT the same as before (*i.e.*, unstable). We note that these branches effectively take care of all kinds of temporal link lost and discommunication between nodes, including the worst case of root reelection. In addition, to evaluate the performance of our improvement, we conducted systematic experiments under the same conditions in *Experiment 3*. In Figs. 8(a) and 8(b) the average SE and the maximum SE from our improved FTSP are shown for the stable network case, and for the unstable network case, respectively. Both results verify that the improved FTSP outperforms the original FTSP in its synchronization precision. Finally, we note the power consumption in the improved FTSP is nearly the same to the one in the original FTSP, because we observe the battery life time is nearly the same for both cases throughout the experiments.

5. Conclusion

Systematic control experiments identified and clarified yet another mechanism, in which synchronization error is generated from temporal frequency (skew) variations in Mica2Dot motes. Based on this insight, a simplistic improvement of FTSP is proposed, which is easily implemented with small computational and communication costs. Its effectiveness is confirmed through comparative experiments. Although our findings and improvement have been obtained in a particular testbed with Mica2Dot motes, its essential idea can be useful for any other environments if skew is estimated from linear regression. One candidate on this line is a smaller and cheaper (injection-locked) CMOS ring oscillator as a GHz clock circuit which should exhibit a certain amount of temporal frequency variation [13, 14]. In addition to this, as mentioned in Section 1, recent results in [10–12] have enabled temperature-aware clock skew estimations by utilizing the information of temperature dependence of the oscillator frequency. These self-calibration methods should be naturally incorporated into our proposed scheme. Therefore, our scheme here can possibly be useful in dynamic environments of working temperature if the temperature dependence of the oscillator frequency is available. More experiments including multi-hop networks and/or using other kind of clock oscillators would lead to further improvements of clock synchronization and their applications.

References

[1] M. Maroti, G. Simon, A. Ledeczi, and J. Sztipanovits, "Shooter localization in urban terrain," *IEEE Computer*, vol. 37, no. 8, pp. 60–61, 2004.

[2] J. Elson, L. Girod, and D. Estrin, "Fine-grained network time synchronization using reference broadcasts," in *Proc. 5th Symposium on Operating Systems Design and Implementation*, pp. 147–163, 2002.

[3] S. Ganeriwal, R. Kumar, and M.B. Srivastava, "Timing-sync protocol for sensor networks," in *Proc. 1st ACM Conference on Embedded Network Sensor Systems*, pp. 138–149, 2003.

[4] J.-H. Chiang and T. Chiueh, "Accurate clock synchronization for IEEE 802.11-based multi-hop wireless networks," *IEEE ICNP'09*, pp. 11–20, 2009.

[5] A.R. Swain and R.C. Hansdah, "A model for the classification and survey of clock synchronization protocols in WSNs," *Ad Hoc Networks*, vol. 27, pp. 219–241, 2015.

[6] M. Maroti, B. Kusy, G. Simon, and A. Ledeczi, "The flooding time synchronization protocol," in *Proc. 2nd ACM Conference on Embedded Networked Sensor Systems*, pp. 39–49, 2004.

[7] C. Lenzen, P. Sommer, and R. Wattenhofer, "PulseSync: an efficient and scalable clock synchronization protocol," *IEEE/ACM Transactions on Networking*, vol. 23, no. 3, pp. 717–727, 2015.

[8] H.-A. Tanaka, O. Masugata, D. Ohta, A. Hasegawa, and P. Davis, "Fast, self-adaptive timing-synchronization algorithm for 802.11 MANET," *IEE Electronics Letters*, vol. 42, no. 16, pp. 932–934, 2006.

[9] Crossbow Technology, Available at http://www.xbow.com/Products/Product_pdf_files/Wireless_pdf/MICA2_Datasheet.pdf.

[10] Z. Yang, L. Cai, Y. Liu, and J. Pan, "Environment-aware clock skew estimation and synchronization for wireless sensor networks," in *Proc. 2012 IEEE, INFOCOM*, pp. 1017–1025, 2012.

[11] J.M. Castillo-Secilla, J.M. Palomares, and J. Olivares, "Temperature-aware methodology for time synchronisation protocols in wireless sensor networks," *IEE Electronics Letters*, vol. 49, no. 7, pp. 506–508, 2013.

[12] Z. Yang, L. He, L. Cai, and J. Pan, "Temperature-assisted clock synchronization and self-calibration for sensor networks," *IEEE Trans. on Wireless Communications*, vol. 13, no. 6, pp. 3419–3429, 2014.

[13] K. Takano, M. Motoyoshi, and M. Fujishima, "4.8GHz CMOS frequency multiplier with subharmonic pulse-injection locking," in *Proc. 2007 IEEE Asian Solid-State Circuits Conf.*, pp. 336–339, 2007.

[14] H.-A. Tanaka, A. Hasegawa, H. Mizuno, and T. Endoh, "Synchronizability of distributed clock oscillators," *IEEE Trans. on Circuits Syst. I*, vol. 49, no. 9, pp. 1271–1278, 2002.