International Journal of Future Generation Communication and Networking
Vol. 9, No. 8 (2016), pp. 381-392
http://dx.doi.org/10.14257/ijfgcn.2016.9.8.36

A Novel Two-way Time Synchronization Protocol for Fusion Application

Yong Pang[1, 2, 3†], Guangming Li[1†*], Xiaoming Wu[2,3*], Fuqiang Wang[2,3] and Yifan Hu[2,3] and Qingchao Gong[1]

[1]*School of Mechanical, Electrical and Information Engineering, Shandong University, Weihai, 264209, China*
[2]*Shandong Provincial Key Laboratory of Computer Networks, Jinan, 250014, China*
[3]*Shandong Computer Science Center (National Supercomputer Center in Jinan), Jinan, 250014, China*
student0281@gmail.com, gmli@sdu.edu.cn, wuxm@sdas.org, wangfq@sdas.org, huyf@sads.org, gongqchao@163.com

Abstract

Data fusion application is one of the most popular applications in Wireless sensor networks (WSNs). Time synchronization is an important technology for data fusion applications due to its necessary for data fusion in the networks. Aimed at reliable time synchronization performance, this paper proposes a novel two-way time synchronization protocol (NTSP). Combing node's clock model and two-way packet exchange mechanism, NTSP makes time synchronization protocol more precise and much steadier. NTSP adopts simulated annealing (SA) algorithm to defend random noise and malicious delay attacks. Network simulations are carried out to analyze performances of NTSP on time synchronization accuracy, convergent rate and defensive degree for malicious delay attacks. Comparing to traditional time synchronization protocols, NTSP is an effective time synchronization protocol in WSNs.

Keywords*: wireless sensor networks; time synchronization; simulated annealing algorithm.*

1. Introduction

As the fast development of wireless sensor networks(WSNs), many wireless applications are widely used in forest, architecture, military fields and other areas [1].Among various research interests in WSNs, time synchronization technology has received much increasing attention. Time synchronization technology is necessary in many WSNs aspects such as data fusion applications, sleeping and waking up mechanisms [2], localization mechanism [3] and scheduling mechanism [4] and so on. Nodes' hardware makes output time different. The restricted power supply requires low-power and low-complexity technologies in WSNs. As one of the supported technologies, time synchronization technology has to be precise, steady and low-power.

In recent years, many time synchronization protocols have been proposed. Traditional time synchronization protocols such as RBS [5], TPSN [6] and FTSP [7] are devoted to synchronizing nodes' time. RBS uses packet exchange mechanism to synchronize nodes' time, but it bears transmission delay, receive delay and badly influence of nodes' distribution. RBS is not suitable for sparse network. By utilizing two-way time synchronization mechanism, TPSN can meet high reliability requirements for WSNs. However, TPSN does not take account of node's clock model. In TPSN, nodes have increasing time synchronization error. FTSP uses clock model to estimate

ISSN: 2233-7857 IJFGCN

node's clock parameters precisely, but its one-way time synchronization mechanism makes node blind to parent death. FTSP is incompetent to steady requirements for data fusion applications. In new time synchronization protocols, fireflies synchronization method synchronizes oscillator by interacting with others with pulse couplings [8]. Another new time synchronization protocols are distributed consensus time synchronization (DCTS) protocols [9, 10]. In distributed consensus time synchronization protocols, nodes achieve synchronization by converging themselves to a virtual clock. These time synchronization protocols require nodes dense distribution. And protocol for special applications is proposed in [11].

This paper proposes a new time synchronization protocol (NTSP) for special requirement in data fusion applications. NTSP utilizes two-way time synchronization mechanism and clock parameters to compensate time offset. Then, it uses influence factor to weight current clock parameters to improve network defensive degree. The rest of this paper is organized as follows. In section II, challenges of time synchronization technology are discussed and clock time model is presented. Section III gives a brief review of time synchronization protocols. Then NTSP is presented in section IV. Section V shows simulation results. Finally, in section VI we draw our conclusions.

2. Challenges of Time Synchronization and Clock Model

Traditional time synchronization protocols are always achieving synchronization in two ways. On one hand, by comparing nodes' local time in the same instant, time offset of each other can be gotten. On the other hand, node estimates clock model to compensate time offset from its parent. No matter which way is adopted, challenges of packet delay will be faced. As shown in Figure 1, packet suffers from send delay, access delay, transmission delay, propagation delay, receive delay and receiver processing delay [7].

(1) Send delay. Time spent at sender to construct time synchronization packet and transmit the packet to MAC layer.

(2) Access delay. Time spent in accessing to the transmit channel. The access delay is always affected by channel busy degree and network load condition.

(3) Transmission delay. Time used in sender physical layer to send the whole packet from antenna.

(4) Propagation delay. Time cost in packet propagation from sender to receiver.

(5) Receive delay. Time spent in receiving packet from antenna in receiver.

(6) Receiver process delay. This time is taken in receiver to process packet in its upper layers.

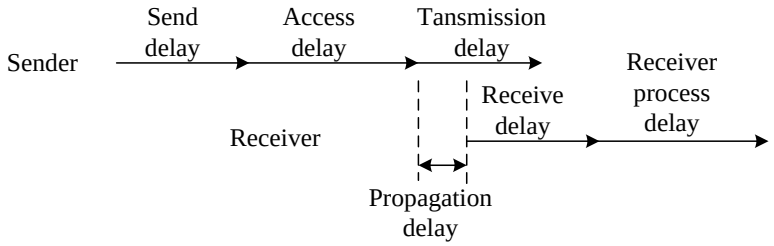

Figure 1. Packet Delay

Besides packet delay, node bears time offset resulting from quartz crystal. Each node is equipped with a crystal oscillator to record physical time. By counting oscillation and using special circuit conversion, node obtains its current local time. However, node's crystal oscillator is influenced by three factors: environmental factor, manufacturing factor and aging. They lead to node's time drift [12].

The output of an ideal quartz crystal oscillator is a sinusoidal analog-signal with a fixed frequency. This signal will turn to relevant digital signal with conversion circuit processing. By detecting digital signal, oscillation's quantization value will be gotten. Since oscillation period is so small that quantization value can be regarded as node's local time. Figure 2 illustrates the principle of node's local time with an ideal quartz crystal oscillator.

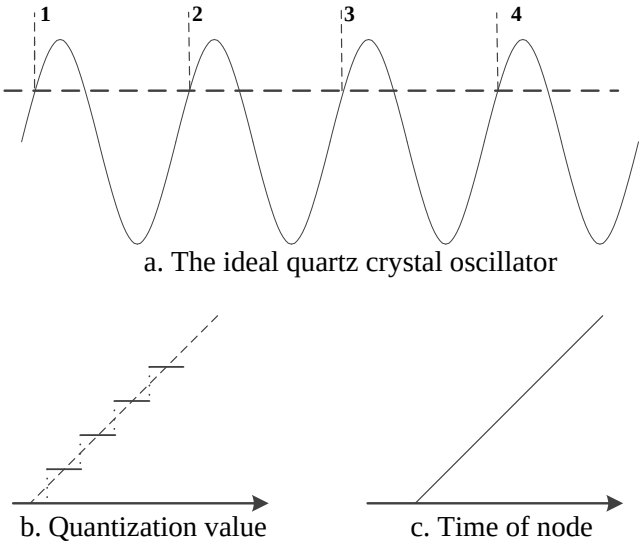

Figure 2. The Principle of Node's Local Time

Figure 2 shows that the corresponding clock skew of ideal quartz crystal is 1. Quartz oscillator always experiences frequency drift and frequency offset which results from aging and environmental factors. Node's power-on delay will appear when node is powered on. Frequency offset, frequency drift and power-on delay will change node's clock model in some degree. Actual time model of node usually has a time phase offset and a little changing clock skew, as shown in Figure 3.

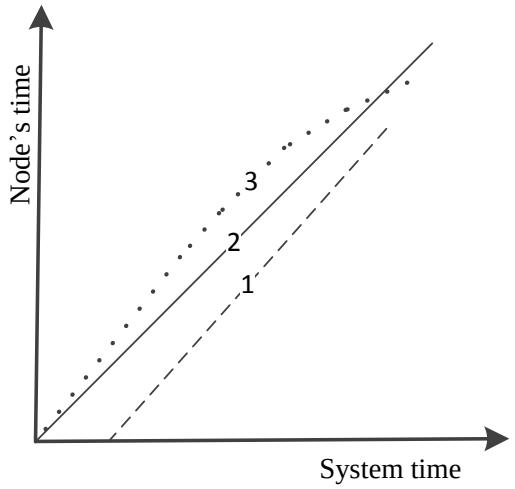

Figure 3. Actual Clock Model of Node

Different nodes' clock models are shown in Figure 3. Line 1 represents the clock model with power-on delay. Line 2 shows the model of ideal crystal oscillator and Line

3 represents the model affected by aging and environmental factors. In Figure 2(a), the output of ideal crystal oscillator is: $t_1 = \sin(2\pi f_0 t)$, where t and t_1 represent current real time and crystal oscillator's output respectively, and f_0 is the nominal frequency value of crystal oscillator. If we consider power-on delay, frequency offset and frequency drift, the output signal of crystal oscillator will be expressed as:

$$t_1 = \sin(2\pi f_0 t + f^{'} t + f^{''} t + \phi(t) + \theta) \quad (1)$$

where θ represents power-on delay, $f^{'}$ represents frequency offset, $f^{''}$ indicates frequency drift and $\phi(t)$ refers to the rest high order terms [12]. From perspective of experience, frequency offset of quartz crystal oscillator is typically 10^{-5}~10^{-4} and frequency drift is often around 10^{-11} [12]. The high order term is so little that it can be omitted. Then crystal oscillator's output will be expressed as:

$$t_1 = \sin(2\pi f_0 t + f^{'} t) \quad (2)$$

The node's clock model is:

$$t_1^{'} = clock\_skew * t + offset \quad (3)$$

To sum up, power-on delay, frequency offset and frequency drift make time differences in crystal oscillators. According to node's clock model expression, estimation of relative clock skew and offset is the key to time synchronization technology.

3. Related Work

As a currently hot technology of WSNs, time synchronization has been given much attention and many protocols have been proposed in recent years. These protocols include: RBS [5], TPSN [6], FTSP [7], Firefly synchronization method [8, 14], TSMP [13] and DCTS [10, 15].

In RBS, reference node broadcasts time synchronization packet. Neighbor nodes who receive this packet record receiving time based on their local clocks. One neighbor sends receiving time in a new packet to another. The one who receives new packet can compute time offset by its receiving time and the receiving time embedded in packet. In RBS, a large number of packets are essential and the density of network has to be large. RBS is not suitable for a sparse distributed network.

In TPSN, node starts time synchronization process with sending a time synchronization request packet. It will receive reply packet from parent node. By using four time-stamps in these two packets, node obtains and compensates time offset from parent node. It will make its clock synchronized with parent node. There is no consideration on node's clock model in TPSN. It results in an increasing time offset after synchronization operation. We set packet transmission delay as d and clock offset as Δ. Values of d and Δ will be gotten by equations (4) and (5). Son node can estimate the current parent's time by equation (6).

$$\Delta = [(T_2 - T_1) - (T_4 - T_3)] / 2 \quad (4)$$

$$d = [(T_2 - T_1) + (T_4 - T_3)] / 2 \quad (5)$$

$$t_{estimated}^{'} = t_{local} + \Delta \quad (6)$$

Four time-stamps of T_1, T_2, T_3 and T_4 representing nodes' local time of two nodes are shown in Figure 4.

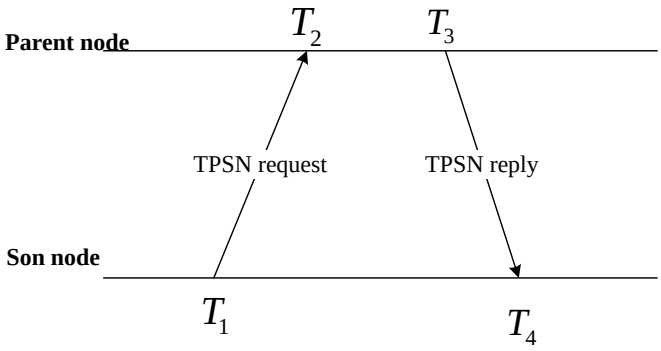

Figure 4. Time Synchronization Mechanism in TPSN

FTSP is a time synchronization protocol that uses clock model to estimate node's clock parameters. Root node broadcasts one-way time synchronization packet with bringing its sending time. Node records its receiving time. By using sending time and receiving time, node can obtain its clock skew and clock offset. By using clock model, FTSP has more precise time synchronization accuracy than TPSN and RBS. However, its one-way time synchronization mechanism makes node blind to parent's death. Multiple routes make FTSP have different synchronization errors with different route.

Firefly algorithm refers to some pulse-coupled synchronization protocols [8, 14]. In firefly algorithm, nodes are seen as identical pulse-coupled oscillators. If one burst, it will give a state increment to its neighbor. Firefly algorithm makes whole network synchronized in physical state rather than time. It does not need any synchronization packet. Because of its strict requirements, firefly algorithm is still in theoretical research.

TSMP assigns network slots, channels and other network sources in network construction stage. Assignment information is broadcasted in whole network. Nodes work in their own slots and communicate with their assigned nodes. In every communication period, there is a guard time for node's synchronization. When node receives time synchronization packet in the guard time, it will use time-stamp in packet to adjust itself and synchronize its clock. Otherwise, node thinks it loses synchronization in the network. Then it requests rejoining the network. TSMP is a cross-layer time synchronization protocol with combing network layer, MAC layer and physical layer. It always needs multiplexing in channels and frequency.

DCTS refers to some distributed consensus time synchronization algorithms [10, 15]. In DCTS, nodes periodically broadcast their own local time to make neighbors synchronized. DCTS achieves time synchronization relying on neighbors rather than parent node. So in DCTS, there is no need for hierarchical network topology. It is suitable for the dynamic network topology, but it has a slow convergence. DCTS needs so many time synchronization packets that it makes heavy synchronization network overhead.

In a summary, RBS and TPSN don't have precise synchronization. FTSP and DCTS need too many time synchronization packets. Firefly algorithm and TSMP have great complexity. We present a new two-way time synchronization protocol to balance the time synchronization precision and synchronized complexity.

4. A New Two-way Time Synchronization Protocol

4.1. The Principle of NTSP

Aiming to realize high precision and much steadier synchronization mechanism, we propose NTSP for data fusion applications. To make NTSP more sensitive to parent

fail, NTSP adopts two-way time synchronization mechanism to achieve nodes' synchronization. In data fusion applications, the most common communication is one hop communication. So we study synchronization in one hop. According to clock model presented in section II, time of node A and node B can be expressed as:

$$t_A = a_0 t + b_0 \qquad t_B = a_1 t + b_1 \tag{7}$$

Once we obtain node's relative clock offset $b_1 - b_0$ and relative clock skew $a_1 - a_0$, node's clock can be synchronized. In traditional two-way time synchronization mechanism, four time-stamps are recorded in two synchronization packets. Assuming that two nodes are synchronization in time t_0, after a period of time T, time of node A and B are:

$$t_A = a_0(t+T) + b_0 \qquad t_B = a_1(t+T) + b_1 \tag{8}$$

Based on the former assumption, node A and node B do not have any time difference in t_0 .It means that: $a_0 t + b_0 - a_1 t - b_1 = 0$.

In TPSN, after a time synchronization period T, time of node A and node B are T_1 and $T_2 - d$, respectively. $T_2 - T_1 = a_0 t_0 + a_0 T + b_0 - a_1 t_0 - a_1 T - b_1$. Because of $T_2 = T_1 + d + \Delta$, we can obtain node's clock skew and relative node skew between one hop nodes as followed.

$$a_1 = \frac{a_0 T - \Delta}{T} \qquad a_1 - a_0 = -\frac{\Delta}{T} \tag{9}$$

After obtaining relative clock skew of nodes, we can get relative offset:

$$T_1 = \frac{a_0 T - \Delta}{T}(T + t_0) + b_1 \qquad T_2 = T_1 + d + \Delta \tag{10}$$

$$b_1 - b_0 = \frac{t_0}{T}\Delta - d \tag{11}$$

To achieve accurate time synchronization between nodes, NTSP compensates node's time offset using relative clock skew and relative time offset. In NTSP time compensations occur in every reading time operations. As shown in Figure 5, node compensates its time many times during a time synchronization period when node reads its time. When it transmits messages to its parent or sons, its clock adjustments will occur.

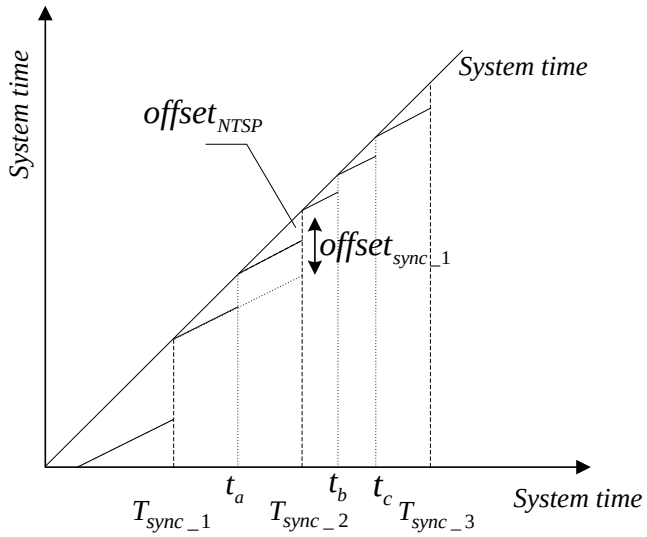

Figure 5. Time Compensation in NTSP

Figure 5 is also based on the above assumption of nodes' synchronization in t_0. According to relative clock skew, node computes time offset as expressed:

$$t_{B->A} = t_{B->A,n-1} + \frac{t_B - t_{B->A,n-1}}{(a_1 - a_0)_{n-1}} \tag{12}$$

The goal of NTSP time compensation is to remove time offset caused by different clock sources. To complete this goal, NTSP uses time increment from the latest synchronization operation and the latest relative clock skew to estimate parent's actual time increment from last synchronization instant $t_{A,n-1}$. In NTSP, relative offset between nodes is not essential. It only needs to update relative clock skew to make nodes synchronized by:

$$(a_1 - a_0)_n = (a_1 - a_0)_{n-1} - \frac{\Delta}{T} \tag{13}$$

In NTSP, the first time synchronization operation is made by traditional time synchronization protocol TPSN. By packets exchanging, node synchronizes its clock in one synchronization round instead of more than two or more synchronization processes in other protocols [7]. After the first time synchronization, we set *x* as the time synchronization error. The relative clock skew will be:

$$a_1 - a_0 = \frac{-\Delta + x}{T} \tag{14}$$

With time increasing, the influence of first time synchronization error *x* will decrease sharply. Finally, the first time synchronization error can be neglected and the above assumption condition can be fulfilled.

4.2. Influence Factor and Simulated Annealing Algorithm

WSNs are more vulnerable to many malicious attacks than traditional networks. Some attackers make node synchronization failed and destroy the whole network operation. Besides malicious attackers, there are some random noises in the WSN which will bring some random time delays. These two factors influence network synchronization precision and network safety in WSNs. To improve the performance of NTSP on these problems, we employ the influence factor α to update the relative clock skew as follows:

$$(a_1 - a_0)_n = \alpha(a_1 - a_0)_{n-1} - (1-\alpha)\frac{\Delta}{T} \tag{15}$$

The relative clock skew is updated by information of the current clock drift and the old relative clock skew. On one hand, it is difficult to directly compute the value of influence factor by theoretical analysis. On the other hand, simulated annealing (SA) algorithm is effective in obtaining system global optimization rather than local optimization. We get value of influence factor α by simulated annealing algorithm rather than computing it by theoretical analysis.

Simulated annealing algorithm originates from physical anneal phenomenon. At first, object has a higher temperature and a higher energy. When object's temperature falls down, its energy will decrease. In simulated annealing algorithm, there is a probability to accept worse solution to the problem. This behavior prevents simulated annealing algorithm from local optimization. In NTSP, node's influence factor α is set as objective function and synchronization error between parent's time and son's is evaluation function. If the *n*-th time synchronization error is less than the *n-1*-th time synchronization error, the value of influence factor in the *n*-th time round will be kept in the *n+1*-th time synchronization round. And if the situation is reverse, the value of influence factor in the *n*-th time round will be kept with the probability:

$$\exp(\frac{|sync\_error_n - sync\_error_{n-1}|}{T}) \quad (17)$$

4.3. The Summary of NTSP

The network executes following steps to achieve network synchronization and defend random noises and malicious delay attacks.

Step 1: At the beginning of topology construction, nodes power on one by one. Root node broadcasts network forming packet to its neighbors with layer Id 0. Node that receives this packet makes its own layer Id one plus layer Id in the packet. Then this node transmits a new forming packet with its own layer Id. Finally, each node in network will has its own layer Id and its own parent node. This step is similar with the operation in traditional two-way time synchronization protocol.

Step 2: The following step is time synchronization period. Time synchronization period consists of two periods: the period using traditional method and the period using clock model and simulated annealing algorithm. In the first period, node uses four time-stamps to obtain the estimated time offset from its parent node, and compensate this time offset by directly adding the estimated value. In the first time synchronization round, node's power-on delay are regarded to be compensated.

Step 3: After the first time synchronization round, node utilizes time-stamps to get its relative clock parameters and update the parameter by iteration and simulated annealing algorithm in the second period. Node compensates its time offset by estimating relative time increment of parent.

Step 4: During network working period, if node cannot receive packets from its parent two times in continuous synchronization rounds, it treats its parent dead. It broadcasts rejoin require packet to obtain a new layer Id and a new parent. Then this node will repeat step 2~step 3 to be synchronized with its parent.

Figure 6 shows the total working process of NTSP in a four hops network and node in layer 0 is the root node in the network.

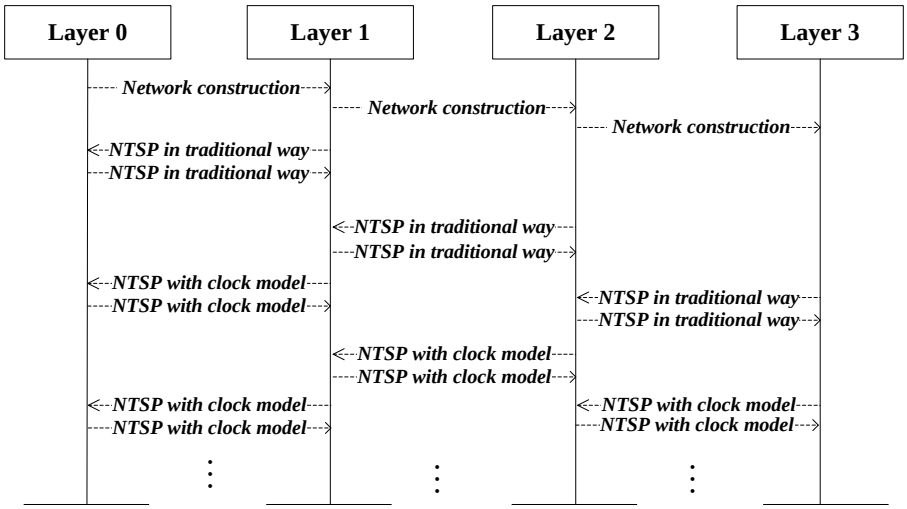

Figure 6. The Working Process of NTSP

5. Network Simulations

5.1. Experimental Setup

To measure performances of our proposed NTSP, we carry out some experiments in NS2 network simulator on time synchronization error, convergent speed and safety

degree of NTSP. In every time synchronization round, time synchronization error represents the current time difference between node and its parent. This metric is the basic metric for time synchronization protocols. Convergent speed is used to measure the rate of network synchronized. Faster convergent speed means the protocol make quickly reaction to system time change..Safety degree refers to the degree that protocol can defend malicious attacks. If a time synchronization protocol cannot defend malicious attacks, attacker will destroy the whole network by making nodes out of synchronization.

In our network simulations, 25 nodes in a 5 * 5 grid are adopted to test performances of NTSP. the distance of nodes is 10m. The central node works as the root node, whose time is regarded as the system time. Nodes' distribution is shown in Figure 7. Aim to simulate nodes' different physical clock models; we assume nodes' clock skews distribute in a random distribution with mean 0 and standard deviation 20 ppm. Node's communication radius is 10 meters. It makes node has a transmission range of 1 hop. In our simulations, the radio propagation model is two-ray ground reflection model.

Figure 7. 25 Nodes Network

5.2. Experimental Results in Synchronization Error and Convergent Speed

In this experiment, we analyze the time synchronization performance and convergent speed, so the improved NTSP with SA is not essential. To analyze time synchronization error of NTSP in multi-hops network, two classical time synchronization protocols TPSN and FTSP are used. There is no delay noise or malicious delay attack in this simulation. Nodes are powered on between 0s and 12s before time synchronization round starts. Every node has a 0.5s power-on delay increment from former node. We set every node begin its time synchronization round at 15s later from its power-on instant to make sure the whole network construction is finished. Time synchronization errors for different protocols are shown in Figure 8.

In simulation process of NTSP, son node receive synchronization packet from its parent and analyze embedded time-stamp to update it clock. Node changes its clock after every synchronization round. Differences between nodes' local time and system time represent time synchronization errors. Time synchronization errors of different protocols are shown in Figure 8.

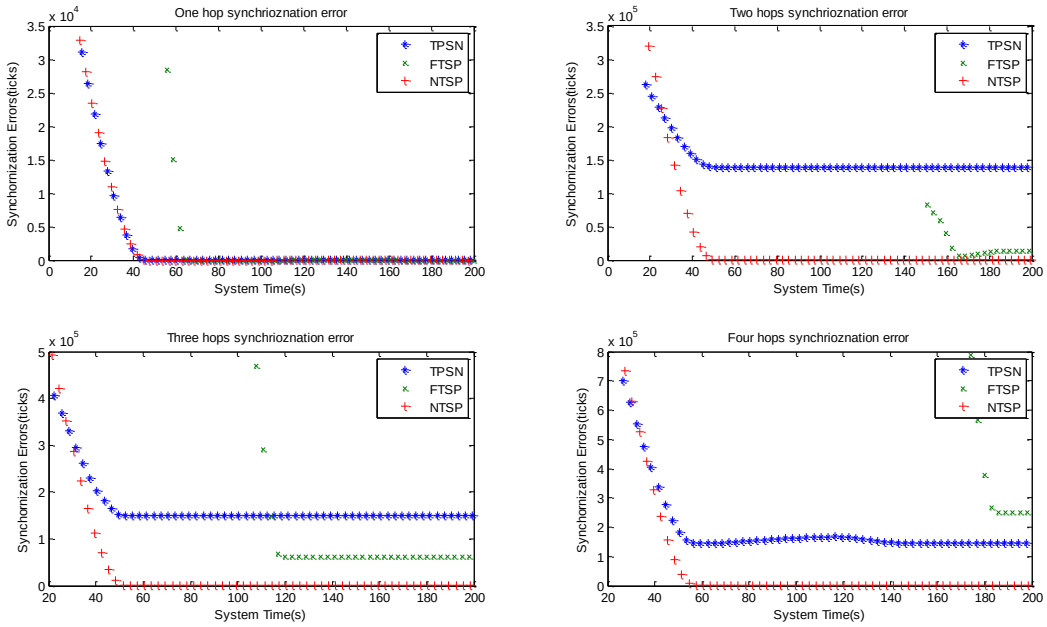

Figure 8. Synchronization Errors in Multi-Hops Network

In Figure 8, time synchronization performances in multi-hops networks are shown in the first/second/third and fourth sub-figures. Time synchronization error is quantified on tick which is a little time increment when node's quartz crystal oscillator counts 1. Three protocols all achieve good synchronization performances in one hop communication. In the multi-hops communications, time synchronization error performance under NTSP is much better than that under TPSN and FTSP. TPSN always has a large time synchronization errors in multi-hops communications. Time synchronization error performance of FTSP is better than TPSN in two hops and three hops communications. However, FTSP is the worst one in four hops communications. The slowest convergent speed makes it have a bad time synchronization performance at the beginning of network synchronization period, especially for nodes that far away from root node. In NTSP, the first time synchronization round adopts traditional synchronization method in TPSN, so NTSP can make node synchronized rapidly. Its time compensation mechanism with clock model makes accurate time synchronization.

Comparing with FTSP, NTSP has a faster time synchronization convergent speed. In FTSP, the first node has to wait for time synchronization periods ROOT_TIMEOUT to be root node and node in n-th layer has to wait extra time synchronization periods (n-2) * ENTRY_SEND_LIMIT to get time information of root node. The larger the network is, the larger the waiting time will be. Figure 8 shows that NTSP has big advantages in time synchronization errors and convergent speed.

5.3. Performance Analysis on Safety Degree

After analyzing performances of NTSP without SA in synchronization error and convergence, this part will test safety degree of full NTSP. In full NTSP, object function in SA is node's influence factor value. Time synchronization error is set as evaluated function. If the synchronization error is less than the one in the last synchronization round, the influence factor will keep its current value in the next synchronization round. Otherwise, the influence factor value will be retained in some probability. If the influence factor changes its value, the new value is gotten by adding a random value between -0.01~0.01. We introduce malicious delays in Gaussian distributions with mean 0 and standard deviation 625 ticks. Without comparing with

TPSN and FTSP, NTSP and fully NTSP with SA are analyzed in safety degree performance. Nodes' power-on delays are not in consideration. Both of them have a 10s time synchronization period. Their time synchronization errors in multi-hops communications are shown in Figure 9.

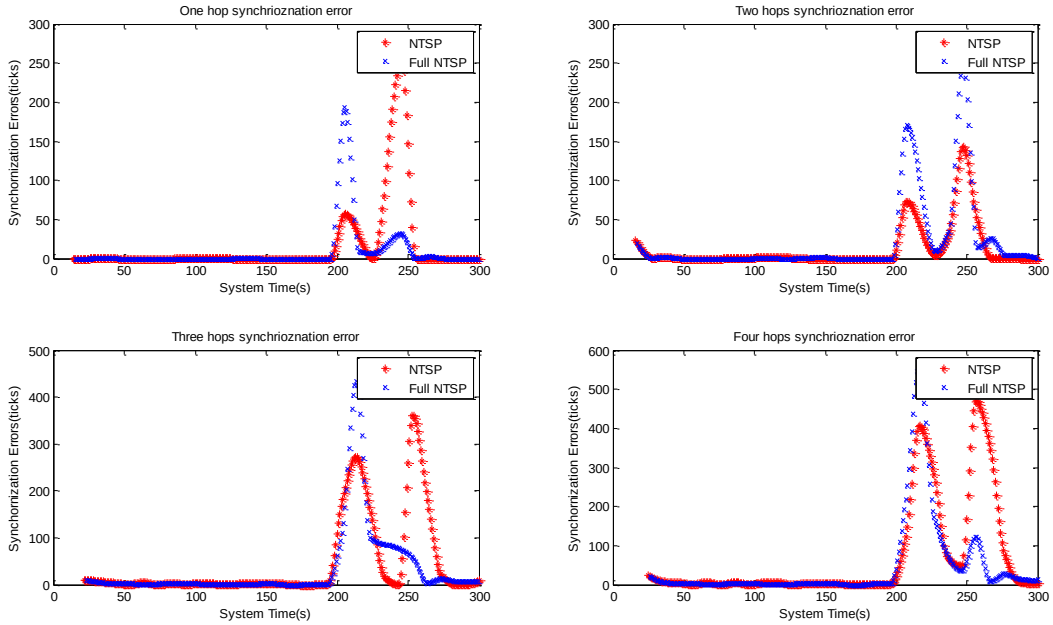

Figure 9. Safty Degree of NTSP and full NTSP

Figure 9 presents safety degree performances of NTSP and full NTSP in multi-hops communications. In 200s~210s and 245s~250s, there are Gaussian random delays in received time synchronization packets. In Figure 9, we can see two time errors peaks in every sub-figure. The first delay attack makes both NTSP and full NTSP have large synchronization errors. But the second delay attack leads to different influences to NTSP and full NTSP. We can find that there is always a little peak following the former one in full NTSP. This phenomenon is produced by influence factor of full NTSP. When delay attack occurs, the influence factor changes its value to adjust the relative clock parameters and the corresponding clock compensation progress will happen next. Synchronization error peaks results from the second delay attack under NTSP are higher than that under full NTSP for one hop, three hops and four hops communications. In Figure 9, full NTSP has a little synchronization error jitter in two hop situation. Because i when the bad influence factor value is dropped, the new value will be produced in random. And if an influence factor value brings a bad time synchronization performance on synchronization accuracy, it still be kept in a certain probability.

6. Conclusion

A novel two-way time synchronization protocol for data fusion applications has been proposed in this paper. It is based on two-way packet exchanges mechanism, continuous clock model and SA algorithm. By carrying out some network simulation experiments, NTSP is proved much more precise and has a more quickly convergence speed in network. Two-way time synchronization exchange mechanism is adapted to fulfill steady link requirement in data fusion applications. The one-way mechanism is used to lose synchronization overhead. The clock parameter update progress utilizes SA algorithm. It makes protocol more stable for malicious delay attacks or random noises.

This protocol is adaptable to data fusion applications in WSNs for its excellent performance in synchronization precision and steady. Because of its accurate time synchronization, NTSP makes synchronization period longer possible in a given synchronization error. In this way, NTSP will need less time synchronization packets and have much less network overhead. Our future work is to make time synchronization period longer and power consumption for synchronization less.

Author Contribution

Y.P. and G.L. contributed equally and share the first authorship. Corresponding authors are G.L. and X.W.

Acknowledgements

This work is supported by Shangdong Province Young and Middle-Aged Scientists Research Awards Fund, China (No.BS2013DX019, BS2013DX019, No. BS2013DX021, BS2013DX019), Basic Research fund of Shandong Academy of Sciences (No. [2015]25), National Natural Science Foundation of China (No. 61401257), Shandong Academy of Science Doctoral Fund (No. [2012]58), Shandong Academy Young Scientists Fund Project (No. 2013QN037). Additional support is from Science and Technology Development Project of Weihai. The authors thank anonymous reviewers and the editor for their valuable comments to improve the presentation of the paper.

References

[1] P. Corke, T. Wark, R. Jurdak, H. Wen, P. Valencia and D. Moore, "Environmental wireless sensor networks", Proceedings of the IEEE, vol. 98, no. 11, **(2010)**.
[2] D. Y. Gao, L. J. Zhang and H. C. Wang, "Energy saving with node sleep and power control mechanisms for wireless sensor networks", The Journal of China Universities of Posts and Telecommunications, vol. 18, no. 1, **(2011)**.
[3] Y. Peng and D. Wang, "A review: wireless sensor networks localization", Journal of Electronic Measurement and Instrumentl, vol. 25, no. 5, **(2011)**.
[4] M. R. Palattella, N. Accettura, L. A. Grieco, G. Boggia, M. Dohler and T. Engel, "On optimal scheduling in duty-cycled industrial iot applications using IEEE802. 15.4 e TSCH", IEEE Sensors Journal, vol. 13, no. 10, **(2013)**.
[5] J. Elson, L. Girod and D. Estrin, "Fine-grained network time synchronization using reference broadcasts", ACM SIGOPS Operating Systems Review, vol. 36, no. SI, **(2002)**.
[6] S. Ganeriwal, R. Kumar and M. B. Srivastava, "Timing-sync protocol for sensor networks", Proceedings of the 1st international conference on Embedded networked sensor systems, **(2003)**.
[7] M. Maróti, B. Kusy, G. Simon and Á. Lédeczi, "The flooding time synchronization protocol", Proceedings of the 2nd international conference on Embedded networked sensor system, **(2004)**.
[8] Z. L. An, H. S. Zhu, X. R. Li, C. N. Xu, Y. J. Xu and X. W. Li, "Nonidentical linear pulse-coupled oscillators model with application to time synchronization in wireless sensor networks", IEEE Transactions on Industrial Electronics, vol. 58, no. 6, **(2011)**.
[9] L. Schenato and F. Fiorentin, "Average TimeSynch: a consensus-based protocol for clock synchronization in wireless sensor networks", Automatica, vol. 47, no. 9, **(2011)**.
[10] M. K. Maggs, S. G. O'Keefe and D. V.Thiel, "Consensus clock synchronization for wireless sensor networks", IEEE Sensors Journal, vol. 12, no. 6, **(2012)**.
[11] J. Dai and Y. Wang, "Study on OFDM Symbol Timing Synchronization Algorithm", International Journal of Future Generation Communication and Networking, vol. 7, no. 1 **(2014)**.
[12] A. Delawari, "Time synchronization in wireless sensor networks", Delft University of Technology, Delft, **(2013)**.
[13] K. S. J. Pister and L. Doherty, "TSMP: Time synchronized mesh protocol. Proceedings of the International Symposium on Distributed Sensor Networks", **(2008)**.
[14] U. Ernst, K. Pawelzik and T. Geisel, "Delay-induced multistable synchronization of biological oscillators", Physical Review E, vol. 57, no. 2, **(1998)**.
[15] M. Li, G. Zheng and J. Li, "Clock Self-Synchronization Protocol based on Distributed Diffusion for Wireless Sensor Networks", International Journal of Future Generation Communication and Networking, vol. 7, no. 5, **(2014)**.

