TELKOMNIKA Indonesian Journal of Electrical Engineering
Vol. 12, No. 9, September 2014, pp. 6874 ~ 6883
DOI: 10.11591/telkomnika.v12i9.4990

A Cooperative Time Synchronization Protocol for Wireless Sensor Networks

Min Li*[1], Guoqiang Zheng[1], Jishun Li[2], Lei Fu[1]
[1]School of Information Engineering, Henan University of Science and Technology,
Henan Luoyang, China
[2]Henan Key Laboratory for Machinery Design and Transmission System,
Henan University of Science and Technology, Henan Luoyang, China
*Corresponding author, e-mail: limin0919@163.com

Abstract

The synchronous precision of synchronization protocol is low and the scalability is limited strictly in large scale wireless sensor networks (WSNs).Considering the issue, a novel cooperative time synchronization protocol based on pulse-coupled oscillators and distributed diffusion (CTSP) is proposed in this paper. The protocol works in two phases: clock tick synchronization phase and time synchronization phase. In the clock tick synchronization phase, the phases of the pulse-coupled oscillators is changed by the pulse coupled signal of the SINK node. The tick synchronization of all nodes is achieved by distributed diffusion. In time synchronization phase, the average time of reference nodes is spread to a limited number of hops based on two-way message exchange mechanism. Moreover, in order to achieve the synchronization of entire network, it adopts the method of mutual diffusion to finish the approximately synchronization between the node time and the average time of all nodes. By comparing CTSP with TPSN, we show that, CTSP can synchronize the network quickly with good precision convergence speed and scalability, which appropriates for large-scale WSNs.

Keywords: *wireless sensor networks, pulse-coupled oscillators, cooperative synchronization, distributed diffusion*

1. Introduction

Wireless sensor networks (WSNs) normally consist of a large number of sensors distributed over a given area. These low-cost sensors have limited computing, communication, and sensing capacity [1]. WSNs can be used for monitoring [2-3], object localization and tracking [4-5], etc. Most of these applications require the operation of data fusion, power management, and transmission scheduling among a large set of sensor nodes, which, in turn, require all the nodes running on a common time frame.

However, every individual sensor in the WSN has its own clock. Different clocks drift from each other over time due to many factors, such as imperfection of the oscillators and environmental changes. This makes clock synchronization between different nodes indispensable. In addition, sensor nodes in a WSN are too energy constrained and computation limited to use any complex synchronization schemes. Due to all aforementioned challenges, several time synchronization schemes for WSNs have been proposed since Elson and Romer first discussed this problem in 2002 [6], including the reference broadcast synchronization (RBS) protocol [7], time synchronization protocol for sensor networks (TPSN) [8], the delay measurement time synchronization protocol (DMTS) [9], the flooding time synchronization protocol (FTSP) [10], etc. Most of the proposed synchronization methods just focus on the minimizing of synchronization error and energy consumption, ignoring the requirement for scalability. At present, the time synchronization protocols for single-hop networks are very mature and the synchronization error can achieve about up to a dozen microseconds. The cost is lower which can satisfy most applications. As to multi-hop synchronization, it results in the accumulation of synchronization error over hop distance in large-scale WSNs. The theoretical analysis and numerical experiment show the synchronization errors are proportional to the distance of hops between nodes and reference nodes [11]. So, there must be some synchronization error accumulation and can not satisfy synchronization accuracy in large-scale

Received October 29, 2013; Revised June 28, 2014; Accepted July 21, 2014

WSNs. In addition, the complicated calculation and frequent data packets exchange will generally burden for the normal running of WSNs. Furthermore, with the gradual increase of the network scale and the network-based application, it becomes more and more time to achieve synchronization [12].

In this paper, a cooperative time synchronization protocol (CTSP) is proposed to deal with the issue of low precision and scalability in sensor nodes. According to the distribution characteristics and energy information of nodes, master nodes are selected. Based on the partition, the algorithm works in two phases clock tick synchronization phase and time synchronization phase Simulation results show that the unified clock-tick can ultimately be realized for network nodes. In addition, a salient feature of the proposed method is that, in the regime of asymptotically dense networks, it can maintain global synchronization in the sense that all multi-hop network nodes can successfully achieve conversion from synchronicity to synchronization. And beside that it can extended holding synchronization time and has the higher robustness and scalability.

The remainder of this paper is organized as follows: The related works are reviewed in Section 2. Section 3 formulates the time-synchronization problem considered in this paper. Section 4 proposes a cooperative time synchronization protocol to analyze the time synchronization issues. The results of experiments and simulations are discussed in Section 5, followed by the conclusions of this paper given in Section 6.

2. Related Works

In recent years, numerous synchronization protocols have been proposed, focusing on different aspects of the synchronization problem in WSNs [13] . For the traditional protocols, they need a root node and are tree-based, and they are not fully distributed, which means that they are fragile to link or node failures. Thus, these traditional protocols are not optimal for handling clock synchronization in random mobile sensor network. Existing distributed protocols [14-16], these protocols and the associated theoretical results are obtained assuming that the topology of the network is connected or joint connected, which holds no longer in random mobile sensor network. These protocols also have slow convergence speed.

There are some works which investigate Pulse-Coupled Oscillator (PCO) for sensor networks. The nonlinear dynamics of large populations of PCO were studied to describe the synchronous fireflies flashing, observed in the south east of Asia since the past two centuries. The PCO algorithm [17] makes much more liberal use of the physical communication constraints that are acknowledged possible in traditional packet-switched point-to-point network models. From the theoretical point of view, the protocol [14] is based on gossip averaging algorithms, and the protocols GTSP in [16] and ATS in [15] are based on average consensus algorithm, which have slow convergence speed as pointed in [18]. The study of consensus for sparse, mobile Ad Hoc networks is proposed in[19]. Recent results in [20] show that the approximate model used in [21] to prove convergence does not, in fact, warrant convergence for all connected networks. Note that the converging speed of the time synchronization is a critical problem in practice, while most of existing consensus based protocols, which aim to reach an average value within the network, are time-consuming. And besides the PCO algorithm is only provides a unified ticking rhythm across sensor nodes, namely synchronicity not the synchronization of time. In order to realize the time synchronization, the time of each wireless sensor nodes need to be synchronized. Hence it is of great interest to develop a protocol which owns much faster converging time while maintaining the advantages of consensus.

3. Summarize the Protocol Algorithm

The periodical process of CTSP as shown in Figure 1 is composed of two major parts: clock tick synchronization based on pulse-coupled oscillators and time synchronization. Each cycle of synchronous execution process is as follows:

1) The clock tick synchronization based on pulse-coupled oscillators: Firstly, SINK node emits m pulses with equidistant zero-crossing. The surrounding nodes receive this pulse sequence, and based on the locations of the observed zero-crossings, the surrounding nodes predict when the next pulse will be transmitted. Then, these nodes emit pulse at their predicted times and an aggregate pulse sequence is generated. Although the prediction at an individual

node may not be perfect, under certain conditions on the pulse and in asymptotically dense networks [22-23], the zero-crossings of the aggregate waveform sequence will be at the same positions as the zero-crossings of the original waveform sequence emitted by the SINK node due to spatial averaging. This aggregate pulse sequence is heard by the nodes lying further away from SINK node and these nodes perform prediction as described above and emit their pulses to their predicted time [3]. Peripheral nodes being added to the ranks of the sending synchronization pulse by outward recastion formula. After m pulses, the new agent to join the ranks and the more nodes pulse can generate sufficient energy through coupling. At last, synchronization pulses are sent by all the nodes in the networks at the same time, namely, achieving synchronization state [24-25].

2) The time synchronization phase: Firstly, with two-way message exchange models, all neighbor nodes time in the master-node domain is obtained, and the average time of nodes in broadcast domain is calculated. Secondly, the master-node is synchronized to the average time of all nodes in the master-node broadcast domain, and then defines the master's time as reference time. The diffusion-nodes are chosen on the basis of average transmission delay and energy, which just begins from the master-node and its spread to, ϖ hops distance nodes from master-node. The nodes within ϖ hops distance from the master nodes will receive more clock synchronization information from the same master node or different master nodes. By using of the information to update local cycle the clock and operating μ cycle according to the process of implementation, the network will complete synchronization process at a time.

Assume that N sensor nodes of large-scale WSNs have high density in the rectangular area of $W \times W$ according to a uniform probability distribution. In addition to the SINK, the hardware facilities of any other network node are similar and the communication rang of sensor nodes is R . Using straightforward broadcasting or flooding, SINK can realize the initial information of startup synchronization and the reference nodes election transfer operation to sensor nodes. The implementation of two stages will be described in the following discussion.

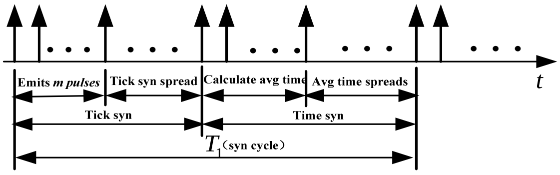

Figure 1. Cycle Implementation of Synchronization Protocol

4. The Cooperative Time Synchronization Protocol Scheme

4.1. Clock Tick Synchronization Based on Pulse-Coupled Oscillators

In the phase, each node is considered as a controllable oscillator. The coupling interaction between nodes in the network is mainly finished through transmitting and receiving the periodic narrow pulse signal, and then realizes the node phase synchronous. In this scheme, each node (say node i) in the sensor network is associated with an increasing monotonic phase function $\Phi_i(t)$ taking values from 0 to 1, defined as:

$$\Phi_i(t) = \frac{t}{T} + \Phi_i(0) \tag{1}$$

If a node is isolated, the state function x_j increases linearly from 0 to 1 smoothly as a function of time as follows:

$$x_j = f(\Phi_j) = \Phi_j, \ \ \Phi_j \in [0,1], \ \ j = 1,2,...,N \tag{2}$$

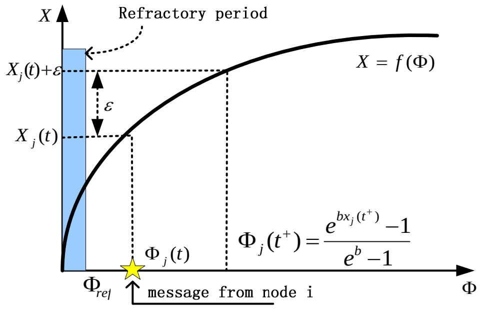

Figure 2. General Model of Pulse-Coupled Updating Dynamics

The node emits a pulse when the state function achieves the threshold value ($x_j = 1$). After the pulse titillated, the node will immediately reset its state to zero so that emission pulse can be sent in the periodic T . If a node is not isolated, it can receive pulses from other nodes and then change the status variable. The phase function $\Phi_i(t)$ replaces the state function x_j changes accordingly as:

$$\begin{cases} x_j(t^+) = B(x_j(t^+) + \varepsilon) \\ \Phi_j(t^+) = \dfrac{e^{bx_j(t^+)}-1}{e^b-1} \end{cases} \tag{3}$$

And $B(x) = \begin{cases} x\ (0 \le x \le 1) \\ 0 \quad (x < 0) \\ 1 \quad (x > 1) \end{cases}$. This means that a node receiving a pulse either emits the pulse at the same time or shortens the waiting time for the next cycle of emissions. It can be shown that only when the nodes emit the pulse simultaneously will they be insensitive to coupling, and therefore achieve synchronization. If there is time t_0 such that meets (4), we consider all nodes achieve clock tick synchronization:

$$\Phi_1(t) = \Phi_2(t) = \ldots = \Phi_i(t) = \ldots = \Phi_n(t), \forall t > t_0, n = N \tag{4}$$

It may trigger infinity and circulation firing problem, because of coupling delay is not considered, namely infinite feedback [23]. Node i firing can result in node j and other nodes firing, coupling data packet which sending back from j and other nodes firing may causes node i firing again. If i firing breaks out in the network, a new cycle of the firing nodes will be happen again and cause infinity and circulation firing. In order to avoid infinite feedback, we supposed that after a node fires a pulse, it enters a short refractory period, during which no signal can be received from other nodes. That means the node cannot response to the new firing signal. It can better solve infinite feedback problem of nodes.

4.2. The Time Synchronization Phase

The pulse-coupled algorithm only provides a unified ticking rhythm across sensor nodes, namely synchronicity not the synchronization of time. In order to realize the time synchronization, each sensor node time needs to be synchronized. So we should make use of the concept of distributed diffusing and employ master-nodes and diffusion-nodes dynamic election mechanism, the master-nodes and diffusion-nodes are chosen on the basis of the energies and average transmission delay. The network time synchronization is achieved by using the inter-diffusion method, the average time of the master-node domain is diffused finite hops based on the two-way message exchange mechanism. The entire network nodes time has been approximately synchronized to average time. The master-nodes election must satisfy the following rules：

Rule 1. Assume that each node maintains threshold value φ. During the election of the master-nodes, each node generates a random number λ, $\lambda \in (0,1)$. δ represents the ratio of the current residual energy and the first maximum energy of the node, ζ is expressed as follow:

$$\zeta = \lambda - (1-\delta) \qquad (5)$$

Provided that $\zeta > \varphi$ implies that the node can be declared as a master-node. In (5), the threshold value of φ determines the number of nodes allows declaring, the ratio of the master-nodes for network nodes is $\gamma = 1 - \varphi$.

Rule 2. All nodes, satisfying Rule 1, wait certain time and send a master-node statement message in its broadcast domains stochastic. If other nodes satisfy Rule 1 in its broadcast range, these nodes will exit the as the master-nodes competitive; If neighbor nodes receive different statement messages or packet collisions in the scope of their broadcasts domains, these nodes will immediately send a respond for conflicting information. Upon receiving the response message, the node that statements issues packet in accordance with probability 1/2 determines whether to continue sending the statement message as the primary node until not existing the neighbor nodes ,which receive declaration packets from different nodes in broadcasts domains of sending node statement message; If its neighbor nodes in broadcast range receive only the statement message, then the node that sent statement message can begin to execute synchronization after waiting for a certain period of time. The master-nodes election is multi-cycle; each τ second will be re-election to the master-nodes in synchronous time $\mu\tau$.

4.2.1. The Average Time of Node in Master-Node Broadcast Domain

Assuming that S nodes are elected to be the master-node, and the number of nodes in each master-node broadcast domain is $n_j (j = 1, 2, ..., s)$, $c_j^l = (c_1^l, c_2^l, ..., c_{n_j}^l)$ is the time value of n_j neighbor nodes in the master-node election broadcast domain at time l, $c_k^l (k = 1, 2, ..., n_j)$ is the time value of node at time l, where c_1^l is the master-node's time value. The acquisition process of average time in master-node broadcast domain as follows：

1) The master-node broadcasts a ch-quest packet (including the sync-start of start time synchronization, the master-node ID, the local time value) to start a new cycle of synchronization;

2) According to the ch-quest packet, the neighbor nodes send ACK response packet containing a timestamp (including the local time value when node receiving ch-quest packet, the local time value when node transmitting ACK packet, the neighbor nodes ID) after certain time of random waiting；

3) When received the response packet the master-nodes start to calculate the propagation delay between nodes, and then send a sync-continue packet(including sync-flag, the delay d_k between master-node and responsive neighbor node k, the node k ID and the transmission time of the current broadcast information) to the neighbor node. After the neighbor node k receiving the sync-continue packet, we can obtain the delay d_k ,and other neighbor nodes is not repeated information exchange by receiving the sync-continue packet until the master-node reissue a sync-continue packet and then wait for a maximum delay time $D_{\max}$ but failure to receive the timestamp information from neighbor nodes. All of which show the master-node obtains the time information of all the neighbor nodes.

Figure 3 shows the master-node A broadcasts the start time synchronization packets at time l, according to the described steps to implement the information exchange process.

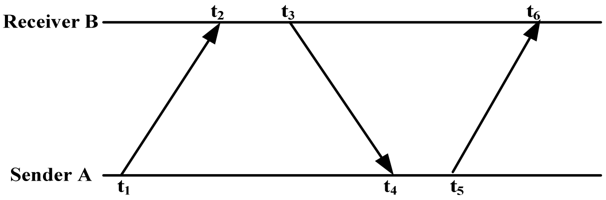

Figure 3. A Sender A -receiver B Two-way Timing Exchange Model

Δ_k and d_k denote the clock drift and propagation delay between node k and neighbor nodes, then Δ_k and d_k can be expressed as:

$$t_2 = t_1 + \Delta_k + d_k \tag{6}$$

$$\begin{cases} \Delta_k = \dfrac{(\mathrm{T}_2 - T_1) - (T_4 - T_3)}{2} \\ d_k = \dfrac{(\mathrm{T}_2 - T_1) + (T_4 - T_3)}{2} \end{cases} \tag{7}$$

At time l, the master-node clock time $c_1^l = T_1$ was main time of the neighbor nodes which can be written as:

$$c_k^l = c_1^l + \Delta_k \tag{8}$$

4) From (7) and (8), all nodes average time $\overline{c_j^l}$ and average propagation delay $\overline{d_j}$ within the master-node broadcast domain at time l take the form:

$$\begin{cases} \overline{c_j^l} = \sum_{k=1}^{n_j} c_k^l / n_j = c_1^l + \sum_{k=1}^{n_j} \Delta_k / n_j \\ \overline{d_j} = \sum_{k=1}^{n_j} d_k / n_j \end{cases} \tag{9}$$

4.2.2. The Selection of Diffusion-Nodes and the Diffusion of Average Time

Using a reference time of the average time of the master-node, the concrete selection rules of diffusion-nodes which implement the average time diffusion as follows：

Rule 3. After receiving the ch-quest packet of first-order master-node or diffusion-node, node produces random number η, where $\eta \in (0,1)$. $\zeta = \eta - (1 - \delta)$ is calculated according to the method of election the master-node and node energy. If $\zeta \geq \varphi_1$, the node can be a diffusion-node, otherwise cannot become the diffusion-node. The threshold value φ_1 determines the number of election diffusion-nodes and relates to node density, communication radius and so on. Since δ is variation with energy, the threshold value φ must adjust according to $\varphi_1 = \varphi_1 - \nu$ after each cycle of synchronization, where ν can set for any small positive according to application.

Rule 4. The node received the ch-quest packet for the upper master-node or diffusion-node, and then if $d_k > \overline{d_j^{\varpi_L}}$ (where $\overline{d_j^{\varpi}}$ corresponds to the average single-hop delay, ϖ_L stands for hop communication and d_k denotes the message delay between node and the master-node or the upper diffusion-node.), the node can be diffusion-node, otherwise cannot become the diffusion-node. Time spreads out from the master-nodes and the diffusion process of average time is as follows：

0-Diffusion: the master-node sends synchronous packet which contains the following information: the master-node ID; the average time $\overline{c_j^l}$; the average time transmission hop number ϖ , each diffusion time the value minus 1; the average delay $\overline{d_j}$.

1-Diffusion: when the 0-diffusion carries out, the 1-diffusion nodes and the neighbor nodes which cannot obtain average time information of the same master-node need to communicate information in the broadcast domain, and then computes the average single-hop delay $\overline{d_j^{\varpi}}$ for all the neighbor nodes and the 1-diffusion nodes updates the receiving master-node average time for: $\overline{c_j^l} = \overline{c_j^l} + d_{0,k}$,where $d_{0,k}$ stands for information propagation delay between the current diffusion-node and its master-node; the receiving average time transmission hop number ϖ in the diffusion packet minus 1; the average delay $\overline{d_j}$ between the master-node and neighbor nodes is replaced by $\overline{d_{1,\ k}}$,which is the average delay between current diffusion-node and its neighbor nodes; the updated diffusion packet was broadcast to the next hop of the neighbor nodes.

f -Diffusion: the f -Diffusion process is similar with 1-Diffusion.The diffusion process is repeated until ϖ hop distances from the master-node.

Where ϖ depends on the precision and the speed of the synchronization and must be able to sure the adjacent master-node obtain the average time in two master-node domains at least and is used as a time reference to synchronistically update, so ϖ satisfies following conditions:

$$\begin{cases} \sqrt{2}W/R \ge \varpi = 2W/(R\sqrt{\pi\gamma N}) \ge 2 \\ SS^{\mu}\varpi \ge \sqrt{2}\,W/R \end{cases} \tag{10}$$

The master-node ID is set to avoid repeatedly receiving the time synchronization information from the same master-node domain. During a certain diffusion nodes performing diffusion, the neighbor nodes have received the average time diffusion information from the same master-node, which has was been confirmed based on the recording the master-node number, and it is no longer involved in the diffusion node information exchange; If all the neighbor nodes of diffusion-nodes have been already received the average time diffusion information from the same master-node, the diffusion-node will end the diffusion process after having been waited on for a certain time.

According to the received average time diffusion packets of the master-node in the each cycle of synchronization time, the node calculates the new time T_{new}：

$$T_{new} = [\overline{c_M(l)} + d_k + L \cdot T_{local}]/(L+1) \tag{11}$$

Where $\overline{c_M(l)}$ is the average time of master-nodes domain, d_k is node propagation delay between k node and the master-node and L is the received synchronization diffusion packets number which minus 1. If $|T_{new} - T_{local}| > \delta$, the update of node is such that T_{new}, otherwise to maintain local time T_{local} at constant value.

5. Experiment and Simulation Results

In order to validate the synchronous effect of the proposed synchronization algorithm simulates the experiment on the Mica Z platform. Giving the correlative parameter value for the simulation: monitoring area is $500 \times 500\mathrm{m}^2$, 1000 nodes are randomly deployed, communication radius of sensor nodes are $R = 10m$, the coupling strength is $\varepsilon = 0.02$, cycle time is $T = 10s$, the simulation time is 50min, and each experiment is the mean of 100 simulations.

5.1. Analysing the Convergence of CTSP

In the previous section, we analyzed a synchronous version of CTSP algorithm and given its implement procedure. We will analyze the convergence of CTSP in this section. Suppose that the deviations for all the nodes time and standard time are uniform distributed in *[L$_t$, Ht]* at time l, CTSP satisfies convergence theorem as follow.

Theorem 1. For large-scale sensor network, CTSP can be gradual convergence in *C*, which equals to the average clock of all the nodes in the network.

Prove Assuming that the number of main nodes is S each selected round, H_t^j and L_t^j denote the maximum and minimum value of deviations for the average time of S master-nodes domains after j set; $c_n^j(l)$ and $c_{std}(l)$ stand for the arbitrary nodes time and standard time after j set of synchronization, then after synchronizing μ set, the arbitrary node time $c_n^{\mu}(l)$ satisfies:

$$L_t < L_t^1 < ... < L_t^{\mu-1} < L_t^{\mu} \le c_n^{\mu}(l) - c_{std}(l) \le H_t^{\mu} < H_t^{\mu-1} < ... < H_t^1 < H_t \qquad (12)$$

We know $H_t^j \ge C$ and H_t^j is nonincreasing. Letting the infimum of the series H_t^j be M ,we have $\lim_{t\to\infty} H_t^j = M \ge C$ .Suppose $M \ne C$ We will derive a contradiction.

Consider the function $x(\alpha)=\sum_{i=1}^{\alpha} n^i x$. Choose x such that $M-\frac{n^{n+1}-1}{n-1}x=M-x(n)=C$ where n is the number of sensors. For any $\alpha(\alpha = n, n-1,...,1)$ define Γ_{α}^1 to be the set of sensors whose values are greater than $M - x(n)$ and Γ_{α}^2 to be the set of the rest of the nodes. For x, there must exist a time *t* such that $H_t < M + x$; also, there must be some node whose value is less than $C = M - x(n)$ because *C* is the average value. Starting from sets Γ_{α}^1 and Γ_{α}^2 at time *t*, we have $\left|\Gamma_n^2\right| \ge 1$. After the first average operation for nodes that are in Γ_{α}^1 and Γ_{α}^2, we have $\left|\Gamma_{n-1}^2\right| \ge 2$. After the first average operation on nodes in Γ_{n-1}^1 and Γ_{n-1}^2, we have $\left|\Gamma_{n-2}^2\right| \ge 3$. So, this contradicts that the infimum of H_t^j is M .Therefore, we have $\lim_{t\to\infty} H_t^j = C$. In the same way, we can prove that $\lim_{t\to\infty} L_t^j = C$ .Combining these two results, we have that all the values on the sensors converge to *C.*

Provide a statement that what is expected, as stated in the "Introduction" chapter can ultimately result in "Results and Discussion" chapter, so there is compatibility. Moreover, it can also be added the prospect of the development of research results and application prospects of further studies into the next (based on result and discussion).

5.2. CTSP Versus TPSN

Assuming the average synchronization error of each hop is υ, the time to realize synchronization between one hop neighbors is almost identical results which are τ/ϖ. In the condition of the same network parameter, the synchronization error of CTSP and TPSN can be expressed as:

$$\begin{cases} \Delta_{CTSP} = n \cdot \upsilon / SS = 2W \cdot \upsilon \Big/ \left(SS \cdot R\sqrt{\pi\gamma N} \right) \\ \Delta_{TPSN} = \upsilon \cdot \sqrt{2}W \Big/ R \end{cases} \tag{13}$$

From comparison, it can be found that synchronous error of CTSP is greatly reduced. The computer simulation results of the TPSN and CTSP algorithm are given for comparison, by means of the Figure 4 we can find that, with the wireless hops increasing, the synchronization convergent rate increase as logarithm mode approximately.

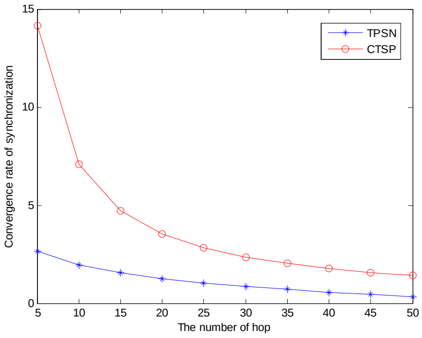

Figure 4. Synchronization Convergent Rate

The synchronization proportion valve and keeping the synchronization time of the two algorithms with different network scale conditions are given in Figure 5 and Figure 6 Through the contrast and comparison of synchronization proportion valve of different network scales, we can find the CTSP algorithm always provides an effective technique to synchronize for all kinds of network under various scenarios. As well as the synchronization performance of TPSN algorithm is relatively higher by the influence of the variety of the network scale. Therefore, CTSP algorithm can adapt to the change of network scale very well.

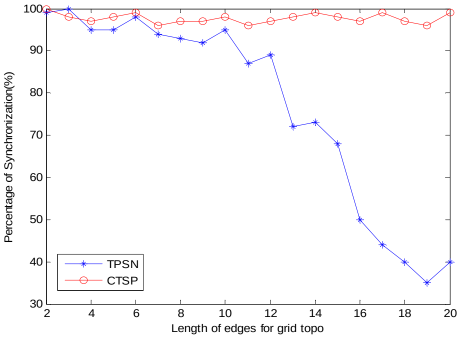

Figure 5. Synchronization Proportion

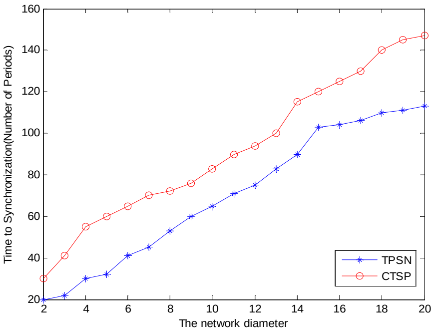

Figure 6. Holding Synchronization Time

6. Conclusion

In this paper, a cooperative synchronization protocol, named CTSP, has been proposed to solve the problems associated with low accuracy and poor scalability, which widely exist in most clocks in WSNs. The pulse-coupled is used for clock tick synchronizing, and the network nodes at the same time with distributed cooperation diffusion are presented. And besides, the convergence of algorithm is analyzed in theoretical, and the synchronization error expression is given. That directly proves the better synchronization error performance of CTSP. The simulation results demonstrate the correctness of theoretical analysis and CTSP can effectively hold longer synchronization time, improve the rate of convergence of synchronization, and significantly reduce the synchronization error. As our future work, it is interesting to evaluate the performance of the proposed CTSP in a network-wide scenario with a dynamic topology

References

[1] Daojing He, Lin Cui, Hejiao Huang. Design and verification of enhanced secure localization scheme in wireless sensor networks. *IEEE Transactions on Parallel and Distributed Systems.* 2009; 20(7): 1050-1058.
[2] Yik-Chung Wu, Qasim Chaudhari, Erchin Serpedin. Clock synchronization of wireless sensor networks. *IEEE Signal Processing Magazine.* 2011; 28(1): 124-138.
[3] Duzun Dong, Xiangke Liao, Yunhao Liu. Edge self-monitoring for wireless sensor networks [J].*IEEE Transaction on Parallel and Distribed Systems.* 2012; 23(3): 514-527.
[4] Wai-Leong Yeow, Chen-Khong Tham, Wai-Choong Wong. Energy efficient multiple target tracking in wireless sensor networks. *IEEE Transactions on Vehicular Technology*. 2007; 56(2): 918–928.
[5] Jiming Chen, Kejie Cao, Keyong Li. Distributed sensor activation algorithm for target tracking with binary sensor networks. *Cluster Computing*. 2011; 14(1): 55-64.
[6] Jeremy Elson, Kay Romer. Wireless sensor networks: A new regime for time synchronization. *Computer Communication Review*. 2003; 33(1): 149–154.
[7] Jeremy Elson, Lewis Girod, Deborah Estrin. *Fine-grained network time synchronization using reference broadcasts. 5th Symposium on Operating Systems Design and Implementation.* 2002: 147-163.
[8] Saurabh Ganeriwal, Ram Kumar, Mani B Srivastava. Timing-sync protocol for sensor networks. *Proceeding of First International Conference on Embedded Networked Sensor Systems*. 2003: 138-149.
[9] Ping Su. Delay measurement time synchronization for wireless sensor networks. *Intel Research*. 2003.
[10] Shibo He, Jiming Chen, Youxian Sun. On optimal information capture by energy-constrained mobile sensor. *IEEE Transactions on Vehicular Technology*. 2010; 59(5): 39–49.
[11] Hu An-Swol, Sergio D Servetto. On the scalability of cooperative time synchronization in pulse-connected networks. *IEEE Transactions on Information Theory*. 2006; 52(6): 2725-2748.
[12] Byoung-Kug Kim, Sung-Hwa Hong, Kyeong Hur. Energy-efficient and rapid time synchronization for wireless sensor networks. *IEEE Transactions on Consumer Electronics*. 2010; 56(4): 2258-2266.
[13] Ill-Keun Rhee, Jaehan Lee; Jangsub Kim. Clock synchronization in wireless sensor networks: An overview. *Sensors*. 2009; 9(1): 56–85.
[14] Nicolas Marechal, Jean-Benoit Pierrot, Jean-Marie Gorce. Fine synchronization for wireless sensor networks using gossip averaging algorithms. *EEE International Conference on Communication*s. 2008: 4963–4967.
[15] Luca Schenato, Federico Fiorentin. Average timesynch: a consensus-based protocol for time synchronization in wireless sensor networks. *Automatica.* 2011; 47(9): 1878-1886.
[16] Philipp Sommer, Roger Wattenhofer. *Gradient clock synchronization in wireless sensor networks*. Proceedings of IPSN. 2009: 37-48.
[17] Roberto Pagliari, Anna Scaglione. Scalable network synchronization with pulse-coupled oscillators. *IEEE Transactions on Mobile Computing*. 2011; 10(3): 392-405.
[18] He Jianping, Cheng Peng, Shi Ling, Chen Jiming. Time synchronization in WSNs: A maximum value based consensus approach. *Proceedings of the IEEE Conference on Decision and Control*. 2011: 7882–7887.
[19] Alekeish Khaled, Ezhilchelvan Paul. Consensus in Sparse, Mobile Ad Hoc Networks*. IEEE Transactions on Parallel and Distributed Systems*. 2012; 23(3): 467–474.
[20] Fumito Mori, Takashi Odagaki. Synchronization of coupled oscillators on small-world networks. *Physica D: Nonlinear Phenomena.* 2009; 238(14): 1180-1185.
[21] Lucarelli Dennis, Wang IJeng. *Decentralized synchronization protocols with nearest neighbor communication*. Proc. Second Int'l Conf. Embedded Networked Sensor Systems (SenSys '04). 2004: 62-68.
[22] Wu Jianshe, Jiao Licheng, Ding Ranran. Average time synchronization in wireless sensor networks by pairwise messages. *Computer Communications*. 2012; 35(2): 221–233.
[23] Xiao Y Wang, Rajeev K Dokania, Alyssa Apsel. PCO-based synchronization for cognitive duty-cycled impulse radio sensor networks. *IEEE Sensors Journal.* 2011; 11(3): 555-564.
[24] Mengfan Cheng, Hanping Hu. Impulsive synchronization of chaotic systems with time-varying parameters. *Journal of Computational Information Systems.* 2012; 8(11): 4759-4768.
[25] Jiming Chen, Qing Yu, Yan Zhang. Feedback-based clock synchronization in wireless sensor networks: A control theoretic approach. *IEEE Transactions on Vehicular Technology.* 2010; 59(6): 2963-2973.