International Journal of Future Generation Communication and Networking
Vol.7, No.5 (2014), pp.11-22
http://dx.doi.org/10.14257/ijfgcn.2014.7.5.02

Clock Self-Synchronization Protocol based on Distributed Diffusion for Wireless Sensor Networks

Min Li [1], Guoqiang Zheng [1]and Jishun Li [2]

[1]*Information Engineering College, Henan University of Science and Technology, Luoyang 471023, China*
[2] *Henan Key Laboratory for Machinery Design and Transmission System, Henan University of Science and Technology, Henan Luoyang, China*
[1]*limin0919@163.com,lykaroline@gmail.com*

Abstract

Aimed at the low precision and scalability of synchronization protocol in wireless sensor networks (WSNs), this paper proposes a clock self-synchronization protocol based on distributed diffusion (DDCSS). According to the energy, distribution and the average transmission delay of nodes, DDCSS chooses a set of master-node and diffusion-nodes which are used to perform local diffusion in each round. And then the average clock of master-node field is spread to the surrounding nodes. After that the surrounding nodes update the local clock by receiving the average clock. Eventually through several rounds of election master-nodes and diffusion-nodes, networks implement mutual diffusion and the node clock is approximate synchronization the average clock of network nodes. The simulation results demonstrate that, compared with the traditional synchronization methods, DDCSS can synchronize the network quickly with good precision convergence speed and scalability.

Keywords: *Wireless sensor networks, Distributed diffusion, Clock self-synchronization*

1. Introduction

Time synchronization for wireless sensor network is a topic of research that has been explored in a wide variety of applications, specifically in target tracking, intrusion detection, industrial automation and other fields. The power supply of wireless sensor networks is an important issue to save energy. Considering from the perspective of improving energy efficiency, the whole network clock synchronization also have an important role in energy saving. Therefore, more and more attention has been attracted to achieve global synchronization for wireless sensor networks [1-2].

In recent years, numerous synchronization protocols have been proposed, focusing on different aspects of the synchronization problem in WSNs [3]. 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 [4-6] 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. The protocol [4] is based on gossip averaging algorithms, and the protocols GTSP in [5] and ATS in [6] are based on average consensus algorithm, which have slow convergence speed as pointed in [7]. The study of consensus for sparse, mobile Ad Hoc networks is proposed in [8]. Recent results in [9] show that the approximate model used 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

ISSN: 2233-7857 IJFGCN

average value within the network, are time-consuming. In addition, with the expansion of the network size, there is in need of more and more time to achieve network Synchronization process.

Unlike the work in these protocols, our new DDCSS implementation is a fully distributed diffusion-based protocol. According to the energy, distribution and the average transmission delay of nodes, DDCSS dynamic chooses a set of master-node and diffusion-nodes. And then the average clock of nodes which in master-node field was spread to a limited number of hops based on message exchange between pairs of nodes. The clocks of all the network nodes were approximately synchronized to the average clock of nodes by inter-diffusion between the nodes. With the result that the synchronization error of the whole network is smoother, the algorithm is simpler, the cost is further reduced and the speed of constringency is faster.

2. Summarize the Protocol Algorithm

Like the traditional large-scale system, time synchronization of the sensor network produces a certain synchronization error because of temperature, the frequency noise, phase noise, the asymmetric link delay, *etc* [10-11]. In addition, the characteristics of self-organization and limited resources in sensor networks also need to be considered. Therefore, DDCSS is a self-organization protocol and hasn't precise time server, and the objective of synchronization is not only to make the entire network maintain a relatively consistent time but also neighbor nodes with smaller synchronization error. But there will be produced bigger difference in different node clock for clock drifting in a certain period of time, which the entire network has synchronized to a relatively consistent time. So, the equipment adopts the periodic synchronization according to the characteristics of node clock. In addition to the entire network can maintain a relatively consistent time, the network time has to be converted to standard time general when the users use it. So, the synchronization system structure of building is necessary in Figure 1. Time of the sensing region is converted to an universal standard time by SINK nodes which using time conversion algorithm, nodes of the sensing region maintains a relatively consistent time by performing DDCSS as shown in Figure 1.

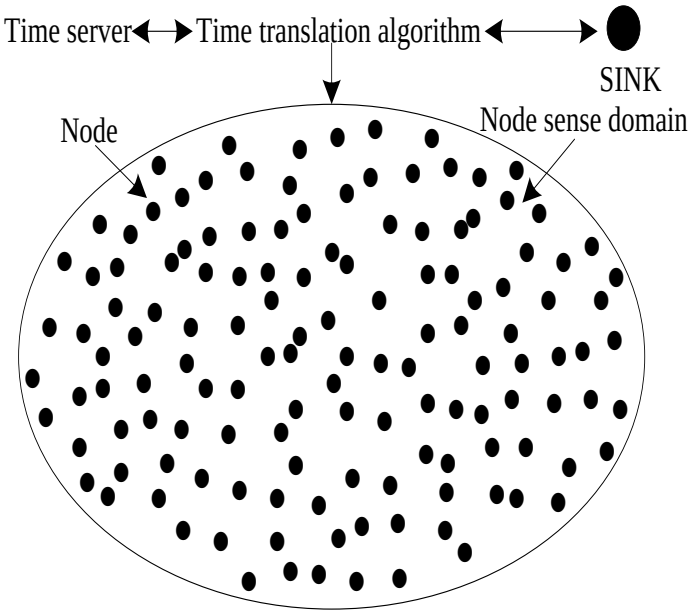

Figure 1. The System Structure

Figure 2 shows each round synchronous execution process of DDCSS. Firstly, with two-way message exchange models, all neighbor nodes clock in the master-node domain is obtained, and the average clock of nodes in broadcast domain is calculated. Secondly, the master-node is synchronized to the average clock of all nodes in the master-node broadcast domain, and then defines the master's clock as reference clock. 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 *m* cycle according to the process of implementation, the network will complete synchronization process at a time.

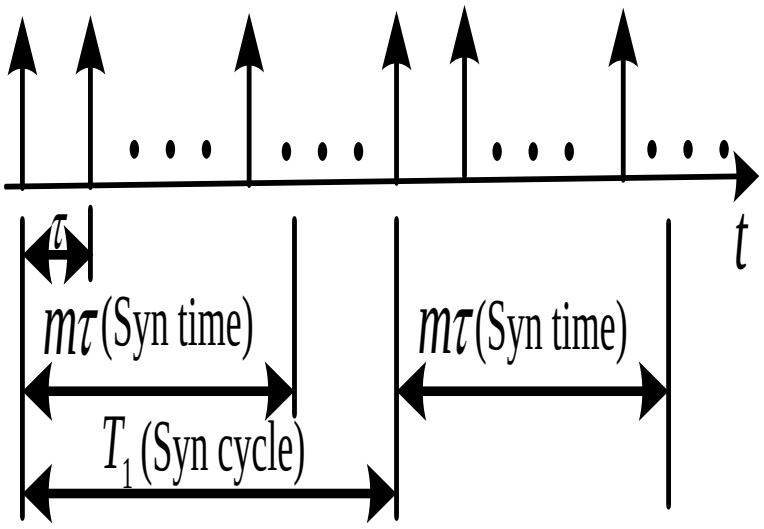

Figure 2. Cycle Implementation of Synchronization Protocol

2.1. The Algorithm for the Master-node's Election

Node energy and distribution of these two factors must be considered for the master-node election. In order to make all nodes maintain in a relatively balanced residual energy state, and besides increase the lifetime and local connectivity of the network, it is possible to elect the master-nodes which have more residual energy. At the same time, for making master-nodes have a certain independent covering range and the diffusion can be executed parallel in the clock diffusion process, we must consider the distribution of master-nodes. The election of master-node must meet the following two rules.

Rule 1. Assume that each node maintains a 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:

$$\zeta = \lambda - (1 - \delta) \tag{1}$$

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

master-nodes for network nodes is $\gamma = 1 - \phi$. In order to make the number of master-nodes relatively stable during each round, each node maintains a threshold value after each round of synchronous press type adjust:

$$\varphi = \varphi - \varepsilon \tag{2}$$

where ε is the quantity of every clock to adjust and depends on the average energy consumption of each node in per round synchronization period.

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, 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 neighbor nodes in broadcast range receive only the statement message, node that sent statement message executes synchronization after waiting for a certain time. The master-nodes election is multi-cycle and each τ will be re-election to the master-nodes in synchronous time $m\tau$.

2.2. The Average Clock 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 clock 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 node's clock at time l , where c_1^l is the master-node's clock. The acquisition process of average clock in master-node broadcast domain as follows:

Step1) The master-node broadcasts a ch-quest packet (including the sync-start of start clock synchronization, the master-node ID, the local clock value) to start a new cycle of synchronization;

Step2) According to the ch-quest packet, the neighbor nodes send ACK response packet containing a timestamp (including the local clock value when node receiving ch-quest packet, the local clock value when node transmitting ACK packet, the neighbor nodes ID) after certain clock of random waiting;

Step3) 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 clock 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 A the timestamp information from neighbor nodes. All of which show the master-node obtains the clock information of all the neighbor nodes. Figure 3 shows the master-nodebroadcasts the start clock synchronization packets at time l , according to the described steps to implement the information exchange process.

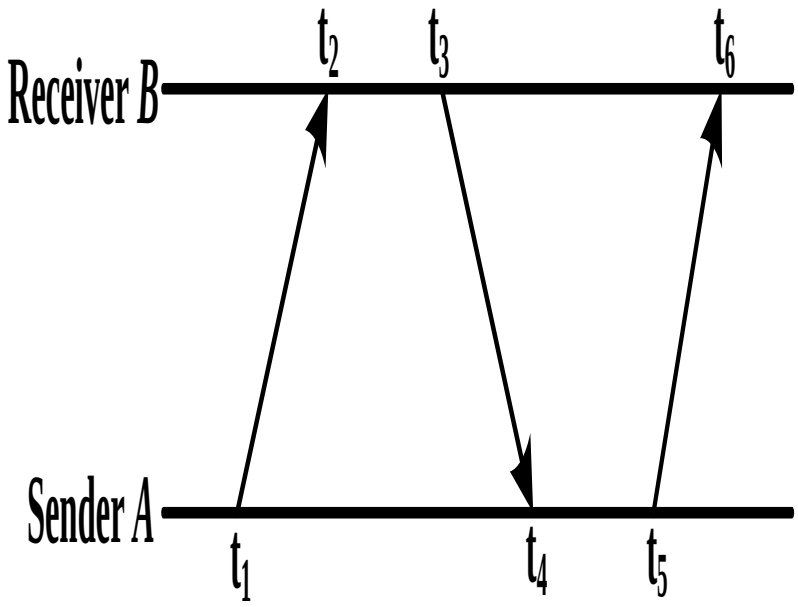

Figure 3. A Sender *A*-receiver *B* Two-way Timing Exchange Model

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

$$t_2 = t_1 + \Delta_k + d_k \tag{3}$$

$$\Delta_k = \frac{(t_2 - t_1) - (t_4 - t_3)}{2} \quad ; \quad d_k = \frac{(t_2 - t_1) + (t_4 - t_3)}{2} \tag{4}$$

At time l, the neighbor nodes that lasts $c_1^l = t_1$ clock time can be written as:

$$c_k^l = c_1^l + \Delta_k \tag{5}$$

Step4) From (4) and (5), all nodes average clock $\overline{c_j^l}$ and average propagation delay $\overline{d_j}$ within the master-node broadcast domain at time l take the form

$$\overline{c_j^l} = \sum_{k=1}^{n_j} c_k^l / n_j = c_1^l + \sum_{k=1}^{n_j} \Delta_k \Big/ n_j \; ; \; \overline{d_j} = \sum_{k=1}^{n_j} d_k / n_j \tag{6}$$

2.3. The Selection of Diffusion-Nodes and the Diffusion of Average clock

Using a reference clock of the master-node's average clock, the concrete selection rules of diffusion-nodes as follows:

Rule 3. After receiving the ch-quest packet of first-order master-node or diffusion-node, node produces random number η , where $\eta \in (0,1)$. $\zeta = \eta \text{-} (1-\delta)$ is calculated according to the method of election the master-node and node energy. If $\zeta \geq \phi_1$, 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 $\phi_1 = \phi_1 - \nu$ 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

$$d_k > \overline{d_j^{\varpi_L}} \tag{7}$$

the node can be diffusion-node, otherwise cannot become the diffusion-node. 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-nod. Clock spreads out from the master-nodes and the diffusion process of average clock is as follows:

At the beginning of each round of synchronization, if statement initiated synchronization node success, node is referred to as the master-node or 0-diffusion nodes. The neighbor nodes which meet the diffusion-nodes election rules called the 1-diffusion nodes in the master-node broadcast domain. After 1-diffusion nodes have implemented the diffusion of average clock information, the neighbor nodes which renewing consent average synchronous clock information are called 2-diffusion nodes. In a similar way, after *f-1* level（*f-1*≤*n-2*）diffusion-nodes have implemented the diffusion of average clock information, the neighbor nodes which renewing consent average synchronous clock information are called *f* level diffusion-nodes. The clock starts from the master-node diffusion, if t_k is not consider at the passage of node information exchange period, the diffusion process of average clock in the master-node domain 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 clock transmission hop number ϖ , each diffusion clock 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 clock 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 clock follows.

$$\overline{c_j^l} = \overline{c_j^l} + d_{0,k} \tag{8}$$

where $d_{0,k}$ stands for information propagation delay between the current diffusion-node and its master-node; the receiving average clock 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 $d_{1,k}$,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.

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

$$\sqrt{2}W/R \geq \varpi = 2W/(R\sqrt{\pi\gamma N}) \geq 2 \; ; \quad SS^{m}\varpi \geq \sqrt{2}W/R \tag{9}$$

where W is network coverage area, R is the communication radius, N is the number of nodes, γ is percentage of nodes by master-node.

The master-node ID is set to avoid repeatedly receiving the clock synchronization information from the same master-node domain. During a certain diffusion nodes performing diffusion, the neighbor nodes have received the average clock 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 clock diffusion information from the same master-node, the diffusion-node will end the diffusion process after having been waited on for a certain time.

2.4. The Clock Information Update of DDCSS

According to the received average clock diffusion packets of the master-node in the each cycle of synchronization time, the node calculates the clock.

Rule 5. If the average clock synchronization diffusion packet is the first received by node from a master-node in each synchronization period, T_{local} can express the local clock, this research gives the new clock T_{new}:

$$T_{new} = \overline{c_M(t)} + d_k \tag{10}$$

If $|T_{new}\text{-}T_{local}| > \delta$, the update of node is such that T_{new}, otherwise to maintain local clock T_{local} at constant value;

Rule 6. If a node receives a packet synchronization diffusion of L master-nodes and synchronization diffusion packets of different master-nodes, T_{new} has been obtained according to receiving the message node's local clock T_{local}, synchronous clock information $\overline{c_M(t)}$ and the single-hop delay d_k which between the diffusion nodes:

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

If $|T_{new}\text{-}T_{local}| > \delta$, the update of node is such that T_{new}, otherwise to maintain local clock T_{local} at constant value.

3. Simulation Results and Discussion

In this section, the convergence and performance of DDCSS scheme is demonstrated, and besides DDCSS comparison with RBS and TPSN protocol just described.

3.1. Analyzing the Convergence of DDCSS

Suppose that the deviations for all the nodes clock and standard time are uniform distributed in $[L_t, H_t]$ at time t, DDCSS satisfies convergence theorem as follow.

Theorem 1. For large-scale sensor network, DDCSS 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 clock 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 m set, the arbitrary node clock $c_n^m(l)$ satisfies:

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

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

$$M - (n^{n+1} - 1)x/(n-1) = M - x(n) = C \quad (13)$$

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 $|\Gamma_n^2| \ge 1$. After the first average operation for nodes that are in Γ_α^1 and Γ_α^2, we have $|\Gamma_{n-1}^2| \ge 2$. After the first average operation on nodes in Γ_{n-1}^1 and Γ_{n-1}^2, we have $|\Gamma_{n-2}^2| \ge 3$. 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*.

3.2 Comparison with RBS and TPSN

Given the limitations of testing environment, we needed to validate the scalability and convergence of DDCSS by comparing it with other decentralized synchronization algorithms that run at the application layer. We chose the decentralized implementation of the RBS and TPSN protocol for comparison. Assume that N nodes have high density in the rectangular area of $W \times W$ according to a uniform probability distribution.

The computer simulation results of the other schemes and DDCSS algorithm are given for comparison, by means of the Figure 4 we can find that, the synchronization convergent time of the three algorithms increased with the increasing of network scale. And the synchronization convergent time of RBS is longest than others, DDCSS can is speed up the convergence of nodes in short time. Figure 5 shows the cost results for the three algorithms in different network size. Since simultaneous consumption of energy is distributed to every node in the network, the more transmissions RBS and TPSN will require, and the more savings on communication overhead DDCSS will achieve.

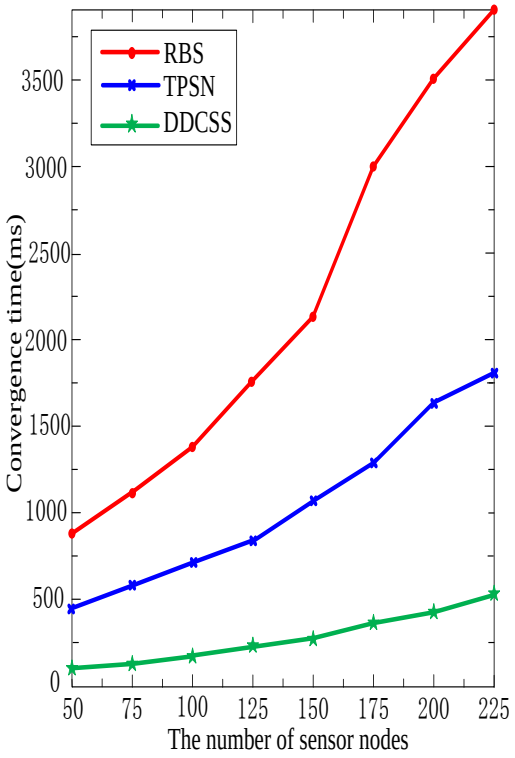

Figure 4. Synchronization Convergent Time

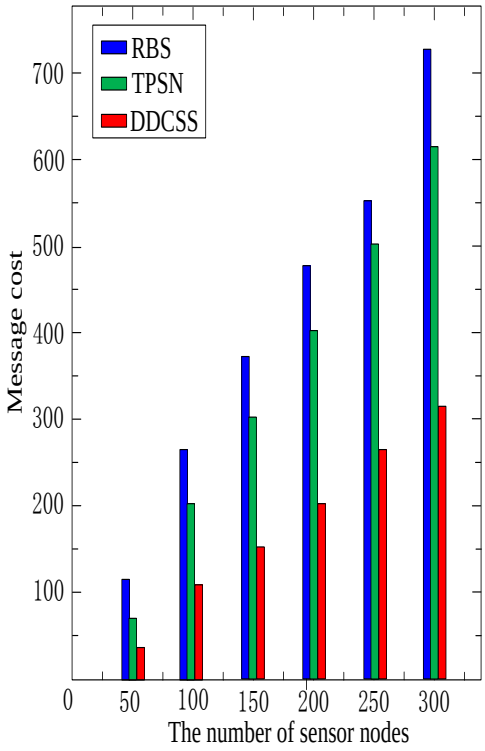

Figure 5. Communication Overhead

The average synchronization errors with multihop communications are plotted in Figure 6. This figure shows that the average synchronization error under RBS and TPSN is always larger than that under DDCSS for different hop distances. In addition, Figure 6 shows that the average synchronization error under TPSN increases as the hop distance increases, whereas the average synchronization error under DDCSS is almost the same for different hop distances. For RBS and TPSN, a certain amount of synchronization error is introduced for each hop, and the error accumulates as hop distance increases. Since the sensors take the average clock diffusion, there is no accumulation of synchronization errors in DDCSS, and thus, the errors for different hops are about the same. This experiment demonstrates an advantage of using DDCSS, *i.e.*, DDCSS algorithm can adapct to the change of network scale very well.

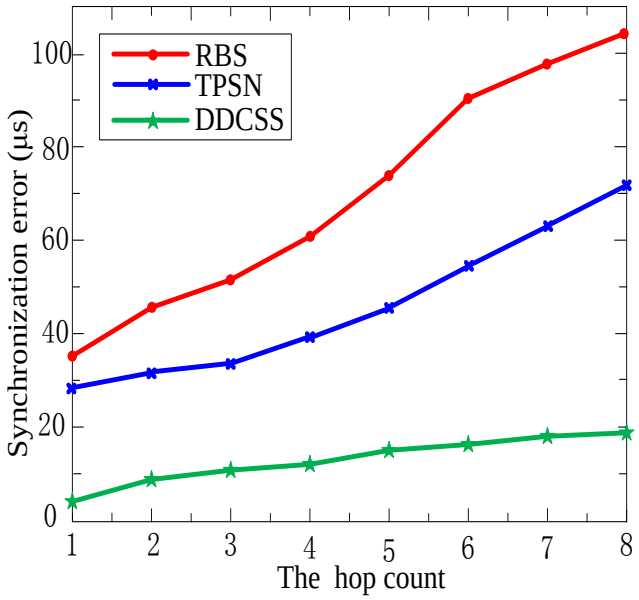

Figure 6. Average Synchronization Error for Multichip Communications

4. Conclusion

In this paper, we have presented clock self-synchronization protocol based on distributed diffusion for WSNs. The main advantage of this protocol is self-organizing realizing network synchronization not depending on any specific nodes, the energy within the feeling network synchronous process depletion is distributed to every node in the network. Our experiments show that the clock synchronization scheme has better accuracy than RBS and TPSN. The performance analysis shows that our scheme achieves significant savings on communication overhead. So the protocol is adaptable to unstructured environment, high reliability, the algorithm is simple, and can adjusts the synchronization parameters according to the demand of network application, which more suitable for large-scale wireless sensor networks.

Acknowledgments

This project is supported by the National Science & Technology Major Program of China (Grant No. 2012ZX04004011) ， the Project of Basic and Advanced Technology Research of Henan Province of China (No. 122300413209). Education Department of Henan Province Key Science and Technology Research Project (12B510010).

References

[1] Wu, Yik-Chung, Q. Chaudhari and E. Serpedin, “Clock synchronization of wireless sensor networks”, IEEE Signal Processing Magazine, vol. 28, no. 1, **(2011)**, pp. 124-138.
[2] D. Dong, X. Liao and Y. Liu, *et al*, “Edge self-monitoring for wireless sensor networks”, IEEE Transaction on Parallel and Distribed Systems, vol. 23,no. 3,**(2012)**, pp. 514-527.
[3] I. K. Rhee, J. Lee and J. Kim, *et al*, “Clock synchronization in wireless sensor networks: An overview”, Sensors, vol. 9, no. 1, **(2009)**, pp. 56-85.
[4] N. Maréchal, J. B. Pierrot and J. M. Gorce, “Fine synchronization for wireless sensor networks using gossip averaging algorithms”, IEEE International Conference on Communications, (**2008)**, pp. 4963-4967.
[5] L. Schenato and F. Fiorentin, “Average timesynch: a consensus-based protocol for time synchronization in wireless sensor networks”, Automatica, vol. 47, no. 9, **(2011)**, pp. 1878-1886.
[6] P. Sommer and R. Wattenhofer, “Gradient clock synchronization in wireless sensor networks”, In Proceedings of IPSN, (**2009)**, pp. 37-48.

[7] R. Pagliari and A. Scaglione, "Scalable network synchronization with pulse-coupled oscillators", IEEE Transactions on Mobile Computing, vol. 10, no. 3, **(2011)**, pp. 392-405.
[8] J. He, P. Cheng and L. Shi, *et al*, "Time synchronization in WSNs: A maximum value based consensus approach", Proceedings of the IEEE Conference on Decision and Control, (**2011)**, pp. 7882-7887.
[9] Alekeish, Khaled, and P. Ezhilchelvan, "Consensus in Sparse, Mobile Ad Hoc Networks," IEEE Transactions on Parallel and Distributed Systems, vol. 23, no. 3, **(2012)**, pp. 467-474.
[10] Mori, Fumito, and T. Odagaki "Synchronization of coupled oscillators on small-world networks", Physica D: Nonlinear Phenomena, vol. 238, no. 14, **(2009)**, pp. 1180-1185.
[11] Wang, Y. Xiao, R. K. Dokania and A. Apsel, "PCO-based synchronization for cognitive duty-cycled impulse radio sensor networks", IEEE Sensors Journal, vol. 11, no. 3, **(2011)**, pp. 555-564.

Authors

Min Li received the B.S. degree from Luoyang Institute of Science and Technology, Luoyang, China, in 2008. She is currently working toward the M.S degree with the Electronic Information Engineering, Henan University of Science and Technology, Luoyang, China. Her current research interests include synchronization techniques, cooperative communications for wireless sensor networks and wireless communication systems.

Guoqiang Zheng received the Ph.D. degree in Communication and information systems professional from Xi'an Jiao tong University, China, 2008. He is a professor at Henan University of Science and Technology on College of Electronic Information Engineering, China. His research interests include wireless communication technology, network communication protocol and software radio theory.

Jishun Li received the Ph.D. degree in Mechanical manufacturing and automation from Shanghai Jiao tong University, 1996. He is a professor at Henan University of Science and Technology on school of Electromechanical Engineering, China. He is currently the chief of Henan Key Laboratory for Machinery Design and Transmission System, China. His research interests include Mechanical and electrical integration, precision measurement theory, signal acquisition and processing, numerical control technology and CAE.

