IPHV7I20045X
International Journal Of Advanced Research and Innovation -Vol.7, Issue .II
ISSN Online: 2319 – 9253
Print: 2319 – 9245

CAPABLE ERA MANAGEMENT FOR PORTABLE FLOODED NETWORKS

#1SAICHITRA.D - M.Tech Pursuing,
#2 K S NAGESHWARA RAO -Assistant Professor,
Department of Computer Science & Engineering,
Malla Reddy Institute of Technology, Hyderabad.

Abstract : *Aiming at the problem of high propagation delay and node movement in underwater wireless sensor networks (UWSNs), this paper presents a Cluster-Wise Time Synchronization (CWTS) protocol. The time synchronization process of CWTS is divided into two phases: inter-cluster synchronization phase and intra-cluster synchronization phase, and part of them can be executed concurrently for reducing the number of packets generated in synchronization. Moreover, during the process of calculating clock skew and offset, CWTS distinguishes the propagation delay of downlink from that of uplink caused by node movement in order to improve time synchronization accuracy. We demonstrate through simulations that CWTS can reduce synchronization errors as well as energy consumption compared to traditional protocols.*

Keywords: *Time Synchronization, Underwater Wireless Sensor Networks, Synchronization Error*

1. Introduction

Underwater Wireless Sensor Networks (UWSNs) have a lot of potential application areas such as oceanographic data collection, disaster prevention, pollution monitoring, offshore exploration, and military surveillance [1-3]. UWSN is a new area of wireless sensor network in underwater environments which has challenges to be overcome such as long propagation delay, severely limited bandwidth and unpredictable node movement [4-6]. All the above distinct features of UWSNs give birth to new challenging areas for every level of the network protocol suite. UWSNs consist of a variable number of sensors and vehicles that are deployed to perform collaborative monitoring tasks over a given area. To achieve this objective, sensors and vehicles self-organize in an autonomous network in order to adapt to the characteristics of the underwater environment. Many problems arise with UWSNs that need to be solved in order to enable underwater monitoring in the new environment. Among them, providing clock synchronization is a well known fundamental issue due to the unique characteristics of UWSNs [7][8]. A number of important network functionalities require that nodes have a common notion of time, including time stamping of events, distributed data aggregation, MAC and localization. Meanwhile, numerous time synchronization protocols have been proposed for terrestrial wireless sensor networks, such as RBS [9], TPSN [10] and FSTP [11]. However, they cannot be directly applied to UWSNs, because most of them assume negligible propagation delay among sensor nodes, which is not true in UWSNs. UWSNs must rely on underwater acoustic communications because high-frequency radio signals used in traditional ground based sensor networks can be rapidly absorbed by water. Compared with terrestrial wireless sensor networks, the channel of UWSNs have the following characteristics: 1) long and unstable propagation delay; 2) the movements of underwater sensors are driven not only by the float of water, but also by other uncertain factors such as tides, animal interference and ships, which further complicates time synchronization by introducing dynamic propagation delays; 3) constrained energy due to limited power supply restricts the time synchronization overhead [12][13]. Therefore, many research results in land-based sensor networks as well as traditional Ad hoc networks cannot be applied to UWSNs directly, which requires new time synchronization protocol to be designed for the new features of the UWSNs in order to ensure that the performance of UWSNs can meet the actual underwater environmental needs.

In this paper, we propose a novel Cluster-Wise Time Synchronization (CWTS) protocol. CWTS consists of inter-cluster synchronization process and intra-cluster synchronization process, and part of them can be executed concurrently for reducing the number of packets generated in synchronization.

Moreover, during the process of calculating clock skew and offset, CWTS distinguish the propagation delay of downlink from that of uplink caused by node movement in order to improve time synchronization accuracy. Numerical simulations confirm the suitability of our protocol in terms of both synchronization accuracy and energy consumption.

The remainder of the paper is organized as follows: Section 2 presents a brief overview of related work. Section 3 introduces the network topology, while Section 4 presents the technical details of our time synchronization protocol. Performance evaluation is described in Section 5. Finally, we conclude the paper in Section 6.

2. Related work

In the literature, there are various time synchronization protocols for distributed systems like terrestrial radio sensor networks. RBS (Reference Broadcast Synchronization) [9] is a well-known receiver-receiver synchronization protocol. It synchronizes a set of receivers with one another by periodically sending beacon messages to their neighbors using the network's physical layer broadcast. Recipients use the message's arrival time as a point of reference for comparing their clocks. However, RBS requires extra message exchange to communicate the local timestamps between any two nodes which intend to become synchronized. TPSN (Timing-sync Protocol for Sensor Networks) [10] is based on the simplistic approach of conventional sender-receiver time synchronization, which employs a two-way message exchange for synchronization. Although TPSN takes care of propagation delays, it does not take the clock skew into account during synchronization period. Instead, it only computes offset, which severely limits its accuracy. FTSP (Flooding Time Synchronization Protocol) [11] does not rely on a fixed network hierarchy but updates it continuously, it supports network topology changes including mobile nodes. The applied flood-based communication protocol in FTSP provides a very robust network, but it is not applicable to underwater sensor network mainly because it also assumes instant reception. Additionally, it requires hardware calibration, which is not a completely software solution. TSHL (Time Synchronization for High Latency) [14] is the first time synchronization algorithm designed for high latency networks specifically. It uses one-way communication to estimate the clock skew and two-ways communication to estimate the clock offset. TSHL assumes underwater sensor networks are static and therefore suffers from sensor nodes mobility, especially when nodes move fast. D-Sync [15] provides an indication of the relative motion between nodes. In D-Sync, Doppler shift is used in order to infer the propagation delay and an ordinary least square estimator is applied to provide an estimate for both clock skew and offset. However, in deriving the estimators, communication channel variability is not taken into account, which reduces the accuracy of synchronization. MU-Sync [16] runs two times of linear regression to estimate the clock skew and clock offset for cluster based UWSNs. MU-Sync assumes that the one-way propagation delay can be estimated as the average round trip time which causes extra errors and has relative high message overhead. Although MU-Sync claims to be able to solve the mobility issue, it has relatively high message overhead. Moreover, MU-Sync applies half of the round trip time to compute one way propagation delay, which involves significant errors when sensor nodes move fast or regular nodes experience a long response time.

3. Network architecture

In this paper, we consider a cluster based underwater wireless sensor network architecture as shown in Figure 1. The network consists of three types of nodes: beacons, cluster-heads and ordinary nodes. Beacons have unlimited energy resources and perfect timing information. For example, they can synchronize to UTC (Universal Time Coordinated) time constantly using GPS services without recalibrating their atomic clocks or performing any synchronization algorithms. In this regard, they provide the time reference for the sensors positioned underwater. If there are two or more beacons, they can communicate with each other through radio frequency (RF) links. Beacons communicate with cluster-heads and ordinary nodes through acoustic links. Each cluster has and only has one cluster-head. All ordinary nodes connect to their cluster-head via single hop. The cluster-heads of different clusters connect to a beacon through multiple hops.

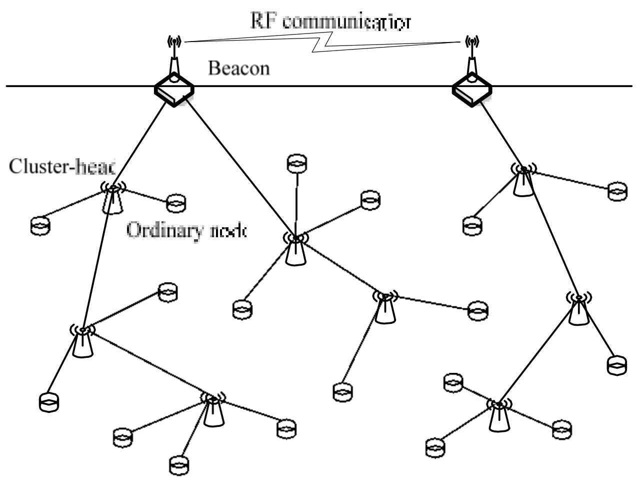

Figure 1. The architecture of a cluster based underwater wireless sensor network

We consider that UWSNs are composed of a large number of nodes uniformly scattered in monitoring fields and represented by $G=(V, E)$. The sensor nodes are assumed static, or they have low mobility with respect to signal propagation speed. Every sensor node has the same transmission range and each sensor node is able to communicate with other sensor nodes within its range. We also assume that each node is assigned with a triplet of coordinate (x, y, z), where each coordinate represents the hop distance of the node from one anchor. We assume that the set of n sensor nodes is represented as a set N with $n=|N|$. All sensor nodes have the same communication range of r, which represented as a sphere volume of radius r in UWSNs.

Definition 1. The function $\delta(u, v)$ defines the distance between two nodes n_u and n_v in a 3D Euclidean space as:

$$\delta: N\times N\rightarrow\Gamma:\delta(u, v),$$

$$\delta(u,v)=\sqrt{(u_x-v_x)^2+(u_y-v_y)^2+(u_z-v_z)^2} \qquad (1)$$

Two nodes n_u and n_v are neighbors and connected by a link if $\delta(u, v)< r$ and the link between n_u and n_v is denoted by $e(u,v)$. We construct the network topology with RNG (Relative Neighborhood Graph) [17], then two nodes become neighbor nodes if and only if for any arbitrary node n_p, $\delta(u, v)\leq\max\{\delta(p, u),\delta(p, v)\}$. For a three dimensional Euler space embedded, if the arcuate area formed by the intersection of two sphere centered at n_u and n_v (with radius $\delta(u, v)$) is empty, then n_u and n_v are adjacent nodes. RNG algorithm is simple and is easily built in a distributed way. There is no crossing edges in a RNG because at least one edge in any pair of crossing edges must be removed according to their definitions and the time complexity is $O(n^3)$. While constructing from a Delaunay Triangulation Graph structure, its complexity of lower bound is $O(n\log(n))$ [18]. In addition, a computational method with the complexity of $O(n^2)$ for the RNG in a three dimensional space is given in [19]. As underwater sensors float with currents, their movements are constrained in different horizontal planes and they are likely to maintain a steady position relative to each other. The construction of RNG does not require that the exact positions of nodes and their neighbors be known. For each node, only the corresponding mutual distances to its neighbors are required. Therefore, RNG is expected to be more suitable in modeling UWSN, which achieves more accurate results and behaves more consistently than other models.

4. Time synchronization protocol

The time synchronization of CWTS is divided into two phases: inter-cluster synchronization phase and intra-cluster synchronization phase. In the inter-cluster synchronization phase, cluster-heads synchronize themselves with beacons in a sender-receiver mode. In the intra-cluster synchronization phase, ordinary nodes synchronize themselves with cluster-heads in receiver-receiver mode. In order to reduce the number of packets generated in synchronization, part of them can be executed concurrently.

4.1. Data packet structure

The data structure used in CWTS includes timestamp, synchrnization request packet (Sync-Req) and request response packet (Sync-Ack). The timestamp consists of two 32-bits unsigned interges. One of them denotes second, the other represents nanosecond. Both of them are required from the MAC layer of local sensor node. Each Sync-Req is a 16-bytes packet, which consists of destination address, source address, identifier, hops, the number of iterations and timestamp T_1. Each Sync-Ack is a 32-bytes packet, which consists of destination address, source address, identifier, hops, the number of iterations, timestamp T_1, timestamp T_2 and timestamp T_3.

4.2. Time synchronization process

In order to perform time synchronization for pairs of clocks, most synchronization protocols rely on estimating the clock skew and offset, which presents the relation between the time measured by two different clocks. CWTS also yields to this approach as the follow equation:

$$T_A(t)=\alpha_a t+\beta_a, \qquad (2)$$

where $T_A(t)$ stands for the measured time of the sensor node A; t is the reference time (UTC time); α_a and β_a are the relative clock skew and offset respectively. The basic idea of time synchronization is that the unsynchronized nodes exchange packets with the reference node (already synchronized) in order to estimate clock skews and offsets in local clocks, so that they can correct their local times with that of UTC time, and maintain time synchronization by periodical updating their local clocks.

Figure 2 illustrates the process of time synchronization in CWTS protocol. At the beginning of synchronization, cluster-head *C* sends *Sync-Req* packets to beacon *B* and all ordinary nodes in its local cluster, such as node *A*. Ordinary node *A* is not need to reply the request packet. After a while, beacon *B* replies cluster-head *C* with a *Sync-Ack* packet. And then, cluster-head *C* will synchronize with beacon *B*. After that, cluster-head *C* broadcasts *Sync-Ack* packets to all ordinary nodes in its local cluster. At last, these ordinary nodes synchronize themselves with cluster-head *C*.

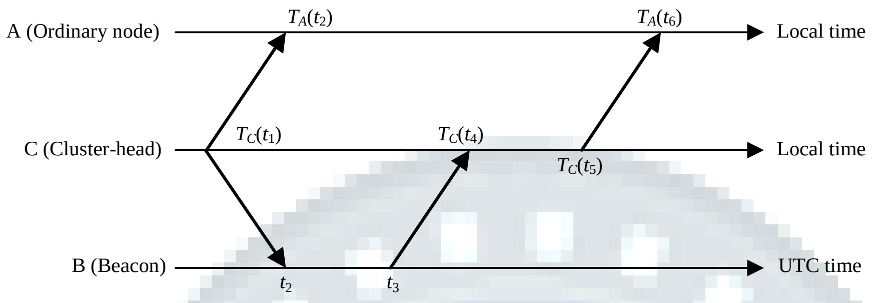

Figure 2. The process of synchronization in CWTS protocol

Let d_{CB} be the deterministic part of the message transmission delay between cluster-head *C* and beacon *B*. It can be pre-computed in the deployment phase of the network or estimated via an extra message exchange between cluster-head *C* and beacon *B*. On the other hand, the random delays in uplink and downlink are modeled as independent and identically distributed zero mean Gaussian random variables X_{CB} and Y_{CB} respectively, with common variance equal to σ^2. The UTC time t_2 can be calculated as:

$$t_2 \quad t_1 \quad d_{CB} \quad X_{CB}, \tag{3}$$

The timestamp of cluster head *C* at UTC time t_1 can be calculated as:

$$T_C(t_1) = \alpha_c(t_2 \quad d_{CB} - X_{CB}) + \beta_c, \tag{4}$$

Similarly, the UTC time t_4 can be calculated according as:

$$t_4 \quad t_3 + d_{CB} + Y_{CB}, \tag{5}$$

Moreover, the timestamp of cluster head *C* at UTC time t_4 can be calculated as:

$$T_C(t_4) = \alpha_c(t_3 \quad d_{CB} \quad Y_{CB}) + \beta_c, \tag{6}$$

Assuming that $t_3=t_2+\varepsilon$, $z_c=[(T_C(t_1)- t_2)+(T_C(t_1)- t_3)]/2$, then we get:

$$z_c=\alpha_c[t_2+\varepsilon/2+(Y_{CB}-X_{CB})/2]+\beta_c+(Y_{CB}-X_{CB})/2, \tag{7}$$

Let $w_c=(Y_{CB}-X_{CB})/2$, as $t_2>>\varepsilon/2+(Y_{CB}-X_{CB})/2$, then we get:

$$z_c \quad \alpha_c t_2 \quad \beta_c \quad w_c, \tag{8}$$

where w_c obeys Gaussian distribution of $N(0,\sigma^2/2)$.

After *k* iterations of equation (8) with the MAC timestamps, cluster head *C*'s clock skew and offset can be estimated.

After that, ordinary node *A* in the cluster of *C* calculates its estimated time t_1 as:

$$\hat{t}_1 = \frac{T_C(t_1) - \hat{\beta}_C}{1+\hat{\alpha}_C}, \tag{9}$$

Let d_{AC} denotes the deterministic part of the message transmission delay between ordinary node *A* and cluster-head *C* and the random portion of the downlink delay X_{AC} is a zero mean Gaussian random variable with variance σ^2, then the estimated UTC time t_1 of ordinary node *A* can be calculated as:

$$\hat{t}_2 = \hat{t}_1 + d_{AC} + X_{AC}, \tag{10}$$

The timestamp of ordinary node *A* at UTC time t_2 can be calculated as:

$$T_A(t_2) = \alpha_a \hat{t}_2 + \beta_a, \tag{11}$$

Applying equation (10) to equation (11), we can get:

$$T_A(t_2) = \alpha_a(\hat{t}_1 + d_{AC} + X_{AC}) + \beta_a, \tag{12}$$

Applying equation (9) to equation (12), we can get:

$$T_A(t_2) = \alpha_a(\frac{T_C(t_1) - \hat{\beta}_c}{1+\hat{\alpha}_c} + d_{AC} + X_{AC}) + \beta_a, \tag{13}$$

Similarly, we can also get the linear model of $T_A(t_6)$ and $T_C(t_5)$, which can be used to calculate ordinary node *A*'s clock skew and offset together with equation (13). After that, cluster head *C*'s neighboring clusters can synchronize their clock times by treating cluster head *C* as their reference node. This process is repeated until all sensor nodes in the network have been synchronized.

5. Performance evaluation

5.1. Simulation settings

We use Aqua-Sim [20] as simulation framework to evaluate our approach. The data packets are also generated by the simulator. Aqua-Sim is an ns-2 based underwater sensor network simulator developed by underwater sensor network lab at University of Connecticut. Aqua-Sim can simulate the attenuation and propagation of acoustic signals. It can also simulate packet collision in underwater sensor networks.

In simulations, we assume that beacons own UTC time and have unlimited energy resources. Other sensor nodes have their own internal clocks and the errors for processing and exchanging of packets among sensor nodes obey Gaussian distribution [21]. We use MAC layer timestamp during the process of simulation. The sensor nodes follow the random-walk mobility pattern. Each sensor node randomly selects a direction and moves to the new position with a random speed between the minimal speed and maximal speed, which are 1 m/s and 5 m/s respectively. They are sufficient for the simulation of current Autonomous Underwater Vehicles (AUVs) or other mobile nodes. Other simulation parameters and their default values are listed in Table 1.

We compared the performance of Cluster-Wise Time Synchronization (CWTS) protocol with that of TSHL [14] and MU-Sync [16]. Without loss of generality, every data point is obtained as the average of 1000 simulation runs.

Table 1. Simulation parameters and their default values

Simulation parameters	Default values
The number of beacons	3
The number of sensor nodes	50
The size of region	1 km×1 km×1 km
Average node degree	4.66
Initial clock skew	50 ppm
Initial clock offset	80 ppm
Time granularity	1 µs
Jitters in receivers	10 µs
Time interval of packets	1 s

5.2. Simulation results

In the first set of simulations, we compared the errors with time elapsed since synchronization in different synchronization protocols. As shown in Figure 3, the errors of three synchronization protocols are proportional to the time after synchronization. CWTS achieves lower errors than that of TSHL and MU-Sync in the same circumstances. The average error of CWTS is only 18.5% of TSHL and 47.1% of MU-Sync after synchronization of 10^5 seconds. The low precision of TSHL is caused by its big estimation error of clock skew. It assumes that the propagation delay in synchronization process is same, which is not true in mobile underwater sensor networks. The relative position between nodes changes with node movement and causes time-varying propagation delay. MU-Sync runs linear regression twice to calculate the clock skew and clock offset, but it applies half of the round trip time to compute one way propagation delay, which involves significant errors when sensor nodes move fast or regular nodes experience a long response time. Although the time point MU-Sync used in the second linear regression has decreased propagation delay, the local clock deviation still large because of the large propagation delay estimation error.

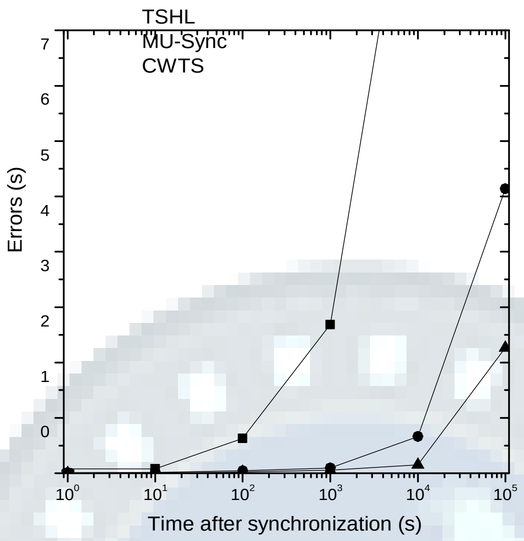

Figure 3. The comparison of errors vs. time after synchronization

In the second set of simulations, we compared the errors with average node speed in different synchronization protocols. The synchronization time is set to 10 seconds. As shown in Figure 4, TSHL obtains the highest synchronization error among the three synchronization protocols when their average node speeds are same. Moreover, with the increase of average node speed, the curve of TSHL shows a rapid rising compared with other synchronization protocols. The curve of MU-Sync indicates that the error is approximately linear with the average node speed. The error of CWTS is obviously lower than that of TSHL and MU-Sync. Furthermore, with the increase ofaverage node speed, the accumulated errors of CWTS display a mild growth. On average, the error of CWTS is 39.6% lower than that of TSHL and 29.1% lower than that of MU-Sync.

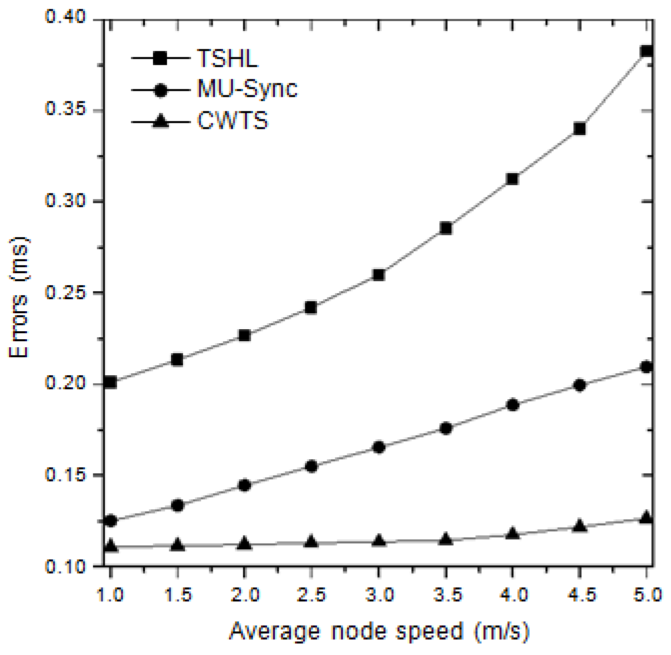

Figure 4. The comparison of errors vs. average node speed.

In the third set of simulations, we compared the number of packets with time elapsed since synchronization in different synchronization protocols. As shown in Figure 5, the number of packets is proportional to the time elapsed since synchronization. CWTS produces the minimum number of packets while MU-Sync generates the maximum number of packets. Moreover, the curve of CWTS indicates gentler slope compared with that of TSHL and MU-Sync. Overall, CWTS decreases 30.2% of the number of packets than that of TSHL and 42.3% of the number of packets than that of MU-Sync on average after time synchronization. This is because part of inter-cluster synchronization phase and intra-cluster synchronization phase in CWTS can be executed concurrently, which in turn reduces the number of packets generated in synchronization.

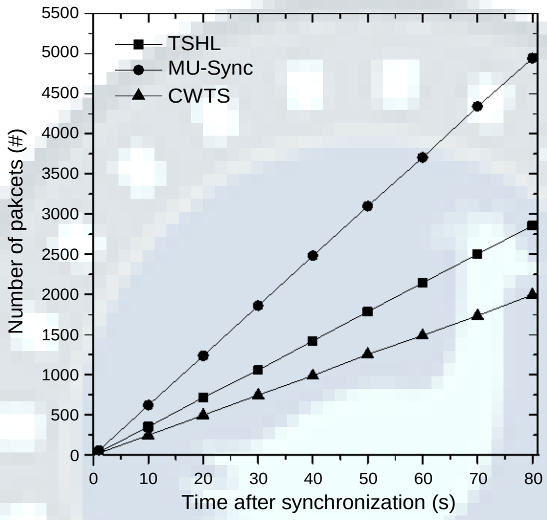

Figure 5. The comparison of number of packets vs. time after synchronization

In the last set of simulations, we compared the number of iterations with error tolerance in different synchronization protocols as shown in Figure 6. The amount of energy consumed by re-synchronization is represented by the number of iterations over a period of 10^5 seconds of operating time for various values of error tolerance. The number of iterations is inversely proportional to error tolerance in three synchronization protocols. Under the same condition, CWTS outperforms TSHL and MU-Sync due to its ability to estimate skew more accurately. On average, CWTS only needs 42.3% of iterations compared to TSHL and 20.4% of iterations compared to MU-Sync.

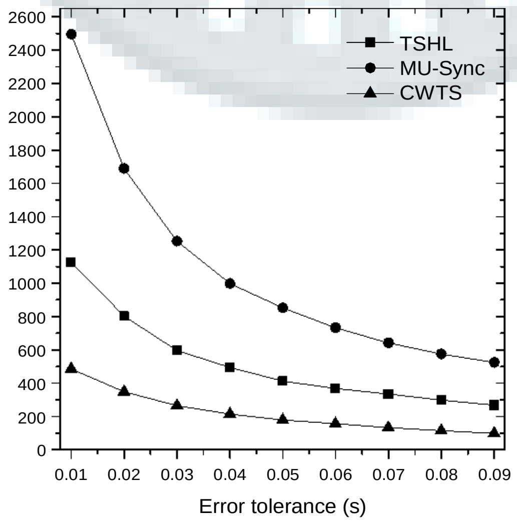

Figure 6. The comparison of number of iterations vs. error tolerance

6. Conclusions

This paper proposes a Cluster-Wise Time Synchronization (CWTS) protocol for underwater wireless sensor networks. The time synchronization process of CWTS is divided into two phases: inter-cluster synchronization phase and intra-cluster synchronization phase, and part of them can be executed concurrently for reducing the number of packets generated in synchronization. During the process of calculating clock skew and offset, CWTS distinguishes the propagation delay of downlink from that of uplink caused by node movement in order to improve time synchronization accuracy. We demonstrate through simulations that CWTS can reduce synchronization errors as well as the number of synchronization packets than traditional protocols, which in turn achieves the purpose of reducing energy consumption. In future work, we will investigate the influence of MAC layer activities to the performance of time synchronization protocols, such as packet loss and retransmission. Moreover, we will analysis the adaptability of our protocol and evaluate its performance through ocean experiments.

REFERENCES

[1] R. B. Manjula, S. M. Sunilkumar, "Issues in underwater acoustic sensor networks," International Journal of Computer and Electrical Engineering, vol. 3, no. 1, pp. 1793-8163, 2011.
[2] Y. G. Chen, X. M. Xu , L. Zhang, "Design of RC-LDPC codes and its application for shallow water acoustic communications," Journal of Convergence Information Technology, vol. 7, no. 12, pp. 177-185, 2012.
[3] W. J. Shen, H. X. Sun, E. Cheng,et al, "SNR estimation algorithm based on pilot symbols for DFT-Spread OFDM systems over underwater acoustic channels," Journal of Convergence Information Technology, vol. 6, no. 2, pp. 191-196, 2011.

[4] S. Basagni, C. Petrioli, R. Petroccia, M. Stojanovic, "Optimized packet size selection in underwater wireless sensor network communications," IEEE Journal of Oceanic Engineering, vol. 37, no. 3, pp.321-337, 2012.
[5] J. M. Jornet, M. Stojanovic, M. Zorzi, "On joint frequency and power allocation in a cross-layer protocol for underwater acoustic networks," IEEE Journal of Oceanic Engineering, vol. 35, no. 4, pp.936-947, 2010.
[6] C. Detweiler, M. Doniec, I. Vasilescu, D. Rus, "Autonomous depth adjustment for underwater sensor networks: design and applications," IEEE/ASME Transactions on Mechatronics, vol. 17, no. 1, pp. 16-24, 2012.
[7] I. F. Akyildiz, D. Pompili, T. Melodia, "Underwater acoustic sensor networks: research challenges," Ad Hoc Networks, vol. 3, no. 3, pp. 257-279, 2005.
[8] J. Liu, Z. Zhou, Z. Peng, et al, "Mobi-Sync: efficient time synchronization for mobile underwater sensor networks," In Proceedings of the ACM International Workshop on Underwater Networks, pp.1-5, 2009.
[9] J. Elson, L. Girod, D. Estrin, "Fine-grained network time synchronization using reference broadcasts," In Proceedings of the Fifth Symposium on Operating Systems Design and Implementation, pp. 147-163, 2002.
[10] S. Ganeriwal, R. Kumar, M. Srivastava, "Timing-sync protocol for sensor networks," In Proceedings of the First International Conference on Embedded Networked Sensor Systems, pp.138-149, 2003.
[11] M. Maroti, B. Kusy, G. Simon, et al, "The flooding time synchronization protocol," In Proceedings of the Second International ACM Conference on Embedded Networked Sensor Systems, pp.39-49, 2004.
[12] A. Y. Teymorian, W. Cheng, L. Ma, et al,"3D Underwater sensor network localization," IEEE Transactions on Mobile Computing, vol. 8, no. 12, pp.1610-1621, 2009.
[13] G. Isbitiren, O. B. Akan, "Three-dimensional underwater target tracking with acoustic sensor networks," IEEE Transactions on Vehicular Technology, vol. 60, no. 8, pp.3897-3906, 2011.
[14] A. Syed, J. Heidemann, "Time synchronization for high latency acoustic networks," In Proceedings of the International Conference on Computer Communications, pp.1-12, 2006.
[15] F. Lu, D. Mirza, C. Schurgers, "D-sync: doppler-based time synchronization for mobile underwater sensor networks," In Proceedings of the Fifth International Workshop on UnderWater Networks, pp.1-8, 2010.
[16] N. Chirdchoo, W. Soh, K. Chua, "Mu-sync: A time synchronization protocol for underwater mobile networks," In Proceedings of the Third ACM International Workshop on UnderWater Networks, pp. 35-42, 2008.
[17] G. Toussaint, "The relative neighborhood graph of a finite planar set", Pattern Recognition, vol. 12, no. 4, pp. 261-268, 1980.
[18] P. Bose, L. Devroye, W. Evans, D. Kirkpatrick, "On the spanning ratio of gabriel graphs and β-skeletons," Journal on Discrete Mathematics, vol. 20, no. 2, pp. 412-427, 2006.
[19] K. Supowit, "The relative neighborhood graph with an application to minimum spanning trees," Journal of Association for Computing Machinery, vol. 30, no. 3, pp. 428-448, 1983.