This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE ICC 2009 proceedings

Tri-Message: A Lightweight Time Synchronization Protocol for High Latency and Resource-Constrained Networks

Chen Tian[1] Hongbo Jiang[1] Xue Liu[2] Xinbing Wang[3] Wenyu Liu[1] Yi Wang[1]
[1]Department of EIE, Huazhong University of Science and Technology, Wuhan, Hubei, China
[2]School of Computer Science, McGill University, Montreal, Quebec, Canada
[3]Department of Electronic and Engineering, ShangHai JiaoTong University, No.800 DongChuan Road Shanghai
[1]{tianchen,hongbojiang,liuwy,ywang}@mail.hust.edu.cn, [2]xueliu@cs.mcgill.ca, [3]xwang8@sjtu.edu.cn

***Abstract*—Existing terrestrial synchronization protocols including RBS, FTSP, TPSN, LTS and TSHL have already achieved high precision in radio networks, but none of them perform well in high latency networks like acoustic sensor networks. In this paper, we present Tri-Message: a lightweight time synchronization protocol for high latency and resource-constrained networks. As its name suggests, only three message exchanges are required in one synchronization process. Meanwhile, Tri-Message utilizes very simple mathematical operations to calculate the clock skew and offset. Specially, Tri-Message is feasible for many extremely long latency applications such as space exploration because it has an increasing synchronization precision with the increasement of distance.**

I. Introduction

Time synchronization is a critical piece for many distributed systems. Some distributed synchronization protocols, such as NTP [1], have been proposed and investigated thoroughly for many years. Emerging resource-constrained applications such as sensor networks, however, are extreme cases where nodes often collaboratively process time-sensitive data like target location [9]. In these networks, time synchronization should be mind of energy/computation consumption due to the limited resource capabilities, as well as precision [11].

In this paper, we present Tri-Message: a lightweight time synchronization protocol for high latency and resource-constrained networks. The novelty of our work is that, only three messages in one Tri-Message synchronization process are needed. In addition, Tri-Message utilizes some very simple mathematical operations to calculate the clock skew and offset as well as provides a satisfactory estimation. Another advantage of Tri-Message is that synchronization precision of Tri-Message is increasing with the increased distance, which makes it feasible for high latency applications such as space exploration.

The remainder of this paper is organized as follows. We introduce the background and related works in Section II. Section III presents the basic idea behind Tri-Message, followed by the mathematical framework and analysis. Section IV provides a performance comparison of Tri-Message and TSHL. We conclude in Section V.

II. Backgrounds and Related Works

A. Background

Booted at different times, there must be some way for distributed network nodes to determine a common time base, which we call the process of time synchronization. What are the obstacles of time synchronization? Two factors should be dealt with: clock *offset* and clock *skew* (clock drift speed). Clock *skew* is caused by variations in crystal oscillation frequency. Without *skew*, *offset* can be determined by a single pair of messages exchange if we can compensate for any sources of uncertain latencies in the path. When *skew* exists, already synchronized nodes would eventually drift out of synchronization sooner or later, hence a synchronization protocol should deal with both *offset* and *skew*.

Due to different assumptions in which sources of variation are dominant [6], [8], existing protocols have adopted different approaches to eliminate one or several sources of errors simultaneously. Seldom have they taken propagation delay into consideration. In high latency networks, clock drift continues just *during* the synchronization process. An accurate time synchronization scheme must account for this source of error.

We start with a few definitions. We refer to t as global clock times and node name (e.g. node A) to denote local clock readings in this paper. The subscript indicates the local clock reading index, for example, A_1 is the first local clock reading recorded at node A.

B. Previous Protocols

There are just two fundamental schemes to synchronize clocks: Sender-Receiver and Receiver-Receiver. As a Sender-Receiver two-way scheme shown in Figure 1(a), NTP works well over Internet paths with high latency and high variability and estimates both *offset* and *skew* [1]. Because of its long-term bi-directional time information exchange, NTP is unsuitable for many other high latency applications like acoustic sensor networks.

Existing terrestrial sensor network synchronization protocols achieve high precision in radio networks. Reference Broadcast Synchronization (RBS) [2] eliminate transmitter

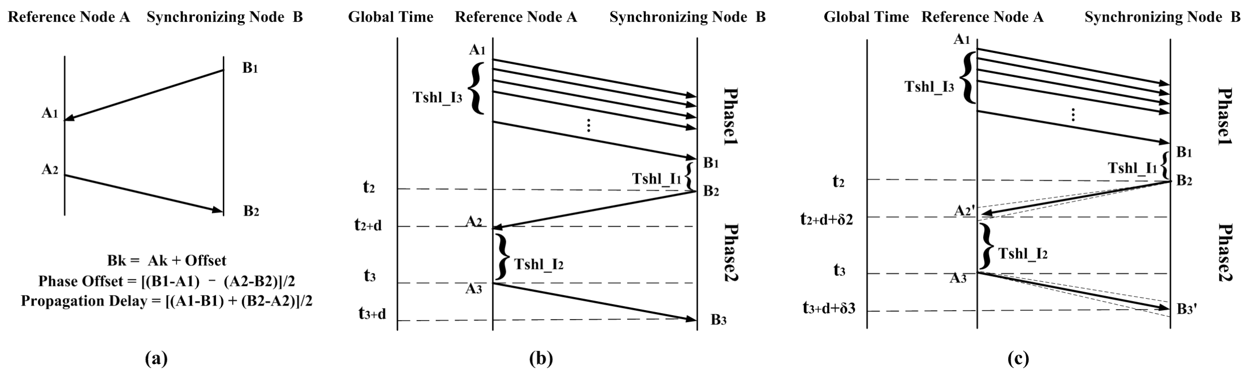

Fig. 1. (a) Sender-Receiver two-way synchronization (b) TSHL: Synchronization (c) TSHL: Synchronization with jitter δ_2 and δ_3

side uncertainties by Receiver-Receiver style synchronization. Flooding Time Synchronization Protocol (FTSP) [4] eliminates timestamp uncertainty by timestamping in the MAC and PHY (radio) message layer and also account for byte alignment jitter. Both RBS and FTSP deal with *skew*, but they do not consider propagation delay at all because RF signal travels at the speed of light. Due to their assumptions, none of abovementioned protocols work well in high latency networks. Taking into account propagation latency, Timing-sync Protocol for Sensor Networks (TPSN) [3] performs better in high latency networks [6]. But, TPSN does not take clock *skew* effect during the synchronization process into consideration, which is also critical to achieve a reasonable synchronization precision and stableness for high latency networks.

Time Synchronization for High Latency networks (TSHL) is the first protocol that takes into account both high propagation latency and process skew effect for high latency networks [6]. Its procedure is shown in Figure 1(b). Without loss of generality, given A is the time base: an original anchor or a node already compensated its *skew* and *offset* thereafter serve as an anchor. Node B is supposed to be synchronized which has its skew a and offset b. B_m denote the clock reading of Node B at real global time t_m. We have $B_m = at_m + b, t_m = (B_m - b)/a$, hence B could synchronize its clock once obtaining a and b. Let d refer to the standard propagation delay between nodes, and let $Tshl\_I_1$ and $Tshl\_I_2$ denote two transmission intervals between successive messages. We also let $Tshl\_I_3$ represent total time span of the Beacon phase.

TSHL splits time synchronization process into two phases. In the first phase, node A sends a group of timestamp beacons to node B, enabling node B to estimate its clock drift a through linear regression to time base (More technical details can be found in our technical report [12]). In the second phase node B enters *skew-synchronized* state, and a *skew-compensated* two-way exchange is taken.

However, time jitters during the synchronization process will affect the synchronization accuracy. Here we analyze the impact of those uncertainties shown by thin dashed lines in Figure 1(c). Time deviations from d are represent by δ_m, where m is the message index. We deliberately use a superscript on the timestamp to indicate an error-affected value. For example, B'_3 means real B_3 value is polluted by jitter. As shown in Figure 1(c), A_2 and B_3 deviate to A'_2 and B'_3 respectively. Nevertheless, the calculated a' is different from real skew a. We refer τ_a to be relative skew error and $a' = a(1-\tau_a)$. It's obvious that the smaller absolute value of τ_a, the better the synchronization precision. Let $t_k = t_3 + d + \triangle$, $\triangle$ denotes time passed since the last synchronization completed. Clock offset error $\triangle t_k$ can be calculated according the following theorem.

Theorem 1: The long-term offset error of TSHL is dominated by skew error . The instant Offset error $\triangle t_k$ of TSHL increases together with relative skew error τ_a and the propagation delay d, and can be expressed by

$$\triangle t_k \approx (\delta_2 - \delta_3)/2 + \tau_a(\triangle + d + Tshl\_I_2/2), \qquad (1)$$

Proof(details in [12]): Without jitter, we have $A_2 = t_2 + d$, $A_3 = t_3$, $B_2 = at_2 + b$ and $B_3 = a(t_3 + d) + b$. That is,

$$b = [(B_3 + B_2) - a(A_2 + A_3)]/2$$

Next we analyze the effect of jitters. Since $a' = a(1-\tau_a)$, we have

$$\begin{aligned} b' &= [(B'_3 + B_2) - a'(A'_2 + A_3)]/2 \\ &= [b + at_2 + a(t_3 + d + \delta_3) + b \\ &\quad -a(1-\tau_a)(t_2 + \delta_2 + t_3 + d)]/2 \\ &\approx b + [a(\delta_3 - \delta_2) + a\tau_a(t_2 + t_3 + d)]/2 \end{aligned}$$

Next, let $\triangle$ denote time interval since synchronization, since

$$\begin{aligned} t'_k &= (B_k - b')/a' = (at_k + b - b')/a' \\ &\approx (1+\tau_a)t_k + (\delta_2 - \delta_3)/2 - \tau_a(t_2 + t_3 + d)/2 \end{aligned}$$

we have the offset error $\triangle t_k$

$$\begin{aligned} \triangle t_k &= t'_k - t_k \\ &\approx (\delta_2 - \delta_3)/2 + \tau_a[t_k - (t_2 + t_3 + d)/2] \\ &= (\delta_2 - \delta_3)/2 + \tau_a[t_3 + d + \triangle - (t_2 + t_3 + d)/2] \\ &= (\delta_2 - \delta_3)/2 + \tau_a(\triangle + d + Tshl\_I_2/2) \end{aligned}$$

We argue that TSHL is not applicable for resource-constrained networks in three-fold. First, while it achieves/

considerable high precision, the energy and computation consumption of TSHL are significantly high: a typical synchronization between two-nodes costs 27 packets; computation-heavy linear regression algorithm is required for an accurate estimation of node *skew*. Furthermore, τ_a is affected by first phase beacon numbers and jitters. Finally, we found τ_a is sensitive to $Tshl\_I_3$, neglected by [6] but greatly affect synchronization precision. This is shown by our simulations.

III. Tri-Message

A. Assumptions and Overview

Our assumptions are exactly the same with [6]. Consistent with previous works, we only assume propagation delay is almost constant over the message exchange. This assumption is reasonable for underwater acoustic network and space communication. We also assume most errors had already been compensated by MAC/PHY layer timestamping, and all remain uncertainties can be treated as a receive time jitter, which follows Gaussian distribution, add to propagation delay d. The third assumption is that clocks are short-term-skew-stable. That is, clock *skew* maintains constant during the synchronization process. Long term instability can be countered by resynchronization.

We focus on two nodes' situation, one node and one anchor, to illustrate its operation. While these equations and the protocol are specific to synchronization between two hosts, they could be easily generalized to multi-hop time synchronization other previous protocols.

As its name suggests, only three message exchanges are needed for a single Tri-Message synchronization process. Assume that anchor A has no *skew* and *offset* error. B has local clock *skew* β and *offset* α. Apart from previous expression, we reveal the relations between B's clock reading B_m and global clock t_m as

$$\begin{aligned} B_m &= \beta t_m + \alpha \\ t_m &= (B_m - \alpha)/\beta \end{aligned} \tag{2}$$

The general process of Tri-Message is shown in Figure 2(a). First, the anchor node A sends a message to node B, at the same time captures the transmit timestamp A_1 in MAC/PHY layer and put the timestamp in the message; node B captures its own receive timestamp B_1 during the reception of the message and, save the send timestamp A_1 contained in the first message. Next, two nodes swap their roles and B saves the transmit timestamp B_2 and A records the receive timestamp A_2. Then they swap again. A sends the third message and put the transmit timestamp A_3 together with A_2 in the packet. At last, B receives the third message so that all 6 timestamps are known to B.

After three-message exchanges, B has 6 timestamps $A_1, A_2, A_3, B_1, B_2, B_3$. From the global clock view, we have 6 reference equations [12], and we have

$$\begin{aligned} \beta &= (B_3 - B_1)/(A_3 - A_1) \\ \alpha &= (B_1 + B_2)/2 - (A_1 + A_2)\beta/2 \end{aligned} \tag{3}$$

Next, node B is able to estimate its *skew-offset-compensated* global time by equation (2).

Our algorithm draws the concept of skew modeling from RBS [2], skew compensation during the synchronization exchange from TSHL [6].As opposed to prior works, we synchronize a node to a time base anchor by only three messages, which is extremely energy efficient. The computational complexity is also tractable: no linear regression is needed as as opposed to [6]. To our best knowledge, this is the most efficient and practical synchronization protocols for high latency networks.

B. Discuss and analysis

In this subsection, we investigate the impact of jitters to the synchronization performance of Tri-Message. By taking into account time deviations δ_m shown in Figure 2(b), the estimated skew β' can be expressed by

$$\begin{cases} \beta' = \frac{B_3' - B_1'}{A_3 - A_1} = \beta(1 - \frac{\delta_1 - \delta_3}{t_3 - t_1}) = \beta(1 - \tau_\beta) \\ \tau_\beta = (\delta_1 - \delta_3)/(t_3 - t_1) \end{cases}$$

where τ_β is the relative skew error. We now have

Theorem 2: Tri-Message cause offset error given by

$$\begin{aligned} \triangle t_k = {} & (\delta_2 - \delta_1)/2 + (\delta_1 - \delta_3) \\ & \left(\frac{\triangle}{2d + Tri\_I_1 + Tri\_I_2} + \frac{4d + Tri\_I_1 + 2Tri\_I_2}{4d + 2Tri\_I_1 + 2Tri\_I_2}\right) \end{aligned} \tag{4}$$

The proof of the theorem can be found in [12] due to space limit. Interestingly, Tri-Message causes decreasing offset error with the increase of propagation delay due to the first item in equation 4. This characteristic makes Tri-Message feasible for extremely high latency applications like space exploration networks, as mentioned in Section I.

IV. Performance Evaluation

In this section we present simulation results of Tri-Message and the comparison with TSHL [6], which is the closest one to our work, considering precision in high latency networks. Due to space limitation, energy and computation evaluation are presented in related technical report [12].

A. Simulation setup

The Tri-Message and TSHL protocols are both implemented by a custom event driven, packet level simulator designed for an acoustic networks with high latency. There are two nodes in the simulation scenario: node A is an anchor with no *skew* and zero *offset*; Node B's clock has some *skew* and *offset* relative to the global time. We modeled all uncertainties in one message delivery process to a single δ_k by introducing a Gaussian receive jitter, similar to that in FTSP [4] and TSHL [6]. In our simulations, granularity is fixed because the error caused by granularity can be combined with the error caused by interrupt handling. The granularity of the clocks is set to 1μs, which is common in sensor networks [6]. We allow the following adjustable parameters in our simulations:

- Initial node clock skew and offset

This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE ICC 2009 proceedings

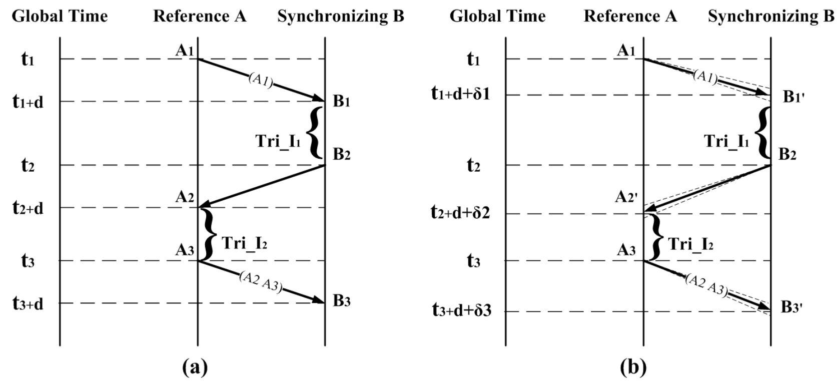

Fig. 2. (a) Tri-Message: Synchronization (b) Tri-Message: Synchronization with jitter δ_1,δ_2 and δ_3

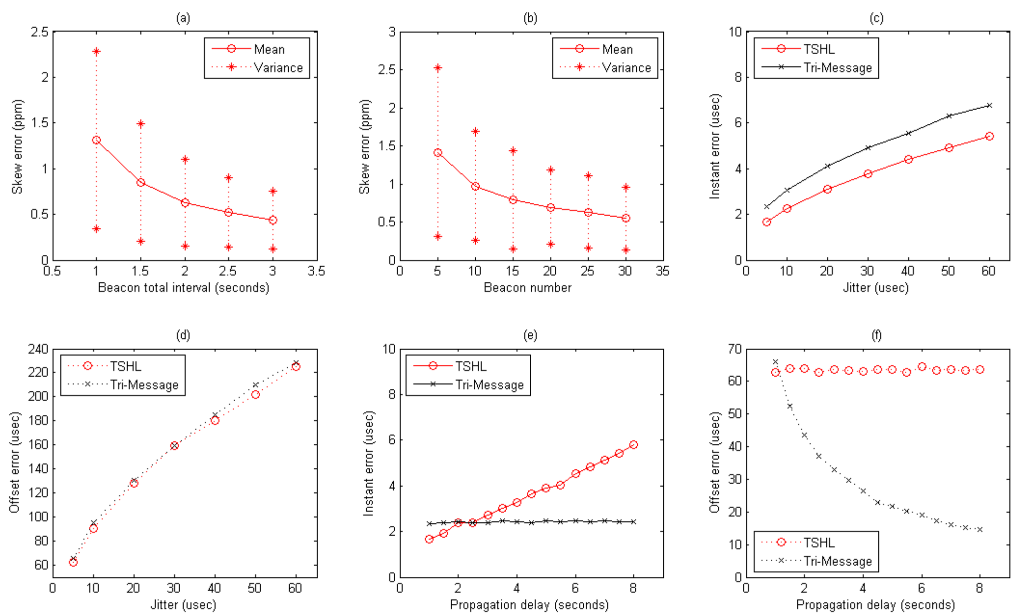

Fig. 3. (a) TSHL: Effect of beacon interval (b) TSHL: Effect of beacon number (c) instant error on varying jitters (d) offset error on varying jitters (e) instant error on varying propagation delays (f) offset error on varying propagation delays

- Jitter distribution
- Propagation delay
- Message intervals

Each data point shown in a graph is the mean absolute value of 100 simulation runs. Error bars show standard deviations. Unless specifically mentioned, the following parameters are used in all experiments: Skew = 40 ppm (parts per million), Offset = 10μs, Propagation Delay = 1s, Receive Jitter = 5μs. Consistent with [3], [4], [6], three evaluation metrics are used in our work: *Skew Error*, *Instant Error* and *Offset Error*.

We distinguish the presentation of message intervals by labeling them with protocol names for clarity. As shown in Figure 2(a), Tri-Message has two intervals: Tri_I_1 and Tri_I_2. With an additional Beacon interval, intervals of TSHL are denoted as $Tshl\_I_1$, $Tshl\_I_2$ and $Tshl\_I_3$ in Figure 1(b) and (c). Note that $Tshl\_I_3$ is the total span of beacons, and should be divided by beacon numbers to get per-message interval between successive beacons. The total synchronization time of Tri-Message can be presented as $3d+Tri\_I_1+Tri\_I_2$, and TSHL is $3d + Tshl\_I_1 + Tshl\_I_2 + Tshl\_I_3$.

B. TSHL Parameters investigation

Results in [6] have shown that TSHL accuracy is directly proportional to beacon message number, receive jitters or the granularity the clocks used. However, we found that TSHL is also related to beacon interval. Before comparative evaluation, parameters are investigated to figure out a proper parameter settings for TSHL.

First, we vary beacon interval $Tshl\_I_3$ to investigate its effect in terms of skew error. Beacon numbers are fixed to 25. As shown in Figure 3(a), skew error decreases when beacon interval increases. Through tracing, we found the reason: τ_a decreases hence skew error decreases, when $Tshl\_I_3$ increases. Although not mentioned in [6], we believe this is reasonable because the longer delay between successive data points, the better linear regression solution converge to real solution. For the rest simulations in this paper, we fix $Tshl\_I_3$ to 2 seconds to achieve a relative precise skew.

Next, we vary the number of beacons in the first phase. Shown in Figure 3(b), TSHL skew is sensitive to beacon numbers, which is also verified by [6]. For the rest simulations, we use a constant 25 of beacon number for TSHL.

C. Comparison of errors

We now compare Tri-Message with TSHL in terms of three kinds of errors we mentioned before. Because message intervals could affect both algorithms, we set their total process time to be equal. Since $Tshl\_I_1$ and $Tshl\_I_2$ has little effect compared with $Tshl\_I_3$, we let them be close to zero and set $Tshl\_I_3$ to maximum hence optimize the performance of TSHL

$$Tri\_I_1 + Tri\_I_2 = Tshl\_I_3 = 2sec \tag{5}$$

First, we investigate the receive jitter effect on the accuracy of both algorithms. We fix propagation delay to be 1 seconds. The jitter is addictively incremented by 10μs. The result is shown in Figure 3(c) and (d). For clear presentation, only mean value are presented in Figure 3(c)-(f). Similar to our theoretical analysis, Both Tri-Message and TSHL offset error is directly proportional to the receive jitter in Figure 3(d). We also show the effect of receive jitter on instant error in Figure 3(c). We conclude that Tri-Message is as sensitive as TSHL with respect to jitters. Tri-Message looks inferior to TSHL in this simulation because propagation delay is 1second only. If delay is longer, Tri-Message can outperform TSHL, as we will demonstrate in next simulation.

Next, we measure instant error and offset error as a function of propagation delay. Here jitter is set to be 5μs constant. We expect that the increase in propagations will reduce the skew error of Tri-Message, as discussed in Section III. The delay is incremented by 0.5 second step. Figure 3(e) shows that, TSHL instant error is increased along with propagation delay, consistent with our theoretical analysis. Figure 3(f) demonstrates that offset error using Tri-Message decreases along with propagation increases. On the contrary, TSHL is insensitive to propagation delay as proved in [6]. This characteristic makes Tri-Message more applicable for extremely high latency networks.

Finally, we vary the node skew from 10 ppm to 100 ppm with respect to the global clock. Since both protocols model the skew, they should be adapted to any clock skew. Table I validates our expectation: the skew error of Tri-Message is considerably comparable to TSHL. The results are accordant with that in [6].

TABLE I
MEAN SKEW ERROR OVER A VARIETY OF SKEWS

Node skew(ppm)	TSHL(ppm)	Tri-Message(ppm)	diff(ppm)
10	0.65153847	0.65152399	0.00001448
40	0.64092848	0.65215201	0.01122353
70	0.65306124	0.65584639	0.00278515
100	0.65688778	0.65544851	0.00143927

V. CONCLUSIONS

None of the existing terrestrial synchronization protocols are applicable for high latency networks. In this paper, we have proposed Tri-Message, a lightweight time synchronization protocol for high latency and resource-constrained networks which achieves high precision time synchronization, as well as only impose very small energy and computation cost. Another encouraging advantage is that, Tri-Message has an increasing synchronization precision with the increase of distance, which makes it feasible for high latency applications such as space exploration.

ACKNOWLEDGMENT

This work was supported in part through Chinese National 863 project (No.2007AA01Z223) and National Natural Science Foundation of China (No.60572063, No.60803115, No. 60873127).

REFERENCES

[1] D. Mills, Internet time synchronization: the network time protocol; RFC 1129, Internet Request for Comments, no. 1129, Oct. 1989.
[2] J. Elson, L. Girod, and D. Estrin, Fine-grained network time synchronization using reference broadcasts, in Proceedings of the Fifth Symposium on Operating Systems Design and Implementation (OSDI 002), Boston, MA, USA, December 2002, pp. 147–163.
[3] S. Ganeriwal, R. Kumar, and M. B. Srivastava, Timing-sync protocol for sensor networks, in Proceedings of the First International ACM Conference on Embedded Networked Sensor Systems (SenSys). Los Angeles, California, USA: ACM Press, 2003, pp. 138–149.
[4] M. Maroti, B. Kusy, G. Simon, and A. Ledeczi, The flooding time synchronization protocol, in Proceedings of the Second International ACM Conference on Embedded Networked Sensor Systems (SenSys).Baltimore, MD, USA: ACM Press, 2004, pp. 39–49.
[5] J. V. Greunen and J. Rabaey, Lightweight time synchronization for sensor networks, in Proceedings of the Second ACM international conference on Wireless sensor networks and applications. San Diego, CA, USA: ACM Press, 2003, pp. 11–19.
[6] A. Syed and J. Heidemann, Time synchronization for high latency acoustic networks, in Proceedings of IEEE INFOCOM 2006
[7] I. F. Akyildiz, D. Pompili, and T. Melodia, State-of-the-Art in Protocol Research for Underwater Acoustic Sensor Networks, in Proc. of ACM International Workshop on UnderWater Networks (2006)
[8] M. Horauer. K. Schossmaier, U. Schmid, and T. Vienna, PSynUTCłevaluation of a high precision time synchronization prototype system for ethernet lans, in Proceedings of 34th Annual Precise Time and Time Interval Meeting (PTTI), Reston, Virginia, USA, December 2002.
[9] J. Liu, Y. Zhang and F. Zhao, Robust Distributed Node Localization with Error Management, MobiHoc 2006, 2006, Florence, Italy.
[10] H. William, B. P. Flannery, S. A. Teukolsky, and W. T. Vetterling. Numerical Recipes in C: The Art of Scientific Computing. New York, NY: Cambridge University Press, 1992
[11] J. Elson and D. Estrin, Time Synchronization for Wireless Sensor Networks, IPDPS Workshop PDC, 2001
[12] C. Tian, W. Liu, H. Jiang, and Y. Wang. Tri-Message: A Lightweight Time Synchronization Protocol for High Latency and Resource-Constrained Networks. Tech. Rep. Available at: http://alexandretian.googlepages.com/HUSTMC-R-2007-1003.doc