# Time Synchronization of Underwater Wireless Sensor Networks

Li Liu
*Shandong University*
*P.R. China*

## 1. Introduction

Large propagation delay and node movement are considered to be two significant attributes that differentiate an underwater wireless sensor network (UWSN) from a ground wireless sensor network (WSN). The acoustic-based media dramatically narrows the bandwidth of communication of UWSN due to slow propagation speed. An underwater sensor node can move out of and into another node's range frequently in an unstable underwater environment. In this chapter, the author will elaborate the study of investigating the property and the impact of these two attributes of UWSN. Then, this chapter will describe a prototype of a synchronization protocol which is suitable for UWSN considering the effects of both propagation delay and movement. With its protocol algorithm, no time synchronization is necessary if the time stamps of the received data packets are within the tolerance. In this fashion, the network underwater does not need to perform global time synchronizations frequently nor periodically, which reduces the time used to synchronize clocks among sensor nodes. Finally, this chapter will discuss simulation results which show the time cost for synchronization is linear to the data packets exchanged with this protocol.

## 2. Three Characteristics of UWSN Time Synchronization

### 2.1 Uncertain Interrelationship

The interrelationship among synchronizing parties are erratic since the underwater sensor nodes are not as stable as those on the ground due to undercurrents. In other words, underwater sensor nodes oscillate along with the jumbled waves all the time. The undetermined vertical movement is tremendously larger than the horizontal movement, and therefore this changes the topology after the network was deployed. This topology change affects the time synchronization because sensor nodes in the networks usually are synchronized with reference clock model, e.g., the Reference Broadcast Synchronization (RBS) (Elson et al., 2002). Therefore, each sensor node should know neighbors which are in its acoustic communication range and those which are not waiting for acknowledgement too long time and consuming too much power as Fig 1 shows. Node B may be thrown out from node A's acoustic range to position C in space. Once a neighbor sensor node is out of communication range, sensor A would stop trying to neither synchronize with it nor pass data packets to it, e.g. B. When the Node B travels to another node's territory, such as position D in Fig 1, it could join another node E's data-collecting cluster.

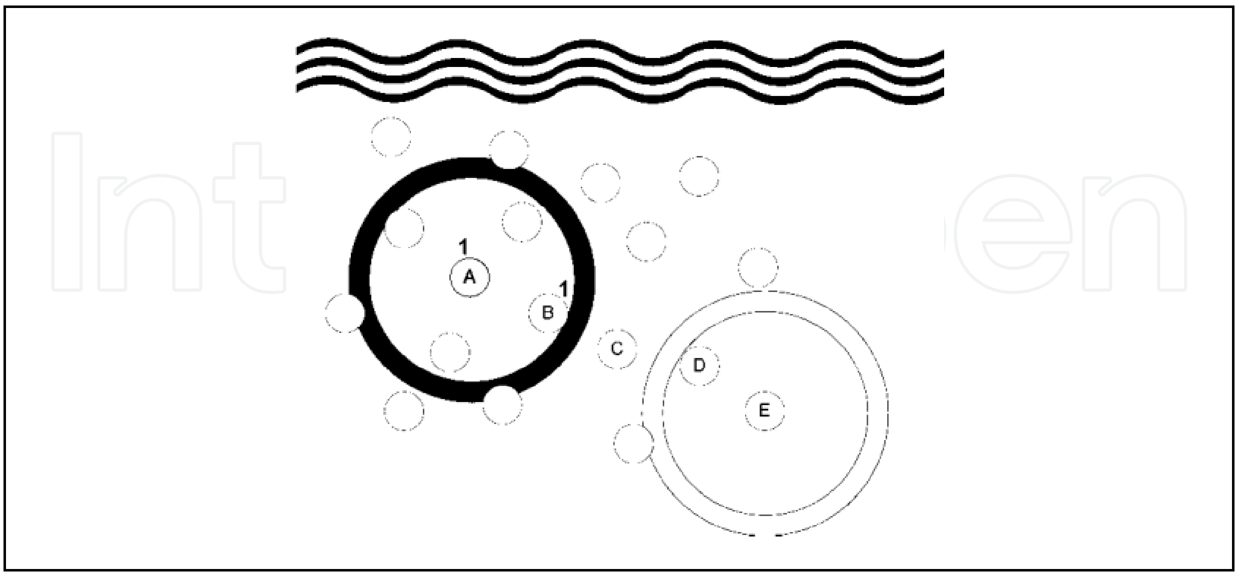


Fig. 1. Neighbour Node Uncertainty in UWSN

### 2.2 Synchronizing with New Node

Reliability, such as data accuracy of a new join-in node is another concern. Data collected for profiling and future analysis highly depend on a cluster of sensor nodes in a three dimensions space. High density of nodes gives better accuracy of environmental data (Xie et al., 2010). Vertical and horizontal waving undercurrents would bring a new sensor node into another sensor node's territory. However data provided by the new joining sensor to a cluster in the UWSN can be accepted only if the new node' clock is synchronized with the cluster. Therefore, there are more join-quit processes of nodes in an UWSN than those in a WSN on the ground because of the unstable issue.

Most sensor nodes in an UWSN are deployed by binding to ropes which are docked onto the bed of water or floater. The relative positions of sensor nodes are easily changed by the tension of the rope. On the other hand, the shape and weight of an underwater sensor affects the rope length caused by tension's change. Many other factors, such as temperature and mineralization of water, cause the degree of rope tension perplexed. The tension change of rope brings in uncertainty of nodes' positions mentioned above.

### 2.3 Propagation Delay

Third, due to the large propagation delay and low data bandwidth in an UWSN, beacon frame exchanged between two underwater sensor nodes should be simple and reduced in the total amount. Like most of radio frequency used on the ground, underwater acoustic signal also uses one channel for receiving and sending data (Kong, et al., 2005; Pompili, et al., 2006). Furthermore, propagation delay varies with many factors, e.g., the density and the purity of water, animal noise, etc.

A well-designed time synchronization scheme for underwater sensor network should be aiming to improve the synchronization process with careful consideration of these three issues.

In the rest of this chapter, the author will first talk about related background knowledge including the strategy of testing propagation delay for an UWSN, the network model, the clock model because there is little literature talking about the time synchronization issue in an underwater wireless sensor networks ever before. Then, this chapter uses an example time synchronization protocol of UWSN to illustrate how to design synchronizing algorithms which are suitable for underwater distributed systems. Overview of implementing the algorithm will also be presented. Finally, this chapter will show initial simulation results of the prototype protocol.

## 3. Background Knowledge

### 3.1 Test Strategy of Propagation Delay

Due to the uncertain factors listed above, it is difficult to find a reasonable constant value to compute the propagation delay since it varies when each of environment factors changes. Mathematics and mechanics analysis sometime are not able to depict right incidence to the UWSN. The best way to estimate the potential effect from environment is applying prototype trial measuring before deploying the whole UWSN underwater.

For the propagation delay and node mobility, we could have the following trial deployment to obtain the environmental parameters by measurements instead of estimations, e.g., the speed of acoustic and the swing amplitude of a node bound to rope, etc. Because the bottom end of a rope which ties up the sensor is anchored in a deeper position, the segment between fixed end and sensor end swung by undercurrent. The formula to calculate acoustic propagation speed is (1) in fresh water and (2) in sea water according to Kinsler's book (Kinsler, et al., 1982), respectively.

$$\begin{aligned} c(P,t) = 1402.7 + 488t^2 - 482t^2 + 135t^3 \\ + (15.9 + 2.8t + 2.4t^2) \end{aligned} \tag{1}$$

where c = Speed of sound in meters/sec, P = Pressure in bars (1 bar = 100 kPa), and t = 0.01T where T is the temperature in Celsius.

$$\begin{aligned} c(D,S,t) = 1449.05 + 45.7t - 5.21t^2 + 0.23t^3 \\ + (1.333 - 0.126t + 0.009t^2)(s-35) + \Delta(D) \end{aligned} \tag{2}$$

where $\Delta(D) \cong 16.3D + 0.18D^2$, at latitude 45 degrees in the oceans, where c = Speed of sound in meters/sec, t = T/10 Where T is the temperature in degrees Celsius, S = Salinity in parts per thousand, and D = Depth in kilometers.

For other latitudes in degrees, replace $D$ with $D(1-0.0026\cos 2\varphi)$. This gives c with a standard deviation of 0.06 m/s down to a depth $D = 4\ km$ in oceanic waters.

A more complicated correction gives a standard deviation of 0.02 m/s:

$$\begin{aligned} (D,S) = (16.23 + 0.253t)D + (0.213 - 0.1t)D^2 \\ + [0.016 + 0.0002(S-35)](S-35)tD \end{aligned} \tag{3}$$

Operators need to use precise device to get C (the speed of sound) and S (salinity in parts per thousand) in order to get an accurate value of acoustic propagation speed c in testing environment before deploying the whole WSN underwater since it is impossible for sensors to get accurate speed themselves.

To obtain the real value of swing amplitude which determines the mobility of node, operators could use camera to record the trail deployed sensors for a certain period of time and analyze the maximum, minimum and average swing amplitudes angle $\theta_{\max}$ , $\theta_{\min}$ , $\theta_{avg}$ with the help of image processing technique. Then, the maximum, the minimum and the average swing amplitudes can be calculated by formula set below.

$$d_{horizontal} = \sin\theta \cdot l \text{ and } d_{vertical} = (1 - \sin\theta) \cdot l \tag{4}$$

where $l$ is the length between node and fixed point at the bottom.

After carrying out the trial deployment, we could get the real environmental specification. Therefore, an underwater sensor needn't try to censoring these parameters because the sensor cannot get accurate value of some parameters in statistics, e.g., the swing amplitude. An operator could assign the creditable data into formula for later computation in the sensors, instead.

### 3.2 Network Model

An UWSN is a dense network consisting of a large number of resource-constrained sensor nodes with neither reference nodes nor a root node. This is a realistic deployment scenario in that a WSN is inherently infrastructure-less (Hu, et al., 2008) where many sensors autonomously organize themselves into a connected structure. Thus, it is often desirable to minimize the dependency of time-synchronization on infrastructure nodes. Each node maintains a sufficient number of neighbors to accelerate the synchronization process. The number of neighbors (undercurrent moving) can be easily adjusted by changing the transmission power when the synchronization information is broadcasted. A bidirectional neighbor relationship is not needed in this scheme. For further reduction of the synchronization overhead each node piggybacks the synchronization information on beacon messages that are periodically broadcast to refresh each node's neighbor list. In the current work, we assume that there are some reliable broadcast techniques such as (Tang, et al., 2001) are used.

### 3.3 Clock Model

Each sensor node has its own physical clock, calculated by counting pulses of its hardware oscillator running at a particular frequency. In practice, sensors' oscillators run at slightly different frequencies and the frequency varies unpredictably, depending on ambient factors such as temperature and humidity. Hence, sensors' clocks are subject to a divergence or clock offset. Based on Sichitiu's paper (Sichitiu, et al., 2003), for a relatively extended period of time (minutes to hours), the clock can be approximated with good accuracy by an oscillator of fixed frequency. The local clock of a sensor node i can thus be approximated (Lamport, et al., 1985) as

$$T_i(t) = \alpha_i t + \beta_i \tag{5}$$

where t is the physical time like UTC, $\alpha_i$ is the drift rate (frequency)of *i*, and $\beta_i$ is the offset between the local clock and the physical time.
Using equation (5), we can compare the local clocks of two nodes in a network, say node 1 and node 2 as:

$$C_1(t) = \alpha_{12} C_2(t) + \beta_{12} \tag{6}$$

We call $\alpha_{12}$ the relative drift, and $\beta_{12}$ the relative offset between the clocks of node 1 and node 2. If two clocks are perfectly synchronized, then their relative drift is 1--meaning that the clocks have the same rate- and their relative offset is zero--meaning that they have the same value at that instant.

## 4. An Example Protocol Algorithm

The example synchronization protocol is based on the Interactive Convergence Time Synchronization (ICTS) algorithm similar to the one in (Lamport, et al., 1985). In ICTS, the network-wide synchronization is achieved by having each node first derive the time offsets between itself and all of its neighbors by exchanging messages. Each node then computes the average of the measured clock offset and uses it to adjust its own clock. As long as less than one third (half) of neighbor nodes are mis-behaving with Byzantine (non- Byzantine) faults, all the sensor nodes in the neighborhood will establish a common equilibrium time.

### 4.1 Time offset

The protocol applies the single message broadcast method which is used in FTSP (Mar´oti, et al., 2004) to compute the offset between two nodes. FTSP successfully eliminates major sources of uncertainties in the packet recommission (i.e., transmission time, access time, reception time, jitter of interrupt-handling and encoding/decoding time) by performing MAC-layer timestamping multiple times for every message at each byte boundary and embeds a final error-corrected and averaged timestamp into the message. The only uncertainty is the propagation time (for packets to traverse the wireless link) which is often very small and can be safely ignored. According to Mar´oti's findings (Mar´oti, et al., 2004), using only 6 timestamps per message, FTSP achieves the time stamping accuracy of 1.4 μs on the Mica2 platform. Thus, one radio broadcast is sufficient for all the neighbors to accurately calculate the time offsets between their clocks and sender's clocks, each of which is simply the difference between transmission and reception timestamps.

### 4.2 ICTS with Propagation Delay

Let s be the sensor node performing time-synchronization and $n_s$ is the number of S's neighbors. $T_i$ and $T_s$ represent the send and receive timestamps. $p_{i,s}$ is the propagation delay when message leave node i until reach node s. Node S can then calculate the time offset between itself and node i as $\Delta_{s,i} = T_s - T_i - p_{i,s}$. After obtaining the equation for i = 1…ns from all of its neighbors, s computes its new clock value at time t or $T_s^{'}(t)$ as:

$$
\begin{aligned}
T_s^{'}(t) &= T_s(t) + \frac{1}{n_s+1}\sum_{i=1}^{n_s}\Delta_{s,i} \\
&= T_s(t) + \frac{1}{n_s+1}\sum_{i=1}^{n_s}(T_s - T_i - p_{i,s}) \\
&= T_s(t) + \frac{1}{n_s+1}\sum_{i=1}^{n_s}(T_s - T_i) - \sum_{i=1}^{n_s} p_{i,s}
\end{aligned}
\tag{7}
$$

We could figure out the value of $\sum_{i=1}^{n_s} p_{i,s}$ with the strategy in Section III since the limit of $\sum_{i=1}^{n_s} p_{i,s}$ is the expected value of propagation delay.

The denominator $n_s + 1$ comes from the fact that node S's own clock is also considered for the computation of a new clock value.

Sensors terminate the initial synchronization when the local clock gets stabilized (i.e., $\left|T_s^{'}(t) - T_s(t)\right| < \varepsilon$ , $\varepsilon$ is a predefined parameter determining the synchronization accuracy). However, synchronization at a single point is insufficient, as the discrepancies in clock drift rate of different sensors will cause nodes to go out of synchronization after a short period of time. Thus, to maintain an acceptable accuracy, it is necessary to periodically execute ICTS for resynchronization, shown in Fig. 2. The appropriate resynchronization interval can be determined by the bound of the time offset and the maximum relative drift rate among sensor nodes (Sivrikaya, et al., 2004).

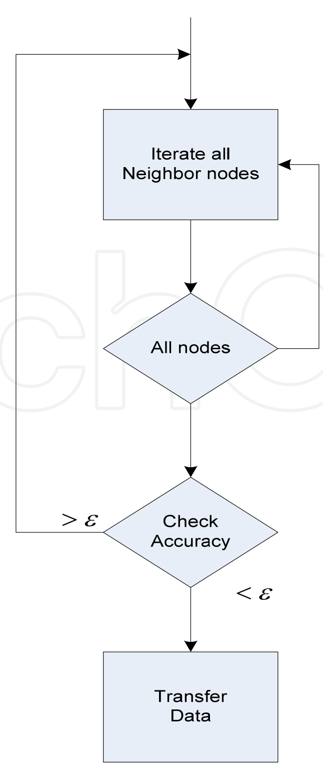


Fig. 2. Clock resynchronization

### 4.3 Data Packets

The neighbor sensors oscillate in the range and at the border of a node if the node is regard as a "sink point" in its territory. Limited by single acoustic channel and lower bandwidth the sink node cannot request a resynchronization with all the nodes once a neighbor sensor shifts over the acoustic range no matter a neighbor node is carried away from or brought in the sink node's territory by undercurrent or ocean wave. We could design a protocol which timestamps each data message. The timestamp will help the profile manager (introduced later) to determine if the data is confidential to be used or not. We could use the relative drift rate between the local clocks of nodes s and i which is defined as $\alpha_{s,i} = \alpha_i / \alpha_s$ to judge data confidence since there is no reference to the physical time, a node's drift rate (e.g., $\alpha_i$ or $\alpha_s$) cannot be directly measured.

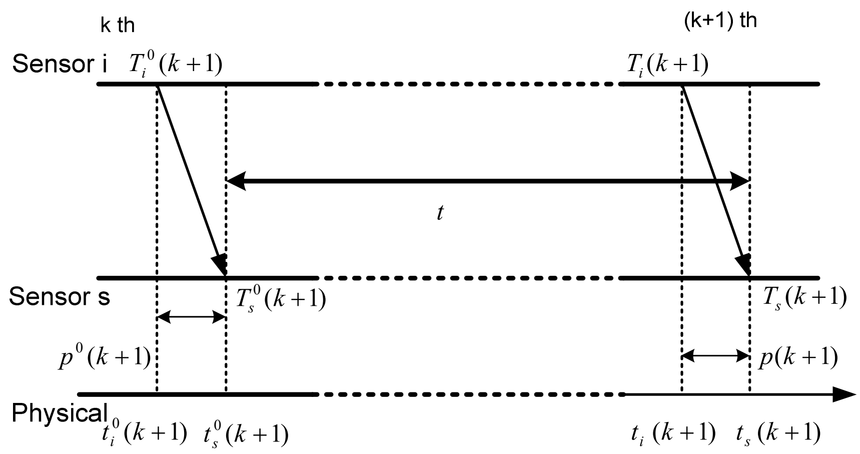


Fig. 3. ICTP with propagation

The example protocol scheme derives the relative drift rate indirectly as follows. Assume that each sensor performs synchronization periodically. In Fig. 3, $T_i\ (k)$ and $T_s\ (k)$ are the MAC-layer timestamps that record the sending time and the receive time, respectively, of the data message at the k-th iteration where k = 1, 2, … Let $t_i(k)$ and $t_s(k)$ denote the physical times corresponding to $T_i(k)$ and $T_s(k)$. Assume that sensor i and s finish their next data from $t_i\ (k+1)$ till $t_s\ (k+1)$ on physical clock. Their local clock readings of the two times are $T_i(k+1)$ and $T_s(k+1)$, respectively. The propagation delay in these two processes are $p(k)$ and $p(k+1)$ as well. t is the time between two iterations, $t = t_s\ (k+1) - t_s\ (k)$. We can express t in terms of local clocks:

$$\begin{aligned} & T_s(k+1) - T_s^0(k+1) \\ & = \alpha_s t_s(k+1) + \beta_s(k+1) - \alpha_s t_s^0(k+1) - \beta_s(k+1) \\ & = \alpha_s(t_s(k+1) - t_s^0(k+1)) \\ & = \alpha_s t \end{aligned} \quad (8)$$

and

$$
\begin{aligned}
&T_i(k+1) - T_i^0(k+1) \\
&= \alpha_i t_i(k+1) + \beta_i(k+1) - \alpha_i t_i^0(k+1) - \beta_i(k+1) \\
&= \alpha_i(t_s(k+1) - p(k+1)) + \beta_i(k+1) \\
&\quad - \alpha_i(t_s^0(k+1) - p^0(k+1)) - \beta_i(k+1) \\
&= \alpha_i(t_s(k+1) - t_s^0(k+1)) + (-p\ \ (k+1) + p^0(k+1)) \\
&= \alpha_i t
\end{aligned}
\tag{9}
$$

Therefore, the relative drift rate $\alpha_{s,i}$ can be derived by formula (10) with timestamps of packet inside the UWSN. We do not need to care about physical time outside.

$$
\alpha_{s,i}(k) = \frac{\alpha_i}{\alpha_s} = \frac{\alpha_i t}{\alpha_s t} = \frac{T_i(k+1) - T_i^0(k+1)}{T_s(k+1) - T_s^0(k+1)} \tag{10}
$$

### 4.4 Profiling Synchronization

As mentioned in the introduction, a sensor, which is brought into another sensor's territory by the undercurrent, should be examined the clock first to guarantee that data provided by this sensor has a confidential clock, that is a right relative clock drift to the existing cluster. The protocol creates a profile manager whose function is to maintain a history profile recording relative clock drift between node s and all its neighbor nodes and the nodes who have been its neighbors before. Profile manager (PM) establishes one history profile copy

$$
\{\alpha_{s,i}(k)\}_{k-q}^{k} = \{\alpha_{s,i}(k-q), \alpha_{s,i}(k-q+1), \cdots, \alpha_{s,i}(k)\}
$$

for each neighbor node i's last q relative clock drift with node s by the k-th iteration. $\{\alpha_{s,i}(k)\}_{k-q}^{k}$ exhibits a strong temporal correlation, as they represent the quality of neighbors' clocks and are updated at each iteration. Profile manager calculates a mean value μ for each profile copy with discrete or continuous probability distributions depending on the number of messages which the neighbor nodes provided. For discrete probability distributions, the protocol uses variance to compute μ, for continuous probability distributions, and we could use normal distribution to generate μ which is the location in Gaussian distribution.

With the value μ profile manager, check the timestamp of every data message provided by its neighbor. If

$$
(\alpha_{s,i}(k) - \mu)^2 < \lambda \tag{11}
$$

in discrete probability distributions and

$$
|\alpha_{s,i}(k) - \mu| < \lambda \tag{12}
$$

in continuous probability distributions, the profile manager treats the message as a confidential data message and buffers the data, if not, the data will be dropped because of untrusting. $\lambda$ is a predefined accuracy value.

The profile manager (PM) will also help decide the resynchronization interval for a particular sensor cluster. As we discussed above, the confidence of data provided by neighbor nodes settle on whether the data packet could be accepted by the existing sensor cluster, a subsystem of the whole underwater network. In overall view, higher acceptance rate stands for higher utilization of censured data. If most of sampled data packets are dropped due to accuracy requirement $\lambda$, it does not reduce the utilization of censuring data but also dries out power supply since underwater is more energy consuming. The criterion of switching the node's mode from transferring data to resynchronization is determined by the data packet acceptance rate. Profile manager creates a global table called Global Confidential Table (GCT) aiming to record the accept data packet ratio. The GCT is a one dimension fixed size table which marks "1" standing for acceptance of data packet. Default value is "0" which means the packet does not meet the $\lambda$ requirement. The protocol defines a threshold R as the number of acceptance data packets in GCT, shown in Fig 4. If ratio of acceptance data packets to table size is below R the profile manager will stop the node receiving data and start resynchronization until local clock accuracy reaches requirement formula (4) and (5). The upper GCT in Fig. 4 shows that the ratio is higher than the threshold and the lower one means that the cluster needs to be resynchronized.

$Threshold = 6$

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| $T$ | $T$ | | $T$ | | $T$ | $T$ | $T$ | | $T$ |

$size = 10$

$Threshold = 6$

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| $T$ | | | $T$ | | $T$ | | $T$ | | $T$ |

$size = 10$

Fig. 4. Global Confidential Table

The whole process flow is shown in Fig 5.
Because there is no bidirectional neighbor relationship for every two nodes, each node maintains the relative clock drift in its own acoustic range, a cluster for profiling data. On the other hand, adjacent sensors' clusters must have overlap. The overlap plays the role to keep the whole relative clock as close as possible to a unique value. Therefore, the whole network stays in a low relative clock drift level with the help of profile manager and frequent resynchronization.

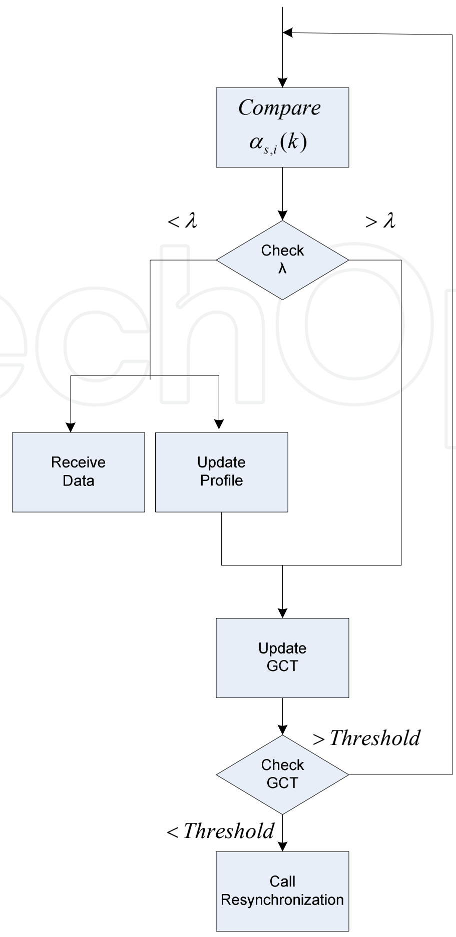


Fig. 5. Shift between sending data and time synchronization

## 5. The Effect of Undercurrent to Synchronization

The mobility of each node in an UWSN brings unfastened neighbor problem to a data profiling cluster. Sensors are deployed in different layers in an open space underwater. If we clip the space out from the whole by outmost sensors' furthest audio reachable range in one data profiling cluster the clipped space could be likened to a rubber balloon filled with water. The shape is easily changed when pressure comes outside. The pressure to the data profiling space in real world is undercurrent. Water moves along with many factors e.g., wind on the ocean surface, earth's rotation, etc., to unpredicted orientations. That is to say, if we research the synchronization of UWSN, we could not dismiss the high mobility even the sensors are anchored relative stable.

The second characteristic of the network underwater is that we cannot treat sensors underwater as 2 dimensions plane layout. Research on wireless sensor network above the ground usually assumes that the network is deployed onto the controlled environment without thinking too much about the latitude value. That is to say, the horizontal distance between two nodes above the ground plays more important role in research work on attributions of wireless sensor network above the ground. However, the network underwater exists in a real 3-dimension world. The vertical movement is as important as the horizontal movement when nodes are in a fluid environment. We need to use cube or sphere to describe the behavior of a node underwater instead of rectangle or circle in plane.

## 6. Simulations

The simulation consists by two sub phases. In the first part, we simulate the time synchronization with the traditional ICTP protocol running on our test case. Then, we simulate the example algorithm considering the effect of movement of UWSN. The profile manager (PM) took participate in this phase working abovementioned.
As the reason this chapter discussed in Section 2, the simulation use a trail deployment of sensors to measure the environmental factors. It is assumed that the real acoustic speed could be tested by professional device and calculated by. For simplicity, this simulation uses the mean value of acoustic, 1500 m/s as simulation parameter. Other parameters are shown in Table 1.

| **Parameter Name** | **Value** |
|---|---|
| Simulation Radius | 100 m |
| *Acoustic range* | 35 m |
| *Acoustic speed* | 1500 m/s |
| *Sensor clock drift* | ± 0.3 ms/sec. |
| *Initial clock offset* | ±1.0 ms |
| Threshold of accuracy | 350 µs |

Table 1. Parameters configuration

### 6.1 Synchronization of ICTP with propagation delay

The simulation deployed 30 sensor nodes in a cube whose side length is 100m. Every dimension of each node position is assigned randomly by a pseudo random number generator. Therefore, nodes are independent in spatial relationship. Fig 6 gives a node deployment scenario.

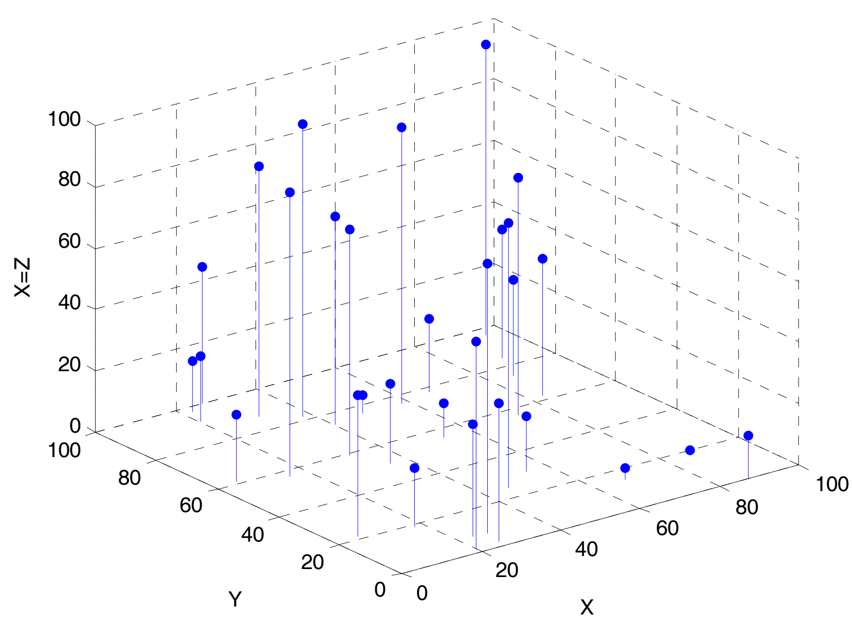


Fig. 6. Sensor nodes in 3D view

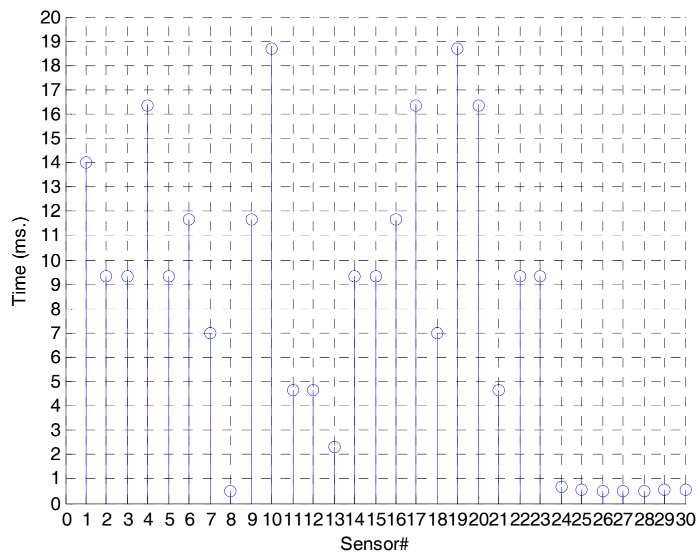


Fig.7. Time cost for each node in one UWSN

Fig 7. shows the time cost of the 30 sensors sending 100 data packets to all their neighbor nodes with ICTP synchronization method. We can find that the time cost varies due to different relative clock drift and offset of a node and its neighbor node(s).

### 6.2 Simulation Result of UWSN Synchronization Protocol

As it is described in previous paragraphs, the propagation delay of UWSN is 4 times bigger than transmission. Based on the observation strategy in Section 3, the simulation approximate the relationship between propagation delay and packet transmission to an integer multiple. First, we simulate the time cost that a node sends 100 data packets to all its neighbor sensor nodes when propagation delay is four times of transmission time.

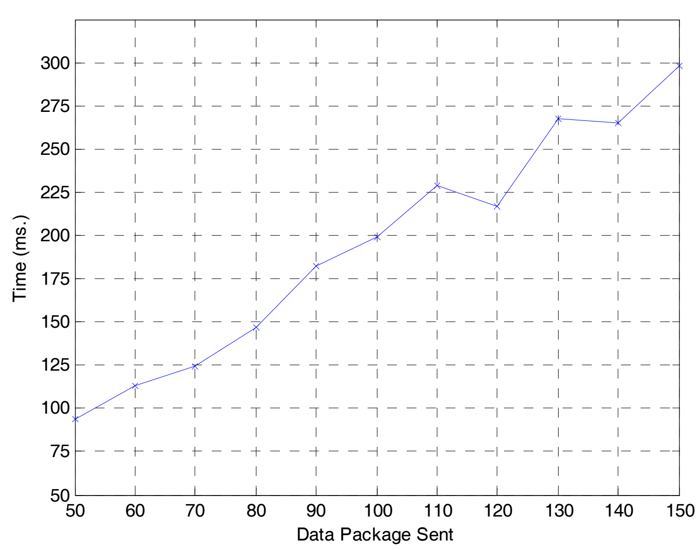


Fig. 8. 30 nodes send different number of packets when propagation delay is four times of transmission time.

Fig 8 shows the time cost curve. The total time cost goes up with total amount of data packets to be sent. Then, we add 5 more nodes to the space to structure a new network underwater.

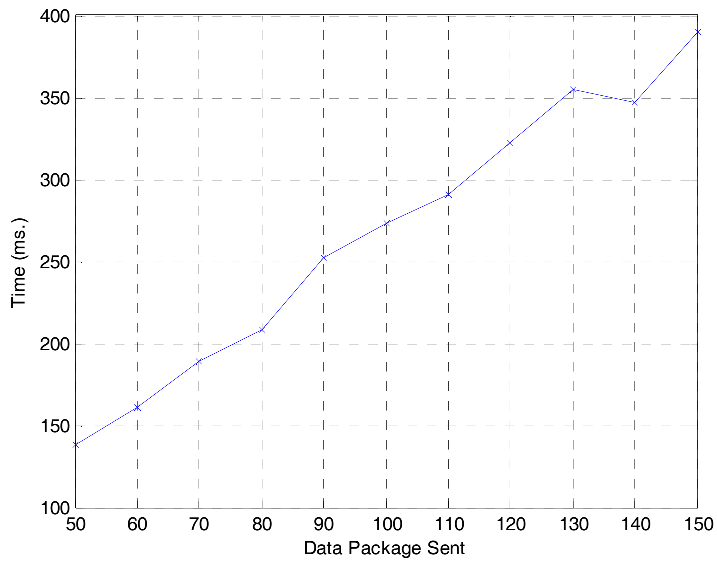


Fig. 9. 35 nodes send different number of packets when propagation delay is four times of transmission time.

Fig 9. shows the result that 35 nodes send 100 data packets to their neighbor node(s) when the propagation delay is four times of transmission time in the ICTP synchronization protocol. The time cost goes up almost the same as it goes up in the previous structure. Then the simulation deploys another five sensors into the network. There is nothing quite different but the starting point and ending point both shifted up for 50 ms in Fig 10.

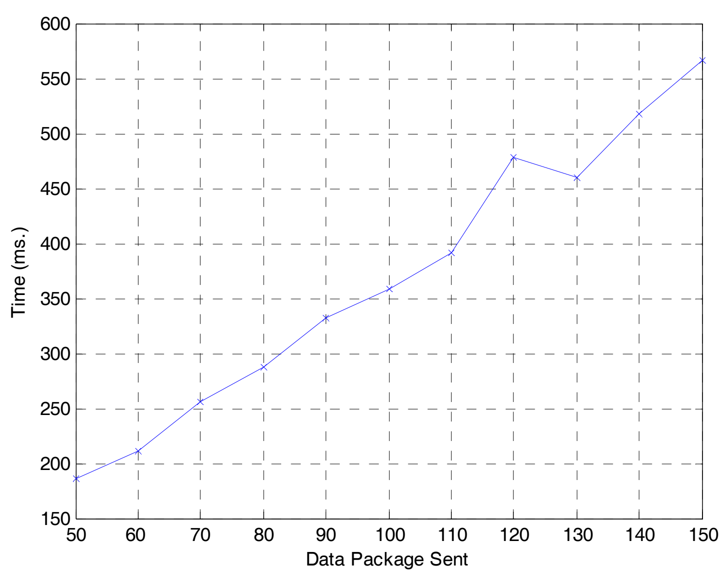


Fig. 10. 40 nodes send different number of packets when propagation delay is four times of transmission time.

To combine these three curves, result is in Fig 11.

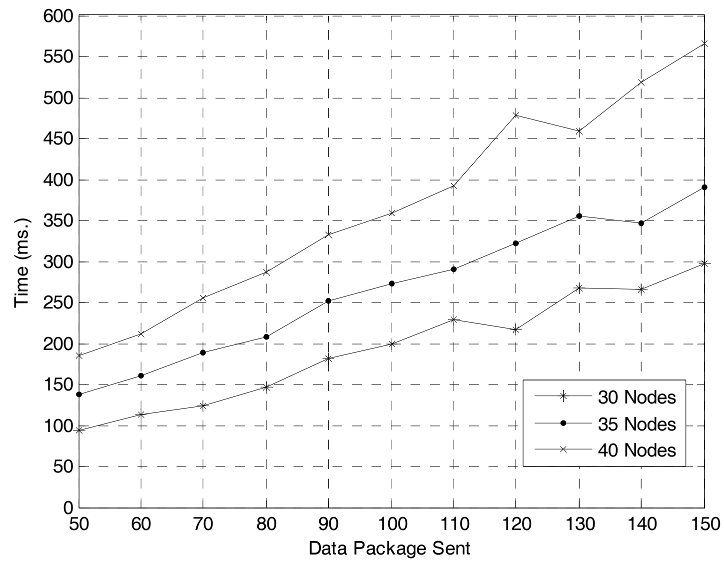


Fig. 11. 30, 35 40 nodes send different number of packets when propagation delay is four times to transmission time.

Next, simulation obtains the characteristic when propagation delay is five times to transmission time in a 30 nodes UWSN.

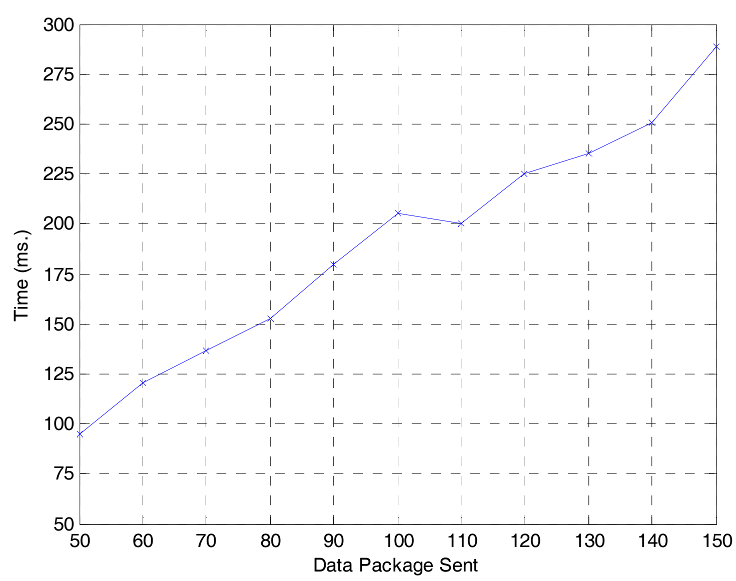


Fig. 12. 30 nodes send different number of packets when propagation delay is five times to transmission time.

In Fig 12, the total time cost increase along with the packet amount almost in the same way when the propagation delay is only four times of the transmission. Readers can compare the two curves in one chart shown in Fig 13.

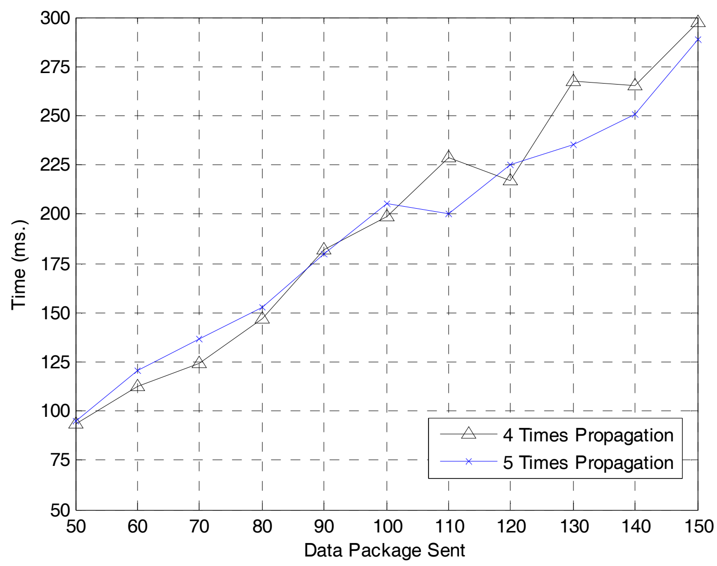


Fig. 13. 30 nodes send different number of packets when propagation delay is four or five times of transmission time.

## 7. Conclusion

In this chapter, we review those factors that are essential to the design of a new time synchronization protocol for an Underwater Wireless Sensor Netwrok (UWSN). We use a linear synchronization algorithm as an example to show these key points of proposing new protocols. The effect of large propagation delay of acoustic media in communication is addressed in simulating the demo prototype protocol. The simulation results demonstrate the difference of an UWSN time synchronization protocol by applying the new design pattern and by using the classical method. Simulation results also suggest the relationship between network performance and related factors.

## 8. References

Elson, J. E.; Girod, L. & Estrin, D. (2002). Fine-Grained Network Time Synchronization using Reference Broadcasts, *Proceedings of The Fifth Symposium on Operating Systems Design and Implementation*, pp. 147–163, ISBN 978-1-4503-0111-4, Boston, MA, USA, December 2002, New York, NY, USA

Hu, X.; Park,T. & Shin, K. G. (2008). Attack-tolerant time-synchronization in wireless sensor networks, *Proceedings of INFOCOM 2008*, pp. 41-45, ISBN 978-1-4244-2025-4, Phoenix,AZ, USA, April 2008, IEEE, Piscataway, NJ, USA

Kinsler, L.; Frey, A.; Coppens, A. & Sanders, J. (1982). *Fundamentals of Acoustics*, John Wiley & Sons, ISBN-10: 0471029335, New York, NY, USA

Kong, J.; Cui, J.; Wu, D.; & Gerla, M. (2005). Building underwater ad-hoc networks and sensor networks for large scale real-time aquatic applications, *Proceedings of Military Communication Conference 2005*, pp. 1-7, ISBN 978-0-7803-9393-6, Atlantic City, NJ, USA, October 2005, IEEE, Piscataway, NJ, USA

Lamport, L. & Melliar-Smith, P. (1985). Synchronizing clocks in the presence of faults. *Journal of the Association for Computing Machinery*, Vol. 32, No. 1, (1985) 52–78, ISSN 0004-5411

Mar´oti, M.; Kusy, B.; Simon, G. & L´edeczi, A. (2004). The flooding time synchronization protocol, *Proceedings. of SenSys 2004,* pp. 39-49, ISBN 1-58113-879-2, Baltimore, MD, USA, November 2004, ACM Press, New York, NY, USA

Pompili, D.; Melodia, T. & Akyildiz, I. F. (2006). Routing algorithms for delay-insensitive and delay-sensitive applications in underwater sensor networks, *Proceedings of The 12th Annual International Conference on Mobile Computing and Networking,* pp. 298-310, ISBN 1-59593-286-0, Los Angeles, CA, USA, September 2006, ACM Press, New York, NY, USA

Sichitiu M. L. & Veerarittiphan, C. (2003). Simple, accurate time synchronization for wireless sensor networks. *Proceeding of IEEE Wireless Communications and Networking* 2003, pp. 1266-1273 ISBN 1525-3511, New Orleans, LA, USA, March 2003, IEEE, Piscataway, NJ, USA

Sivrikaya, F. & Yener, B. (2004). Time synchronization in sensor networks: a survey, *IEEE Network Magazine's special issue on" Ad Hoc Networking: Data Communications & Topology Control,* Vol. 18, No. 4, (2004) 45-50, ISSN 0890-8044

Tang, K. & Gerla, M. (2001). Mac reliable broadcast in ad hoc networks. *Proceedings of IEEE Military Communication Conference 2001*, pp. 1008-1013, ISBN 0-7803-7225-5, Vienna, VA, USA,October 2001, Piscataway, NJ, USA

Xie, P.; Zhou, Z.; Peng, Z.; Cui, J. & Shi, Z. (2010). SDRT: a reliable data transport protocol for underwater sensor networks. *Ad Hoc Networks*, Vol. 2, No. 003, (2010) 1-15, ISSN 1570-8705

ISBN 978-953-307-261-6
Hard cover, 418 pages
**Publisher** InTech
**Published online** 14, December, 2010
**Published in print edition** December, 2010

Fax: +385 (51) 686 166
www.intechopen.com

Fax: +86-21-62489821