Journal of Computers Vol. 27, No. 3, 2016, pp. 177-199
doi:10.3966/199115592016102703016

Designing High Performance of the Self-coupled Time Synchronization Protocol on Sea Surface for the Argo Float Sensors Network

Chia-Sheng Tsai[1*] Yu-Cheng Wang[1] Hsueh-Sheng Huang[2]

[1] Department of Computer Science and Engineering, Tatung University, Taipei 104, Taiwan, R.O.C.
icstsai@gmail.com, starckwang@gmail.com

[2] Research and Development Department, Group, Laboratory, TPV Technology Limited, Taipei 235 Taiwan, R.O.C.
alexhuang0727@gmail.com

Received 28 December 2015; Revised 2 May 2016; Accepted 30 May 2016

Abstract. For the sea surface salinity (SSS) applications, the data collected by the floats needs to be uploaded as a satellite passes through. The floats distributed on sea surface are equipped with sensors to measure interested information to deliver to visible satellites passing through the sky. The architecture can be divided into three subsystems, the satellite, the Argo float sensors and the data center. In this paper, we evaluate the performance of the self-coupled time synchronization protocol in the subsystem "Argo float sensors". The basic idea is to achieve global synchronization through local pulse coupling. We will discuss the effect of various network parameters such as the degree of the node and the network diameter to the performance of the self-coupled mechanism. Furthermore, we will propose an improvement over this scheme in the paper. It reduces its message complexity by randomly turning on and off the sensor nodes for synchronization. We find that the self-coupled time synchronization protocol can achieve synchronization unless the network is sparse. It exhibits good cost-performance characteristics especially in the network environment of a great number of sensor nodes. Integrating with the proposed improvement, time synchronization can be effectively and economically reached as we will demonstrated in the paper.

Keywords: degree, hierarchical structure, pulse coupling, sea surface salinity (SSS), sensor networks, synchronization

1 Introduction

Global warming is a tough challenge threatening the very existence of our civilization. Many research projects are dedicated to study various aspects of this critical phenomenon. National Aeronautics and Space Administration (NASA) in the United States and its international partners started project Aquarius/SAC-D to study sea surface salinity and its effects on global climate and ocean/ground water circulation. In the Aquarius project, many Argo float sensors capable of collecting and distributing measured data have been built and deployed by voluntary participating countries around the world. In short, the architecture can be divided into three subsystems, the satellite, the Argo floats and the data center in the Fig.1 [1].

In the satellite subsystem, a Low Earth-Orbit (LEO) satellite needs to be launched and put into place to collect the data from the floats. The satellite uses two types of satellite-terrestrial communications. One is between the satellite and the Argo float sensors and another is between the satellite and the ground data centers. When the satellite talks to the Argo float sensors, the satellite is the sole source for the download channel. However, the upload bandwidth is usually considered contentious. In order to send the measured data, the Argo float sensors fight to get access to the upload channel. The location of the Argo float sensor can be recorded in the satellite and provided to the data center for float distribution prediction [3].

[*] Corresponding Author

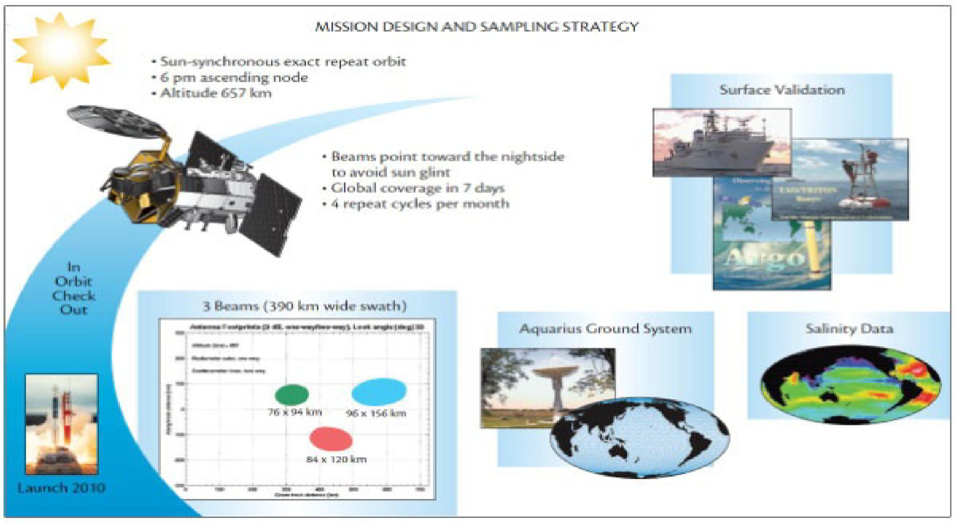
MISSION DESIGN AND SAMPLING STRATEGY
• Sun-synchronous exact repeat orbit
• 6 pm ascending node
• Altitude 657 km
• Beams point toward the nightside to avoid sun glint
• Global coverage in 7 days
• 4 repeat cycles per month
In Orbit Check Out
Launch 2010
3 Beams (390 km wide swath)
76 x 94 km
96 x 156 km
84 x 120 km
Surface Validation
Argo
Aquarius Ground System
Salinity Data

Fig. 1. The Aquarius/SAC-D mission concept [2]

The main function of Argo float sensors is to measure and store scientific data on board. Various sensors are built into the Argo float to sample or monitor the desired environmental parameters. Each Argo's float sensor will pump fluid into an external bladder and rise to the surface over about 6 hours while measuring temperature and salinity at typically 10-day intervals [4-5]. The satellite can around the earth three times to complete the data transmission. In Fig.2, the float descends and drifts for 10 days in around 1000 meters under the water. Then, it dives to around 2000 meters and, then, starts to emerge to the surface. In the beginning of 2014, 80% of floats profile to depths greater than 1500 meters. Be-sides, 12% profile to between 1000 meters and 1500 meters. As it moves upward, it measures the temperature and sea surface salinity.

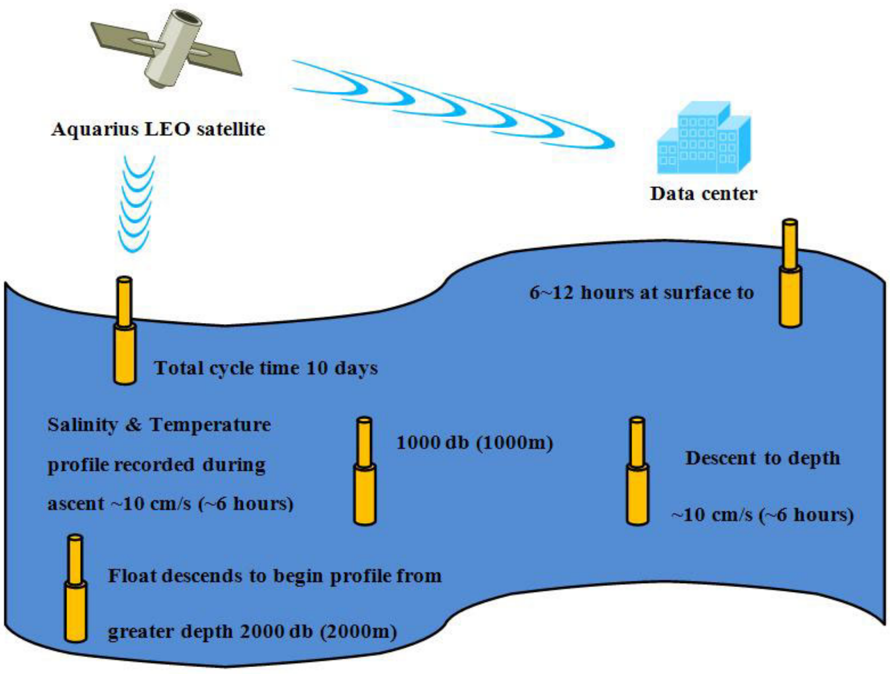

Fig. 2. The three subsystems architecture

It stays in the surface for about 6 to 12 hours and sends the collected data to the passing Aquarius satellite in the sky [6]. In Fig.3, while the Argo array is currently complete at 3918 floats, to be maintained at that level, national commitments need to provide about 800 floats per year.

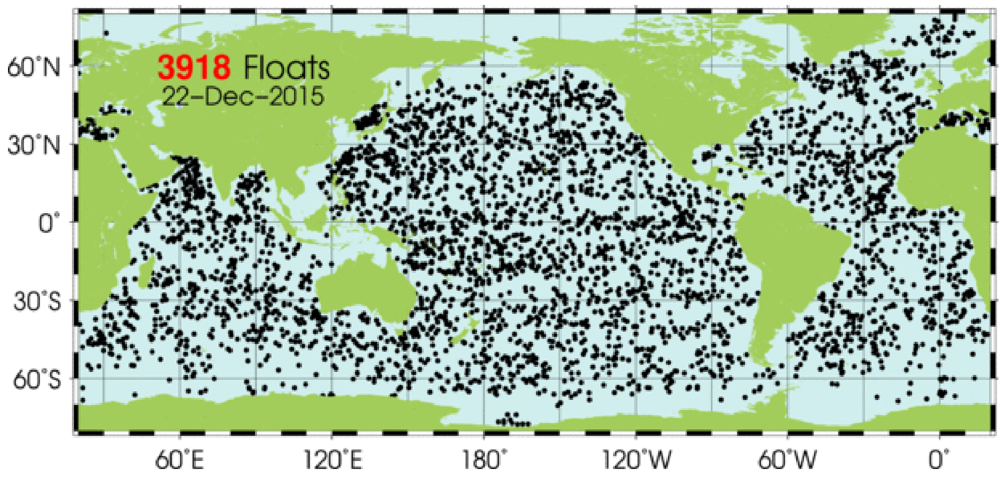

Fig. 3. Positions of the floats

For sea surface salinity application of the Argo float sensors network, the time of each sensor node is critical information to accurately rebuild the event and carry out meaningful analysis. Hence, precise synchronization is an important function in the Argo float sensors network in sea surface salinity application. However, it is not an easy job to maintain time synchronization among these distributed Argo float sensors. In the recent years, time synchronization remains an active research area in the field of the wireless sensor network [7-11]. In the literature, we observe that additional hardware is required in each sensor node to reach time synchronization in some proposed schemes [12-13]. Evidently, this type of approach will increase the cost of the sensor. In addition to the hardware cost, there are numerous performance factors that should be considered when we design a synchronization scheme for the wireless sensor network. We will discuss some of these important factors in the following.

1. **Scalability:** In a sensor network, there are hundreds of thousands of sensor nodes. We must consider a meth-od of time synchronization that can scale well in huge volume and high density environments. Or, we might deploy some new nodes to expand our sensing fields after initial deployment in some situations. The new sensor nodes should follow the design synchronization scheme quickly and easily. Specifically, we refer to the factors including the number of message exchanges, the intensity of computation, and the synchronization convergence time associated with the task. These are costs and should not increase drastically as the size of the network grows. Otherwise, the cost will be high enough to prevent the deployment of such task.

2. **Robustness:** A sensor network is usually deployed in unattended areas, e.g., high mountains, a dangerously steep valley, and volcanoes. The user wishes to observe some special data for a long time. But, due to the restriction imposed in the environment, a few sensor nodes may not communicate well. Or, some nodes can be damaged during the deployment. These problems should not compromise the functions of the network. So, we should find a way to compensate for the dead nodes and those in isolated locations and keep the whole network operate normally.

3. **Precision:** In some applications such as earthquake simulation, the sensor nodes need to be precisely synchronized with one another. When an earthquake happens, the sensor nodes report the related information in every different area to the user, such as the epicenter, earthquake intensity, and the caused damage etc. If the sensor nodes are not synchronized, the collected data may be not useful. This is because the reported events from the sensors can not be accurately sorted according to the inconsistent clocks. As a result, it will be difficult to make a faithful recurrence of the earthquake in the simulation.

4. **Efficiency:** Power consumption is one of the most important considerations for any protocol design in the wireless sensor network. In the sensor network, the message delivery and reception need to consume power. So, we wish to minimize the number of message exchanges in time synchronization. On the other hand, it will conserve power if the nodes are activated only when necessary. They can remain idle otherwise. In this paper, the proposed self-coupled scheme allows the sensor nodes to be turned on and off for the sake of power saving while maintaining a satisfactory level of synchrony.

5. **Period:** In the sensor network, we synchronize all nodes at once. However, the clocks of the nodes may still proceed at different speed. As time passes, these discrepancies will lead to unacceptable asynchrony. Without proper handling, any pair of nodes will be far apart in their respective times. Since we

need periodical effort in synchronizing all sensor nodes, the cost of the scheme incurs constantly and can not be regarded as one-time expense.

In this paper, we propose a self-coupled method that can achieve synchronization while exhibiting good cost characteristics. Later we will show in the simulation that it outperforms the existing hierarchical schemes with respect to the said performance metrics in the Aquarius/SAC-D project.

The remainder of this paper is structured as follows. In Section 2, we describe a routing protocol that provides inspiration to our protocol design. In addition, several existing synchronization protocols specifically designed for the wireless sensor network will be discussed in this Section. In Section 3, we will introduce the proposed synchronization scheme, the self-coupled time synchronization, in greater detail. An enhancement to save the message complexity and power is also suggested in the Section. In Section 4, we explain the simulation model and demonstrate the simulation results. Finally, we conclude this paper and point out possible future works in Section 5.

2 Related works

Sergio Barbarossa proposed a protocol with information propagation based on mutual coupling of dynamic systems. The information spreads as the result of the couplings among adjacent nodes. One example is that the coupled nodes mutually act to adapt their clocks. In the protocol, time synchronization can be reached by each node using the mathematical Equation (1):

$$\dot{\theta}_i(t) = \omega_i + \frac{K}{c_i}\sum_{j=1}^{N} a_{ij} F\left[\theta_j(t) - \theta_i(t)\right] \tag{1}$$

where $\theta_i(t)$ is the time of the *i*th sensor. a_{ij}'s are real variables that describe the level of coupling between sensors *i* and *j*. They are all positive values indicating that sensor *i* makes an adjustment to reduce the time difference to sensor *j*. Depending on the application, these values can be determined according to strength of pulses or distance. The larger the value is, the stronger the coupling becomes. K is a control loop gain that is used to judge the synchronization convergence time. A small *K* may result in a prolonged synchronization convergence time while a large *K* can cause violent oscillation preventing synchronization. c_i is a coefficient that individually quantifies how the *i*th sensor adapts its clock to external impacts. ω_i is the original clock speed of the *i*th sensor. By its self, sensor *i* will continue ticking at ω_i. The function, $F(\cdot)$, should be an monotonically increasing nonlinear odd function. The definition of odd functions indicates that $f(-x) = -f(x)$. Two possible choices for $F(x)$ are Equation (2) or (3) as suggested in [14-16]. The reason for using a monotonically increasing function is the following. Adequate adjustment should be made to the sensor node *i* based on the difference between its clock and its neighbors'. As the difference grows, the corresponding adjustment should not decline. Hence, only monotonically increasing functions will be suitable in this application. And, the nonlinear property is instrumental to restrain the adjustment with rather severe clock discrepancies. It is known that aggressive adjustment may lead to violent oscillations and, eventually, divergence.

$$F(x) = \frac{e^{\lambda x} - 1}{e^{\lambda x} + 1} \tag{2}$$

$$F(x) = \sin(2\pi \frac{x}{T}) \tag{3}$$

The pros and cons of this protocol are discussed in the following. In this coupled dynamic system, synchronization can be achieved efficiently. By the aforementioned local timing exchanges, a sensor can adjust its clock according. With every node following the protocol, it is shown that global synchronization can be reached rather quickly. This method does not require a timing distribution tree. Neither is it subject to the single point of failure in the timing reference node. However, it is unclear if it suffers from long synchronization convergent time and excessive message exchanges. If so, then this approach is not justified in the wireless sensor network. Next, we will review some of the synchronization schemes based on hierarchical architecture. In general, these schemes form a synchronization tree within the network and use it as the platform to distribute reference clock. In the time synchronization research area, there are many variations based on a hierarchical structure [17-22]. They share common characteristics in building a tree-like structure. In the structure, nodes are divided into layers of various depths. The functions of a node may vary according to its location in the tree-like structure. Time synchronization is carried out one layer at a time from the root down to the leaves. In the following, we will introduce two such protocols and discuss their strength and weakness.

Mamun-Or-Rashid, Hong and In [23] proposed a protocol called passive cluster based clock synchronization in sensor networks. The protocol first creates clusters by the passive clustering method. The passive clustering method is carried out on demand and the formation of clusters is dynamic. The clustering is initiated when the first data message is sent. In the passive clustering method, nodes can have four states: (1) Initial, (2) Cluster head, (3) Gateway, and (4) Ordinary. First, a timing distribution platform is built based on the passive clustering method. This platform can be persistent and used for periodic synchronization. Or, it is possible to construct it dynamically when there is a need for synchronization. In this method, the nodes receiving a request of time synchronization from any node become the initial nodes marked in Fig.4a. Then, the initial nodes follow the rule of first declaration to compete for being cluster heads in Fig.4b. An initial node can announce itself as a cluster head without further checking with the neighbor nodes in Fig.4c. Upon receiving such an announcement, the neighbor nodes will give up the competition of being cluster heads. If a node receives packets from two cluster heads, then it becomes a gateway. Besides the cluster heads and gateways, the remaining nodes are called ordinary nodes in this platform.

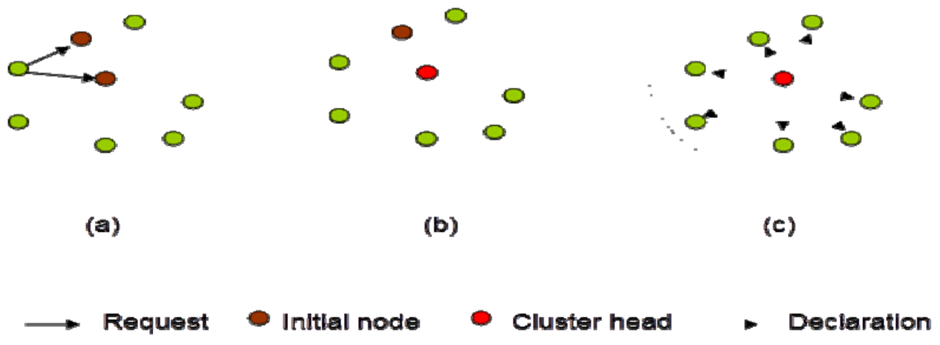

Fig. 4. Initialization in passive clustering method

Once the platform is created in Fig.5, the synchronization is performed following the asynchronous averaging algorithm. The cluster heads collect the clock information from its respective neighborhoods. Within each cluster, the information is averaged by the cluster head and sent back to the neighbor nodes. The nodes in the cluster use this information to adjust their local time. The inter-cluster synchronization is achieved though the gateways bridging the clusters.

Since a gateway is under the influence of the associated clusters, the averaged time information of one of its cluster will be used in the averaging process in another cluster. It is not clear how the cluster oriented distribution platform performs. It seems to reduce the message overhead and make the synchronous process energy efficient. However, it is not free to build the clusters. If they are created on demand, a repeated cost will be involved in this procedure.

On the other hand, the cluster heads and gateways will have higher workload than the ordinary nodes if the static structure is employed. Once the clusters are in place, it is unclear how long it will take for the network to synchronize. We suspect a longer convergence time when comparing with the flat topology where all nodes carry out the asynchronous averaging algorithm. The reason is that the distribution of time information among clusters mainly depends on the gateways in the cluster oriented approach. The benefit of relieving the ordinary nodes from more engaging computation seems mitigated by the longer convergence time. Hence, it is not conclusive to claim the cluster oriented approach has better performance.

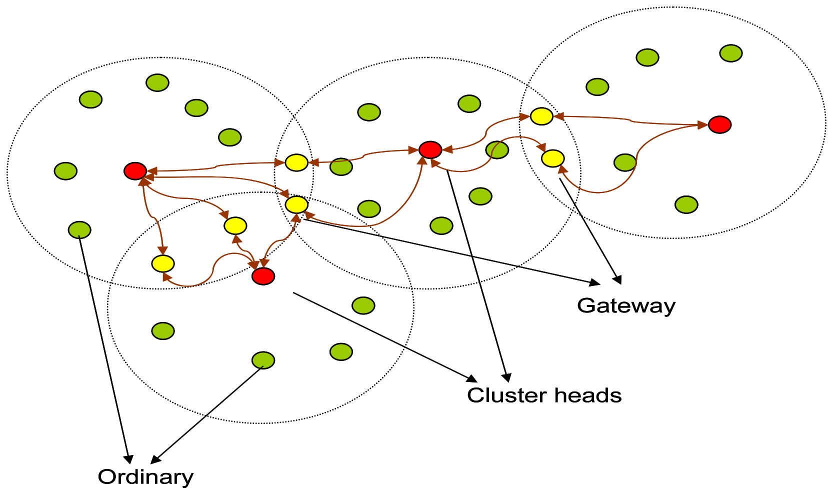

Fig. 5. States of the structure

Ganeriwal, Kumar and Srovastava [24] proposed a time synchronization protocol, which is called the timing-sync protocol for sensor networks (TPSN). It aims at providing network-wide time synchronization in the sensor network. The TPSN works in two phases for time synchronization. In the initial phase called the *level discovery phase* (LDP), a hierarchical tree structure is established in the network. Fig.6 shows the propagation of the level discovery phase. The second phase known as *synchronization phase* (SP) performs a pair-wise synchronization along the edges of the discovered tree. It establishes a global time throughout the network. Eventually, all nodes in the network synchronize their clocks to the reference node (known as the root node). The level discovery phase starts at the root node or the initiator (known as the sink). The root node is assigned as level number 0 and initiates this process by broadcasting a *level_discovery* packet to its neighbor nodes. This packet contains the identity and the level number of its sender. When the neighbor nodes of the root node receive this packet, they will assign themselves a level number accordingly. The assigned level number is greater than the level contained in the packet they just received by one. This phase will continue until all nodes in this sensing field obtain their level numbers.

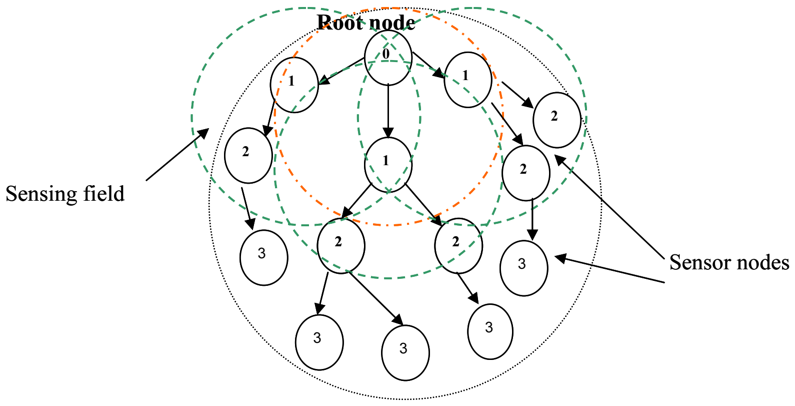

Fig. 6. LDP of TPSN

Fig.7 shows the two-way message exchange between nodes A and B. T_1 and T_4 are the measured time epochs by the local clock of A. On the other hand, T_2 and T_3 are measured by the local clock of B. At the time T_1, node A sends a *synchronization_pulse* packet to node B. This *synchronization_pulse* packet contains two pieces of information, (i) the level number of A and (ii) the value of T_1. At the time T_2, node B receives the packet and records the receiving time locally. T_2, of the clock of node B, is equal to $T_1 + \Delta + d$, where Δ and d represent the clock drift time between the pair of nodes and the propagation delay, respectively. It should be noted that Δ may not be time invariant. However, it is considered constant for the short period of the two-way exchange process. At the time T_3, node B finishes processing *synchronization_pulse* packet and sends back an *acknowledgement* packet to node A. This acknowledgement packet contains the level number of *B* and the values of T_1, T_2 and T_3. At the time T_4, node *A* receives the packet. Similarly, T_4, of the clock of node *A*, is equal to $T_3 - \Delta + d$. Finally, these parameters are used by node *A* to calculate the Δ and *d* in the following equations.

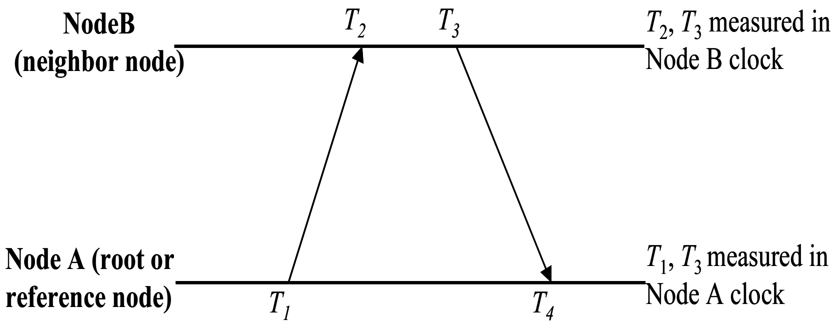
NodeB
(neighbor node)
T_2
T_3
T_2, T_3 measured in Node B clock
Node A (root or reference node)
T_1
T_4
T_1, T_3 measured in Node A clock

Fig. 7. Two-way Message Exchange Method

When the Δ and *d* are calculated by Equation (4) and Equation (5) accordingly, the node *A* can correct its clock so that it synchronizes to node *B*. After the LDP, the synchronization phase will begin at the root node. Base on the established tree structure, the synchronization phase (SP) starts its process at the root node. It broadcasts a *synchronization_pulse* packet to its neighbor nodes. Each neighbor node will use the two-way message exchange method to synchronize with the root node. After synchronizing with the root node, its neighbor nodes will continue to broadcast their *synchronization_pulse* packets to synchronize with their respective neighbor nodes. Eventually all nodes in the sensing field are synchronized to the root node. The pros and cons of this protocol are discussed in the following. The protocol can reach time synchronization by the two-way message exchange method. And, there is no complicated step to create the tree-like structure. But, it still divides into two steps to complete time synchronization. In a stretched network, the protocol will need to spend much time to create the tree-like structure because it will have a greater number of levels. As a result, the synchronization convergence time increases linearly as the number of levels grows since synchronization is achieved one level at a time. In summary, this type of protocols suffers from the tree construction cost and poor scalability in convergence time.

$$\Delta = \frac{(T_2 - T_1) - (T_4 - T_3)}{2} \quad \textbf{(4)}$$

$$d = \frac{(T_2 - T_1) + (T_4 - T_3)}{2} \quad \textbf{(5)}$$

3 Self-coupled time synchronization for argo float sensors network

In this Section, we give an in-depth introduce to the self-coupled synchronization method. The system model, simulation flowchart, system characteristics, performance metrics and an enhancement will be covered in the following discussion.

3.1 System model

In the wireless Argo float sensors network, a node *I* has its local time $\theta_i(t)$ where *t* is the virtual global time in the network. The virtual global time is the standard clock in the network. In our model, we assume the identical communication range and two nodes within the communication range of each other are neighbors. The local time of each node is periodically exchanged with its neighbors. When receiving the clock information from the neighbors, the sensor node uses Equation (6):

$$\dot{\theta}_i(t) = \dot{\theta}_i(t-\Delta t) + \frac{K}{c_i}\sum_{j=1}^{N} a_{ij} F\left[\theta_j(t) - \theta_i(t)\right] \quad \textbf{(6)}$$

to compute its clock rate, $\dot{\theta}(t)$. $\dot{\theta}(t)$ is set to 1.0 initially.

It should be noted that in Equation (1). ω_i denotes the clock rate of node *i*. All ω_i 's are nominally the same for their respective nodes. The main difference of Equation (1) and (6) is that the former intends to synchronize the time while the latter aims to synchronize both the time and the rate. The other parameters in Equation (6) are described in Section 3.2. They are listed in the following for convenient reference.

- K : the control loop gain.
- a_{ij} : the weight for the coupling between the two nodes *i* and *j*. In the simulation, if the node *j* is in the coverage radius of node *i*, then $a_{ij} = 1$. Otherwise, $a_{ij} = 0$.
- c_i : the sensitivity parameter. For the sake of simplicity, it is set to 1 for all nodes.
- *F*(*x*): the adjustment function. We let $F(x) = \sin(2\pi \frac{x}{T})$. T is the cycle period.

The clock time is computed and adjusted every *Δt* period with the following Equation (7):

$$\theta_i(t+\Delta t) = \theta_i(t) + \Delta t \dot{\theta}_i(t) \quad \textbf{(7)}$$

Based on the system model, the simulation model is produced. Fig.8 illustrates the simulation flowchart for the self-coupled scheme. We used the normalized method in [25]. We will next introduce the function of each phase in the flowchart. In the initialization phase, the related topological parameters are determined which include the number of sensor nodes, deployment area, communication range of the nodes, and so on. Next, we have to decide how to deploy all sensor nodes. In sea surface salinity application of the Argo float sensors networks, sensor nodes are randomly deployed in the area of interest.

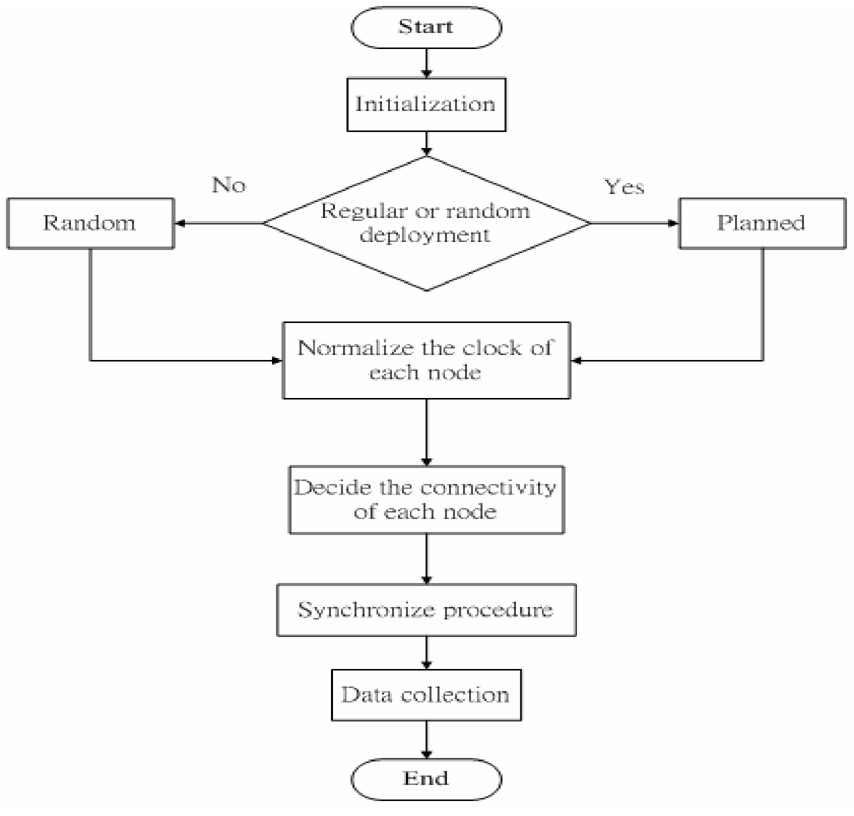
Start
Initialization
No
Yes
Random
Regular or random deployment
Planned
Normalize the clock of each node
Decide the connectivity of each node
Synchronize procedure
Data collection
End

Fig. 8. Simulation Flowchart

In other cases, they are deployed following a planned pattern. For instance, they can be placed in a chessboard like crossbar switches or in hexagonal cells like mobile networks [26-27]. So, we consider both random and planned deployment methods in our simulation. According the flowchart, we then randomly assign a clock rate to each sensor node which defines the speed of clock ticking. The initial time of node *i* is also randomly assigned within a small range. Another important system parameter is the degree. It represents the number of neighbors of a sensor node. It should be noted that this parameter is in statistical sense in the random deployment method. That is, it denotes the average degree of each node. As we will show later, this degree parameter turns out to be a significant system characteristic in the self-coupled synchronization scheme. In the synchronization phase, it is important step to adjust the clock rate and the time difference of the sensor node. In the data collection phase, we record the performance metrics of interest. These metrics will be covered in the next section.

3.2 System characteristics and performance metrics

In this section, we will discuss the related system parameters and their impacts on the synchronization performance. We will introduce these parameters and metrics including node configuration, virtual cluster, and synchronization time in the next few subsections. The node configuration is discussed in Subsection 3.2.1. It is an input in our simulation. It is important to understand if the distribution of sensor nodes will cause any difference in performance while other factors remain the same. Subsection 3.2.2 covers the concept of virtual cluster, which represents a synchronous region. Nodes in different virtual clusters are most likely asynchronous in time. We use the adjacency matrix to represent the network topology [28]. Subsection 3.2.3 discusses synchronization time of the self-coupled method. Synchronization time is the period of time needed for the Argo float sensors network to reach synchronization. Of course, a shorter synchronization time is in favor. The number of virtual clusters and synchronization time are indeed the outputs in the simulation while the node configuration plays the input role.

3.2.1 Node configuration

In our simulation, we consider two deployment methods as explained previously. In general, there are practical applications that can employ the random or planned deployment methods. We would like to study if the deployment methods will cause performance discrepancy in the self-coupled synchronization scheme. Figure 9 shows an example of the random deployment method. In this example, there are 50 nodes scattered between +0.3~-0.3 in the *x* axis and +0.3~-0.3 in the *y* axis. The communication range is set to 0.1. In the Argo float sensors network, each node has its own communication range. When a node is in the communication range of the initiator, it is called a single hop node. Or, equivalently, it is one layer from the initiator. In the tree-like structure, the number of layers is one of the important characteristics of the Argo float sensors network. In the case of Fig.9, it is estimated that there are about 8 hops between the two farthest nodes in the worst case because the distance of the diagonal of the region is $6\sqrt{2} = 8.48$ and the communication range is 0.1. The larger the number of the layers is, the longer the synchronization time required by the hierarchical synchronization procedures is.

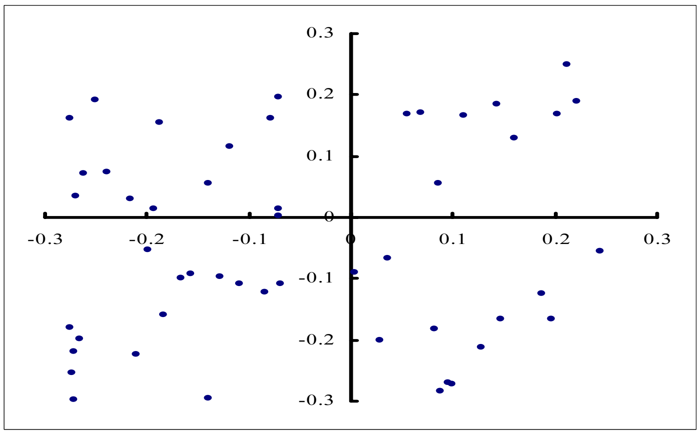

Fig. 9. Random deployment

We would like to explore if such a relation exists for the self-coupled approach. The result of this exploration can help us to further understand the scalability of this approach. In addition to the layers in the network, it is our belief that the degree value also plays an important role in the synchronization procedure. The self-coupled method relies on the feedback of the neighbors to adjust its local clock. In the random deployment method, it is difficult to have precise control over this value. In general, the node density is tuned to achieve different average degrees. Next, we show an example of the planned deployment in Fig.10. We set the communication range to 0.1 for all sensor nodes. So, the maximum degree of Fig. 10 is equal to 4. Other similar patterns with various degrees may be carefully created to study the impact of degrees to the synchronization schemes. Fig.11 and Fig.12 depict the patterns with the maximum fixed degrees of 8 and 10. The networks generated by the two deployment methods are used for the study on the effect to the protocol performance.

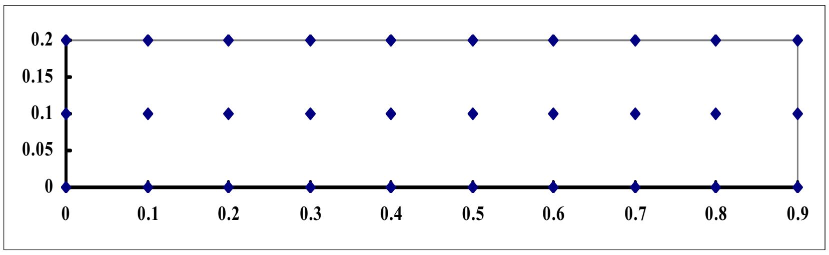

Fig. 10. Planned deployment (Degree = 4)

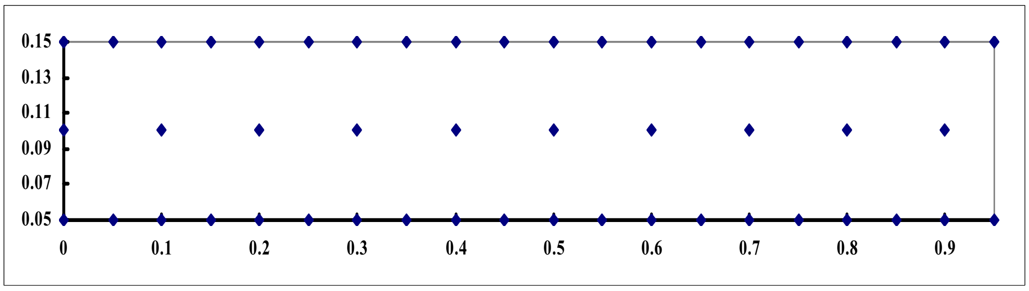

Fig. 11. Planned deployment (Max Degree = 8)

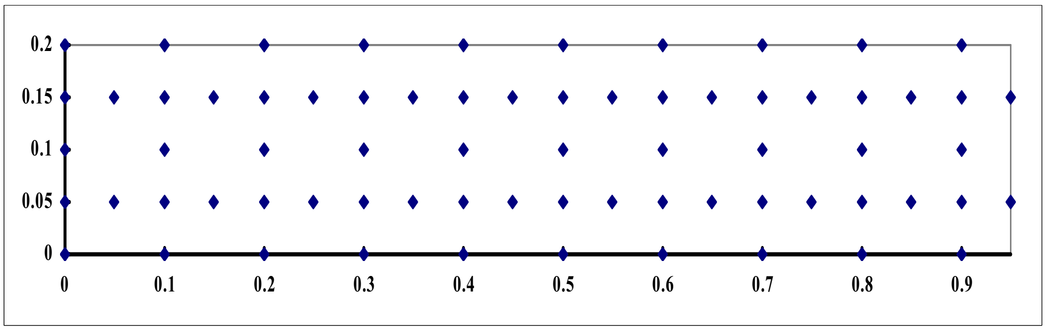

Fig. 12. Planned deployment (Max Degree = 10)

3.2.2 Virtual cluster

A virtual cluster is a group of nodes. There exists a viable connection route between any pair of nodes in the group. Because every node has a limited communication range, not all nodes can communicate with each other in a random deployment. If there are two such groups, we say there are two virtual clusters in the network. It should be noted that there is a single virtual cluster in the planned deployment. On the other hand, once a random deployment is produced, the virtual clusters can be identified by the adjacency matrix (Fig.13). In our simulation, we compute the distance between two nodes and decide if they are neighbors in the planned deployment. If they are neighbors, then the corresponding entries in the adjacency matrix will be set to 1. Otherwise, the entries will be set to 0. In the meantime, aij is set to the same value as its corresponding entry in the adjacency matrix. In other words, the time difference between one node and any other node in the network will have the same impact to the local clock adjustment. In the simulation, the number of clusters in the topology can be calculated using the adjacency matrix.

aij	*i*=0	1	2	3	4	5	6	7	8	9
j=0	1	1	0	0	0	0	0	1	1	1
1	1	1	0	0	0	1	0	0	0	0
2	0	0	1	0	0	0	0	0	0	0
3	0	0	0	1	1	0	1	0	0	0
4	0	0	0	1	1	0	1	0	0	0
5	0	1	0	0	0	1	0	0	0	0
6	0	0	0	1	1	0	1	0	0	0
7	1	0	0	0	0	0	0	1	1	1
8	1	0	0	0	0	0	0	1	1	0
9	1	0	0	0	0	0	0	1	0	1

Fig. 13. Adjacency matrix

For example, there are two clusters and an isolated node in Fig.13. The two clusters include nodes 0, 1, 5, 7, 8, 9 and nodes 3, 4, 6, respectively. The virtual cluster is very important in achieving synchronization. Different clusters are considered in different synchronization region and will in general be asynchronous. Despite of the vast research efforts on term weighting schemes, further study is still needed to reveal whether inverse category frequency can be beneficial to term weighting schemes and how to realize it. Our study in this paper just tries to make some contributions to this problem.

For example, there are two clusters and an isolated node in Fig.13. The two clusters include nodes 0, 1, 5, 7, 8, 9 and nodes 3, 4, 6, respectively. The virtual cluster is very important in achieving synchronization. Different clusters are considered in different synchronization region and will in general be asynchronous. Despite of the vast research efforts on term weighting schemes, further study is still needed to reveal whether inverse category frequency can be beneficial to term weighting schemes and how to realize it. Our study in this paper just tries to make some contributions to this problem.

3.2.3 Synchronization time

In the simulation, we first randomly assign clock rates to all nodes between 0.3 and 1.3. Then, the initial time is randomly set to 0, 1, or 2. Fig.14 and Fig.16 show the local clock rate of each node in the *y* axis and the number of iteration in the *x* axis. Fig.15 and Fig.16 show the local times move along and become synchronous as the simulation moves on. In the simulation, one iteration means the amount of time required for the one local exchange among all the neighbors. Each iteration is of the same length. Specifically, we set the iteration time, Δt, to 0.01 time unit. Fig.14 is one case that all nodes synchronized before 200 calculated iteration. Fig.15 presents that the local clocks become stable after around 200 iterations. Then, it turns into almost a straight line. Fig.16 shows the case where all nodes cannot synchronize together because there are multiple virtual clusters. These clusters synchronize within themselves independently. Hence, there are several bundles of lines in Fig.17. Each bundle represents that the local clocks in a virtual cluster synchronize together.

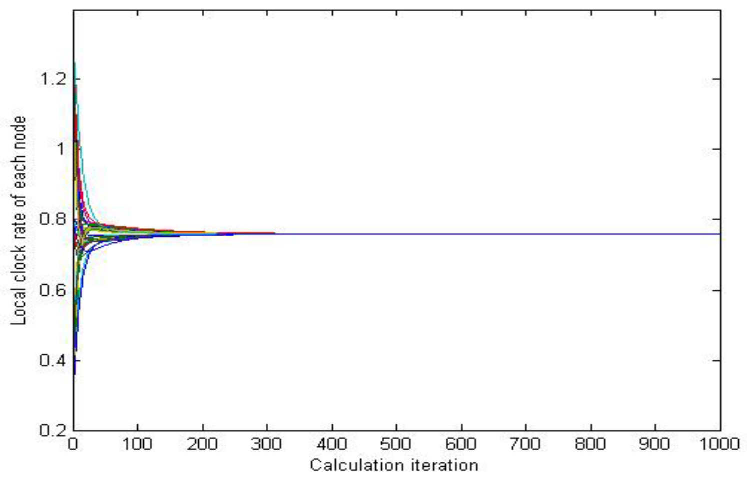

Fig. 14. Synchronize clock rate diagram

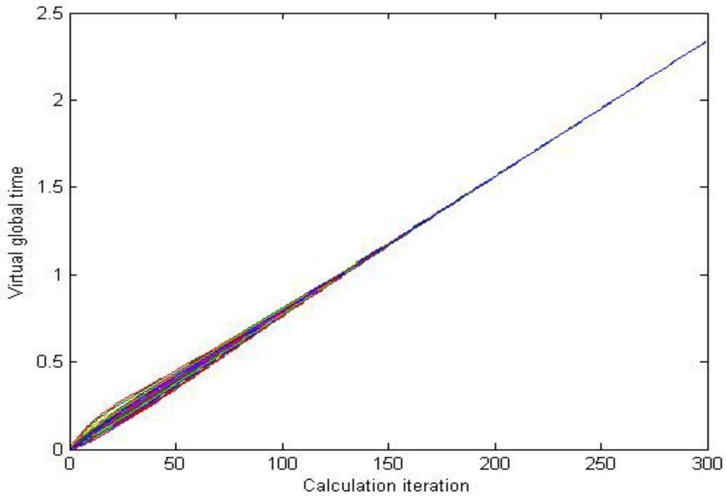

Fig. 15. Virtual global time diagram

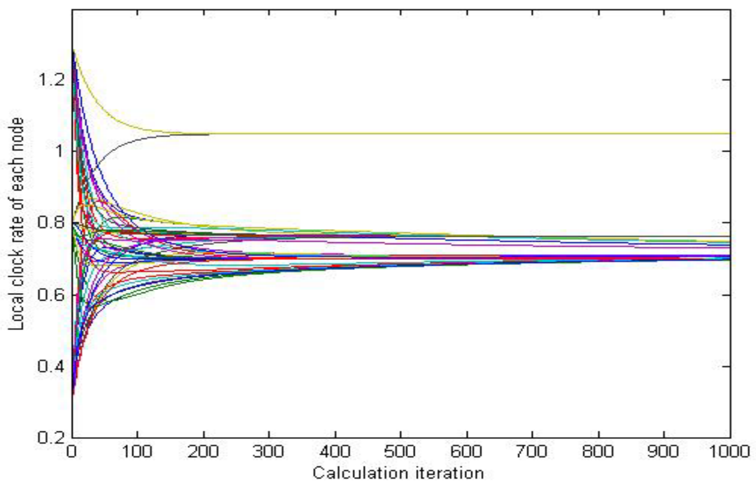

Fig. 16. Nonsynchronize clock rate diagram

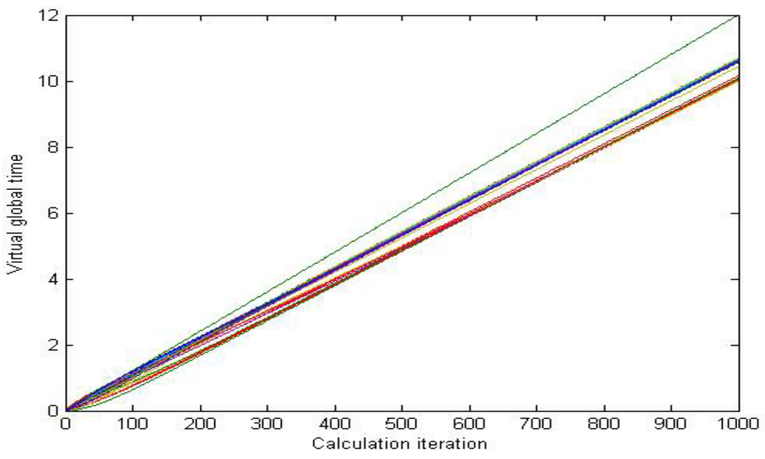

Fig. 17. Virtual global time diagram

3.3 Impacts of degree and random participation

In our simulation, we wish to identify the most significant factor affecting the system performance in the self-coupled synchronization protocol. Node degree in the fixed areas, the communication range of each node, the diameter of the topology, node's virtual cluster and the suitable calculation iteration are important in the protocol. The node degree is related to the number of sensor nodes in the fixed area and the communication rage of each node. The diameter of the topology is the farthest distance between any two nodes in the Argo float sensors network. We use the hop to be a unit of the distance. One hop is the maximum range that the sensor nodes can communicate with each other. The virtual cluster is the number of groups that the topology organized. Each cluster ideally synchronized in the same time field. The calculation iteration is the synchronized times when the nodes' time converged in the fixed range. And, we consider the regular deployment of the sensor nodes. In our simulation, we design the degree 2, 4, 8, 10. The degree is the maximum number of the neighbor nodes. There are some edge nodes that they have few neighbor nodes. Random participation is an enhancement to cope with the limited energy in the sensor nodes. In this enhancement, only a part of the nodes are randomly selected to actively join the protocol in a synchronization cycle. That is, they will perform the self-coupled procedure as we describe previously. The rest of the nodes modify their clocks by passively listen to these active nodes. They merely apply Equation (6) to modify their clocks without contributing their clock information in the process. A different set of active nodes will be chosen in the beginning of each synchronization cycle. In summary, this method can save the transmission energy in four fifths of the nodes in a synchronization cycle. Statistically, the load of synchronization is evenly distribution among the nodes.

4 Performance evaluations and analysis

In this Section, we show the data of our simulation. The Visual C++ program is created to simulate our protocol. Different deployment patterns are considered in the simulation.

4.1 Simulation environment and parameters

Specifically, two categories of deployment are considered in the simulation. We want to know whether the node density is one important factor in the protocol. So, we consider that the degree increase in the planned deployment. We also design the different sensing range and the number of nodes in the fixed sensing field ± 0.3 X ± 0.3. In addition, we simulate the topology becomes double in the sensing field ± 0.6 X ± 0.6. And, we proved that self-coupled method is suitable for high density and great topology in Argo float sensors networks. In the randomly deployment, we also evaluate the degree and clusters relationship. We also find a function that can capture the relationship between the degree and clusters. Then, we simulate the random participation that it randomly selects only one fifth sensor nodes to operate the self-coupled scheme. The other nodes only can be influenced by the one fifth nodes in their sensing ranges. Finally, we simulate the error of the self-coupled method. The error is between the average clock of all nodes and the modulated local clock of each node.

4.2 Static traffic scenarios

In this Sub-section, we explore the performance of the planned deployment. In the first simulation, there are 10 nodes lined up linearly. That is, each node has the degree of two. Except for the boundary nodes, all nodes have two neighbors. We ignore the boundary nodes in the simulation. The first and the last nodes are boundary nodes. Therefore, we consider 8 nodes in this simulation. Fig.18 represents the simulation result. Although the degree is merely two in this network, its planned topology still results in one virtual cluster. It doesn't have any isolated sensor nodes. And, we define that the synchronization is achieved if the difference between the clock rates of every pair of nodes is less than 5% of each clock rate of this pair of nodes.

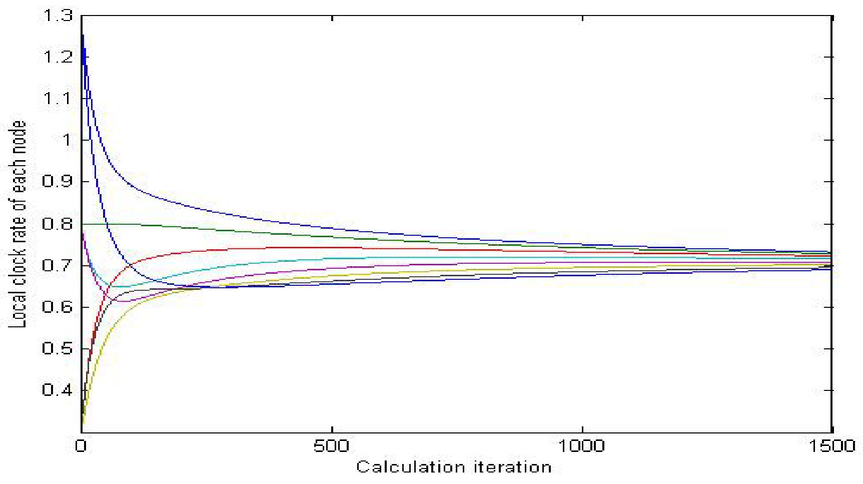

Fig. 18. Degree = 2 Time Synchronization

This is a reasonable benchmark as observed in the literature [15]. According to this definition, all sensor nodes can synchronize in 1500 calculation iteration. Fig.19 represents the simulation result that the degree is merely four. All sensor nodes are synchronized in 1000 iterations. Although its number of iteration is still high, its synchronization time is faster than that in the previous case where the network has the degree of two.

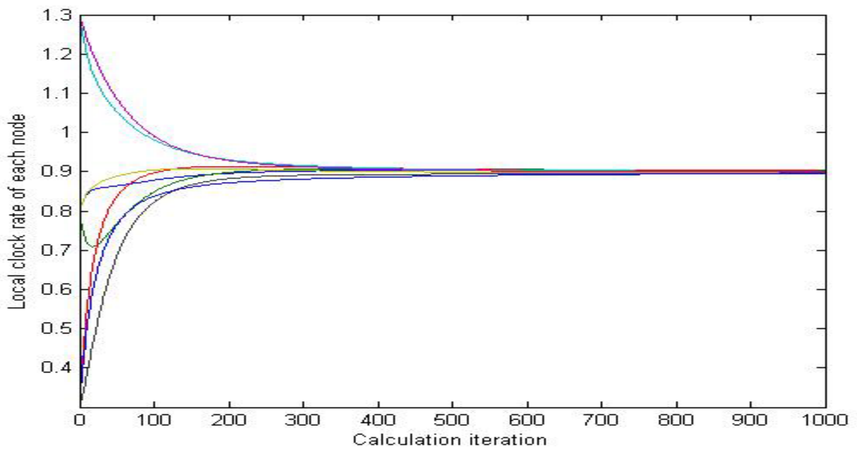

Fig. 19. Degree = 4 time synchronization

Fig.20 represents the simulation result that the Maximum degree is eight. Fig.21 represents the simulation result that the Maximum degree is ten. Degree 8 and 10 conform our hope that all nodes can synchronize in 1000 iteration.

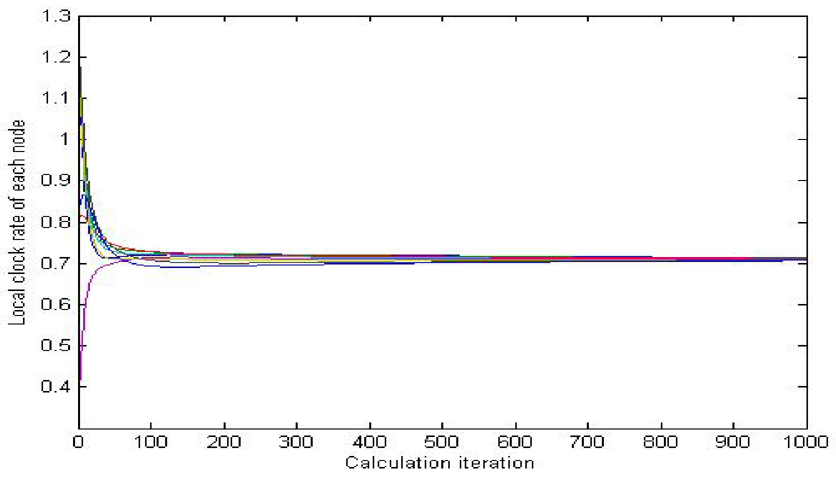

Fig. 20. Max degree = 8 time synchronization

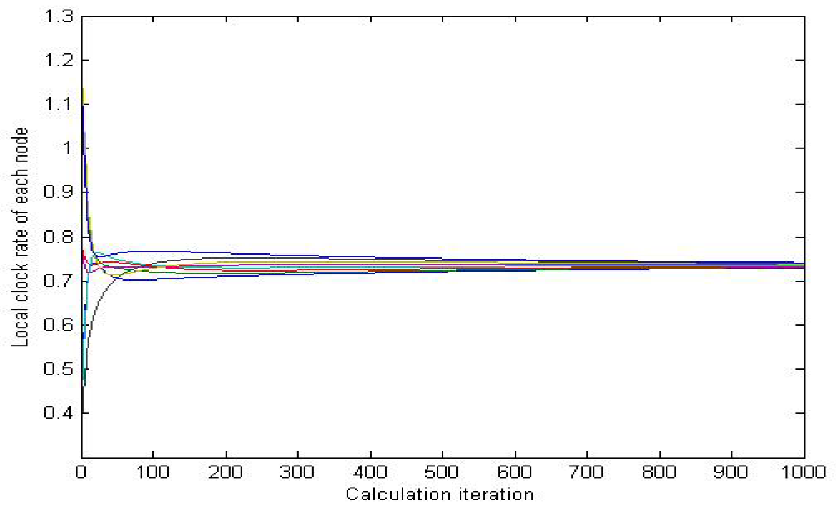

Fig. 21. Max degree = 10 time synchronization

4.3 The analysis of random deployment

Sensor nodes are often deployed in areas that are dangerous or hard to get access to. In this subsection, we study the performance in the cases of random deployment. We want to confer with the influencing aspects in the protocol. But, the method in randomly deployed sensor nodes has some factors to influence the node density. It includes sensor node's sensing range, the number of sensor nodes in the topology, and the larger topology. The cluster does not include the isolated sensor nodes. If degree equal to zero, we consider it is an isolated sensor node. And, we give a synchronized definition that the difference of all nodes' clock rate is below 5% through after 1000 calculation iteration. We calculate the difference of all nodes' clocks per 100 calculation iteration. If it is below 0.1%, we represent the new calculation iteration that means all nodes synchronized early. Table represents the difference sensing range simulations. Table 1 shows the ten samples in 0.05 sensing range. Table 2 and Table 3 are in 0.1 and 0.2 sensing ranges. Table proved the self-coupled method in randomly deployment is suitable for high density Argo float sensors networks. In Table 1 and Table 2, there are some isolated sensor nodes. And, most of them cannot only generate one cluster. It is the most important factor that all nodes cannot synchronize. They are self-synchronized. But, finally it is not converged. Table 3 shows early synchronized state in high density and one cluster case. We consider the hierarchical protocol performance becomes worse when the topology becomes bigger. We set the sensing field double in ± 0.6 X ± 0.6 to simulate. We relatively increase density in the simulation. Table 4 shows the result. Although, there are more sensor nodes to be synchronized, its performance is still in our hope.

Table 1. Sensing range = 0.05

Influence / Environment	Average degree	Max. degree	Min. degree	Cluster (d>=1)	Synchronized
Nodes:	1.36	3	0	16	No
50	1.28	5	0	14	No
Time range:	0.96	3	0	12	No
0~2	1.44	4	0	15	No
Sensing range	1.2	3	0	13	No
0.05	1.44	5	0	10	No
	0.96	3	0	14	No
Sensing field:	1.4	4	0	11	No
±0.3×±0.3	1.32	3	0	14	No
	1.6	0	0	11	No

Table 2. Sensing range = 0.1

Influence / Environment	Average degree	Max. degree	Min. degree	Cluster (d>=1)	Synchronized
Nodes:	4.04	10	1	4	No
50	4.4	11	0	7	No
Time range:	4	10	1	8	No
0~2	3.48	7	0	4	No
Sensing range	4.24	8	0	4	No
0.1	3.84	7	1	4	No
	5	11	0	4	No
Sensing field:	4.16	8	0	2	No
±0.3×±0.3	3.36	8	0	1	No
	3.68	7	0	3	No

Table 3. Sensing range = 0.2

Influence / Environment	Average degree	Max. degree	Min. degree	Cluster (d>=1)	Synchronized	Converged clock speed	Calculation iteration (K=1000)
Nodes:	13	21	4	1	Yes	1.220	400
50	14.16	23	6	1	Yes	1.160	300
Time range:	13.16	18	6	1	Yes	0.840	400
0~2	14.04	22	6	1	Yes	0.980	200
Sensing range	12.96	21	5	1	Yes	0.920	300
0.2	13.08	18	4	1	Yes	0.840	200
	13.96	20	5	1	Yes	1.040	200
Sensing field:	13.96	23	6	1	Yes	1.000	200
±0.3×±0.3	12.76	21	3	1	Yes	0.880	300
	14.36	24	4	1	Yes	1.020	400

Table 4. Diameter

Influence / Environment	Average degree	Max. degree	Min. degree	Cluster (d>=1)	Synchronized	Converged clock speed	Calculation iteration (K=1000)
Nodes:	17.96	32	4	1	Yes	1.081~1.088	1000
200	17.66	31	3	1	Yes	0.940	600
Time range:	18.71	36	7	1	Yes	1.050	600
0~2	17.62	31	4	1	Yes	0.965	500
Sensing range	18.3	32	5	1	Yes	0.987~0.994	1000
0.2	17.17	29	4	1	Yes	1.083~1.088	1000
	18.08	33	4	1	Yes	0.963~0.982	1000
Sensing field:	16.14	28	4	1	Yes	0.994~0.996	1000
±0.6×±0.6	18.05	34	4	1	Yes	0.948~0.961	1000
	18.32	35	4	1	Yes	0.935	500

In Table 5 to 7, we try to increase the number of sensor nodes in the fixed sensing range 0.1. Table 5, Table 6 and Table 7 individually represent the number of sensor nodes 100, 150 and 200. In these cases, there are no isolated sensor nodes. In Table 5, there is three cases that the topology only generates one cluster. But, the difference of all nodes' clock rate is not below 5%. We consider that they are not synchronized. It maybe synchronizes in more calculation iteration. Table 6 represents that all nodes synchronized in 1000 calculation iteration. Table 7 represents that its performance is better than Table 6.

Table 5. Number of sensor nodes = 100

Influence / Environment	Average degree	Max. degree	Min. degree	Cluster (d>=1)	Synchronized
Nodes:	8.44	14	2	2	No
100	9.4	20	3	1	No
Time range:	8.2	17	2	2	No
0~2	8.62	16	1	3	No
Sensing range	7.92	15	2	1	No
0.1	9.52	21	2	2	No
	8.46	18	2	3	No
Sensing field:	8.72	19	2	3	No
±0.3×±0.3	8.32	14	1	2	No
	7.54	13	2	1	No

Table 6. Number of sensor nodes = 150

Influence / Environment	Average degree	Max. degree	Min. degree	Cluster (d>=1)	Synchronized	Converged clock speed	Calculation iteration (K=1000)
Nodes:	14.17	24	2	1	Yes	0.94~0.98	1000
150	13.52	26	2	1	Yes	0.85~0.87	1000
Time range:	14	26	3	1	Yes	1.07~1.10	1000
0~2	13.41	23	6	1	Yes	1.12~1.13	1000
Sensing range	13.04	29	2	1	Yes	0.91~0.93	1000
0.1	13.27	24	5	1	Yes	1.16~1.20	1000
	12.39	22	5	1	Yes	1.01~1.06	1000
Sensing field:	13.89	26	4	1	Yes	0.89~0.90	1000
±0.3×±0.3	12.87	22	6	1	Yes	1.04~1.06	1000
	12.53	21	3	1	Yes	1.08~1.09	1000

Table 7. Number of sensor nodes = 200

Influence / Environment	Average degree	Max. degree	Min. degree	Cluster (d>=1)	Synchronized	Converged clock speed	Calculation iteration (K=1000)
Nodes:	19.58	36	4	1	Yes	1.043~1.049	800
200	17.58	29	5	1	Yes	0.90~0.91	1000
Time range:	16.64	30	4	1	Yes	1.044~1.045	600
0~2	17.31	31	7	1	Yes	0.97~0.98	1000
Sensing range	17.07	29	5	1	Yes	1.014~1.017	600
0.1	17.06	28	6	1	Yes	0.938~0.942	600
	17.41	34	3	1	Yes	1.116~1.124	700
Sensing field:	17.54	33	6	1	Yes	1.042~1.047	800
±0.3×±0.3	18.81	34	5	1	Yes	1.066~1.073	600
	18.13	31	5	1	Yes	1.057~1.061	800

We will consider the relation of average degree and the number of clusters. We used the simulated samples to describe Fig.22. Each blue dot is one sample in the relation of degree and clusters. The orange line is the trend that all blue dots generate. The trend is used the power function to simulate. And, it can capture the relationship through the orange line.

$$G(x) = 15.675935 * x^{-1.02} \tag{8}$$

In the self-coupled method, we hope that the topology only has one cluster. In these situations, it must synchronize after the calculation iteration. So, we discuss the suitable average degree by Equation (8). We set the cluster equal to one and the average degree is 14.85243. The best average degree is approximate 15 that all sensor nodes generate only one cluster.

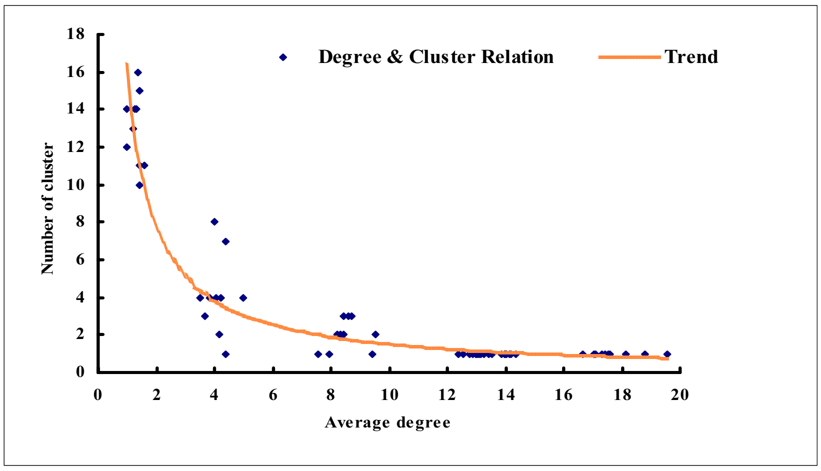

Fig. 22. Degree & cluster trend

4.4 The analysis of random participation

We consider if the self-coupled method reach time synchronization in some few nodes operating. We randomly selected one fifth nodes to operate the self-coupled meth-od. The one fifth nodes reference their neighbors to reach time synchronization and other nodes only reference the one fifth nodes in the sensing range. And, we discuss if it can reach time synchronization.

Table 8 to 10 represents the one fifth nodes randomly participated in one hundred sensor nodes in the protocol. Table 8, Table 9 and Table 10 are individually in the sensing range 0.1, 0.2 and 0.3. In Table 8, it is not synchronized because the node density is lower and the reference nodes are only twenty. Table 9 and Table 10 are the synchronized cases.

Table 8. Random participation (Sensing range = 0.1 in 100 nodes)

Influence / Environment	Average degree	Max. degree	Min. degree	Synchronized
Nodes: 100 Time range: 0~2 Sensing range 0.1 Sensing field: ±0.3×±0.3	8.94	16	3	No
	8.42	18	2	No
	8.42	15	2	No

Table 9. Random participation (Sensing range = 0.2 in 100 nodes)

Influence / Environment	Average degree	Max. degree	Min. degree	Synchronized	Converged clock speed	Calculation iteration (K=1000)
Nodes: 100 Time range: 0~2 Sensing range 0.2 Sensing field: ±0.3×±0.3	27.82	44	12	Yes	1.000	600
	28.02	43	14	Yes	1.199	700
	27.5	39	11	Yes	0.950	500

Table 10. Random participation (Sensing range = 0.3 in 100 nodes)

Influence / Environment	Average degree	Max. degree	Min. degree	Synchronized	Converged clock speed	Calculation iteration (K=1000)
Nodes: 100	47.4	78	20	Yes	1.100	400
Time range: 0~2 Sensing range 0.3	46.02	77	22	Yes	1.100	400
Sensing field: ±0.3×±0.3	47.4	76	18	Yes	1.250	200

Table 11 to Table 13 represents the one fifth nodes randomly participated in two hundred sensor nodes in the protocol. Table 11, Table 12 and Table 13 are individually in the sensing range 0.1, 0.2 and 0.3. Table 11 is still not synchronized. Table 12 and Table 13 converged earlier than Table 9 and Table 10. In the approach, we consider that the reference nodes are fewer and the converged rate is earlier. Because the different noise is fewer, all calculation iteration is fewer. And, it can decrease the message complexity.

Table 11. Random participation (Sensing range = 0.1 in 200 nodes)

Influence / Environment	Average degree	Max. degree	Min. degree	Synchronized
Nodes: 200	18.55	30	3	No
Time range: 0~2 Sensing range 0.1	18.07	31	4	No
Sensing field: ±0.3×±0.3	17.63	35	2	No

Table 12. Random participation (Sensing range = 0.2 in 200 nodes)

Influence / Environment	Average degree	Max. degree	Min. degree	Synchronized	Converged clock speed	Calculation iteration (K=1000)
Nodes: 200	55.81	80	19	Yes	0.875	300
Time range: 0~2 Sensing range 0.2	57.71	89	24	Yes	0.950	200
Sensing field: ±0.3×±0.3	54.46	84	22	Yes	0.825	300

Table 13. Random participation (Sensing range = 0.3 in 200 nodes)

Influence / Environment	Average degree	Max. degree	Min. degree	Synchronized	Converged clock speed	Calculation iteration (K=1000)
Nodes: 200	97.77	164	48	Yes	1.350	100
Time range: 0~2 Sensing range 0.3	98.63	164	40	Yes	0.975	100
Sensing field: ±0.3×±0.3	98.67	171	41	Yes	1.000	100

4.5 The error of self-coupled simulation

It usually considers the synchronization error in the time synchronization in different protocols. The error is the difference between the average time of all sensor nodes and the adjustment local clock of each sensor node.

Fig.23 is one case that all nodes are synchronized. C(0) means all nodes pass through 0 calculation iteration. It only has three values in C(0). Because we normalized all time clocks in 0, 1 and 2, its difference has three values in C(0). We simulated C(0) to C(500) and show the adjustment simulations per 100 calculation iteration. Each dot represents the error of each sensor node. All sensor nodes' error gradually decrease from C(0) to C(500). Although there are few nodes that the error is not approached to zero, they must decrease the error by the calculation iteration. Fig.24 is one case that all nodes are not synchronized. There are only some nodes that normally decrease the error. Other sensor nodes are not in the same cluster. They synchronize themselves so finally the error of all nodes cannot decrease to zero.

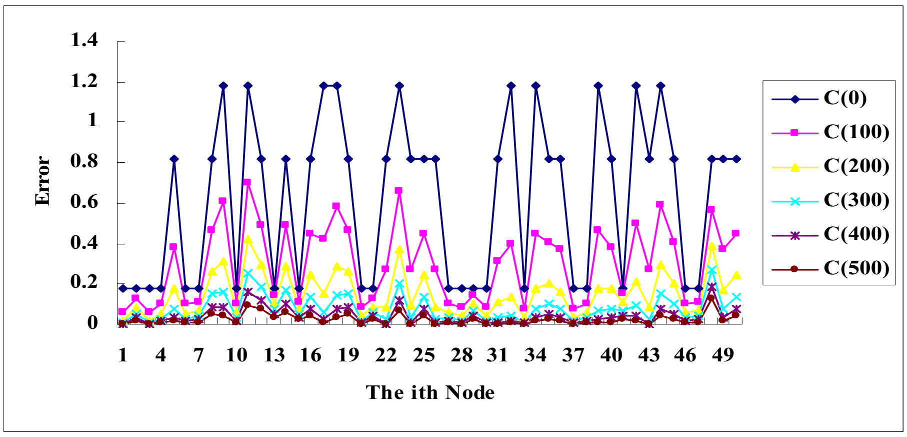

Fig. 23. Synchronized error

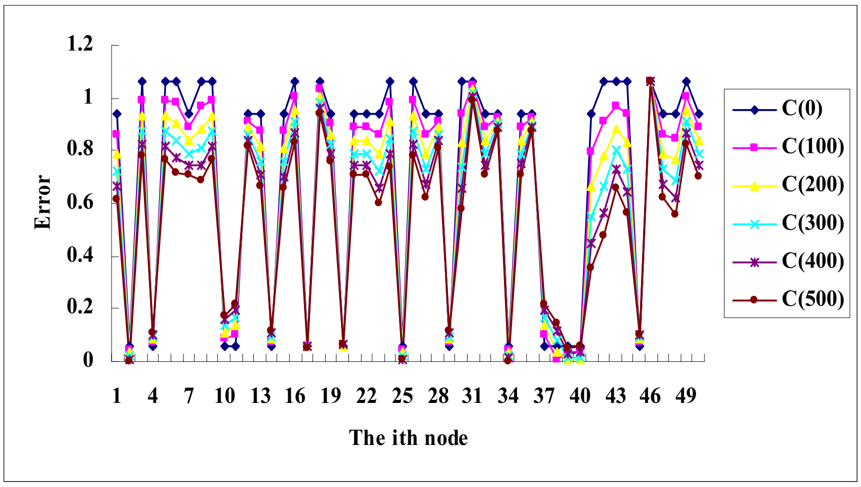

Fig. 24. Non-synchronized error

5 Conclusions and future work

There are many existing researches regarding the time synchronization in the Argo float sensors network. In this paper, we introduce two types of approaches, the hierarchical approaches and the self-coupled approaches. We focus our study in the performance of the self-coupled time synchronization protocol. Specifically, both the randomly and planned deployment methods with various degree values are considered. An enhancement based on random participation is also suggested to reduce the overhead. When compared with the hierarchical schemes, the self-coupled method eliminates the effort spent on tree construction. It also exhibits better scalability. In our simulation, we obtain some interesting observations. A theoretic curve is line-fitted to predict the number of virtual clusters given the average degree. In the random deployment method, the average degree not smaller than 15 is a sufficient condition for synchronization. When this condition is not met, there is no guarantee that the network can be synchronized at all because there will be isolated nodes. We also verify that the synchronization process accelerates as the degree increases. We conclude that self-coupled method is good candidate for the synchronization problem in the Argo float sensors network. It is most suitable for the Argo float sensors network of high density and wide deployment region.

References

[1] Tsai, C. S., Wang, Y. C., & Wang, H. K. (2015). A transmission control protocol with high throughput of using low earth-orbit satellite to collect data from the floats on sea surface. *Journal of Computers, 26*, 78-96.

[2] Lagerloef, G., Colomb, F. R., Le Vine, D., Wentz, F., Yueh, S., Ruf, C., Lilly, J., Gunn, J., Chao, Y., de Charon, A., Feldman, G., & Swift, C. (2008). The Aquarius/SAC-D mission: Designed to meet the salinity remote-sensing challenge. *Oceanography, 21*, 69-81.

[3] Yue, T. W., Wang, Y. C., & Yen, W. (2008, October). *Fast sensor identification technology for sea surface salinity measurement*. Paper presented at the Proceedings of the 2008 IEEE International Conference on Systems, Man, and Cybernetics, Singapore.

[4] Boutin, J., & Martin, N. (2006). ARGO upper salinity measurements: Perspectives for L-band radiometers calibration and retrieved sea surface salinity validation. *IEEE Geoscience and Remote Sensing Letters, 3*, 202-206.

[5] Chang, Y. S., Cheng, H. T., & Lai, H. J. (2009, October). *Metadata miner assisted integrated in-formation retrieval for Argo ocean data*. Paper presented at the Proceedings of IEEE International Conference on System, Man, Cybernetics, San Antonio, TX.

[6] Tsai, C. S., & Yang, G. F. (2008, October). Research on an assistant ad hoc network to aid the measurement of salinity-temperature-depth. Paper presented at the Proceedings of IEEE International Conference on System, Man, Cybernetics, Singapore.

[7] Sun, K., Ning, P., & Wang, C. (2006). Secure and resilient clock synchronization in wireless sensor networks. *IEEE Communications on Selected Areas, 24*, 395-408.

[8] Chen, T., Liu, J., Jing, G., Liu, Z., & Bin, Z. (2006, June). *Time synchronization in acoustic localization system based on wireless sensor network*. Paper presented at the Proceedings of the 6th World Congress on Intelligent Control and Automation, Dalian, China.

[9] Yang, Y., & Yang, K. (2006, August). *Time synchronization for wireless sensor networks using the principle of radar systems and UWB signals*. Paper presented at the Proceedings of the International Conference on Information Acquisition, Shandong, China.

[10] Deng, G. Y., & Zhang, F. (2006, November). *A power management for probabilistic clock synchronization in wireless sensor networks*. Paper presented at the Proceedings of the International Conference on Communication Technology, Guilin, China.

[11] Noh, K. L., Chaudhari, Q., Serpedin, E., & Suter, B. (2006, October). *Power-efficient clock synchronization using two-way timing message exchanges in wireless sensor networks*. Paper presented at the Proceedings of the Military Communications Conference, Buffalo, NY.

[12] Kwon, T. J., Gerla, M., Varma, V. K., Barton, M., & Hsing, T. R. (2003). Efficient flooding with passive clustering: An overhead-free selective forward mechanism for ad hoc/sensor networks. *Proceedings of the IEEE, 91*(8), 1210-1220.

[13] Xu, J., Cheng, J., Han, L., & Zhou, J. (2013, July). *Lifetime scaling law of ordinary clustering ultra-wideband sensor network*. Paper presented at the 9th International Wireless Communications and Mobile Computing Conference, Sardinia, Italy.

[14] Barbarossa, S., & Celano, F. (2005, June). *Self-organizing sensor networks designed as a population of mutually coupled oscillators*. Paper presented at the Proceedings of the 6th IEEE Workshop on Signal Processing Advances in Wireless Communications, New York, NY.

[15] Mirollo, R. E., & Strogatz, S. H. (1990). Synchronization of pulse-coupled biological oscillators. *SIAM Journal on Applied Mathematics, 50*, 1645-1662.

[16] Barbarossa, S. (2005, May). *Self-organizing sensor networks with information propagation based on mutual coupling of dynamic systems*. Paper presented at the Proceedings of Workshop on Wireless Ad hoc Networks, New York, NY.

[17] Su W., & Akyildiz, I. F. (2005). Time-diffusion synchronization protocol for wireless sensor networks. *IEEE/ACM Transactions on Networking, 13*, 384-397.

[18] Long, D. S., & Tao, X. (2006, May). *Cluster-based power efficient time synchronization in wireless sensor networks*. Paper presented at the Proceedings of the IEEE International Conference on Electro/information Technology, East Lansing, MN.

[19] Shin, K. Y., Lee, K. Y., & Lee, K. (2006, June). *CRIT: A hierarchical chained-ripple time synchronization in wireless sensor networks*. Paper presented at the Proceedings of the 2006 IEEE Inter-national Conference on Sensing and Control Networking, New York, NY.

[20] Kim, H., Kim, D., & Yoo, S. E. (2006, April). *Cluster-based hierarchical time synchronization for multi-hop wireless sensor networks*. Paper presented at the Proceedings of the 20th International Conference on Advanced Information Networking and Applications, Vienna, Austria.

[21] Rout, R. R. & Ghosh, S. K. (2012). Enhancement of lifetime using duty cycle and network coding in wireless sensor networks. *IEEE Transactions on Wireless Communications, 12*, 656-667.

[22] Padmavati, & Aseri, T. C. (2014, March). *Comparison of routing protocols in wireless sensor net-work using mobile sink: A survey*. Paper presented at the Proceedings of the engineering and Computation-al Sciences, Venice, Italy.

[23] Mamun-Or-Rashid, M., Hong, C. S., & In, C. H. (2005, July). *Passive cluster based clock synchronization in sensor network*. Paper presented at the Proceedings of the Advanced Industrial Conference on Telecommunications/Service Assurance with Partial and Intermittent Resources Conference/ E-Learning on Telecommunications Workshop, Lisbon, Portugal.

[24] Ganeriwal, S., Kumar, R., & Srivastava, M. B. (2003, November). *Timing-sync protocol for sensor networks*. Paper presented at the Proceedings of the 1st international conference on Embedded net-worked sensor systems, Baltimore, MD.

[25] Giles, R. K., & Spencer, B. F. (2007, March). *Hierarchical PSD damage detection methods for smart sensor networks*. Paper presented at the Proceedings of the World Forum on Smart Materials and Smart Structures Technology, San Diego, CA.

[26] Bui, M., Myoupo, J. F., & Sow, I. (2007, February). *An operational model for mobile target tracking in wireless sensor networks*. Paper presented at the Proceedings of the 2nd International Symposium on Wireless Pervasive Computing, San Juan, PR.

[27] Gu, H., Wang, K., Wang, H., Zhang, J., & Wang, C. S. (2007, March). *Routing in hexagonal wireless sensor networks*. Paper presented at the Proceedings of the International Conference on Wireless and Optical Communications Networks, Pisa, Italy.

[28] Cormen, T. H., Leiserson, C. E., Rivest, R. L., & Stein, C. (2001). *Introduction to algorithms*. Cambridge, MA: The MIT Press.