International Journal of Recent Technology and Engineering (IJRTE)
ISSN: 2277-3878 (Online), Volume-7 Issue-4s2, December 2019

Cluster based Time Synchronization Algorithm for Mobile Underwater Sensor Networks (UWSN)

L. Sivagami, J. Martin Leo Manickam

Abstract: ***Underwater Sensor Network (UWSN) performs the functionality of data transmission through wireless mobile nodes in the aquatic environment. But, due to the harsh and uneven nature of the aquatic environment, the operation of the UWSN is effected badly as no time synchronization between nodes, In this paper, we propose to develop a Cluster based Time Synchronization Algorithm for Mobile UWSN. This algorithm aims at synchronizing the time between the network nodes by performing inter cluster as well as intra cluster synchronization. In this way, once the nodes are time synchronized, the network operation will be carried out smoothly, in turn consuming lesser energy to perform the network functionality.***

Keywords: UWSN, Cluster, Synchronization, Mobile, Underwater

I. INTRODUCTION

Under Water Sensor Networks (UWSN)

Underwater Wireless Sensor Network (UWSN) is an emerging technology in the Wireless Sensor Network (WSN) that has been designed for the underwater environment. The UWSN faces several issues like restricted bandwidth, increased propagation delay, irregular node mobility, etc. These unique characteristics of the UWSN need to be handled appropriately in each layer of the network protocol. Because of the distinct features of the UWSN, maintaining clock synchronization in the network has turned into a serious issue. To carry out the network functioning such as time stamping of all the events, data aggregation in distributed manner, localization, MAC, etc in the correct manner, it is necessary to maintain time in the same proportion. There are several applications in which UWSN are applied. They are military operations, disaster avoidance, data gathering, pollution level examining, offshore exploring, etc [1]. UWSN applications like Seismic level examining has been extensively employed in the underwater environment. Extraction of oil from the underwater area and ocean wave monitoring is also performed through the UWSN and supports well in determining any changes in the underwater environment which may lead to earthquake or tsunami. UWSN is also employed in examining of various equipment as well as handling its operations which can be either long term or short term based. When UWSN is employed in short term basis, only the first few operations needs to be examined once the equipment is deployed. When the UWSN is employed in long term basis as in seismic monitoring, there is a need to ensure flawless communication in wireless mode, self configuring into multihop network, synchronization of time, energy proficiency, localization, etc. Some of the other applications of UWSN are chemical leak sensing and controlling, sensing biological occurrence like leak of oil in ocean, underwater data gathering, phytoplankton concentration analyzing, etc [2]. The distinct features of UWSN are lesser bandwidth, longer propagation delay, increased error probability, mobile sensor node, etc. These factors give rise to many issues in the smooth operation of the UWSN in each protocol stack layer. Also, power requirement of the UWSN is high for the transmission of the underwater data, since it consume high energy in data transmission. Due to these characteristics of the UWSN, the designing of the time synchronization algorithm has become very complication [3].

Time Synchronization Algorithms for UWSN

Most of the applications in UWSN are based on time synchronization operation. For instance, the data mining operation in UWSN needs synchronization of nodes since it uses TDMA which provides global time information and utilized in the Medium Access Control (MAC) protocols. Also, the localization algorithm in UWSN directly assumes that all the operations will be time synchronized [4]. In most of the distributed applications, synchronization of time is an important criteria. In the conventional networking and operating systems, the database queries as well as security concepts consider time synchronization as critical. NTP is the canonical technique in internet to maintain distributed time synchronization. In sensor networks, in which applications include acoustic beamforming, target tracking, etc, synchronization of time is very important since there is need to combine various time sensitive data and then process. Also in sensor networks, the energy usage during synchronization process needs to be maintained low [5].

In wireless sensor network, time synchronization is very critical in order to perform all its network operations. When time is synchronized, the involved data will be processed and analyzed appropriately and will be capable of predicting the further system operation correctly.

Manuscript published on 30 December 2019.
*****Correspondence Author(s)
L. Sivagami, Research Scholar, Sriram Engineering College, Anna University, Chennai India.
J. Martin Leo Manickam, St. Joseph's College of Engineering, Chennai India.

For instance, in the mobile target tracking operation, it is necessary for the distributed system to match the sensing time as well as the sensing location in order to estimate the direction as well as the speed of the mobile object.

Also, time synchronization helps the processing in application layer, medium access control layer and then the networking layer. For instance, time synchronization is required in the Slotted FAMA MAC protocol [6].

In UWSN, several types of time synchronization algorithms are developed such as TSHL, ML-Sync, D-Sync, etc. All of these proposed protocols try to overcome the issues like higher propagation delay. But, still several disadvantages are seen. For instance, TSHL has been developed keeping static network in mind and hence node mobility is not taken into consideration. The MU-Sync considers the mobility based issues. But it is not effective in terms of energy utilization. The D-Sync algorithm does not consider the skew property while determining the Doppler Shift [7]. However, special deployment is needed and may also depend on the velocity preciseness. But, measuring exact velocity and deploying perfectly is very hard to assure in the UWSN [8].

II. RELATED WORKS

Jun Liu et al [4] have presented a pairwise, cross-layer, time synchronization mechanism for mobile underwater sensor networks, called TSMU. The proposed mechanism enhances the propagation delay determination by using the Doppler effect and is assisted by the Kalman filter usage.

Zhengbao Li et al [6] have proposed an energy efficiency distributed time synchronization algorithm (called ''E^{2}DTS'') for underwater acoustic node mobility networks. The clock skew as well as the offset value are calculated in E^{2}DTS. The relation between the unpredictable propagation delay and the mobility of the node is analyzed. Next the clock skew is computed accordingly. Then the skew corrected nodes transmits its local timestamp towards the beacon inorder to calculate the clock offset.

Jun Liu et al [7] have presented Mobi-Sync, a new time synchronization mechanism for mobile underwater sensor networks. Mobi-Sync differentiates it from the conventional techniques related to the terrestrial WSN by taking into account the spatial correlation between the mobility format of the surrouding UWSN nodes. This makes the Mobi-Sync mechanism capable of precisely calculating the high dynamic propagation delay.

Ying Guo et al [8] have presented a Mobile Counteracted Time Synchronization technique known as "Mc-Sync". Mc-Sync is a new time synchronization mechanism for mobile underwater acoustic sensor networks. In this technique, two mobile nodes are utilized as reference nodes to control the mobility of the sensor nodes. The trajectories of the selected mobile reference node is examined accurately and then developed according to the underwater environment.

Oriol Pallares et al [9] have analyzed the time synchronization issues in the UWSN deployed in shallow water environment, by considering most of the communication issues faced in the aquatic environment and then monitoring its response in the simulation as well as real time test environment. Then a hybrid frame is presented which is developed on the basis of the time synchronization utilizing the LFM as well as OFDM communication along with channel impulse response equalization.

Kulurkar et al [10] have presenteda Time-Synchronization underwater sensor-network. In the proposed synchronization technique, energy usage control is a serious factor considered while developing the network. The node lifetime is dependent on the battery power, hence the usage of battery power needs to be controlled effectively. Many works have developed algorithm to predict the energy usage through simulation, code analysis and also through energy models. But, in most of the applications, the network operation is controlled by the sensor functioning or the other external conditions.

Jinwang Yi et al [11] have defined situations in which the signals from numerous beacons are highly lagging in time. Next, the technique also determined the factor like signal arrival as a critical factor which degrade the TDoA performance in situations when the transmissions are inconcurrent. To solve this issue, a technique for tracking the submersible is proposed known as Time of Arrival based Tracked Synchronization (ToA-TS). This ToA-TS technique enhances the localization feature in GPS for the situations like beacon inconcurrent transmissions and also when two sided communication is incapable in submersibles.

III. CLUSTER BASED TIME SYNCHRONIZATION ALGORITHM FOR MUWSN

Overview

The basic idea of time synchronization is that the unsynchronized nodes exchange messages with the synchronized reference node in order to estimate clock skews and offsets in local clocks. In this paper we propose a Cluster based Synchronization algorithm for UWSN. Initially, clusters are formed and cluster heads are elected as per the energy efficient hierarchical clustering algorithm [12]. Then we apply the inter-cluster and intra-cluster synchronization phases of [1] without considering the security features:

In the proposed algorithm, UWSN is made up of three kind of network nodes: Anchors, cluster heads (CH) and cluster members (CM). Anchors have unlimited energy resources and perfect timing information. They communicate with CHs and CMs through acoustic links.

Figure 1 shows the block diagram of proposed algorithm.

Published By:
Blue Eyes Intelligence Engineering
and Sciences Publication (BEIESP)

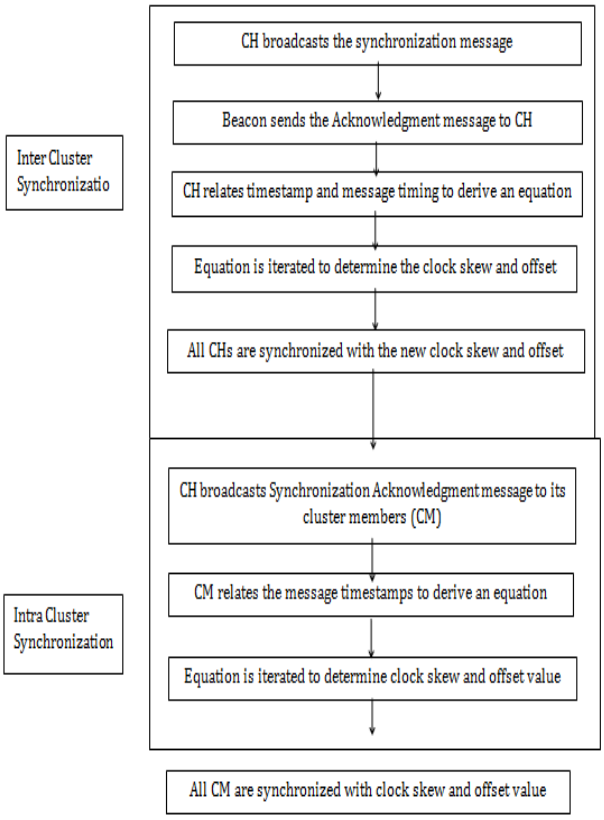

Fig. 1: Block Diagram

Cluster Formation

The steps involved in the cluster formation are as follows:

1. When the sensor nodes (N_i) are deployed in the network, each sensor transmits an advertisement message (AD_mes) to the nodes within its transmission range to declare itself as the volunteer cluster head (VCH) with probability z.
   Note: The probability z to choose the cluster head is estimated based on the energy level of the node. i.e. the node with maximum energy is declared as CH.
2. Each N_i after receiving AD_mes, forwards it to the nodes which are n-hop away from CH.
3. In case, N_i receives AD_mes and it is not a VCH, then it joins the cluster of nearest CH by sending back a Join_mes to the node which sent the AD_mes.
4. When N_i is neither a CH nor it as joined a cluster, it becomes a CH by itself. i.e it becomes a forced CH (FCH). Also, if any N_i does not receive the AD_mes within time t, it decides that it is not within the n-hops of any VCH and declares itself as FCH.
5. In addition, as all the sensor nodes within the cluster are at n-hop away from CH, CH can transmit the collected data to processing centre after every time t.

Figure 2 represents the formed clusters with elected cluster heads.

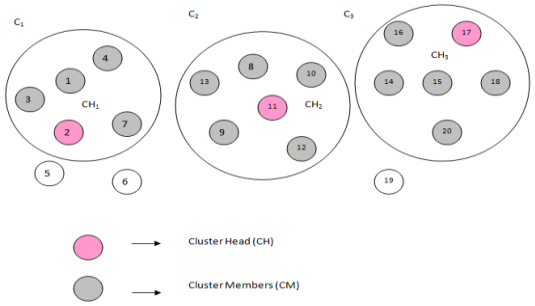

Figure 2: Clusters with elected CH

3.3. Inter Cluster Synchronization Phase

The whole network consists of three types of nodes: anchors, cluster heads and cluster members (CM). The anchors are considered to have unlimited energy resources and perfect timing details and link with CHs and CMs. Each cluster has only CH and numerous CMs which are directly linked to the CH.

During the inter-cluster synchronization phase, CHs synchronize themselves with anchors in sender-receiver mode. The CH communicates with the anchor through multiple hops to achieve synchronization. Based on the response received from the anchor and by iterating the process, the clock skew and offset value is determined. This process is described in algorithm 1.

Algorithm 1

Published By:
Blue Eyes Intelligence Engineering
and Sciences Publication (BEIESP)

Notations	Meaning
CH_i	Cluster Head i
S_{ID}	Source Identity
D_{ID}	Destination Identity
ID	Identifier
HC	Hop Count
k	Iteration count
Sync_Req	Synchronization Request packet
M_1	Synchronization Request Message
$K_{Pri(CH)}$	Private Key of cluster head
H	Hash function
I_{CHi}	Identifier of CH
t_1	M_1 transmission time
x,y,z	3 dimension co-ordinates
D_{CH_B}	Distance between Cluster head and beacon
CH_x, CH_y, CH_z	x,y,z co-ordinates of Cluster Head
B	Anchor node
B_x, B_y, B_z	x,y,z co-ordinates of Beacon
CM	Cluster Member
t_2	message M_1 reception time
$T_{CHi}(t_1)$	Timestamp in CH for t_1
X_{CH_B}	random delay in uplink
Y_{CH_B}	random delay in downlink
Sync_Ack	Synchronization Acknowledgment
α_{CHi}	clock skew of cluster head
β_{CHi}	initial offset of clock in cluster head
M_2	Synchronization Acknowledgment Message
I_B	Identifier of Beacon
t_3	M_2 transmission time
$T_{CHi}(t_4)$	Timestamp of CH at time t_4
ϵ	time interval between t_2 and t_3
Sync_msg	Synchronization Message

1. The CH_i creates the Sync_Req packet using the node details like S_{ID}, D_{ID}, ID, HC and k.

$$Sync\_Req = < S_{ID}, D_{ID}, ID, HC, k > (1)$$

2. Then the CH_i generates M_1, containing the following fields

$$M_1 = \{ I_{CHi}, T_{CHi}(t_1), Sync\_Req \} \quad (2)$$

3. Then CH_i estimates D_{CH_B} according to equation (3) considering the network with respect to 3-Dimension co-ordinates (x,y,z):

$$D_{CH\_B} = [(CH_x - B_x)^2 + (CH_y - B_y)^2 + (CH_z - B_z)^2]^{1/2} \quad (3)$$

4. Next CH_i broadcasts M_1 to B and its CM.

$$CH_i \longrightarrow B, CM$$

5. On receiving M_1, B determines the message reception time, t_2 according to equation (4) given below:

$$t_2 = t_1 + D_{CH\_B} + X_{CB} \quad (4)$$

6. B generates Sync_Ack packet.

$$Sync\_Ack = < S_{ID}, D_{ID}, ID, HC, k > (5)$$

7. The time stamp is estimated according to equation (6) given below:

$$T_{CHi}(t_1) = \alpha_{CHi} [t_2 - D_{CH\_B} - X_{CH\_B}] + \beta_{CHi} \quad (6)$$

8. Next B generates M_2 containing the following fields

$$M_2 = \{ I_B, T_{CHi}(t_1), t_2, t_3, Sync\_Ack \} (7)$$

9. Then B sends M_2 to CH_i at t_3.

$$CH_i \xleftarrow{M_2} B$$

10. On receiving M_2 from B, CH_i records its arrival time, according to equation (8) given below:

$$T_{CHi}(t_4) = \alpha_{CHi} [t_3 - D_{CH\_B} - Y_{CH\_B}] + \beta_{CHi} \quad (8)$$

11. $T_{CHi}(t_1)$, t_2 and t_3 are equated as shown in equation (9) shown below:

$$Q_{CH} = 1/2 [(T_{CHi}(t_1) - t_2) + (T_{CHi}(t_4) - t_3)] (9)$$

12. Then applying the equation (6) and (9) and also given the relation between t_2 and t_3, the Q_{CH} can be rewritten as given in equation (10):

$$Q_{CH} = \alpha_{CHi}[t_2 + \epsilon/2 + (Y_{CH\_B} - X_{CH\_B})/2] + \beta_{CHi} + (Y_{CH\_B} - X_{CH\_B})/2 (10)$$

$$\text{Where } t_3 = t_2 + \epsilon \quad (11)$$

13. Since $t_2 >> \epsilon/2 + (Y_{CH\_B} - X_{CH\_B})/2$, the equation (10) is rewritten as in equation (12) given below:

$$Q_{CH} = \alpha_{CHi} . t_2 + \beta_{CHi} + \delta_{CHi} \quad (12)$$

$$\text{Where } \delta_{CHi} = (Y_{CH\_B} - X_{CH\_B})/2 \quad (13)$$

14. On performing k iterations of Q_{CH} as in equation (12), the α_{CHi} and β_{CHi} values can be estimated.

15. Once α_{CHi} and β_{CHi} values are determined, B broadcasts the Sync_msg to the surrounding CHs.

16. On receiving the Sync_msg, the remaining CH consider CH_i as its reference node and synchronise its α_{CHx} and β_{CHx} with respect to α_{CHi} and β_{CHi}.

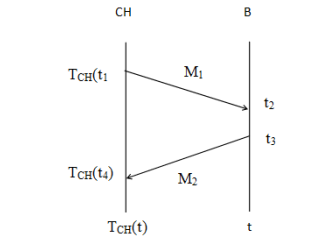

Figure 3: Inter Cluster Synchronization

In this way, all the cluster heads are synchronized by estimating the clock skew and offset of one cluster head and then making the remaining cluster heads to adapt accordingly. Thus, inter cluster synchronization will ensure that all the Cluster heads are orderly synchronized. Figure 3 shows the inter-cluster synchronization process.

Intra-Cluster Synchronization Phase

During the intra-cluster synchronization phase, cluster members synchronize themselves with CHs in receiver-receiver mode. All the members of a cluster are at a distance of one hop from its cluster head. The cluster members receive the details of cluster head and then synchronize its clock skew and offset accordingly. This process is described in algorithm 2.

Algorithm 2

Published By:
Blue Eyes Intelligence Engineering
and Sciences Publication (BEIESP)

Notations	Meaning
CH	Cluster Head
CM	Cluster Member
B	Anchor
Sync_Ack	Synchronization Acknowledgment
S_{ID}	Source Identity
D_{ID}	Destination Identity
ID	Identifier
HC	Hop Count
K	Iterations
M_3	Synchronization Acknowledgment Message
I_{CHi}	Identifier of CH
$T_{CHi}(t_5)$	Timestamp
t_5	Message transmission time
β_{CHi}	initial offset of clock in CH
α_{CHi}	clock skew value of CH
CH_x, CH_y, CH_z	x,y,z co-ordinates of CH
CM_x, CM_y, CM_z	x,y,z co-ordinates of CM
t_6	Message reception/arrival time
$T_{CMi}(t_6)$	Timestamp of cluster member
α_{CMi}	clock skew of CM
β_{CMi}	initial offset of clock in CM
X_{CH_CM}	random delay in uplink

1. The CH_i generates the Sync_Ack packet containing the required information as shown below:

$$Sync\_Ack = < S_{ID}, D_{ID}, ID, HC, k > \quad (14)$$

2. Then CH_i generates the M_3 as given below:

$$M_3 = \{ I_{CHi}, T_{CHi}(t_5), Sync\_Ack \} \quad (15)$$

3. The CH_i broadcasts the M_3 to all its CM.

$$CH_i \longrightarrow M_3 \longrightarrow CM$$

4. CM estimates its estimated time, t_5 according to equation (16) given below:

$$t_5 = [T_{CHi}(t_5) - \beta_{CHi}]/ [1 + \alpha_{CHi}] \quad (16)$$

5. Next CM_j estimates D_{CH_CM} according to equation (17) shown below:

$$D_{CH\_CM} = [(CH_x - CM_x)^2 + (CH_y - CM_y)^2 + (CH_z - CM_z)^2]^{1/2} \quad (17)$$

6. timestamp is estimated according to equation (18) given below:

$$t_6 = t_5 + D_{CH\_CM} + X_{CH\_CM} \quad (18)$$

7. Next the CM_j estimates the $T_{CMi}(t_6)$ according to equation (19) given below:

$$T_{CMi}(t_6) = \alpha_{CMi} .t_6 + \beta_{CMi} \quad (19)$$

8. Applying equation (18) and then (16) in (19) provides the $T_{CMi}(t_6)$ value as shown in equation (20):

$$T_{CMi}(t_6) = \alpha_{CMi} .[\{[T_{CHi}(t_5) - \beta_{CHi}]/ [1 + \alpha_{CHi}]\} + D_{CH\_CM} + X_{CH\_CM}] + \beta_{CMi} \quad (20)$$

9. By k iterations of equation (20), the α_{CMi} and β_{CMi} value can be determined for the CM in the given cluster.

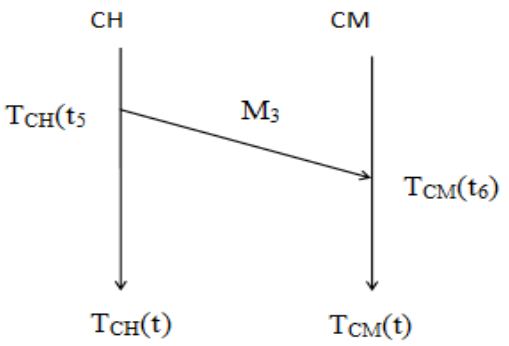

Figure 4: Intra Cluster Synchronization

In this way, all the cluster members of every cluster are synchronized by the cluster head. This process is followed within every cluster in the UWSN. Figure 4 shows the intra-cluster synchronization process.

IV. EXPERIMENTAL RESULTS

SIMULATION PARAMETERS

The proposed Cluster based Time Synchronization Algorithm (CTSA) is implemented using AquaSim module of NS2 [13]. The proposed CTSA is compared with the E^2DTS algorithm [6] and the performance is evaluated in terms of offset error, accuracy, number of messages and average energy consumption.

The simulation settings and parameters are summarized in table 1.

Table 1: Simulation Parameters

Number of Nodes	50,75,100,125 and 150
Area size	1000 X 1000m
MAC	Underwater sensor mac
Simulation Time	500 sec
Time after Sync	100,150,200,250,300 and 350
Transmission Power	4.0
Receiving Power	0.75
Initial Energy	10000 Joules

Results & Analysis

Varying the number of Nodes

In this experiment, the results are measured against number of nodes.

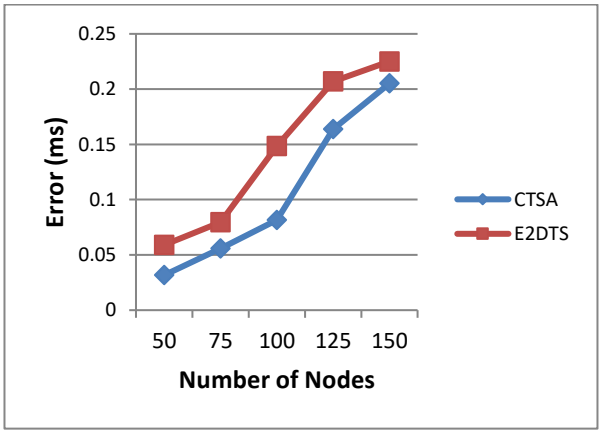

Figure 5: Clock Error for varying the nodes

Figure 5 shows the clock error measured for CTSA and E2DTS algorithms when the nodes are varied. As we can see from the figure, the error of both the algorithms linearly increase, as the number of nodes are increased. However, CTSA has 30% less error when compared to E2DTS algorithm.

Published By:
Blue Eyes Intelligence Engineering
and Sciences Publication (BEIESP)

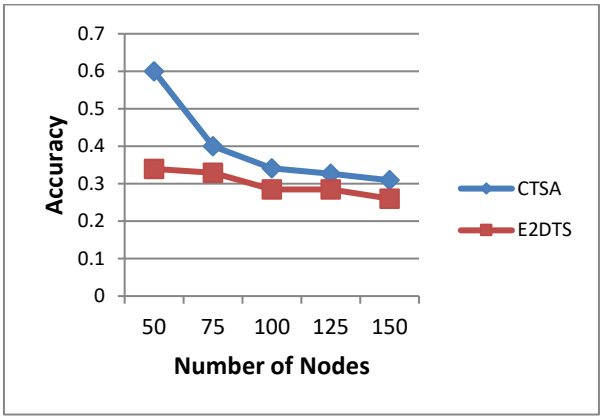

Figure 6: Accuracy for varying the nodes

Figure 6 shows the accuracy measured for CTSA and E2DTS when the nodes are varied. As we can see from the figure, the accuracy of both the algorithms decreases as the clock error increases. But accuracy of CTSA is 21% of higher when compared to E2DTS.

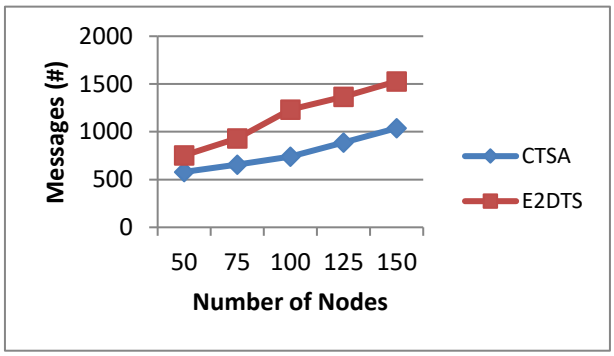

Figure 7: Number of messages for varying the nodes

Figure 7 shows the number of messages measured for CTSA and E2DTS when the nodes are varied. As we can see from the figure, the number of messages increases for both the algorithms when the nodes are increased. However CTSA has 31% less number of messages when compared to E2DTS.

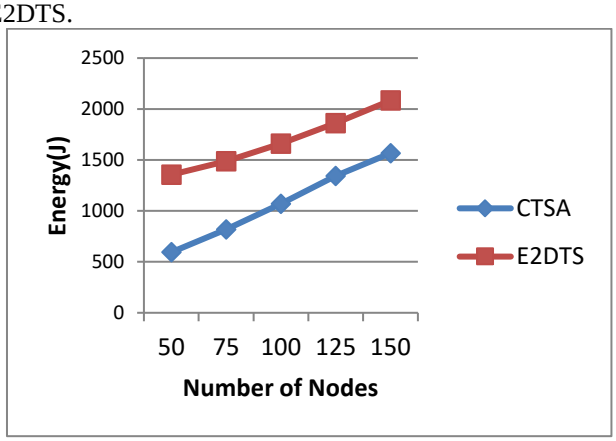

Figure 8: Energy Consumption for varying the nodes

Figure 8 shows the average energy consumption measured for CTSA and E2DTS when the nodes are varied. As we can see from the figure, the energy consumption of both the algorithm increases as the number of nodes is increased. Hence the energy consumption of CTSA is 38% less when compared to E2DTS.

Varying the Time after Synchronization

In the second experiment, the results are measured against time elapsed after recent synchronization.

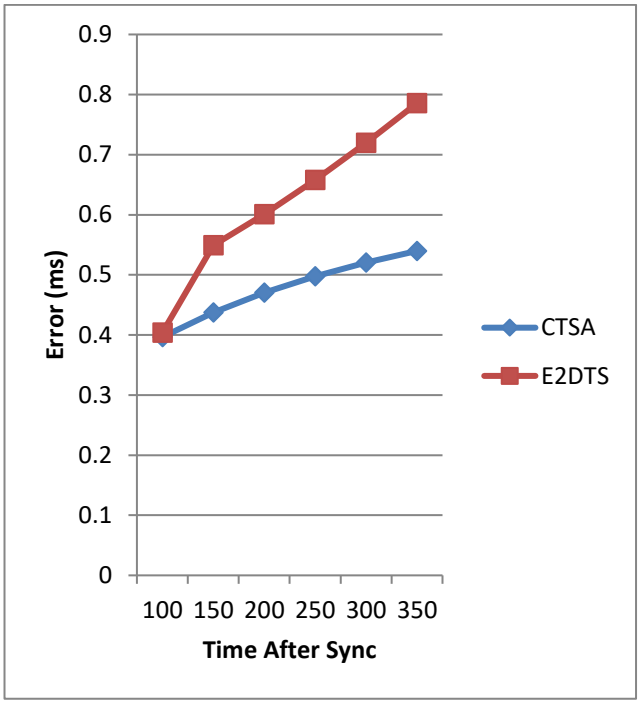

Figure 9: Clock Error for varying Time after Sync

Figure 9 shows the clock error measured for CTSA and E2DTS algorithms when the time after synchronization is increased. As we can see from the figure, the error of both the algorithms linearly increase, as the time is increased. However, CTSA has 21% less error when compared to E2DTS algorithm.

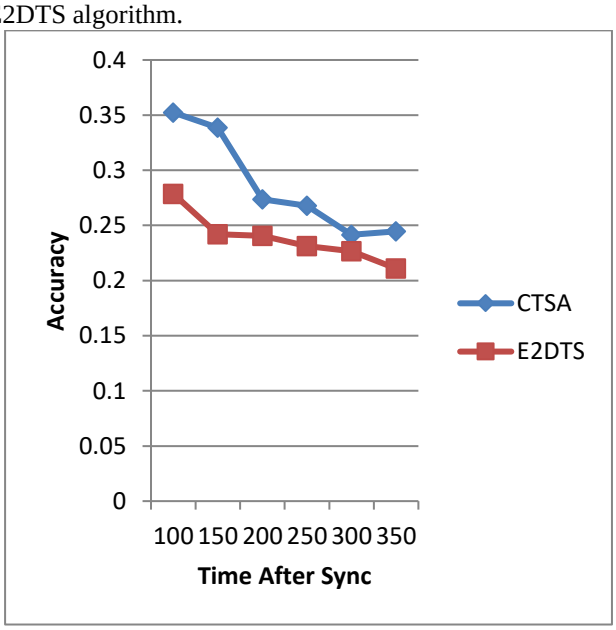

Figure 10: Accuracy for varying Time after Sync

Figure 10 shows the accuracy measured for CTSA and E2DTS when the time after synchronization is increased. As we can see from the figure, the accuracy of both the algorithms decreases as the clock error increases. But accuracy of CTSA is 16% higher when compared to E2DTS.

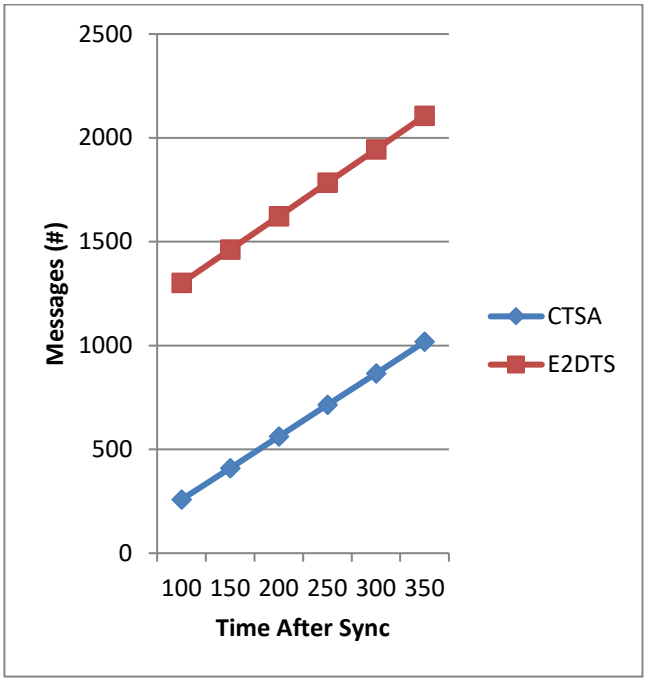

Figure 11: Number of messages for varying Time after Sync

Figure 11 shows the number of messages measured for CTSA and E2DTS when the time after synchronization is increased. As we can see from the figure, the number of messages increases for both the algorithms when the time is increased. However CTSA has 64% less number of messages when compared to E2DTS.

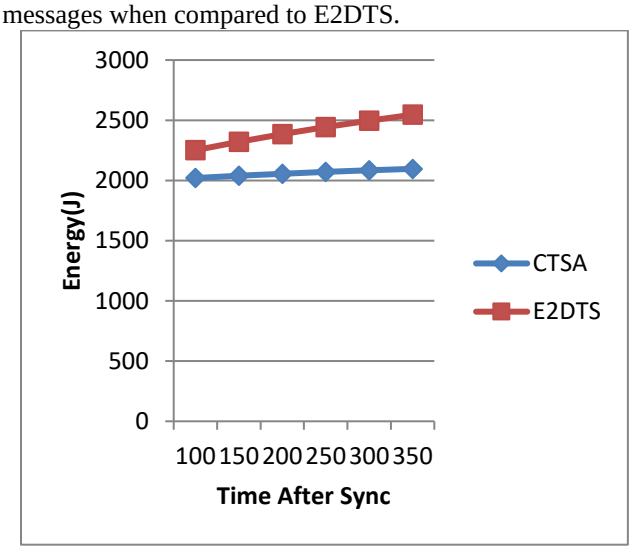

Figure 12: Energy Consumption for varying Time after Sync

Figure 12 shows the energy consumption measured for CTSA and E2DTS when the time after synchronization is varied. As we can see from the figure, the energy consumption of both the algorithm increases as the time is increased. Hence the energy consumption of CTSA is 14% less when compared to E2DTS.

V. CONCLUSION

In this paper, we have proposed a Cluster based Time Synchronization Algorithm (CTSA) for Mobile UWSN. In the proposed algorithm, synchronization is performed in two phases: Inter Cluster Synchronization and Intra Cluster Synchronization. In the inter cluster synchronization phase, CH broadcasts its synchronization request message to the anchor. The anchor responds by sending the synchronization acknowledgment message to the CH. The CH relates the timestamps and timings associated with the message, and derives an equation. On several iterations of the derived equation, the clock skew and offset value is determined. Then all the remaining CHs are synchronized with the determined clock skew and offset values. In the Intra cluster synchronization phase, the CH transmits the synchronization acknowledgment message to all its CMs. The CMs relate the timestamps and derive an equation, which on several iterations determines the clock skew and offset value. Then all the CMs synchronize themselves with the estimated clock skew and offset values. By simulation results, it is shown that CTSA attains reduced clock error and number of messages when compared to the existing algorithm.

REFERENCES

1. Ming Xu, Guangzhong Liu,Daqi Zhu and Huafeng Wu,(2014), *A Cluster-Based Secure Synchronization Protocol for Underwater Wireless Sensor Networks*,Hindawi Publishing Corporation, International Journal of Distributed Sensor Networks,Volume 2014, Article ID 398610,pp:1-13.
2. Anil C.B., Vani Sreekumar and Manjari Joshi,(2014), *A SURVEY ON VARIOUS TIME SYNCHRONIZATION TECHNIQUES IN UNDERWATER SENSOR NETWORKS*,INTERNATIONAL JOURNAL OF COMPUTER ENGINEERING & TECHNOLOGY (IJCET),Volume 5, Issue 12, pp. 116-122.
3. Vidya.R, Ambika.G.N, P Haripriya and Poojashree N.S, (2016), *Innovative Routing and Time Synchronization in Underwater Sensor networks*, International Journal of Scientific & Engineering Research, Volume 7, Issue 1.
4. Jun Liu, Zhaohui Wang, Zheng Peng, Michael Zuba, Jun-Hong Cui and Shengli Zhou,(2011), *TSMU: A Time Synchronization Scheme for Mobile Underwater Sensor Networks*, IEEE Globecom.
5. Affan A. Syed and John Heidemann,(2006), *Time Synchronization for High Latency Acoustic Networks*, IEEE,Infocom.
6. Zhengbao Li, Zhongwen Guo, Feng Hong and Lu Hong,(2013), *E2DTS: An energy efficiency distributed time synchronization algorithm for underwater acoustic mobile sensor networks*, Ad Hoc Networks,Vol-11,pp:1372–1380.
7. Jun Liu,Zhong Zhou,Zheng Peng,Jun-Hong Cui and Lance Fiondella,(2013), *Mobi-Sync: Efficient Time Synchronization for Mobile Underwater Sensor Networks*, IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS, VOL. 24, NO. 2.
8. Ying Guo and Yutao Liu,(2013), *Time Synchronization for Mobile Underwater Sensor Networks*,JOURNAL OF NETWORKS, VOL. 8, NO. 1.
9. Oriol Pallares, Pierre-Jean Bouvet and Joaquin del Rio,(2015), *Hybrid time synchronization for Underwater Sensor Networks*,ACTA IMEKO,Volume 4, Number 3,pp:30 - 35.
10. P.G.Kulurkar and Yogadhar Pandey (2013), *TIME-SYNCHRONIZATION IN UNDERWATER SENSOR NETWORKS*, International Journal of Computer Technology and Electronics Communication,Vol-2,Issue-1.
11. Jinwang Yi,Diba Mirza,Curt Schurgers and Ryan Kastner (2013), *Joint Time Synchronization and Tracking for Mobile Underwater Systems*,ACM,WUWNet.
12. L. Sivagami and J. Martin Leo Manickam,(2016), *Cluster-Based MAC Protocol for Collision Avoidance and TDMA Scheduling in Underwater Wireless Sensor Networks*, The Computer Journal Advance Access.
13. Network Simulator: http:///www.isi.edu/nsnam/ns

Published By:
Blue Eyes Intelligence Engineering
and Sciences Publication (BEIESP)