*Research Article*

# A Novel Loop Based Fine Grained Network-wide Time Synchronization over Constrained Delay Paths in Wireless Sensor Networks

**[1]K L V Sai Prakash Sakuru, [2]N Bheema Rao**

[1]Department of Electronics and Communication Engineering, National Institute of Technology, Warangal, Telangana, India
Email - sai@nitw.ac.in
[2]Department of Electronics and Communication Engineering, National Institute of Technology, Warangal, Telangana, India
Email - nbr@nitw.ac.in

**Article History**: Received:11 January 2021; Accepted: 27 February 2021; Published online: 5 April 2021

**Abstract-** Sensor Networks play a vital role in today's ever-growing automation process. Time synchronization between the nodes impacts the performance of the network and applications. Most of the existing time synchronization algorithms either run a tree algorithm or cluster-based algorithm and then perform time synchronization. Both the algorithms suffer from communication and computation efficiency. When a node fails due to an energy shortage, these algorithms need to rerun the setup, thereby the remaining nodes consume energy. Hence, we propose a novel loop-based fine-grained network-wide time synchronization over constrained delay paths for the sensor networks. When a packet is transmitted, the neighborhood node on receiving this packet will add a time offset information and broadcast. The process continues until the loop is formed and then uses the information set and helps to synchronize its clock information. The simulations and results validate the performance of the proposed algorithm.

**Keywords – wireless sensor networks, time synchronization, loop based, network-wide, estimation error**

## I. INTRODUCTION

Distributed sensor networks play an essential role in many applications, such as agriculture, border security, fire control, hospitals, smart cities, transport, etc. Data fusion, coordinated data transmission, coordinated on-off node control needs tight time synchronization [1], [2]. The sensor nodes are of small form factor, consisting of a communication module, computation module, and a power supply. Low-cost oscillators provide the clock on the node to keep the cost of the node minimal. These oscillators lose time, i.e., offset and skew concerning their neighbors. To make a worthy interpretation of the small form sensor node's data, the physical time on the sensor nodes plays a crucial role. As different oscillators are equipped in other nodes, no two node clocks show the same time in a distributed environment. On the other hand, time synchronization algorithms provide a cost-effective way to place all the network nodes on the same timescale concerning a global clock.

The most prevalent approach to network-wide synchronization is to form a spanning tree network or divide the network into clusters of single-hop and run a tree algorithm among the cluster heads. The former suffers from the error trying to propagate along the path and suffer from the network's higher computation and communication complexity and node failure. While the latter, the cluster-based time synchronization protocols, suffer from communication complexity in electing cluster heads, maintaining the communication paths, and node failure, thereby increasing energy consumption.

*Given the above drawbacks, we have developed a novel loop-based fine-grained network-wide time synchronization algorithm over constrained delay paths in sensor networks that address the intermediate node failure and eliminates the requirement of re-establish cluster or tree formation.*

The remaining paper is written as follows: Section II, a brief survey of the related work presented, and an iterative node-pair clock synchronization algorithm is explained in section III. The Loop concept is elaborated in section IV. Problem formulation regarding offset and skew is described in section V. Section VI presents the proposed algorithm. The simulation and results are shown in section VII and finally concludes with our work's future direction in section VIII.

## II. RELATED WORK

Time Synchronization in Wireless Sensor Network (WSN) is one of the interesting areas of research. Almost all the algorithms aim to synchronize the nodes, focusing mainly on time required for synchronization, energy consumption, computation energy, and synchronization error. As most of the nodes are battery-driven, reducing the energy consumption plays an important role. The applications dictate the synchronization error allowed.

The most prevalent approach to network-wide synchronization is to form a spanning tree network. The root node has the Global Time (GT) and synchronizes the child nodes along the branches using a bi-directional message exchange approach. Miklos Maroti et. al., [3], Jan van Greunen et. al., [4], Rahamatkar S, et. al., [5], L He [6], K L V Sai Prakash et. al., [7], Roberto Solis et. al., [8], Saurabh Ganeriwal et. al., [9]; first, will create a spanning tree of all the nodes in the network. Then, along the branches from the root node, the time information is exchanged. The problem with such algorithms is that error tries to propagate along the path and suffer from the network's higher computation complexity.

On the other hand, the cluster-based time synchronization protocols reduce the communication energy. Zhang J et. al., [10], Wang Z et. al., [11], J. Wu et. al., [12], C In et. al., [13], Jeremy Elson et. al., [14]; first, divide the network into clusters of single-hop and form a tree algorithm among the cluster heads to cover the complete network.

## III. Iteriative Node-pair Clock Synchronization Algorithm

In this paper [15], we provide a node pair iterative time synchronization algorithm. We begin by considering the case when $\alpha = 1$ the skew whence we need to determine only $\delta$, the offset. We assume that node 0 has the correct time and is sending the timestamped packets to node 1. Let $t_k$ be the time at which node 0 send the k-th timestamped packet and $r_k$ be the time on the clock of node 1 when this packet is received at node 1 after random delay on the network, $D_k$ . It is easy to see that $r_k = t_k + \delta + D_k$ i.e., $\delta = r_k - t_k - D_k$ .

We assume $D_k \in [a, b]$ and $0 \le a \le b < \infty$, i.e., $D_k$ are bounded random variables and that a and b are known. Hence, after receiving n timestamped packets, we see that $\delta$ should satisfy $\delta \in [r_k - t_k - b, r_k - t_k - a]$ for $1 \le k \le n$. This means that $\delta \in \left[\underline{\delta n}, \overline{\delta n}\right]$, where

$$\underline{\delta n} = \max_{\{1 \le k \le n\}} \left( \{r_k - t_k\} - b \right) = \delta - \left( b - \max_{\{1 \le k \le n\}} \{D_k\} \right) \quad (1)$$

$$\overline{\delta n} = \min_{\{1 \le k \le n\}} \left( \{r_k - t_k\} - a \right) = \delta - \left( \min_{\{1 \le k \le n\}} \{D_k\} - a \right) \quad (2)$$

The last equality in the above is obtained by noting that $r_k = t_k + \delta + D_k$. Let $\tilde{\delta}(n) := \overline{\delta n} - \underline{\delta n}$ represents the uncertainty width in the estimation of $\delta$ after n received timestamps. The, we have $\tilde{\delta}(n) = (b - a) - \left( \max_{\{1 \le k \le n\}} D_k - \min_{\{1 \le k \le n\}} D_k \right)$ we want to highlight that we have not made any assumptions regarding the random sequence $D_k$ of delays that the timestamp packets experienced, except that they are constrained.

It is intuitively clear that max will approach close to b and min will approach close to a, thereby $\tilde{\delta}(n)$ converges to zero, if the a-priori bounds are accurately selected. Thus, eventually, the offset can be estimated.

## IV. Loop Concept

We start with assuming that, we have a collection of clocks in a loop say 3 as shown in Figure 1. Let us assume that clock 1 with time representation as $t_1$ and clock 2 with time representation as $t_2$ have a relative skew $\alpha_{12}$ and relative offset $\delta_{12}$, and similarly $t_2$ and $t_3$ have relative skew $\alpha_{23}$ and relative offset $\delta_{23}$, and similarly $t_3$ and $t_1$ have relative skew $\alpha_{31}$ and relative offset $\delta_{31}$.

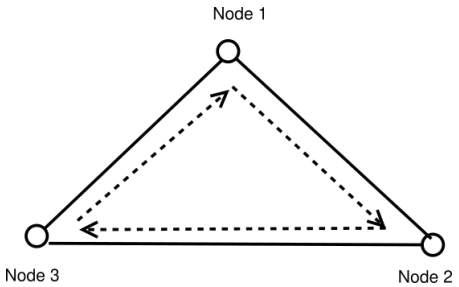

Figure 1: Three node network model

So, we can write the relationships between the clocks as:

$$t_2 = \alpha_{12} \times t_1 + \delta_{12} \quad (3)$$

$$t_3 = \alpha_{23} \times t_2 + \delta_{23} \quad (4)$$

$$t_1 = \alpha_{31} \times t_3 + \delta_{31} \quad (5)$$

We will formulate the problem, to capture the key ideas behind our intuition.

## V. Problem Formulation

### *5.1 Only Offset*

In this case, we assume that all clocks in the loop are running at same speed and have only relative offsets present. The equations 3 to 5 can be rewritten as

$$t_2 = t_1 + \delta_{12} \quad (6)$$
$$t_3 = t_2 + \delta_{23} \quad (7)$$
$$t_1 = t_3 + \delta_{31} \quad (8)$$

Rearranging and on simplification we get

$$t_1 = t_1 + \delta_{12} + \delta_{23} + \delta_{31} \quad (9)$$
$$\delta_{12} + \delta_{23} + \delta_{31} = 0 \quad (10)$$

We can now formulate that **the total offset in a loop is zero**.

### *5.2 Only Skew*

In this case, we assume that all clocks in the loop have zero relative offsets and have only relative skews present. The equations 3 to 5 can be rewritten as

$$t_2 = \alpha_{12} \times t_1 \quad (11)$$
$$t_3 = \alpha_{23} \times t_2 \quad (12)$$
$$t_1 = \alpha_{31} \times t_3 \quad (13)$$

Rearranging and on simplification we get

$$t_1 = \alpha_{12} \times \alpha_{23} \times \alpha_{31} \times t_1 \quad (14)$$
$$\alpha_{12} \times \alpha_{23} \times \alpha_{31} = 1 \quad (15)$$

We can now formulate that **the product of all skews in a loop is 1**.

## VI. Proposed Algorithm

### *6.1 Single Loop Fine Grain Multihop Synchronization Algorithm*

Based on the above formulations we now develop algorithms for a single loop multihop networks under the two conditions. We begin by considering a 'n' node network as shown in Figure 2. To analyze, we assume that the nodes have relative offset $\delta_{ij}$ and relative skew $\alpha_{ij}$ between node 'i' and node 'j' forall (ij) ∈ [(12), (23), ……, (n1)]. Further, we will also assume that a packet is transmitted from node 1 to node 2, from node 2 to node 3, and so on up to node n to node 1, before a second packet is transmitted as shown in Figure 3. On circulating the packet, all the nodes will timestamp the received and transmitting time of the packet. Let $t_i$ (k) be the time at which node 'i' sends the kth timestamped packet and $r_i$(k) be the time on the clock on node 'j' when the packet is received at node 'j' after a random delay $D_{ij}$ (k) on the network. We assume the delay $D_{ij}$ (k) ∈ [a, b] are bounded random variables and that 'a' and 'b' are apriori known. This means

$$r_i\,(k) = \alpha_{ij} \times t_i\,(k) + \delta_{ij} + D_{ij}\,(k) \quad (16)$$

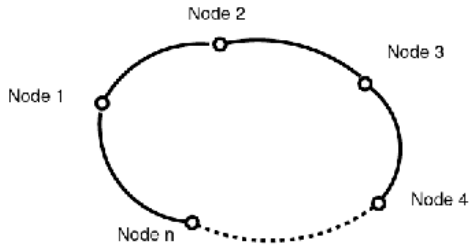


Figure 2: 'n' Node Single Loop Network

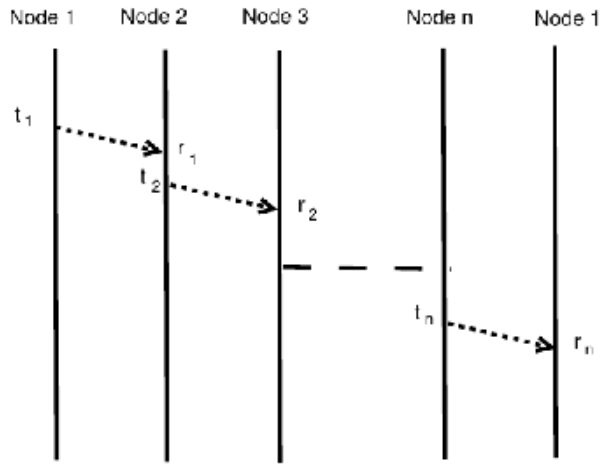


Figure 3: Packet flow diagram in a 'n' node single loop network

We will consider the two cases to describe our multihop algorithm that minimizes the uncertainty region in estimating the relative offset and relative skew between the nodes.

*6.2 Only Offset*

Under the assumption that all the nodes have clock running at same speed we can rewrite the equation 16 as

$$r_i(k) = t_i(k) + \delta_{ij} + D_{ij}(k) \quad (17)$$

and as the following inequality

$$r_i(k) - t_i(k) - b \leq \delta_{ij} \leq r_i(k) - t_i(k) - a \quad (18)$$

This means that $\delta_{ij} \in [\underline{\delta_{ij}}(k), \overline{\delta_{ij}}(k)]$, where

$$\underline{\delta_{ij}}(k) = \{r_i(k) - t_i(k)\} - b = \delta_{ij} - (b - \{D_{ij}(k)\}) \quad (19)$$

$$\overline{\delta_{ij}}(k) = \{r_i(k) - t_i(k)\} - a = \delta_{ij} - (\{D_{ij}(k)\} - a) \quad (20)$$

Using the expression 10, we know that the total offset in a loop is 0 (zero).

$$\sum_{\{(ij)\in[(12),(23),\ldots,(n1)]\}} \delta_{ij} = 0 \quad (21)$$

$$\delta_{\{ij\}} = - \sum_{\substack{\{(lm)\in[(12),(23),\ldots,(n1)]\} \\ (lm \neq ij)}} \delta_{lm} \quad (22)$$

Now with this additional information from the other nodes, the synchronization algorithm works as follows:

$$- \sum_{\substack{\{(lm)\in[(12),\ldots,(n1)]\} \\ (lm \neq ij)}} \delta_{lm} \in \left[ - \sum_{\substack{\{(lm)\in[(12),\ldots,(n1)]\} \\ (lm \neq ij)}} \overline{\delta}_{lm}(k) , - \sum_{\substack{\{(lm)\in[(12),\ldots,(n1)]\} \\ (lm \neq ij)}} \underline{\delta}_{ij}(k) \right] \quad (23)$$

Now the update equations for synchronization algorithm are

$$\underline{\delta}_{ij}(k) = \max\left[\underline{\delta}_{ij}(k-1),\ \underline{\delta}_{ij}(k), -\sum_{\substack{\{(lm)\in[(12),(23),\ldots,(n1)]\}\\(lm\neq ij)}} \overline{\delta}_{lm}(k)\right] \tag{24}$$

$$\overline{\delta}_{ij}(k) = \min\left[\overline{\delta}_{ij}(k-1),\ \overline{\delta}_{ij}(k), -\sum_{\substack{\{(lm)\in[(12),(23),\ldots,(n1)]\}\\(lm\neq ij)}} \underline{\delta}_{lm}(k)\right] \tag{25}$$

And the uncertainty interval $\tilde{\delta}_{ij}(k)$ is given by

$$\tilde{\delta}_{ij}(k) := \overline{\delta}_{ij}(k) - \underline{\delta}_{ij}(k) \tag{26}$$

*6.3 Only Skew*

We assume that all the $n$ nodes clocks running at zero offset but have different skews. In this case, we can rewrite equation 16 as

$$r_i(k) = \alpha_{ij} \times t_i(k) + D_{ij}(k) \tag{27}$$

and as the following inequality

$$\frac{r_i(k) - b}{t_i(k)} \leq \alpha_{ij} \leq \frac{(r_i(k) - a)}{t_i(k)} \tag{28}$$

This means that $\alpha_{ij}(k) \in \left[\underline{\alpha}_{ij}(k),\ \overline{\alpha}_{ij}(k)\right]$ where

$$\underline{\alpha}_{ij}(k) = \frac{r_i(k)-b}{t_i(k)} \tag{29}$$

$$\overline{\alpha}_{ij}(k) = \frac{(r_i(k)-a)}{t_i(k)} \tag{30}$$

Using the expression 15, we know that the product of skew in a loop is 1 (one).

$$\prod_{\{(ij)\in[(12),(23),\ldots,(n1)]\}} \alpha_{ij} = 1$$

$$\alpha_{ij} = \frac{1}{\left\{\prod_{\substack{\{(lm)\in[(12),(23),\ldots,(n1)]\}\\(lm\neq ij)}} \alpha_{lm}\right\}} \tag{31}$$

Now with this additional information from the other nodes, the synchronization algorithm works as follows:

$$\alpha_{ij} \in \left[\underline{\alpha}_{ij}(k),\ \overline{\alpha}_{ij}(k)\right]$$

$$\frac{1}{\left\{\prod_{\substack{\{(lm)\in[(12),\ldots,(n1)]\}\\(lm\neq ij)}} \alpha_{lm}\right\}} \in \left[\frac{1}{\left\{\prod_{\substack{\{(lm)\in[(12),\ldots,(n1)]\}\\(lm\neq ij)}} \overline{\alpha}_{lm}(k)\right\}}, \frac{1}{\left\{\prod_{\substack{\{(lm)\in[(12),\ldots,(n1)]\}\\(lm\neq ij)}} \underline{\alpha}_{lm}(k)\right\}}\right] \tag{32}$$

Now the update equations for synchronization algorithm are

$$\overline{\alpha}_{ij}(k) = \min\left\{\overline{\alpha}_{ij}(k-1), \overline{\alpha}_{ij}(k),\ \frac{1}{\left\{\prod_{\substack{\{(lm)\in[(12),(23),\ldots,(n1)]\}\\(lm\neq ij)}} \underline{\alpha}_{lm}(k)\right\}}\right\} \tag{33}$$

$$\underline{\alpha}_{ij}(k) = \max\left\{\underline{\alpha}_{ij}(k-1), \underline{\alpha}_{ij}(k),\ \frac{1}{\left\{\prod_{\substack{\{(lm)\in[(12),(23),\ldots,(n1)]\}\\(lm\neq ij)}} \overline{\alpha}_{lm}(k)\right\}}\right\}$$

And the uncertainty interval $\tilde{\alpha}_{ij}(k)$ is given by (34)

$$\tilde{\alpha}_{ij}(k) := \overline{\alpha}_{ij}(k) - \underline{\alpha}_{ij}(k) \tag{35}$$

## VII. Simulations and Results

### *7.1 Three node network*

For a three node network, as shown in figure 4, we have simulated to observe the behaviour of the two algorithms i.e., only offset and only skew. In the simulations we have taken the bounded delay $[a_i,b_i]$, which is identical for all the three links as [1,5]. The random delay, the message packet takes from transmitting node to receiving node is assume to be uniformly distributed in the above bounded delay interval for all the links. We will assume that a packet is transmitted from node 1 - node 2 - node 3 - node 1 in a circular fashion before they circulate the second packet. On circulating the packet, all the nodes will timestamp the received and transmitting time of the packet. At time '$t_i$' a packet is transmitted by node 'i' and the same is received at node 'j' at time '$r_j$'on its clock. The convergence curve of the estimated uncertainty region for the 50 message packets transmitted is observed for only offset, and only skew. We have also plotted the expected uncertainty width convergence curve of a single node pair (only one link). In this case also we have considered the delay bounds as [1, 5] and the random delay of the message packets to be uniformly distributed. It can be observed from the graphs that the multi-hop synchronization gives better estimates of uncertainty width than the node pair synchronization.

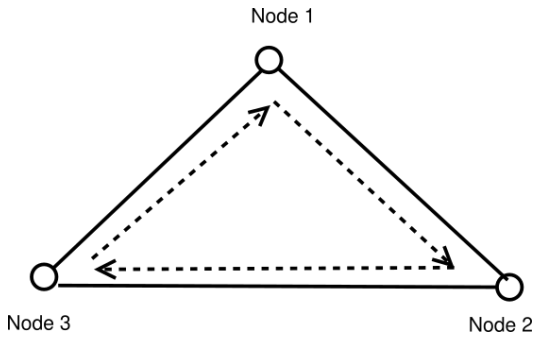


Figure 4. Three node network

### *7.2 Only Offset*

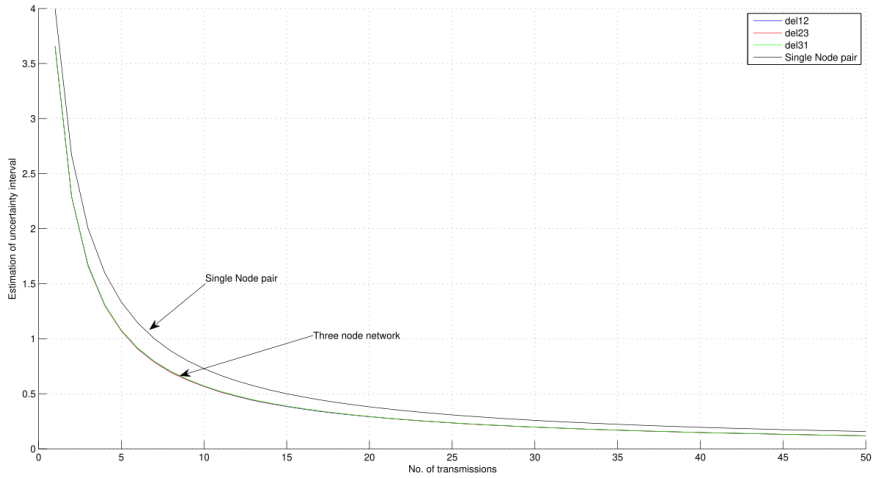


Figure 5: Simulation of Sample path the uncertainty offset width for the three node pairs. The expected uncertainty offset width with `pairwise synchronization' is also shown.

### *7.2 Only Skew*

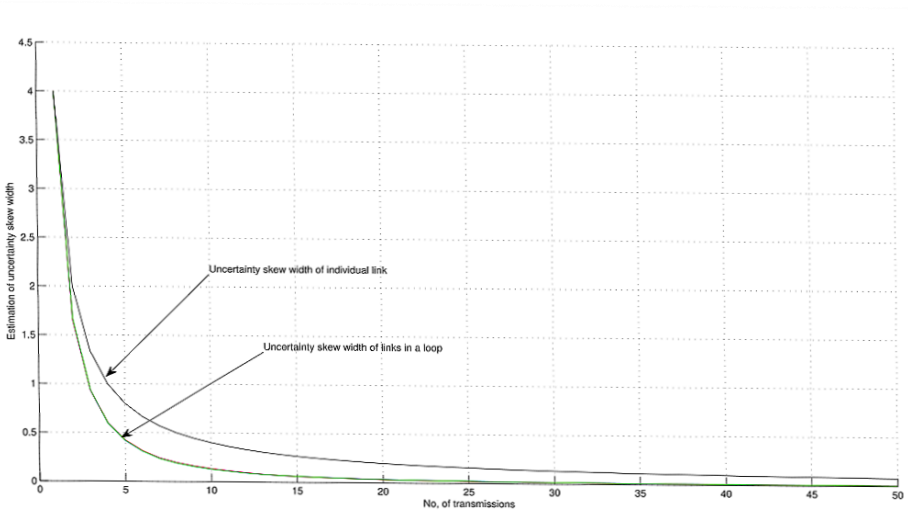


Figure 6: Simulation of Sample path the uncertainty slope width for the three node pairs. The expected uncertainty slope width with `pairwise synchronization' is also shown

## VII. CONCLUSION & FUTURE WORK

We present a novel loop-based fine-grained network-wide time synchronization algorithm motivated by Kirchhoff's voltage law. The total offset sum in the loop as given in the equation 10 is zero, and the entire slope product in the loop as shown in the equation 14 is one. From Fig 5, we can conclude that the uncertainty offset error in estimating the offset for the three-node pairs using loop-based time synchronization is much better than that of single node-pair synchronization. It is also true in the case of uncertainty slope error, as shown in Fig.6. The analytical

and simulation results show that in a three-loop network, on average, approximately 15 time-stamped message packets are required to be transmitted compared to an average of 20 time-stamped message packets as are necessary for a single node pair. The uncertainty reduction of 10-time units to the 1-time unit was considered for the above simulation purposes. This outstanding outcome is an improvement compared to the RBS method, which needs 80 message packets for three receivers to synchronize. The loop network need not have to generate a spanning tree or divide the network into clusters, thereby reducing energy consumption and extending the lives of the nodes and network. With these encouraging results, we plan to analyze energy analysis and the time taken for synchronization as the network size grows.

REFERENCES


[1] Bharath Sundararaman, Ugo Buy and Ajay D Kshemkalyani, Clock Synchronization for Wireless Sensor Networks: A Survey, Elsevier Journal on Adhoc Networks, Vol 3, Issue 3, May 2005, P 281-323. DOI: 10.1016/j.adhoc.2005.01.002.

[2] Fikret Sivrikaya and Bulent Yener, Time synchronization in Sensor Networks: A Survey, IEEE Network, Vol 18, issue 4, p 45-50, July-Aug 2004.DOI: 10.1109/MNET.2004.1316761.

[3] Miklos Maroti, Branislav Kusy, Gyula Simon and Akos Ledeczi, The Flooding Time Synchronization Protocol, SenSys'04, November 3–5, 2004, Baltimore, Maryland, USA.

[4] Jana van Greunen and Jan Rabaey, Lightweight Time Synchronization for Sensor Networks, WSNA'03, September 19, 2003, San Diego, California, USA.

[5] Rahamatkar S., and Agarwal A, A Reference based, Tree Structured Time Synchronization approach and its analysis in WSN, International Journal of Adhoc, Sensor & Ubiquitous Computing (IJASUC) Vol.2, No.1, March 2011 DOI : 10.5121/ijasuc.2011.2103 20.

[6] L. He, Time Synchronization Based on Spanning Tree for Wireless Sensor Networks, 4th International Conference on Wireless Communications, Networking and Mobile Computing, Dalian, 2008, pp. 1-4, doi: 10.1109/WiCom.2008.846.

[7] K L V Sai Prakash Sakuru and N Bheema Rao, K-Hop Iterative Time Synchronization over a Linear Connected Wireless Sensor Networks, 4th International Conference on Smart Computing and Informatics (SCI-2020), Hyderabad, INDIA.

[8] Roberto Solis, Vivek S Borkar and P R Kumar, A New Distributed Time Synchronization Protocol for Multihop Wireless Networks, 45th IEEE Conference on Decision and Control, January 2007,DOI: 10.1109/CDC.2006.377675.

[9] Saurabh Ganeriwal, Ram Kumar and Mani B Srivastava, Timing-sync Protocol for Sensor Networks, SenSys '03: Proceedings of the 1st international conference on Embedded networked sensor systems, Nov 2003, p 138–149. DOI:10.1145/958491.958508.

[10] Zhang J., Lin S., Liu D, Cluster-Based Time Synchronization Protocol for Wireless Sensor Networks, Algorithms and Architectures for Parallel Processing, ICA3PP 2014, Lecture Notes in Computer Science, vol 8630. Springer, Cham. DOI:10.1007/978-3-319-11197-1-55.

[11] Wang, Z.; Zeng, P.; Zhou, M.; Li, D.; Wang,J., Cluster-Based Maximum Consensus Time Synchronization for Industrial Wireless Sensor Networks, J. . Sensors 2017, 141. DOI: 10.3390/s17010141.

[12] J. Wu, L. Zhang, Y. Bai and Y. Sun, Cluster-Based Consensus Time Synchronization for Wireless Sensor Networks, IEEE Sensors Journal, vol. 15, no. 3, pp. 1404-1413, March 2015, doi: 10.1109/JSEN.2014.2363471.

[13] C. In, C. Hong and M. Mamun-Or-Rashid, Passive Cluster Based Clock Synchronization in Sensor Network, Advanced Industrial Conference on Telecommunications/Service Assurance with Partial and Intermittent Resources Conference/E-Learning on Telecommunications Workshop, Lisbon, Portugal, 2005 pp. 340-345. doi: 10.1109/AICT.2005.68.

[14] Jeremy Elson, Lweis Girod and Deborah Estrin, Fine Grained Network Time Synchronization using Reference Broadcasts, ACM SIGOPS Operating Systems Review, December 2002. DOI:10.1145/844128.844143.

[15] K L V Sai Prakash Sakuru and N Bheema Rao, An Iterative Node-pair Time Synchronization (INTS) for Wireless Sensor Networks, Electronic Systems and Intelligent Computing, ESIC 2020, Lecture Notes in Electrical Engineering 686, P 495-505, https://doi.org/10.1007/978-981-15-7031-5-47.