Journal of Intelligent & Fuzzy Systems 36 (2019) 4531–4543
DOI:10.3233/JIFS-179005
IOS Press

A testbed validated simple time synchronization protocol for clustered wireless sensor networks for IoT

G.S.S. Chalapathi*, Vinay Chamola and S. Gurunarayanan
Department of Electrical and Electronics Eng., Pilani Campus, BITS-Pilani, Pilani, Rajasthan, India

Abstract. Wireless Sensor Networks (WSNs) are set to play an important role in the Internet of Things (IoT). WSNs are deployed for many IoT applications like Smart-Street Lighting, Smart-Grid, etc. Time Synchronization Protocol (TSP) is an important protocol in WSNs and it is used for many of its operations. Most of the existing TSPs for WSNs are simulation-based works, which do not fully prove their effectiveness for WSNs. Further, the Line-of-Sight (LOS) conditions in which the WSN nodes are deployed can significantly affect the performance of these TSPs. However, most of the existing protocols neither talk about the LOS conditions in which these protocols were tested nor prove their effectiveness for different LOS conditions. To address these aspects, a synchronization protocol for cluster-based WSNs called a *Simple Hierarchical Algorithm for Time Synchronization* (H-SATS) has been proposed in this work and its performance is tested on a densely deployed large-sized WSN testbed in different LOS conditions. Further, H-SATS has been compared with the traditional regression-based method, which is the core synchronization scheme for different synchronization protocols in clustered WSNs. Experiments show that H-SATS outperforms the regression method in terms of synchronization accuracy to a maximum of 26.7% for a 30-node network.

Keywords: Cluster-based topology WSN, line-of-sight (LOS) conditions, non-line-of-sight (NLOS) condition, time synchronization protocol, wireless sensor networks (WSN)

1. Introduction

Rapid technological advancements made over the past 30 years has made it possible to connect over the internet not just computers but also different appliances, devices, etc. This has led to the advent of the Internet of Things (IoT). Wireless Sensor Network (WSN) is set to play an important role in the IoT. WSNs are an integral part of many IoT applications like smart-street lighting systems [3], smart city [13], smart grids [27], etc. Also, WSNs have been used for various applications like agriculture [24], industrial monitoring [17], environmental monitoring [20], home automation [12], object tracking [8], personal health monitoring [2], etc. Some inherent advantages of WSN like low cost, low data bandwidth requirements, bidirectional communication capabilities, etc. [32] make it suitable for the IoT applications.

A WSN is a network of a number of devices called *nodes*, which are responsible for cooperatively sensing an event or phenomenon locally. Each node is typically equipped with relevant sensor(s), a microcontroller to provide the intelligence and a radio for communication. WSNs are characterized by low data rates, low power consumption, ad-hoc joining and leaving of nodes, self-configuring and self-healing capabilities during link failures, etc. Most of the

*Corresponding author. G.S.S. Chalapathi, Department of Electrical and Electronics Eng., Pilani Campus, BITS-Pilani, Pilani, Rajasthan-333031, India. Tel.: +91 1596 255880; Fax: +91 1596 244183; E-mail: gssc@pilani.bits-pilani.ac.in.

nodes in the network are battery powered because they are often deployed in an environment where a continuous power source is not available.

Time Synchronization Protocol (TSP) is one of the most important protocols in a distributed network like WSN. TSPs provide a common reference of time either for the entire network or for some part(s) of the network. The common time reference provided by a TSP could be either (i) an absolute time like Coordinated Universal Time (UTC)/local time for a zone or (ii) relative/local time which is specific to the network or part of the network or (iii) just provide a basis to decide the chronology of some events in the WSN. The kind of time reference provided by a TSP depends on the requirements of the WSN application. TSP plays an important role in applications/functions like data gathering and fusion, time-based localization, TDMA based communication, power management protocols, etc. Time-Synchronization is all the more important for a WSN used in an IoT network because the sensor data recorded by a WSN is monitored and analyzed at remote locations. Thus, it is important to timestamp such sensor data according to a common clock valid across this IoT network. But generic TSPs like NTP [7] are not suitable for a WSN which has limitations like low bandwidth, low energy availability, etc. It is possible to achieve time synchronization among the WSN nodes if they are all equipped with devices like Global Positioning System (GPS) modules. But it is not possible to have a GPS module on every node as it increases the cost of each node tremendously. Further, GPS connectivity is very limited in indoor locations, in cloudy conditions, etc. [14]. We elaborate on the requirements and characteristics of a TSP in WSN in the next section.

There are several TSPs which have been proposed and even used in WSNs like RBS [14], TPSN [28], FTSP [25], SLTP [29], L-SYNC [21], etc. Many of them have demonstrated a synchronization accuracy of the order of microseconds while others have demonstrated an accuracy of lower orders. But there are very few protocols which have been demonstrated on a hardware testbed. We require a TSP which is simple and efficient (in terms of the computations involved), accurate, and which has been proved suitable for WSN on a hardware platform. Also, most of the works do not describe the environment in which the experiments have been performed. The environmental conditions, i.e., whether the communicating nodes are Line-of-Sight (LOS) or Non-Line-of-Sight (NLOS) significantly affects the performance of a TSP, especially in IoT applications which have NLOS environments (like Home Automation application). We, therefore, propose a Simple Hierarchical Algorithm for Time Synchronization (H-SATS) whose performance has been tested in this work on a large size testbed. Further, we show its performance compared to other relevant protocols in different environmental conditions like dense LOS and a mixture of LOS and NLOS network. The scope of this work is limited to WSNs in which the nodes are organized into clusters i.e. a cluster-based WSN. We will elaborate on the reasons for choosing cluster-based WSN in the next section.

The contributions of this paper are as follows:

a) It presents H-SATS which is a simple and efficient TSP for a hierarchical clustered WSN.
b) It achieves microsecond synchronization accuracy for varying network sizes.
c) It shows the performance of H-SATS on a real WSN testbed.
d) It also reports a performance analysis carried out for different LOS conditions which proves the degree of effectiveness of H-SATS in different environments.

Rest of the paper is organized as follows. Section 2 discusses the background of this work and previous works related to this paper. We describe H-SATS along with its mathematical basis in Section 3. The experimental setup and the methodology used to demonstrate our algorithm are described in Section 4. The results of the experiments performed on the hardware testbed have been presented and analyzed in Section 5. Finally, Section 6 offers conclusions.

2. Literature survey

2.1. Background

In a WSN, each node keeps its own local time using cheap crystal oscillators which do not have a very precise clock tick rate. However, we cannot use high-precision crystal oscillator on every node as it increases the cost of the network tremendously. Further climatic conditions like temperature, humidity, etc. and aging of the crystals affects the accuracy of the local time at these nodes [9]. Therefore it is very essential to synchronize these nodes to a level of accuracy required by the application. Some industrial, health and automotive applications require a micro-second accuracy time synchronization protocol [1].

Time synchronization in WSN is particularly challenging as the nodes are deployed in harsh environments like industrial plants [17], volcanoes [11], etc. They are deployed for unattended operation targeted to work with minimal or no maintenance, generally for a few months to a couple of years. Thus the synchronization protocol must be **robust** to node failures and must provide synchronization of required accuracy-level. Also, since a WSN is an ad-hoc network which involves joining and leaving of nodes in a dynamic manner, the TSP should be **scalable** with the increasing network size without significant degradation in its performance. Also since a WSN is largely battery operated, the TSP should be **energy efficient**. We say a TSP is energy efficient if it requires the least number of packets for achieving the synchronization and/or requires infrequent resynchronizations. Further, WSNs are often used in critical applications like health monitoring, surveillance, etc. which require immediate reporting of emergency events. Therefore the TSP protocol must be **immediate** and thus must be computationally efficient. A detailed discussion about the characteristics and requirements of a TSP in WSNs can be found in [9].

As stated earlier, it is not possible for all the nodes in a WSN to be equipped with either a GPS device or high accuracy crystal oscillator. Thus each network typically consists of either one or few nodes called the anchor or master nodes which provide the time reference for synchronization of all other nodes in the network which are involved in the synchronization process. These anchor nodes are either equipped with a GPS device or they simply provide the reference for the local clock of the network. All other non-anchor nodes determine their local clocks with respect to these anchor nodes. These non-anchor nodes are called agent nodes [4, 5].

Time synchronization typically involves the exchange of packets between the anchor nodes and the agent nodes periodically. The period of such exchange depends on the level of synchronization accuracy required, the efficiency of the TSP, capabilities of the WSN nodes, etc. As stated above, it is desired that this period is as large as possible for the energy efficiency of the TSP and for prolonging the lifetime of the WSN nodes. However, various dynamic factors in a WSN like nondeterministic network access time, processor load on the WSN node, propagation time etc. cause time delays in the delivery of the packet to the receiver after it is generated at the sender. These delays like send time, access time, propagation time, reception time, receive time, etc. become a source of significant error in time synchronization. They have been elaborately discussed in [9, 25]. Of these, send time, receive time, access time and interrupt handling times are non-deterministic in nature while transmission and reception times and propagation delays are deterministic [25].

Having discussed the background of TSPs in WSN, we describe the related work in the next subsection.

2.2. Related work and problem addressed

As stated before, the focus of this work is a WSN with cluster-based topology. This kind of topology is extremely energy efficient, particularly in route formation, data fusion, etc. [21, 31] and these functionalities/applications are most common parts of a generic WSN. Thus we chose to focus our TSP for a clustered WSN. In a cluster-based topology, a few nodes are grouped together to form clusters. Each cluster has a cluster-head (CH) and other nodes are the cluster members (CMs).

TSPs for clustered WSN have been reported by L-SYNC [21], SLTP [29] and PC-Avg [23]. In PC-Avg, the CH collects information about the local time of its CMs, calculates its average and sends this average to the CMs. PC-Avg shows very poor synchronization accuracy and does not account for the previously mentioned non-deterministic and deterministic delays during packet transmission and reception. Also, it requires frequent synchronizations which undesirably increases the packet traffic in the network and thus decreases the lifetime of the nodes [29].

SLTP [29] and L-SYNC [21] synchronize the CMs to their respective CH. They use the regression-based scheme to improve the synchronization accuracy. SLTP and L-SYNC differ from each other in the way form the clusters. Both SLTP and L-SYNC have poor synchronization accuracy of the order of milliseconds, which is not sufficient for many applications. Also, they do not account for non-deterministic and deterministic delays in the packet transmission.

Recent works like CCTS [15] and CMTS [33] use consensus-based schemes to achieve synchronization. They perform an intra-cluster synchronization first which is followed by an inter-cluster synchronization to converge the virtual clocks on the nodes to a consensus clock. However, CCTS and CMTS use a large number of iterations to achieve convergence. Thus, they are not practically suitable for energy-constrained WSN nodes and thus we have not compared our protocol with CCTS and CMTS. Also, CCTS does not consider the non-deterministic delay in packet transmissions. It writes-off these delays by saying that MAC-layer timestamping will take care of these delays. But that is not true in reality [25].

Importantly, the performance of CMTS, CCTS, SLTP, L-SYNC, and PC-Avg have only been demonstrated on simulators. Their performance on a real WSN testbed and their suitability for WSNs is thus questionable. Simulator-based works cannot give a complete picture of the performance of a TSP as they make many assumptions at a high level of abstraction, they do not consider packet loss and its effect on synchronization accuracy [6]. Another aspect that the works on TSPs lack are that they do not do a performance analysis under different LOS conditions. The performance of a TSP is affected by the fact whether the communicating WSN nodes are in LOS condition or in NLOS condition. Thus, as the conditions vary, the results may vary. We need to know the degree of efficiency of a TSP in a given condition. This would also help us choose an appropriate TSP for a given scenario. We, therefore, address the above issues in our TSP called H-SATS that is presented in this paper. We discuss the methodology of H-SATS in the next section.

3. H-SATS

3.1. Clock model and mathematical basis

A typical network having a cluster based topology considered in H-SATS is shown in Fig. 1. A Base-Station (BS) is connected to the internet. All the cluster heads are connected to the BS either directly or in a hierarchical manner. A cluster consists of a cluster head and cluster members. Each cluster member communicates only to its CH. The CH nodes are chosen a priori on the basis of their hardware capabilities like superior computation capacity compared to a CM, availability of continuous power supply or greater battery energy availability, etc. We consider a network where nodes are stationary and assume that the nodes are connected via symmetric links for the sake of simplicity. A CM which can hear from more than one CH is called a gateway node. The network is divided into clusters using the cluster formation technique discussed in SLTP [29] or L-SYNC [21]. We can incorporate the feature by which all the nodes in the cluster become CH in turns so that no single node is burdened with CH role which could lead to draining of energy resources on that node.

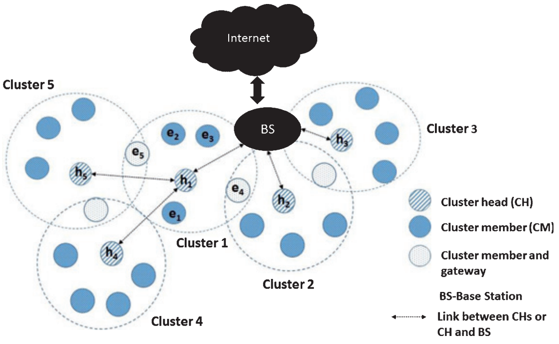


Fig. 1. A typical network considered.

We consider a WSN network consisting of n nodes defined by the set $\mathcal{N}$. Further, we have u cluster heads in this network defined by the set $\mathcal{H} \triangleq \{h_1, h_2, \ldots h_u\}$ and there are v cluster members defined by the set $\varepsilon \triangleq \{e_1, e_2, \ldots, e_v\}$. The network consists of only nodes which are either CHs or CMs and there is no node which is both a CH and CM simultaneously. We define a function $f_{CM}(x, y)$ which tells whether a node y is a CM of a CH x or not, i.e.

$$f_{CM}(x, y) = \begin{cases} 1 \text{ if node } y \text{ is a CM of the CH } x \\ 0 \text{ otherwise.} \end{cases} \tag{1}$$

We also define a function $g_{CH}(x, y)$, which tells whether a CH node y is a child of the BS or CH node x or not, i.e.

$$g_{CH}(x, y) = \begin{cases} 1 \text{ if CH node } y \text{ is a child of } x \\ 0 \text{ otherwise.} \end{cases} \tag{2}$$

The clock model considered for this work is the skew-offset model [9, 4, 18]. This model relates the local time of a node with the global time using the following equation.

$$C_i(t) = \alpha_i t + \beta_i, \tag{3}$$

where t is the global time or the time of the reference clock, α_i is the skew and β_i is the offset of the local clock of the ith node with respect to the reference clock. Skew refers to the frequency or the rate of the local clock with respect to the reference clock and offset is the difference between the local clock and reference clock at time t. Ideally $\alpha_i = 1$ and $\beta_i = 0$ when the local clock is fully synchronized to the reference clock. However, due to the clock crystal's imperfections as discussed before, in reality, these ideal conditions will not be true. Further, the skew is not constant over time, i.e., $\frac{d\alpha_i}{dt} \neq 0$ is not strictly true. But skew can be approximated to a fixed value as the change of skew with time is very slow [9, 22].

H-SATS is a simple synchronization scheme which was presented in a preliminary form in [10]. In [10], all the CMs are synchronized to its CH only. The present work H-SATS extends [10] by achieving network-wide synchronization which is very helpful for a hierarchical WSN used in an IoT network. Further, this work also does a performance analysis of H-SATS for different LOS conditions, i.e., in a complete LOS condition where all the nodes in a cluster are LOS and a mixed-LOS condition where some CMs are LOS with their CH while other CMs are NLOS with their CH. A detailed explanation of the cluster-tree formation, cluster-formation phase, and the mathematical basis of H-SATS is also presented in this work. The CH tree formation phase and the cluster formation phase are described in the following subsection.

3.2. *CH tree formation and cluster formation phases*

We refer to Fig. 1 to explain different phases of H-SATS. The BS initiates the CH tree formation phase by broadcasting *Hello_packet*. The BS is at *level-0* and this level-information is also present in this *Hello_packet* of the BS. The CHs h_1, h_2, and h_3 which are able to hear these packets assign themselves a *level-1*. These CHs, in turn, broadcast *Hello_packet* with its level information and also its id denoted as *CH_ID*. The CHs which hear these packets assign themselves as *level-2* and also note the *CH_ID* of their parent CH. In the example that we are referring to in Fig. 1, h_4 and h_5 hear the *Hello_packet* sent by h_1 and they become the children of h_1. If a CH hears the *Hello_packet* of more than one parent CH, it chooses the CH whose packet is received with greater signal strength as its parent CH. In this way, CH tree is formed. Note that this CH tree formation is performed only once as the CHs are fixed and are stationary.

The cluster tree formation phase is followed by the cluster formation phase. The CHs begin this phase by broadcasting the *CM_joining packet*. This packet also contains the *CH_ID* of the broadcasting CH. All the nodes (which are not CHs), which hear this *CM_joining* packet from a CH, join the cluster of that particular CH. In this case, the nodes e_1, e_2, e_3, e_4 and e_5 hear the *CM_joining* packet of h_1 and join the Cluster 1. A node which hears the *CM_joining* packets of multiple CHs joins the cluster of more than one CH. For instance, the node e_4 hears the *CM_joining* packets of h_1 and h_2 and thus joins both Cluster 1 and Cluster 2. Such a node becomes a cluster gateway node as it is a part of more than one cluster. Since this gateway node has more than one CH, it records the *CH_ID*s of all its CHs. It sends this list of its CHs to all its CHs. So e_4 sends the list of its CHs to h_1 and h_2 which will help h_1 and h_2 to communicate with each other through e_4. After the cluster formation phase is complete, when a new node joins the network, it can search for a CH by broadcasting a *CH_initial* packet. Any CH which hears this packet and is ready to join this newly joining node into its cluster, responds to this packet. If the new node receives a response from more than one CH, it joins more than one cluster and it becomes a gateway node. This phase is followed by the synchronization phase.

3.3. *Synchronization phase*

The level-1 CHs are first synchronized to the BS. In Fig. 1, the BS synchronizes h_1 (i.e.CH of cluster 1), h_2 and h_3. These CHs, in turn, synchronize their children CHs i.e. in this case h_1 synchronizes h_4 and h_5. Once the CHs are synchronized, the CHs synchronize the CMs within their clusters. For instance, h_1 synchronizes its own CMs i.e. e_1, e_2, e_3, e_4 and e_5. The method by which a CH is synchronized with its parent CH is the same as the method by which a CM is synchronized with its CH. We first describe the procedure for synchronizing a CH with its parent. The parent could be a CH or the BS itself. In H-SATS each child CH is synchronized to its parent CH. Thus the clock model in Equation (3) can be written as

$$C_{ih}(t) = \alpha_{ih} C_h + \beta_{ih}, \tag{4}$$

where C_{ih} is the local time of the child CH, C_h is the local clock of the parent CH, β_{ih} is the relative offset and α_{ih} is the relative skew of the ith child CH with respect to the parent CH.

Synchronization in H-SATS is performed using a two-way message exchange between the parent CH and the child CH as represented in Fig. 2. The packets are time stamped by each node as per its local clock on a packet transmission and reception. The first packet was sent from the parent CH at $T_{1,1}$ as per its local clock. This packet is received by the child CH at $T_{2,1}$ as per its local clock. The child CH acknowledges this packet by sending another packet to the parent CH at

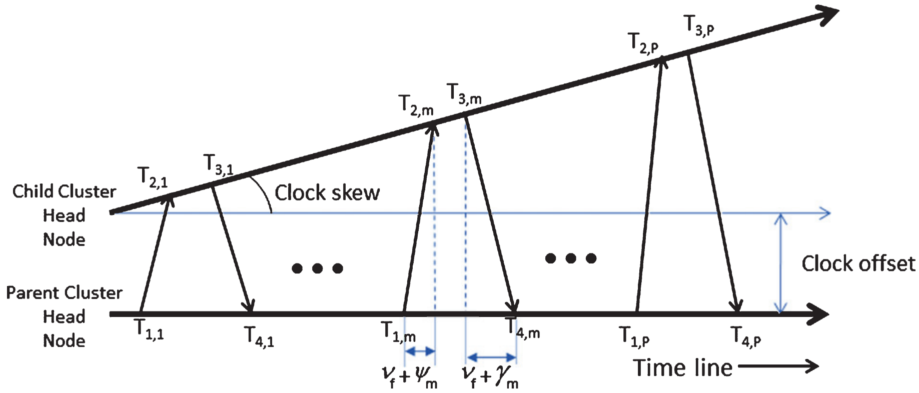


Fig. 2. Two-way message exchange between parent CH head node and child CH node along the time axis.

$T_{3,1}$ as per child CH's local clock. The parent CH receives this acknowledgment packet at $T_{4,1}$ as per its local clock. The packet exchanged between a child CH and its parent CH will contain the time stamps of transmission and reception, i.e., $T_{1,1}$, $T_{2,1}$, $T_{3,1}$, and $T_{4,1}$. MAC-layer timestamping as used in [25, 28] can be employed here to reduce the send time, access time and reception time delays in the transmission. Thus, when the acknowledgment packet is received at the parent CH, it gets all the four local time stamps of one complete two-way exchange. A total of P such two-way exchanges are performed between a parent CH and one child CH. Using Equation (4) we can relate the time stamps of an mth two-way exchange (where $1 \leq m \leq P$) as follows

$$T_{2,m} = \alpha_{ih}(T_{1,m} + \nu_f + \psi_m) + \beta_{ih} \tag{5}$$

$$T_{3,m} = \alpha_{ih}(T_{4,m} - \nu_f - \gamma_m) + \beta_{ih}, \tag{6}$$

where ν_f represents the deterministic delays, ψ_m and γ_m represent the non-deterministic delays in the packet exchange. Note that these delays are measured with respect to the parent CH's local clock.The above Equations (5 and 6) can be written as follows

$$T_{2,m} = \alpha_{ih} T_{1,m} + \beta_{ih} + \alpha_{ih}(\nu_f + \psi_m) \tag{7}$$

$$T_{3,m} = \alpha_{ih} T_{4,m} + \beta_{ih} - \alpha_{ih}(\nu_f + \gamma_m). \tag{8}$$

We observe from Equation (7) that as α_{ih}, ν_f and ψ_m are always positive, $T_{2,m}$ lies above the line $\alpha_{ih} T_{1,m} + \beta_{ih}$. Similarly from Equation (8) we see that since α_{ih}, ν_f and γ_m are always positive, $T_{3,m}$ lies below the line $\alpha_{ih} T_{4,m} + \beta_{ih}$. We know that α_{ih} is the rate of child CH's clock with respect to parent CH's clock. So if we plot the timestamps of child CH (on the y-axis) with respect to parent CH's timestamps, α_{ih} is the slope and β_{ih} is the y-intercept. So very good estimate of the α_{ih} and β_{ih} can be obtained by fitting a line for these points $(T_{2,m}, T_{1,m})$ and $(T_{3,m}, T_{4,m})$ where $1 \leq m \leq P$. An approximate yet a good estimate for this fit-line can be obtained by taking two points M1 and M2 such that

$$M1 = \{0.5 * (T_{4,l} + T_{1,l}), 0.5 * (T_{2,l} + T_{3,l})\} \tag{9}$$

$$M2 = \{0.5 * (T_{4,r} + T_{1,r}), 0.5 * (T_{2,r} + T_{3,r})\} \tag{10}$$

where $l = arg \min\limits_{1 \leq m \leq P} (T_{4,m} - T_{1,m})$ and $r = arg \min\limits_{1 \leq m \leq P, m \neq l} (T_{4,m} - T_{1,m})$.

From Equations (9 and 10), we can note that M1 refers to the two-way exchange with the least deterministic and non-deterministic delays while M2 refers to the next least. Thus when a line is drawn through M1 and M2, the slope of this line gives α_{ih} and the intercept of it gives β_{ih}. Thus we get

$$\alpha_{ih} = \frac{(T_{2,l} + T_{3,l}) - (T_{2,r} + T_{3,r})}{(T_{4,l} + T_{1,l}) - (T_{4,r} + T_{1,r})} \tag{11}$$

$$\beta_{ih} = \frac{(T_{2,l} + T_{3,l})}{2} - \alpha_{ih} \frac{(T_{4,l} + T_{1,l})}{2} \tag{12}$$

The above mentioned simple method to estimate the relative skew and offset has an accuracy comparable to the much complicated and computationally intensive maximum-likelihood estimate [26]. Note that [26] just gives the basis for this simple synchronization scheme and it is a simulation-based work. The current work improvises on this scheme by applying it for clustered WSN and demonstrates its efficacy on a testbed. Further, we have also employed the cluster tree formation, cluster formation phases and a schedule for message exchanges to achieve a network-wide synchronization. Once a CH is synchronized, it initiates the synchronization with its CMs. The CMs of a cluster are synchronized to their CH in the same method as a child CH is synchronized to the parent CH. Each CM performs Q two-way exchanges with its CH and then the relative offset and relative skew of a CM with respect to the CH is calculated.

Thus, after the synchronization phase, the CMs are synchronized to their CH and the CHs are synchronized to the BS. Thus, any sensor data that is recorded at a CM can be easily timestamped according to the internet time by the BS which is synchronized to the internet clock and also to a CM in the WSN. The algorithm of the synchronization phase of H-SATS is presented in Algorithm 1. The important notations used in this paper are summarized in Table 1.

Algorithm 1. H-SATS Algorithm

```
Synchronization Phase
For i = 1:u do
A node $h_i \in \mathcal{H}$ performs two-way exchanges $P$ times
   with each of its parent CH (or BS) $h_j$, (where $h_j \in \mathcal{H}$
   $\cup BS$ and $g_{CH}(h_i, h_j) = 1$).
$h_j$ calculates relative skew and offset of $h_i$ using
   Equations (9–12)
end for
for $i = 1{:}u$ do
   for $j = 1{:}v$ do
      A node $h_i \in \mathcal{H}$ performs two-way exchanges $Q$ times
         with each of its CM $e_j$, (where $e_j \in \varepsilon$ and
         $f_{CM}(h_i, e_j) = 1$).
$h_i$ calculates relative skew and offset of $e_j$ using
   Equations (9–12)
   end for
end for
```

3.4. Analytical comparison with regression-based algorithm

In this subsection, H-SATS is compared with a regression-based synchronization scheme. The regression-based method has been used both in SLTP [29] and L-SYNC [21]. Both L-SYNC and SLTP have been shown to be scalable time synchronization algorithms for clustered WSN. As stated before, SLTP and L-SYNC differ from each other in the cluster formation method while both use linear regression in the synchronization phase. Linear regression-based synchronization method has very high computational complexity and thus is more expensive than H-SATS. The number of multiplications, as well as additions to be performed for time synchronization in the linear regression method, are much higher as compared to those in H-SATS. The computational complexities for synchronization between two nodes (for instance CH and its CM) are compared in Table 2. In this table, Q represents the two-way exchanges performed between the two nodes which are trying to get synchronized with each other (like a CM with its CH). H-SATS requires one subtraction (which can be seen as an addition) to calculate a two-way exchange's duration which is computed by $(T_{4,m} - T_{1,m})$. Thus Q two-way exchanges require Q addition (or subtraction) operations. Further, $2(Q-2)$ subtractions are required for comparison operations to determine the two-way exchange with minimum delays. Further, 4 additions are required in Equations (9 and 10). It requires a total of 7 multiplications, 4 of which are required in Equations (9 and 10) and the rest of the 3 multiplications are required in Equation (12). The size of the timestamp variable in the microcontroller of a WSN node needs to be increased to 32-bits to maintain a clock of microsecond resolution and also to measure a considerable time duration before the clock overflows. When computationally intensive 32-bit multiplications are involved, a reduction in the number of multiplications to an order of 1 gives H-SATS a significant computational edge over regression-based method. WSN processors like MSP430 of TelosB WSN nodes may find it difficult to support the intense computations involved in the regression-based method. Powerful WSN nodes like

Table 1
Notation summary

Notation	Meaning
$\mathcal{N}$	Set representing all the nodes in the network
n	total number of nodes in the network
$\mathcal{H}$	Set containing all the CHs in the network
ε	Set containing all the CMs present in the network
u	number of CHs in the entire network
v	number of CMs in the entire network
P	Number of two-way exchanges between parent CH (or BS) and child CH
Q	Number of two-way exchanges between parent CH (or BS) and child CH

Table 2
Computational Complexity of H-SATS and regression-based method for synchronization between two nodes

Protocol	Additions	Multiplications
H-SATS	($3Q$+3)	7
Regression-based method [21, 29]	($4Q$-1)	($2Q$+6)

NXP's JN5179 [16] or Lotus [19], can execute such intensive computations as they are equipped with 32-bit processors. However, they do so at the cost of consuming higher processor time and power which is generally not desirable for energy constrained WSN nodes. Also, the use of such expensive powerful nodes inhibits their large-scale deployment in WSNs. Thus, H-SATS, which is less computationally intensive, is a suitable TSP for energy-constrained WSN nodes.

4. Experimental setup and methodology used

We have compared H-SATS with the traditional regression-based scheme on TelosB nodes [30] which are based on MSP-430, a 16-bit microcontroller. The network was organized to form clusters of 4 CMs and one CH, i.e., 5 nodes per cluster. These nodes were deployed in two different environments:

1. Line of Sight(LOS)- where all the CMs and CH could communicate without any obstacle
2. Mixed LOS- Mixture of LOS and NLOS where two of the CMs in a cluster were LOS and other two CMs were NLOS with their CH. We will refer to this environment as mixed-LOS environment for brevity in the future

We describe the experimental setup for the above two environments in the following sub-sections.

4.1. LOS environment

The WSN nodes were deployed in an open field as shown in Fig. 3. Each cluster was deployed in an area of 5 sq. meters. The inter-cluster distance was less than 1 meter, i.e., it is a very dense deployment. The experiment was initially performed with just one cluster turned on. Other clusters were turned on one-by-one gradually so as to generate the traffic which will help us analyze the performance of these algorithms in dense traffic conditions. A total of 30 WSN nodes, i.e., 6 clusters were used in this experiment. The CHs were first synchronized to the BS. Then the synchronization packets were exchanged between a CH and its CMs. The CH unicast the synchronization packets to each CM one-by-one and the CM replied to them via a unicast communication. However, the traffic caused by the packet exchange in one cluster did cause packet drops in other clusters. A snapshot of a part of the network deployed can be seen in Fig. 3. The WSN nodes are encircled in red.

Fig. 3. Picture showing part of the network deployed with WSN.

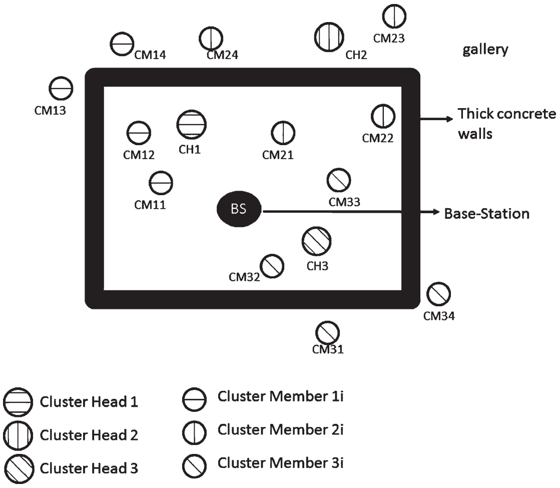


Fig. 4. An indicative diagram depicting part of the deployment for mixed LOS (a mixture of LOS and NLOS) environment.

4.2. Mixed-LOS environment

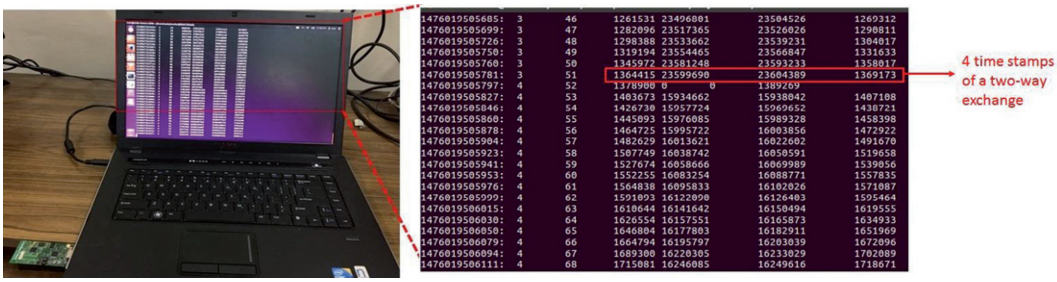
In this deployment, some of the nodes were placed inside a lab and the rest of the nodes were deployed in the gallery outside the lab (as shown in Fig. 4). The BS was inside the lab. The nodes in the gallery were separated from the nodes inside the lab by thick concrete walls. The CH of a cluster could communicate LOS with two of its CMs and other two CMs were in an NLOS position. These CMs in NLOS position with respect to their CH were separated by a thick concrete wall to provide a formidable obstacle for packet transmission. The experiment was performed with only one cluster turned on initially. The other clusters were turned on one-by-one to generate the traffic.

4.3. Methodology used

The methodology described in this subsection used to perform H-SATS and to compare it with the regression-based method is common to both environments mentioned previously. In H-SATS, as stated before, each CM is synchronized with its CH. In the current implementation, we employ a training phase to perform the synchronization and a testing phase to evaluate the performance of the TSP. The synchronization phase (i.e training phase) of H-SATS algorithm has been summarized in Algorithm 1. It is to be noted that the testing phase is not a part of the H-SATS algorithm as such. The training and testing phases are described below in detail.

4.3.1. Training phase

In this phase, each CH initiates two-way packet exchanges (described in section 3.1) with its parent CH (or the BS if the CH is level-1 CH). It performs P two-way exchanges with the parent CH (or BS) and it identifies the two-way exchanges with minimal and next minimal delays as shown in Equations (9 and 10). The relative offset and relative skew of the child CH with respect to the parent is calculated using Equations (11 and 12). This procedure is then repeated by all the CHs with all its CMs to synchronize the CMs to their respective CH. The only difference is each CH performs Q two-way exchanges with each of its CM one-by-one. In the current evaluations, P and Q values have been taken to be 17 the reason for which will be explained in the next section. Also, the timestamps of the two-way exchanges of all CHs with their CMs are forwarded to a PC which has the base-station (as shown in Fig. 5) connected to it. The calculations for H-SATS and regression method (i.e., to calculate the relative skew and offsets) are done on a PC in the current implementation. Figure 5 shows the base-station node connected to the PC on the left. The right side portion of the figure shows the zoomed portion of a part of the PC's screen in which four timestamps collected for a single two-way exchange are encircled in red.



```
1476019505685:  3    46    1261531 23496801          23504526          1269312
1476019505699:  3    47    1282096 23517365          23526026          1290811
1476019505726:  3    48    1298388 23533662          23539231          1304017
1476019505750:  3    49    1319194 23554465          23566847          1331633
1476019505760:  3    50    1345972 23581248          23593233          1358017
1476019505781:  3    51    1364415 23599690          23604389          1369173
1476019505797:  4    52    1378900 0        0        1389269
1476019505827:  4    53    1403673 15934662          15938042          1407108
1476019505846:  4    54    1426730 15957724          15969652          1438721
1476019505860:  4    55    1445093 15976085          15989328          1458398
1476019505878:  4    56    1464725 15995722          16003856          1472922
1476019505904:  4    57    1482629 16013621          16022602          1491670
1476019505923:  4    58    1507749 16038742          16050591          1519658
1476019505941:  4    59    1527674 16058666          16069989          1539056
1476019505953:  4    60    1552255 16083254          16088771          1557835
1476019505976:  4    61    1564838 16095833          16102026          1571087
1476019505999:  4    62    1591093 16122090          16126403          1595464
1476019506015:  4    63    1610644 16141642          16150494          1619555
1476019506030:  4    64    1626554 16157551          16165873          1634933
1476019506050:  4    65    1646804 16177803          16182911          1651969
1476019506079:  4    66    1664794 16195797          16203039          1672096
1476019506094:  4    67    1689300 16220305          16233029          1702089
1476019506111:  4    68    1715081 16246085          16249616          1718671
```

Fig. 5. Base-station collecting the timestamps of the cluster and forwarding it to a PC. The right portion of the figure shows a zoomed out portion.

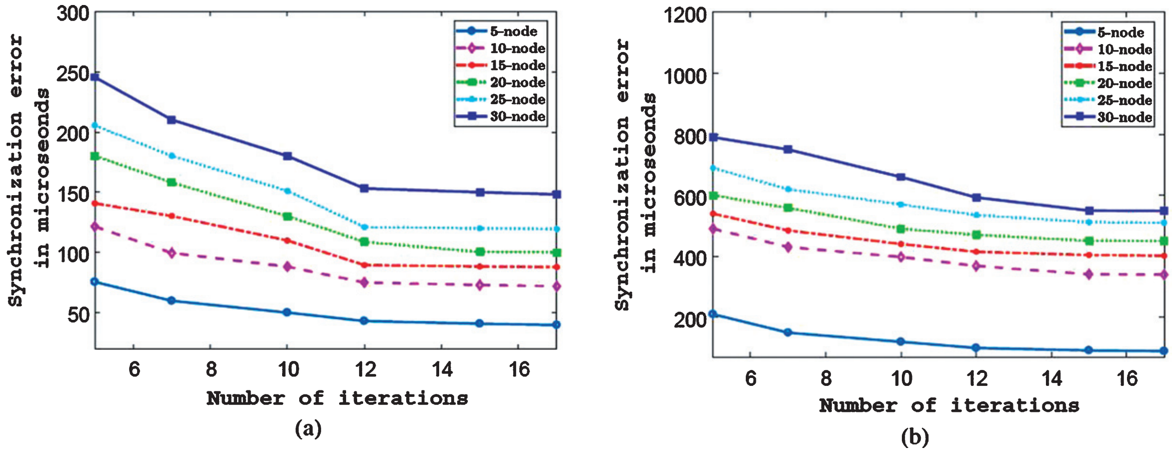
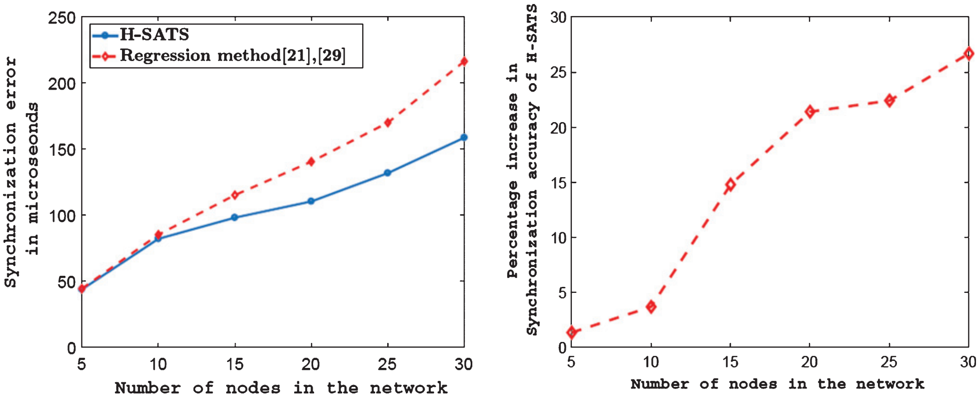


Fig. 6. Variation of synchronization error with the number of two-way exchanges for varying network sizes (a) for LOS environment (b) for mixed-LOS environment.

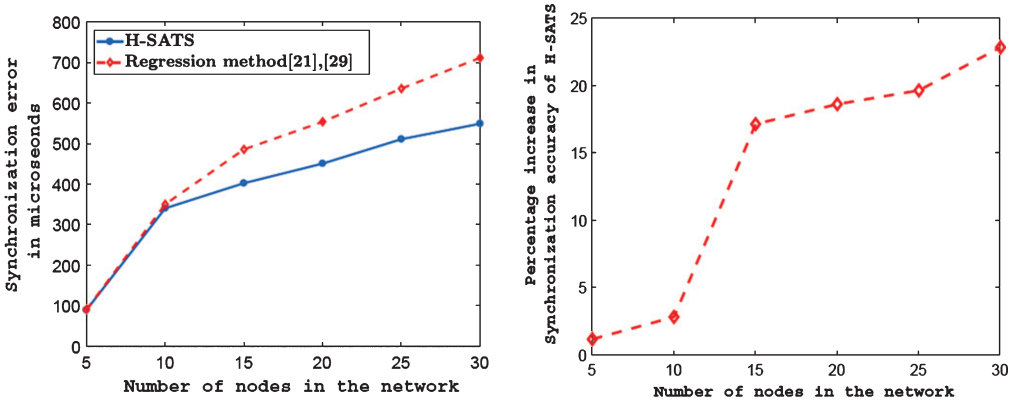
4.3.2. Testing phase

After the synchronization of all nodes is complete, we need to test the synchronization accuracy obtained by both the synchronization schemes. For this, we need to generate a common event which is observed by both the CH and the CMs in a cluster. The CH and CMs will record the occurrence of the event as per their local clock. Each CM will send its timestamp for that event to the CH. Using the Equation (4), the CH will estimate the local time of occurrence of the event as per each CM's local clock by using its own local clock and the relative skew and offset estimated in the training phase. This estimated local time at each CM for the event occurrence is compared with the actual local time at each CM for event occurrence (which was sent by each CM to the CH individually). The difference in these estimated and actual times gives the synchronization error for each CM. This is repeated *S* times in each cluster, i.e., a common event is generated *S* times and the average synchronization error is calculated. The common event is generated by broadcasting packets (called *testing packets*) sent by an independent node which is not part of any cluster of this WSN. We refer to this node as the *third-party node*. This third- party node broadcasts the testing packets at a transmission power level sufficient for it to communicate to all the nodes in the WSN. In fact, having a dense deployment of the clusters makes the area of the network small enough for this third-party node to communicate to all the nodes. The propagation time of the testing packet from the third-party node to each node in the network is almost the same. This method of generating a common-event ensures that the event is seen almost at the same time by all



(a) (b)

Fig. 7. (a) Synchronization error for varying network sizes in LOS environment (b) Percentage increase in synchronization accuracy of H-SATS compared to regression method for varying network sizes in LOS environment.



(a) (b)

Fig. 8. (a) Synchronization error for varying network sizes in the mixed-LOS environment (b) Percentage increase in synchronization accuracy of H-SATS compared to regression method for varying network sizes in the mixed-LOS environment.

the nodes in a cluster. Also, this method ensures that errors like send-time, access time, etc. do not affect the accuracy of the testing phase.

5. Results and Analysis

The nodes were deployed in two different environments as described in the previous section and synchronization error was calculated for H-SATS and the regression-based method. Each CH performed 17 two-way exchanges with each of its parent CH (or BS) and with each of its CM in training-phase, i.e., $P = 17$ and $Q = 17$. We have chosen this number of iterations by analyzing our experimental results. Figure 6(a) and (b) show experimentally obtained variation of synchronization error for different sized networks as a function of the number of two-way exchanges in LOS and mixed-LOS environments respectively. From these figures, we can see that as the number of two-way exchanges increase, the synchronization error shows very little variation. The third-party node was used to generate 15 common-event broadcast packets in testing-phase, i.e., $S = 15$. The synchronization error in microseconds for LOS environment deployment for networks of varying size is shown in Fig. 7a. The percentage increase in synchronization accuracy of H-SATS compared to the regression-based method for varying network size is shown in Fig. 7b. Note that we have referred to the regression-based method as *regression method* in Fig. 7a and 8a for brevity. For a 5-node network, i.e., when there was only one cluster in the network, the regression method gave a synchronization accuracy almost equal to H-SATS. But as the network size increases, H-SATS performed better than the regression method and there is a significant difference in synchronization error for a large network. In a 30-node network, H-SATS had a synchronization error of 158.3 μ s compared to the regression method's 216 μ s. Thus H-SATS showed 26.7% better accuracy than the regression method as seen in Fig. 7b. This difference in performance can be understood from the fact that as the network size increases, there will be significant traffic in the network. This, in turn, increases packet drops, access times, receive time, transmission and reception times, thereby increasing deterministic and non-deterministic delays in packet transmission, particularly during the training phase. Performance of the regression method is affected by all the packets in the training phase and thus its accuracy decreases. On the other hand, H-SATS is affected by just two points M1 and M2, i.e., packets which have minimal deterministic and non-deterministic delays.

The synchronization error in microseconds for the deployment in a mixed-LOS environment for various network sizes is shown in Fig. 8a. The percentage increase in synchronization accuracy of H-SATS compared to the regression method for varying network size is shown in Fig. 8b. In this network also, the regression method had almost the same accuracy as H-SATS for the 5-node network, i.e., when there is only one cluster in the network. There is a sharp increase in synchronization error for the regression method as the network size increases. For a 30-node network, the synchronization error for the regression method is 711.2 μ s compared to H-SAT's 548.63 μ s, i.e., H-SATS has 22.83% better accuracy as seen in Fig. 8(b). As stated before, as traffic increases the non-deterministic and deterministic delays increase which affects the regression method more than H-SATS. Also, in the mixed-LOS environment where some nodes in the cluster are NLOS, we have observed greater packet drops in this deployment which increases the synchronization error. Also, we can see that the difference in synchronization error for a 30-node network between the two synchronization schemes is 57.7 μ s in a LOS environment. But this difference increases to 162.41 μs in a mixed-LOS environment. Thus we see that as more and more nodes become NLOS, the advantage of H-SATS over the regression method in terms of synchronization accuracy increases.

6. Conclusion

We have described and implemented a simple yet accurate synchronization protocol called H-SATS for cluster-based stationary WSNs. H-SATS has also been compared to the regression method used in other TSPs for clustered WSNs. These tests were performed on a real WSN testbed which thus gives a credible proof of performance of H-SATS. H-SATS is computationally more efficient than the regression-based method which is very important for battery constrained WSN. Also, H-SATS has been analyzed in different LOS conditions, i.e., for completely LOS and for a mixture of LOS and NLOS conditions. This shows the stability in the performance of H-SATS in different environments. The difference in synchronization accuracy of H-SATS compared to the regression-based method increases with the network size. Also, in mixed-LOS environments, increase in the synchronization accuracy of H-SATS over the regression method is much higher than in LOS environments. Thus it is very important to choose a simple TSP like H-SATS in WSNs which has some nodes in NLOS condition. Further, we see that H-SATS is especially suited for large sized clustered WSN.

References

[1] A. Elsts, S. Duquennoy, X. Fafoutis, G. Oikonomou, R. Piechocki, and I. Craddock, Microsecond-accuracy time synchronization using the IEEE 802.15.4 TSCH protocol, *Proceedings of 41st Conference on Local Computer Networks Workshops (LCN Workshops)*, 2016, pp. 156–164.

[2] A. Milenković, C. Otto and E. Jovanov, Wireless sensor networks for personal health monitoring: Issues and an implementation, *Computer Communications* **29**(13) (2006), 2521–2533.

[3] A. Zanella, N. Bui, A. Castellani, L. Vangelista and M. Zorzi, Internet of things for smart cities, *IEEE Internet of Things* **1**(1) (2014), 22–32.

[4] B. Etzlinger, H. Wymeersch and A. Springer, Cooperative synchronization in wireless networks, *IEEE Transactions on Signal Processing* **62**(11) (2014), 2837–2849.

[5] D. Bartel, B. Etzlinger and A. Springer, Distributed clock synchronization and ranging in time-variant wireless networks, *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2014, pp. 5744–5748.

[6] D. Djenouri and M. Bagaa, Synchronization protocols and implementation issues in wireless sensor networks: A review, *IEEE Systems Journal* **10**(2) (2016), 617–627.

[7] D.L. Mills, Precision synchronization of computer network clocks, *ACM SIGCOMM Computer Communication Review* **24**(2) (1994), 28–43.

[8] F.H. El-Fouly, R.A. Ramadan, M.I. Mahmoud and M. Dessouky, Resource aware and reliable data reporting algorithm for object tracking in WSNs, *Journal of Intelligent and Fuzzy Systems* **31**(1) (2016), 99–113.

[9] F. Sivrikaya and B. Yener, Time synchronization in sensor networks: A survey, *IEEE Network* **18**(4) (2004), 45–50.

[10] G.S.S. Chalapathi, R. Manekar, V. Chamola, K.R. Anupama and S. Gurunarayanan, Hardware validated efficient and simple Time Synchronization protocol for clustered WSN, *Proc of IEEE Region 10 Conference (TENCON)*, 2016, pp. 2162–2166.

[11] G. Werner-Allen, J. Johnson, M. Ruiz, J. Lees and M. Welsh, Monitoring volcanic eruptions with a wireless sensor network, *Proceedings of the Second European Workshop on Wireless Sensor Networks*, 2005, pp. 108–120.

[12] H. Ghayvat, S. Mukhopadhyay, X. Gui and N. Suryadevara, WSN- and IOT-based smart homes and their extension to smart buildings, *Sensors* **15**(5) (2015), 10350–10379.

[13] I.S.H. MartÃ-nez, I.P.O.J. Salcedo and I.B.S.R. Daza, IoT application of WSN on 5G infrastructure, *Proceedings of the International Symposium on Networks, Computers and Communications (ISNCC)*, 2017, pp. 1–6.

[14] J. Elson, L. Girod and D. Estrin, Fine-grained network time synchronization using reference broadcasts, *SIGOPS Operating Systems Review* **36**(SI) (2002), 147–163.

[15] J. Wu, L. Zhang, Y. Bai and Y. Sun, Cluster-based consensus time synchronization for wireless sensor networks, *IEEE Sensors* **15**(3) (2015), 1404–1413.

[16] JN5179 datahseet. https://www.nxp.com/docs/en/datasheet/JN517X.pdf

[17] L. Krishnamurthy, R. Adler, P. Buonadonna, J. Chhabra, M. Flanigan, N. Kushalnagar, L. Nachman and M. Yarvis, Design and deployment of Industrial sensor networks: Experiences from a semiconductor plant and the North sea, *Proceedings of the 3rd International Conference on Embedded Networked Sensor Systems (SenSys '05)*, 2005, pp. 64–75.

[18] L. Schenato and F. Fiorentin, Average timesynch: A consensus-based protocol for clock synchronization in wireless sensor networks, *Automatica* **47**(9) (2011), 1878–1886.

[19] Lotus WSN node- datasheet. https://www.memsic.com/userfiles/files/Datasheets/WSN/6020-0705-01_A_LOTUS.pdf

[20] M.F. Othman and K. Shazali, Wireless sensor network applications: A study in Environment monitoring system, *Procedia Engineering* **41** (2012), 1204–1210.

[21] M. Jabbarifar, A.S. Sendi, H. Pedram, M. Dehghan and M. Dagenais, L-SYNC: Larger Degree Clustering Based Time-Synchronisation for Wireless Sensor Network, *Proceedings of the Eighth ACIS Int. Conference on Software Engineering Research, Management and Applications*, 2010, pp. 171–178.

[22] M.L. Sichitiu and C. Veerarittiphan, Simple, accurate time synchronization for wireless sensor networks, *Proceedings of the IEEE Wireless Communications and Networking (WCNC 2003)*, 2003, pp. 1266–1273.

[23] M. Mamun-Or-Rashid, C.S. Hong and C.-H. In, Passive cluster based clock synchronization in sensor network, *Proceedings of the Advanced Industrial Conference on Telecommunications/Service Assurance with Partial and Intermittent Resources Conference/E-Learning on Telecommunications Workshop (AICT/SAPIR/ELETE'05)*, 2005, pp. 340–345.

[24] M. Yan, P. Liu, R. Zhao, L. Liu, W. Chen, X. Yu and J. Zhang, Field microclimate monitoring system based on wireless sensor network, *Journal of Intelligent and Fuzzy Systems* **35**(2) (2018), 1325–1337.

[25] M. Maróti, B. Kusy, G. Simon and A. Lédeczi, The flooding time Synchronization protocol, *Proc of the II Intl Conference on Embedded Networked Sensor Systems*, 2004, pp. 39–49.

[26] Q.M. Chaudhari, E. Serpedin and K. Qaraqe, On maximum likelihood estimation of clock offset and skew in networks with exponential delays, *IEEE Transactions on Signal Processing* **56**(4) (2008), 1685–1697.

[27] R. Morello, S.C. Mukhopadhyay, Z. Liu, D. Slomovitz and S.R. Samantaray, Advances on sensing technologies for smart cities and power grids: A review, *IEEE Sensors* **17**(23) (2017), 7596–7610.

[28] S. Ganeriwal, R. Kumar and M.B. Srivastava, Timing-sync Protocol for Sensor Networks, *Proceedings of the 1st International Conference on Embedded Networked Sensor Systems (SenSys '03)*, 2003, pp. 138–149.

[29] S. Nazemi Gelyan, A.N. Eghbali, L. Roustapoor, S.A. Yahyavi Firouz Abadi and M. Dehghan, SLTP: Scalable lightweight time synchronization protocol for wireless sensor network, *Mobile Ad-Hoc and Sensor Networks* (2007), 536–547.

[30] Telosb- datasheet. http://www.memsic.com/userfiles/files/Datasheets/WSN/telosb_datasheet.pdf

[31] W.R. Heinzelman, A. Chandrakasan and H. Balakrishnan, Energy-efficient communication protocol for wireless microsensor networks, *Proceedings of the 33rd Annual Hawaii International Conference on System Sciences*, 2000, p. 10.

[32] Z. Song, M.T. Lazarescu, R. Tomasi, L. Lavagno and M.A. Spirito, High-Level Internet of Things Applications Development Using Wireless Sensor Networks, *Springer International Publishing*, 2014, pp. 75–109.

[33] Z. Wang, P. Zeng, M. Zhou, D. Li and J. Wang, Cluster-based maximum consensus time synchronization for industrial wireless sensor networks, *Sensors* **17**(1) (2017), 141.1–141.16.