Performance Analysis of DTSA in Distributed Network

Vijay Masne
Project Student, M.E. (MT)
Department of Computer Science and Engineering
G.H.Raisoni College of Engineering, Nagpur, India

Urmila Shrawankar
Guide
Department of Computer Science and Engineering
G.H.Raisoni College of Engineering, Nagpur, India

ABSTRACT
In a distributed system, synchronization of time is an important functionality for supporting real-time scenario like online banking applications, database queries and real time applications etc. A clock synchronization is necessary to measure the duration of activities that start on one node and terminate on another one. Although the some of the existing algorithms for time synchronization depend on sending and receiving messages from different system. Scheduling such task sets with time constraints requires timing information to be accurate. Synchronizing these clocks allows services in higher layers to assume a global time within certain bounds of accuracy. The system clocks of different nodes are not works on same clock time. Usually, there is an offset between the times, so accuracy is not achieved. The clock time collection interval, in which clock time of remote nodes are collected and corrected with global time, which determines the period of the correction of the local clocks. Such a decomposition can reduce delay and achieve accuracy in network.

The paper presented in approach towards Delay Tolerant Synchronization Algorithm (DTSA) which is combination of Network Time Protocol (NTP) and Delay Tolerant Protocol (DTP) which synchronizes nodes in a network with Global Time Server (GTS) using NTP. Simulation results show that the DTSA reduces delay and achieve accuracy in scattered network.

General Terms
Synchronization, Operating System and Algorithms

Keywords
Clock Synchronization, Clock Accuracy, Clock drift, scattered environment, Group synchronization, Network Delay, Synchronization issues.

1. INTRODUCTION
The work of the system is used to remove the various issues in designing the synchronization protocols from the previous research and the clock synchronization problems [1], [2]. The problem arises during data transmission in the scattered environment where connected nodes dealing with their local clocks which may differ and this problem is resolved in this system by the application nodes to get synchronized with each other. This leads to relatively high accuracy in a network. In this system multiple modes of operations are required like sleep mode, wake up mode, multiple access modes, data integration mode, etc. [3]; for making the network to be synchronized in terms of time.

The problem with the internal hardware of the computer system may give the wrong clock time and the clocks have an error in the scattered network is overcome in this system architecture. In this system all the nodes in the network deals with the real time system by the clock synchronization technique, where a network runs at the same time with any kind of clock drift between the nodes [4] -[6]. In the previous work of the clock synchronization there were the problems of clock offset, measuring delays and clock drift or clock skew, these are overcome in this system by using the Network Time Protocol (NTP) algorithm and also supporting the Delay Measurement Based technique by using the correct software. The local clock is to have to correspond to reference clocks [7], [8] so that the local clocks need to be adjusted by some amount that is known as clock offset. The delay measurement technique provides the capability to the client to launch a clock time to arrive at the reference clock at a particular time; therefore it gives a measure of the transit time to the particular time server of the clock time.

The main aim of the system is to manage different or multiple unrelated processes running on the different machines in the scattered environment and also to make consistent decisions about the ordering of events in the system. In this work, without paying attention to what the network latencies are among the machines or how imprecise their clocks may be, the system ensures that all machines reports at the same time. In this work to accurately synchronize the clocks the Network Time Protocol (NTP) algorithm is used, which is a networking protocol for clock synchronization between computer systems over packet-switched, variable-latency data networks. NTP provides Coordinated Universal Time (UTC) including scheduled leap second adjustments [7].
The rest of the paper is organized as follows. Section II introduces the related work, and the important notations are summarized in Sec. III. In Sec. IV, the clock synchronization model are discussed and formulated based on the measurement results. The Delay Tolerant Scattered Algorithm (DTSA) is given in Sec. V. The performance evaluation of DTSA compared to existing algorithms is discussed in Sec. VI, followed by conclusion in Sec. VII.

2. RELATED WORK
Many different schemes have been proposed that seek to achieve timing synchronization in networks. These methods can be divided into approaches that take advantage of synchronization sources within the network. The pros and cons of each scheme are briefly summarized below:

In terms of DTNs, there have been some efforts for clock synchronization. The Timestamp Transformation Protocol (TTP) [7] solves the temporal ordering problem in sparse adhoc networks. The protocol does not synchronize locks, but

transforms message timestamps at each node to its local timestamp with some error bound as a message moves from hop to hop. Simulation results show that the clock inaccuracy increases linearly with time and the number of hops.

The Double-pairwise Time Protocol (DTP) [6]-[9] provides time synchronization in DTNs with a modified NTP. The DTP achieves a clock estimation error lower than the NTP by explicitly estimating the relative clock frequency using back-to-back messages with a controllable interval in between. However, the DTP is a reference node clock synchronization that assumes at least one time server in the network, and it does not work in a distributed environment where there is no reference node to spread the correct reference clock information [5].

The Asynchronous Diffusion (AD) protocol [7] provides distributed clock synchronization by asynchronously averaging clock values with the contacted neighbors. However, the AD is inefficient in DTNs where the connection is dynamic and often limited to just a few neighbors.

The Network Time Protocol (NTP) [2] has been widely used to synchronize computer clocks in the Internet. The NTP enables synchronization between the hierarchically arranged servers and clients. The clocks of servers are adjusted by trusted time references [7], [8]. However, the NTP is intended for connecting Internet where the synchronization operation can be conducted between the reference node and clients continuously and frequently. One drawback of NTP is its relatively moderate achievable accuracy in the range of 1 ms PTP shows a good long term stability and is deterministic and convenient to use [9], [10]. The simplicity of the approach is however also a problem, since the master node is failure whole network fail.

For achieving clock synchronization between the nodes, clock drift is a major problem as the delay increased, synchronization could not be accurate, since to measure this delay several delay measurement algorithms Next Generation Network Algorithm (NGN) transfer data in a form of packet on a network that reduces delays in transformation [3], [7] , [11].

Clocks may run at different speeds and drift apart to report different times. Two events on two different systems may actually occur at exactly the same time and thus be tagged with identical timestamps, there was no algorithm which compares messages to pick one over another and rely on them. Reference Broadcast Synchronization (RBS) [12] broadcast packets that were received by several receivers who then compare the times, senders never timestamp a packet, but they include a packet identifier which receivers use to refer to packets when comparing the receipt times. Since less information is available, the estimation of all unknown parameters skews, offsets, and delays remains impossible. Internet of Things [13] - [16]; synchronize the clock servers by the Internet. The clock synchronization system had many levels and was self-contained, so easily organized, managed and controlled, and it improved the security of clock synchronization. It had good adaptability, so that network transformation was avoided. Depends on the clock skew adjustment. Clock skew was measured by inter-packet delays (IPDs) [9], [13] it was the difference in arrival times between consecutive packets. The TTEthernet clock synchronization algorithm is inherently fault-tolerant. However, the synchronization quality decreases with the number of faulty components. The algorithm detects faulty TTEthernet devices, in particular faulty compression master (CM), and remove them, the diagnosis algorithm improves the precision in the system [17], [18].

3. IMPORTANCE OF CLOCK SYNCHRONIZATION

There is no specific clock synchronization scheme available to achieve a greater scalability with the higher order of accuracy of protocols techniques for the synchronization of the local clocks of nodes in a scattered environment. The Accurate Clock Synchronization in Scattered Network while communicating on the network is a very important task because there are some time-based computations on multiple machines in which some applications that measure elapsed time, agreeing on deadlines and also many real time processes may need accurate timestamps [18]. To synchronize the clocks accurately it is very important to overcome the problem of clock offset, clock drift and delay in the transaction while communicating on the scattered network. This work of clock synchronization is needed because it can measure delays between scattered components in the network during the transaction. It is also useful to establish the ordering of the events which are either casual ordering of events or concurrent or overlapping execution of events where no causal relationship between the events in the scattered network [13]. Clock synchronization is needed for accurate timestamps to authenticate or identify business transactions, serializability in scattered databases.

The goal of the work is to synchronize the clocks which are Logical Clock on the scattered communicating network by both Global and Local time synchronization. Global time synchronization is used to assigning the timestamp to a globally sensed event and the Local time synchronization is used to precisely the event localization [19], [20]. The logical clock synchronization in which computers are synchronized based on the relative ordering of events were in the physical clock synchronization, the actual time on computers are synchronized.

3.1 Clock Drift or Clock Skew

In the scattered communicating network every node having each clock has a value that differs from real time (International Atomic Time (TAI) is being the best representation); no clock is perfect [21]. A quartz-based electronic clock maintains a consistent inaccuracy, but keeps imperfect time. This base 'inaccuracy' is known as 'Clock Skew' or 'Clock Drift'. The Clock Drift is the difference in reading between a clock and a nominal perfect reference clock per unit of time of the reference Clock. In this work of the accurate clock synchronization in scattered network also overcome the problem of clock drift or clock skew. All the computer systems in the network are generally synchronized to a standard time called Coordinated Universal Time (UTC) where UTC is the primary time standard by which the world regulates time and clocks [21]-[24]. The clock supports basically three types of clocks that are-

1. Perfect clock:

The timer ticks ‘H’ interrupts a second.

$$dc/dt = 1$$

2. Fast clock:

The timer ticks more than ‘H’ interrupts a second.

$$dc/dt > 1$$

3. Slow clock:

The timer ticks less than ‘H’ interrupts a second.

dc/dt < 1

Where dC is the number of seconds counted by the software clock and dt is every UTC second.

The Clock Drift is very small, but adds up over time. For clock time, typical drift rate is about one second every 10^6 seconds=11.6days. Best atomic clocks have drift rate of one second in 10^{13} seconds = 300,000 years [12], [17]

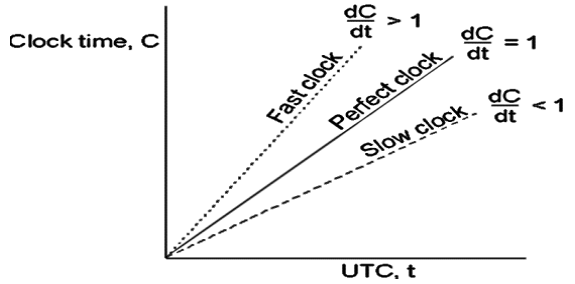

Fig 1: Types of Clocks for Clock Drift

For the fast clock the Clock drift or clock skew is greater than zero, for the perfect clock it is exactly equal to zero, whereas for the slower clocks it is less than 1.

3.2 Delay Measurement Based Techniques

The Delay Measurement technique provides accurate clock synchronization in the scattered network for measuring the delays in transmission time of token and this transmission time is assumed to be constant. The variation in clock time transmission delay introduces an error in the clock offset. In this system this measured data helps in decision making in fault detection in the network. After removing such a clock skew or clock drift, the resulting delay is then used as a true measurement for clock synchronization of the scattered network [25], [26].

The problem of temporal ordering in the network solves by using the Timestamp Transmission Protocol (TTP) [4], [23]. It does not synchronize the clocks but the clock time timestamp transforms at each node in the network to its local timestamp with the some error bounds as a clock time moves from node to node. To avoid the network transformation, the clock synchronization architecture for the internet has good adaptability. It facilitates clock synchroni- zation system management and organization, and also ensures the accuracy of clock synchronization in the network [8], [24] - [26].

3.3 Network Time Protocol (NTP)

The Network Time Protocol (NTP) is used to synchronize the clocks of the computer client or server to another server or reference time source present in the scattered network. It provides accuracies typically within a millisecond on LANs and up to a few tens of milliseconds on WANs relative to Coordinated Universal Time (UTC) [27]. NTP is used to maintain and construct a set of clock servers and transmission paths as a synchronization subnet in the scattered network. NTP supports multiple time servers across the Internet like primary servers, secondary servers, and tertiary servers etc., the primary servers are directly connected to UTC receivers and secondary servers synchronized with the primary servers whereas the tertiary servers synchronized with secondary etc. It Scales up to large number of servers and clients [28], [29].

NTP provides a connectionless transport mechanism and data integrity because it builds on the User Datagram Protocol (UDP) and Internet Protocol (IP) which, so that it is readily adaptable to other protocol suites. There need some implementations in the NTP itself for including the features of acquisition, node discovery in the network or authentication [30] - [33].

4. IMPLEMENTATION MODEL

4.1 System Design

In the system architecture of Clock Synchronization in Scattered Network clock synchronization is done in the nodes while communicating with the other nodes in the network of either LAN, WAN or a combination of both the networks. Accurate time synchronization in a diverse and large internet system [34], it describes the NTP designed. The architecture of the system in which scattered subnet of clock servers operates in hierarchical, self- organizing, master-slave configuration synchronizes local clocks within the distribute subnet and to national time standards via LAN or WAN.

The redistribution of time may also possibly by servers within a network via local routing algorithms and time daemons [35]. It has the features of platform independence, works in various networking environments, accurate and reliable clock synchronization and also by using the NTP algorithm it supports both Windows and Linux operating system to overcome the drawback of platform dependency in scattered network [36] –[39].

The system of accurate clock synchronization having the implementation process model for the clock server host which includes the three different processes for the synchronization of the clocks that are as follows:

a) Receive process
b) Update procedure for local clock process
c) Transmit process in the scattered network shown in Fig. 2.

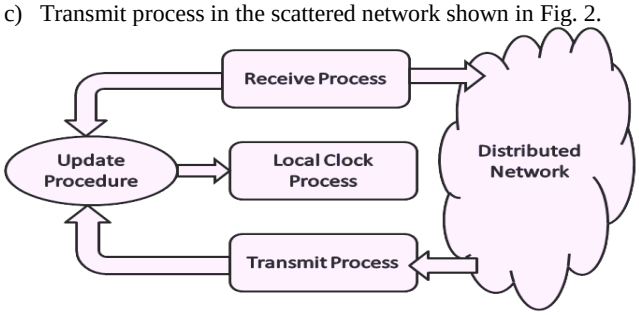

Fig 2: Implementation Process Model

4.2 System Architecture

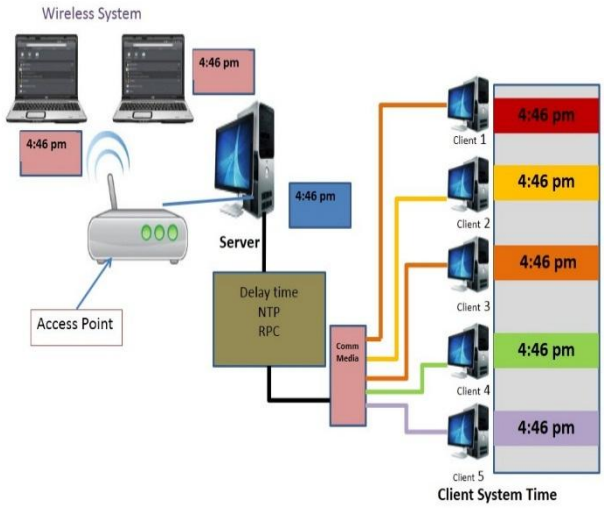

Fig 3: Clock Synchronization in Scattered Network (CSISN) Architecture

The system time of a node may change, so some interval of time the system clock must be checked and modify same as server time for synchronization. The deviation between synchronized clocks immediately before the resynchronization instant (drift-offset) is a linear function of the drift rates of the clocks and the duration of the synchronization interval. If the drift offset of a particular clock move outside a given precision interval, the clock is considered faulty. Since low-quality clocks have a wide drift rate margin and poor long term stability, the precision of the global time base is reduced if low quality oscillators are deployed. Furthermore there is a higher probability that the clocks have to be replaced because they drift, over time, outside the specified drift rate interval due to their poor long term stability.

One solution to this problem is to continuously adapt the drift rate of all clocks to minimize the drift offset. In order to guarantee a reliable global time base of good accuracy and precision, the stability of such a scattered drift rate adaptation algorithm must be investigated. It must be shown that the scattered clocking system is free of unintended long-term oscillations and that it does not drift away from the rate of the external physical time. A proposed architecture is a combination of a delay tolerant scattered algorithm for clock state synchronization with an NTP protocol for clock rate adaptation. This combined algorithm tolerates arbitrary failures of a specified number of clocks in the clock state synchronization and a specified number of failing silent failures in the clock rate synchronization.

4.3 Principle of Operation

Let us assume that the local server is connected to a global server (GS) time receiver. The time server periodically broadcasts clock times containing a synchronization event and information to be able to place this synchronization event. The time server has to synchronize the global time of its node with the time received from the GS. This synchronization is unidirectional and thus asymmetric, as shown in figure 4.

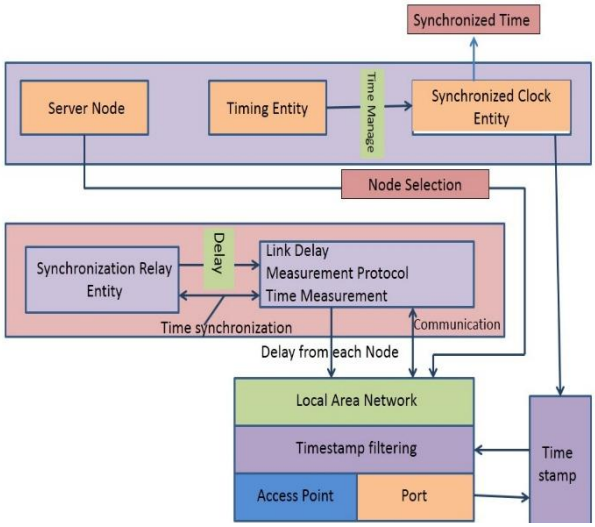

Fig 4: Workflow of CSISN

The processing is as follows.

1. The local time server node sends the request to the GS.
2. The request is sent to collect information in the database and sends the NTP clock times to the requested nodes in the network.
3. The clock time is sent, each clock time contains the local timestamp, it is necessary to determine the hierarchy and manage the association together with previously received timestamp and other information. The receive process receives clock times and possibly the clock times in other protocols and also the information from directly connected time code receivers.
4. The offset between the node clock and the local clock is computed and incorporated into the data base along with other information useful for error estimation by discarding the inferior data and node selection, when clock time is received.
5. The next step is to update received clock time and at other times by the clock server in the network.
6. The update procedure processes the offset data from each and every node and then selects the best one node from the network.
7. The local clocks process used and operates the offset data which are produced by the previous update procedure and adjust the timing of the local clock.
8. Compute the time offset of the source node for synchronization.
9. Forward the synchronized time to the connected node.

If another node is connected to this primary network then the unidirectional synchronization functions in the same manner. The other node considers the synchronized time of the primary node as the time reference and synchronizes the secondary node.

Whereas internal synchronization is a cooperative activity among all members of a network, external synchronization is an authoritarian process: the time-server forces its view of external time on all its subordinates. From the point of view of fault tolerance, such an authoritarian regime has a problem: If the authority sends an incorrect clock time all its obedient subordinates will behave incorrectly. However, in the case of external clock synchronization the situation is under control because of the "inertia" of time. Once a node has been externally synchronized, the fault-tolerant global time within a network acts as a monitor of the time server. A time gateway

will only accept an external synchronization clock time if its content is sufficiently close to its view of the external time. The time server has only a limited authority to correct the clock time.

5. DELAY TOLERANT SCATTERED ALGORITHM (DTSA)

Algorithm:

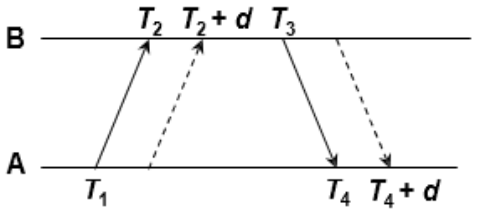

Fig 5: Clock offset in the Communication Channel

In order to compensate for the offset and drift, and to synchronize all the nodes on request-path to the reference node, the DTSA algorithm works as follows.

Step 1: Send ntp time request from LTS to GTS.

Step 2: Waiting for the response from GTS.
If GTS respond with update time
then
Compute its clock with a remote server, as
$\delta = (T2 - T1) + (T4 - T3)$
where
T1 is the client's timestamp of the request packet transmission,
T2 is the server's timestamp of the request packet reception,
T3 is the server's timestamp of the response packet transmission and
T4 is the client's timestamp of the response packet reception.
therefore
T3 – T1 is the time elapsed on the client side between the emission of the request packet and the reception of the response packet and T4 – T2 is the time the server waited before sending the answer
LTS synchronized
else
Resend request
end if

Step 3: A LTS waiting for request.

Step 4: If a DTSA-REQ message is received, it notes the receive time T2. If it is a reference node or a synchronized node, it replies with a DTSA-REP message containing timestamps T1, T2, T3, and Tr along with other information. However, an unsynchronized intermediate node recursively forwards the request to the reference node after saving ID of its client in the server, and the values of T1 and T2 in T1old and T2old, respectively.

Step 5: if DTSA-REP message is received, it calculates the value of propagation delay by following equation.

$$d = \frac{(T2 - T1) + (T4 - T3)}{2} \quad (1)$$

end if

Step 6: Then, it finds the new value of Tr by adding d to the received value of Tr as given by (2).

$$Tr = Tr + d. \quad (2)$$

Step 7: It records the global and local timestamps where global=Tr and local=T4. Moreover, if node is an intermediate node, it retrieves the values of T1 and T2, calculates the value of Tr by adding to it the elapsed time T4, and then forwards the reply to its client node.

$$Tr = Tr + (T3 - T4). \quad (3)$$

end if

Local time from the global time as follows:

a) **When the skew is far away from 1:** A node is considered to be perfectly synchronized when the current value of skew or drift is 1. Therefore, the value of the skew can tell a node if it should initiate a time synchronization request or not. Accordingly, if the current value of skew is much lesser or greater than 1, the node will request timing information by sending a DTSA-REQ message to the reference node; otherwise, it will not request an update.

b) **When the local time is different from global:** Although an overheard message is not yet compensated for offset, skew, or propagation delay; however, it can provide a rough idea of the global time. On receiving DTSA-REP message, a node can check if its local time is almost same as the global timestamp in Tr field. If the two times are much different, depending on the required accuracy, the node will initiate a time synchronization request; otherwise, it will not request for any update.

Such an adaptive re-synchronization interval helps reduce the overhead of unnecessary time synchronization requests in the long run. It also provides for the long-term synchronization in a self-configuring way. In addition to this, the DTSA algorithm (Algorithm 1) performs a few other functions, which include restarting the process for election of the reference node on detecting a failure, broadcasting messages periodically by an active reference node, and calculating the values for updating the local clock.

5.1 CSISN Model works with the following assumptions:

Following assumptions, which are realistic and commonly found in literature, are made in the proposed algorithm:

i. Sensor nodes are uniquely identified by their numeric IP.
ii. Time-stamping of messages is possible for each node.
iii. Neighboring nodes can communicate over wireless and LAN channel.
iv. Broadcasting of messages is possible in the network.
v. Skew and connectivity do not change during the short interval between synchronization request and reply.
vi. Propagation delay in one direction is exactly equal to the other.
vii. A simple linear relationship exists between the clocks of two sensor nodes in a short duration.

5.2 Working of DTSA algorithm

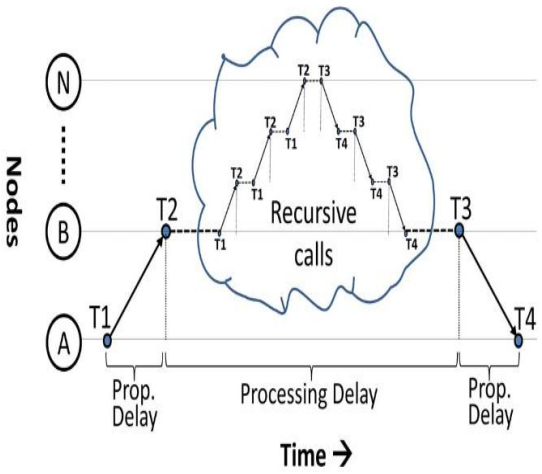

Fig 6: Request-and-reply mechanism in DTSA

The uncertainties in clock time delivery include the following time delays: send, access, transmission, propagation, reception, receive and the byte alignment time [5], [6], [16]. Although the DTSA algorithm uses request-and-reply mechanism similar to TPSN but in a recursive manner, it uses time-stamping of messages like FTSP that removes all the sources of uncertainties in message delivery delays except the propagation delay [6].

Consider the request-and-reply mechanism of DTSA in Fig. 6 where node A sends a request at T1 to another node B that receives it at time T2 according to their local clocks. However, the real time measured by an ideal clock at these nodes is denoted by t1 and t2, respectively. The following equation is:

$$t_{2=}\, t_{1+}\, S_{A+}\, P_{A\to B\,+}\, R_B \qquad (4)$$

Where S_A is the time taken to send message (send + access + transmission time) at node A, R_B is the time taken to receive the message (reception + receive time) at node B, and $P_{A\to B}$ is the propagation time between nodes A and B.

As the time-stamping of messages removes all the sources of uncertainties except the propagation delay can be written as follows, for DTSA:

$$t_{2=}\, t_{1+}\, P_{A\to B} \qquad (5)$$

The corresponding equation in terms of local clocks, which involves clock drifts as well, can be written as follows:

$$t_{2=}\, t_{1+}\, P_{A\to B\,+} D_{t1}^{A\to B} \qquad (6)$$

Now, the relative drift between the two nodes from t1 to t4, the difference of their respective drifts, can be more or less depending on the value of two drifts, and is given by the following equation:

$$RD_{t1\to t4}^{A\to B} = D_{t1}^{A\to B} - D_{t4}^{A\to B} \qquad (8)$$

From (8), we can get the value of $D_{t1}^{A\to B}$ and put in (9) to get the following equation:

$$t_{2=}\, t_{1+}\, P_{A\to B}\; D_{t4}^{A\to B} + RD_{t1\to t4}^{A\to B} \qquad (9)$$

After receiving the request message at time T2, node B forwards this request to another node in a recursive manner until it reaches the node N, which replies with timing information that is forwarded to node B through the same path. The node B at time T3 forwards this reply to node A which receives it at time T4 according to their local clocks. Using the analysis

Similar to (6), the following equation can be derived:

$$T_4 = T_3 + P_{B\to A} + D_{t3}^{B\to A} \qquad (10)$$

Note that by subtracting (9) and (10) gives the following equation

$$T_4 = T_3 + P_{B\to A} - D_{t4}^{A\to B} \qquad (11)$$

Subtracting (11) from (10) gives the following equation:

$$T_2 - T_4 = T_1 + P_{B\to A} - D_{t4}^{A\to B} + RD_{t4}^{A\to B} - T_3 - P_{B\to A} + D_{t4}^{A\to B} \qquad (12)$$

Rearranging (12), and substituting$(P_{A\to B} - P_{B\to A})$ with P^{UC} gives (13), where P^{UC} represents the uncertainty in propagation time

$$(T_2 - T_1) - (T_4 - T_3) = P^{UC} + RD_{t1\to t4}^{A\to B} + 2D_{t4}^{A\to B} \qquad (14)$$

As we know that the offset $T = ((T_2 - T_1) - (T_4 - T_3))\,/2$ therefore (14) can be written as follows:

$$T_f = P^{UC} + RD_{t1\to t4}^{A\to B} + 2D_{t4}^{A\to B} \qquad (15)$$

To correct the clock at T4 at node A, subtract the value of $D_{t4}^{A\to B}$ from (16), which results in the following equation:

$$Error = T_f - D_{t4}^{A\to B} = \frac{P^{UC}}{2} + \frac{RD_{t1\to t4}^{A\to B}}{2} \qquad (16)$$

(4)

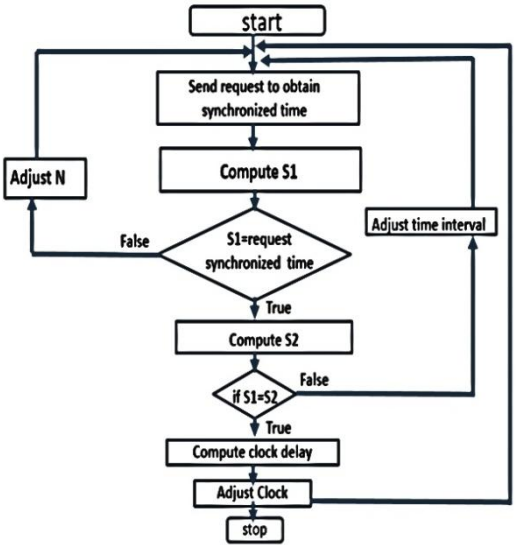

Fig 7: Operational flow chart

The implementation must guarantee that in no case it is possible for faulty external synchronization to interfere with the proper operation, i.e., with the global time, within a network. The worst possible failure scenario in case the external time server is failing maliciously is a common mode

deviation of the global time from the external time with the maximum correction rate. The internal synchronization within a network will not be affected by this controlled drift from the external time [30] - [33].

6. PERFORMANCE ANALYSIS

This section describes the components used for our measurements and the experimental scenarios that perform with the various measuring instruments. Each node is equipped with the above-mentioned features of a CSISN node, i.e., a clock synchronization unit (CSU).

6.1 Measuring Instruments

The following quantities have to be measured:

a) The delay of the network node.
b) Delay of the clock ticks delivered by the CSU.
c) Time difference between corresponding ticks of two synchronous CSUs.
d) The delay between sending a clock time at one node and receiving clock time by another node.

6.2 Experimental Results

This section contains the experimental results of our measurements. Different scenarios are evaluated to reason about the behaviour of the physical and the logical computer clocks.

Table 1. Shows comparative clock delay in network. Based on NTP some algorithm are compared like User Datagram Protocol (UDP),Internet control Message Protocol (ICMP), Reference Broadcast Synchronization (RBS), Time Synchronization Protocol for Sensor Network (TPSN), Precision Time Protocol (PTP), Recursive Time Synchronization Protocol (RTSP) and Flooding Time Synchronization Protocol (FTSP) and from the comparison shows in Table. 1 it observed that the delay in network using DTSA is less i.e. 15ms, so that accuracy in network is more. Similarly.

Evaluation

Table 1: Comparison of DTSA with other Algorithm

Parameter	UDP	ICMP	RBS	TPSN	PTP	FTSP	RTSP	DTSA
Root Delay (sender to receiver)	493 ms	364 ms	29.17 ms	20 ms	360 ms	18 ms	32 ms	15 ms

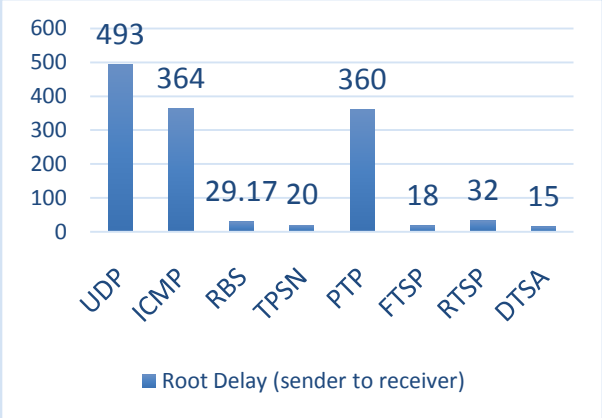

Fig 8: Root delay in network

From the analysis it is clear that, some of the algorithm are lacking to deal with scattered network. On the other hand, DTSA provide better behaviors, and in particular, our solution clearly achieves the best performance in scattered network. This confirms the effectiveness of the implemented algorithm.

7. CONCLUSION

The algorithm presented in this paper synchronizes LTS with GTS to provide synchronized time to scattered network. An analysis of the sources of errors shows that the two sources of errors are variation in propagation delays and relative drift between local clocks, which are duly compensated by the algorithm. The accuracy of DTSA is improved by using NTP and DTA. Further improvement in accuracy is gained by the compensation of propagation delay and adjustment of the clock time. Analytical and results demonstrate that DTSA algorithm achieves more accuracy and reduces delay than existing clock synchronization protocols under various network conditions. The DTSA algorithm can provide more accurate timing information in data collection from a distributed environment.

8. REFERENCES

[1] Na Wang, Haihui He, "Time Synchronization for Failure Tolerance in Wireless Sensor Network" 13th ACIS International Conference on Software Engineering, pp. 181-184, IEEE computer society 2012.

[2] Zexin Jiang," A Novel Simulation Time and Wall Clock Time Synchronization Algorithm for HLA", pp.257-260, IEEE computer society 2012.

[3] Yi Wan and Zhongping Chen," A Novel Synchronization Method for DS-CDMA Systems", pp. 596-602, IEEE 2012.

[4] Saverio Bolognani, Ruggero Carli, Enrico Lovisari and Sandro Zampieri, "A randomized linear algorithm for clock synchronization in multi–agent systems", pp.20-27, 51st IEEE Conference in USA 2012.

[5] Andre´ C. Pinho, Daniel R. Figueiredo and Felipe M. G. Franc," A Robust Gradient Clock Synchronization Algorithm for Wireless Sensor Networks", IEEE 2012.

[6] Vangelis Gazis, Eleni Patouni, Nancy Alonistioti and Lazaros Merakos," A Survey of Dynamically Adaptable Protocol Stacks",vol.12, no.1,pp.3-23, IEEE 2010.

[7] David M. E. Ingram, Pascal Schaub, and Duncan A. Campbell," Use of Precision Time Protocol to Synchronize Sampled-Value Process Buses", , vol. 61, no. 5,pp.1173-1180, IEEE Transactions , May 2012.

[8] Reinhard Exel," Clock Synchronization in IEEE 802.11 Wireless LANs using Physical Layer Timestamps", IEEE 2012

[9] Xiali Li,Min Xi, Yongcun Cao and Yong Lu," Clock Synchronization Using Expectation-Maximization Algorithm in Wireless Sensor Network",pp.1166-1169, International Conference IEEE 2012.

[10] Ruggero Carli, Giada Giorgi, Claudio Narduzzi," Comparative analysis of synchronization strategies in sensor network with misbehaving clocks", IEEE 2012.

[11] Michael Kevin Maggs, Steven G. O'Keefe, and David Victor Thiel," Consensus Clock Synchronization for Wireless Sensor Networks", vol. 12, no. 6, pp. 2269-2277, IEEE Journal June 2012.

[12] Bong Jun Choi, Hao Liang, Xuemin (Sherman) Shen, Fellow and Weihua Zhuang," DCS: Distributed Asynchronous Clock Synchronization in Delay Tolerant Networks", vol. 23,no. 3, pp. 491-504, IEEE TRANSACTIONS 2012.

[13] Xiong Xu, Zhenhua Xiong, Xinjun Sheng, Jianhua Wu, and Xiangyang Zhu," A New Time Synchronization Method for Reducing Quantization Error Accumulation over Real-Time Networks: Theory and Experiments", IEEE 2013.

[14] Halgurd S. Maghdid and Ihsan Alshahib Lami," Dynamic Clock-Model of Wi-Fi Access-Points to help Indoors Localisation of Smartphones",pp.268-273, IEEE 2012.

[15] Ming-Feng Wu and Chih-Yu Wen," Distributed Cooperative Sensing Scheme for Wireless Sleep EEG Measurement", , vol. 12, no. 6,pp.2035 -2047, IEEE Sensors Journal, June 2012.

[16] Zhe Yang, Lin Cai, Yu Liu and Jianping Pan," Environment-Aware Clock Skew Estimation and Synchronization for Wireless Sensor Networks",pp.1017-1025,IEEE Proceedings 2012.

[17] J. Quesada, J. Uriarte Llano, R. Sebastián, M. Castro and E. Jacob," Evaluation of Clock Synchronization Methods for Measurement and Control using Embedded Linux SBCs",IEEE 2012.

[18] Young Kyu Lee, Sung Hoon Yang, Taeg Yong Kwon and Chang Bok Lee," Evaluation of Synchronization Performance with PTP", pp.624-625, IEEE 2012.

[19] Peter Philipp and Simon Altmannshofer," Experimental Validation of a New Moving Horizon Estimator Approach for Networked Control Systems with Unsynchronized Clocks",pp.4939-4948, IEEE Conf. June 27-June 29 2012.

[20] Slavko S ajic, Nebojsa Maletic, Branislav M. Todorovic and Milan S unjevaric," Frequency Hopping Synchronization Scheme Based on Real-Time Clock", pp.293-297,IEEE 19th International Conference 2012.

[21] Jie Zhang, Jie Wu, Zhao Han, Liefeng Liu and Kaiyun Tian," Low Power, Accurate Time Synchronization MAC Protocol for Real-time Wireless Data Acquisition", IEEE 2012.

[22] Andrea Bondavalli, Francesco Brancati, Alessandra Flammini, and Stefano Rinaldi," Master Failure Detection Protocol in Internal Synchronization Environment", IEEE Trans.,vol. 62, no. 1,pp.4-12, January 2013.

[23] Daniele Fontanelli, David Macii," Master–less Time Synchronization for Wireless Sensor Networks with Generic Topology", IEEE 2012.

[24] Cong Liu and Jie Wu," On Multicopy Opportunistic Forwarding Protocols in Nondeterministic Delay Tolerant Networks" IEEE Trans. on Parallel and Distributed Systems,vol. 23, no.6, pp. 1121-1128 June 2012.

[25] Tamas Kovacshazy and Balint Ferencz," Performance Evaluation of PTPd, a IEEE 1588 implementation, on the x86 Linux platform for Typical Application Scenarios", IEEE 2012.

[26] Muhammad Akhlaq and Tarek R. Sheltami," RTSP: An Accurate and Energy-Efficient Protocol for Clock Synchronization in WSNs", IEEE Trans., vol. 62, no. 3,pp. 578-591, March 2013.

[27] Michal Pravda, Jiri Vodrazka, and Pavel Lafata," Simulations and Measurements of Packet Network Synchronization by Precision Time Protocol", pp.116-120, IEEE 2012.

[28] Tal Mizrahi Marvell Yokneam," Slave Diversity: Using Multiple Paths to Improve the Accuracy of Clock Synchronization Protocols", IEEE 2012.

[29] Petar Djukic and Prasant Mohapatra," Soft-TDMAC: A Software-Based 802.11 Overlay TDMA MAC with Microsecond Synchronization" IEEE Trans.,vol. 11, no. 3,pp.478-494, March 2012.

[30] Doudou Messaoud, Djenouri Djamel and Badache Nadjib," Survey on Latency Issues of Asynchronous MAC Protocols in Delay-Sensitive Wireless Sensor Networks" , pp.1 -23, IEEE 2012

[31] M. Akhlaq and Tarek R. Sheltami," The Recursive Time Synchronization Protocol for Wireless Sensor Networks", IEEE 2012.

[32] Jaeshin Jang, Sang Wu Kim, and Sunghong Wie," Throughput and Delay Analysis of a Reliable Cooperative MAC Protocol in Ad Hoc Networks", vol. 14,no. 5,pp.524-532,IEEE October 2012.

[33] Xiong Xu,Zhenhua Xiong,Xinjun Sheng,Jianhua Wu,and Xiangyang Zhu," A New Time Synchronization Method for Reducing Quantization Error Accumulation over Real-Time Networks: Theory and Experiments", IEEE 2013

[34] Bong Jun Choi, Halo Liang, Xuemin Shen, Fellow, and Weihua Zhuang,"Scattered Asynchronous Clock Synchronization in Delay Tolerant Networks", IEEE Trans. on parallel and scattered systems,vol.23, no.3, pp. 491-504, March 2012.

[35] Hao Chen, Lin Shi, Jianhua Sun, Kenli Li and Ligang He, "A Fast (Remote Procedure Call) RPC system for Virtual Machines", IEEE Trans. on parallel and scattered systems, pp. 1-11, 2012.

[36] Hani Mehrpouyan, Steven D. Blasting, Tommy Svensson, "New Scattered Approach for Achieving Clock Synchronization in Heterogeneous Networks", IEEE Globecom, 2011.

[37] Stefano Bregni, Ravi Subrahmanyan, "Synchronization over Ethernet and IP in Next-Generation Networks". IEEE Communication Magazine, pp. 130-131, Feb 2011.

[38] Wilfried Steiner, Bruno Dutertre, "Layered Diagnosis and Clock-Rate Correction for the TTEthernet Clock Synchronization Protocol", pp. 244-253, IEEE 2011.

[39] Nikolaos M. Freris, Scott R. Graham and P. R. Kumar, "Fundamental Limits on Synchronizing Clocks Over Networks", IEEE Trans.,vol. 56, no. 6, pp. 1352-1364, June 2011.