IMPACT OF LOW ENERGY TIME SYNCHRONIZATION METHODS IN WIRELESS SENSOR NETWORKS

Kamlesh Kumar Gautam, Surendra Kumar Gautam, Dr. P.C. Agrawal

Abstract-In the current time a large no of Time Synchronization methods are used. Each method has own characteristics and Synchronization precision. Each method has its own feature and characteristics. Each method has advantages and disadvantages. Each method has its own limitations, working environmental conditions and provided its results. Some of the methods provide accurate results and some of the time synchronization methods provide approximate results. Time Synchronization methods have large series like as RBS, Post Facto synchronization, TPSN, LTS, External vs. Internal, on demand, GPS, offset, subset synchronization, DMTS ,FTSP, RATS, GRADIENT TIME SYNCHRONIZATION PROTOCOL, PLTS etc. These time synchronization methods Precision, Lifetime, Scope and Availability, Efficiency, Cost and Form Factor. DMTS, RATS and FTSP synchronization precesson or errors have shown in μs .

Keywords: DMTS, FTSP, RATS, GPS, PLTS, TPSN, LTS

1 INTRODUCTION

A large no of Time Synchronization methods are used for wireless sensor networks. In this paper 21 Times Synchronization methods are discussed and studied.These method are responsible for calculating pressure , time, energy, cost, humidity for environmental monitoring, land slide detection, to synchronize the clock, monitoring can be done with help of wireless sensor nodes. Each method has own characteristics and Synchronization precision. Each method has its own feature and characteristics. Each method has advantages and disadvantages. Each method has its own limitations, working environmental conditions and provided its results. Some of the methods provide accurate results and some of the time synchronization methods provide approximate results. Time Synchronization methods have large series like as RBS, Pre Facto synchronization, Post Facto synchronization, On Demand Time Synchronization, TPSN, LTS, External vs. Internal, on demand, GPS, offset, subset synchronization, DMTS ,FTSP, RATS, GRADIENT TIME SYNCHRONIZATION PROTOCOL, PLTS, PR etc. These time synchronization methods Precision, Lifetime, Scope and Availability, Efficiency, Cost and Form Factor. using time synchronization methods environmental tempreture also forcasted with the help of installed large no of sensor nodes.

1.1 Time Synchronization

To synchronize frequency means to adjust the clocks to run at the same frequency and to synchronize time means to set their time at a particular epoch to be exactly the same [15].
Time synchronization is a critical piece of infrastructure for any distributed system.
WSN have unique requirements in the scope, lifetime, and precision of the synchronization achieved.

- WSN makes extensive use of synchronized time: proximity detection into a velocity estimate
- Existing time sync methods need to be extended to be mindful of time and energy that they consume.
- The heterogeneity of sensor network applications, the need for energy efficiency and the variety of hardware need to be taken in count when creating a new sync method [20].

1.2 Clock synchronization

To synchronize clock means to synchronize the clocks in both frequency and time. In this paper, our algorithm will synchronize clocks, i.e., it will synchronize both frequency and time. Clock synchronization might not be necessary at all times, except during sensor reading integration. In such a case, providing clock synchronization all the time will be a waste on the limited available resources of sensors. The sensor clocks can be allowed to go out of sync, and then re-synchronize only when there is a need for synchronization, thereby saving resources [15].

1.3 Characterizing Time Synchronization

In studying their application to sensor networks, we have found it useful to characterize the different types of time synchronization along various axes. We consider certain metrics to be especially important [4]:

- **Precision**—either the dispersion among a group of peers, or maximum error with respect to an external standard.

- Kamlesh Kumar Gautam Research Scholar (CSE) Mewar University, Chittorgarh (Rajasthan), India , kamleshgautam2003@gmail.com
- Surendra Kumar Gautam) Asst. Prof. Department of Computer Applications, SRM University NCR Campus Modi Nagar,Ghaziabad, India . gautamsurendra@yahoo.com
- Dr. P.C. Agrawal Guest Professor (CSE) Mewar University, Chittorgarh (Rajasthan) and Retired Scientist(Director level) form Ministry of Information Technology, New Delhi

- **Lifetime**—which can range from persistent synchronization that lasts as long as the network operates, to nearly instantaneous (useful, for example, if nodes want to compare the detection time of a single event).
- **Scope and Availability**—the geographic span of nodes that are synchronized, and completeness of coverage within that region.
- **Efficiency**—the time and energy expenditures needed to achieve synchronization.
- **Cost and Form Factor**—which can become particularly important in wireless sensor networks that involve thousands of tiny, disposable sensor nodes [20].

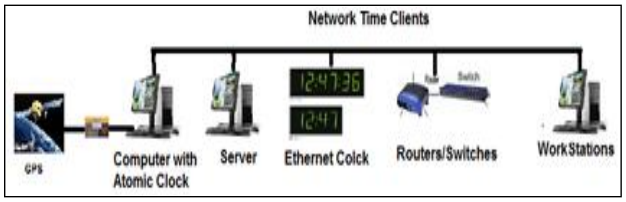

Fig. 1. Network Time client and time synchronization

2 INTERNAL TIME SYNCHRONIZATION

Internal time synchronization is the synchronization of all clocks in the network for master and client, without a predetermined master time. The only goal here is consistency among the network nodes. External synchronization requires consistency within the network *and* with respect to the externally provided system time.
In everyday life, we are mostly faced with external synchronization, namely with keeping wristwatches and clocks in computers, cell phones, PDAs, cars, microwave ovens, etc. synchronized to the legal time[17].

3 EXTERNAL SYNCHRONIZATION

The synchronization of all clocks in the network to a time supplied from outside the network is referred to as *external* synchronization. NTP performs external synchronization, and so do sensor nodes synchronizing their clocks to a master node. Note that it makes no difference whether the source of the common system time is also a
node in the network or not [17].

4 CONTINUOUS TIME SYNCHRONIZATION

- Higher Communication Overheads
- Higher Power Consumption

Continuous Time Synchronization defined a protocol for continuous clock synchronization in wireless sensor networks by extending the IEEE 802.11 standard for wireless local area networks [22].
The cornerstone of the protocol by Mock et al. is the use of *continuous* clock synchronization. This is in contrast with the IEEE 802.11 standard, which uses *instantaneous* clock synchronization. In the case of instantaneous synchronization, a node computes a local clock error and adjusts its clock using this computation. This results in abrupt changes in local clock time, which can cause time discontinuity [22].
Continuous clock synchronization avoids such discrepancies by spreading the correction over a finite interval. The local clock time is corrected by gradually speeding up or slowing down the clock rate. However, this approach suffers from a high run-time overhead since clocks need to be adjusted every clock tick.
Advantages:

- The protocol provides reasonably good accuracy results when message transmission between master and slaves is tight.
- The protocol improves over the IEEE 802.11 protocol by using continuous, rather than instantaneous, clock Updates [22].
- The message complexity of the protocol is quite low because the protocol requires only one message per synchronization round.
- The protocol accounts for potential message losses, a common situation in wireless networks.

Disadvantages:

- The protocol assumes tight communication between the master and the slaves. Consequently, the protocol cannot accommodate multi-hop communication because of its longer and unpredictable delays.
- A considerable amount of energy is spent on performing clock correction. This problem is exacerbated by the fact that clock correction is continuous.
- The computational load on each node is high due to the rate-based correction method.

5 ON-DEMAND SYNCHRONIZATION

On-demand synchronization can be as good as continuous synchronization in terms of synchronization quality, but much more efficient. During the (possibly long) periods of time between events, no synchronization is needed, and communication and hence energy consumption can be kept at a minimum. As the time intervals between successive events become shorter, a break-even point is reached where continuous and on-demand synchronization performs equally well. There are **two kinds** of on-demand synchronization [17]:

5.1 EVENT-TRIGGERED ON-DEMAND SYNCHRONIZATION

(A) *Event-triggered* on-demand synchronization is based on the idea that in order to time-stamp a sensor event, a sensor node needs a synchronized clock only immediately after the event has occurred. It can then compute the time-stamp for the moment in the recent past when the event occurred. Post-facto synchronization is an example for event-triggered synchronization [17]
.

(B) **TIME-TRIGGERED ON-DEMAND SYNCHRONIZATION**
We use *time-triggered* on-demand synchronization if we are interested in obtaining sensor data from multiple sensor nodes for a specific time. This means that there is no event that triggers the sensor nodes, but the nodes have to take a sample at precisely the right time. This can be achieved via *immediate* synchronization [17].

6 GLOBAL TIME SYNCHRONIZATIONIN A SENSOR NODES

Global synchronization in a sensor network [10]**:**

- all-node-based method,
- cluster-based method, and
- fully localized diffusion-based method.

7 RATE VS. OFFSET

Rate vs. offset Sometimes it is sufficient if differences ci(hi(t1))−ci(hi(t2)) (e.g., temperature differences) obtained from different sensor instances can be compared.In this case, rate calibration is sufficient. If, however, absolute values ci(hi(t)) (e.g., absolute temperature values) originating from different sensor nodes are to be compared, offset calibration is needed.

8 SUBSET OF NODES TIME SYNCHRONIZATION

Only subsets of nodes might occasionally need synchronized time with less-than-maximum precision. A subset of the nodes, Event-triggered synchronization can be limited to the collocated subset of nodes which observe the event in question.

9 Post-Facto Synchronization

Post Facto Synchronization Idea [16]:

• It is harmonize energy efficiency with accuracy
• It try to keep nodes turned off most time
• It is multi-modal method
• It is low-power method of synchronizing clocks in local area when accurate timestamps for specific events are needed WINS platform.
• basis of post-facto synchronization
• nodes' radios, CPU powered down
• It has low-power pre processor powered
• If pre processor detects potentially interesting signal
• Pre processor powers on CPU
• Further analysis is required
• If CPU decides that event needs to be reported
• CPU turns on according to node's radio

- Components consuming most energy – powered down most time
- It brings up significant problems:
  How to achieve accurate timestamps when radio and CPU are turned off most time?

9.1 Normally clocks of nodes unsynchronized

• If impulse of event arrives clock of nodes unsynchronized.
• Eeach node records time of impulse with its local time.
• Immediately afterwards: third party node (beacon) broadcasts

9.2 Synchronization pulse to all nodes in area

• Nodes receive pulse, use it as current time reference
• Generally nodes normalize timestamps

9.3 Not applicable in all situations

• limited in scope to transmit range of beacon
• Instant of synchronized time
•It will not practical for sending the timestamps over long distance and time

9.4 Adequate in special situations

• It is used for beam forming applications
• It is used for localization systems
• It is used for comparing relative arrival time of signal to set of detectors

9.5 Single-hop networks

In a single-hop network, one node acts as the sink node and initiates the network wide time synchronization. The remaining ten nodes synchronize themselves to the sink node [12]. The single-hop network topology is shown in Figure. .

A sensor node can directly communicate and exchange messages with any other sensor in the network. However, many wireless sensor network applications span several domains or neighborhoods. (Nodes within a neighborhood can communicate via single hop message transmission.) The network is often too large, making it impossible for each sensor node to directly exchange messages with every other node [13].

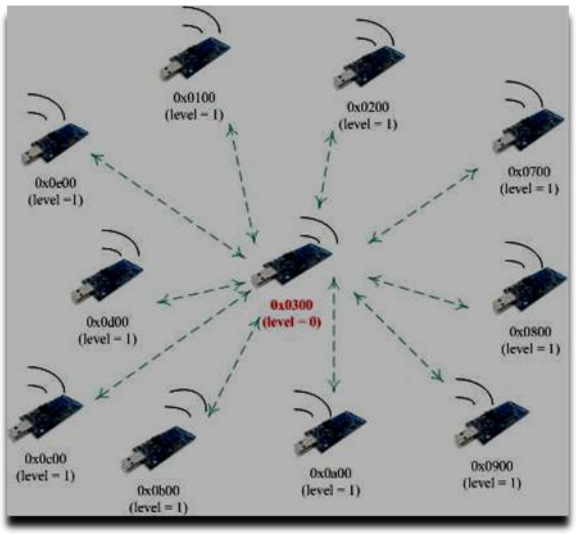

Fig. 2. Single-hop network topology

9.6 Multi-hop networks

To evaluate the synchronization performance in a multi-hop network, we place twelve sensor nodes in a configuration where three monitored nodes are three hops away from the sink, see Figure . We record the results for three nodes in level 3 for 50 rounds of periodical synchronization and analyses the synchronization error between these three nodes and their parent node [12]. The results are shown in Figure

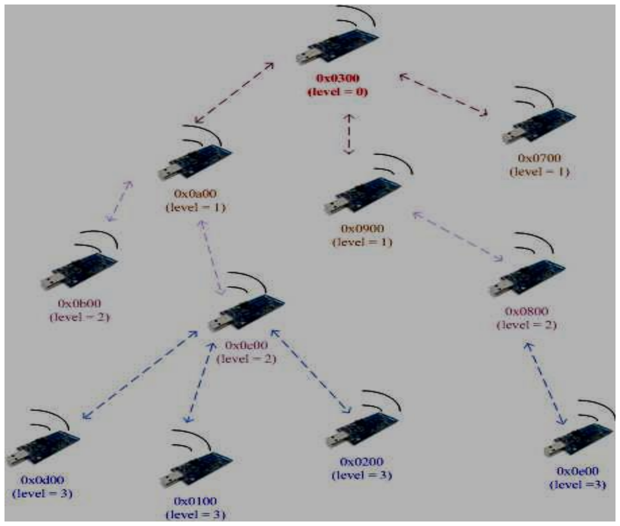

Fig.3 Multi hop network topology

The need for multi-hop communication arises due to the increase in the size of wireless sensor networks. In such settings, sensors in one domain communicate with sensors in another domain via an intermediate sensor that can relate to both domains.**[13]**

9.7 Multi-hop synchronization

RBS offers local synchronization, but it can be extended to provide global synchronization. It can be provided with multi-hop synchronization.

As can be seen in the figure above, X and Y cannot hear each-other, but they are both heard by node 4. Node 4 can relate the clocks in one neighborhood to clocks in the other.

9.8 Multi-hop Communication

Multi-hop synchronization is required in sensor networks that span several node neighborhoods. Evidently, the possibility of multiple hops would introduce a high degree of variability in message transmission time between multiple receivers of the same message [13].

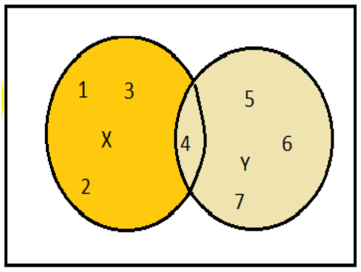

Fig. 4. X topology where multi-hop synchronization is required.

Multi-hop Communication Advantages

- The largest sources of error (send time and access time) are removed from the critical path by decoupling the sender from the receivers.
- Clock offset and skew are estimated independently of each other; in addition, clock correction does not interfere with either estimation because local clocks are never modified.
- Post-facto synchronization prevents energy from being wasted on expensive clock updates.
- Multi-hop support is provided by using nodes belonging to multiple neighborhoods (i.e., broadcasting domains) as gateways.
- This protocol is applicable to both wired and wireless networks.
- Both absolute and relative timescales can be maintained.

Multi-hop Communication Disadvantages

- The protocol is not applicable to point-to-point networks; a broadcasting medium is required.
- For a single-hop network of n nodes, this protocol requires O(n2) message exchanges, which can be computationally expensive in the case of large neighborhoods.
- Convergence time, which is the time required to synchronize the network, can be high due to the large number of message exchanges.
- The reference sender is left unsynchronized in this method. In some sensor networks, if the reference sender needs to be synchronized, it will lead to a significant waste of energy.

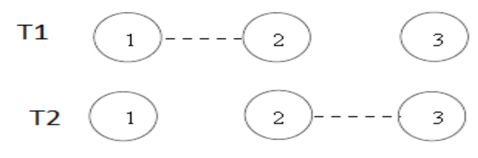

Fig.5. Store and forward Communication

10 PRE-FACTO SYNCHRONIZATION

We might refer to time-triggered synchronization as pre-facto synchronization.

- Usually Time point triggers Synchronization
- Event can occur between any interval of time synchronization. This actually may not be "Pre Facto"[23].

11 REFERENCE BROADCAST SYNCHRONIZATION (RBS)

Reference Broadcast Synchronization introduction

RBS is an time synchronization protocol which instead of estimating the error, exploits the broadcast channel available to remove as much as possible from the critical path [2].

- RBS is used to synchronize a set of receivers with one another.
- Nodes periodically send messages to their neighbors using physical-layer broadcast.
- The message has no timestamp, nor it is important exactly when it is sent.
- The recipients use the message arrival time as a point of reference for comparing their clocks.
- This method removes the sender's non determinism from the critical path and improves accuracy.

Precise network synchronization is hampered by nondeterminism. The estimates for latency are confounded by events that last for an amount of time that we can not determine [2].

RBS does not readjust the clock. Also it does not consider the nondeterministic errors such as send time delay and access time delay, since broadcasting creates the same amount of these errors for all nodes. The most important disadvantage of RBS is that it is not scalable. In other words, by increasing the number of nodes, synchronization accuracy falls and the number of exchanged messages for synchronization increases [5]. Nondeterministic transmission delays are detrimental to the accuracy of a synchronization protocol because they make it difficult for a receiver to estimate the time at which a message was sent and vice versa. In general, the time involved in sending a message from a sender to a receiver is the result of the following four factors, all of which can vary non deterministically [13].

If we decompose the sources of message latency, we would have these components [11], [10]:

- *Send time*. This is the time spent at the sender to construct the message and to transfer it to the network interface.
- *Access time*. The delay incurred by the MAC protocol, for gaining access to the transmit channel, for example several control packet must be exchanged at the MAC layer before data can be sent. The MAC delay also depends on how many nodes there are in the network, and how many of them are sending messages currently.
- *Propagation delay*. The time needed for the message to reach the destination, once it has left the sender. This time is small when communicating nodes are within the reach of each-other. On the other hand, it is big if other nodes serve as intermediaries, because then time is spent for packet queuing and processing.
- *Receive time*. Processing required for the packet at the receiver. If the arrival time of the packet is timestamped in a low enough layer of the receiver's stack, the overhead of packet processing is eliminated from the computations.

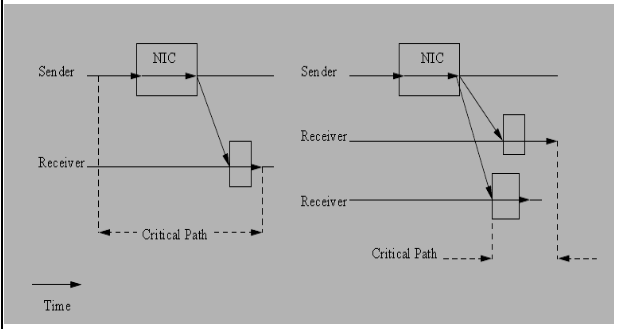

Fig. 6. Time-critical path for traditional protocols (left) and RBS protocol (right)

The RBS algorithm can be split into three major events [14]:

- *Flooding:* a transmitter broadcasts a synchronization request packet.
- *Recording:* the receivers record their local clock time when they initially pick up the *sync* signal from the transmitter.
- *Exchange:* the receivers exchange their observations with each other.

11.1 RBS Protocol - phase offset estimation

The RBS protocol used to compute the phase offset of n receivers is [2]:

- A transmitter broadcasts m reference packets.
- Each of the n receivers records the time that the reference was observed, according to its local clock.
- The receivers exchange their observations.
- Each receiver i can compute its phase offset to any other receiver j as the average of the phase offsets implied by each pulse received by both nodes i and j.

11.2 K RBS Protocol - clock skew estimation

All clocks don't run at the same rate due to frequency differences in the oscillators [2].

- The phase difference between two clocks will change over time.
- We need a clock skew estimation algorithm.
- Instead of averaging the phase offsets from multiple observations we perform a least-squares linear regression.
- This offers a fast way to and the line through the phase error observations over time.

12 TIME-SYNC PROTOCOL FOR SENSOR NETWORKS (TPSN)

- Traditional sender-receiver synchronization (RTT-based)
- *Initialization phase: Breadth-first-search flooding*
    - Root node at level 0 sends out a *level discovery* packet
    - Receiving nodes which have not yet an assigned level set their level to +1 and start a random timer
    - After the timer is expired, a new level discovery packet will be sent
    - When a new node is deployed, it sends out a *level request* packet after a random timeout

Why this random timer? [6]

12.1 Synchronization phase of TPSN

- Root node issues a *time sync* packet which triggers a random timer at all level 1 nodes
- After the timer is expired, the node asks its parent for synchronization using a synchronization pulse
- The parent node answers with an *acknowledgement*
- Thus, the requesting node knows the round trip time and can calculate its clock offset

Child nodes receiving a synchronization pulse also start a random timer themselves to trigger their own synchronization [6].

TPSN uses the same sources of uncertainty as RBS does (send, access, propagation, and receive), with the addition of two more [14]:

- **Transmission time:** the time for the packet to be processed and sent through the RF transceiver during transmission.
- **Access time:** the time for each bit to be processed from the RF transceiver during signal reception.

12.2 The TPSN works in two phases:

12.2.1 Level Discovery Phase: this is a very similar approach to the flooding phase in RBS, where a hierarchical tree is created beginning from a root node [3], [14].

12.2.2 Synchronization Phase: in this phase, pair-wise synchronization is performed between each transmitter and receiver [3], [14].

13 LIGHTWEIGHT TIME SYNCHRONIZATION (LTS)

Lightweight Time Synchronization (LTS) [Greunen and Rabaey 2003] was proposed for applications where the required time accuracy is relatively low. The pair-wise synchronization on LTS is similar to TPSN except for the treatment of the uncertainties (LTS adopts a statistical model for handling the errors). The simulation results show that the accuracy of LTS is about 0.5 seconds [9], [18].

14 TIMESTAMP SYNCHRONIZATION (TSS)

TSS provides internal synchronization on demand. Node clocks run unsynchronized, that is time stamps are valid only in the node that generated them. However, when a time stamp is sent to another node as part of a message, the time stamp is transformed to the timescale of the receiver. For messages sent over multiple hops, the transformation is repeated for each hop [17].

Time-stamp transformation is achieved by determining the age of each time stamp from its creation to its arrival at a sensor node. On a multi-hop path, the age is updated at each hop. The time stamp can then be transformed to the receiver's local timescale by subtracting the age from the time of arrival. The age of a time stamp consists of two components: (1) the total amount of time the time stamp resides in nodes on the path, and (2) the total amount of time needed to transfer the time stamp from node to node. The first component is measured using the local, unsynchronized clocks. The second component can be bounded by the round-trip time of the message and its acknowledgment [17].

15 DELAY MEASUREMENT TIME SYNCHRONIZATION (DMTS)

DMTS is a simple and efficient one-way synchronization protocol, and its single-hop synchronization precision is about 30μs [8].

There are also some synchronization protocols based on one-way message exchange as well as the measurement of delay. An example of such a protocol is Delay Measurement Time Synchronization (DMTS). DMTS has the advantage of low computational complexity and extremely low communication, but lower synchronization precision compared to TPSN due to more uncertainty from delay factors. Delay Measurement Time Synchronization (DMTS) is very energy-efficient and computationally lightweight since it needs only one time broadcast to synchronize all the nodes in a single-hop network. The receivers measure the time delay and set their time as the sender's sending timestamp plus measured time transfer delay [12].

16 FLOODING TIME SYNCHRONIZATION PROTOCOL (FTSP)

• Each node maintains both a local and a global time

• Global time is synchronized to the local time of a reference node

- Node with the smallest id is elected as the reference node

• Reference time is flooded through the network periodically

• Timestamping at the MAC Layer is used to compensate for deterministic message delays

• FTSP provides good global synchronization at low communication cost on small networks, it potentially incurs large skews between neighboring nodes due to the large stretch of the employed tree structure [6], [7].

- Measurements were performed in an eight-by-eight grid of Berkeley Motes, where each Mote has a direct radio link to its eight closest neighbors. With this setup, the network synchronized in 10 minutes to an average (maximum) synchronization error of 11.7 μs (38 μs), giving an average error of 1.7 μs per hop [17].
- FTSP achieves the required accuracy by utilizing a customized MAC layer time-stamping and by using calibration to eliminate unknown delays. FTSP is robust to network failures, as it uses flooding both for pair-wise and for global synchronization. Linear regression from multiple timestamps is used to estimate the clock drift and offset. The main drawback of FTSP is that it requires calibration on the hardware actually used in the deployment (and thus, it is not a purely software solution independent of the hardware). FTSP also requires intimate access to the MAC layer for multiple timestamps. However, if well calibrated, the FTSP's precision is impressive (less than 2µs) [18].

17 Rapid Time Synchronization (RATS*)*

Employs a network- wide broadcast to disseminate the local time of the root node. As in the FTSP protocol, linear regression is used to estimate and compensate clock skews [7].

Rapid Time Synchronization (RATS) is a proactive timesync protocol that utilizes RITS to achieve network-wide synchronization with microsecond precision and rapid convergence. Rapid Time Synchronization (RATS), a proactive time synchronization protocol, achieves network-wide timesync in medium size networks with microsecond precision and rapid convergence. The basic idea is the same as in RITS, except that the direction of messages is reversed: instead of converge cast from multiple nodes to the sink, we use a
broadcast from a single node, the root, to all other nodes in the network. Rapid Time Synchronization (RATS) protocol, in which a root node broadcasts its local time in the network using RITS. In a 60-node 10-hop network the algorithm achieved network-wide synchronization in 4 seconds, and the maximum and average time synchronization errors were 26µs and 2.7µs, respectively [21].

18 Gradient Time Synchronization Protocol (GTSP)

The GTSP protocol does not exhibit this exceptionally bad growth of the global skew. The basic idea of the algorithm is that logical clock rates are determined by perpetually measuring the relative clock rates of neighboring nodes and setting the own rate to the average of the outcomes of these measurements. An important feature of this technique is that rates are independent of observed logical clock skews, therefore errors are not amplified as by the FTSP protocol [7].

However, this imposes the need to ensure that small skews do not accumulate over time. Hence, nodes regularly increase their logical clock values by the average skew to neighboring clocks. For a network diameter $D = 10$, this results in a global skew comparable to the one of FTSP, but averaging over all neighbors yields a much smaller local skew. The employed message pattern is identical to FTSP and is the reason why GTSP and similar algorithms fail to achieve a global skew of $O(JD)$.

GTSP focuses on improving the synchronization accuracy of neighboring sensor nodes [8].

19 Prediction Based Long-Cycle Time Synchronization (PLTS)

PLTS is a combination of periodic synchronization and prediction synchronization. It makes use of an existing time synchronization protocol to accomplish the periodic synchronization, while during the intervals of periodic synchronization; each node applies a prediction model to calibrate its own logic time according to the crystal oscillator's frequency characteristics. By this means, all nodes can keep synchronization till next periodic synchronization starts. Experiment results show that PLTS can reduce resynchronization frequency remarkably and possesses good merits in saving energy and reducing traffic load [8].

The application scenarios of PLTS are described as follow: sensor networks is deployed in outdoor situation and the environment temperature changes with the alternation of day and night; Sink node possesses a precise clock and broadcast its time information to start the periodic synchronization at a fixed interval; sensor nodes are equipped with low precise clocks, which can be influenced by environment temperature seriously.

As already mentioned above, PLTS is a combination of periodic synchronization and prediction synchronization. PLTS applies DMTS protocol to accomplish the periodic synchronization (not limited to DMTS, other protocols are also potential alternatives). The general processes of periodic synchronization are as follow: when a fixed interval times out, Sink node sets its time-level as 0 and broadcasts synchronization messages with its current time to the networks; when a sensor node receives synchronization message with time level *n*, it first synchronizes itself to the sender and sets its own time-level as *n*+1, and then it stamps the synchronization message with its current time before sending the massage out; the synchronization process continues similarly until all the nodes are synchronized to Sink node. As the detail process of periodic synchronization can be seen in literature, we put our emphasis on describing the prediction synchronization of PLTS in the following contents [8].

PLTS makes use of the oscillator's average frequency in each time slot to represent its actual working frequency, the longer the periodic synchronization interval is, the larger the deviation between oscillator's average frequency and actual working frequency is. Hence, the accuracy of prediction synchronization decreases accordingly.

PLTS makes use of Winters Method based model but not the simple linear model as TBS adopted to calibrate sensor nodes' logical time, and thus its prediction synchronization error is less than TBS.
To evaluate the performance of PLTS in reducing resynchronization frequency, we compare it with TBS, PulseSync and DMTS under same synchronization accuracy requirements. DMTS is a simple and efficient one-way synchronization protocol, and its single-hop synchronization precision is about 30μs, while PulseSync is a high precision time synchronization algorithm, its single-hop synchronization error [8].

20 Cluster-Based Method

We use clusters to organize the whole network. The cluster head nodes are synchronized by using the first method and in each cluster the members are synchronized with the cluster head. These two methods are not localized; each synchronization process involves all the nodes in that network partition [10].

Clusters are used to organize the whole network. The cluster heads are synchronized using the first method. A second synchronization round synchronizes the members within each cluster with their cluster head. Although the all-node-based method and the cluster-based method are not localized (because they involve all the nodes in the system), they have some interesting properties with respect to reducing the synchronization error [19].

21 Fully Localized Diffusion-Based Method

To achieve full scalability, we propose a fully localized diffusion-based method with both synchronous and asynchronous implementations, in which each node exchanges and updates information locally with its neighbors. The diffusion-based algorithm can also be extended straightforwardly. synchronous diffusion-based algorithm that is fully localized. The direct diffusion method is used to reinforce the best path from the source to the sink. The goal of our diffusion-based algorithm is to use local operations to achieve global consensus.

Diffusion-based load balancing assumes the computation load on the computers is fine enough to be treated as a continuous quantity. By using diffusion, each computer can give its load to other connected computers or take load from connected computers if it is under-loaded. Cybenko and Boillat analyzed the diffusion method in load balancing. They gave the sufficient and necessary condition for the convergence of the method.

Fully localized diffusion-based method with both synchronous and asynchronous implementations in which each node exchanges and updates information locally with its neighbors. No global operations are required. In the synchronous rate-based algorithm, neighboring nodes exchange clock reading values proportional to their clock difference in a set order [19].

22 ASYNCHRONOUS DIFFUSION (AD)

AD supports the internal synchronization of a whole network. The algorithm is very simple: each node periodically sends a broadcast message to its neighbors, which reply with a message containing their current time. The receiver averages the received time stamps and broadcasts the average to the neighbors, which adopt this value as their new time. It is assumed that this sequence of operations is atomic, that is the averaging operations of the nodes must be properly sequenced. Simulations with a random network of 200 static nodes showed that the synchronization error decreases exponentially with the number of rounds [19].

23 CONCLUSION

In current scenarios there are many time synchronization methods apply in wireless sensor networks. Each method has its own feature and characteristics. Each method has advantages and disadvantages. Each method has its own limitations, working environmental conditions and provided its results. Some of the methods provide accurate results and some of them provide approximate results. In this paper 21 Times Synchronization methods are discussed and studied.
DMTS is a simple and efficient one-way synchronization protocol, and its single-hop synchronization precision is about 30μs. DMTS is a simple and efficient one-way synchronization protocol, and its single-hop synchronization precision is about 30μs, while PulseSync is a high precision time synchronization algorithm, its single-hop synchronization error.
Rapid Time Synchronization (RATS) protocol, in which a root node broadcasts its local time in the network using RITS. In a 60-node 10-hop network the algorithm achieved network-wide synchronization in 4 seconds, and the maximum and average time synchronization errors were 26μs and 2.7μs, respectively. FTSP the network synchronized in 10 minutes to an average (maximum) synchronization error of 11.7 μs (38 μs), giving an average error of 1.7 μs per hop. FTSP's precision is impressive (less than 2μs)

REFERENCES

[1] Prakash Ranganathan, Kendall Nygard," TIME SYNCHRONIZATION IN WIRELESS SENSOR NETWORKS: A SURVEY," Department of Computer Science, North Dakota State University, Fargo, ND, USA prakashranganathan@mail.und.edu International Jouirnal of UbiCamp (IJU), Vol.1 No. 2, April 2010

[2] Koninis Christos "Time Synchronization in Wireless Sensor Networks," Research Academic Computer Technology Institute University of Patras, Greece October 14th, 2009.

[3] Ganeriwal, S., Kumar, R., Srivastava," M.B.: Timing-sync Protocol for Sensor Networks," In: 1st International Conference on Embedded Networked Sensor Systems, pp. 138–149 (2003)

[4] Jeremy Elson and Deborah Estrin," Time Synchronization for Wireless Sensor Networks,"Department of Computer Science, University of California, Los Angeles; and USC/Information Sciences Institute _ jelson,estrin@isi.edu, To appear in Proceedings of the 2001 International Parallel and Distributed Processing Symposium (IPDPS), Workshop on Parallel and Distributed Computing Issues in Wireless Networks and Mobile Computing, April 2001, San Francisco, CA, USA. Also published as UCLA CS Technical Report 200028.

[5] Sepideh Nazemi Gelyan1, Arash Nasiri Eghbali2, Laleh Roustapoor2, Seyed Amir Yahyavi Firouz Abadi3, and Mehdi Dehghan2," SLTP: Scalable Lightweight Time Synchronization Protocol for Wireless Sensor Network," 1 Dept. Of Computer Engineering, Islamic Azad University, Firouzkooh Branch 2 Computer Engineering Department, Amirkabir University of Technology ,3 Electrical & Computer Engineering Department, University of Tehran H. Zhang et al. (Eds.): MSN 2007, LNCS 4864, pp. 536–547, 2007. © Springer-Verlag Berlin Heidelberg 2007

[6] *Roger Wattenhofer,"* Clock Synchronization YouTube Clock Synchronization," Chapter 9 *Ad Hoc and Sensor Networks*

[7] Christoph Lenzen, Philipp Sommer, Roger Wattenhofer, "Optimal Clock Synchronization in Networks," Computer Engineering and Networks Laboratory ETH Zurich, Switzerland , lenzen@tik.ee.ethz.ch , sommer@tik.ee.ethz.ch, wattenhofer@tik.ee.ethz.ch, November 4–6, 2009, Berkeley, CA, USA. Copyright 2009 ACM 978-1-60558-748-6 ...$5.00

[8] Wentao Jiang12, Limin Sun2, Junwei Lv1, Feng Wang3," A Prediction based Long-cycle Time Synchronization Algorithm for Sensor Networks,"Department of Control Engineering, Naval Aeronautical and Astronautical University, Yantai, China1 Institute of Software, Chinese Academy of Sciences, Beijing, China2 School of Computing Science, Simon Fraser University, British Columbia, Canada3 978-1-4244-5637-6/10/$26.00 ©2010 IEEE

[9] Jana van Greunen, Jan Rabaey," Lightweight Time Synchronization for Sensor Networks," University of California, Berkeley ,2108 Allston Way, Suite 200, Berkeley, CA 94704, USA +1 510.666.3102, janavg@eecs.berkeley.edu, jan@eecs.berkeley.edu, WSNA'03, September 19, 2003, San Diego, California, USA. COPYRIGHT 2003 ACM 1-58113-764-8/03/0009…$5.00.

[10] Blerta Bishaj ," Synchronization in Sensor Networks,"Helsinki University of Technology

[11] Jeremy Elson, Lewis Girod and Deborah Estrin, "Fine-Grained Network Time Synchronization using Reference Broadcasts," 2002

[12] Shujuan Chen, Adam Dunkels, Fredrik Osterlind, Thiemo Voigt," Time Synchronization for Predictable and Secure Data Collection in Wireless Sensor Networks," ,Swedish Institute of Computer Science Mikael Johansson KTH Stockholm, The Sixth Annual Mediterranean Ad Hoc Networking WorkShop, Corfu, Greece, June 12-15, 2007 Page 165-172

[13] Bharath Sundararaman, Ugo Buy, and Ajay D. Kshemkalyani," Clock Synchronization for Wireless Sensor Networks: A Survey," Department of Computer Science, University of Illinois at Chicago 851 South Morgan Street Chicago, IL 60607 buy@cs.uic.edu March 22, 2005

[14] Robert Akl, Yanos Saravanos and Mohamad Haidar," Hybrid Approach for Energy-Aware Synchronization" *University of North Texas Denton, Texas, USA Chapter 18 Page-413-418*

[15] Santashil PalChaudhuri (santa@cs.rice.edu), Amit Kumar Saha (amsaha@cs.rice.edu), David B. Johnson (dbj@cs.rice.edu), "Adaptive Clock Synchronization in Sensor Networks," Department of Computer Science Rice University Houston, TX 77005, *IPSN'04,* April 26–27, 2004, Berkeley, California, USA. Copyright 2004 ACM 1-58113-846-6/04/0004 ...$5.00

[16] Jeremy Elson - Deborah Estrin, "Time Synchronization for Wireless Sensor Networks," Fakultät Informatik - Institut fur Systemarchitektur - Professur Rechnernetze Dresden, 28.01. 2008 Birgit Pretscheck TU Dresden, 28.01.08 Wireless Sensor Networks Folie 5 von 26, TECHNISCHE UNIVERSITAT DRESDEN

[17] Kay Romer, Philipp Blum, Lennart Meier," Time Synchronization and Calibrationin Wireless Sensor Networks," ETH Zurich, Switzerland Chapter 1

[18] Suyoung Yoon, Chanchai Veerartittiphan, and Mihail L. Sichitiu ," Tiny-Sync: Tight Time Synchronization for Wireless Sensor Networks,"Dept. of Electrical and Computer Engineering, North Carolina State University, Raleigh, NC 27695, ACM Journal Name, Vol. V, No. N, Month 20YY, Pages 1–33.

[19] Qun Li, Member, IEEE, and Daniela Rus, Member, IEEE," Global Clock Synchronization in Sensor Networks," IEEE TRANSACTIONS ON COMPUTERS, VOL. 55, NO. 2, FEBRUARY 2006.

[20] Marco Valero: Dr. Yingshu Li(Instructor) ," Time Synchronization for Wireless Sensor Networks,"CSC8980 Wireless Sensor Networks, GeorgiaState University

[21] Branislav Kus´y†, Prabal Dutta‡, Philip Levis‡, Mikl´os Mar´oti†, ´Akos L´edeczi†_, and David Culler‡ ," Elapsed Time on Arrival: A simple and versatile primitive for canonical time synchronization services,"†Institute for Software Integrated Systems Vanderbilt University Nashville, Tennessee 37203, USA E-mail:{branislav.kusy,miklos.maroti, Computer Science Division University of California, Berkeley California 94720, USA

[22] IEEE, Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications. *IEEE Std. 802.11,* November 1997.

[23] Jaynesh Doshi, Prof. Ivan Stojmenovic, "Time Synchronization in Wireless Sensor Networks," Wireless Ad Hoc Networking (Fall 2008)