TIME SYNCHRONIZATION IN MEASUREMENT NETWORKS

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

ZAHİT EVREN KAYA

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCES
IN
ELECTRICAL AND ELECTRONICS ENGINEERING

JANUARY 2010

Approval of thesis:

TIME SYNCHRONIZATION IN MEASUREMENT NETWORKS

submitted by **ZAHİT EVREN KAYA** in partial fulfillment of the requirements for the degree of **Master of Science in Electrical and Electronics Engineering Department, Middle East Technical University** by,

Prof. Dr. Canan Özgen
Dean, Graduate School of **Natural and Applied Sciences** _____

Prof. Dr. İsmet Erkmen
Head of Department, **Electrical and Electronics Engineering** _____

Prof. Dr. Semih Bilgen
Supervisor, **Electrical and Electronics Engineering Dept., METU** _____

Examining Committee Members:

Asst. Prof. Dr. Cüneyt Bazlamaççı
Electrical and Electronics Engineering Dept., METU _____

Prof. Dr. Semih Bilgen
Electrical and Electronics Engineering Dept., METU _____

Turan Demirci, M.Sc.
Power Electronics Group, TUBITAK-SPACE _____

Assoc. Prof. Dr. Özgür Barış Akan
Electrical and Electronics Engineering Dept., METU _____

Asst. Prof. Dr. Şenan Ece Güran Schmidt
Electrical and Electronics Engineering Dept., METU _____

Date: 28.01.2010

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Last name: ZAHİT EVREN KAYA

Signature:

ABSTRACT

TIME SYNCHRONIZATION IN MEASUREMENT NETWORKS

Kaya, Zahit Evren

M.Sc. Department of Electrical and Electronics Engineering

Supervisor: Prof. Dr. Semih Bilgen

January 2010, 72 pages

AMR (Automatic Measurement Reading) applications usually require measurement data to be collected from separate locations. In order to combine the data retrieved from separate sources into a meaningful result, all sources should share a common time sense. Therefore, it is necessary to implement a synchronization scheme in measurement networks. In this thesis, a synchronization scheme which combines GPS (Global Positioning System) and two high accuracy WSN (Wireless Sensor Network) time synchronization algorithms will be proposed and evaluated. The synchronization accuracy of the proposed method is compared to the accuracy of NTP (Network Time Protocol) by simulation. This research work is fully supported by the Public Research Grant Committee (KAMAG) of TUBİTAK within the scope of National Power Quality Project of Turkey with the project No: 105G129.

Keywords: Network time synchronization, wireless sensor networks, GPS, NTP, Automatic measurement reading

ÖZ

ÖLÇÜM AĞLARINDA ZAMAN EŞLEME

Kaya, Zahit Evren

Yüksek Lisans, Elektrik ve Elektronik Mühendisliği Bölümü

Tez Yöneticisi: Prof. Dr. Semih Bilgen

Ocak 2010, 72 sayfa

Otomatik Ölçüm Okuma uygulamaları, coğrafi açıdan birbirinden uzak konumlarda bulunan ölçüm cihazlarından gelen verilerin bir konumda birleştirilmesini gerektirebilmektedir. Dağınık konumlardan gelen ölçüm verilerinin anlamlı bir bütün oluşturabilmesi için tüm ölçüm verisi kaynakları arasında ortak bir zaman anlayışı olması gerekir. Bu durum ölçüm ağlarında zaman eşleme yöntemlerinin uygulanmasını gerektirmektedir. Bu tezde, GPS (Küresel Konumlama Sistemi) ve yüksek hassasiyetli iki Kablosuz Algılayıcı Ağı zaman eşleme algoritması birleştirilerek oluşturulan bir zaman eşleme yöntemi sunulacak ve değerlendirilecektir. Sunulan yöntem tarafından sağlanan zaman eşleme hassasiyeti NTP Ağ Zaman Eşleme Protokolü tarafından sağlanan hassasiyetle benzetim yoluyla karşılaştırılacaktır. Bu araştırma TÜBİTAK Kamu Araştırmaları Grubu tarafından, 105G129 referans numaralı Türkiye Güç Kalitesi Milli Projesi Kapsamında desteklenmiştir.

Anahtar Kelimeler: Ağ zaman eşleme yöntemleri, Kablosuz algılayıcı ağları, GPS, NTP, Otomatik ölçüm okuma

To my parents, who offered me unconditional support throughout the course of this thesis…

ACKNOWLEDGEMENT

I am heartily thankful to my supervisor, Prof. Dr. Semih Bilgen, whose encouragement, guidance and support from the initial to the final level enabled me to develop an understanding of the subject.

Also, I'm very grateful to Prof. Dr. Muammer Ermiş and Turan Demirci, M.Sc., from the TUBITAK UZAY (Space Technologies Research Institute) Power Electronics Group for their support for my research.

TABLE OF CONTENTS

LIST OF TABLES

TABLES

LIST OF FIGURES

FIGURES

LIST OF ABBREVIATIONS

WSN	: Wireless Sensor Network
NTP	: Network Time Protocol
GPS	: Global Positioning System
AMR	: Automatic Meter Reading
SSI	: System Synchronization Interface
PPS	: Pulse-Per-Second
TNPQ	: TUBITAK National Power Quality
UTC	: Universal Coordinated Time
RBS	: Reference Broadcasting System
TPSN	: Time-sync Protocol for Sensor Networks
TDP	: Time Diffusion Protocol
MAC	: Medium Access Control
FTSP	: Flooding Time Synchronization Protocol
WAN	: Wide Area Network
TNPQP	: TUBITAK National Power Quality Project
TUBITAK	: The Scientific and Technological Research Council of Turkey
TUBITAK-SPACE	: TUBITAK Space Technologies Research Institute
TEIAŞ	: Turkish Electricity Transmission Company
PTP	: Precision Time Protocol

CHAPTER 1

INTRODUCTION

In modern computer networks, accurate timekeeping is critically important, because most of the tasks done by computers and microcontroller-based systems require a common time sense to be established between the members of the network. For instance, if the network is used for collecting measurement data or event logging, detected events and measurement data should be accurately time stamped to allow correlating between different nodes of the network. Otherwise, it is impossible to create a frame of reference between all devices on the network. AMR (Automatic Meter Reading) can be given as an example where data is collected from source on separate locations. In order to combine the data retrieved from separate sources into a meaningful result, all sources should share a common time sense. Some of the other applications which require time synchronization may be listed as [1];

- Distributed database journaling and logging
- Intruder detection, location and reporting
- Aviation traffic control and position reporting
- Stock market buy and sell orders
- Distributed network gaming and training
- Network monitoring, measurement and control

- Secure document timestamps (cryptography, watermarking etc.)

Computer clocks are not inherently accurate. Crystal oscillators cause drift of several PPM (part per million) and clock driver software has a budget of non-determinism up to tens of milliseconds. Therefore, it is not possible to achieve meaningful time synchronization between nodes by allowing their clocks run on their own. A network time synchronization scheme should be implemented.

The work on this thesis primarily focused on fulfilling the synchronization needs of measurement networks which are networks used for event logging and measurement purposes where data is collected from distant geographical positions and transferred to a center by internet for storing and data aggregation. Although, time synchronization concepts in this text are general to be applied to any similar system, a particular measurement system should be mentioned to explain the selection of network hierarchy and delay parameters used in the simulations reported in the next chapters. Specifically, the needs of TUBITAK National Power Quality Project (TNPQP) have been addressed in this study.

The remainder of this thesis is organized as follows: In the remainder of this chapter, TNPQP is briefly described to illustrate a potential implementation case of synchronization methods and explain the reasoning behind the selection of simulation parameters.

In Chapter II, computer clock and synchronization error concepts are reviewed. In Chapter III, current synchronization methods, which can be utilized for measurement networks, are investigated in detail. NTP is investigated in Chapter IV. NTP and WSN synchronization methods

are evaluated to be utilized in measurement networks in Chapter V. Simulation results for two of these synchronization methods are presented in Chapter VI. Finally Chapter VII presents an overview of the achievements and shortcomings of the study, derives conclusions and suggests future work.

1.1 General Information About TNPQP

The term "power quality" can be described as the degree of continuity of electrical service and stability of this service in terms of voltage and frequency. It is also necessary for the voltage and current waveforms to be absolute sinusoidal waveforms [2].

To determine and solve the power quality problems in the electrical production and distribution services in Turkey, Space Technologies Research Institute (TUBITAK-SPACE), which is organized under The Scientific and Technological Research Council of Turkey TUBITAK), conducts the TPQNP. Consisting of nine sub-projects, the main goal of TNPQP is to increase the quality of the power in the Turkish Electricity Transmission System in the medium and long term. One of the nine sub-projects of TNPQP, namely National Power Quality Monitoring Project, is described in the following part of the text, because of its need for time synchronization.

The National Power Quality Monitoring sub-project aims to place National Power Quality Monitors to points selected to be the most critical ones in the Turkish Electricity Transmission System to monitor the changes in the Power Quality Parameters (Frequency, Magnitude of Supply Voltage, Flicker, Voltage Sags and Swells, Voltage Interruptions, Transient Voltage Parameters, Supply Voltage Unbalance, Harmonic Voltages and Currents, Interharmonic Voltages

and Currents, Mains Signaling Voltages on the Supply Voltage, Rapid Voltage Changes, Measurements of Underdeviation & Overdeviation Parameters), and Active-Reactive Power Flows from National Power Quality (NPQ) Monitoring Center. There are 4000 points to be monitored in the transmission system. On the other hand, within the scope of the project, measurement and monitoring will be carried out in 150 monitors chosen in a way that they will thoroughly represent Turkey in general [2]. After the project is completed, the infrastructure established by TUBITAK-SPACE will be extended by Turkish Electricity Transmission Company (TEİAŞ) to monitor the whole transmission system.

1.2 Need for Time Synchronization in TNPQP

NPQ Monitors are connected "off-line" to the NPQ Center within TEİAŞ Gölbaşı Facilities. Besides the monitoring purposes, NPQ Monitors are used for recording power quality events. Therefore, time synchronization is necessary to make correlating power quality events observed on different monitors possible. Another reason for TNPQP to require accurate time synchronization is power quality measurement standards declared by IEC. Details of this standard and the level of time synchronization accuracy required by TNPQP is discussed in the following parts of this thesis. Thus, the aim of this thesis is to investigate the synchronization requirements of TNPQP and elaborate on possible alternative solutions.

1.3 Hardware Used in NPQ Monitors

In NPQ Monitors, a data acquisition card is used for collecting data about power quality parameters and then the collected data is processed and reported to the NPQ Center by PCs. It is assumed that internet connection is possible everywhere the monitors are placed [3].

The data acquisition card selected to be utilized in NPQ Monitors is DAQ2205 from Adlink Technology. DAQ2205 is a high performance data acquisition card with 64 single ended and 32 differential ended analog input, and 16-bit A/D resolution. DAQ2205 boards can be found in PCI, PCI Express and PXI interfaces. Windows and Linux drivers and interfaces are also provided by the manufacturer.

PCs within the NPQ monitors are planned to be operated using a Linux operating system. Adlink Technology provides a Linux Driver (D2K-DASK-X) to allow the programmers to develop custom applications. With this driver package, Application Programming Interfaces (APIs) are provided to access all functionalities of DAQ2200 series data acquisition cards.

In order to meet the user-specific timing requirements and for synchronizing multiple cards, the DAQ2205 card provides flexible user-controllable timing signals to connect to external circuitry or additional cards [4].

System Synchronization Interface (SSI) provides timing synchronization between multiple cards. Bidirectional structure of SSI interface allows four DAQ2205 cards to be synchronized by choosing one of them as the master and the others as slaves. SSI interface cannot be used to synchronize external devices; it is designed for card-to-card synchronization only.

Users of DAQ2205 can also choose the card to be synchronized and triggered by external sources. Using the EXTTIMEBASE, EXTDTRIG and EXTWFTRIG inputs of Auxiliary Functions Input port, the card can be connected to external time and pulse sources (e.g. GPS). NPQ Monitors in the same location can be synchronized to each other using the SSI interface. On the other hand, NPQ Project requires monitors in different geographic locations to share a common time sense. Therefore, implementation of a synchronization scheme is mandatory.

In the scope of TNPQP, the main time synchronization method is chosen to be the utilization of Pulse-per-Second (PPS) signal of Global Positioning System (GPS) receiver, because of its greater time synchronization accuracy. Utilization of GPS also enables time synchronization between network nodes located on distant geographical positions. Furthermore, PPS signal provided by GPS a receiver can be used for both synchronizing the DAQ card and generating a fixed frequency output.

Although the main goal of GPS utilization is to synchronize DAQ card readings via external trigger and timebase inputs of the card, the PPS signal can also be used for synchronization of the computers which stores and forwards the measurement data. To achieve this, the PPS signal provided by the GPS receivers should be connected to the computers through a low latency interface like parallel port.

Despite the fact that GPS synchronization results in high accuracy in time stamping the measurements, it is possible to lose GPS signals due to electromagnetic interference and problematic positions of the receivers. Furthermore, cost of GPS receivers may be a factor to be considered in large scale networks. Hence, the present study investigates techniques for supplementing GPS synchronization with other high accuracy network synchronization algorithms.

CHAPTER 2

COMPUTER CLOCKS AND SYNCHRONIZATION PROBLEM

Some of the cases in which accurate time keeping is necessary were given as examples in the preceding chapter. In this chapter, computer clock concept and main reasons of the need for time synchronization are investigated.

2.1 Computer Clocks

Computer clocks consist of two parts: a quartz oscillator that generates periodic interrupts and a software driver to count these interrupts in a register [5]. This register contains the current value of the clock. In the following discussion the terms “local clock” or “computer clock” always refer to the value kept in this register and t represents the coordinated universal time (UTC).

Despite the fact that the frequency of a quartz oscillator depends on the conditions, it can be modeled with good accuracy by a fixed-rate oscillator [6], [7]:

$$t_i(t) = a_i t + b_i \qquad (1)$$

where a_i is the drift and b_i is the offset of node i's clock. Local time of node 1 can also be represented by using (1) relative drift a_{12} and relative offset b_{12} between node 1 and node 2 :

$$t_1(t) = a_{12}t_2 + b_{12} \qquad (2)$$

If the two nodes are perfectly synchronized, their relative drift is equal to 1 and their relative offset value is equal to zero.

2.2 General Concepts of Synchronization

The term "network time synchronization" can be used to define the equalization of local clock values of all the nodes constituting a network or only a part of that network locally. Network time synchronization methods can be classified in many different ways. One reasonable way to classify these methods can be to determine what degree they are dependent to a global clock [9].

- Global clock: This is the strictest type of synchronization. In this type, all nodes of a network are synchronized to a global time, namely UTC.

- Relative clock: Instead of maintaining a global time scale among the nodes, a synchronization method may aim to synchronize each node to another node within the network. The resulting clock may be totally different from the global time, UTC. Usually, this kind of synchronization methods is implemented for Wireless Sensor Networks which have hardware with limited energy resources.

- Physical ordering: Relaxing the synchronization of nodes to a level where only the correct ordering of events is necessary helps the nodes to preserve resources like power, memory and processing capability. This kind of synchronization is enough for many kinds of applications.

2.3 Sources of Clock Synchronization Error

Synchronization error mainly results from the sources of non-determinism in a network. When synchronization packets travel between nodes, they face delay at certain phases of communication. Deterministic delays can be tolerated by nodes in clock conversion calculations. On the other hand, some phases of network communication cause non-deterministic delays. The main phases of communication where non-determinism can arise are as follows:

- Send time: This is the time needed for the sender to prepare the synchronization message.
- Access time: This is the time that the network interface needs to release the message into an appropriate communication channel.
- Propagation time: Time needed for the message to propagate between nodes.
- Receive time: Time needed for the network interface of the receiver to process the incoming data and transfer it to the host.

The effects of these sources on synchronization differ from a synchronization method to another. For instance, methods which timestamp the synchronization messages on phases closer to the propagation suffer less from send time. Propagation time is also a

factor mostly ignored because of its relatively less contribution to the total packet delay, but there exist exceptions [10].

2.4 Design Criteria for Network Time Synchronization Methods

As indicated in the preceding discussion, design of time synchronization methods require some certain needs to be emphasized. Following part is a review of fundamental requirements of network time synchronization methods.

Precision: Required precision of time synchronization differs from an application to another. One type of application may require the local clocks of nodes to be as close as possible to each other where another type requires only a correct physical order among measurement readings.

Low complexity: To prevent extra processing and memory load which in turn may affect the real tasks of nodes, algorithms run on the nodes shouldn't have high time and storage complexity.

Resilience: Many measurement applications require the nodes to operate in a hard, even hostile, environment. Time synchronization should be immune to node losses and addition of new nodes.

Even if it's not one of the common requirements of time synchronization, security can also be important for some applications. Especially for government applications, user transactions and measurement information may be preferred to be secured.

Another requirement which is not common to all measurement applications is power efficiency. Some measurement and event logging tasks may utilize wireless, battery-operated nodes as part of the network. In that case, to reduce the cost and the need for maintenance “always-on” synchronization schemes should be avoided for less power consumption.

CHAPTER 3

TIME SYNCHRONIZATION METHODS

Traditional wired networks and wireless networks differ in some aspects, e.g. network interface hardware, network stack complexity on nodes and communication delays. Wired and wireless networks have similar types of delays on network path, but magnitudes of delays are different. For instance, a microcontroller-based system which use 802.15.4 protocol stack may suffer less from access time, because of relative simplicity of its network interface, where a PC with much higher computational power may lower receive and send delays.

Network time synchronization is a subject which has been studied for years by academia. Many time synchronization methods are proposed for different types of network topologies and network nodes. When it is measurement networks under consideration, it may be hard to indicate a single type of network topology, node hardware or communication protocol. Therefore, it may be necessary to narrow the scope of this concept to networks like NPQ project has, e.g. internet connection from each node to a single server. As indicated above, NTP is most commonly used time synchronization method for Internet. It has many sub-protocols under it for reducing synchronization error and, for its WAN implementations; its synchronization accuracy can be improved to tens of milliseconds (without any external hardware).

IEEE-1588 Precision Time Protocol (PTP) is a high precision time synchronization protocol which is mainly used for automation and

control networks based on LAN architectures. Since PTP utilizes IP multicasts to achieve time synchronization, this protocol can be extended to cover WANs. Since in IEEE-1588, the main improvement on time synchronization made by delay measurement phase of the algorithm, switches on the network should be PTP-aware to efficiently propagate the PTP packets [28]. On the other hand, if the switches on the network are not PTP-aware, delay measurement/correction part of PTP doesn't work as efficiently as in the LAN case. Therefore, resulting time synchronization accuracy may not be better than NTP can provide [29].

In [31], authors propose a time synchronization method which use the core of NTP time synchronization methods and introduce some improvements for disruption tolerant networks. In this method, namely Double-Pairwise Time Synchronization Protocol (DPT), a pairwise message exchange is done by two network nodes. In contrast to the message exchange used in NTP which utilize a single packet for each round of synchronization, DPT uses two messages back-to-back in each round. The purpose of sending messages back-to-back is they can be queued very closely in the buffers. By this, they experience similar delay queuing delays. Authors apply NTP core and DPT to a 20-node IP network topology. It is claimed in [31] that maximum synchronization error is 19.3 ms where this value is 53.8 ms for NTP core.

Wang et al. discuss the suitability of NTP to a frequency measurement network which has similar requirements with TNPQP [32]. For measuring phasor angle error at some strategic points, frequency disturbance recorders are placed in different locations in North America. Monitors report the measurement data to a monitoring center. The authors present test results which show that NTP achieve 17 ms maximum time synchronization error value in 40

minutes for 32 s poll rate. The authors argue that this level of synchronization error is not suitable for phasor angle error measurements. On the other hand, this level of maximum synchronization error may be considered acceptable for TNPQP, since power quality measurement standards require the maximum synchronization error to be at most 20 ms.

Over the years NTP has been the dominant technique used for Internet time synchronization. Furthermore, reference software implementation of NTP makes it more advantageous in cases where the synchronization method should be tailored to application-specific requirements. On the other hand, it is considered to be useful to discuss the efficiency of some other time synchronization methods. Wireless Sensor Networks is an area of research which is very active since the beginning of this decade. Academic activity in this specific type of networks results in a diverse set of novel methods. One of the main topics which are investigated by WSN researchers is the time synchronization problem. Some Wireless Sensor Network time synchronization methods, like TPSN and RBS, are claimed to result in better accuracy than NTP on wireless networks. Magnitudes of delay and non-determinism differ between wireless network and wired networks. On the other hand, microsecond-level time synchronization accuracy reported by the authors makes it attractive to compare WAN performance of these algorithms with NTP in its basic form.

3.1 Synchronization Methods for Sensor Networks

With increasing number of applications, WSN synchronization became an attractive research area for computer scientists. In reference [8], a review of studies about synchronization problem was

made by Sivrikaya. Since the date that this survey was published, new methods with different approaches to the problem have been proposed. This part summarizes the recent development on the subject.

Elson and Estrin outline the synchronization problem and present a low-power synchronization scheme, post-facto synchronization, in reference [11]. In post-facto synchronization nodes are normally unsynchronized. When the nodes encounter an event they record the time of that event. A third party node sends a synchronization pulse to all the nodes in its scope. Nodes receiving this synchronization pulse estimate the exact time of past events with respect to this reference pulse. Since all nodes in the sender's scope take the synchronization pulse almost at the same time, a common time sense between the nodes is founded. Post-facto synchronization is utilized in Reference-Broadcast Synchronization (RBS) method [11] which will be reviewed below.

Instead of receiver-receiver synchronization, Ganeriwal et al. propose a sender-receiver synchronization scheme, Timing Sync Protocol for Sensor Networks (TPSN), which aims to maintain network-wide time synchronization [12]. Consisting of two steps, namely level discovery and synchronization, this algorithm synchronizes local clocks of all nodes in the network to a reference node which is probably equipped with a GPS receiver.

In reference [13], Maróti et al. propose an algorithm, namely the Flooding Time Synchronization Protocol (FTSP) for WSN, which is a combination of MAC layer time-stamping and clock skew compensation with linear regression. The goal of the FTSP is to provide network-wide time synchronization with low communication

overhead. Furthermore, the structure of the FTSP provides an implicit dynamic topology management for robustness.

In reference [14], the authors present two light-weight synchronization algorithms. These algorithms are Mini-sync and its form for more constrained resources, Tiny-sync. By these algorithms, time offset and drift information are delivered with tight deterministic bounds with them. Having this information, nodes are enabled to know offset and drift error parameters.

Greunen and Rabaey argue that computation requirements of the synchronization algorithms can be reduced by relaxed accuracy constraints [15]. They propose lightweight tree-based synchronization (LTS) algorithms by which accuracy is sacrificed for power and complexity considerations.

A more recent tree based algorithm utilizes both sender-receiver and receiver-receiver synchronization [16]. After building a multiple-level spanning tree and dividing it into subtrees, parents of each subtree initiate sender-receiver synchronization with one of the children nodes. Then the synchronized child node and the other children nodes of that subtree make receiver-receiver synchronization.

Sun et al. propose secure synchronization schemes where the effect of malicious nodes on a WSN can be removed by providing multiple synchronization paths for each node [17]. By these schemes even the effect of malicious source nodes on a WSN can be eliminated by utilizing multiple source nodes for synchronization.

In the remainder of this section, an overview of recently proposed WSN synchronization schemes is presented without ignoring the need

of reviewing the pioneering work done in this area. To be able to compare the performances of the investigated synchronization methods with non-WSN time synchronization methods like NTP, the main criterion is chosen to be the synchronization accuracy. Especially, the methods with microsecond-level accuracy are considered.

3.1.1 Reference-Broadcast Synchronization

Reference-Broadcast Synchronization (RBS), proposed by Elson, Girod and Estrin [11], is a synchronization scheme where a set of receiver nodes are synchronized with one another. In this scheme, nodes periodically send synchronization messages to their neighbors. These synchronization messages contain no explicit timestamp for the receivers to adjust their clocks. Instead, the receiving node uses the reception time of the synchronization message as a reference for comparing its clock. The authors argue that send time and access time are parameters dependent to the instantaneous load on the sender's CPU and the network. This makes them difficult to estimate. They add that relative differences of propagation time can be neglected, since their contribution to non-determinism is in nanosecond level. Therefore, removal of the sender's non-determinism from the critical path results in a more accurate synchronization than the sender-receiver schemes can provide. Figure 1 shows the critical paths of the sender-receiver and receiver-receiver synchronization schemes.

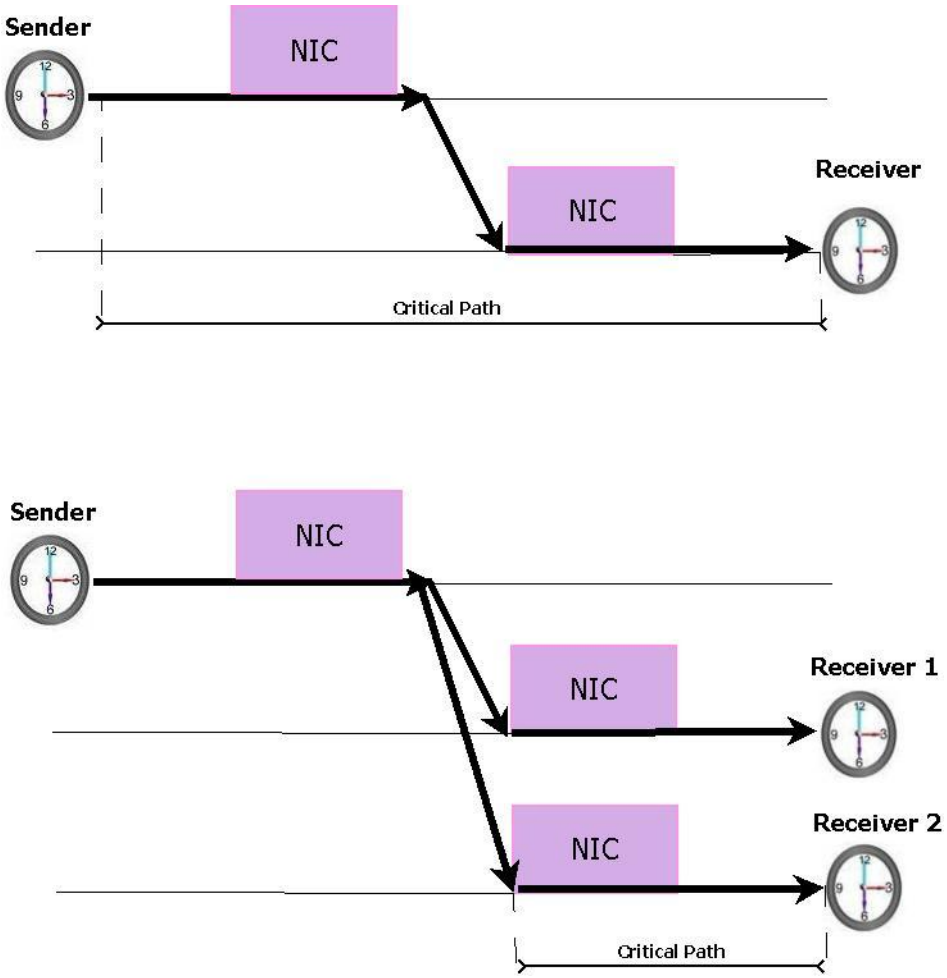


Figure 1 – Paths of receiver-receiver and sender-receiver schemes (Adapted from [11]).

The simplest form of RBS can be modeled as the broadcast of a single pulse to two receivers [11].

- Transmitter broadcasts a reference packet to two receivers (i an j).
- Receivers record the time that the reference packet was received.
- Receivers exchange the recorded reception times.

These steps establish local synchronization among receivers.

The authors claim that it is possible for the WSN to synchronize themselves to a global time scale. To establish this kind of synchronization one of the nodes in the network can be equipped with a GPS receiver and the PPS output of the GPS receiver is treated as it was a reference broadcast source.

The authors used a set of Berkeley Motes to test this algorithm and they empirically observed that the receiver error was Gaussian. This information is used in RBS to increase the synchronization precision statistically. Instead of sending one reference packet, the transmitter broadcasts m reference packet to let the nodes to calculate an average value for the phase offset. The authors claim that using 30 reference broadcasts improved the precision from 11 μsec to 1.6 μsec. and for a group of 20 receivers the precision was 5.6 μsec.

RBS was tried by the same authors on two different systems. On Berkeley Motes RBS could synchronize clocks within 11 μsec with 19200 baud radios. On Compaq IPAQs using 11MBit/sec 802.11 network a precision of 6.29± 6.45 μsec was achieved. Utilization of kernel timestamps resulted in a precision of 1.85±1.28 μsec.

3.1.2 Timing-sync Protocol for Sensor Networks

Ganeriwal et al. presented Timing-sync Protocol for Sensor Networks (TPSN) that aims at providing network-wide time synchronization in a sensor network [12]. The algorithm is based on the conventional sender-receiver synchronization. The authors argue that the classical approach of doing a handshake between a pair of nodes is a better approach than synchronizing a set of receivers, in case the packets

are time stamped at the moment when they are sent, i.e.at MAC layer.

TPSN basically consists of two steps. The first step of the algorithm, namely "level discovery phase", is to create a hierarchical topology in the network where every node is assigned a level. In this hierarchical structure, a node belonging to level i should be able to communicate with at least one node belonging to the upper layer, i-1. Level discovery phase occurs when the network is deployed. First of all, the root node is assigned a level 0 and it initiates level discovery process by broadcasting a level_discovery packet. This packet contains the identity and the level of the sender. The nodes receiving the level_discovery packet assign themselves a level, one greater than the level they have received. After assigning themselves appropriate levels, nodes broadcast their own level_discovery packets. At the end of this phase all nodes in the network are covered by the hierarchical structure.

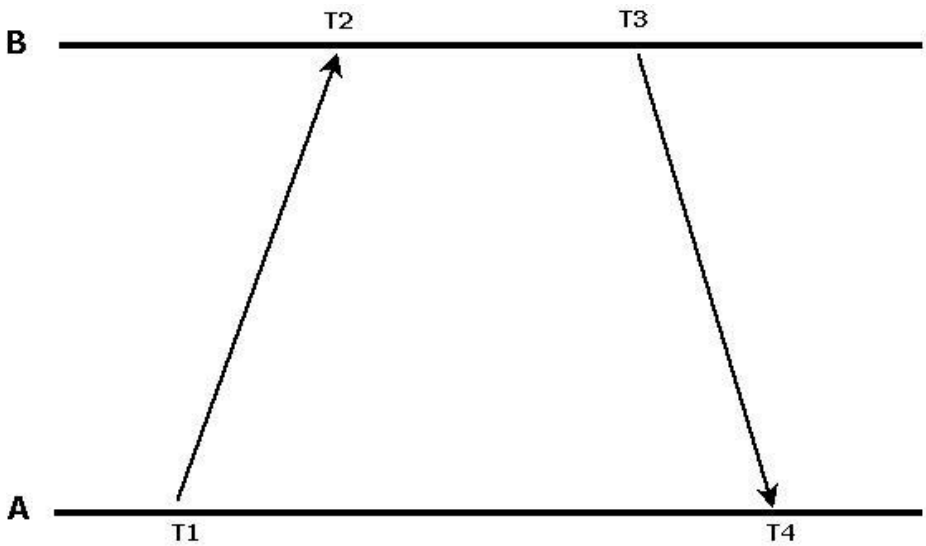


Figure 2 - Pairwise synchronization (Adapted from [12]).

After the foundation of a hierarchical structure, the second stage of TPSN is initiated by the root node. This stage is called the "synchronization phase". In this phase, pairwise synchronization is performed along the edges of the hierarchical structure built in the previous phase. Two-way message exchange is shown in Figure 2. In this figure, times T1 and T4 are measured by the local clock of node A where times T2 and T3 are measured by the local clock of node B. Synchronization begins with the "synchronization pulse" packet sent by node A at time T1. Receiving the packet, node B records the reception time T2 where T2 is equal to T1+ d. Here Δ and d represent the clock drift between two nodes and propagation delay, respectively. Then, at time T3, node B sends back an "acknowledgement packet" to node A. This acknowledgement packet contains the level number of node B and the time values T1, T2 and T3. Receiving the acknowledgement packet at time T4, node A has sufficient information to calculate the clock drift and propagation delay. This calculation is performed as follows:

$$\Delta = \frac{(T2 - T1) - (T4 - T3)}{2} \quad (3)$$

$$d = \frac{(T2 - T1) + (T4 - T3)}{2} \quad (4)$$

Ganeriwal et al. implemented both TPSN and RBS on Berkeley Motes to compare the performances of the two algorithms. As indicated above, error of RBS on Berkeley Motes was 11 μsec. However, the authors reported a 29.13 μsec error for RBS and 16.9 μsec error for TPSN on the same platform. The authors argue that the 6.29±6.45

μsec precision reported for RBS results from the superior crystals used in IPAQs. They claim that sender's contribution to the total non-determinism is very little to consider when the packets are time stamped at MAC layer.

3.1.3 Flooding Time Synchronization Protocol

Maroti et al. proposed a network-wide time synchronization algorithm, Flooding Time Synchronization Protocol (FTSP) [13], which utilizes MAC layer timestamping to remove the non-determinism with the send time and linear regression to compensate the clock skew. In contrast to TPSN and RBS, FTSP uses a one-way messaging scheme to achieve pairwise synchronization. FTSP can be investigated by decomposing it into two procedures: time stamping and clock drift management.

FTSP utilize a single radio message to synchronize a group of receivers. Time stamping part of FTSP requires this single radio message to be timestamped right before it is sent on the sender side and right after it is received on the receiver side. For the time stamp on the transmitted message to be closer to the real transmission time, time stamping on the sender side is performed before the bytes containing the time stamp are transmitted. Furthermore, timestamp value is normalized by subtracting an estimated transmission time and the average of the jitter on the interrupt handling time. On the receiver side, byte alignment time is computed from the transmission time and bit offset. Then this value is subtracted from the local time to obtain the correct time stamp to be embedded into the message by the receiver. The couples of timestamps embedded by the sender and the receiver constitute synchronization points. To achieve better precision, radio messages can be time stamped more than once.

Authors report that with 6 time stamps the synchronization error is improved from tens of microseconds to 1.4μsec.

After obtaining a set of synchronization points, offset of the two time stamp values in each synchronization point is calculated. Then, linear regression is used to find the line L best approximating the change in the offset values versus time. The synchronization error is calculated by subtracting the value of L at a specific time instance from the offset value at that time instance. Greater number of synchronization points cause the L to approximate the dataset letter which in turn results in improved precision.

As indicated in reference [13], Maroti et al. implemented the FTSP with a scenario of 60 Mica motes deployed in a 5x12 array where the nodes were able to communicate with their neighbors only. The authors report that the complete synchronization was achieved in 14 minutes. Average and maximum synchronization were 2μsec and 10μsec, respectively.

3.1.4 Time Diffusion Protocol

Su and Akyildiz proposed a network-wide time synchronization algorithm, Time-Diffusion Protocol (TDP) [18]. TDP allows a network to reach an equilibrium time and maintain a small time deviation tolerance from the equilibrium time. The time deviation tolerance can be adjusted based on the application of the sensor networks. TDP architecture, as illustrated in Figure 3, consists of many different algorithms and procedures. Since a detailed description of each of these algorithms and procedures is beyond the scope of this thesis, only the main characteristics of TDP are discussed in the following part of the text.

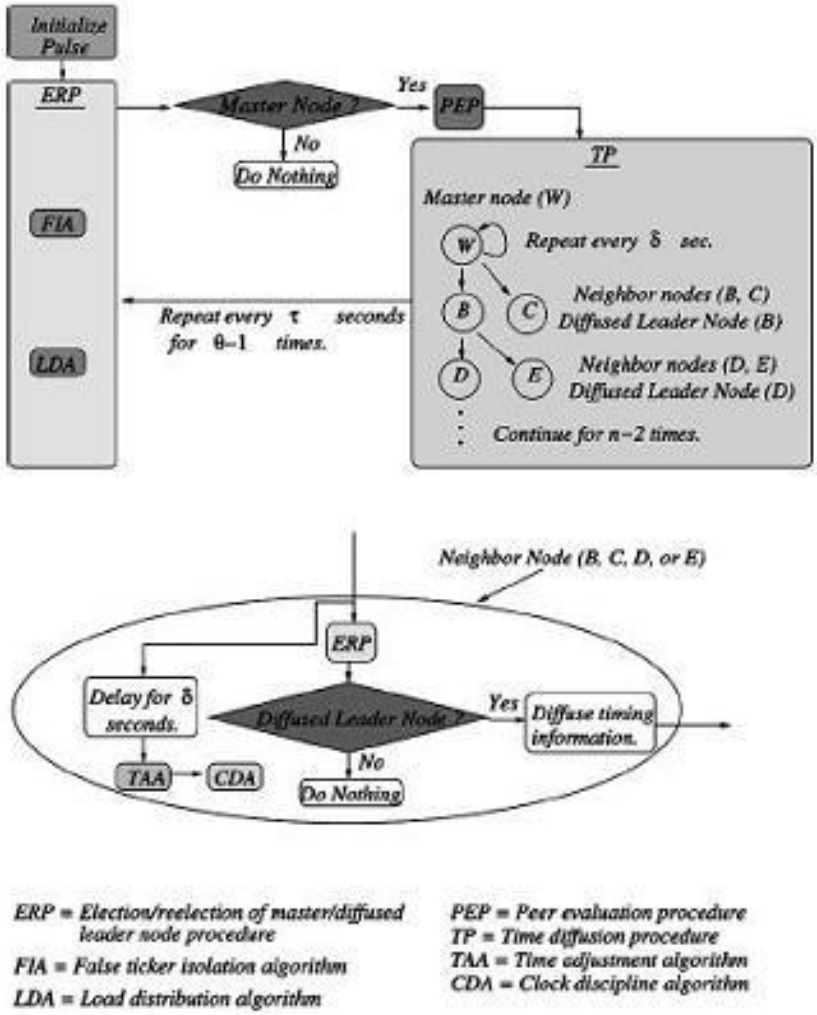


Figure 3 - TDP Architecture [18]

Figure 4 shows the time schedule of TDP. Master nodes are reelected at every τ seconds. The master node broadcast timing information to the neighbor nodes at every δ seconds. This timing information is used as timing reference and every time the master node sends a timing information packet, the neighbor nodes self-determine to become diffused leader nodes which broadcast timing information to their neighbors. The time deviation tolerance depends on the

duration of the TDP active period. This parameter can be adjusted for different types of sensor networks.

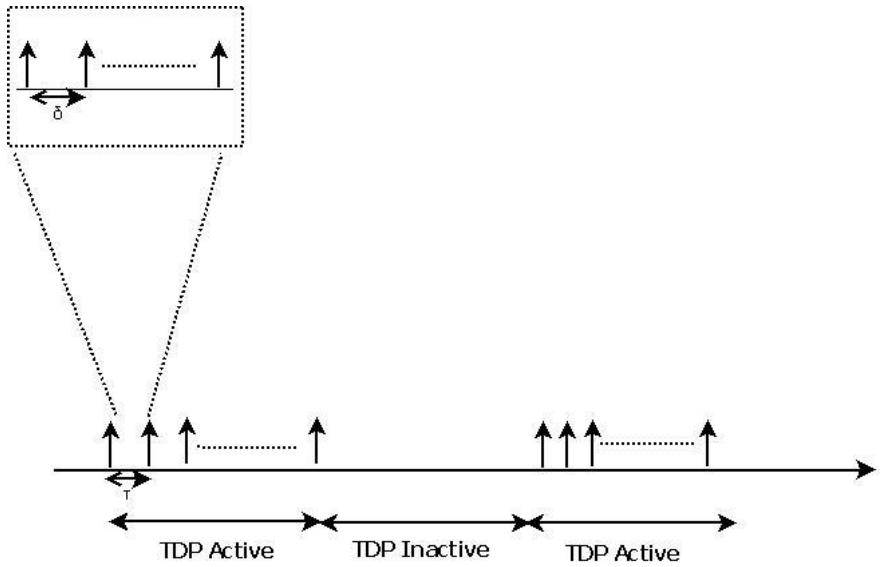


Figure 4 - TDP active/inactive schedule (Adapted from [18]).

TDP starts with an "initialize" pulse sent by the sink through direct broadcast or multi-hop flooding. Receiving the initialize pulse, nodes self-determine to become the master node with the election/reelection of master/diffused leader node procedure (ERP). ERP consists of the false ticker isolation algorithm (FIA) and load distribution algorithm (LDA) as shown in Figure 3. Then, the master nodes elected by ERP start the peer evaluation procedure (PEP) while other nodes do nothing. The main objective of PEP procedure is to remove false tickers, clocks that deviate from their neighbors, from becoming a master node or a diffused leader node. After procedure PEP, the elected master nodes start to diffuse the timing information messages at every δ for duration of τ seconds. The neighbor nodes which receive timing information messages self-determine to become a diffused leader node using ERP and they also adjust their clocks by

implementing the time adjustment algorithm (TAA) and clock discipline algorithm (CDA) after waiting δ seconds.

In [19], authors compare TDP and RBS. Lee et al. presents some simulation results which show the large-scale performance of TDP is better than RBS can provide. On the other hand, for small WSNs, RBS is a more accurate method. The main advantage of TDP over RBS is the decrease in the infrastructural overhead. Utilization of a hierarchical structure in TDP reduces the message traffic necessary for synchronization.

3.1.5 Secure and Resilient Clock Synchronization

In their recent work, Sun et al. proposed a network-wide synchronization model which includes some techniques to provide redundant ways for each node to synchronize its clock with the common source [17]. As a result of multiple paths from nodes to the source node, the synchronization model can tolerate partially missing or false synchronization information provided by compromised nodes. Two types of techniques are developed using this method: level-based and diffusion-based clock synchronization.

Level based clock synchronization consists of level discovery and synchronization phases. The source node initiates the level discovery phase by unicasting a level discovery message to its immediate neighbors. A level discovery message contains the sender's identity and its level number, authenticated with the pairwise key shared between the sender and the receiver. After receiving an authenticated level discovery message from the source node S, each neighbor of S sets its own level to 1. Then the neighbor nodes send level discovery messages to their neighbors except the source node. After the

foundation of the hierarchical structure the synchronization phase is initiated by the source node. Synchronization messages sent by the source node to its neighbors periodically, since the clocks of the sensor nodes tend to drift with time. The nodes receiving the synchronization message begin pairwise synchronization with S. Then they send unicast their own synchronization messages. Consider a node, i, with level greater than 1. When this node receives a synchronization message from the parent node j, node i calculates the pairwise clock difference with node j. Node i uses this information to calculate a candidate source clock difference by $\delta_{i,S}^{(j)}=\delta_{i,j}+\delta_{j,s}$. Node i needs 2t+1 source clock differences to tolerate t malicious nodes. Node i sets the source clock difference $\delta_{i,s}$ as the median of 2t + 1 candidate source clock differences. Level based clock synchronization is aimed to be used for static networks.

The authors propose diffusion-based synchronization for dynamic sensor networks. Diffusion-based synchronization doesn't require any hierarchical structure to be established. In this scheme, the source node initiates the synchronization by sending unicast a synchronization message to its neighbor nodes. Receiving one of this messages, each neighbor node synchronize its clock difference and send its own synchronization messages to its neighbors, again except for the source node S To tolerate up to t malicious neighbor nodes, each node should receive 2t + 1 synchronization messages to update its source clock difference by calculating the median of 2t + 1 candidate values.

Sun et al. also presented an extended model for multi-source environments. In this scheme, for a node to tolerate s malicious source nodes, 2s + 1 source clock difference values is obtained by

that node. Then the real source clock difference is determined by calculating the median of 2s + 1 candidate source clock differences.

Sun, Ning and Wang report different synchronization error values for changing number of malicious neighbor nodes and malicious source nodes. They present simulation results for sensor networks of 50, 100, 150 and 200 nodes. For each of these node numbers, level-based and diffusion-based algorithms are tested separately. Furthermore, the number of malicious neighbor nodes for each node is changed from 0 to 3. Figure 5 shows the results of the simulation. The results show that the level-based algorithm results in much better precision than the diffusion-based one do. They relate this difference in precision to the greater time drift during the diffusion-based synchronization procedure.

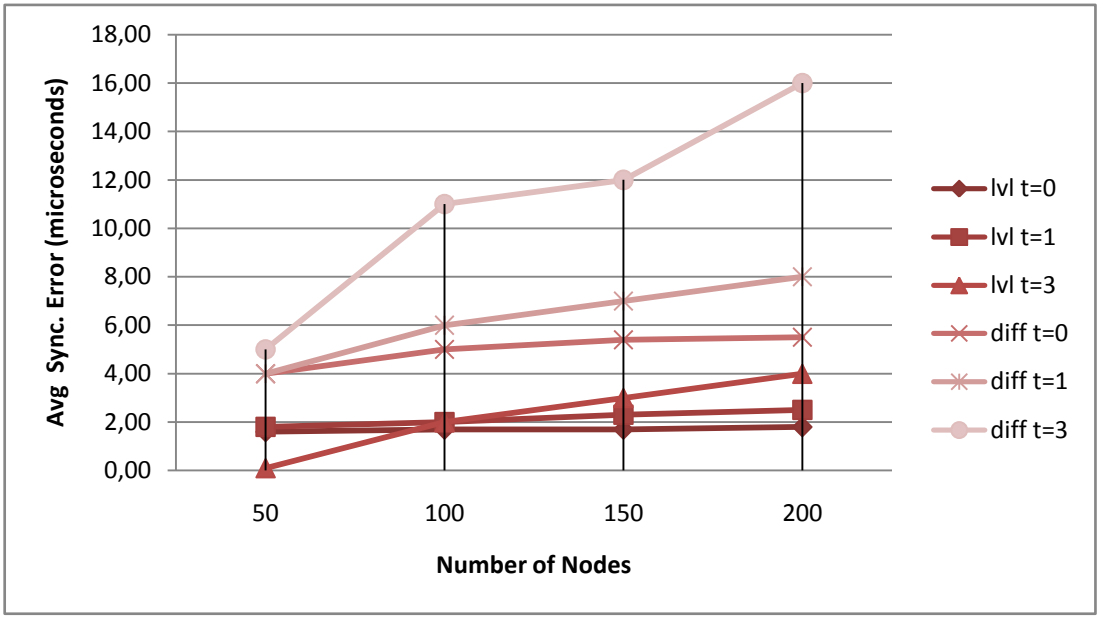


Figure 5 - Simulation results for the algorithms of Sun et al (Adapted from [17]).

3.1.6 A Tree-Based Hierarchical Time Synchronization Scheme

He and Kuo [16] have proposed a tree-based time synchronization scheme for WSNs. In the first part of this scheme, a spanning tree of all nodes in the network is created. Then, this spanning tree is divided into multiple subtrees. The subtrees are synchronized by one of the two algorithms proposed by the authors. When all the subtrees of the network are synchronized, the synchronization of the entire network is accomplished. By synchronizing the root node, the node at the top of the tree hierarchy, to an external clock, the synchronization of the whole network to an external clock can be achieved.

For the first part of the synchronization scheme, a root node assigns itself level 0. Then it begins to broadcast children-find messages through the network. The immediate neighbors of the root node receive these messages and assign themselves with one greater level than the root node has level 1. Each node with a level assigned to it broadcast its own children find messages and each neighbor node which receives a children find message assign itself one greater level than level of the sender. A node should neglect the children-find messages after the first message it receives. The spanning tree is divided into s subtrees after its creation. Each subtree includes a father node and several child nodes. The subtrees are identified by the father node's level. Figure 6 illustrates an example for the creation of such subtrees.

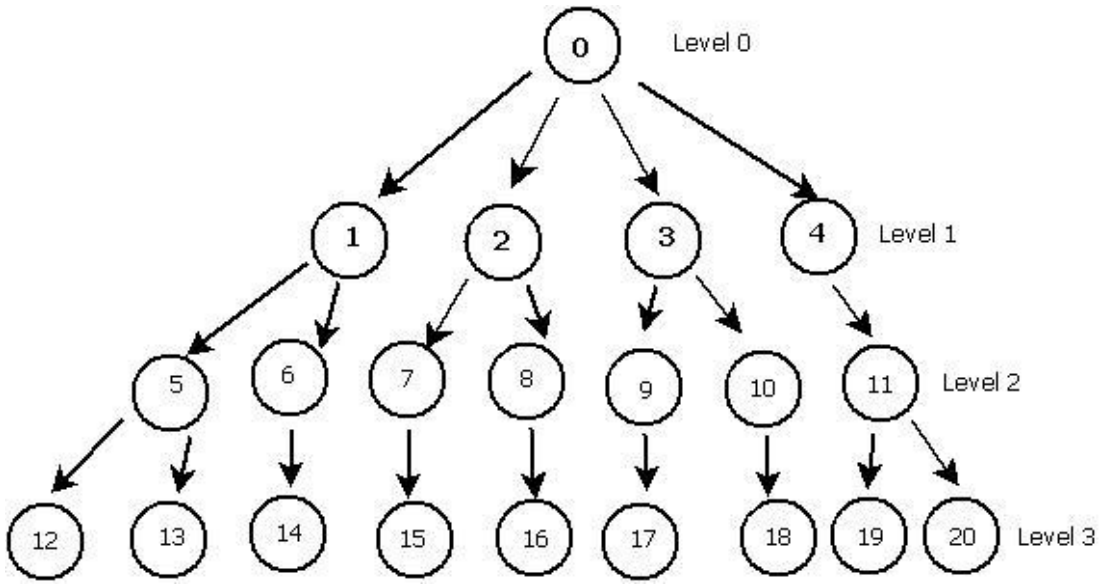
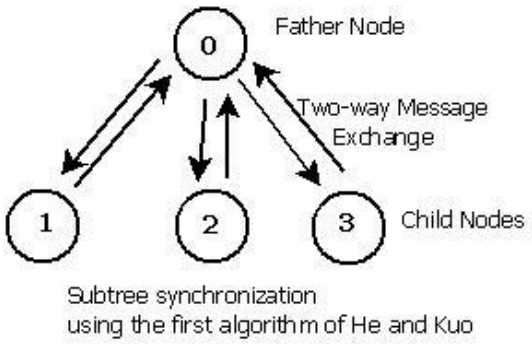


Figure 6 - Dividing the spanning tree into multiple subtrees (Adapted from [16]).

The authors propose two different algorithms to achieve the synchronization of subtrees. In the first algorithm, the father node in each subtree performs two-way message exchange with each of the children nodes. This is the same procedure with the sender-receiver synchronization of TPSN. After all the children nodes in the subtree are synchronized to the father node, the synchronization of the subtree is achieved. The second algorithm is the improved version of the first one. In this scheme, the father node initiates two-way message exchange with only one of the children nodes. The remaining part of the subtree synchronization resembles the structure of RBS. The father node broadcasts reference clock-adjusting packets to all the children nodes, including the synchronized one. Receiving the clock- adjusting packets nodes exchange their clock information and try to estimate the clock offset values. Figure 7 shows both of the algorithms. The authors argue that the latter approach results in better synchronization performance.



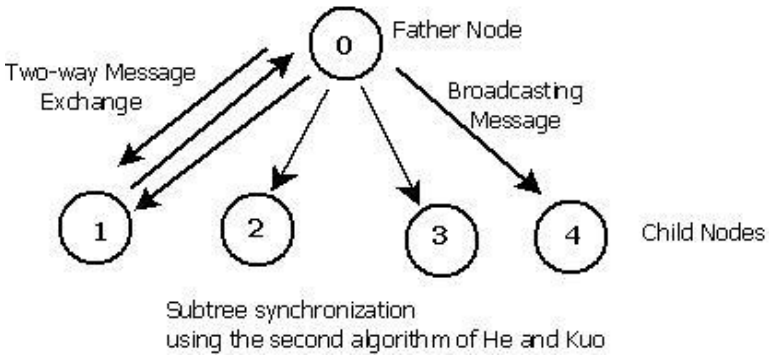


Figure 7 - Two approaches for subtree synchronization (Adapted from [16]).

The authors use simulation to present a comparison of the synchronization performances of the proposed scheme and RBS. They argue that, for a node number of 200, the synchronization delay of the proposed scheme provides 50% improvement over RBS where the synchronization error improvement is more than 30%. For 200 nodes, the reported synchronization error of the proposed scheme is approximately 16μsec while this value is 24μsec for RBS.

3.1.7 Pulse-sync Time Synchronization

In their work with a recent date, Lenzen et al. argue that for most of the work and theoretical lower bounds on timing accuracy in WSN time synchronization literature are based on too harsh assumptions to describe the theoretical limits of clock synchronization. It is claimed by the authors of [30] that jitters are random in nature and

clock drift does not vary arbitrarily, at least, the range of change in these parameters is narrow when a specific class of distributed systems is chosen.

The authors criticize the approach that FTSP utilize for time synchronization, because of the possible cumulative propagation of synchronization error through the network, e.g. self-amplification of the error. They propose a method called PulseSync in which a flooding pulse mechanism is used. The provider of the first pulse becomes the root of synchronization and it can be used to spread UTC time to the entire network. To avoid the collision which may result from uncontrolled broadcast of time pulses, flooding is scheduled. The authors present test results which are obtained by using 20 Micaz nodes as the test bed. Test results show that the proposed algorithm results in a maximum time synchronization error value of 38 μs.

3.2 Selected WSN Time Synchronization Algorithms

Among the synchronization schemes which are described in the preceding part of the text, two algorithms, RBS and TPSN, have been benchmarked for evaluation of the recently proposed time synchronization algorithms, because of their reported synchronization accuracies.

CHAPTER 4

NETWORK TIME PROTOCOL

Network Time Protocol (NTP) which was designed to synchronize clocks of hosts and routers in the internet is one of the oldest protocols. The Version 1 specification of the protocol was published by David Mills of the University of Delaware in July 1988. NTP is currently maintained by Mills, his graduate students and volunteers. NIST estimated that 10-20 million NTP servers and clients deployed in the Internet and its tributaries all over the world, e.g. every Windows XP computer is an NTP client [1].

The reason for the commonness is the available software implementations and public time servers. NTP allows Internet users to have accurate time synchronization without having expensive hardware. Furthermore, open implementations of the protocol make it possible to customize it for requirements. Currently, NTPv4 is the latest version of the protocol and the specification is bundled with a reference implementation.

The goals of NTP defined by Mills' group are [1],

- Provide the best accuracy under prevailing network and server conditions.

- Resist many and varied kinds of failures, including two-face, fail-stop, malicious attacks and implementation bugs.
- Maximize utilization of Internet diversity and redundancy.
- Automatically organize subnet topology for best accuracy and reliability.
- Self contained cryptographic authentication.

4.1 NTP Architecture

NTP uses a hierarchical structure where each NTP server is assigned a level. Levels are called stratum and have numbers to state the place of the server in the hierarchy, in other words the distance to the synchronization root. Each computer in the hierarchy, except the root server, synchronizes itself to one of the computers in the upper layers.

Stratum 0 devices are high accuracy clock sources like atomic clocks or reference radio broadcasts. GPS clocks can also be used as Stratum 0 devices, since they provide Pulse-Per-Second (PPS) signals which have global nanosecond-level accuracy [20]. Stratum 1 computers are directly connected to Stratum 0 devices. They act as servers for Stratum 2 computers. Stratum 2 and lower-strata computers synchronize themselves to the one higher stratum computers.

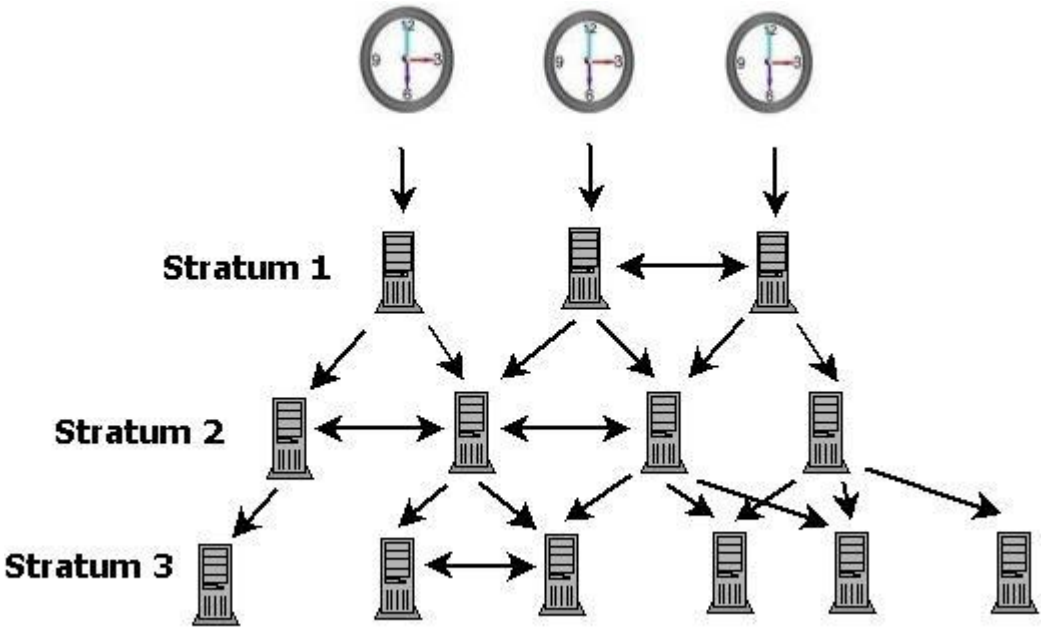


Figure 8 - NTP subnet topology with stratum no.

Synchronization process in NTP is designed to reduce jitter and avoid misleading servers. For these purposes, Stratum 2 and lower-strata computers initiate synchronization with more than one time server and create the best data sample using the samples collected from these servers. Furthermore, Stratum 1 time servers which cause faulty synchronization results are dropped to Stratum 2 level by Stratum 2 computers.

When we look at the software architecture and algorithms used in NTP, implementation of an adaptive mechanism to select best time sources and preserve clock stability can be seen. Redundancy of time servers allows the computer to choose the most stable time source for synchronization. NTP use a filter for each time source to select the best offset sample from eight samples collected from that source. Selected sample is processed by Selection and Clustering Algorithms to pick best truechimer (time sources which provide stable samples)

and discard falsetickers (inconsistent time sources). The weighted average of picked truechimers is calculated by the Combining Algorithm. Clock jitter is minimized by Clock Discipline Algorithm implemented at the last stage.

4.2 NTPv4 Reference Implementation

NTP consists of a number of data structures and algorithms to improve time data source quality and local clock stability. In this section, NTP Version 4 reference implementation is discussed to make the reasoning for simulation parameters chosen in next sections clear. For this part NTPv4 Reference Implementation specification is used as a guide. On the other hand, to keep the discussion in the scope of measurement networks we are trying to find a synchronization solution for; only the parts used in simulations are mentioned.

4.3 Modes of Operation

There are three modes of operation identifying the role of a computer in a synchronization subnet; primary server, secondary server and client. Primary servers (stratum 1) are directly connected to a reference clock like GPS or a reference radio broadcast. Secondary servers may have multiple downstream and upstream servers or clients [21]. Clients request synchronization from one or more servers where they can't provide synchronization to any other computer.

Table 1 - NTP Modes of operation

Association Mode	Assoc. Mode Code	Packet Mode
Symetric Active	1	1 or 2
Symetric Active	2	1
Client	3	4
Server	4	3
Broadcast Server	5	5
Broadcast Client	6	NA

Besides mode of operations, there are three protocol variants for NTP servers, symmetric, client/server and broadcast. As it can be seen from Table 1, variants are assigned association mode and packet mode indicators. In the client/server variant a client sends mode-3 packets to a server and the server responses with mode-4 packets. Servers provide synchronization to more than one client, but it doesn't accept synchronization from them. In the symmetric variant, peers act as both clients and servers to each other. In this variant, symmetric-active packets can be exchanged between symmetric-active associations, or a symmetric-passive association can be activated by a symmetric-active packet and sends a symmetric-passive packet as response.

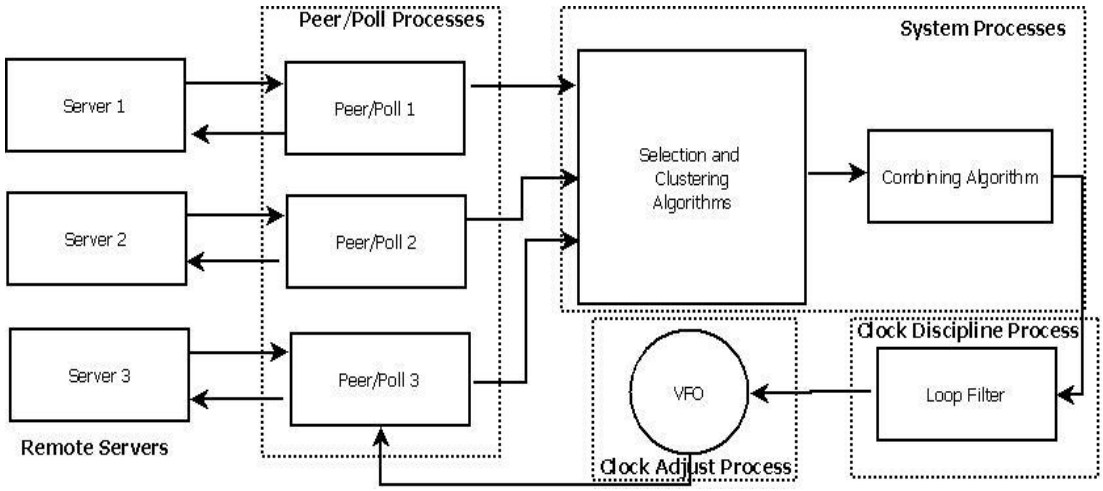


Figure 9 - NTP protocol architecture (Adapted from [1]).

4.4 Implementation Model

NTP implementation model consists of four main processes; poll process, peer process, system process and clock discipline process. Poll process is used for transmitting messages to clients. Peer process is used for receiving messages from servers. The client sends a synchronization request to a number of servers. The request packet is processed by the servers and sent back to the client.

System process includes selection and clustering algorithms which is used for choosing the most stable time source among a set of samples. Byzantine methods (e.g. Marzullo's Algorithm) are used for discarding falsetickers. Remaining truechimer sources are also processed by combining algorithm to obtain a single offset value.

Clock discipline process is used for stabilizing the system clock, which is represented as a VFO (Variable Frequency Oscillator) in the Figure 9. Clock discipline process works in cooperation with clock adjust process which compensates the clock offset and maintain fixed frequency.

4.5 Data Types

There are three NTP time formats; 128-bit date, 64-bit timestamp format and 32-bit short format. 128-bit format format spans 584 billion years, 64-bit format spans 136 years and 32-bit format is used for delay and offset calculations only. 128-bit, 64-bit and 32-bit formats result in .05 attosecond, 232 picoseconds and 1.5 microsecond resolution, respectively. Seconds field of the date format is divided into two parts, the era field and the timestamp field to make it easier to convert between data types. In the following

chapter, simulations of NTP uses 32-bit short format for time representation, since the resolution is sufficient to compare with other synchronization methods simulated.

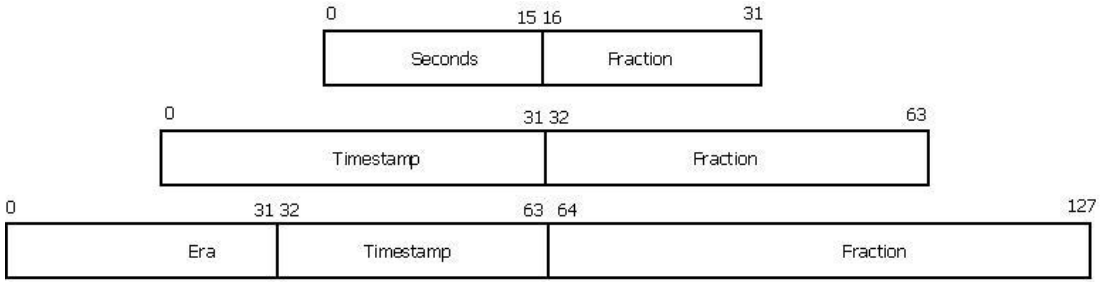


Figure 10 - NTP time formats (Adapted from [1])

4.6 Data Structures

NTPv4 requires some global parameters to be defined for a standards-compliant implementation. Some of these parameters are globally constant. For instance, port number is assigned by IANA and it is same for all implementations. On the other hand, some of the parameters are allowed to be reconfigured or change during operation. The list of global parameters is shown in Table 2.

Table 2 - NTP global parameters

Name	Value	Description
VERSION	4	Version number
TOLERANCE	15e-6	Frequency tolerance (Φ) (s/s)
MINPOLL	4	Minimum poll exponent (16 s)
MAXPOLL	17	Maximum poll exponent (36 h)
MAXDISP	16	Maximum dispersion (s)
MINDISP	.005	Minimum dispersion increment
MAXDIST	1	Distance threshold (s)
MAXSTRAT	16	Maximum stratum number
PORT	123	NTP port number

Besides global parameters, NTP requires some packet header variables to be defined. NTP packet header has some fields which are manipulated during synchronization process. NTP packet header is placed after the IP, UDP and underlying protocol headers.

The following variables are also used in NTP simulation which is discussed in the following chapter.

- org: The time the client sent the synchronization request
- rec: The time the server received the synchronization request
- xmt: The time the server transmitted the timestamp
- dst: The time the client received the timestamp
- rootdelay: Total roundtrip delay to the reference clock, in NTP short format
- rootdisp: Total dispersion from the reference clock, in NTP short format

4.7 On-Wire Protocol

NTP on-wire protocol is the main mechanism where servers and clients exchange synchronization requests and timestamps. It is basically a pairwise synchronization scheme where a client initiates synchronization procedure by transmitting a request packet. The server receives the synchronization request and processes this packet by manipulating the necessary fields in the NTP header of the received packet. The server then sends the packet back to the client. From this point on, client has sufficient data to calculate roundtrip delay and offset to that particular server.

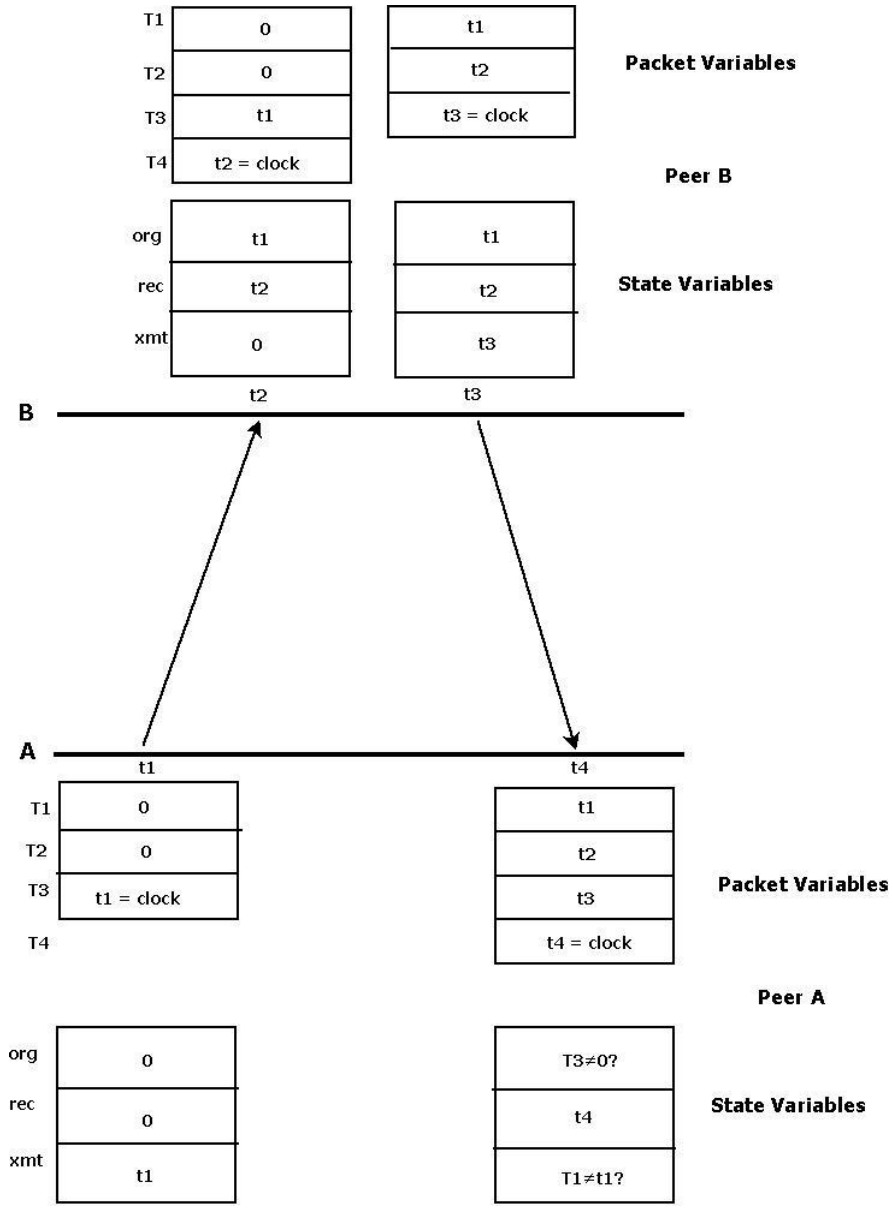


Figure 11 - NTP On-wire protocol (Adapted from [1]).

On-wire protocol uses four timestamps, T1, T2, T3 and T4. It also uses three state variables org, rec, xmt. As you can see from the Figure 11, as the packet is exchanged between peers A and B, it is timestamped on every transmission and reception event.

The four most recent timestamps are used for calculating relative offset of A to B, θ, and the roundtrip delay, δ:

$$\boldsymbol{\theta = T(B) - T(A) = \frac{1}{2}[(T_2 - T_1) + (T_3 - T_4)]} \qquad (5)$$

$$\boldsymbol{\delta = T(ABA) = (T_4 - T_1) + (T_3 - T_2)} \ . \qquad (6)$$

4.8 Peer Process

Peer process runs when a client receives a server packet. As part of the peer process Clock Filter Algorithm is used for aggregating incoming server packets and finding the samples which is most likely to represent the correct time. As the result of Clock Filter Algorithm, offset, roundtrip delay and clock jitter is calculated.

There is a peer process, poll process and association for each server [21]. The statistical variables calculated by peer process are used by system and poll processes. Furthermore, if it is decided to be implemented, authentication also takes place in this process. Peer process is a very large process and its detailed explanation is beyond the scope of this part.

4.9 System Process

Statistical samples, offset, roundtrip delay and jitter created by the Clock Filter Algorithm are processed by clustering and selection algorithms of system process. Selection algorithm scans all the associations and discards falsetickers. Clustering Algorithm eliminates the samples which are statistically furthest from the centroid and leaves a minimum number of truechimers as result.

Combining Algorithm aggregates the resulting samples from Clustering Algorithm to obtain a final offset value and selects the most stable association as peer for performance evaluation.

Selection algorithm used for discarding falsetickers leaving a minimum number of truechimers is proposed by Keith Marzullo [22].

4.10 Marzullo's Algorithm

Marzullo's Algorithm is an agreement algorithm which uses Byzantine Principles for estimating accurate time from noisy sources. The algorithm determines an optimal interval from a set of estimates with confidence intervals.

For Marzullo's Algorithm to be implemented, clock source should be represented with their maximum error (e.g. confidence interval). For instance, 10±2, 12±1 and 11±1 can be represented as [8,12], [11,13] and [10,12], respectively. As it can be seen from the Figure 12, the interval consistent with the largest number of sources is [11,12].

Marzullo's Algorithm begins with preparing a table of intervals of [c-r,c+r], where c is the center of the interval. Each interval is represented by two tuples, in the form of <offset, type>. One tuple is for the beginning of the interval and has type -1, <c-r,-1> and the other tuple is for the end of the interval with type +1, <c+r, +1>.

- The algorithm use the following variables with the table of tuples:
- best: The largest number of overlapping intervals found.
- cnt: The current number of overlapping intervals.

- beststart: The beginning of the best largest interval found so far.
- bestend: The end of the best largest interval found so far.
- i: An index

The algorithm iterates like follows:

1. The table of tuples is built.
2. The table is sorted by offset.
3. Initializing variables, best=0, cnt=0.
4. LOOP going in ascending order for each value on the table:
5. cnt=cnt-type[i]
6. if cnt>best then best=cnt, beststart=offset[i], bestend=offset[i+1]
7. END LOOP, return [beststart,bestend] as the optimal interval.

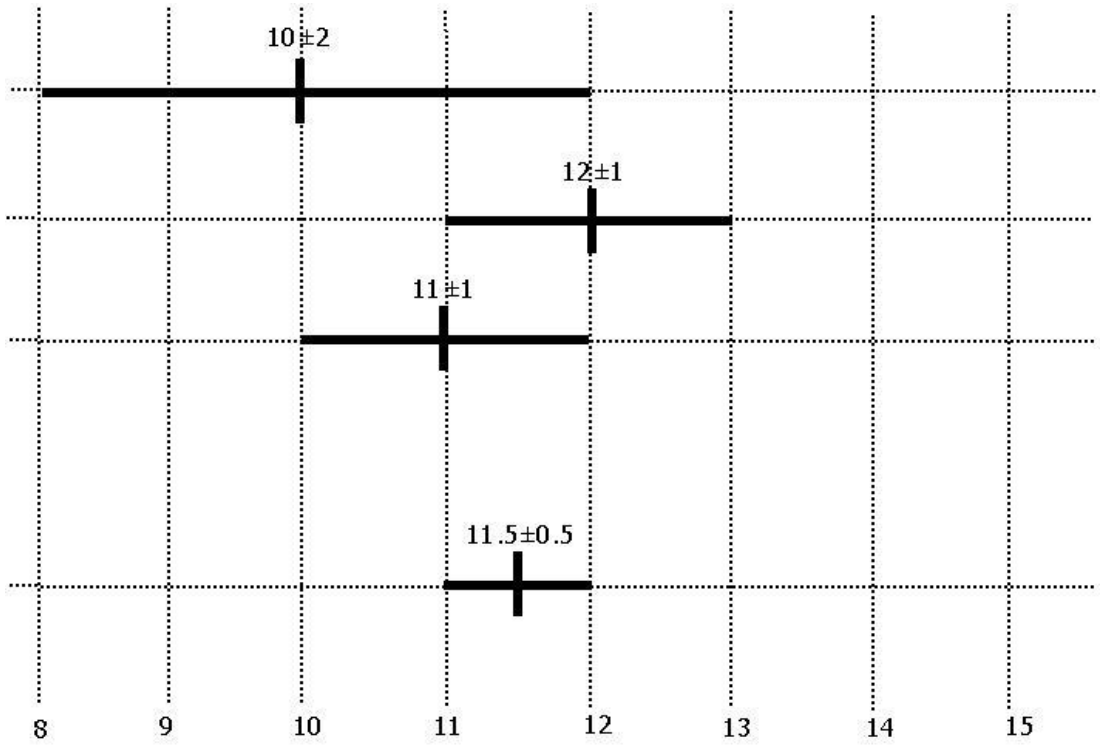


Figure 12 - Marzullo's Algorithm example

Marzullo's Algorithm is preferred for NTP, since it is efficient in time and space. Time efficiency of the algorithm is O(m log m) and space efficiency is O(m) [24].

4.11 Poll Process

In NTP reference implementation, associations support poll process which is used for sending packets in symmetric, client/server and broadcast modes. This process runs continuously, whether or not the services are reachable.

Polling frequency is determined by one of the poll process variables, host poll exponent hpoll, which is represented in log2 seconds. Besides the polling frequency, poll process has some variables like the last time a packet sent, next time a packet will be sent, the number of times in which the services were unreachable and a shift register for keeping the number of sent packets.

4.12 Synchronization Accuracy

Despite the fact that NTP is one of the oldest protocols in Internet, there is not too many measurement records showing how accurate NTP results for different network topologies or delay values. One comprehensive survey about NTP is made by Mills' team in 1999 and shows some important characteristics of the protocol [23].

In [23], the cumulative distribution function of absolute time offsets is presented for a total of 38,772 servers running NTPv2 and NTPv3. The median of offsets is 23.3 ms, mean is 234 ms and maximum offset is 686 ms. Another important statistics which is shown in [23]

is the cumulative distribution of peer-peer absolute roundtrip delays. This data was collected using 182,538 samples where synchronization distance exceeding 1 second had been discarded, since this level of delay is excluded by the specification of NTP. Curves show median delay of 118 ms, mean delay of 173 ms and max delay of 1.91 s.

CHAPTER 5

EVALUATION OF WSN ALGORITHMS AND NTP REGARDING THE NEEDS OF TNPQP

The measurement networks whose needs this study aims to address consist of large number of nodes which are spread to a large geographical area. The nodes constituting the network are equipped with sensors or other types of measurement devices from which data is collected and transferred to a measurement center via the network. For the data collected from distributed nodes to be combined to a meaningful high-level result, this process being called data fusion, each part of data obtained from different nodes should be associated with a correct time order.

To emphasize the importance of time synchronization in measurement networks more clearly, timing accuracy needed by TNPQP can be given as an example. In TNPQP, monitors placed on different cities of Turkey report their measurement data to NPQ center and the measurement data from different monitors is aggregated to calculate power quality parameters.

Test and Measurement Methods - Power Quality Measurement Methods standard IEC 61000-4-30, issued by the International Electrotechnical Commission (IEC), defines the parameters and methods to obtain reliable power quality measurement results. Measurement intervals and how the measured data is aggregated are also defined by the standard. The standard defines that the time

clock uncertainty for the PQ events and measurement intervals should be at most ±20 ms for 50 Hz power systems and 16.7 ms for 60 Hz power systems [3]. As explained in the preceding part of the text, computer clocks are not inherently accurate and a network time synchronization scheme should be implemented to achieve time synchronization accuracy to fulfill the requirements of this standard.

5.1 Combining Reference Broadcast and Hierarchical Pairwise Synchronization Using Traditional Wired Networks

TPSN and RBS are two WSN time synchronization methods which are experimentally proven by their authors to give better performance than NTP when implemented in wireless networks [11] [12]. These algorithms cannot be used in their basic form for traditional wired network. On the other hand, synchronization logic of these two algorithms may be useful for synchronizing traditional wired networks when the network partially consists of GPS attached nodes. In this part, a method which is based on the combination of reference broadcasting and hierarchical pairwise synchronization is proposed.

As mentioned before, RBS method uses reference broadcasts to synchronize multiple receivers. On the other hand, the method is aimed to synchronize wireless networks. Delays encountered in traditional wired network communications exhibits non-determinism of greater magnitudes and broadcasts require a common medium to be useful as references. Therefore, RBS method isn't expected to result in high accuracy time synchronization in Wide Area Networks (WAN) in its basic form.

Reference broadcasting logic of RBS can be utilized in WANs when the goal is to improve the GPS time synchronization accuracy. PPS

signals can be considered as global reference broadcasts. Each node receiving a PPS signal sends the reception time to a server. Server, which is assumed to have a local clock value equal to UTC, compares its own PPS reception time to the PPS reception times of the clients and sends the difference to them. Receiving the response message, each client corrects its local clock.

In the absence of PPS signal, a software scheme should keep the clocks of the clients synchronized. For wired networks where some of nodes are equipped with GPS receivers, pairwise synchronization can be used synchronize a node with PPS signal and a node with no PPS signal. The track of nodes with active PPS signal can be kept in the server and provided to the nodes which need a pair to synchronize themselves.

Combining pairwise synchronization method and PPS signals as reference broadcasts, treated by the clients as globally broadcasted reference pulses, a synchronization method to fulfill the requirements of National Power Quality Project can be designed. In the following parts of the text, simulation results and some suggestions for a real-world implementation of the proposed scheme is discussed.

The real-world implementation of the proposed method requires GPS receivers to be attached to the hosts through low-latency interfaces. Therefore, parallel port can be an appropriate choice for connecting the receivers to the hosts. Where it's viable, serial port connection to GPS receivers can also be used to retrieve high precision timebase by some hosts.

As indicated before, the hosts which are planned to be placed in measurement locations in scope of the NPQ Monitoring Project runs Linux operating systems. Parallel port interrupts can be captured in kernel level to decrease the latency and non-determinism.

5.2 Utilization of GPS Pulse-Per-Second Signals To Improve Synchronization Accuracy of NTP

A Pulse-Per-Second (PPS) signal is an electrical signal which indicates the start of a second. One of PPS signal providers is GPS. Some GPS receiver devices output PPS signals. This feature of GPS is commonly used for time synchronization purposes.

In [25], the author investigates the relative precision of PPS signals of three different GPS receivers which are accessible on the market (Trimble Lasen LP, Motorola M12+ Oncore and Sigtec MG5001). The precision of PPS signals provided by three different GPS receivers are compared to a reference clock (Leica CRS1000). Measurement results reported by the author shows that the maximum offset of three PPS providers over 1 hour is 1.5μs. When maximum synchronization results reported by the authors of the time synchronization methods discussed in the preceding parts of the text are considered, this level of error on PPS signal timing precision is acceptable.

GPS PPS signals can be utilized on the Clock Disciplining part of the NTP architecture. Mills' team provides a drive for integrating PPS signals to existing NTP software suite [25]. This driver establishes an interface for PPS signals produced by GPS, radio clocks or a cesium clock. It is used for removing accumulated clock jitter.

Interfacing a PPS source to NTP drivers introduces some non-determinism caused by serial or parallel interface and the time consumed by the driver to process the PPS signals. PPS driver provided by Mills' team has mechanisms for reducing the effects of interface and processing time. On the other hand, to simplify the development, these algorithms aren't implemented in simulation setup. Instead, the delay caused by interface and driver is represented by a variable delay.

CHAPTER 6

SIMULATION OF NTP AND THE ALTERNATIVE TECHNIQUES

Two of Wireless Sensor Network time synchronization algorithms investigated in the preceding parts of the text, namely RBS and TPS, are claimed by their authors to result in better synchronization accuracy than NTP when they are implemented in wireless networks. NTP is a complicated protocol with algorithms to improve synchronization accuracy and clock stability. On the other hand, it may be useful to discuss the performance WSN synchronization methods for traditional wired networks, if synchronization is needed by a subnet with limited numbers of computers and well-defined network topology. For these type of networks WSN time synchronization algorithms may result in accuracy values comparable to the ones resulted by NTP.

A hybrid scheme which is obtained by combining reference broadcast synchronization and hierarchical pairwise synchronization was proposed in the preceding parts of the text. In this part, the simulation setup and details chosen as simulation parameters are discussed. Then, simulation results for the proposed method and NTP are presented.

6.1 Simulation Setup

Simulation of the proposed method and NTP is made in OMNet++ environment. INET framework, which is provided as a base for algorithms designed to work on internet, made it easier to focus on the development of the codes of the investigated algorithms rather than establishing a realistic simulation environment.

6.2 Network Topology

Simulated topology is designed to address the requirements of TNPQP. TNPQP monitoring network consists of power quality monitors which are positioned to different geographical locations of Turkey. The power quality monitors are connected to a server by the Internet and monitor-to-monitor communication is also done by using the Internet. Therefore, the topology illustrated in Figure 15 chosen to represent the TNPQP monitoring network where cloud figures represent the internet connection.

Flat network topology provided by INET framework is chosen for simulation. To simplify the implementation of the algorithms, the routers between clients are substituted with end-to-end delay units. Each client in the network is connected to a server through an end-to-end delay unit which models the delay encountered by the PPS reception time packets while they are transferred from the client to the server or vice versa.

In RBS, for each broadcast receiver in the network there is also a synchronization pair which it will communicate for pairwise synchronization. The connection between a client and its synchronization pair is also through an end-to-end delay unit.

Sample topology for 20 clients is shown in Figure 13. In the figure, end-to-end delay units are represented by clouds. The simulations are conducted with a 150-node structure to represent the NPQ Monitors placed in distant positions.

Proposed scheme requires a client to get a synchronization pair by sending a request packet to the server. For simulations, pair assignment is modeled as a fixed matching process. Server provides a fixed pair for each requesting client. For networks with fixed hosts, like TNPQP, this scheme should work well.

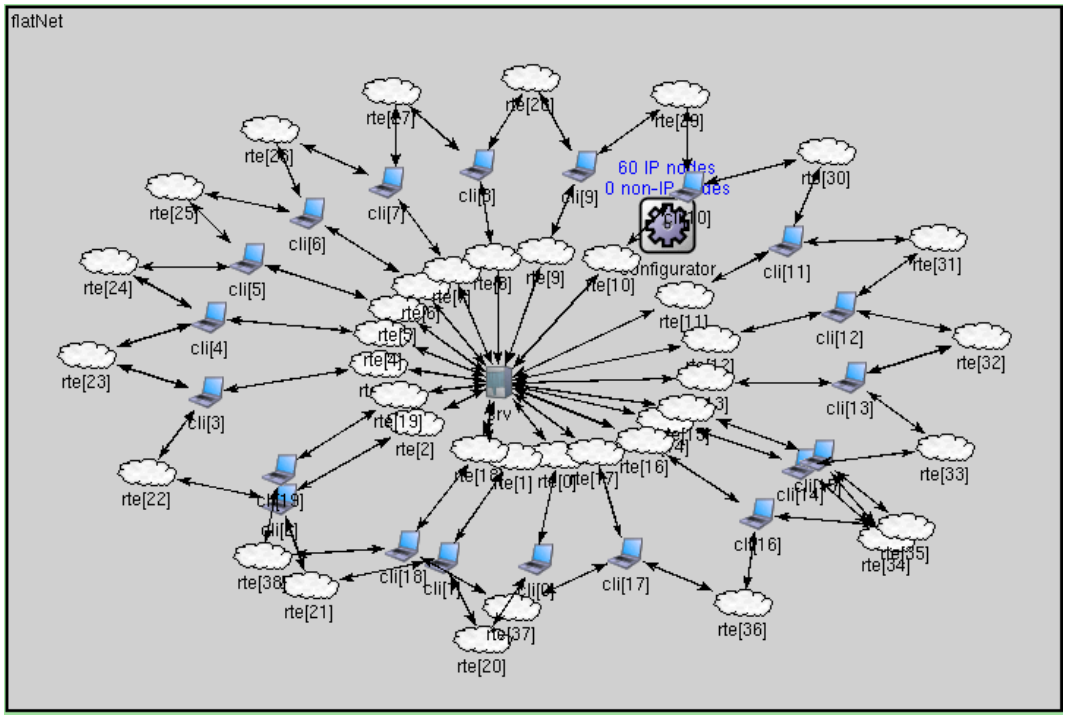


Figure 13 - Flat network topology with end-to-end delay units

6.3 Modeling the End-to-end Delay and Clock Drift

To establish a realistic model of end-to-end delay between hosts, literature which presents statistical models of end-to-end delay in

internet was investigated. Simulated scheme assumes UDP as the transport layer protocol, because of the smaller magnitude of non-determinism it provides. Therefore, the main focus of investigation was the experiments which use UDP for the transport layer.

In [26], Sanghi et al. investigate the end-to-end behavior of the Internet. The authors use 100000 UDP packets which are transferred from a client to a sink with regular time intervals. The packets are exchanged between three university campuses on three different states of USA. The authors present 85 ms average end-to-end delay and 62 ms standard deviation. These values are used in simulation to model the end-to-end delay units.

Another important parameter for obtaining a realistic simulation setup is the clock drift of simulated hosts. For determining this parameter, examples provided by NTP team in [1] are used. In [1], it is stated that, clock drift of a PC running Linux operating system can increase up to 11 PPM after powering up the system. This value is used in simulations as the drift of the local clock. Drift of a computer clock is also influenced by temperature. On the other hand, effect of temperature, which is stated in [1] to be 1 PPM per ˚C, is neglected in simulations. In all simulations, packet loss probability has been assumed to be nil.

6.4 Confidence of Simulation Results

To obtain meaningful results, synchronization algorithms were simulated on a 150-node network topology for multiple times. Confident stopping rule described in [27] was used to determine the required number of repetitions to obtain confident results.

The simulations are repeated for different times for each case to result in 90% confidence. As it's described in [27], the below formula was used to determine the number of repetitions:

$$n = \frac{4s_n^2 t_g^2}{\Delta^2 y_n^2}$$

where n is the number of repetitions, s_n^2 is the variance of synchronization error, t_g is a constant to define the level of confidence and Δ is the confidence interval and y_n is the observed mean of the measurements.

6.5 Simulation Results

In this part, simulation results of the proposed scheme and NTP with and without GPS clock discipline are presented and compared. Synchronization accuracy is chosen as the comparison metric.

6.5.1 Performance of the Proposed Method

In this part, simulation results of the proposed synchronization method, for run times of one minute, one hour and eight hours are presented. Besides, the change of maximum synchronization error with increasing number of nodes which are not equipped with a GPS receiver is shown and discussed.

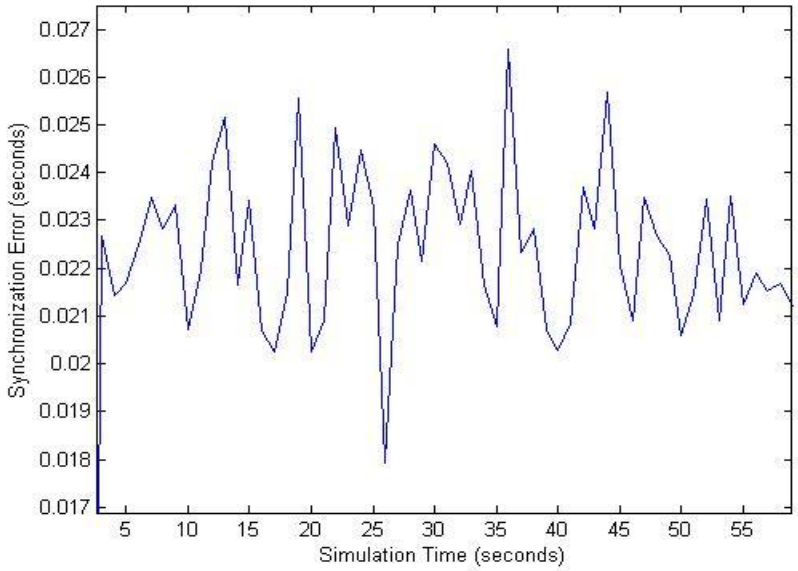


Figure 14 - Change of synchronization error within one minute

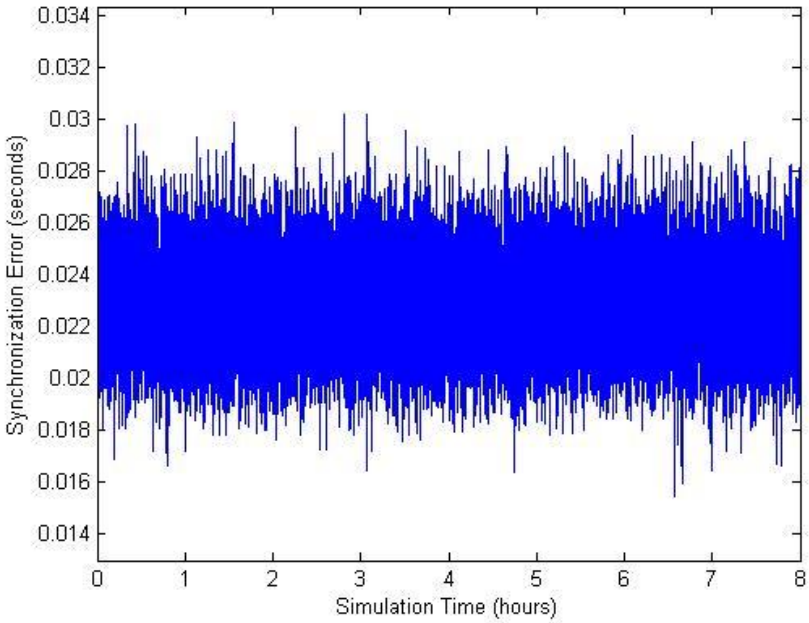


Figure 15 - Change of synchronization error within 8 hours

As it can be seen from Figures 14 and 15, maximum synchronization error is bounded in a way that it doesn't increase with time. These figures have been obtained for a single node without GPS reception in

a 150-node network to illustrate the fact that, within eight hours, the algorithm runs without affecting the stability of local clocks of the clients.

Figure 16 shows that the synchronization accuracy doesn't degrade dramatically with the decreasing number of GPS receivers within the measurement network. Maximum synchronization error for a 150-node network with all the nodes equipped with GPS receivers was 23 ms where the error was 72 ms for 75 of the nodes were assumed not to have GPS receivers. The results shown in Figure 16 are reported after 75, 30, 30 and 30 repetitions to obtain results with 95% reliability.

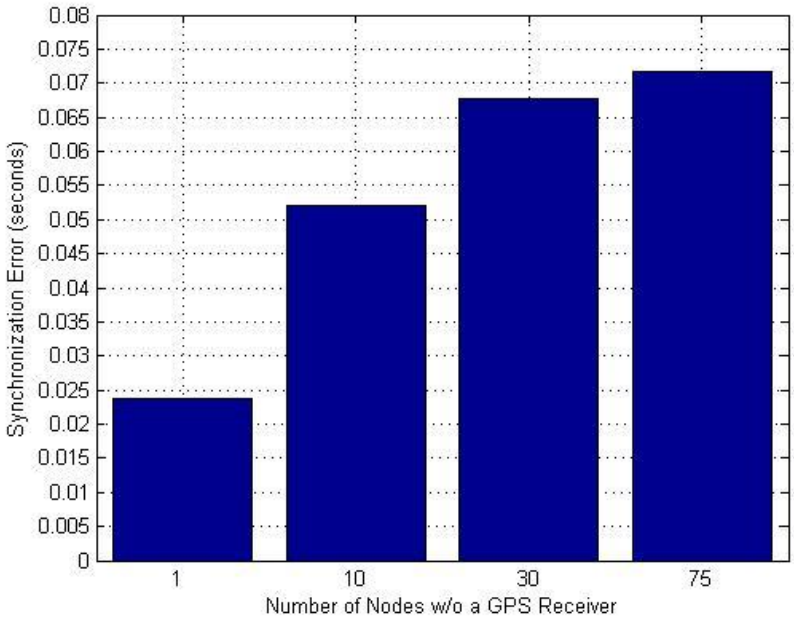


Figure 16 - Maximum synch. error for increasing number of nodes w/o GPS

6.5.2 Simulation Results for NTP

In this part, simulation results for NTP, implemented with and without GPS clock discipline are presented. In Figures 17 to 20, the change of average synchronization error is shown for one minute and eight hour periods. As it can be seen from the figures, synchronization error is bounded for both cases with and without GPS clock discipline. Figures 21 and 22 show the change of maximum synchronization error with increasing number of nodes with and without GPS clock discipline method, respectively.

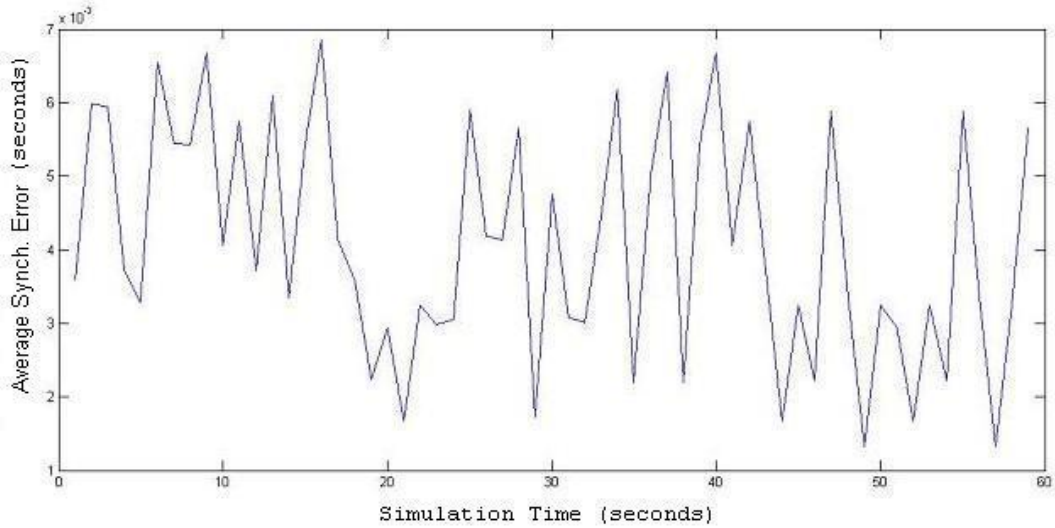


Figure 17 - Change of NTP with GPS sync. error in one minute

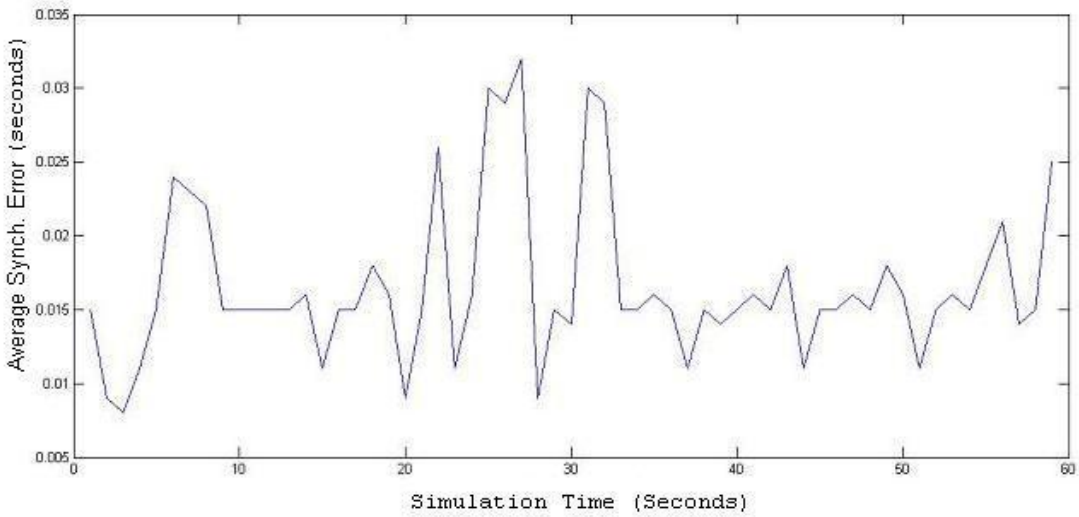


Figure 18 - Change of NTP without GPS sync. error in one minute

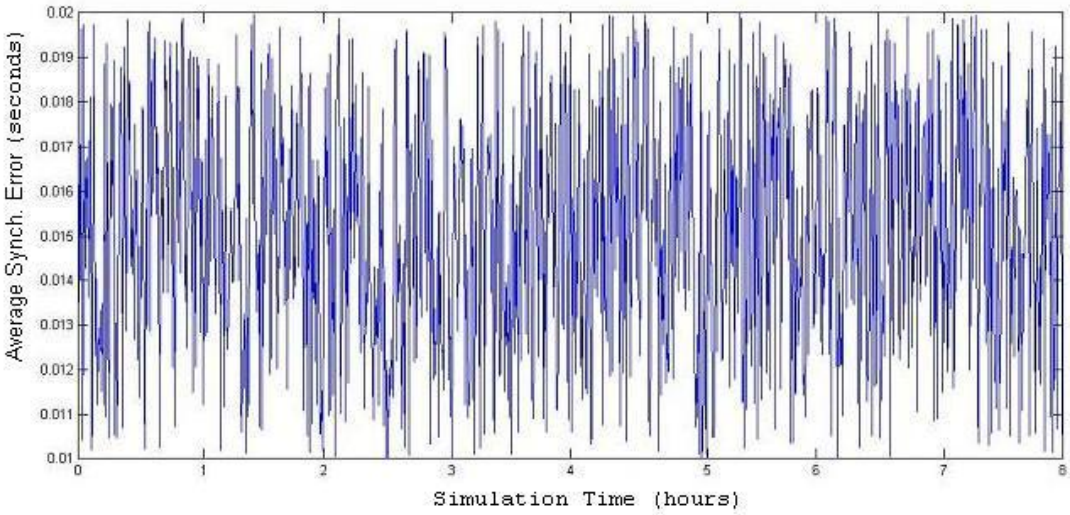


Figure 19 - Change of NTO without GPS sync. error in eight hours

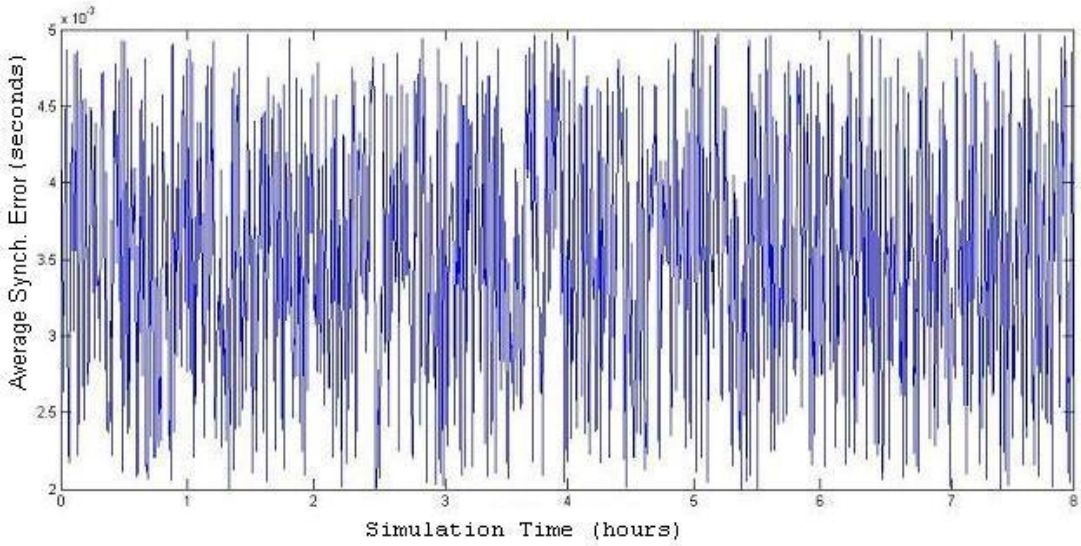


Figure 20 - Change of NTP with GPS sync. error in eight hours

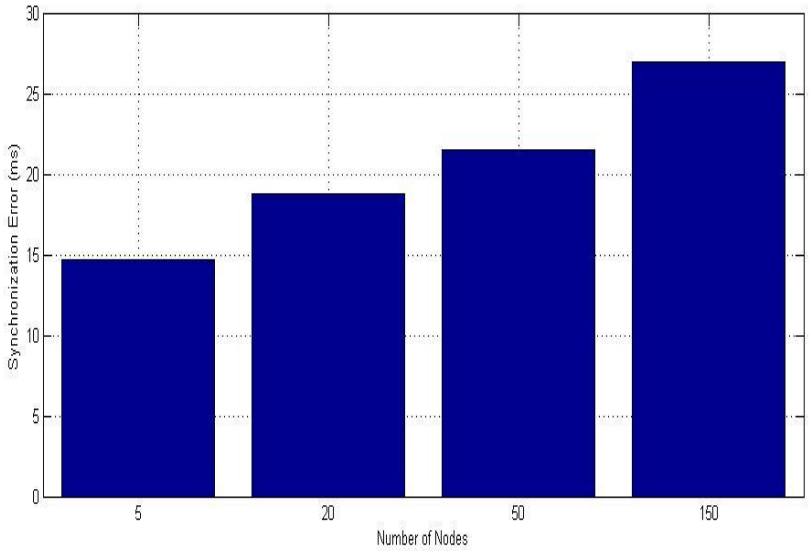


Figure 21 - Max. synch. error for different numbers of nodes with GPS receivers.

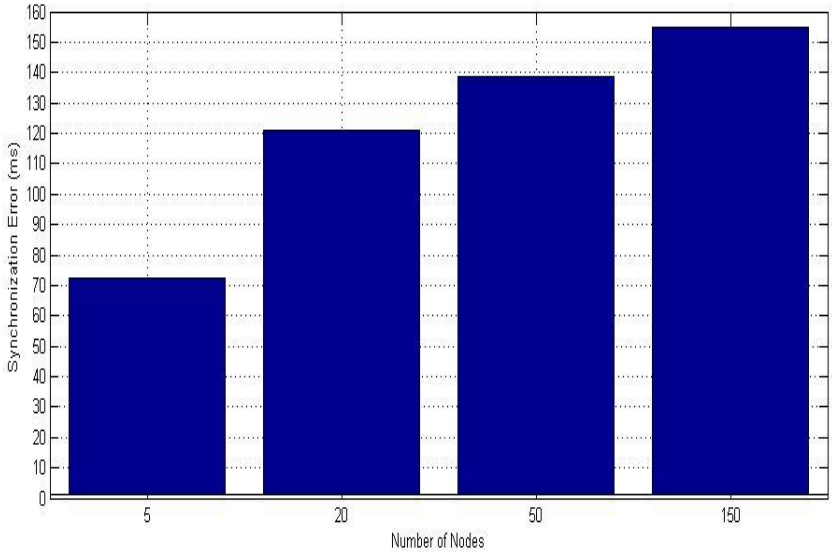


Figure 22 - Max. synch. error for different numbers of NTP nodes without GPS receivers.

As it can be seen from the figures, implementation of GPS clock discipline method reduces the time synchronization error drastically. For a 150-node network, maximum synchronization error is reduced to 26.975 ms from 154.76 ms, and an average synchronization error of 15.327 ms, which is in the acceptable range for power quality measurement standards, was obtained. Figure 23 shows the change of average synchronization error with increasing number of nodes. Simulation was run for 30 times for each number of nodes to obtain results with 95% reliability.

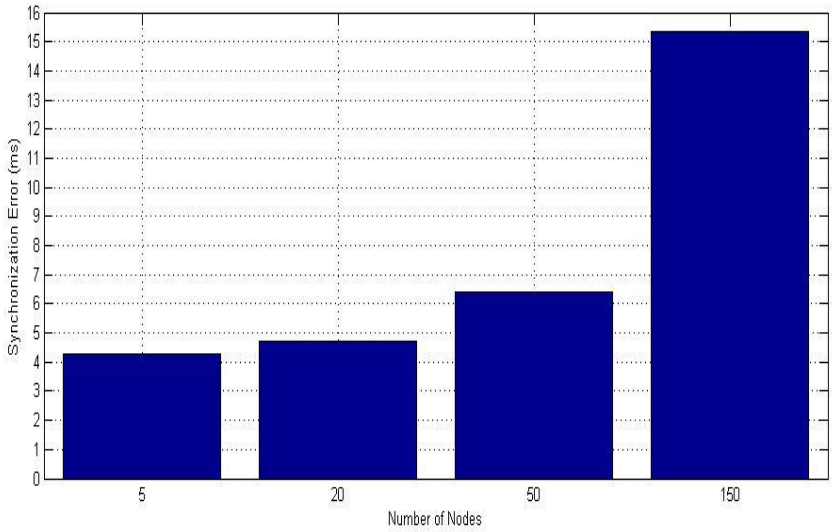


Figure 23 - Avg. synch. error for different numbers of NTP nodes with GPS receivers.

Figures 24, 45 and 26 show the change of synchronization error with number of NTP stratum-1 servers, end-to-end delay and poll rate. As it can be seen from Figure 24, increasing the poll rate from 60 seconds to 1 hour affects synchronization accuracy. Degradation in synchronization accuracy is 15 ms.

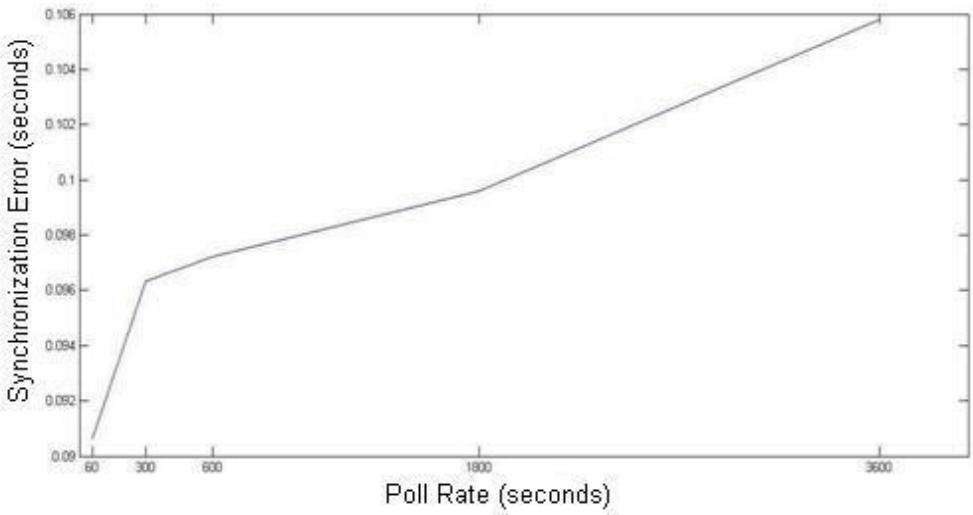


Figure 24 - Change of sync. eror with increasing poll rate

To determine whether the end-to-end delay between NTP clients and servers is an important factor in synchronization, synchronization error was recorded with increasing magnitude of end-to-end delay. With end-to-end delay changing from 10 ms to 150 ms, maximum synchronization error remains below 120 ms, and varies within a range of only 30 ms, as seen as in Figure 25.

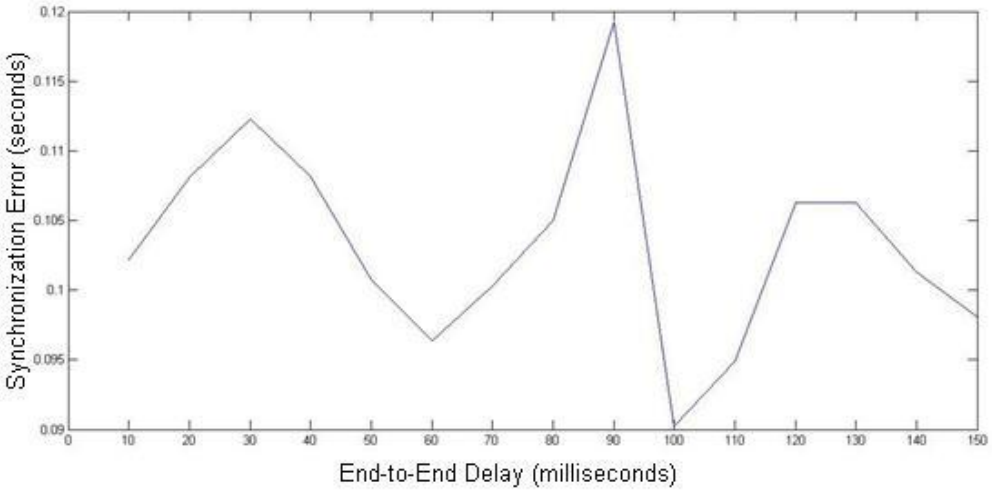


Figure 25 - Change of sync. error with increasing end-to-end delay

Number of servers is another important factor which may affect synchronization accuracy. To observe the effect of the number of servers, a 150-node setup is used. As it can be seen from Figure 26, introduction of new servers to the network improves the synchronization performance. Adding new servers make it possible for the existing servers to share burden of incoming requests and synchronization packets.

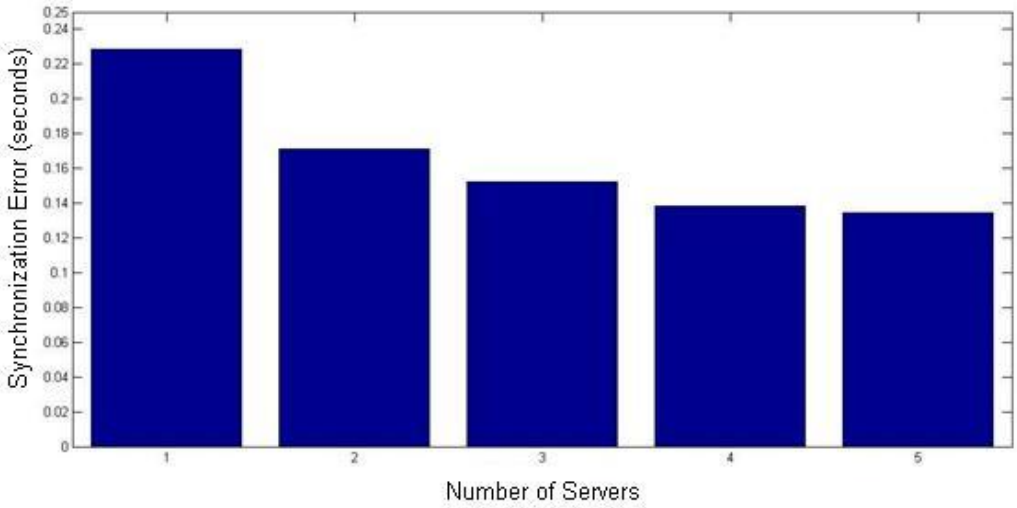


Figure 26 - Change of sync. error with increasing number of NTP servers

6.6 Comparing the Simulation Results of NTP and the Proposed Method

As indicated in the preceding chapter, synchronization error should be less than 20 ms to fulfill the requirements of a power quality measurement network. When the simulation results of NTP and the proposed method are investigated, it can be seen that organizing large networks as groups of smaller networks can be a good strategy for hierarchical methods. By doing this, the depth of the network and distance of a node to the synchronization root can be reduced.

The method proposed in this thesis can be used for time synchronization of large-scale wired networks, if a big portion of nodes are equipped with GPS receivers. GPS receivers are used as global reference broadcast providers and help to reduce synchronization error. On the other hand, even if all nodes except one node are equipped with GPS receivers, synchronization accuracy doesn't reach the level acceptable by measurement standards. Synchronization error of 23 ms with all nodes equipped with a GPS

receiver may not be satisfactory when the synchronization accuracy is compared to NTP. NTP with GPS receivers results in a synchronization accuracy which can be acceptable for measurement purposes. Average synchronization error of 15.327 ms for a 150-node network is better than the proposed method provides for a network with same size and same number of GPS receivers. Besides, NTP PPS drivers developed by Mills' team may perform better than the simulated mechanism, since they contain algorithms to reduce the non-determinism caused by PPS hardware interface and driver processing time. Therefore, NTP with GPS clock disciplining can be accepted as a convenient time synchronization method for large-scale measurement networks.

CHAPTER 7

ACHIEVEMENTS AND SHORTCOMINGS OF THE THESIS WORK

In this thesis, time synchronization problem and time synchronization of networked systems are discussed from the measurement networks aspect and TNPQP is given as a specific implementation case. Measurement networks can differ from other networked systems by their need for data aggregation. For the collected data to be aggregated into a meaningful result, data collected from separate sources should share a common time references.

As a measurement network, TNPQP requires a certain level of time synchronization accuracy to satisfy power quality measurement standards. The synchronization accuracy level required by TNPQP may be regarded as high compared to other applications some of which require only correct time ordering. This high accuracy time synchronization need causes traditional synchronization methods to be insufficient. Furthermore, computer hardware, network topology, distance of power quality monitors to each other, power consumption requirements of the monitors prevent researchers from implementing an existing method.

In the beginning of this thesis, some general concepts about time synchronization and the timing requirements of modern computer systems are explained. Then, the time synchronization requirements, network topology and hardware of TNPQP are briefly mentioned to

illustrate the platform where the synchronization methods discussed in the following chapters can be utilized.

A more detailed discussion about computer clocks and time synchronization phenomena takes place in the second chapter to describe the synchronization problem and the need for time synchronization in computer networks. Besides, design criteria for network time synchronization methods are discussed and reasons for non-determinism in a network time synchronization scheme are explained.

In the third chapter of this thesis, an overview of current time synchronization methods is presented. In the first part of this overview, WSN time synchronization methods are taken into focus, since they are relatively newer methods when compared to the traditional wired synchronization methods and they are claimed to provide accurate synchronization when they are used for wireless networks. RBS and TPSN are experimentally proven by their authors to give better synchronization results than the other methods. On the other hand, implementing reference broadcast synchronization requires a common medium, e.g. wireless communication channel, to broadcast reference packets. This requirement cannot be fulfilled in traditional wired networks and specifically TNPQP network where measurement monitors are placed to diverse geographical locations. Hierarchical structure found in TPSN can be implemented in TNPQP network without any change in the method. However, synchronization accuracy should be improved by some additional schemes to satisfy the demands of TNPQP. This situation about two high-accuracy WSN synchronization methods prevents them to be implemented alone in a measurement network. At this point, the idea of implementing the synchronization idea of two algorithms together may be an acceptable solution. By doing this, it may be possible to

obtain a hierarchical method which does not require a common medium of communication and still stays improvable by adding a reference signal like PPS of GPS. From this idea, a synchronization method is proposed in this thesis and simulation results of this method are also presented.

NTP is one of the oldest protocols used in Internet and it is the network synchronization method with the widest use. NTP is also easy to use in a networked system, since there are publicly available servers, reference implementations and drivers. NTP provides sufficient level time synchronization to most of the computers in Internet. On the other hand, requirements of TNPQP force us to look for an improvement on the performance of NTP. Using GPS PPS signals on the clock discipline stage of NTP may result in time synchronization accuracy values within the range required by power quality measurement standards. The architecture of NTP is investigated in the fourth chapter.

In the fifth chapter, first, the time synchronization accuracy requirements of TNPQP are reviewed. Then, the possible improvements on WSN synchronization methods and NTP are discussed. As indicated before in this chapter, performances of WSN methods and NTP can be improved by using global reference broadcasts like GPS PPS signals. Therefore, two methods which utilize PPS signals are proposed in the fifth chapter to synchronize TNPQP monitors.

Simulation results for proposed methods are presented in the sixth chapter. As it can be seen from the figures, NTP with GPS PPS improvement results in better performance than the proposed method in any case. The reason for that is the clustering and selection algorithms implemented in NTP. NTP clients receive multiple synchronization packets from a number of sources. Clients aggregate

the received packet into a more stable result. The lack of a mechanism like this in the proposed method causes more jitter and larger values of maximum synchronization error. To improve the performance of the proposed method selection algorithms and a clock discipline algorithm may be implemented. Besides, the current form of this proposed method can be utilized as a low-cost and easily implementable synchronization alternative for networked embedded systems where a complete NTP client cannot be installed.

The work presented in this thesis has shortcoming in some aspects. First, algorithms proposed in the preceding chapters may be implemented in a real hardware benchmark to be evaluated in realistic network traffic. NTP has a number of sub-algorithms and requires different drivers when it is used in PPS and non-PPS scenarios. To obtain more realistic results, NTP with PPS support can be implemented for a small group of computers. A further study would be to investigate the behavior the synchronization methods under non-zero packet loss probability. This would be meaningful especially for congested Internet situations. The proposed method can also be implemented for the same hardware and the results can be evaluated. Another addition can be made by discussing and evaluating by simulation some LAN time synchronization methods, e.g. IEEE1588. High accuracy LAN time synchronization methods can be extended to synchronize WAN-scale measurement networks.

REFERENCES

[1] Dave L. Mills, “NTP Overview”, http://www.eecis.udel.edu/~mills, Last access date: 15.11.2009.

[2] NPQ Project Website, http://www.guckalitesi.gen.tr, Last access date: 07.12.2009.

[3] IEC 61000-4-30, Testing and Measurement Techniques – Power Quality Measurement Methods.

[4] Adlink DAQ2200 Series DAQ Boards User Guide, http://www.adlinktech.com/PD/web/PD_Manual.php?PDNo=13&Kind=M, Last access date: 07.12.2009.

[5] K. W. Monington and J. Levine, "Time Synchronization Using the Internet", Frequency Control Symposium, Proceedings of the 1997 IEEE International, 28-30 May 1997, Page(s):395 - 403.

[6] M. L. Sichitiu and C. Veerarittiphan, "Simple, Accurate Time Synchronization for Wireless Sensor Networks", Wireless Communications and Networking, Volume 2, 16-20 March 2003, Page(s):1266-1273.

[7] D. Doley, J. Halpern, R. H. Strong, “On The Possibility and Impossibility of Achieving Clock Synchronization", Proceedings of the 16th Annual ACM Symposium on Theory of Computing, 1984, Pages: 504-511.

[8] F. Sivrikaya and B. Yener, "Time Synchronization in Sensor Networks: A Survey", IEEE Network, Volume 18, Issue 4, July-August 2004, Pages:45-50.

[9] S. Palchaudhuri, A.K. Saha, D. B. Johnsin, "Adaptive Clock Synchronization in Sensor Networks", Third International Symposium on Information Processing in Sensor Networks, 2004, Page(s):340-348.

[10] Z. Tian, X. Luo, G. B. Giannakis, "Cross-Layer Sensor Network Synchronization", Conference Record of the Thirty-Eighth

Asilomar Conference on Signals, Systems and Computers, Vol. 1, 2004, Page(s):1276-1280.

[11] J. Elson, D. Estrin, "Time Synchronization for Wireless Sensor Networks", Proceedings 15th International Parallel and Distributed Processing Symposium, 23-27 April 2001 Pages: 1965-1970.

[12] S. Ganeriwal, R. Kumar, M. B. Srivastava, "Timing-sync Protocol for Sensor Networks", SenSys 2003, November 5-7, 2003, Los Angeles, California, USA.

[13] M. Maróti, B. Kusy, G. Simon and A. Lédeczi, "The Flooding Time Synchronization Protocol", Proceedings of the 2nd International Conference on Embedded Networked Sensor Systems, 2004, Pages: 39-49.

[14] M. L. Sichitiu and C. Veerarittiphan, "Simple, Accurate Time Synchronization for Wireless Sensor Networks", Wireless Communications and Networking, 2003, Volume 2, 16-20 March 2003, Page(s):1266-1273.

[15] Jana Van Greunen and Jan Rabaey, "Lightweight Time Synchronization for Sensor Networks", WSNA 2003, September 19, 2003, San Diego, California, USA.

[16] Liming He and Geng-Sheng Kuo, "A Novel Time Synchronization Scheme in Wireless Sensor Networks", Vehicular Technology Conference, 2006, VTC 2006-Spring, IEEE 63rd, Volume 2, 2006 Page(s): 568-572.

[17] Kun Sun, Peng Ning and Cliff Wang, "Secure and Resilient Clock Synchronization in Wireless Sensor Networks", IEEE Journal on Selected Areas in Communications, Volume 24, Issue 2, Feb. 2006 Page(s):395-408.

[18] W. Su and I. F. Akyildiz, "Time-Diffusion Synchronization Protocol for Wireless Sensor Networks", IEEE/ACM Transactions on Networking, Vol. 13,No.2, April2005.

[19] H. Lee, W. Yu and Y. Kwon, "Efficient RBS in Sensor Networks", Proceedings of the Third International Conference on Information Technology: New Generations (ITNG'06), 10-12 April 2006, Page(s): 279-284.

[20] P. Verissimo, L. Rodrigues and A. Casimiro, "CesiumSpray: A Precise and Accurate Global Time Service for Large-scale Systems", Real-Time Systems, 1997, 12, Pages: 243-294.

[21] Dave L. Mills, "NTPv4 Implementation Guide", http://www.ee cis.udel.edu/~mills, Last Access Date: 07.12.2009.

[22] K. A. Marzullo, “Maintaining the Time in a Distributed System: An Example of a Loosely-Coupled Distributed Service”, Ph.D. dissertation, Stanford University, Department of Electrical Engineering, February 1984.

[23] Dave L. Mills, “NTPv4 Survey”, http://www.eecis.udel.edu/~mills, Last access date: 07.12.2009.

[24] Wikipedia, http://en.wikipedia.org, Last access date: 07.12.2009

[25] NTP PPS Parallel Port Driver, http://www.eecis.udel.edu/~mills/ntp/html/drivers/driver22.html, Last access date: 07.12.2009.

[26] D. Sanghi, A. K. Agrawala, Ó. Gudmundsson and B. N. Jain; “Experimental Assessment of End-to-end Behavior on Internet”, Proceedings of the IEEE INFOCOM'93, 1993.

[27] S. Bilgen, “Reliability of Measurement Results”, Unpublished Technical Report.

[28] National Institute of Standards and Technology, IEEE-1588 website, http://ieee1588.nist.gov/, Last access date: 01.02.2010

[29] Enduron Techonogies Time Synchronization Solutions, http://www.endruntechnologies.com, Last access date: 01.02.2010

[30] C. Lenzen, P. Sommer and R. Wattenhofer, “Optimal Clock Synchronization in Networks”, SenSys’09, November 4-6, 2009, Berkeley, CA, USA

[31] Q. Ye and L. Cheng, “DTP: Double-pairwise Time Protocol for Disruption Tolerant Networks”, The 28th International Conference on Distributed Computing Systems, 2008, Beijing. China

[32] L. Wang, J. Fernandez, J. Burgett, R. Conners and Y. Liu, “An Evaluation of Network Time Protocol for Clock Synchronization in Wide Area Measurements”, Power and Energy Society General Meeting, Conversion and Delivery of Electrical Energy in the 21st Century, 2008, IEEE, Pages(s): 1-5