Consensus in Large Scale Wireless Sensor Actuator Networks: Time Synchronization as a case study.

Rutika Vivek Bhapkar

Problem Report submitted to the

College of Engineering and Mineral Resources

at West Virginia University

in partial fulfillment of the requirements for the degree of

Master of Science in Computer Science

Approved by

Dr. Vinod Kulathumani, Ph.D., Chair

Dr. James D Mooney, Ph.D.

Dr. Elaine Eschen, Ph.D.

Lane Department of Computer Science and Electrical Engineering
Morgantown, West Virginia 2013

Keywords: Time Synchronization, consensus, sensor, actuator, wireless networks

Abstract

Consensus in Large Scale Wireless Sensor Actuator Networks: Time Synchronization as a case study

Time synchronization has achieved great importance in multihop ad-hoc wireless sensor networks. The advancement in low-cost, low power design has given way for the development and use of the sensor networks in variety of applications. Given that different nodes can start at different times and their on-board clocks have different clock drifts, time synchronization techniques are essential for long-term sensor network deployments. The synchronization protocol namely Uniform Convergence gives a clear understanding of the time synchronization principle .This protocol has been demonstrated using motes and the use of a leader value instead of a leader mote. Each mote works independently of each other. Each mote has its own timer. Each mote broadcasts a periodic time message .This message contains local time and offset. This message is then received by the other motes in the network. After a period of time, all the motes start blinking and beeping together synchronously. Thus, they synchronize themselves accordingly over a period of time.

The problem with the sensor network is their dynamics. Due to this problem, a new idea of consensus came into picture. A consensus protocol provides means through which all agents agree on some particular variable of interest. This variable has different interpretations depending upon the application. This report describes the implementation of the time synchronization using the uniform convergence protocol and ways to achieve consensus in wireless sensor networks.

Acknowledgements

I take this opportunity to express my profound gratitude and deep regards to my Mentor Dr Vinod Kulathumani for his constant guidance, monitoring and encouragement throughout the course of this problem report. The guidance, support and the timely help given by him has helped me to complete this task through various stages.

I would also like to extend my gratitude to Dr James Mooney and Dr Elaine Eschen for serving on the committee and supporting the project with their encouragement and cooperation throughout the period of my assignment.

I would like to thank my family for their indispensable support, faith and advice when I needed them the most. I would like to also thank my friends Srinath Chowdary, Akshaya Rane, Siddhita Aparaj Saurabh Chaudhari, Kiran Chaudhari and Pratik Pednekar for being there during the phases of my life.

Table of Contents

List of Abbreviations

WSN	Wireless Sensor Network
FTSP	Flooding Time Synchronization Protocol
TPSN	Timing Sync Protocol for Sensor Networks
RBS	Reference Broadcast Synchronization
PCO	Pulse Coupled Oscillators
MAC	Medium Access Control
UAV	Unmanned Aerial Vehicles
RTS	Request To Send
CTS	Clear To Send
NTP	Network Time Protocol
UTC	Universal Coordinated Time
TDP	Time Diffusion Synchronization Protocol
GCS	Global Clock Synchronization
RFA	Reachback Firefly algorithm
ATS	Average Time synchronization
CCS	Consensus Clock synchronization

List of figures

CHAPTER 1: INTRODUCTION

1.1 Background

Wireless networks when combined with sensing and actuation are used in disaster relief operations, intelligent buildings, pervasive computing etc. Thus, WSAN provides fine grained monitoring, helps to interact with many embedded computers and doesn't need any infrastructure setup as it's a wireless medium. The deployment is quite practical because we can deploy nodes in an area and monitor the area. An emerging technology with a low-cost, low power design and small, wireless sensors and actuators led to the development of the WSAN. Thus, smart environments were created. The enabling technology included powerful sensors that reduced the power consumption. Microsensors that could detect acceleration, motion ,vibration ,magnetic effect ,heat, pressure, light, sound , chemical and biological event .In addition to these sensors, there are actuators like mirrors, motors and micro-robots. The communication provided by these devices are short range and the bit rate is low due to the limited memory and size constraints.

Sensing only systems are known as wireless sensor networks (WSN).Wireless sensor devices are deployed to monitor and interact with the environment and disseminate information about specific phenomena. A wireless sensor device is a battery operated device that can sense physical quantities like temperature, light, sound, movement of an object etc. These devices are also capable of data storage, computation and communication. There has to be a coordination and networking of these devices. These sensor devices work together to achieve a common goal. The data is then collected from all the nodes (sensors) and analyzed. Used for mainly monitoring applications. These are usually composed of large scale networks of embedded systems.

Sensing applications include infrastructure monitoring like placing sensors on a bridge and monitoring the wear and tear of the bridge over a period of time. Camera sensor networks are used on airports and highways to detect any suspicious activity or a speed violation respectively. Sensors are also used in Perimeter security to detect any intrusion attack. Environment monitoring is another area where these sensors are used to detect events like forest fire or variations in climatic conditions.

There is another set of systems that combine the sensing and actuation properties. Robotics, self-configuring structures, robotic surgery, UAV, Autonomous driving are some examples. There is a common goal that is used in all these applications. That goal is co-ordination between the devices. The sensors and the actuators have to co-ordinate with each other to accomplish any task. The following are the examples where co-ordination is used to perform the desired task.

- Leader Election
- Achieving Consensus
- Synchronization
- Desynchronization

A leader is necessary to maintain co-ordination among all the nodes in a multi hop network. The leader looks after the entire network .For nodes to execute any distributed task, one node has to become a coordinator (leader).All the nodes have to follow the leader. Thus, to elect a leader in a network, co-ordination is needed among all the nodes to select a unique leader from among the nodes. Thus, co-ordination is used in a leader election network.

Calculating a group average in an network needs the co-ordination among all the nodes as each node contributes its initial node value. Thus, combining all the values from all the nodes gives

an estimated average of the network. This gives way to the concept of achieving consensus using co-ordination.

Time synchronization is the process of synchronizing the local clocks of multiple nodes. Collaboration among nodes is often required for the data reduction that is critical to the energy-efficiency of a sensor network.

Desynchronization is a biologically inspired concept. Here; the nodes try to perform their tasks as far as possible from each other to avoid collisions. The calling behavior of Japanese Tree frogs is an example of the Desynchronization technique.

1.2 Problem Statement

Traditional clock synchronization protocols for wired networks cannot be used because wireless sensor network protocols require the ability to adapt dynamically, the ability to handle sensor mobility and scalability. To demonstrate time synchronization using a synchronization protocol namely Uniform Convergence. Uniform Convergence makes use of a leader value and not a leader mote. The offset is calculated by each node from the sent message and each node then adjusts its time accordingly so that all the nodes (Telosb) blink together in sync and the Micaz nodes beep together in sync. However, there are many requirements for achieving time synchronization like precision, accuracy, scope, immediacy etc. which pose problems in some situations where the requirements are application specific.

Time synchronization can be centralized wherein a central (root) node provides the reference time which is assumed to be correct. The other nodes synchronize their times with this root node. However, the drawback of this synchronization is the single point of failure. If the root fails, the entire system crashes. The other type of synchronization is decentralized, where a fully distributed, communication approach is used where all the nodes run the same algorithm. There is

no special node as master (root). However, this too has its drawback. In distributed systems, there is no global clock. These clocks drift seconds per day thus accumulating major errors over a period of time. Thus, this drift gets accumulated over a period of time and thus synchronization is lost.

Thus, the concept of consensus was used. Consensus is the coordination of groups of autonomous agents. It can work in a distributed way and thus doesn't need a root or tree based structure. Hence, more accuracy is obtained for the whole network. The skew differences are compensated among the various nodes. Average consensus algorithms are the most common gossiping algorithms that allow a set of nodes to compute the average of some initial variables in a decentralized fashion. They have multiple applications in sensor networking and have received considerable attention recently from a number of engineering communities. The consensus algorithms are the iterative algorithms where the nodes in the network exchange data among each other to achieve agreement. For consensus to take place, the network need not be fully connected. Consensus is achieved locally and eventually a global agreement is reached.

CHAPTER 2: NEED FOR TIME SYNCHRONIZATION

To collaborate for a common application, wireless sensor nodes have to be precisely coordinated for multi-hop communication. Most of the co-ordination depends on the time synchronization among the nodes .Coordinated sleep and wake up schedules have to be time synchronized as the loss of data would be a costly affair in terms of power. Monitoring of physical events also requires tracking sensor values with respect to the timestamps of a global clock.

In centralized systems, there is no need for synchronized time because there is no time ambiguity. A process gets the time by simply issuing a system call to the kernel (root node). When another process then tries to get the time, it will get either an equal or a higher time value. Thus, there is a clear ordering of events and the times at which these events occur.

In distributed systems, there is no global clock or common memory. Each processor has its own internal clock and its own notion of time. These clocks can easily drift seconds per day and accumulate errors over time. Different clocks tick at different rates and this might create a problem to applications that depend on a synchronized notion of time.

In order for sensor nodes to be able to synchronize, they have to possess for a period of time a communication channel where the message delays between nodes can be reliably estimated. However, the enemy of accurate network time synchronization is the non-determinism of the delay estimation process. The source of error in synchronization consists of four parts: send time, access time, propagation time and receive time.

Send Time:

- Time required for constructing the message by the sender.
- It includes context switches and system call overhead incurred by the application.
- It takes into account the time needed to transfer the message from the host to its network interface.

Access Time:

- It's the delay incurred while waiting for access to the transmission channel.
- Contention based MAC must wait for the channel to be clear before transmitting and retransmit in case of a collision.
- Wireless RTS/CTS require an exchange of control packets before the start of any transmission.

Propagation Time:

- This is the time for the message to travel from the sender to the destination node through the channel since it left the sender.
- The propagation time is a major cause of the delay in wide-area networks.

Receive Time:

This is the time for the network interface on the receiver side to get the message and notify the host of its arrival.

CHAPTER 3: ACHIEVING CONSENSUS

Consider a network (a connected graph) G = (N, E) consisting of a set of nodes N = {1… n} and a set of edges E, where each edge {i,j} ∈ E is an unordered pair of distinct nodes. The set of neighbors of node i is denoted by N_i = {j| {i,j} ∈ E}.

Each node i holds an initial scalar value $x_i(0)$ ∈ R, and x (0) = (x_1 (0),…,x_n(0)) denotes the vector of the initial values on the network. We have to calculate (1/n)$\sum_{i=1}^{n} x_i$ (0),in which the nodes communicate with the neighbors.

In an ad hoc wireless network, link failures are common where the nodes are mobile and have limited battery power and transmission range. There are no master nodes and thus the failures have to be resolved by local nodes which communicate local information to one another. The link failures disrupt the communication between the nodes and this results into collisions.

The protocols namely TPSN, FTSP etc. are tree based and are fragile to link or node failures. A consensus based clock synchronization has the following advantages:

- Can work in a distributed way and thus doesn't need a root or a tree based structure.
- More accuracy is obtained for the whole network.
- The skew differences are compensated among the various nodes.

Consensus is the coordination of groups of autonomous agents.A consensus protocol provides means through which all agents agree on some particular variable of interest.This variable has different interpretations depending upon the application.

The consensus distributed algorithms can be classified as Synchronous algorithms and Asynchronous.

Examples of consensus are sensor fusion, group average in distributed computation, attitude alignment in multiple spacecraft setting. Sensors deployed to measure the temperature T of a given source. Due to the additive Gaussian noise, each sensor will have a different sensor reading. A good filter of the Gaussian noise is the mean filter. Thus, the average of the initial values will give a good estimation of the source temperature.

3.1 Synchronous algorithm

All agents update their states at the same time using the latest state values from their neighbors. The order is fixed. They are based on the notion of a round. Every execution of the protocol consists of a sequence of rounds. Every node will start and finish the round synchronously. They are designed to tolerate crash failures. All the nodes have identical running clocks and all the internode communication takes place at exactly the same time in each round of the algorithm. Thus, a global clock is present. Every iteration is performed at the same time in all the nodes. Each node thus waits for the value from every other node before calculating the average (in an average consensus algorithm).

3.2 Asynchronous algorithm

Setting the order in which the states of agents are updated is not fixed and selection of previous values of states used in the update is also not fixed. All the nodes have an internal clock and there is no guaranteed relationship between the clocks. There is no guarantee on communication timing except that if a message is sent then it will be received barring link failure.

3.3 Working of Consensus in wireless networks

Achieving consensus on a common global parameter through distributed algorithms is a key problem in WSN. In centralized schemes, there is a need to send sensor data to a fusion center that causes congestion around the sink node. However, with distributed schemes, this congestion is avoided and the network is resilient to node failures and attacks.

The consensus algorithms are the iterative algorithms where the nodes in the network exchange data among each other to achieve agreement. The network need not be fully connected indeed consensus is achieved locally and a global agreement is reached.

The standard average consensus algorithm involves, at each time step, every system takes a weighted average of its own value with values received from some of the other nodes.

3.4 Ways of achieving consensus

Distributed averaging consensus can be achieved in two ways.

3.4.1 Method 1: Flooding.

Here; each node maintains a table of the initial node values of all the nodes initialized with its own node value only. At every step, the nodes exchange information from their own tables and the tables of their neighbors. After a number of steps equal to the diameter of the network, each node has the knowledge of all the initial values of all the nodes, so the average can be computed.

3.4.2 Method 2: Distributed Linear Iterations

x_i(t+1) = $W_{ii}x_i$(t) + $\sum_{j\varepsilon N_i} W_{ij}\ x_j$(t), where i = 1,…, n, where t =0, 1, 2,…time index and W_{ij} is the weight on x_j at node i.Setting W_{ij} = 0 for j ≠N_i, this iteration can also be written in another form as

x (t+1) = Wx (t)…… (1)

This equation implies that x (t) = W^tx (0) for all t.

Weight matrix W is chosen in such a way that for any initial value x (0), x (t) converges to the average $\bar{x}$ = (1^Tx (0)/n) 1

Here, 1 denotes the vector with all coefficients one.

The matrix equation is:

$\lim_{t\to\infty} W^t$ = (1/n) 11^T…….. (2)

Equations 1 and 2 are called the convergence conditions that are required to converge to the average for any initial vector x (0) ∈ R^n .

3.5 Ways to choose W that guarantee convergence

- Constant Edge weights :Set all the edge weights (for neighboring nodes)equal to a constant α;the self-weights on the nodes are then chosen to satisfy the condition

  W1= 1

- Local Degree weights: Each node selects its weights on the basis of its own degree and the degree of its neighbors.

  W_{ij} = 1/max $\{d_i, d_j\}$ {i, j}∈ E

- Max Degree: Uses global knowledge. Each node is assumed to know the maximum node degree in the network(d_{max}= $max_i\{d_i\}$ and the same weight is associated to each link as follows:

$$w_{ij}=\begin{cases}\frac{1}{d_{max}} & if(i,j)\in E\ and\ i\neq j\\ 0 & if\ (i,j)\notin E\ and\ i\neq j\end{cases}$$

CHAPTER 4: TIME SYNCHRONIZATION

Time synchronization is a process of adjusting the local clocks of multiple nodes. Its a must for basic communication and helps to locate sensors, sources and detect movement. Time synchronization has been observed in the firefly flashing in unison creating bursts of light and the synchronous clapping in a concert hall.

The Human synchrony is an example of time synchronization. The normal heart beat is initiated by an electric impulse.The pacemaker cells of the heart are capable of generating electricity. The pacemaker cells of the heart ,the sino atrial node(SA) ,is a group of cells that together ensures that the heart beats are in tempo.All the cells of the heart have to release their electric charge at the same time.This has to be steadily repeated over a period of time.If not for this synchrony,all the living creatures would have got out of life!

Synchronization can be used by power saving schemes to increase the lifetime of a network. To enable power saving modes, the nodes should have a coordinated sleep and wake up schedules such that the radio shouldn't be turned off when some data is being transmitted. This requires a precise timing between the sensor nodes.

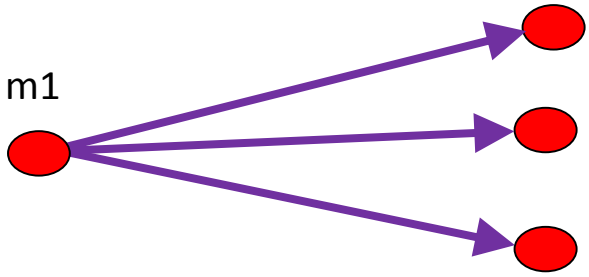
Precise time synchronization can be essential in a WSN to facilitate group operations, such as sensor localization, data aggregation, distributed sampling, source localization etc. Synchronized time stamps can be critical for proper correlation of sensor information from the various sources. In addition, synchronized clocks are essential for shared channel communication protocols, such as Time Division Multiplexing.

Time synchronization can be observed in vehicle tracking scenario. Here, the sensor nodes report the location and the time they sense the vehicle to a sink node. The sink node then combines this information to calculate location and velocity of the vehicle. Thus, precise time synchronization is needed to determine the objects trajectory. Coordinated sleep and wake up schedules of nodes have to be synchronized as the loss of data would be a costly affair.

CHAPTER 5: SYNCHRONIZATION PROTOCOLS

5.1 Reference Broadcast Synchronization

Based on receiver – receiver synchronization. A third party will broadcast a beacon to all the receivers; this beacon does not contain any timing information. The receivers have to compare their clocks to one another to calculate their relative offsets. A broadcast message is used to synchronize a set of receivers with one another, in contrast to the traditional protocols that synchronize the sender of the message with its receiver. Thus, the send time and the access time factors are removed.



m_1 is received simultaneously by multiple receivers: each records a timestamp value contained in m_1.

Fig. 1 RBS procedure

RBS can also be used in multi-hop networks using the concept of zones.

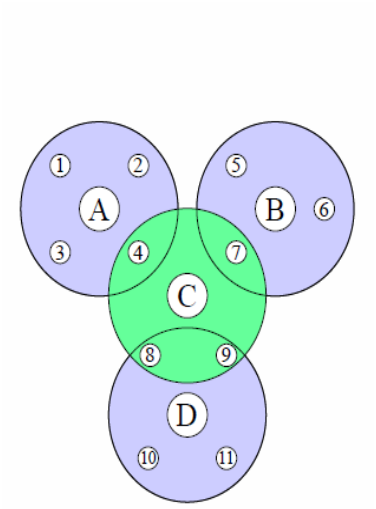


Fig.2a Physical topology

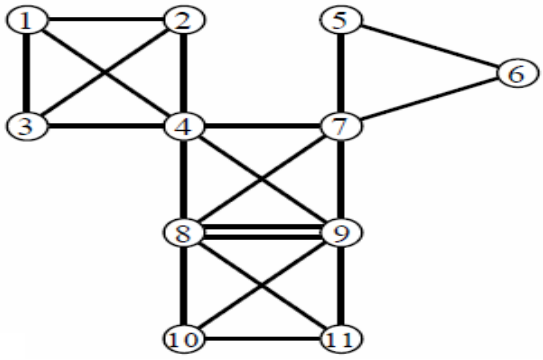


Fig.2b Logical topology

The numbers 1 to 10 are the receivers placed in networks(hops) named A,B,C and D.We can compare the time of E1(R1) with E1(R10) by transforming E1(R1)->E1(R4)->E1(R8)->E1(R10).The conversions can be made automatically by using the shortest path algorithm.

5.1.1 Advantages

- The uncertainty dependent on the sender is removed.
- Uses a sequence of reference messages from the same transmitting station rather than a single message.
- Supports multi hop wireless networks by those wireless stations that belong to multiple neighborhoods.

5.1.1.2 Disadvantages

- Not suitable for TDMA-based MAC protocols since network wide synchronization is not provided.
- Message exchanges with all the neighbors increases the energy consumption and leads to frequent collisions.

5.2 Network Time Protocol

The NTP ensures that the frequency of each host's oscillator is disciplined. Synchronization among hosts is achieved by a hierarchical structure of time servers .In this hierarchical structure, the root is synchronized with UTC and at each level the time servers synchronize the clocks of their subnetwork peers.

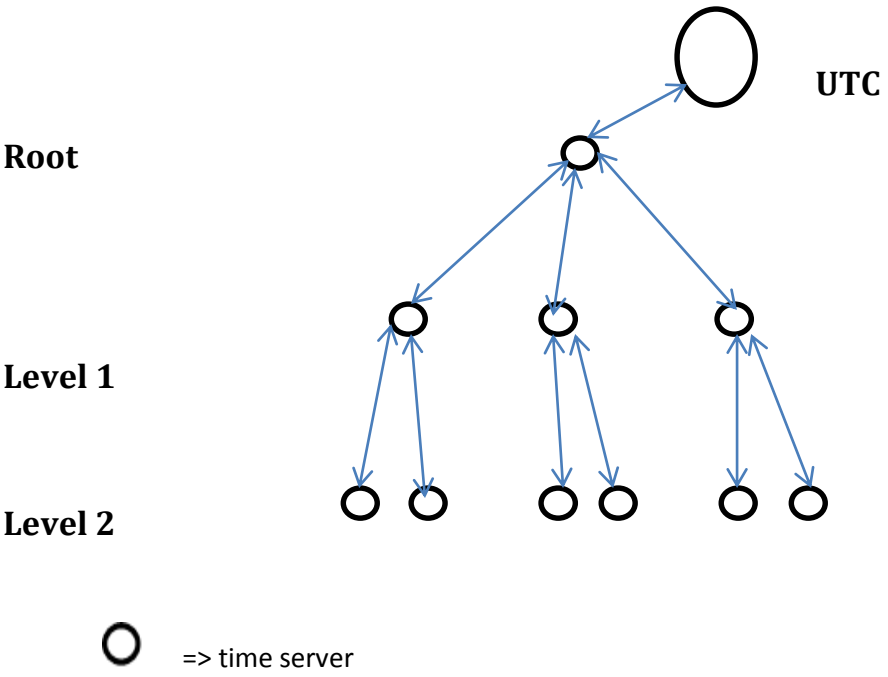


Fig.3 Network Time Protocol Structure

NTP relies on a two way handshake between nodes to estimate the delay between these nodes and calculate the relative offset accordingly.

5.2.1 Advantages

- NTP can unicast, multicast or broadcast protocol messages.

5.2.2 Disadvantages

- It's computationally intensive and not feasible for implementation on the energy constrained sensor nodes.
- It is not suitable for use in sensor networks due to its complexity and energy issues, cost and size factors.

5.3 Timing Sync Protocol for Sensor Networks (TPSN)

TPSN relies on the two-way message exchange scheme shown in Figure 2 to acquire the synchronization between two nodes A and B.

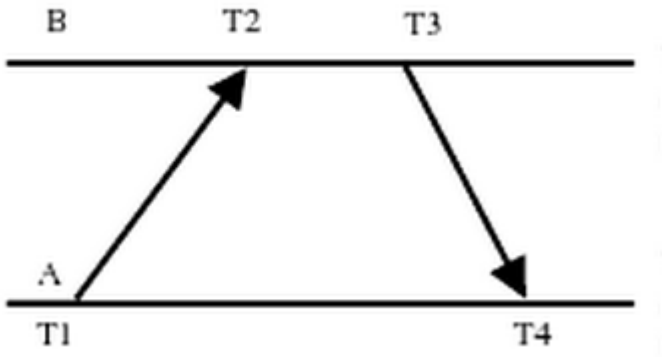


Fig. 4 TPSN message exchanges

It's a Sender-Receiver synchronization that uses a tree structure to organize the network topology. There are two phases involved namely the Level Discovery phase and the Synchronization phase. The first phase creates the hierarchical topology in the network. Each node is allotted a level. There is only one node called the root with a level 0.In the second phase, a node of level i synchronizes to a node of level i-1.At the end of the synchronization phase ,all the nodes are synchronized to the root node. Thus, the entire network is synchronized.T2, T3 are measured in node B clock.T1 and T4 are measured in node A.

Node A initiates the synchronization by sending a synchronization packet at T1 (according to its local clock). This packet includes A's level number, and the value T1.

B receives this packet (according to its local clock) at T2 = T1+¢+ *d*, where ¢ is the relative clock drift between the nodes, and *d* is the propagation delay of the pulse. B responds at time T3 with an acknowledgement packet, which includes the level number of B and the values T1, *T*2, and T3. Then, node A can calculate the clock drift and propagation delay as below, and synchronize itself to B.

$$¢ = \frac{(T2-T1)-(T4-T3)}{2}$$

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

5.3.1 Advantages

- Better performance than RBS and is flexible

5.3.2 Disadvantages

- Multi hop synchronization is not supported.

5.4 Flooding Time Synchronization Protocol (FTSP)

Tailored for applications which have strict precision on resource limited wireless platforms. Achieves a network wide synchronization of the local clocks of the nodes. The sender's time is synchronized with the multiple receivers using a single radio message time stamped at both the sender and the receiver sides. It's sender-receiver synchronization. It has a root structure and all other nodes are synchronized with the root (a single re-elected node that maintains the global time.

The receiving node has sender's transmission time and reception time. The receiver can then estimate the clock offset. Designed for multi hop networks. Root is elected dynamically and is periodically reelected.

5.4.1 Advantages

- Robust.
- Uses flooding of synchronization messages to combat link and node failures.
- Allows dynamically changing topology.

5.4.2 Disadvantages

- Computationally complex.
- Takes time to achieve global time synchronization.

5.5 Uniform Convergence

Instead of a leader node, have all the nodes follow a leader value. Likely leader values are as follows.

- The leader clock could have the smallest value
- The leader clock could have the largest value
- The leader clock could have the mean, median as its value etc.

Local convergence leads to global convergence. Send periodic time sync messages, use easy algorithm to adjust offset.

If recieved _time > local_time+offset then

Offset = i.time – j.local_time;

5.5.1 Advantages

- Automatic fault tolerance
- Simple implementation
- Self-stabilizing from all possible states.
- Periodic time synchronized messages are broadcasted in the network to ensure the leader value is communicated to all the nodes.

5.5.2 Disadvantages

- Duplicate notifications of the same event from a group of nearby sensors can result in significant energy consumption
- The problem of defining what the largest clock value mean when clock reaches maximum value and rolls over.

CHAPTER 6: DESYNCHRONIZATION

Desynchronization is a biologically inspired primitive. Instead of nodes attempting to perform periodic tasks at the same time, nodes perform their tasks as far away as possible from all other nodes. It is useful for scheduling nodes to perform tasks at different time. This property is desirable for resource sharing. Biological systems always change and adapt in response to continuous changes in the environment. Each component of a biological system is governed by local interactions with neighbors, not controlled by a specific leader. Cells acting as oscillators control the animal gaits through desynchronization.Nodes can avoid collisions and message loss by desynchronizing their transmission times. The sampling times can be desynchronized to distribute the energy cost. Male Japanese tree frogs exhibit a self-organized behavior for the desynchronization of their calls. This is explained below.

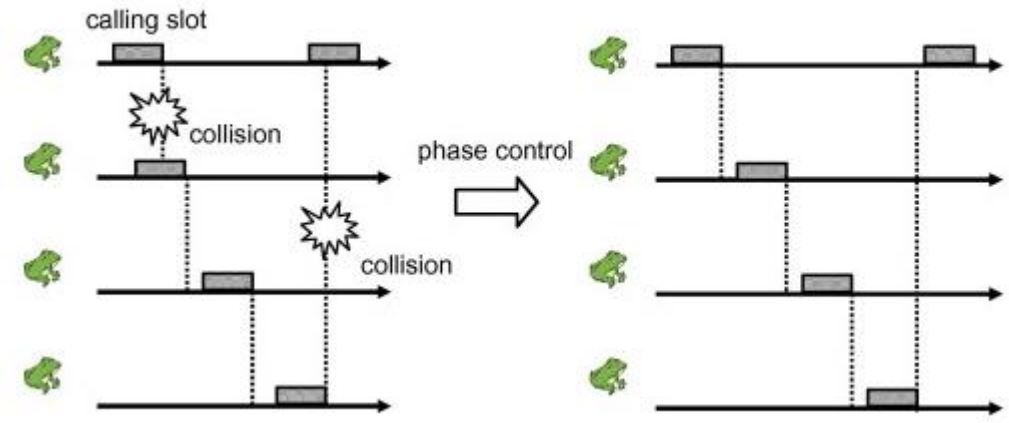


Fig .5 Calling Behavior of Japanese Tree frogs.

The outline of phase control is shown in Fig. 3.The frog calls by making a sound for a certain period of time and then becomes quiet before repeating the call. If two or more male frogs call at random, their calls might overlap. In such a case, the calls interfere with each other and the

female frog (the mating partner) cannot distinguish between the callers. Therefore, each male frog shifts the timing of its calls by listening to the calls of other frogs so as to avoid such overlap. After all frogs establish this interaction pattern, call alternation without interference is achieved within the group.

Applying the primitive in a WSN provides a collision-free self-organizing network without the need for a global clock. Imagine not fireflies flashing in unison, but in a uniformly distributed, round-robin fashion. Each node's clock is independently synchronized with real time, thus, if each node's clock is independently synchronized with real time, all the clocks of the system remain mutually synchronized. Multiple real time clocks (one for each node) are used.

A simple algorithm for achieving desynchronization among a set of n wireless sensor nodes in a single-hop network. We assume that a firing event corresponds to a node broadcasting a wireless firing message that all other nodes can hear. The algorithm works as follows: each node adjusts its phase to be at the midpoint of the two nodes before and after it on the ring. In order to achieve this, a node must pay attention to the timing of the firings before and after its own. If each node can fire closer to the midpoint, then over successive periods this jumping towards the average will bring the system to a state in which all nodes are at the midpoints of their neighbors. This is the desynchronized state.

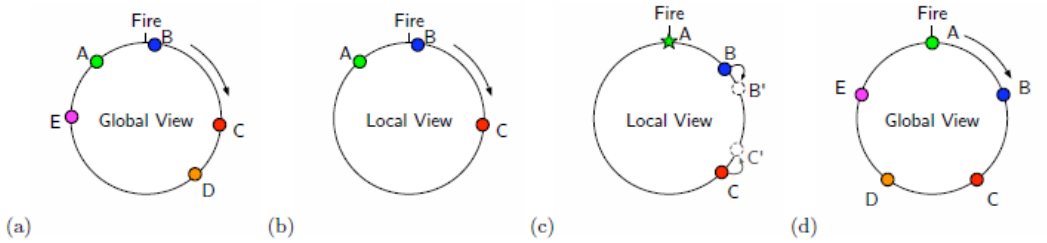


Fig .6 Desynchronization procedure

a) The five nodes are not yet synchronized.

b) B's neighborhood view.

c) A fires, the node that fired immediately before it, node B now knows both of its neighbor's positions. It heard C fire earlier and A just fired.

   Thus, B now knows where it should jump to and calculate its position B'.However, C has jumped to C'.

d) The desynchronized state. The system is stable and every node is at the midpoint.

6.1 Disadvantages

1-round latency before being able to transmit data and (b) a smaller slot size for several rounds until the system re-converges to the desynchronized state. Another limitation is that nodes with equal slots which, although guaranteeing fairness, can also lead to inefficient bandwidth usage. If a node does not have enough data to fully utilize its slot, then the unused bandwidth is wasted.

CHAPTER 7: IMPLEMENTATION DETAILS

7.1Tinyos

Tinyos is an operating system for low power, embedded wireless devices. It's a modern operating system and uses language techniques in an embedded system. Tinyos can be used for communication, storage and timing requirement. It provides a flexible hardware and software boundary.

nesC is the language used which is a dialect of C.A component is the basic unit of nesC.Components connect by interfaces.Components are of two types namely modules and configurations. Modules are components that have variables and a executable code. Configurations are components which wire other components together.

7.2 Demo details

I have implemented the uniform convergence protocol which uses the largest clock value as the leader value. Periodic time synchronized messages are broadcasted in the network to ensure the leader value is communicated to all the nodes. Thus, a sender receiver based technique is used in which a synchronized sender sends the message and the receiver adjusts its offset according to the difference in their times. This algorithm has been tested and successfully implemented and all the nodes converge to a synchronized state.Whenever, new nodes are added in the network or nodes are removed, the network converges to a synchronized state in a very small time.The efficiency of this project is the small convergence time.The convergence time is about one minute.

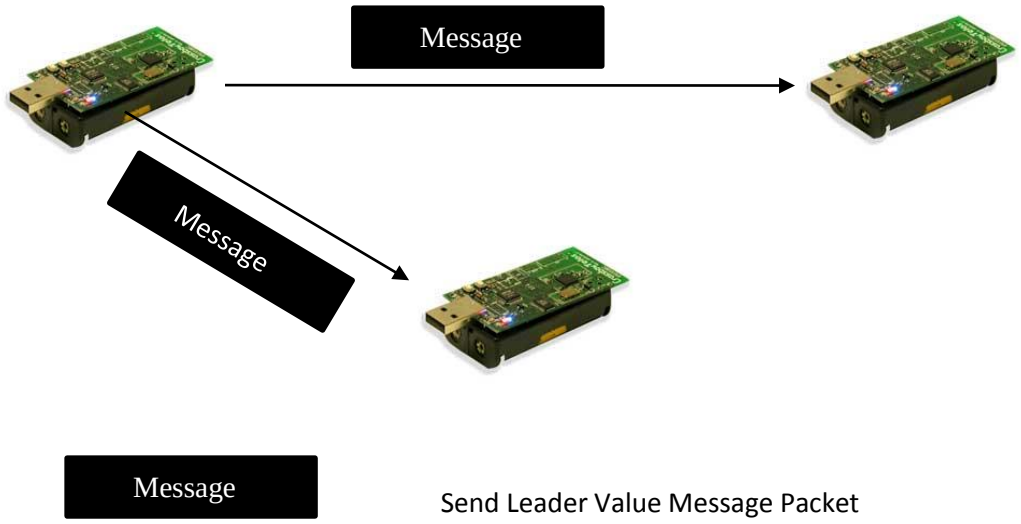


Fig.7 Communication of motes

7.3 Demo Output

- Project implemented on Telosb motes and Micaz motes.
- Telosb Blinks due to the led interface and Micaz beeps due to the Mts300 interface.
- Telosb and Micaz make one pair .On Telosb, the Red led toggles whenever a message is received.
- Periodic blinking takes place for each of the motes .Blue led is used to figure out synchronization when all the nodes are laid out.
- On Micaz node, Yellow led does the same work as the Blue led.
- A sender sends a message (Local_time+offset) to the receiver.When the message is received,the receiver calculates its offset .The new localtime is adjusted in such a way that the receiver also beeps and blinks together periodically with the sender.

- Thus, in a synchronized state, the Blue led (Telosb) and the Yellow led (Micaz) blink together with a synchronized beep.
- The synchronized sender sends the message and the receiver adjusts its offset according to the differences in their time.
- Thus, all the motes get synchronized and synchronously blink and sound together in unison.
- Whenever new nodes are added or old nodes are removed from the network then the network still converges to a synchronized state in a small time..

7.4 Challenges

- A node in a WSN expends a significant amount of energy for wireless communications. The radio transmitter certainly consumes a lot of energy sending each message, but the receiver does too, even when it is just listening.
- For time synchronization to work there must be a fixed point in time from which both sender and receiver can reference the timestamp in a given message.
- Duplicate notifications of the same event from a group of nearby sensors can result in significant energy consumption.

CHAPTER 8: TIME SYNCHRONIZATION APPLICATIONS

- Wireless sensors were deployed on the Great Duck island, off the coast of Maine to study the behavior of the rare bird named Leach's Storm Petrel.
- Monitoring projects like monitoring volcanic eruptions in central Ecuador.
- Measuring micro climatic variations in humidity, temperature and pressure in botanical gardens and vineyards.
- Industrial monitoring and supply chain management.
- Energy efficient MAC protocols with sleep periods.
- Collaborative transmission using space-time coding.
- Object tracking: The size, shape, direction, location, velocity, or acceleration of objects is determined by fusing proximity detections from sensors at different locations.
- Consistent state updates: The current state of an object is most accurately determined by the node that has sighted the object most recently.
- Duplicate detection: The time of an event helps nodes determine if they are seeing two distinct real-world events, or a single event seen from two vantage points.
- Temporal order delivery: Many data fusion algorithms must process events in the order of their occurrence —for example, Kalman filters.

CHAPTER 9: CONCLUSION

Time Synchronization is a problem in wireless networks. Wireless sensor networks are anticipated to play a key role in observing, collecting and disseminating relevant information about a variety of interesting phenomena. Traditional synchronization used with wired networks cannot be used with wireless networks as the network is dynamic and the sensors mobility pose a problem.

The implementation of the time synchronization using Uniform Convergence depicts the synchronous nature of the motes(Telosb and Micaz) which blink together and beep together to show that they are in sync. Thus, a sender receiver based technique is successfully implemented where inspite of the addition of new nodes or removal of nodes from the network, the network still converges to a synchronized state in a very small time.The Uniform Convergence protocol doesn't use a tree based structure and this is an advantage for synchronization.

Synchronization protocols fall short of some major roles and their topology
is tree based .They are not tolerant to node or link failures. Desynchronization is also unable to give a full proof working solution. This gave way for a new research in the consensus field.

Thus, the need of a "fault-tolerant" consensus algorithm, which is robust to link failures that are common in a setting where nodes are mobile, and have limited battery power and transmission range was needed. In recent years, some consensus results of the scale free networks have been exciting and encouraging. Average consensus algorithm, a gossip algorithm, have been used for distributed computing over a long period now and is being recognized as a promising research direction.

Bibliography

Benezit, F. (2009). Distributed Average Consensus for Wireless Sensor. *EPA*, 18.

Bharath Sundararaman, U. B. (2005). Clock Synchronization for Wireless Sensor Networks: A Survey.

Boyd, L. X. (n.d.). Fast Linear Iterations for Distributed Averaging. CA.

Demirbas, M., Arora, A., Mittal, V., & Kulathumani, V. (2006). A Fault Local Self-Stabilizing Clustering Service for Wireless Adhoc Networks. 912 - 922.

Elson, J. E. (2003). *Time Synchronization in Wireless Sensor Networks.* Los Angeles.

Er. Rajni Kaushal, A. N. (2013). Analytical Study of Time Synchronization Protocols for Wireless Sensor Networks. *International Journal of Computer Trends and Technology- volume4Issue3- 2013*, pp. 367,368,369,370.

Fikret Sivrikaya, B. Y. (n.d.). Time Synchronization in Sensor Networks: A Survey. NY.

Forouzan, B. A. (n.d.). *Data Communications and Networking.* Mc Graw -Hill.

Gay, P. L. (2009). *TinyOS Programming.*

http://www.cs.rpi.edu/~yener/PAPERS/WINET. (n.d.).

http://www.cse.wustl.edu/~jain/cse574-06/ftp/time_sync/#Section1.2. (n.d.).

http://www.eecs.harvard.edu/ssr/projects/sync. (n.d.).

http://www.freebsd.org/doc/en/books/handbook/network-ntp.html. (n.d.).

http://www.hindawi.com/journals/ijdsn/2013/192128. (n.d.).

http://www.internationaljournalssrg.org. (n.d.).

http://www.sciencedirect.com/science/article/pii. (n.d.).

http://www.sciencedirect.com/science/article/pii/S0140366411002994#. (n.d.).

Hui Kang, J. L. (n.d.). A Localized Multi-Hop Desynchronization Algorithm for Wireless Sensor Networks. Stony Brook.

Jianping He, P. C. (2011). Time Synchronization in WSNs: A Maximum Value Based Consensus Approach. *50th IEEE Conference on Decision and Control(CDC-ECC)*, (p. 7882).

Julius Degesys, I. R. (n.d.). DESYNC: SelfOrganizing Desynchronization and TDMA on Wireless Sensor Networks. Harvard .

Kulathumani, V. (n.d.). *Time Synchronization.*

Lei Fang, P. J. (n.d.). Asynchronous Consensus Protocols:Priliminary Results ,Simulations and Open Questions.

Nedic, B. T. (n.d.). Distributed Consensus over Network with Noisy Links.

Osvaldo Simeone, U. S.-N. (n.d.). Distributed Synchronization in Wireless Networks.

Scaglione, A. (n.d.). On the wireless communication architecture for consensus problems.

Srivastava, S. G. (n.d.). Timing-sync Protocol for Sensor Networks. LA.

Stacy Pattersony1, B. B. (n.d.). Convergence Rates of Distributed Average Consensus with Stochastic Link Failures. Santa Barbara, CA.

Stephen Dawson-Haggerty, O. G.-E. (2009). *TinyOS 2.1.* San Francisco.

Xianbing Wang, Y. M. (n.d.). Lower Bounds for Achieving Synchronous Early Stopping Conensus with Orderly Crash Failures. Hong Kong.

Zhao Dengchang, A. Z. (2013). Time Synchronization in Wireless Sensor Networks Using Max and Average Consensus Protocol. *Hindawi*, 1-3.