UNIVERSITY OF CINCINNATI

Date: 02/17/2006

I, Hemant R. Mohapatra,

hereby submit this work as part of the requirements for the degree of:

Master of Science

in:

Computer Engineering

It is entitled:

Energy-Aware clustering and localization algorithms for Mobile Sensor Networks

This work and its defense approved by:

Chair: Dr. Yiming Hu

Dr. Carla Purdy

Dr. Kenneth Berman

Energy-Aware clustering and localization algorithms for Mobile Sensor Networks

A thesis submitted to the

Division of Research and Advanced Studies
of the University of Cincinnati

in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE

in the

Department of ECECS

Spring 2006

by

Hemant R Mohapatra

B.Tech. (Indian Institute of Technology, Bombay) 2002

Committee in charge:
Dr. Yiming Hu, Thesis Advisor and Chair
Dr. Carla Purdy
Dr. Kenneth Berman

ABSTRACT

This paper introduces an energy efficient clustering algorithm for sensor networks based on the LEACH protocol. The proposed algorithm adds features to LEACH and aims to reduce the consumption of the scarce network resources in each round of data gathering and communicating, leading to a more graceful breakdown of the network. Also, fault tolerant technique has been proposed to account for the mobilization of a cluster-head or a member node during a round. The algorithm has been simulated on a test-bed of sensor nodes ranging from 50-350 nodes and results show marked reduction in the network energy consumption when compared to LEACH.

The second part of the thesis develops a sensor network localization algorithm based on the Time of Arrival (TOA) technique. Using field based experiments we have found that our localization algorithm converges to a stable distance estimation much quicker when compared to a normal TOA based estimation. This leads to a significant amount of network energy conservation that can be effectively utilized for data transmission and sensing.

ACKNOWLEDGMENTS

I would like to thank my advisor, Dr. Yiming Hu, for providing me so much freedom to explore and discover new areas of Wireless Sensor Networks. His various suggestions and ideas have helped me improve the quality of my work and bring it to the level it is at today. I would also like to thank my other committee members, Dr. Carla Purdy and Dr. Kenneth Berman who graciously agreed to be a part of my thesis committee.

During the course of working on my simulator, I have had the opportunity to discuss my work with many other open-source developers. I would like to thank all those - too numerous to list - who have directly or indirectly provided me with guidance and suggestions for writing a good sensor network simulator. Last, but not the least, I would like to thank my family and friends who have been highly supportive and helpful all throughout the period of my graduate studies.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The fast-emerging field of wireless sensor networks combine sensing, computation and communication into tiny remotely deployable devices called sensor nodes. These nodes usually have a networking stack that supports mesh protocols used to form a widely deployable, well connected data acquisition and transmission base [29, 54, 60]. They also employ smart routing protocols to ensure maximum connectivity in hostile environments [27, 64, 18]. For example, as the fire spreads in a building, the mesh networking connectivity will seek out and use any possible communication path to deliver data to its destination. These sensor nodes are very limited in resources (battery, transmission range, memory) but the composition of hundreds of these devices has unprecedented new technological opportunities.

1.1 Features of WSNs

The mainstay of wireless sensors lies in their adaptability to configure themselves into organized networks and keep sensing data till they run out of battery or are damaged by environmental incidents. Their usage varies from real-time tracking of vehicular activities to monitoring environmental conditions (for example, forest fire, dam water level, humidity etc.), to ubiquitous computing and structure/equipment health monitoring.

Wireless sensors can be used not only as data acquisition devices but also as actuators that can control physical devices based on some electronic triggers. For example, a nuclear plant may use hundreds of tiny sensors deployed around the reactor to detect radiation leaks and trigger an alarm mechanism if the radiation levels are beyond the safety threshold. Unlike a traditional wired network where hundreds of meters of cables need to be laid down through expensive protective conduits, wireless sensor networks are much cheaper to deploy. They are small in size which is an important characteristic since they are primarily used for *non intrusive sensing*. Also, sensor networks are highly scalable as they can be easily extended by just adding extra devices to the already existing infrastructure. Once deployed in the field, the nodes can gather and report data for years running on a single set of batteries.

Wireless sensor networks not only reduce installation and maintenance costs, they are also self-configurable (i.e., they form a working efficient networking mesh by themselves through mutual interaction and decision making) and adapt to the changes in the environment making them very fault-tolerant. The self-organization feature of sensor networks makes it possible to deploy them randomly over the region we are interested to observe without doing a previous survey of the area. These sensors can be installed in any random fashion (for example, dropped from an aircraft) without the knowledge of where the other sensors are located.

On a comparative study of different wireless devices, one would find that wireless sensor nodes do not necessarily communicate directly with the nearest base-station or control tower like many other Wi-Fi devices. They usually just contact their nearest neighboring sensor node and use predefined peer-to-peer protocols to establish a mesh-like interconnect to transmit data between hundreds of nodes in a multi-hop fashion. This mesh like network is highly adept to support newly inducted nodes or to expand to cover a larger geographical area. The strength of a wireless sensor network depends on the number of nodes it constitutes of. Unlike a cell phone network where new calls are denied in the presence of too many active calls in a small area, the interconnection of a sensor network only becomes stronger as nodes are added to it.

There is extensive research in the development of new algorithms for data aggregation [30], clustering [58, 28, 39], positioning [57, 12, 11, 46], ad hoc routing [37, 32, 47], and distributed signal processing [15] in the context of wireless sensor networks. As the algorithms and protocols for wireless sensor network are developed, they must be supported by a low-power, efficient and flexible hardware platform.

1.2 Contribution of the Thesis

In this thesis, we have attempted to solve two different issues that are basic to any sensor network: ”An energy efficient clustering algorithm for WSNs” and ”A fast converging algorithm to find the location of a sensor node in a sensor network”.

For the sake of academic simplicity, some basic assumptions have been made to isolate these two issues so that solutions for ideal cases can be formulated. While trying to find an efficient clustering algorithm, it is assumed that all the nodes are already aware of the location of all other nodes in its range. As we shall see later, this is an important assumption for the success of our algorithm. Similarly, when trying to find an efficient algorithm for finding the location of a sensor node, it has been assumed that there are at least *three* other nodes in the vicinity of the node we are trying to find the location of. Also, we assume that nodes have enough battery power to allow our algorithm to converge to a stable solution.

1.3 Structure of the Thesis

In the next chapter, we give a brief introduction to the field of wireless sensor networks. First we discuss the layout of the physical devices and then go on to present some issues on location awareness of the nodes, the underlying network architecture and the energy model of a sensor network.

Chapter 3 discusses clustering mechanisms in sensor networks and then describes a specific clustering algorithm, LEACH, upon which our own clustering model is based. Chapter 4 describes our clustering model in detail and compares some of it's features with LEACH to establish the benefits of our model. Results of the simulation experiments conducted with our model are presented in Chapter 5. We show that our model results in a lesser number of dead nodes and achieves a higher network lifetime as compared to LEACH.

Chapter 6 presents an outline of the theory behind sensor network localization. It also discusses a few location finding algorithms currently in use. We also discuss our algorithm for finding location of nodes in a sensor network at the end of Chapter 6. In Chapter 7, results of the field experiments conducted to test our localization algorithm are presented. Chapter 8 concludes this thesis with some final observations and possibilities for future research.

Chapter 2

Wireless Sensor Networks

Advances in micro-sensor technology have resulted in the development and deployment of small low cost and low power devices with sensing, computation and communication capabilities. These devices may be deployed in large numbers to form a WSN that monitors a specific parameter like temperature or humidity. Although, individually, these devices may not be very accurate or reliable but their deployment in large numbers greatly enhances their reliability. In addition, WSNs can provide area coverage in a way that was not possible with other wireless devices. They can be used to gather data in very hostile environments such as volcanic sites, chemical/nuclear plant or in an inaccessible environment such as a spaceship or remote terrains.

In this chapter, we shall discuss the application specific design issues pertaining to sensor networks. Also, a few sample scenarios are presented where sensor networks prove useful for data acquisition and system control.

2.1 Current Technology

MEMS technology has enabled a new generation of massive-scale sensor networks suitable for a range of commercial and military applications. The main focus of the production cycle is to keep the devices low-cost and small in size while keeping the power consumption due to data transmission at a minimum.

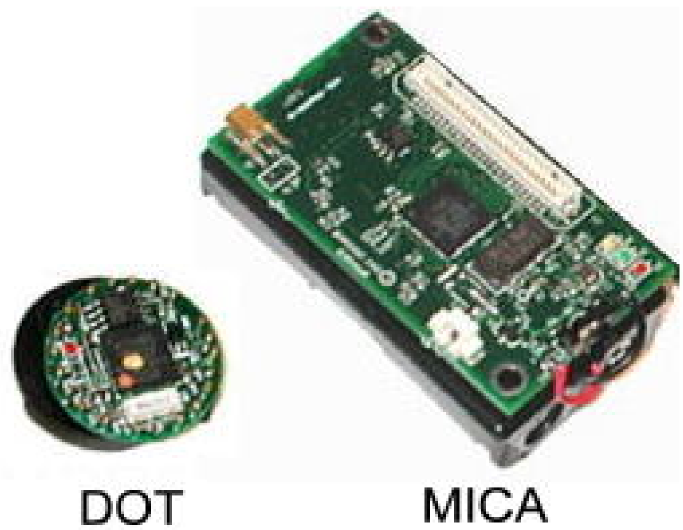


Figure 2.1: MICA Motes Hardware

The Berkeley/Crossbow Mica Motes (Fig. 2.1) are the size comparable to that of a US quarter (25mm) with a multi-channel transceiver, an on-board temperature sensor and a processing unit [5]. The transceiver can work on 8968/916 MHz or 433 MHz Industrial, Scientific and Medical (ISM) bands.

Another successful implementation of small low-power sensor devices in the field is the Smart Dust (Fig. 2.2). These devices are also called motes and they can be s small as the size of a grain of sand, or even a dust particle. Each device contains sensors, computing circuits, bidi-

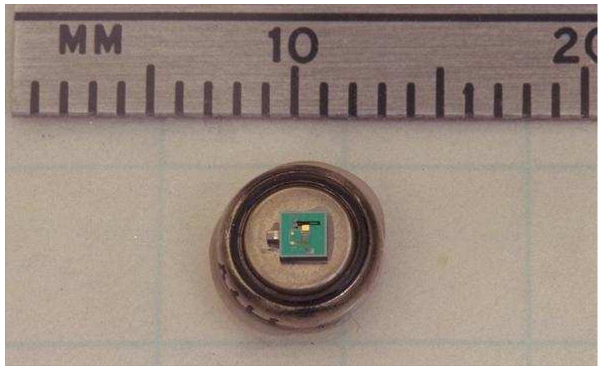


Figure 2.2: A Smart Dust Node [6]

rectional wireless communications technology and a power supply. Motes would gather data, run computations and communicate using two-way band radio with other motes at distances approaching 1,000 feet (300 meters).

2.2 Application of Sensor Networks

Sensor networks are designed to perform high-level information processing tasks such as detection, tracking or classification [61]. There are well defined parameters for each of these tasks, including detection of false alarms, tracking quality or accuracy and classification errors. Applications of sensor networks are quite wide in their scope and vary a lot based on their application requirements: mode of deployment (ad-hoc or manual), sensing modality or power supply (e.g., battery versus wall socket). Most of the sensor network applications can be roughly divided into three parts:

- **Space Monitoring:** Includes environmental and habitat monitoring, indoor climate control, surveillance and intelligent alarms.

- **Condition Monitoring:** Includes structural monitoring, condition based equipment maintenance, medical diagnosis and urban terrain mapping.
- **Interaction Monitoring:** Monitors interaction of things with each other and the encompassing space. Includes disaster management, emergency response, asset tracking, ubiquitous computing and health-care.

This section discusses some of the more popular applications of sensor networks that have been successfully deployed and are currently in use.

2.2.1 Environmental and Habitat Monitoring

Many early variants of WSNs were deployed for environmental and habitat monitoring. It involves complex readings over time across a volume of space that is large enough to exhibit significant internal variations of various sensed parameters. On a small patch of land 10 miles off the coast of Maine, a team from UC Berkeley are conducting an experiment in networked sensing. About 190 wireless sensors are being used to constantly monitor the habitat of the nesting petrels [7] on Great Duck Island.

2.2.2 Bush-fire Response

An ongoing project in the University of Melbourne, Australia, is working towards an emergency response mechanism against forest fires [1]. The basic framework includes an integrated network of sensors on the ground monitoring local moisture levels, humidity, wind speed and direction, together with satellite imagery and longer term meteorological forecasting. This facilitates the determination of fire risk levels in targeted regions as well as valuable information on probable fire direction.

Such a network will provide valuable understanding of bush-fire development and most importantly assist authorities in organizing a coordinated disaster response that will save lives and property by providing early warning for high risk areas.

2.2.3 Smart Transportation

Plenty of sensors are already in use for traffic monitoring purposes. Sensors embedded on roadbeds or alongside highways can measure traffic flows and also come up with correlative data based on time frames of maximum and minimum traffic. Such a data can be very useful in the design of an effective road network as well as constructing safer intersections.

Sensors embedded in the vehicles can warn drivers about the pollutant levels in vehicular emissions, engine temperature and even tire pressure. Sensors can also be used to report traffic violations like over-speeding or checking for vehicle overload before it passes over a bridge. Japanese transportation authority is already using sensors to direct vehicles to the nearest available parking space.

2.2.4 Structural Health Monitoring

Smart Dust Motes, developed by UC Berkeley, enable the researchers to find out the structural integrity of a building at any given time [42]. It can also predict the behavior of the building in near future. Outfitted with wireless radio transceivers, the battery-powered matchbox-sized Motes can be built to sense numerous factors; from light and temperature for energy saving applications to dynamic response for a given structural load, the key characteristic for civil engineering. Even the tiniest movement of a supporting column in a building can reveal the structural soundness and, for instance, suggest that the column is handling more load than it should due to a problem elsewhere in the structure.

Currently, wired seismic accelerometers - which measure movements of structural units - cost upwards of $8,000 each and are tricky to install. As a result, their deployment in buildings is kept to a minimum. The trouble with the current minimalist paradigm of structural monitoring is that a handful of accelerometers in a large building can only provide a big picture of a building's structural integrity. As a result, a problem only becomes visible once the entire building is affected and safety has already been compromised. These issues are being effectively resolved by the use of Smart Dust sensors.

2.3 System Architecture

Sensors typically consist of three main components: sensing electronics, data processing unit, transceiver (communication unit) and power unit (Fig. 2.3). Information from the physical world is retrieved using the sensing unit and converted with an analog to digital converter (ADC) to digital data. This data is forwarded to the processing unit which encapsulates it into a packet and sends it to the sink node for further examination. The node-to-sink communication is facilitated by the transceiver. The power unit supplies operational power to all the components used above.

The optional units, location finder and mobilizer, are used depending on the application. Most application need some knowledge about the location of a sensor node to put the sensed data

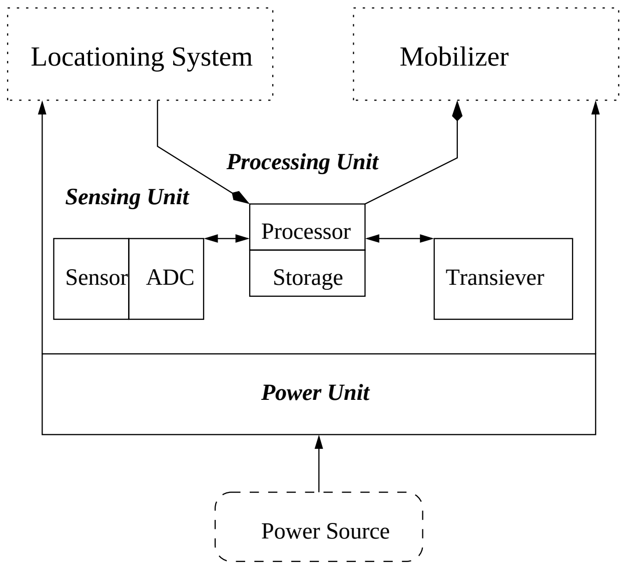


Figure 2.3: Components of a typical Sensor Node redrawn from [29]

into a system-wide perspective. Some other scenarios, like tracking a moving target, may need to use the mobilizer unit as a major system component. Some other sensors also have an additional power generating unit (not shown in Fig 2.3) to scavenge for power to prolong the life-time of a sensor node. Currently, solar cells are the primary power-scavenging tools.

Out of these three, the *communication* unit is the most important as it is responsible for providing the sensors with the functionality of network formation which is the defining characteristic of a smart sensor.

2.3.1 Location Awareness

Sensor nodes are usually spread randomly across the field and therefore, they are not aware of their own exact location. Many sensor network based applications, however, require an estimate of the nodes position to achieve the desired functionality. Many location estimating algorithms have been devised in the past [57, 12, 11, 43] and they have their relative pros and cons. Using GPS for positioning has been suggested in [52] but this is not a feasible solution [57, 14]. First, these GPS locators are physically large and also use up a lot of energy in establishing their coordinates. Secondly, due to the inherent nature of usages of sensor networks, often the sensor nodes are situated in the fields where a direct line-of-sight with the GPS satellites is not possible. Also, GPS devices are quite expensive and therefore, raise the overall cost of the sensor network infrastructure.

Many other position estimation methods are proposed in literature. In [36], a centralized algorithm is proposed which uses convex position constraints derived from connectivity information. The position information that is thus calculated is relative to other neighboring nodes whose location information are known *a-priori*.

In [12], a radio frequency based technique has been proposed to estimate node location. Each sensor network has various beacons spread evenly across which periodically signal overlapping location information to the whole network. Assuming that each node whose location is unknown has atleast three beacons in range (to facilitate triangulation), nodes localize themselves to the centroid of their nearest three beacons.

RSS (Received signal strength) [48] uses the signal strength and attenuation characteristics to calculate the distance between two nodes. RSSI techniques have low accuracy when used to measure distances larger than a few meters.

Techniques like Time of Arrival (TOA) [41], Time Difference of Arrival (TDOA) [55] and Angle of Arrival (AOA) [44] use prohibitively large antenna arrays and are therefore limited in their usefulness. Later, in chapter 6, we have built an algorithm based on the basic TOA techniques that gives a more accurate position estimation for short and mid-range distances.

2.3.2 MAC Layer Protocols

Sensor nodes are a unique variety of wireless devices. They are usually battery powered and therefore, it is usually not possible to change or re-charge the batteries once these nodes have been deployed in the field. Prolonging the battery life of the sensor nodes is therefore the primary aim of any MAC protocol that has been designed for sensor networks. Another important attribute of sensor networks is their scalability and their adaptability to change in network size, node density and the network topology. A good MAC protocol should absorb these frequent changes in network attributes and provide the users with a fault-tolerant infrastructure on top of which applications can be built.

Several MAC layer protocols for sensor networks have been proposed. For a detailed list, the reader could refer to [29]. Typical examples include *Time Division Multiple Access* (TDMA) [34], *Code Division Multiple Access* (CDMA) [2] and contention based protocols like IEEE 802.11 [3].

In [33], the authors have discussed a Self-organizing Medium Access Control for Sensor networks (SMACS). This is a distributed protocol that builds a flat topology for sensor networks. Eavesdrop and Register (EAR) algorithm [33] represents the mobility management aspect of SMACS protocol and enables continuous connection between two nodes in case of node mobility.

In [19], TDMA and FDMA MAC protocols have been discussed. In TDMA, the transmission time is minimized but the full network bandwidth is available to a node for transmission. However, TDMA leads to inefficient use of bandwidth. To improve channel usage, FDMA [53] is used where the bandwidth is subdivided into different frequencies which are assigned to different sensors so that no two sensors have the same transmission frequency.

In our thesis, we have assumed a TDMA based MAC layer. Each simulation round is divided into TDMA time slots and each sensor node sends data to the base-station (or cluster-head) only in its allocated time-slot. TDMA was chosen as the MAC layer protocol purely for it's simplicity and ease of simulation. It must be noted, however, that the work done in this thesis is in no way dependent on the MAC protocol being used by the sensor network.

2.3.3 Routing Techniques

Due to the inherent unstable nature of a sensor network and the power constraints that are put on any network based activity, the standard TCP/IP techniques for routing such as implementation of an IP addressing scheme and maintaining large routing tables at each router node, are rendered useless. In a sensor network, every node has to act as a router if need be. Furthermore, proactive protocols of Distance-Vector (DV) variety need to broadcast to the whole network in case of a change of topology. Since topology of a sensor network changes quite often, a DV type protocol for routing would use up most of the bandwidth. Link-state (LS) [13] type protocols broadcast to only local neighbors in case of a topology change but these protocols converge very slowly and need a lot of transmissions to synchronize multiple changes in topology.

There are various suggested routing techniques to reduce the load on a sensor node in case of a topology change (see *fisheye state routing*) [49] and reactive protocols like *dynamic source routing* (DSR) [32] and *ad-hoc on demand distance vector routing* (AODV) [50]. However, in a sensor network scenario, it is needed to focus on local stateless algorithms that do not require a node to know much more beyond their immediate neighbors.

The primary difference between a sensor network and other networks is that in sensor networks, the classical differentiation of *address* and *content* is no longer valid [18, 26]. This gives rise to a data-centric view of the network [18] where routing decisions are made on the basis of destination attributes and its relation to the attributes of the packet content and *not* on the basis of destination address. Based on this criteria, the routing mechanisms for sensor networks can be classified into two broad categories, described next.

Geographic, Energy Aware Routing

The routing protocols that deliver packets based on the geographic location of a node fall under this category. As is the case with most other sensor network routing protocols, these protocols discover routes *on demand* using light-weight scalable techniques. Therefore, the primary challenge is that of path discovery while keeping it time and energy-efficient. These protocols assume that all nodes know their as well as their one-hop neighbors' geographic locations. Routing destination is specified either as a node with a given location or as a geographic region.

Some common protocols under this category are greedy distance routing [66], compass routing [66], convex perimeter routing [66] and greedy perimeter stateless routing (GPSR) [35]. The geographical and energy-aware routing (GEAR) protocol, discussed in [64], also does some load-balancing on the nodes to avoid a fast depletion of energy.

Attribute Based Routing

This category of sensor network routing protocols represent a more general nature of the network where each node is not aware of its geographical location. Attribute based routing strives to establish a connection between nodes who *need* a certain information to the node *having* that information. To perform this kind of matching, it is assumed that all data that is collected is stored in an object-oriented attribute fashion.

Various attribute based routing techniques are currently in use. *Directed Diffusion* [31] uses a general *sink* and *source* approach. This is an inefficient method of routing as it resorts to an initial flooding of the network in order to discover good paths between sources and sinks. If the amount of information needed is small, *rumor routing* [10] is an attractive alternative. Geographic hash tables (GHT) [22] is another robust attribute based routing protocol that treats the sensor network as a distributed database that stores observations and readings from the sensors for possible later retrieval by query requests that are injected anywhere on the network.

2.3.4 Packet Structure

The design of a data packet is an important feature of sensor network architecture. If the size of the packets are too small, too many transmissions would be needed to deliver data. If they are too large and the data to be sent is small, it may lead to heavy wastage of payload space. As described in the previous section, any of the routing techniques can be chosen (geographic or attribute based) by a given sensor network and all nodes must agree to transmit packets in accordance to that protocol. Based on which routing technique is chosen, the packet structure is also affected.

Fig. 2.4 shows a basic link layer packet structure [62]. The packets are divided into three major parts, the packet header, the payload and the trailer. They are l_h, l_p and l_t bits long respectively. The header contains information about the source and destination identifier. In case attribute based routing is being used, the headers contain attribute identifiers (described in the previous section). Trailers contain error control bits.

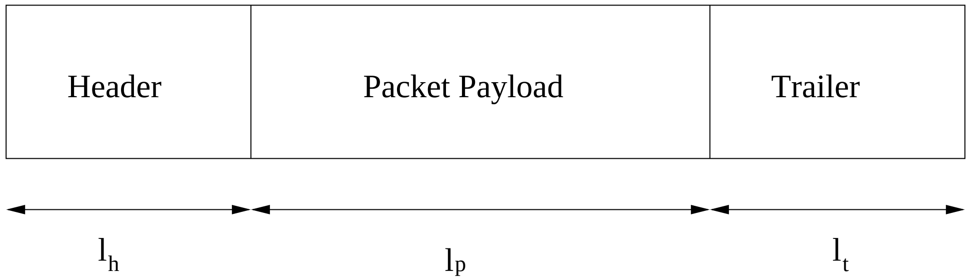


Figure 2.4: Basic structure of a typical Sensor Network Packet, redrawn from [62]

The payload size depends on the type of information that is being transmitted. In case of temperature, humidity and pressure like attributes, the payload can be as small as a few bytes. For power-aware routing, the payload also contains the current battery power for each node. This information is used to calculate an efficient route from the source to the destination. In Chapter 4 we shall see how we have modeled our packets to incorporate mobility management and fault tolerance in our clustering method.

Table 2.1: Optimal packet size in Link Layer [62]

FEC Method	η	Min	Max
Without FEC	0.70	100	500
BCH, $t = 2$	0.88	400	800
BCH, $t = 4$	0.93	1000	1500
BCH, $t = 6$	0.95	1500	3000

Table 2.1 [62] shows a detailed analysis of payload estimation for energy efficiency (η). The payload size without an error control mechanism is found to be in the range of 50-500 bytes. In the presence of an error correcting capability, the payload size varies from 1500 to 3000 bytes. Here, BCH codes are used for error correction with different error correcting capabilities (t = maximum number of bits that can be corrected simultaneously).

2.3.5 Energy Model

Sensor nodes spend most of their energy in sending/transmitting information between each other or to the base-station. To reduce energy consumption is one of the primary design parameters for any sensor network [25]. If the sensors are not equipped with energy-scavenging tools like solar cells, once their batteries are exhausted, they are no longer useful to the application. As we know, sensor networks are based on mesh-networking where each sensor node routes the information one step closer to the sink (or base station). For this reason, it is all the more important that we keep alive as many nodes as possible so as to maintain the overall connectivity of the network. A disconnected subdivision of the network can render the entire network useless [65] and therefore, the level of power consumption must be considered at each stage in designing a sensor network.

Transmission Power Usage

In a radio propagation model in a single-path free-space channel, the transmitted power P_t is related to the received power P_r by the following relation:

$$\frac{P_r}{P_t} = G_t G_r \left(\frac{\lambda}{4\pi d}\right)^2 \tag{2.1}$$

G_t and G_r are transmitter and receiver antenna gains and d is the distance between the transmitter and receiver antenna. $\lambda = c/f$, the wavelength of the transmitted signals, c and f are the speed and frequency of the signals. Using Eq. 2.1, we can derive:

$$P_t = \omega d^2 \tag{2.2}$$

where $\omega = (P_r/G_tG_r)(4\pi/\lambda)^2$. Eq. 2.2 can be re-written in a more general form as:

$$P_t = \omega d^{\alpha} \tag{2.3}$$

where $\alpha (> 1)$ is known as the path loss exponent. For free-space channels, $\alpha = 2$. In most sensor applications, α is assumed to be between 2 and 5.

There are two ways in which packets can be routed from source to the destination node: (1) direct transmission from source to destination (1 hop) or (2) each node forwards the packets all the way through to the destination (N-hop). From Eq. 2.2, we can infer that the power consumed at a receiver node (P_r) is related to the power while transmitting (P_t) by the equation [66]:

$$P_r \propto P_t/d^{\alpha} \tag{2.4}$$

which can be rewritten as:

$$P_t \propto d^{\alpha} P_r \tag{2.5}$$

Hence, for an N-hop transmission over a single-hop transmission, we have a power advantage given by:

$$\eta_{RF} = \frac{P_{t(Nr)}}{N.P_{t(r)}} = \frac{(Nr)^{\alpha} P_r}{N.r^{\alpha} P_r} = N^{\alpha - 1}$$

The N-hop mode of transmission is, therefore, a standard choice for sensor networks due to the obvious power saving advantages. Studies have shown that clustering when coupled with an N-hop environment allows the most efficient use of scarce resources (e.g., power and bandwidth) thereby prolonging the life of the network [51]. In our next chapter, we shall analyze one of the most common N-hop based clustering algorithm, LEACH, and lay the ground-work for further chapters where we incorporate features to LEACH to increase the network lifetime and productivity.

Chapter 3

Clustering in WSNs

Clustering is the method by which sensor nodes in a network organize themselves into hierarchical structures. This way, they are better equipped to use the scarce network resources, such as bandwidth, power and frequency spectrum, more efficiently. Clustering also simplifies the sensor network protocols much the same way sub-netting simplifies the Ethernet protocols. For example, much simpler protocols for broadcasting and routing can used within the cluster. Also, the same time (for TDMA) or frequency slots (for FDMA) can be reused in other non-overlapping clusters. Clustering also allows ease of network health monitoring since some of the nodes can play the role of the watchdogs looking for misbehaving/dead nodes.

Out of various types of sensor network nodes, we restrict ourselves to the case where all the nodes have identical battery power and communication range. There are various ways of choosing a cluster head in such a situation and the basic parameter assumed in most of them is a unique node-ID which distinguishes one node from another. Some of the ways in which these node-IDs are assigned are discussed in [9], [21] and [20]. Some nodes that can reach one or more cluster-heads can act as *gateways*. Non cluster-head nodes choose their cluster heads soon after deployment and transmit data to them. Cluster heads have the job of transmitting data to the base station. Various algorithms have been proposed to get all the benefits of clustering while distributing the load and battery drain of being the cluster-head evenly among all the nodes so that the network dies gracefully and not in segregated disconnected groups [8, 24]. In this chapter we shall concentrate on the LEACH protocol and discuss some of the pros and cons involved with using this protocol for cluster-head selection.

3.1 Clustering with Leach

LEACH (Low-Energy Adaptive Clustering Hierarchy), an energy-conserving communication protocol for wireless sensor networks, was proposed by Heinzelman, Chardrakasan and Balakrishnan in [28]. The basic application scenarios for this protocol are:

- The base station is far from the sensor nodes and is fixed.
- All the sensor nodes in the network have the same initial battery power and are homogeneous in all other ways.

LEACH is a dynamic clustering mechanism. Time is divided into different *rounds* or *intervals* of data transmissions of equal length. For each time interval t_d, cluster-heads are regenerated and the cluster reconstructed. Each sensor node i at the beginning of a round, generates a random number such that $0 \leq random \leq 1$ and compares it to a pre-defined threshold value *T(i)*. If *random* $<$*T(i)*, the sensor node acts as a cluster-head for that round, otherwise it becomes a cluster member.

Assuming that P is the percentage of cluster-heads in a given network, we define:

- $n = 1/P$
- M = current round the network is running in
- G = set of nodes that have *not* been cluster-heads in the last n rounds.

According to [28], the value of threshold for a sensor i is:

$$T(i) = \begin{cases} \frac{P}{1-P(M mod(n))}, & \text{if sensor } i \in G, \\ 0 & \text{otherwise} \end{cases} \tag{3.1}$$

Once a node decides to become a cluster-head, it broadcasts this message to its neighbors (with an upper bound on how *far* it can broadcast) and each non-cluster-head node decides which cluster-head to join based on the signal strength of the cluster-head broadcasts. For the rest of the interval, the nodes send data to their respective cluster-heads and the heads aggregate, compress and route the data to the base station. After each interval, the whole clustering set-up phase restarts. This rotation of nodes becoming the cluster-head allows the network to spend its energy more evenly across all the member nodes and remain active for a longer time. Studies by the authors have shown that LEACH can extend the sensor network life up to eight times longer than its closest competitors (e.g., static clustering, direct transmission and minimum energy transmissions).

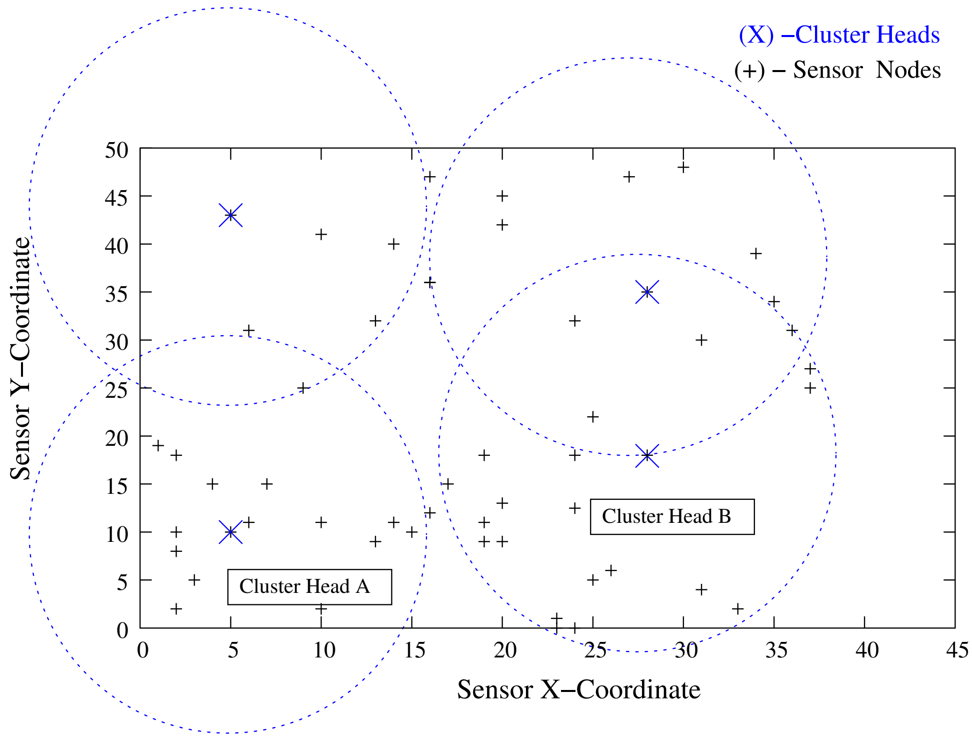


Figure 3.1: A typical Sensor Network layout

3.2 Disadvantages of LEACH

Despite the obvious advantages in using the LEACH protocol for cluster organization, there are a few features that the protocol does not support. LEACH assumes a very homogeneous spread of sensors in a given area of interest. All the sensor nodes are assumed to be reasonably far away from each other so as to to provide support for all non-cluster-head nodes. In a real life scenario, this often is not the case. For example, let us consider a sensor distribution like the one shown in Fig. 3.1 in which most of the nodes are grouped together close to one or two cluster-heads (cluster-head A and B).

Here, A and B will send broadcasts to their neighbors and will end up having too many nodes in their cluster. This would lead to a very fast depletion of energy in A and B and a part of the network would lose connectivity with the rest of the network. LEACH does have an upper limit on the number of nodes a cluster-head can accept under it.

Moreover, LEACH has no provision to account for movement of nodes or cluster-heads during the lifetime of a network. If a node moves away from it's cluster-head, it cannot update its cluster-head even if it has another cluster-head closer to itself after having moved. Also, if a cluster-head moves away from its nodes, it may so happen that some nodes that belong to another cluster-head may have this cluster-head at a much closer distance. LEACH does not allow these nodes to join this cluster in the same round.

In this thesis, we have proposed an alternative to LEACH that tries to solve the above mentioned issues. In the next chapter, we will give a detailed account of the algorithm and its mode of operation.

Chapter 4

Our Algorithm - Motivation and Details

The proposed clustering algorithm aims to rectify some of the loopholes left out by LEACH. As discussed in the previous section, our algorithm supports the following functionalities over and on top of LEACH:

- Mobilization of a cluster-head or a member node during active data transfer in a round.
- Current battery power and the number of members currently under a cluster head are taken into account before a node decides which cluster-head to join.

Our work derives from the research done on LEACH and some LEACH influenced cluster management protocols like PEGASIS [59]. We assume that all nodes are aware of their and their neighbors' physical locations. They all have a processor, a memory and the hardware needed to perform sensing, information gathering and communication. Each sensor has a sensing radius which is decided in a way that would cover all the sensors in the network. Each sensor is also assumed to be capable of sensing self movement in the field (discussed later).

4.1 Details

The algorithm (and the experimental simulations) are theoretical in nature and have not been implemented on the field. However, our primary objective has been to establish a simple and robust clustering technique that can improve the network lifetime considerably.

The basic principle for an energy based approach is that the received energy of a signal attenuates with propagation distance at a rate given by the equation [16, 63]:

$$A_{\|x-x_i\|}(t) \propto \| x - x_i \|^{-\alpha}, 1 \leq i \leq N \tag{4.1}$$

where $\| x - x_i \|$ is the distance between nodes x and x_i, $A_{\|x-x_i\|}$ is the attenuation for the distance the signal travels, and α ($\alpha \in [2,5]$) is the path loss exponent.

Based on the above attenuation model, mutual node distances are calculated at deployment time (or if sensor mobility is detected). The clustering mechanism in our model is similar to LEACH but differs in a few important aspects. In LEACH, the nodes choose their cluster-heads based on the signal strength of broadcast transmissions from the cluster-heads. This may lead to some cluster-heads being overloaded if they are inside a densely populated zone on the field. Our model is based on *confidence value* associated with broadcast from a cluster-head. Confidence value of a cluster-head is a function of (a) distance between the cluster-head and the node, (b) number of nodes already a member of this cluster-head and (c) current battery power of the cluster-head. Essentially, our model first checks if, with the current battery power the cluster-head has, it would be able to support the existing members at maximum data transmission rate. A node decides to join a cluster-head if the head can still support the node with its remaining resources. A higher confidence value for a given $\{$cluster-head $\rightarrow$ node$\}$ combination would imply that the node has a higher probability of joining that cluster-head.

Also, a round-based global threshold for the number of cluster members a cluster-head can accept is defined. For a given round, this threshold is equal to the ratio of total live nodes and number of cluster-heads. Confidence value is directly proportional to the remaining battery power of a cluster-head and is inversely proportional to the number of its cluster members and its distance from a node.

Once deployed, cluster-heads interact amongst themselves and share their mutual location information periodically. During initialization in each round, each node that receives a broadcast from a cluster-head stores it in memory. Out of all the broadcasts, the one with the highest associated Confidence value is chosen as the cluster-head by that node.

Algorithm-1 lays out the pseudo-code for our model. Line-18 of the algorithm is where we check if the cluster-head already has too many members.

Once the clustering has taken place, the nodes get busy in doing data-transmissions for the rest of the time left in the round. The nodes transmit data to their cluster-heads which in turn, aggregate and transmit the data to the base station. We shall see in Chapter 5 that our model enhances the network lifetime to a substantial extent.

4.2 Mobility Support

To make sure that mobility of a cluster-head or a node does not affect performance adversely, we first need to identify that a cluster-head has lost contact with a node or vice versa. There are various ways a node and its cluster-head may lose communication:

1. The cluster-head has moved away and established another cluster group leaving behind

Algorithm 1 Our model for clustering

```
R ← 1/Cluster_Ratio
Th ← Threshold
B_T ← BATTERY_THRESHOLD
MCM ← MAX_CLUSTER_MEMBERS

while (current_round < TOTAL_ROUNDS) do
  for (i = 0 to TOTAL_NODES) do
    if (node_i ≠ head in last R rounds) then
      if (random < Th AND node_i.battery > B_T) then
        node_i ← Head
      end if
    end if
  end for
  for (i, k = 0 to TOTAL_NODES) do
    if (node_k ≠ head AND node_i i = head) then
      D ← Distance(node_i ↔ node_k)
      B ← Battery(node_i)
      CM ← Cluster_Members(node_i)
      if ( CM > MCM  OR B < Battery(To support CM + 1 nodes)) then
        Confidence_value = 0
      else
        Confidence_value ∝ B/(MC * D)
      end if
    end if
  end for
end while
```

a few orphaned nodes.

(a) *Solution (node):* The orphaned nodes will need to contact and join the closest cluster-head with the IS_ORPHANED bit set in the headers. If those cluster-heads have already taken up the maximum number of nodes allowed to them, then the second closest cluster-head should be contacted and so on. To simplify the model, we assume that every cluster-head, when contacted by an orphaned node, increases its member capacity to some extent to accept it as a special case. Also note that in the case of node mobility, we do not follow a confidence_value based approach since the primary concern for the node is to get connected to the network again.

   (b) *Solution (cluster-head):* The cluster-head sends another broadcast at the new location and the nodes that find this cluster-head having a higher confidence_value than their current heads, may decide to join under it. All those nodes that were under this cluster-head and did **not** lose contact with it once it moved will also be informed to choose another cluster-head if it is more economical for them.

2. The node has moved away from the range of it's cluster-head.

   (a) *Solution (node):* The node will become an orphaned node and try to establish contact with the nearest neighbors to find it's own location and the location of the neighbors' cluster-head. With this information, it'll join the cluster-head that is closest. The cluster-head will consider the orphaned node as a special case and accept it in it's cluster.

   (b) *Solution (cluster-head):* The cluster-head that has lost a node will identify it (explained later) and remove the lost node from its transmission schedule.

One important task that has to be done before taking the steps mentioned above is to identify the mobility of a node or a cluster-head. We assume that all sensors are equipped to register self-movement. This can be achieved by exchanging periodic location information from other nodes in the cluster . Location data from at-least three neighboring nodes is needed to ascertain if a node has moved. Also, the cluster-head has to be in constant touch with its member nodes. This is achieved using periodic status heart-beat message exchanges. We are assuming a TDMA based MAC protocol [34]. Every cycle is divided into different time slots. As seen in Fig. 4.1, in every TDMA cycle, the cluster-head sends in 2 bits of status information (1 bit is for sign and other bit can be a 0 or 1). This field is initialized to -1 after every TIMEOUT period and is overwritten (with 0) by the cluster-head every time it sends a message to the members.

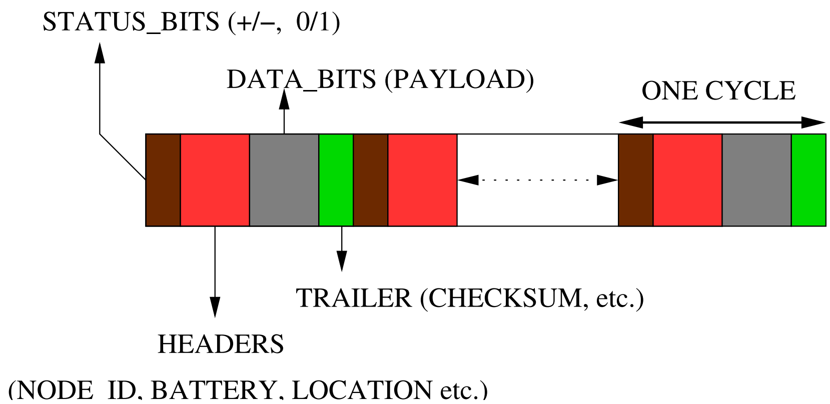


Figure 4.1: A sample TDMA cycle

The member node first reads the headers to find the node-ID of the sender. If it matches the ID of its cluster-head, it accepts the message and before sending any data back, changes the status bit to 1 and overwrites the node-ID field with its own ID. After every cycle, the node resets an internal CYCLE_TIMER (timer limit = TIMEOUT). Assuming that the node has not moved, if it does not receive a status update from the cluster-head before the timer runs out, it assumes that the head has mobilized to some other location and follows a protocol (explained later) to establish contact with other nearby heads. The cluster-head maintains a table of every node currently in it's cluster. If it does not receive a STATUS_BIT for a sensor scheduled to transmit in a cycle, it assumes that the node has moved/broken down and removes it from its database and takes the necessary steps to re-establish the routes etc.

4.2.1 Case I: Mobility of Cluster-Head

If a cluster-head moves to another location it needs to find another cluster for itself to manage. It might become better suited for some nodes it doesn't currently have under its cluster and less suited for some nodes in had in its old cluster before it moved. We need to reconfigure the whole cluster to take care of this change. Take for example, Fig. 4.2: CH1 moves to another location and loses communication with sensor node(23) and node(24).

In this scenario we have the following procedure for fault tolerance. Note that the procedure below is followed after the cluster-head and nodes wait for a time-period of one cycle so that all the nodes under CH1 are aware that their cluster-head has moved:

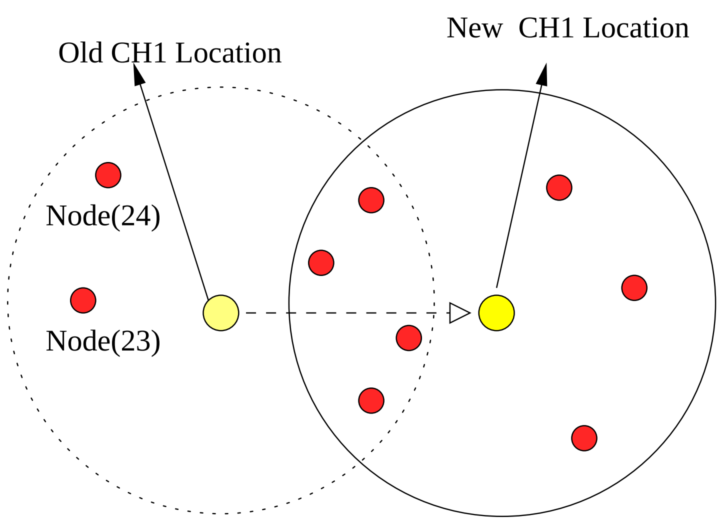


Figure 4.2: Cluster-head mobilization leading to CH1 losing contact with Node(23) and Node(24)

- Nodes who got out of range with CH1 [node(23) and node(24)] will wait for a MOBILITY_TMOUT period (= time spent waiting for a status update from CH1 + time spent waiting for a broadcast from CH1 after it has moved) waiting for status update from CH1. Once they do not receive the updates, they will assume that CH1 is either dead or moved out of range and send a broadcast meant for other cluster-heads requesting to join their clusters. We assume that all sensor have an equal broadcasting range and at deployment, every node has at-least 2 cluster-heads in range. In later stages, some nodes may end up disconnected from the network due to the absence of cluster-heads near them. Also, if a cluster-head which is in range of an orphaned nodes broadcast already has the maximum number of members allowed, it would consider this orphan node as a special case as discussed earlier and would accept it as a member.

- The cluster-head that moves (i.e. CH1) will clear its old member database and send a broadcast to establish another cluster. Members from the old cluster of CH1 still in range of CH1 will end up joining CH1 again. The rest (e.g. node(23) and node(24)) would wait till the end of this broadcast period and infer that they are no longer in range with CH1.

- Some nodes from other clusters may find it more beneficial to join CH1. They will respond to the broadcast sent by CH1 and will join it. Their respective cluster-heads will delete these nodes from their databases once they do not receive any status updates from them.

4.2.2 Case II: Mobility of Nodes

Assuming that all nodes, after startup, via some ranging technique (e.g, triangulation) find out their locations w.r.t. a global coordinate [55], if a node moves to some other location, it might end up in a region where the old cluster-head is no longer best suited for it.

When a node moves, it sends a broadcast to its nearest neighbors. Assuming there are at-least three active neighbors present at all times, we can get the location of this node by triangulation

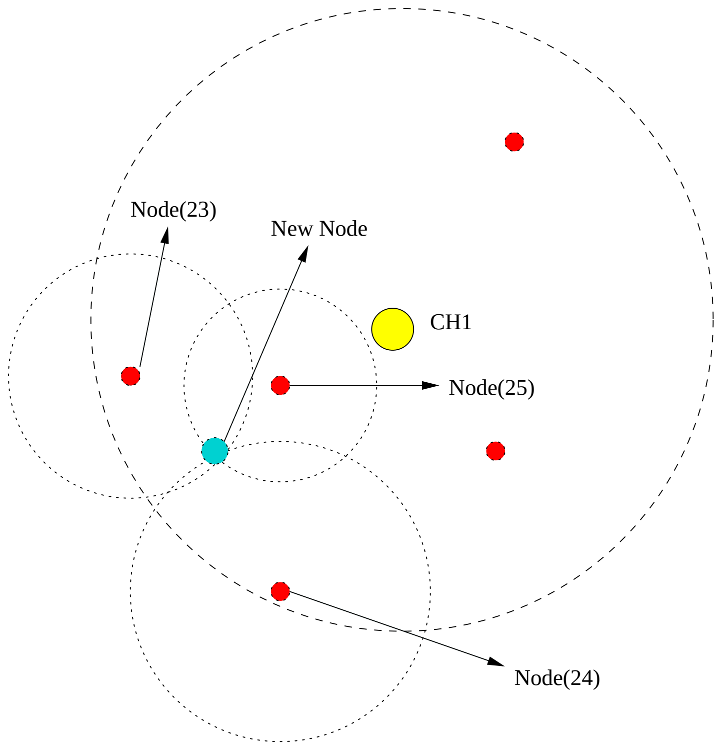


Figure 4.3: Triangulation done by 3 nodes of cluster-head CH2 on a new node that has recently moved away from its old cluster, headed by CH1

as shown in Fig. 4.3. Once the node establishes its new location, it broadcasts a message meant for any other cluster-heads (if any) in range. If there are, then this node gets accepted; otherwise, the node remains disconnected from the network till a cluster-head moves closer to it and tries to establish another cluster. In the scenario where more than one cluster-heads are in range, the head that is closer is chosen by the node. It may so happen that we do not have 3 nodes in the locality of the new node and thus, cannot exactly establish its location. For example, in Fig. 4.4, the new node has only two nodes in its vicinity (node-24 and node-25). In such a case, the node stores both the resultant positions and when trying to join any cluster-head, it assumes the furthest possible location from the the head to calculate the distances from the cluster-head.

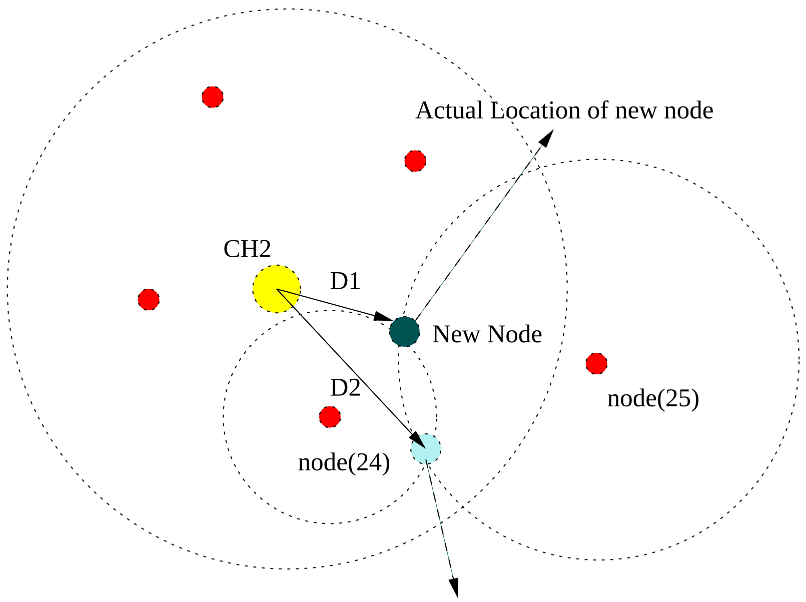


Figure 4.4: Distance between CH2 and NN = D2 $>$ D1. Distance assumed by NN to calculate confidence value between CH2 and NN = D1

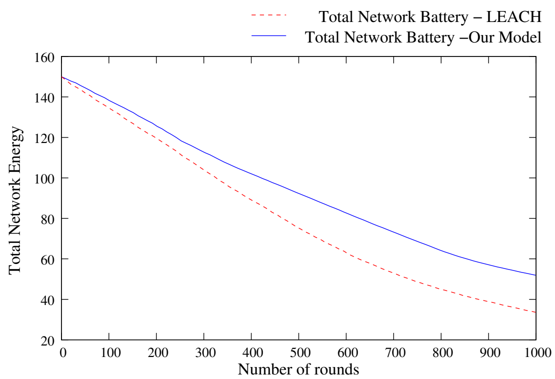
In Fig. 4.4, D2 $>$D1 when measured from CH2 to the New Node (NN). When NN sends out a broadcast requesting to join a cluster, CH2 will reply with its location information. NN will calculate its distance from CH2 based on its two set of coordinates established via ranging and assume the coordinates which gives the maximum distance from CH2 (D2 in this case) even if it is not the actual location. This conservative approach would not let a location to be publicized that might result in a cluster prone to data loss.

Chapter 5

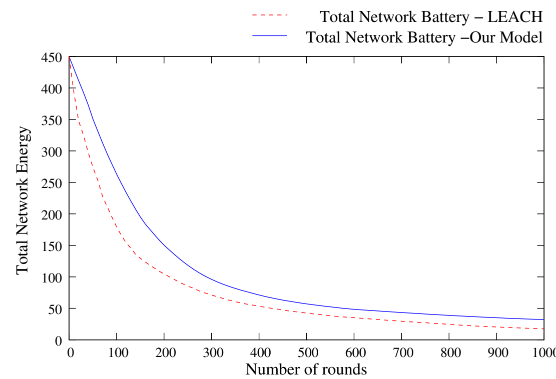
Experimental Results

Part of the work leading to this thesis has been the development of a simulator to experimentally simulate our algorithm and run tests for debugging and improvements of the model. In our thesis, we have formulated the protocols for mobility support in a sensor network but have not incorporated the mobility aspect of our model into the simulation. The simulator, specific to the needs of our model, was coded in C with GCC version 2.95.3 and uses gnuplot for its graphical needs.

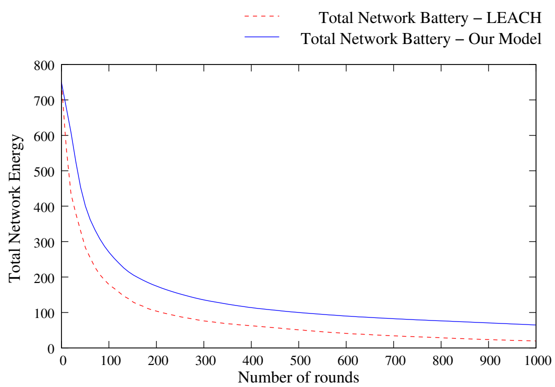
For the purpose of this experiment, the simulator simulates networks with size ranging from 50 to 350 nodes. Each simulation assumes a few default variable values that have been declared in the global header file *sensorHeaders.h* within the code-base. For networks larger than 400 nodes, our algorithm does not give significantly better results than LEACH. The surveillance area is $300 \times 300 \mathrm{m}^2$. Starting battery power (DEFAULT_B_POWER) is 3.0 units which is the same for all the nodes. The ratio of cluster-heads to the total nodes (DEFAULT_CH_RATIO) in the network is 0.05. The base station is assumed to be fixed and located at the origin (0,0) of the coordinate system. For each run of the simulation, the user can change the number of nodes in the network. The network undergoes 1000 rounds in each run.



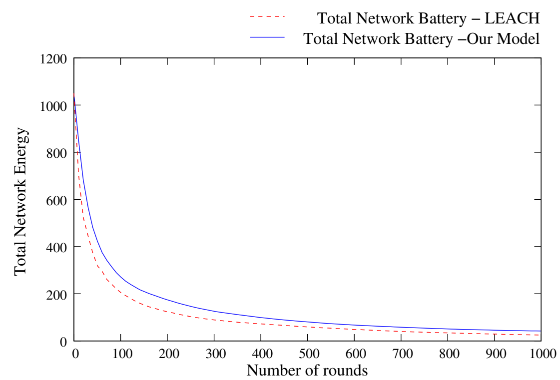
(a) N = 50



(b) N = 150



(c) N = 250



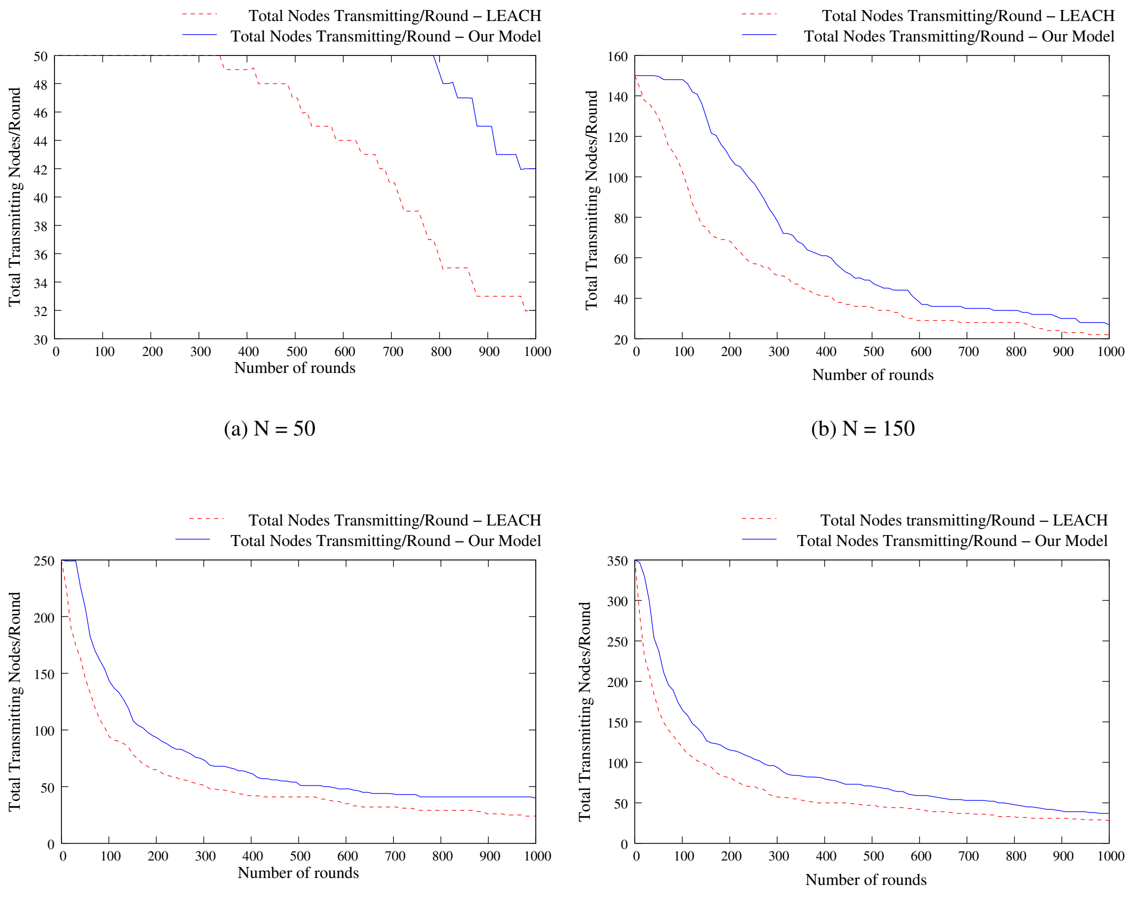
(d) N = 350

Figure 5.1: Total Network Battery Remaining after each Round. Best results are seen for small to medium sized networks

Each round is divided into two phases: (a) cluster set-up phase and (b) data transmission phase. In set-up phase, nodes decide if they have to act as cluster-heads for this round. Each cluster-head transmits a CH_ADVERTISEMENT data packet of length CH_ADVERT_BIT_LENGTH (set to 64). Also, other nodes decide which cluster to join based on the confidence values for the respective cluster-heads. The data transmission phase is made up of two phases again: (a) from the member nodes to their respective cluster-heads and (b) from the cluster-heads to the base-station. We have fixed the message bit length to 64 (SCHEDULE_MSG_BIT_LENGTH). The heads perform data aggregation to reduce the redundancy before transmitting it to the base station. The radio transmission range is set at 20 meters. TDMA is the underlying MAC protocol.

The primary metrics of interest are (i) the total network energy and (ii) total nodes still capable of transmitting at the end of each round. In Fig. 5.1, we compare the total remaining network energy after completion of a round for LEACH and our model. Four different set of readings have been taken for a network size ranging from 50 to 350 nodes. As seen in Fig. 5.1(a), (b), (c) and (d), the total remaining network energy for our model remains substantially higher than that in LEACH for all network sizes below 350. For a network size of 50, the final network energy using LEACH is 38.28 units and with our algorithm, it comes around 57.79 units, an improvement of almost 50%. Beyond network sizes of 400, our results deteriorate to the results obtained from the LEACH algorithm. This could due to the additional overhead of exchanging too many heart-beat messages within a cluster group to identify node or cluster-head mobility. Also, our results are effective for a sparse to mid-sized sensor network. For a large and widely distributed network, our model is not effective.

Figs 5.2(a), (b), (c) and (d) show the total nodes that still have battery power left in them. The simulation starts with a network of 50 nodes and steps up to a size of 350 nodes with a step-size of 50. For a low number of nodes (Fig. 5.2(a)), our model has no dead nodes till the 800^{th} round which is almost twice the number of rounds LEACH runs without a dead node. For all the simulations, dead nodes start appearing in our model much later than they show up in LEACH. For a network size exceeding 400, the performance of our model is not substantially better than LEACH.



(a) N = 50

(b) N = 150

(c) N = 250

(d) N = 350

Figure 5.2: Total Transmitting Nodes after each round. Best results are seen for small to medium sized networks

The plot presented in Fig. 5.3 summarizes the resultant saving of network energy achieved with our model. In this figure, we have compared the final remaining energy in a sensor network over a range of rounds with different samples of total network sizes (ranging from 150 to 350). The resulting over-all plot clearly shows the advantages of our model over LEACH. The starting energy for each node was equal (= 3 units) for both our model as well as LEACH.

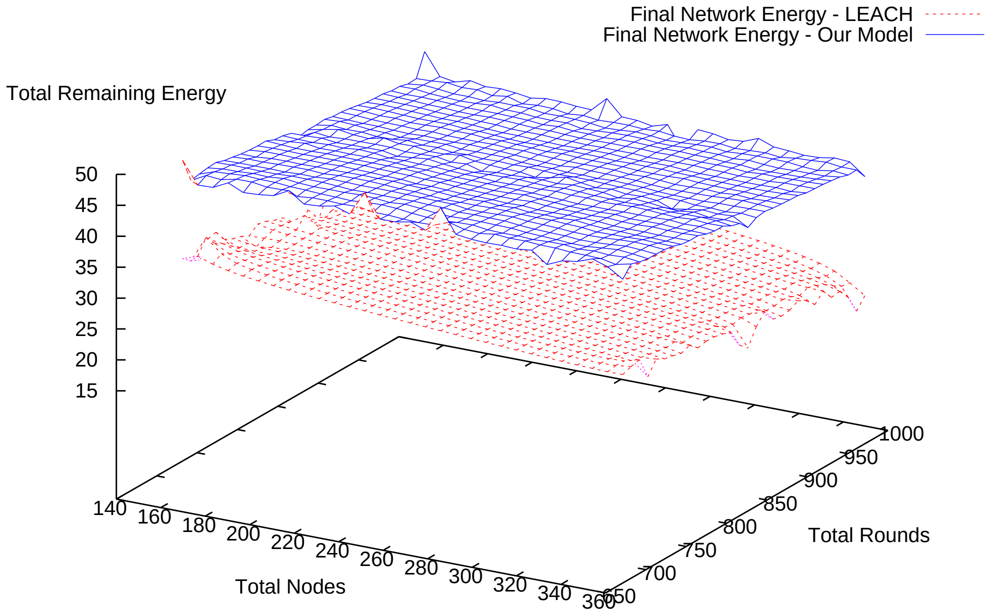


Figure 5.3: Comparative Plot of Total Post-Simulation Network Energy in the Network

Chapter 6

Localization in Sensor Networks

A very useful feature of sensor networks is that it can provide information that is highly localized in space and/or time. For many sensor network applications like target tracking and habitat monitoring, it is necessary to have the exact location from where the information was collected. Current sensor network installations are designed for a static network distribution, i.e., the nodes do not change their locations over time. In such installations, one way to know the position of the nodes is to have the network installer find these positions during deployment. Since this is not a very useful technique (most sensors are dropped from an aircraft), in this chapter we shall look at some common *self-localization* techniques employed by sensor nodes. Besides self-localization methods, *location services* can also be used to estimate the position of a node. However, in this thesis, we have not discussed these methods and concentrated only on the self-localization techniques. At the end of the chapter, we have given an outline of our modified TOA algorithm which uses Round Trip Time (RTT) of ping packets to calculate distances between two wireless devices.

6.1 Positioning through Self-Localization

Self-Localization methods aim at estimating the distance of a receiver from a transmitter by exploiting some known signal propagation characteristics. Using self-localization, a sensor node can calculate its geographic position on it own during the network initialization process. These methods are based on *ranging*, i.e., each node assumes the presence of **three** other nodes with known positions in it's vicinity. Given below is a short description of various ranging methods.

6.1.1 RSSI

Received Signal Strength Indication (RSSI) is a popular ranging technique that gives good results for dense network configurations (i.e., distance between two nodes $<$ 10 mts). The results also deteriorate if the two nodes are not in the Line of Sight (LOS) of each other. For the purpose of our discussion, we will assume a basic free-space path loss model where there are no obstacles to alter or interfere with the RF signals. The RF antenna is assumed to be isotropic, i.e. it radiates RF energy equally in all directions (Fig. 6.1).

The power of signals attenuate according to the inverse square rule:

$$P_{RF} \propto \frac{1}{d^{\alpha}} \tag{6.1}$$

Using this law, we find that the RF signal is reduced by a factor of four every time the distance is doubled. We can convert that to dB:

$$10\log(4) = 6.02dB \tag{6.2}$$

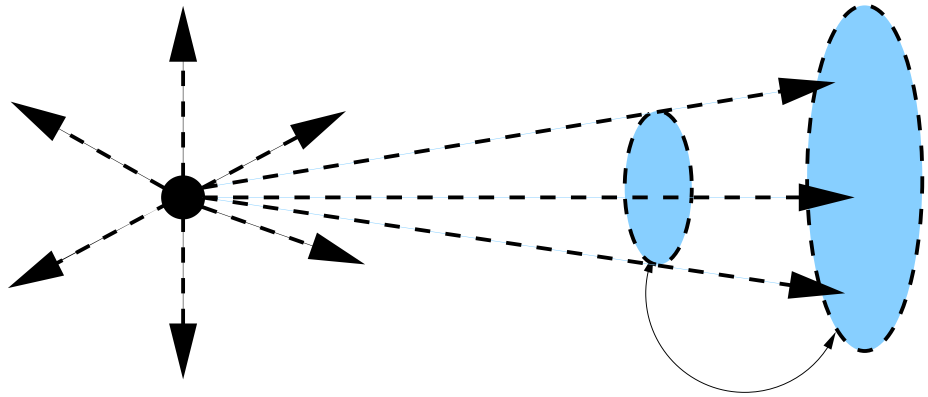


Figure 6.1: Standard Isotropic RF Propagation Model

∴ In free space, the received signal is reduced by 6dB every time we double the distance. Using the information, we can construct a free space path loss model:

$$L_P(dB) = 20\log(f) + 20\log(d) - 27.6 \tag{6.3}$$

where: d is distance between transmitting and receiving antenna (mts) and f is frequency of signals in MHz.

Building further on Eq. 6.3, we can calculate the signal power present at the receiving antenna if we know the antenna gain and the transmission power.

$$P_r = P_t - L_P + G_t + G_r \tag{6.4}$$

where P_r P_t L_P G_t and G_r have their usual meanings.

RSSI techniques use Eq. 6.4 to calculate the distance the signals have traveled to reach the point where signal strength is being measured. In a real life scenario, obstacles in air and on surface lead to multi-path fading and scattering which reduces the accuracy of RSSI ranging in an outdoor environment and also if the distances that are being measured are more than 10 meters.

6.1.2 AOA and TDOA

The Time of Arrival technique exploits triangulation to determine positions of the mobile users. Position estimation by triangulation is based on knowing the distance from the node to at least three other nodes whose positions are already known, called *landmarks*. The landmark nodes determine the time signal takes from the source to the receiver either on the uplink or on the downlink.

For example, when 9-1-1 is dialed from a mobile unit, the controlling base station prompts the mobile to respond to an initial signal. The total time elapsed from the instant the command is transmitted to the instant the mobile responds is detected. This time consists of the sum of the round trip signal delay, and any processing and response delay within the mobile unit. When the processing delay is subtracted from the total measured time, total round trip delay is found. Half of the quantity would be the estimate of the signal in one direction. Multiplying this time with the traveling velocity of the electro-magnetic waves would give the approximate distance of the mobile from the base station. The approximate distance to the mobile determined by two additional receivers could be used to determine the mobile position at the intersection of circles from multiple TOA measurements, as illustrated in Fig. 6.2. The mobile position can be determined accurately if there exists a complete LOS between the mobile station (MS) and the base stations. However the occurrence of non-line-of-sight (NLOS) propagation causes the signal to take a longer path to the base station receiver and the measured TOA is generally larger than the arrival time of an LOS signal. In such a circumstance, there is a need to detect

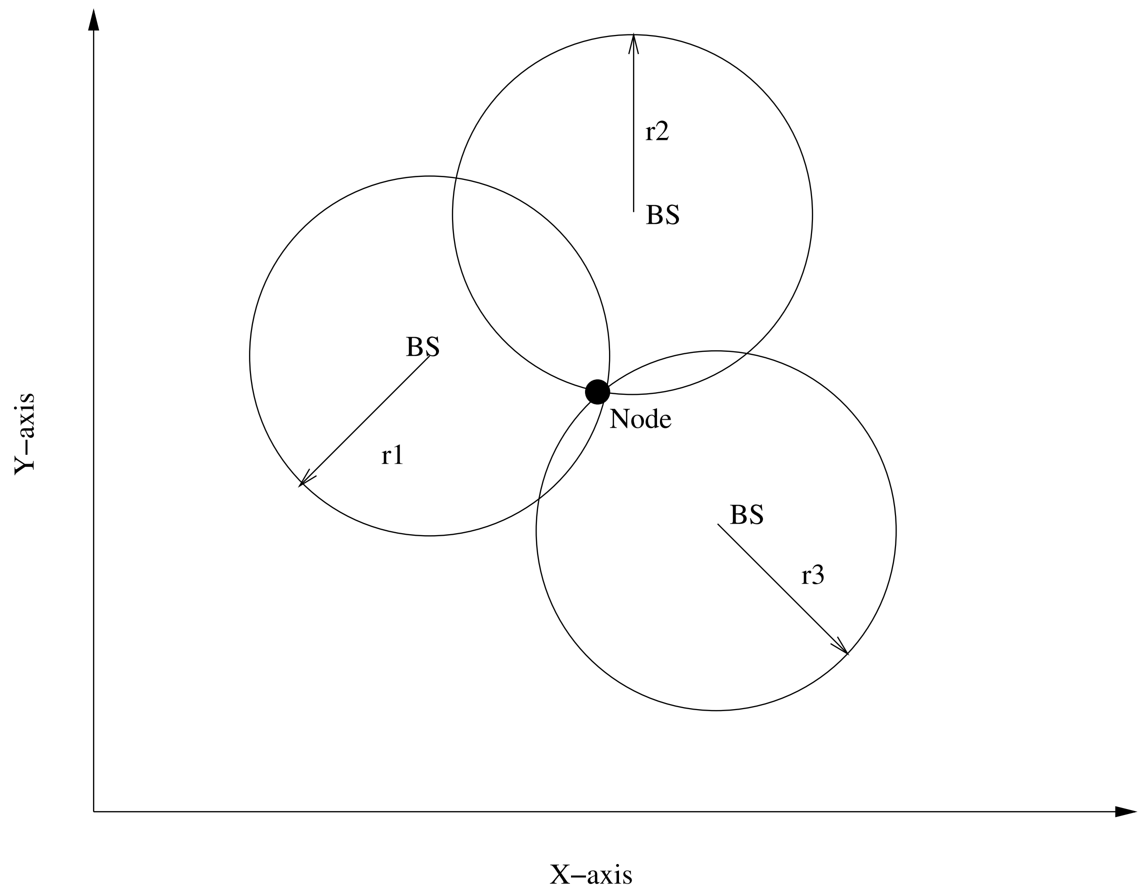


Figure 6.2: TOA location measurement from Three landmark nodes

NLOS and to correct the biased error in the TOA measurements before processing them.

Time Difference of Arrival (TDOA) is also known as the hyperbolic positioning technique. It is made up of two stages. In the first stage, time delay estimation is used to find the time difference of arrival (TDOA) of acknowledgment signals from nodes to base-stations (or landmark nodes). This TDOA estimate is used to calculate the range difference measurements between the base stations. In the second stage, an efficient algorithm is used to determine the position location estimation by solving the nonlinear hyperbolic equations resulting from the first stage.

6.2 Modified TOA - Our Localization Model

Our algorithm to find distances between two nodes is based loosely on TOA. Most classic approaches estimate the time of arrival (TOA) of pure radio packets (and not WLAN packets) for localization purposes. Signal processing algorithms based on cross-correlation techniques are used for these approaches [40]. In this thesis, our primary objective is to use WIFI based wireless devices as a proof of concept and present a few algorithms to justify the effectiveness of our method based on a TCP/IP model. We are trying to use the round-trip travel time of a ping packet in a WIFI system to estimate the distance between two WIFI devices. This section lays the theoretical groundwork on which our experiments are conducted (shown in chapter 8).

6.2.1 Packet transmission in WIFI - IEEE 802.11

In IEEE 802.11, each packet that is sent is immediately acknowledged by the receiver. In other words, there is no TCP/IP *sliding window* involved at the transmitters end. Fig. 6.3 shows a basic data packet transaction between two WIFI nodes. Each time a data packet is sent out, a timer is started at the senders end. Upon receiving the acknowledgment, we stop the timer and note the time the packet takes to travel round-trip. This delay also involves processing time at the remote node as well and must be taken into consideration while calculating the distance. For the sake of simplicity, we assume that the servers on both ends are identical and therefore, the time the local node takes to acknowledge a packet is equal to the time the remote node takes to do so. We call the total delay between sending a packet and receiving an acknowledgment as t_{total} and the delay in sending an acknowledgment at the local node as $t_{processing}$. Using these notations and assuming that the packets travel at the speed of light (c

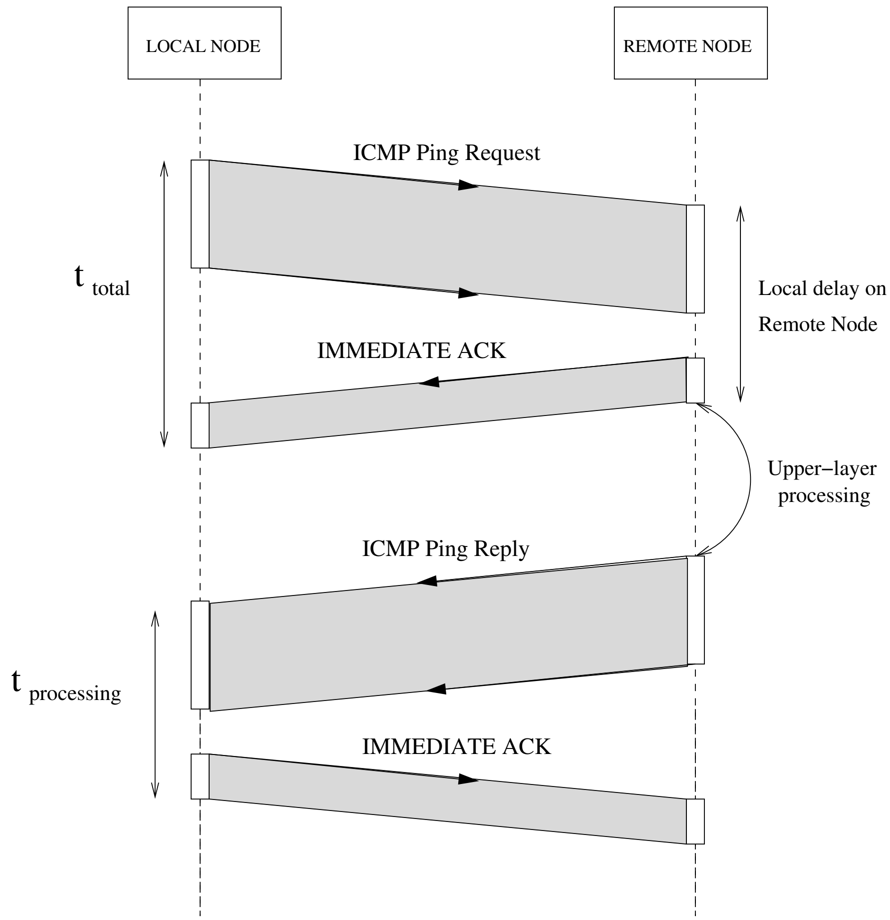


Figure 6.3: Distance Estimation using ICMP Ping request/reply method

$\approx 3 \,.\, 10^8$ m/s) we get:

$$distance = \frac{(t_{total} - t_{processing}).c}{2} \tag{6.5}$$

Since the clock resolution of most wireless cards is around 1μs but the data packets travel around 300mts in that time, we need to improve the accuracy of the measured time-delays by taking multiple readings and using statistical methods to remove errors.

6.2.2 Our Approach

The approach that we have followed is based on the ideas presented in [38] and [23]. We use *ping* packets between two wireless nodes to measure the time-delay in transmissions. We have used a modified source-code for ping where we can change the payload of a ping packet before transmitting it. After the first round of transmission, the second round (and above) encapsulates the distance calculated in the first round to facilitate a faster convergence of the algorithm. A summary of the steps involved is given below:

Round 1:

- Node-1 sends a ping packet (ping[1]) to Node-2 with an ICMP message.
- Node-2 replies back immediately with an ACK_2[1] to Node-1. Later, after upper-layers process the ping packet, Node-2 replies to Node-1 with an ICMP echo_reply (ping_reply[1]) message.
- Node-1 receives Node-2's ACK (ACK_2) and calculates t_{total}.
- Also, Node-1 receives the ICMP echo_reply (ping_reply[1]) from Node-2 and sends back an ACK (ACK_1[1]) and calculates the local delay $t_{processing}$ and therefore, the first estimate of the distance $estimate_1$[1].
- Node-2 receives ACK (ACK_1[1]) from Node-1 and also estimates the distance between itself and Node-1 ($estimate_2$[1]).

This finishes the first round of ping packet exchanges. From Round 2 onwards, the steps involved are as follows:

Round n ($n > 1$):

- Node-1 sends a ping packet (ping[n], n $>$ 2) with inter-node distance calculated in previous round (i.e. $estimate_1$[n-1] during round ping[n-1]) encapsulated within.
- Node-2 receives ping[n] and sends an immediate ACK_2[n] and extracts the estimate of nodal distance that is encapsulated within the ping packet by ping[n].
- Node-1 receives Node-2's ACK (ACK_2[n]) and calculates the total time delay t_{total}.
- After upper-layer processing, Node-2 sends an ICMP echo_reply (ping_reply[n]) message to Node-1 with the distance averaged out as $estimate_2[n] = estimate_2[n-1] + estimate_1[n-1])/2$
- Node-1 receives ping_reply[n] and sends an immediate ACK thereby calculating $t_{processing}$. Using t_{total} and $t_{processing}$, it finds a new distance estimate $(estimate_1[n] = (estimate_1[n-1] + estimate_2[n-1])/2$

The nodes continue to exchange the ping packets till we have 3 consecutive estimates of distances that do not differ from one another by more than 5%.

The algorithm described above has been represented in Fig. 6.4. Ping responses are generated by the operating systems' kernel and are subject to high variations of response-time. In contrast, the data ACKs are handled directly by the hardware of the WLAN radio and are highly predictable [23]. On a standard IEEE 802.11, the MAC processing time (SIFS interval) is 10μs (802.11b) or 16μs (802.11a) with a tolerance of upto $\pm$25 ppm and $\pm$20 ppm respectively. We use the packet sniffer *Ethereal* to sniff packets on a WLAN. Since the time-resolution of Ethereal is around 1μs which translates to around 300 m, we need to take multiple round-trip time measurements to smoothen out the discrepancies. We also show that by encapsulating distance estimations with ping packets, we get much faster convergence of the estimation algorithm. Modifications were made to the ping utility at appropriate places to incorporate this feature.

6.2.3 Limits of Our Approach

Any positioning or distance measuring algorithm suffers from some basic limits [45, 17]. For example, our calculations above do not take into account the clock drifts of the two nodes (PC, in our case) during one round-trip time observation. Assuming a tolerance value of ± 25 ppm and the transmission time of 60μs to 320μs, we may have an error in calculation ranging from 0.9 m to 4.8 m respectively.

Also, the speed of light is $3x10^8$ in *vacuum*. In air, this value decreases due to the dielectric constant (ε) of air. The two nodes may not be in the LOS (line of sight) of each other leading to heavy multi-path fading effects. However, multi-path propagation does not effect time-delay measurements as much as it effects signal strength. Therefore, our time-delay model is more robust and precise than RSSI model for distance measurement.

In the next chapter, we have described the experiments conducted based upon the algorithms presented in this chapter. The results show that our approach is better than standard TOA and RSSI approaches for short range distances.

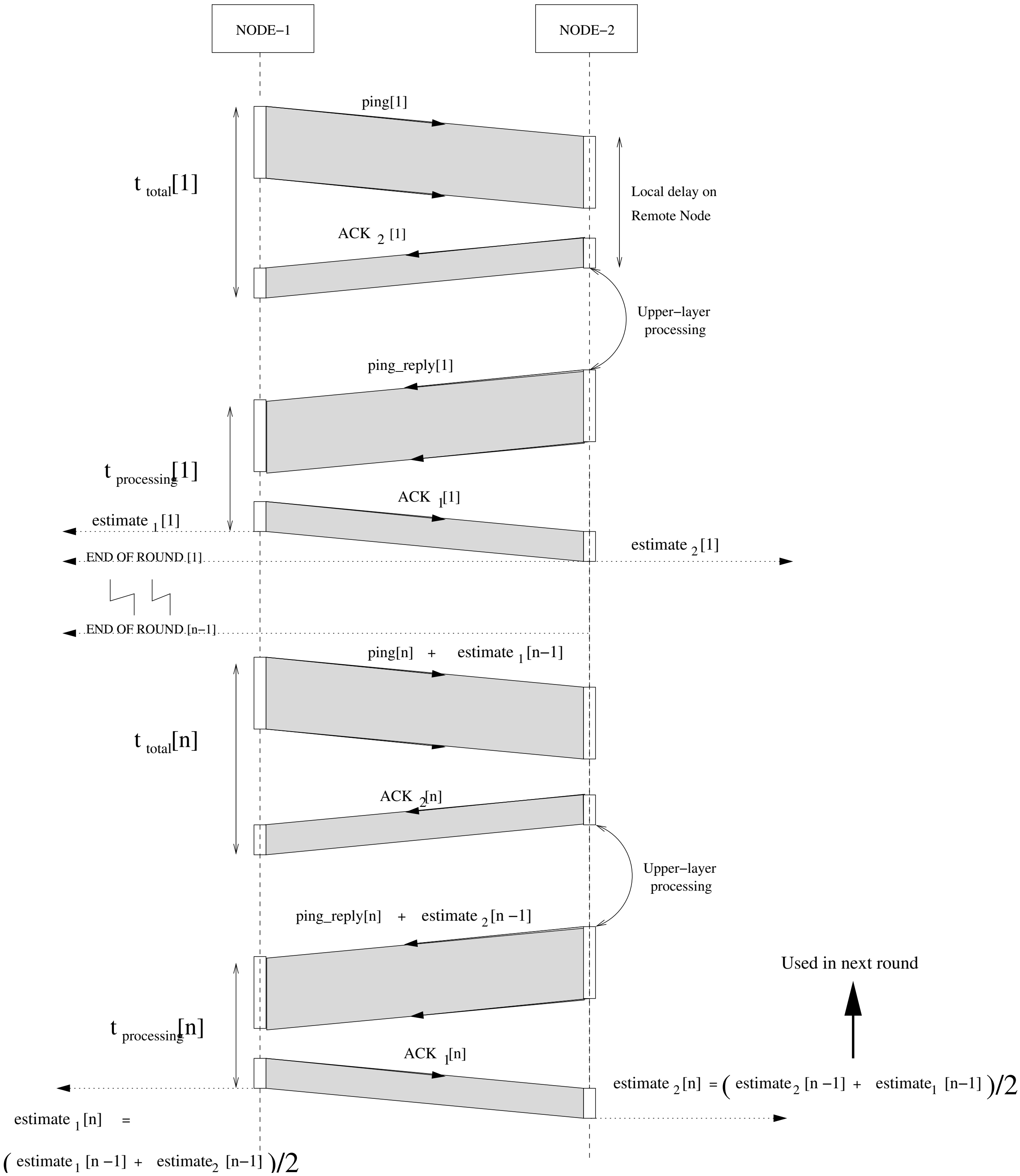


Figure 6.4: Our algorithm (using encapsulated ping) for inter-nodal distance estimation

Chapter 7

Experimental Setup and Results

The experiment was setup in the open area in front of Zimmer Hall and Langsam Library. The choice of area is important so as to have both the nodes in the line of sight (LOS) of each other. Two sets of experiments were conducted. First, using normal ping packets and later, using ping packets with the distance estimations encapsulated in the payload. Round trip times (RTT) measurements were taken at different distances (5, 10, 15, 20) and at each distance, ping trace data was collected for 10 minutes using Ethereal's WLAN packet sniffer module.

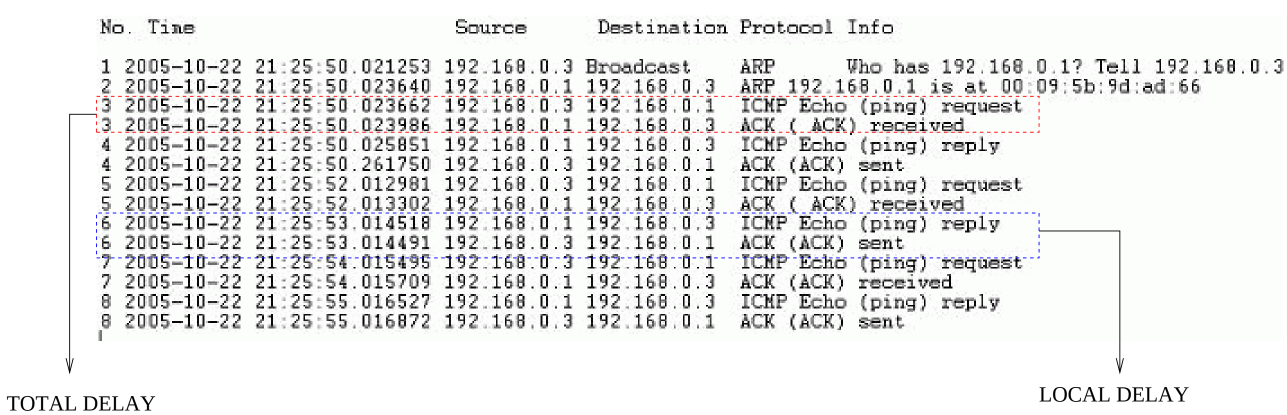
Both laptops used had identical installations of Suse-9.3 Linux, kernel 2.6.10. Also, both the laptops had a Broadcom 802.11b/g WLAN adapter installed. The speed of transmission was set at 11Mbits/s.

Both data and the ACK packets begin with a preamble followed by a Physical Layer Convergence Procedure (PLCP) header which contains the length and modulation type of the packet. For a 11Mbits/s 802.11 network, the time taken to transmit the preamble and the headers is

144+48 μs [23]. After the PLCP header, the actual MAC headers are transmitted in a MAC frame that consists of MAC headers (24 b), IP headers (20 b), UDP headers (8 b), RTP (16 b), Voice (20 b) and the frame-check sequence (4 b). Overall the MAC frame has a length of 92 b which takes 66.91μs to transmit. Therefore, the overall length of the ping data packet is around 258.91μs. The ACK frames are shorter and have a length of 10 bytes plus the CRC (4 b). They are transmitted over a 2 Mbits/s link and take 248μs. In between the data packet and the ACK, there is a short inter-frame space (SIFS), which is of the length of 10μs. In [23], the authors have proven that for a system configuration such as above, the local delay is about 66.91+10+192+56=329.91μs

7.1 Data acquisition and Analysis

We have made use of the Ethereal packet sniffer for packet analysis. The local delay is assumed to be 329.00μs. Packet filters were used for the data capture to filter our all unwanted packets on the network and to sniff only ICMP packets. Fig. 7.1 shows a sample packet capture when the two nodes are 10 meters apart.



No.	Time	Source	Destination	Protocol	Info
1	2005-10-22 21:25:50.021253	192.168.0.3	Broadcast	ARP	Who has 192.168.0.1? Tell 192.168.0.3
2	2005-10-22 21:25:50.023640	192.168.0.1	192.168.0.3	ARP	192.168.0.1 is at 00:09:5b:9d:ad:66
3	2005-10-22 21:25:50.023662	192.168.0.3	192.168.0.1	ICMP	Echo (ping) request
3	2005-10-22 21:25:50.023986	192.168.0.1	192.168.0.3	ACK	(ACK) received
4	2005-10-22 21:25:50.025851	192.168.0.1	192.168.0.3	ICMP	Echo (ping) reply
4	2005-10-22 21:25:50.261750	192.168.0.3	192.168.0.1	ACK	(ACK) sent
5	2005-10-22 21:25:52.012981	192.168.0.3	192.168.0.1	ICMP	Echo (ping) request
5	2005-10-22 21:25:52.013302	192.168.0.1	192.168.0.3	ACK	(ACK) received
6	2005-10-22 21:25:53.014518	192.168.0.1	192.168.0.3	ICMP	Echo (ping) reply
6	2005-10-22 21:25:53.014491	192.168.0.3	192.168.0.1	ACK	(ACK) sent
7	2005-10-22 21:25:54.015495	192.168.0.3	192.168.0.1	ICMP	Echo (ping) request
7	2005-10-22 21:25:54.015709	192.168.0.1	192.168.0.3	ACK	(ACK) received
8	2005-10-22 21:25:55.016527	192.168.0.1	192.168.0.3	ICMP	Echo (ping) reply
8	2005-10-22 21:25:55.016872	192.168.0.3	192.168.0.1	ACK	(ACK) sent

Figure 7.1: Ethereal Packet capture data. Distance = 10 m

Since we are assuming that data in a WLAN travels at the speed of light, we are interested in only those entries where the difference between t_{total} and $t_{processing}$ is less than 1μs since light travels around 300 mts in 1μs and we are trying to measure distances smaller than 20 mts.

7.2 Results and Analysis

The distance between the nodes was calculated using Eq. 6.3 as derived in chapter 6. Two sets of experiments were conducted. In the first experiment, normal ping messages were sent. In the second set of experiments, ping packets also contained the result of the estimation of distance from the previous round. For both the experiments, we stop sending ping packets as soon as the three simultaneous calculated distance estimations differ from each other by less than 5%. Fig. 7.2 shows a plot of the number of ping packets exchanged to reach a stable distance estimate for both sets of experiments.

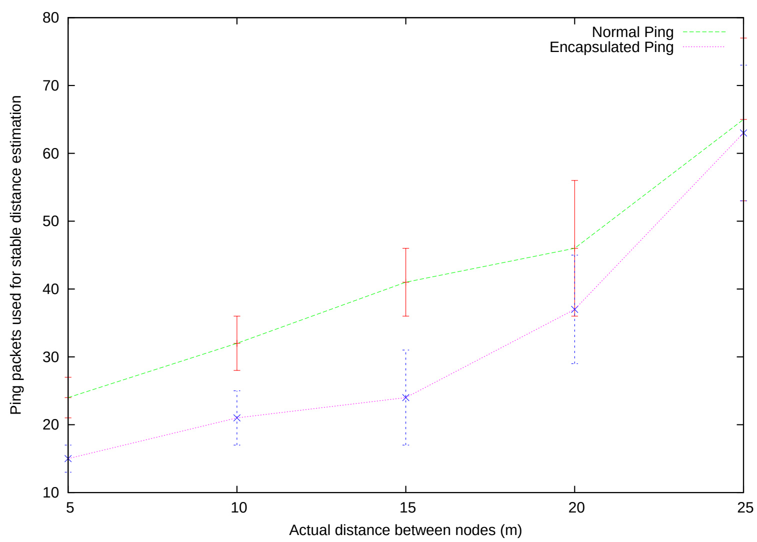


Figure 7.2: Number of ping packets needed to reach a stable distance estimate

As we can see from Fig. 7.2, our modified TOA model gives better convergence of distance estimation algorithm over short ranges (0-20 mts) as compared to normal TOA. Also, since lesser number of ping packets are needed to reach a mutual consensus over inter-nodal distance, it leads to much less wastage of bandwidth. In case of wireless sensors, this directly translates into a substantial conservation of battery power that would, otherwise, have been wastefully spent in calculating node location. For larger distances between the nodes (distance $>$ 20 m.), both approaches (normal and encapsulated) are more or less equally inefficient and are not good choices.

Fig. 7.3 shows the accuracy of distance estimation using ping packets in our modified TOA methods. With the estimated distance sent as a payload variable in ping packets resulted in not only faster but also better and more accurate distance measurements as compared to the normal ping technique.

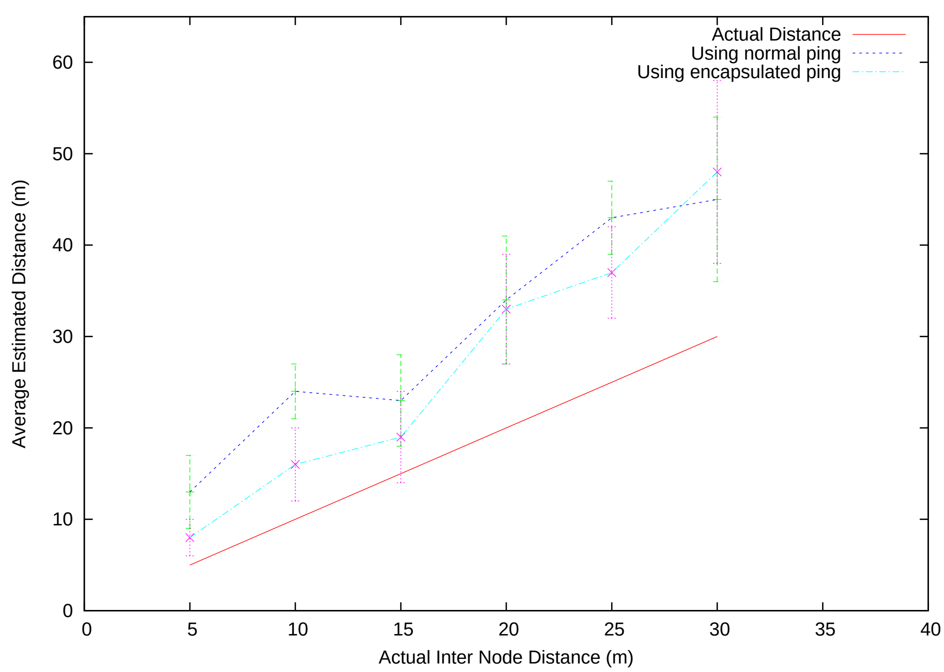


Figure 7.3: Distance calculated using RTT using Normal and Encapsulated ping packets vs. Actual Distances

We have used a set of 20 rounds to estimate inter-nodal distances for the range 0-25 m. Fig. 7.3

plots the average results of estimation at 5, 10, 15, 20 and 25 meters and it also plots the error-bars which show the deviation from the mean distance averaged over all the 20 rounds. As we can see, we have good results from our algorithm for short range distances (0-20 m.) but the estimations deteriorate at longer ranges and at distances above 25 meters, both the algorithms give highly inaccurate estimations and should not be the first choice of any locationing system.

Although all the experiments were done in an open area during night-time to avoid any disturbances from people walking around the campus, some inaccuracies in the results can be attributed to various heavily active wireless nodes that are always active in and around the campus area. Also, the area in front of Zimmer and Langsam Library is surrounded by tall buildings from all sides and it could also have affected the accuracy of our experiments. It should be noted that the data presented in this thesis is highly subject to change based on where the experiments are conducted and also many other environmental conditions. The primary objective of our experiments was to establish with a fair amount of accuracy, the significant improvements our algorithm brings to the field of location estimation using TOA methods with Round Trip Time (RTT) measurements on a WLAN. Actual implementations using a wireless sensor node would have to be much more robust and accurate.

Chapter 8

Conclusion and Future Work

In this thesis, we presented algorithmic improvements on two major sensor network areas: (a) Clustering and (b) Localization. In the first part of our work, we formulated an energy efficient sensor network clustering algorithm based on LEACH and also designed a fault tolerant architecture to deal with mobility of cluster-heads and member nodes. Work done also included implementation of a simulator to simulate our model and a comparative performance analysis with LEACH. Our clustering model has the following advantages over LEACH:

- Our model can accommodate mobility of cluster-heads as well as node using periodic heartbeat messages exchanged between the cluster head and it's member nodes.
- We use the remaining battery power and the current member nodes of a cluster-head as a parameter used by the non cluster-head nodes to decide which cluster-head to join. This approach makes sure that cluster-heads that are in a particularly dense region of the network are not overloaded with too many member nodes under them.
- Unlike LEACH, our design does not assume that all sensor nodes have the same initial

battery power. Since in most real-life scenarios, the battery power of every node in a sensor network may not be same, our model comes one step closer to duplicating a real-world situation for experimentation.

Results of our experiments show that for a network size ranging from 50 to 350 nodes, our model results in higher average network energy as against LEACH. This directly translates to lesser number of dead nodes and thus, better overall network connectivity. We believe that our algorithm can offer effective improvements on the performance, energy-efficiency and robustness of long running sensor networks. Furthermore, our design for a mobility aware sensor system has numerous improvements over existing research and adds a layer of fault tolerance to the overall design.

The second part of our thesis deals with finding an efficient algorithm to measure the distance between two sensor nodes. We used two WLAN devices (PC with Broadcom 802.11b/g cards running on Linux) and measured their mutual distance using the Round Trip Time (RTT) of ping packets sent back and forth between them. Instead of using normal ping packets, we changed the source-code for ping (freely available on the net) and encapsulated the estimated distance for each previous round as a part of the payload for the current round. This approach leads to a much faster convergence of the localization algorithm therefore leading to a lot lesser wastage of battery power as compared to the method where ordinary ping packets were being sent.

In our experiments, we have compared the the estimated distances as well as the number of packets needed to arrive at those estimations using both normal and encapsulated ping packets. As shown in the previous chapter, encapsulated ping packets lead to 25-30% more accurate distance estimation for short range distances (distance $<$ 20 m). Also, since our algorithm converges much faster, a lot fewer number of ping packets (atleast 40% less) are necessary to arrive at a stable estimation thereby saving precious battery power.

8.1 Future Work

We believe that our Clustering and Localization algorithms can improve the performance and lifetime of existing sensor network infrastructures. The algorithms discussed are targeted specially for dense to mid-sized networks where the inter-nodal distance is not more than 20-25 m. For future work, the following areas should be a good place to start:

- The simulator for our clustering model does not currently incorporate mobility of nodes or cluster-heads. A future version of the simulator should include this feature of our algorithm.
- To establish the usefulness of our clustering model, we would have to implement it on sensor network hardware (Mica Motes, for e.g.) and test it on the field to understand the real-world issues faced by our model.
- The Localization algorithm uses RTT of ping packets over WLAN devices which are essentially TCP/IP compliant nodes. Sensor network nodes do not follow TCP/IP networking. Future implementations should port our algorithm on a sensor network hardware (Mica Motes, for e.g.) so that we can start to improve our model to cater to a real-world scenario.
- All RTT calculations in our model assume that the clocks on either of the nodes do not suffer a *clock drift*. Every hardware clock loses time over a period and needs to re-synchronize itself with a reference clock. This drift can be different in each of the nodes and must be incorporated into future implementations of our localization algorithm.

Bibliography

[1] Bushfire response using sensor networks. http://www.sensornetworks.net.au/applic_bushfire.html.

[2] Cdma. http://www.umtsworld.com/technology/cdmabasics.htm.

[3] Ieee 802.11. http://grouper.ieee.org/groups/802/11/.

[4] Mica motes. http://www.spp.co.jp/xbow/motes.jpg.

[5] Mica motes configuration. http://www.tinyos.net/scoop/special/hardware/mica2.

[6] Smart dust. http://robotics.eecs.berkeley.edu/ pister/SmartDust.

[7] R. Szewczyk D. Culler Alan Mainwaring, J. Polastre and John Anderson. Wireless sensor networks for habitat monitoring. In *Proceedings of the ACM International Workshop on Wireless Sensor Networks and Applications (WSNA.02)*, September 2002.

[8] S. Bandyopadhyay and E. J. Coyle. An energy efficient hierarchical clustering algorithm for wireless sensor networks. In *Proceedings of INFOCOM 2003*, 2003.

[9] Stefano Basagni. Distributed clustering for ad hoc networks. In *ISPAN '99: Proceedings of the 1999 International Symposium on Parallel Architectures, Algorithms and Networks (ISPAN '99)*, page 310, Washington, DC, USA, 1999. IEEE Computer Society.

[10] D. Braginsky and D. Estrin. Rumor routing algorithm for sensor networks. In *International Conference on Distributed Computing Systems (ICDCS-22)*, 2002.

[11] N. Bulusu, J. Heidemann, and D. Estrin. Adaptive beacon placement. In *21st International Conference on Distributed Computing Systems (ICDCS-21)*, pages 489–498.

[12] N. Bulusu, J. Heidemann, and D. Estrin. Gps-less low cost outdoor localization for very small devices. Technical report.

[13] Ram Ramanathan C. A. Santivanez and Ioannis Stavrakakis. Making link-state routing scale for ad hoc networks. In *MobiHoc '01: Proceedings of the 2nd ACM international symposium on Mobile ad hoc networking and computing*, pages 22–32, New York, NY, USA, 2001. ACM Press.

[14] Srdan Capkun, Maher Hamdi, and Jean-Pierre Hubaux. Gps-free positioning in mobile ad-hoc networks. In *HICSS*, 2001.

[15] G. Pottie M. Srivastava D. Estrin, L. Girod. Instrumenting the world with wireless sensor networks. In *International Conference on Acoustics, Speech, and Signal Processing (ICASSP 2001)*, 2001.

[16] Y. Hu D. Li, K.Wong and A. Sayeed. Detection, classification, tracking of targets in micro-sensor networks. In *IEEE Signal Processing Magazine*, pages 17–29, March 2002.

[17] Eiman Elnahraway, Xiaoyan Li, and Richard P. Martin. The limits of localization using rss. In *SenSys '04: Proceedings of the 2nd international conference on Embedded networked sensor systems*, pages 283–284, New York, NY, USA, 2004. ACM Press.

[18] Deborah Estrin, Ramesh Govindan, John S. Heidemann, and Satish Kumar. Next century challenges: Scalable coordination in sensor networks. In *Mobile Computing and Networking*, pages 263–270, 1999.

[19] Nathan Ickes Rex Min Amit Sinha Eugene Shih, Seong-Hwan Cho. Physical layer driven protocol and algorithm design for energy-efficient wireless sensor networks. In *ACM SIGMOBILE*, July 2001.

[20] Jie Gao, Leonidas Guibas, John Hershberger, Li Zhang, and An Zhu. Discrete mobile centers. In *SCG '01: Proceedings of the seventeenth annual symposium on Computational geometry*, pages 188–196, New York, NY, USA, 2001. ACM Press.

[21] Mario Gerla and Jack Tzu-Chieh Tsai. Multicluster, mobile, multimedia radio network. *Wirel. Netw.*, 1(3):255–265, 1995.

[22] Ramesh Govindan. Data-centric routing and storage in sensor networks. pages 185–205, 2004.

[23] Andre Gunther and Christian Hoene. Measuring round trip times to determine the distance between wlan node. In *NETWORKING 2005*, pages 768–779, 2005.

[24] G. Gupta and M. Younis. Load-balanced clustering of wireless sensor networks. In *Proceedings of IEEE International Conference on Communications (ICC'03)*, volume 3, pages 1848–1852, May 2003.

[25] Paul J.M. Havinga and Gerard J.M. Smit. Design techniques for low-power systems. *J. Syst. Archit.*, 46(1):1–21, 2000.

[26] John Heidemann, Fabio Silva, Chalermek Intanagonwiwat, Ramesh Govindan, Deborah Estrin, and Deepak Ganesan. Building efficient wireless sensor networks with low-level naming. In *SOSP '01: Proceedings of the eighteenth ACM symposium on Operating systems principles*, pages 146–159, New York, NY, USA, 2001. ACM Press.

[27] W. Heinzelman, J. Kulik, and H. Balakrishnan. Adaptive protocols for information dissemination in wireless sensor networks. In *Proceedings of the fifth annual ACM/IEEE International Conference on Mobile computing and networking*, 1999.

[28] Wendi Rabiner Heinzelman, Anantha Chandrakasan, and Hari Balakrishnan. Energy-efficient communication protocol for wireless microsensor networks. In *HICSS*, 2000.

[29] Y. Sankara I. F. Akyildiz, W. Su and E. Cayirci. A survey on sensor networks. *IEEE Communications Magazine*, August 2002.

[30] Chalermek Intanagonwiwat, Deborah Estrin, Ramesh Govindan, and John Heidemann. Impact of network density on data aggregation in wireless sensor networks. In *ICDCS '02: Proceedings of the 22 nd International Conference on Distributed Computing Systems (ICDCS'02)*, page 457, Washington, DC, USA, 2002. IEEE Computer Society.

[31] Chalermek Intanagonwiwat, Ramesh Govindan, and Deborah Estrin. Directed diffusion: a scalable and robust communication paradigm for sensor networks. In *Mobile Computing and Networking*, pages 56–67, 2000.

[32] David B Johnson and David A Maltz. Dynamic source routing in ad hoc wireless networks. In Imielinski and Korth, editors, *Mobile Computing*, volume 353. Kluwer Academic Publishers, 1996.

[33] V. Ailawadhi K. Sohrabi, J. Gao and G. J. Pottie. Protocols for self-organization of a wireless sensor network. volume 7, pages 16–27, May 2000.

[34] M.A. Youssef K.A. Arisha and M.F. Younis. Energy-aware tdma based mac for sensor networks. May 2002.

[35] Brad Karp and H. T. Kung. GPSR: greedy perimeter stateless routing for wireless networks. In *Mobile Computing and Networking*, pages 243–254, 2000.

[36] KSJ Pister L. Doherty and L. Ghaoui. Convex position estimation in wireless sensor networks. In *In Proceedings of IEEE INFOCOM-01*, 2001.

[37] Woo Hyong Lee, J. Morris Chang, and Yusuf Hasan. Dynamic memory measuring tool for C++ programs. In *Proceedings of The Third IEEE Symposium on Application-Specific Systems and Software Engineering Technology (ASSET 2000)*, Richardson, TX, 2000.

[38] Joel Lepak and M. Crescimanno. Speed of light measurement using ping. Technical Report YSU-CPIP/102-02, phy, January 2002.

[39] Li Li, Fu Lizheng, Luan Guixing, and Wang Ping. A hierarchical architecture for distributed network management. In *International Conference on Internet Computing (1)*, pages 420–423, 2001.

[40] Xinrong Li, Kaveh Pahlavan, and Jacques Beneat. Performance of toa estimation techniques in indoor multipath channels. *IEEE PIMRC*, September 2002.

[41] R. Moses, D. Krishnamurthy, and R. Patterson. A self-localization method for wireless sensor networks. *EURASIP Journal on Applied Signal Processing*, 2002.

[42] K. Chintalapudi D. Ganesan A. Broad R. Govindan N. Xu, S. Rangwala and D. Estrin. A wireless sensor network for structural monitoring. In *Proceedings of the 2nd ACM SenSys*, November 2004.

[43] D. Niculescu and B. Nath. Ad hoc positioning system (aps). In *In Proceedings of GLOBECOM, 2001*, 2001.

[44] D. Niculescu and B. Nath. Ad hoc positioning system (aps) using aoa. 2003.

[45] Drago Niculescu and Badri Nath. Error characteristics of ad hoc positioning systems (aps). In *MobiHoc '04: Proceedings of the 5th ACM international symposium on Mobile ad hoc networking and computing*, pages 20–30, New York, NY, USA, 2004. ACM Press.

[46] L. Nirupama, B. Deborah, and E. Deborah. Scalable coordination for wireless sensor networks: Self-conguring localization systems. In *International Symposium on Communication Theory and Applications (ISCTA)*, July 2001.

[47] Vincent D. Park and M. Scott Corson. A highly adaptive distributed routing algorithm for mobile wireless networks. In *INFOCOM (3)*, pages 1405–1413, 1997.

[48] Neal Patwari and III Alfred O. Hero. Using proximity and quantized rss for sensor localization in wireless networks. In *WSNA '03: Proceedings of the 2nd ACM international conference on Wireless sensor networks and applications*, pages 20–29, New York, NY, USA, 2003. ACM Press.

[49] G. Pei, M. Gerla, and T. Chen. Fisheye state routing: A routing scheme for ad hoc wireless networks. In *Proceedings of the IEEE International Conference on Communications*, pages 70–74, June 2000.

[50] C. Perkins. Ad-hoc on-demand distance vector routing. November 1997.

[51] G. J. Pottie and W. J. Kaiser. Wireless integrated network sensors. *Commun. ACM*, 43(5):51–58, 2000.

[52] T. He R. Stoleru and J. A. Stankovic. Walking gps:a practical solution for localization in manually deployed wireless sensor networks. In *1st IEEE Workshop on Embedded Networked Sensors (EmNetS-I)*, May 2004.

[53] Raphael Rom and Moshe Sidi. Multiple access protocols: Performance and analysis (book review). *SIGMETRICS Performance Evaluation Review*, 20(3):5–6, 1993.

[54] Abu Ghazaleh Sameer Tilak, B. Nael and Wendi Heinzelman. A taxonomy of wireless micro-sensor network models. In *ACM Mobile Computing and Communications Review (MC2R)*, volume 6, pages 43–56, April 2002.

[55] C. Savarese, J.M. Rabaey, and J. Beutel. Locationing in distributed ad hoc wireless sensor networks. In *Proc. 2001 Int'l Conf. Acoustics, Speech, and Signal Processing (ICASSP 2001)*, volume 4, pages 2037–2040. IEEE, Piscataway, NJ, May 2001.

[56] A. Savvides, H. Park, and M. Srivastava. The bits and flops of the n-hop multilateration primitive for node localization problems. In *In First ACM International Workshop on Wireless Sensor Networks and Application*, September 2002.

[57] Andreas Savvides, Chih-Chieh Han, and Mani B. Strivastava. Dynamic fine-grained localization in ad-hoc networks of sensors. In *Mobile Computing and Networking*, pages 166–179, 2001.

[58] Chien-Chung Shen, Chavalit Srisathapornphat, and Chaiporn Jaikaeo. Sensor Information Networking Architecture and Applications. *IEEE Personel Communication Magazine*, 8(4):52–59, August 2001.

[59] H. O. Tan and Ibrahim K. Power efficient data gathering and aggregation in wireless sensor networks. *SIGMOD Rec.*, 32(4):66–71, 2003.

[60] M. Tubaishat and S. Madria. Sensor networks : An overview. Technical report, April 2003.

[61] Ning Xu. A survey of sensor network applications. http://enl.usc.edu/ ningxu/papers/survey.pdf.

[62] I.F. Akyildiz Y. Sankarasubramaniam and S.W. McLaughlin. Energy efficiency based packet size optimization in wireless sensor networks. In *In Proceedings of 1st IEEE Intl. Workshop on Sensor Network Protocols and Applications (SNPA)*, May 2003.

[63] Wei Ye and John Heidemann. Medium access control in wireless sensor networks. pages 73–91, 2004.

[64] Y. Yu, R. Govindan, and D. Estrin. Geographical and energy aware routing: A recursive data dissemination protocol for wireless sensor networks. 2001.

[65] Wing Ho Yuen and Chi Wan Sung. On energy efficiency and network connectivity of mobile ad hoc networks. In *ICDCS '03: Proceedings of the 23rd International Conference on Distributed Computing Systems*, page 38, Washington, DC, USA, 2003. IEEE Computer Society.

[66] Feng Zhao and Leonidas Guibas. *Wireless sensor networks - an information processing approach*. Morgan Kaufmann Publishers.