Cluster based Routing Protocol Optimization for Wireless Sensor Networks

By

Panchal Shailesh D.

08MCE008

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

AHMEDABAD-382481

May, 2010

Cluster based Routing Protocol Optimization for Wireless Sensor Networks

Major Project

Submitted in partial fulfillment of the requirements

For the degree of

Master of Technology in Computer Science and Engineering

By

Panchal Shailesh D.

08MCE008

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

AHMEDABAD-382481

May, 2010

Declaration

This is to certify that

i) The thesis comprises my original work towards the degree of Master of Technology in Computer Science and Engineering at Nirma University and has not been submitted elsewhere for a degree.

ii) Due acknowledgement has been made in the text to all other material used.

Panchal Shailesh D.

Certificate

This is to certify that the Major Project entitled "Cluster based Routing Protocol Optimization for Wireless Sensor Networks" submitted by Panchal Shailesh D. (08MCE008), towards the partial fulfillment of the requirements for the degree of Master of Technology in Computer Science and Engineering of Nirma University of Science and Technology, Ahmedabad is the record of work carried out by him under my supervision and guidance. In my opinion, the submitted work has reached a level required for being accepted for examination. The results embodied in this major project, to the best of my knowledge, haven't been submitted to any other university or institution for award of any degree or diploma.

Prof. Gaurang Raval
Guide, Asso. Professor,
Department Computer Engineering,
Institute of Technology,
Nirma University, Ahmedabad.

Dr.S.N.Pradhan
Professor and PG-Coordinator,
Department of Computer Engineering,
Institute of Technology,
Nirma University, Ahmedabad.

Prof.D.J.Patel
Professor and Head,
Department Computer Engineering,
Institute of Technology,
Nirma University, Ahmedabad.

Dr.K.Kotecha
Director,
Institute of Technology,
Nirma University, Ahmedabad.

Abstract

In the wireless sensor networks (WSNs), the sensor nodes (called motes) are usually scattered in a sensor field - an area in which the sensor nodes are deployed. These motes are small in size and have limited processing power, memory and battery life. The motes in these networks are coordinate to produce high-quality information and each of these scattered motes has the capabilities to collect and route data back to the base stations, which are fixed or mobile. In WSNs, conservation of energy, which is directly related to network lifetime, is considered relatively more important so use of energy efficient routing algorithms is one of the ways to reduce the energy conservation. In general, routing algorithms in WSNs can be divided into flat, hierarchical and location-based routing. In flat, all nodes are assigned equal roles. In hierarchical, however, nodes will play different roles in the network. While in location-based routing, sensor nodes are addressed by means of their locations.

The main objective of this major project is to optimize the Low-Energy Adaptive Clustering Hierarchy (LEACH) protocol. There are two reasons behind the hierarchical routing being explored. One, the sensor networks are dense and a lot of redundancy is involved in communication. Second, in order to increase the scalability of the sensor network keeping in mind the security aspects of communication. Cluster based routing holds great promise for many-to-one and one-to-many communication paradigms that are prevalent in sensor networks.

In this major project, the work has been carried out is implementation of flat based Flooding and Gossiping routing protocols using TinyOS Simulator (TOSSIM), implementation of LEACH routing protocol using NS-2 simulator and finally some modifications in LEACH routing protocol in order to improve the performance of the LEACH protocol. The results were then analyzed based on the suggested evaluation metrics in order to verify their suitability for use in wireless sensor networks.

Acknowledgements

It gives me great pleasure in expressing thanks and profound gratitude to **Prof. Gaurang Raval**, Associate Professor, Department of Computer Science and Engineering, Institute of Technology, Nirma University, Ahmedabad for his valuable guidance and continual encouragement throughout the project work. I heartily thankful to him for his time to time suggestions and the clarity of the concepts of the topic that helped me a lot during this study.

I would like to extend my gratitude to **Dr.S.N.Pradhan,** Professor & M.Tech Coordinator, Department of Computer Science and Engineering, Institute of Technology, Nirma University, Ahmedabad for fruitful discussions and valuable suggestions during meetings and for their encouragement.

I would like to thanks **Dr.Ketan Kotecha,** Hon'ble Director,Institute of Technology, Nirma University, Ahmedabad for providing basic infrastructure and healthy research environment.

I would also like to extend my thanks to Prof. R.J.Joshi, Principal, GEC Modasa, Prof. M.B.Chaudhary, Head of CE/IT department, GEC Modasa, Prof. M.R.Patel, Principal, VGEC Chandkheda, Mr. Mehul Parikh, Lecturer IT department, VGEC Chandkheda, Directorate of Technical Education, Gandhinagar and all staff members of GEC Modasa and VGEC Chandkheda for their kind cooperation and motivations.

Last, but not the least, no words are enough to acknowledge constant support and sacrifices of my family members, especially my wife, because of whom I am able to complete my dissertation work successfully.

- Panchal Shailesh D.

08MCE008

Contents

List of Tables

List of Figures

Chapter 1

Introduction

It is 2010, and a lot work has been done in the area of Wireless Sensor Network, but still a long way to go. Wireless Sensor networks consist of hundreds of thousands of low power multi-functional sensor nodes, operating in an unattended environment, with limited computation and sensing capabilities. Sensor nodes are equipped with small, often irreplaceable batteries with limited power capacities. The use of wireless sensor networks is increasing day by day and at the same time it faces the problem of energy constraints in terms of limited battery lifetime. Various approaches can be taken to save energy caused by communication in wireless sensor networks. One of them is to adopting energy efficient routing algorithms. The routing algorithms in the sensor networks broadly classified into three category: Flat,Hierarchical(or Cluster) and Location based routing. The cluster based routing holds great promise for many-to-one and one-to-many communication paradigms that are prevalent in sensor networks. This dissertation work includes the survey of various cluster based routing protocols and implementation of LEACH routing protocol. The idea proposed in LEACH has been an inspiration for many hierarchical routing protocols. The Finally it propose some modifications to improve the performance of the LEACH routing protocol. The simulation results were then analyzed based on the suggested evaluation metrics in order to verify their suitability for use in wireless sensor networks.

1.1 Objective of the Work

The main objective of the work is to extend LEACH routing algorithm to improve its energy efficiency.

Specific Objectives

- To study hierarchical routing protocols, and analyze LEACH in particular.
- To implement LEACH protocol and analyze its characteristics.
- To propose improvements on LEACH protocol.
- To test and validate the effectiveness of proposed improvement

1.2 Scope of the Work

The scope of this work is to optimize a energy efficient routing protocol(LEACH) for WSNs that can be implemented on existing WSNs nodes. Even though there are many protocols, there is still scope for optimization of this protocol for energy conservation and one that can be implemented on networks of any sizes.

1.3 Motivation of the Work

The motivation behind doing this research is the need to develop a more efficient scheme for routing which utilizes less power as compared to the present schemes. Wireless Sensor Networks is an upcoming branch of Networks. The problem addressed by this network is due to the lack of energy efficient routing protocol.

1.4 Thesis Organization

The rest of the thesis is organized as follows.

Chapter 2, *Literature Survey*, describes general overview of wireless sensor network technology, Design issues, Challenges of routing protocols and its applications in the various fields, Survey of hierarchical based routing algorithms utilize by Wireless Sensor Networks and their approaches towards energy efficiency and reliability.

Chapter 3, *Study of WSN Simulators: TOSSIM and NS-2*, describes basics of TinyOS operating system and characteristics required for the simulator to work with TinyOS environment. TinyOS simulator, TOSSIM [17] is presented which is specifically designed for TinyOS. It also describe and TOSSIM graphical user interface, TinyViz and the extension of TOSSIM simulator PowerTOSSIM[18], which is for simulate the power profile of the sensor nodes. It also covers the mit's extension of NS-2 simulator, for implementation of LEACH routing protocol.

In **chapter 4**, *The Implementation of Flat based Routing protocols using TOSSIM*, includes implementation of existing multi-hop flat based routing approaches (Flooding and Gossiping) using TOSSIM simulator and analysis of results in terms of four evaluation metrics: Latency, Power usage, Overhead and scalability.

In **chapter 5**, *Simulation of LEACH routing protocol using NS-2*, will includes the basic LEACH simulation using NS-2 simulator and analysis on the simulation results by varying the cluster numbers.

In **chapter 6**, *Proposed modification in LEACH protocol.*, will includes the some proposed modification on basic LEACH protocol in order to achieve the main objective of the project work. It will also cover the comparison analysis of the results and improvements of the sensor network performance using modified protocol.

Finally, in **chapter 7** concluding remarks and future work is presented.

Chapter 2

Literature Survey

Wireless sensor networks (WSNs) contain hundreds or thousands of sensor nodes equipped with sensing, computing and communication abilities. Each node has the ability to sense elements of its environment, perform simple computations, and communicate among its peers or directly to an external base station (BS) Deployment of a sensor network can be in random fashion or planted manually. These networks promise a maintenance-free, fault-tolerant platform for gathering different kinds of data. Because a sensor node needs to operate for a long time on a tiny battery, innovative techniques to eliminate energy inefficiencies that would shorten the lifetime of the network must be used. A greater number of sensors allows for sensing over larger geographical regions with greater accuracy. Figure 2.1 shows the basic components of the sensor nodes and wireless sensor network architecture [1].
This chapter will cover the applications of WSNs, basic routing challenges and brief of several hierarchical routing protocols used in the wireless sensor networks. In Summary, various limitations and assumptions of the protocols will cover.

2.1 Applications of wireless sensor networks

Data sensing and reporting in sensor networks is dependent on the application and time criticality of the data reporting. As a result, sensor networks can be categorized

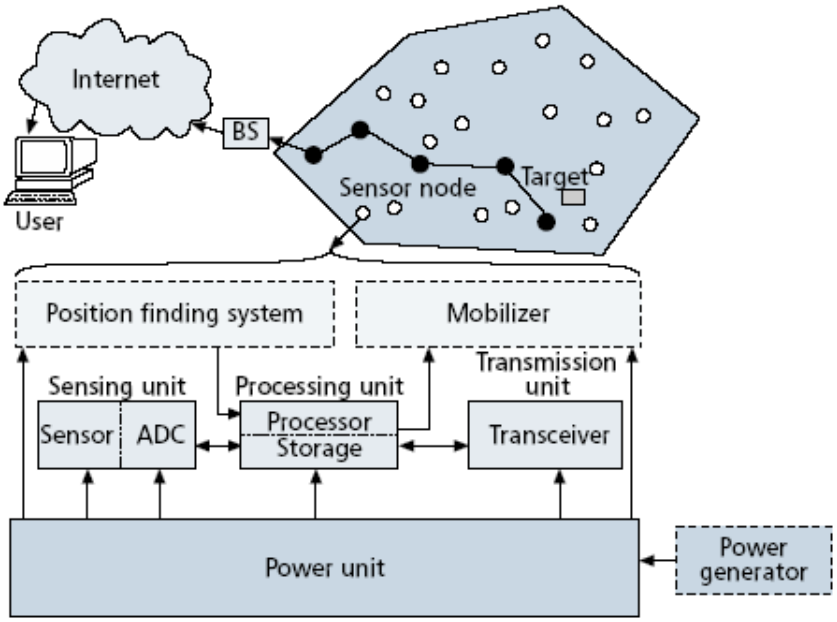


Figure 2.1: Components of sensor node and WSN architecture

as time-driven or event-driven networks. The former type is suitable for applications that require periodic data monitoring. As such, sensor nodes will periodically switch on their sensors and transmitters, sense the environment, and transmit data of interest at constant periodic time intervals. Thus, they provide a snapshot of the relevant attributes at regular intervals. In the latter type, sensor nodes react immediately to sudden and drastic changes in the value of a sensed attribute due to the occurrence of a certain event. These are well suited for time critical applications. Some application examples of WSNs include[1]:

- Target field imaging.
- Intrusion detection.
- Weather monitoring.
- Security and tactical surveillance.
- Distributed computing.

- Detecting ambient conditions such as temperature, movement, sound, light, or presence of certain objects.

2.2 Routing challenges in wireless sensor networks

The design of routing protocols in WSNs is influenced by many challenging factors[1]. Which includes:

- *Ad hoc deployment:* Sensor nodes are deployed randomly. This requires that the system be able to cope with the resultant distribution and form connections between the nodes.

- *Energy consumption without losing accuracy:* Due to limited supply of energy performing computations and transmitting information in a wireless environment with reliability.

- *Computation capabilities:* Due to limited computing power of nodes and therefore may not be able to run sophisticated network protocols.

- *Communication range:* Intersensor communication exhibits short transmission ranges. Therefore,it is most likely that a route will generally consist of multiple wireless hops.

- *Scalability:* The number of sensor nodes deployed in the sensing field may be in the order of hundreds or thousands or more.

- *Hardware constraints:* Consisting of many hardware components, a sensor node may be smaller than a cubic centimeter. These components consume extremely low power and operate in an unattended mode; nonetheless, they should adapt to the environment of the sensor network and function correctly.

- *Control overhead:* When the number of retransmissions in a wireless medium increases due to collisions, latency and energy consumption will also increase.

Therefore, control packet overhead increases linearly with node density.

- *Quality of service:* In some applications, the data should be delivered within a certain period of time from the moment they are sensed; otherwise the data will be useless. Therefore, bounded latency for data delivery is another condition for time-constrained applications.

2.3 Hierarchical routing protocols

Basically recent routing protocols for sensor networks are classified based on the various approaches it pursued. The three main categories explored in [2][3] are data-centric(Flat), hierarchical and location-based. Furthermore, these protocols can be classified into multipath-based, query-based, negotiation-based, QoS-based, and coherent based depending on the protocol operation. However, I have also observed that there are some hybrid protocols that fit under more than one category. D. J. Dechene et.al [4]examine several clustering routing algorithms for Wireless Sensor Networks classified them into Heuristic Algorithms, Weighted Schemes, Hierarchical Schemes and Grid Schemes based on approaches the algorithms. Specifically, they examine the performance these algorithms in terms of the power and quality aspects of these schemes.G.Santhosh Kumar et.al [5] discussed four routing protocols for wireless sensor networks viz. Flooding, Gossiping, Gradient Based Routing (GBR) and LEACH and they have been simulated using TinyOS. Their power consumption is studied using PowerTOSSIM. A realization of these protocols has been carried out using Mica2 motes. The simulation result were evaluted. The dynamic cluster based protocol LEACH stands out among the other three for best power utilization.

Low-Energy Adaptive Clustering Hierarchy (LEACH)

This protocol is proposed by W. R. Heinzelman et.al [6] which minimizes energy dissipation in sensor networks, It is based on a simple clustering mechanism by which

energy can be conserved since cluster heads are selected for data transmission instead of other nodes. The operation of LEACH is broken up into rounds, where each round begins with a set-up phase, when the clusters are organized, followed by a steady-state phase, when data transfers to the base station occur. In order to minimize overhead, the steady-state phase is long compared to the set-up phase.

Set-up phase: During this phase, each node decides whether or not to become a cluster head (CH) for the current round. This decision is based on choosing a random number between 0 and 1. if number is less then a threshold *T(n)*, the node become a cluster head for the current round. The threshold value is set as:

$$T(n) = \begin{cases} \frac{P}{1-P*(rmod\frac{1}{P})} & \text{if } n \in G \\ 0 & \text{otherwise} \end{cases}$$

where, P = desired percentage of cluster head, r = current round and G is the set of nodes which did not become cluster head in last $\frac{1}{P}$ rounds. Once the cluster head is chosen, it will use the CSMA MAC protocol to advertise its status. Remaining nodes will take the decision about their cluster head for current round based on the received signal strength of the advertisement message. Before steady-state phase starts, certain parameters are considered, such as the network topology and the relative costs of computation versus the communication. A Time Division Multiple Access(TDMA) schedule is applied to all the members of the cluster group to send messages to the CH, and then to the cluster head towards the base station. As soon as a cluster head is selected for a region, steady-state phase starts. Figure 2.2 shows the flowchart of the this phase.

Steady-state phase: Once the clusters are created and the TDMA schedule is fixed, data transmission can begin. Assuming nodes always have data to send, they send it during their allocated transmission time to the cluster head. This transmission uses a minimal amount of energy (chosen based on the received strength of the cluster-head advertisement). The radio of each non-cluster-head node can be turned off until the nodes allocated transmission time, thus minimizing energy dissipation in these nodes.

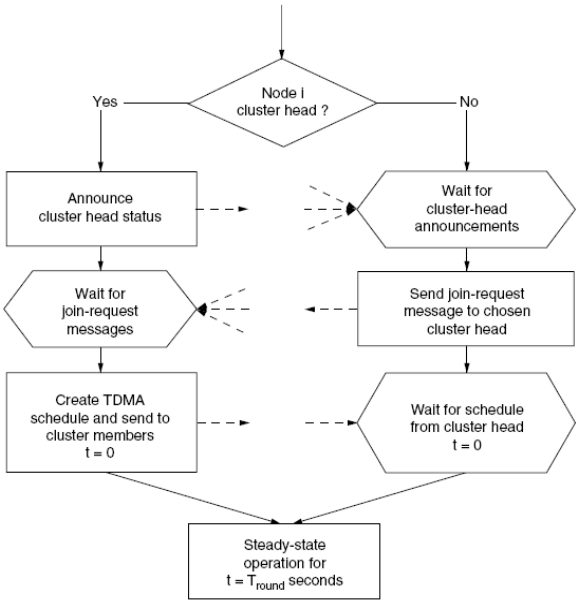


Figure 2.2: Flow chart of the Set-up phase of the LEACH protocol

The cluster-head node must keep its receiver on to receive all the data from the nodes in the cluster. When all the data has been received, the cluster head node performs signal processing functions to generate the composite single signal. For example, if the data are audio or seismic signals, the cluster-head node can beam form the individual signals to generate a composite signal. This composite signal is sent to the base station. Since the base station is far away, this is a high-energy transmission. Figure

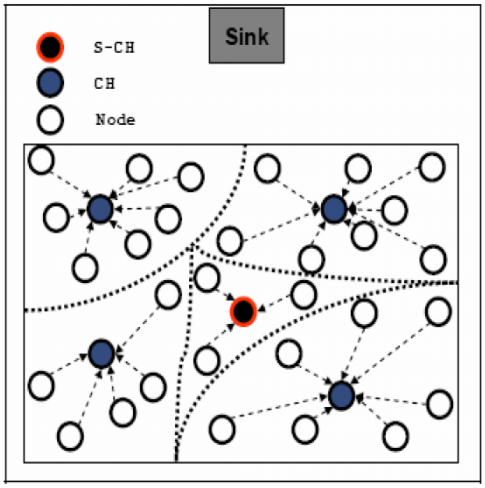
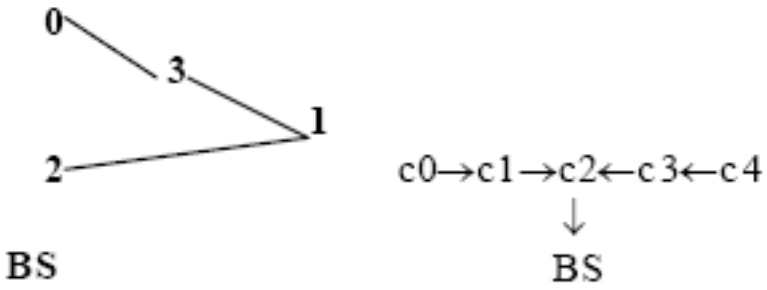


Figure 2.3: Data gathering in LEACH protocol.

2.3 shows the data gathering strategy used by the LEACH protocol. Code Division Multiple Access(CDMA) is utilized between clusters to eliminate the interference from neighboring clusters.

Power-Efficient Gathering in Sensor Information Systems (PEGASIS)

Lindsey et al [7] proposed this protocol, which is an enhancement over the LEACH protocol. The main idea in PEGASIS is for each node to receive from and transmit to close neighbors and take turns being the leader for transmission to the BS. This approach distributes the energy load evenly among the sensor nodes in the network. Sensor nodes are randomly deployed in the sensor field, and therefore, the i^{th} node is at a random location. The nodes will be organized to form a chain, which can either be accomplished by the sensor nodes themselves using a greedy algorithm starting from some node. Alternatively, the BS can compute this chain and broadcast it to all the sensor nodes. For constructing the chain, it is assumed that all nodes have global knowledge of the network and employ the greedy algorithm. The greedy approach to constructing the chain works well and this is done before the first round of communication. To construct the chain, it starts with the furthest node from the BS. To begin with this node in order to make sure that nodes farther from the BS have close neighbors, as in the greedy algorithm the neighbor distances will increase gradually since nodes already on the chain cannot be revisited. Figure 2.4a shows node 0 connecting to node 3, node 3 connecting to node 1, and node 1 connecting to node 2 in that order. When a node dies, the chain is reconstructed in the same manner to bypass the dead node. The leader in each round of communication will be at a random position on the chain, which is important for nodes to die at random locations. In a given round, simple control token passing approach is initiated by the leader to start the data transmission from the ends of the chain. The cost is very less since the token size is very small. In Figure 2.4b, node c2 is the leader, and it will pass the token along the chain to node c0. Node c0 will pass its data towards node c2. After node c2 receives data from node c1, it will pass the token to node c4, and



(a) Chain formation. (b) Token Passing.

Figure 2.4: PEGASIS Protocol operations.

node c4 will pass its data towards node c2.

Threshold sensitive Energy Efficient sensor Network protocol (TEEN)

Manjeshwar et. al.[8] classifies sensor networks in proactive and reactive networks. The nodes in the network periodically switch on their sensors and transmitters, sense the environment and transmit the data of interest. Thus, they provide a snapshot of the relevant parameters at regular intervals called the proactive networks. They are well suited for applications requiring periodic data monitoring. While Networks in which the nodes react immediately to sudden and drastic changes in the value of a sensed attribute, is called reactive networks. As such, they are well suited for time critical applications. In this scheme, approach is same as the LEACH but at every cluster change time, in addition to the attributes, the cluster-head (CH) broadcasts to its members,

Hard Threshold (H_T): This is a threshold value for the sensed attribute. It is the absolute value of the attribute beyond which, the node sensing this value must switch on its transmitter and report to its cluster head.

Soft Threshold (S_T): This is a small change in the value of the sensed attribute which triggers the node to switch on its transmitter and transmit. The nodes sense their environment continuously. The first time a parameter from the attribute set reaches its hard threshold value, the node switches on its transmitter and sends the

sensed data. The sensed value is stored in an internal variable in the node, called the sensed value (SV). The nodes will next transmit data in the current cluster period, only when both the following conditions are true:

a. The current value of the sensed attribute is greater than the hard threshold.

b. The current value of the sensed attribute differs from SV by an amount equal to or greater than the soft threshold.

Whenever a node transmits data, SV is set equal to the current value of the sensed attribute. Thus, the hard threshold tries to reduce the number of transmissions by allowing the nodes to transmit only when the sensed attribute is in the range of interest. The soft threshold further reduces the number of transmissions by eliminating all the transmissions which might have otherwise occurred when there is little or no change in the sensed attribute once the hard threshold. Figure 2.5 shows the operation of the TEEN protocol.

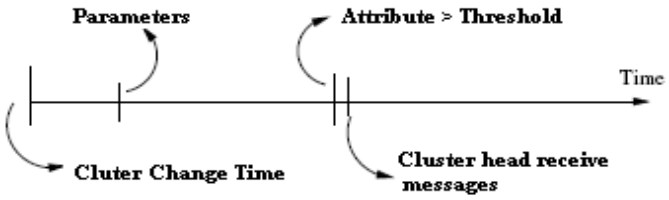


Figure 2.5: Operation of TEEN protocol.

Adaptive Periodic Threshold-sensitive Energy Efficient sensor Network (APTEEN)

Manjeshwar et. al.[9] have proposed APTEEN, which allows for comprehensive information retrieval. The nodes in such a network not only react to time-critical situations, but also give an overall picture of the network at periodic intervals in a very energy efficient manner. Such a network enables the user to request past, present and future data from the network in the form of historical, one-time and persistent

queries respectively. In APTEEN once the cluster head are decided, in each cluster period, the CHs first broadcasts the following parameters:

Attributes(A): This is a set of physical parameters which the user is interested in obtaining data about.

Thresholds: This parameter consists of a hard threshold (H_T) and a soft threshold (S_T) which are similar to used in [8]. Thresholds are used to minimize the number of transmission in order to save the energy.

Schedule: This is a TDMA schedule similar to the one used in [6], assigning a slot to each node.

Count Time(T_C): It is the maximum time period between two successive reports sent by a node. It can be a multiple of the TDMA schedule length and it accounts for the proactive component

In a sensor network, close-by nodes fall in the same cluster, sense similar data and try to send their data simultaneously, causing possible collisions. A TDMA schedule is declared such that each node in the cluster is assigned a transmission slot to avoid collisions, as shown in Figure 2.6.

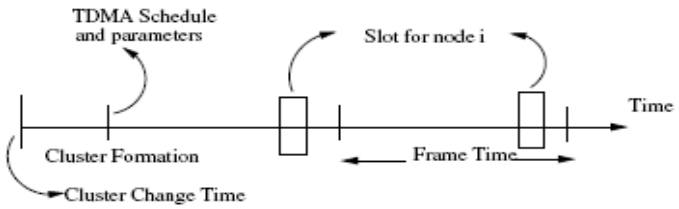


Figure 2.6: Operation of APTEEN protocol.

Hybrid Energy Efficient Distributed clustering (HEED)

Younis, O. et. al[10] proposed Hybrid Energy-Efficient Distributed Clustering (HEED) which is a multi-hop clustering algorithm with focus on efficient clustering by proper selection of cluster heads based on the physical distance between nodes. The most important aspect of HEED is the method of cluster head selection. Cluster heads are determined based on two important parameters:

a. The residual energy of each node is used to probabilistically choose the initial set of cluster heads. This parameter is commonly used in many other clustering schemes

b. Intra-Cluster Communication Cost is used by nodes to determine the cluster to join. This is especially useful if a given node falls within the range of more than one cluster head.

In HEED it is important to identify what the range of a node is in terms of its power levels as a given node will have multiple discrete transmission power levels. The power level used by a node for intra-cluster announcements and during clustering is referred to as cluster power level. Low cluster power levels promote an increase in spatial reuse while high cluster power levels are required for inter cluster communication as they span two or more cluster areas. Therefore, when choosing a cluster, a node will communicate with the cluster head that yields the lowest intra-cluster communication cost. The intra-cluster communication cost is measured using the Average Minimum Reachability Power (AMRP) measurement. The AMRP is the average of all minimum power levels required for each node within a cluster range R to communicate effectively with the cluster head i. The AMRP of a node i then becomes a measure of the expected intra-cluster communication energy if this node is elevated to cluster head. Utilizing AMRP as a second parameter in cluster head selection is more efficient then a node selecting the nearest cluster head.

A Grid-Clustering Routing Protocol (GROUP)

L. Yu et al [11] proposed GROUP, a grid-clustering routing protocol that provides scalable and efficient packet routing for large scale wireless sensor networks. In this algorithm one of the sinks (called the primary sink(PS)), dynamically, and randomly builds the cluster grid. The cluster heads are arranged in a grid-like manner as in Figure 2.7. Forwarding of data queries from the sink to source node are propagated from the Grid Seed (GS) to its cluster heads, and so on. The GS is a node within

a given radius from the primary sink. In terms of cluster head selection, on a given

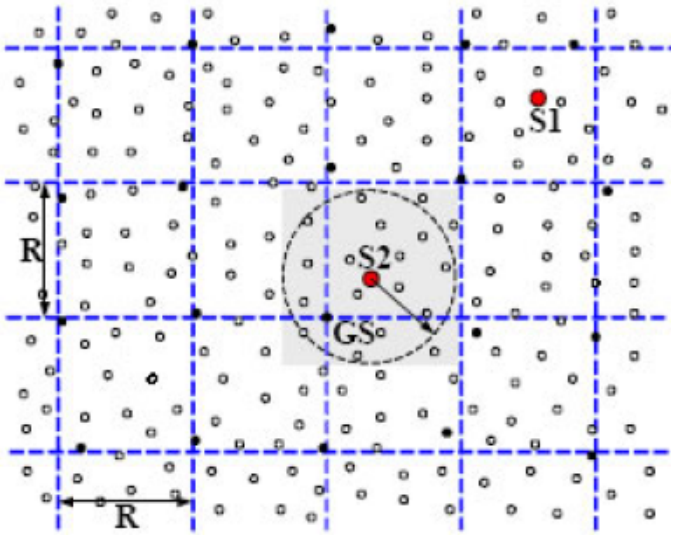


Figure 2.7: Grid example of GROUP.

round the primary sink selects a GS based on residual energy. Once the GS has been selected, the GS selects cluster heads along the corners of the grid at a range R. Each new cluster head will then select more cluster heads along the grid until all cluster heads have been selected. These selections are based on the residual energy of nodes near the corners of the grid. Data transmission in GROUP is dependant on the type of data being collected. In the case of a location unaware data query (data that is not dependant on the location of the sensing node), the query is passed from the central most sink in the network to its nearest cluster head. That cluster head will then broadcast the message to neighboring cluster heads. If the data is location aware, then the requests are sent down the chain of cluster heads towards the specified region using unicast packets. For both data queries, data is transmitted upstream through the chain of cluster heads established during cluster formation. Energy conservation is achieved due to the lower transmission distance for upstream data. Figure 2.8 illustrates the data forwarding during cluster grid construction. The network is transferring to a new cluster grid whose grid seed is GS_1. When the data packet from node N_0 reaches node N_2 via node N_1 according to the previous cluster

grid, node N_2 will transmit it to node N_3 instead of node GS_0. Because node N_3 informed its neighbor that it became a cluster head of the new cluster grid during the cluster grid construction. Then the data packet will transmitted to sink S via GS_1. So cluster grid construction doesn't impact the data forwarding.

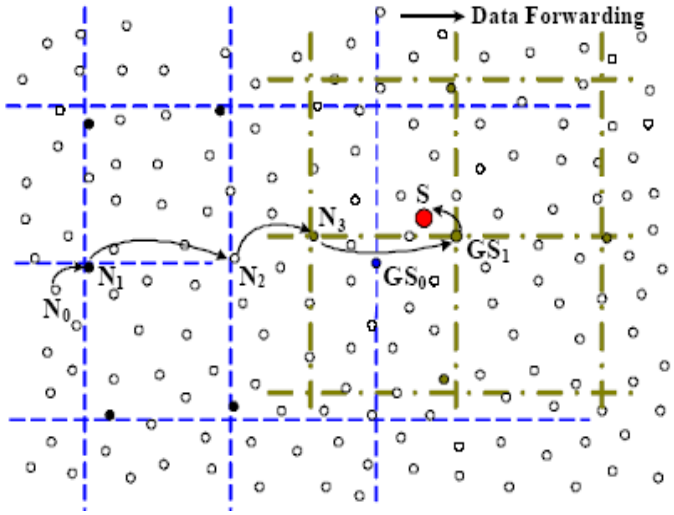


Figure 2.8: Data forwarding during grid construction in GROUP protocol.

E-LEACH Protocol

Energy-LEACH protocol improves the CH selection procedure. It makes residual energy of node as the main metric which decides whether the nodes turn into CH or not after the first round[12]. Same as LEACH protocol, E-LEACH is divided into rounds, in the first round, every node has the same probability to turn into CH, that mean nodes are randomly selected as CHs, in the next rounds, the residual energy of each node is different after one round communication and taken into account for the selection of the CHs. That mean nodes have more energy will become a CHs rather than nodes with less energy.

TL-LEACH Protocol

In LEACH protocol, the CH collects and aggregates data from sensors in its own cluster and passes the information to the BS directly. CH might be located far away from the BS, so it uses most of its energy for transmitting and because it is always on it will die faster than other nodes. A new version of LEACH called Two-level Leach was proposed. In this protocol; CH collects data from other cluster members as original LEACH, but rather than transfer data to the BS directly, it uses one of the CHs that lies between the CH and the BS as a relay station [13].

M-LEACH protocol

In LEACH, Each CH directly communicates with BS no matter the distance between CH and BS. It will consume lot of its energy if the distance is far. On the other hand, Multihop-LEACH protocol selects optimal path between the CH and the BS through other CHs and use these CHs as a relay station to transmit data over through them [14]. First, multi-hop communication is adopted among CHs. Then, according to the selected optimal path, these CHs transmit data to the corresponding CH which is nearest to BS. Finally, this CH sends data to BS. M-LEACH protocol is almost the same as LEACH protocol, only makes communication mode from single hop to multi-hop between CHs and BS.

LEACH-C protocol

LEACH offers no guarantee about the placement and/or number of cluster heads. In [15], an enhancement over the LEACH protocol was proposed. The protocol, called LEACH-C, uses a centralized clustering algorithm and the same steady-state phase as LEACH. LEACH-C protocol can produce better performance by dispersing the cluster heads throughout the network. During the set-up phase of LEACH-C, each node sends information about its current location (possibly determined using GPS) and residual energy level to the sink. In addition to determining good clusters, the

sink needs to ensure that the energy load is evenly distributed among all the nodes. To do this, sink computes the average node energy, and determines which nodes have energy below this average.
Once the cluster heads and associated clusters are found, the sink broadcasts a message that obtains the cluster head ID for each node. If a cluster head ID matches its own ID, the node is a cluster head; otherwise the node determines its TDMA slot for data transmission and goes sleep until its time to transmit data. The steady-state phase of LEACH-C is identical to that of the LEACH protocol.

VLEACH protocol

In VLEACH [16] protocol, the cluster contains vice-CH, the node that will become a CH of the cluster in case of CH dies. In the original leach, the CH is always on receiving data from cluster members, aggregate these data and then send it to the BS that might be located far away from it. The CH dies earlier than the other nodes in the cluster because of its operation of receiving, sending and overhearing. When the CH dies, the cluster becomes useless because the data gathered by cluster nodes will never reach the base station. In VLEACH protocol, besides having a CH in the cluster, there is a vice-CH that takes the role of the CH when the CH dies. Because of this, cluster nodes data will always reach the BS; no need of electing a new CH each time the CH dies. This extends the overall network life time.

2.4 Summary

In this chapter, examined the current state of clustering protocols. Despite the significant overall energy savings approaches, however, the various assumptions made by some protocols raise a number of issues. Like LEACH assumes that all nodes begin with the same amount of energy, which is however not realistic. It also assumes that all nearby nodes have correlated data which is not always true. PEGASIS assumes that the radio channel is symmetric so that the energy required to transmit a mes-

sage from node i to node j is the same as energy required to transmit a message from node j to node i for a given signal to noise ratio (SNR). Protocols presented in this chapter offer a promising improvement over conventional clustering; however there is still much work to be done. Many energy improvements so far have focused with minimization of energy associated in the cluster head selection process or with generating a desirable distribution of cluster heads. Optimal clustering in terms of energy efficiency should eliminate all overhead associated not only with the cluster head selection process, but also with node association to their respective cluster heads. Sensor network reliability is currently addressed in various algorithms by utilizing re-clustering that occurs at various time intervals; however the result is often energy inefficient and limits the time available within a network for data transmission and sensing tasks.

Finally E-LEACH, TL-LEACH, MLEACH, LEACH-C and VLEACH suggested the improvements in the basic LEACH protocols in order to improve the performance of the basic LEACH protocols.

Chapter 3

Study of WSN Simulators: TOSSIM and NS-2

The goal for any simulator is to accurately model and predict the behavior of a real world environment. Developers are provided with information on feasibility and reflectivity crucial to the implementation of the system prior to investing significant time and money. This is especially true in sensor networks, where hardware may have to be purchased in large quantities and at high cost. Even with readily available sensor nodes, testing the network in the desired environment can be a time consuming and difficult task. Simulation-based testing can help to indicate whether or not these time and monetary investments are wise. Simulation is, therefore, the most common approach to developing or testing new protocol for a sensor networks. Choosing a right simulation tools has been a key step to get accurate prediction of real world environment. This chapter will cover the TinyOS Simulator (TOSSIM) and Network Simulator (NS-2).

3.1 TinyOS

TinyOS[17] is an operating system specifically designed for sensor networks. It has a component-based programming model, provided by the nesC language[19], a dialect

of C. TinyOS is not an OS in the traditional sense. It is a programming framework for embedded systems and set of components that enable building an application specific OS into each application. A TinyOS program is a graph of components, each of which is an independent computational entity. Each TinyOS component has a frame, a structure of private variables that can only be referenced by that component. Components have three computational abstractions: *commands, events,*and *tasks.* Commands and events are mechanisms for inter-component communication, while tasks are used to express intra-component concurrency. A command is typically a request to a component to perform some service, such as initiating a sensor reading, while an event signals the completion of that service. Events may also be signaled asynchronously, for example, due to hardware interrupts or message arrival. From a traditional OS perspective, commands are analogous to downcalls and events to upcalls. Commands and events cannot block: rather, a request for service is split phase in that the request for service (the command) and the completion signal (the corresponding event) are decoupled. The command returns immediately and the event signals completion at a later time. Rather than performing a computation immediately, commands and event handlers may post a task, a function executed by the TinyOS scheduler at a later time. This allows commands and events to be responsive, returning immediately while deferring extensive computation to tasks. While tasks may perform significant computation, their basic execution model is run-to-completion, rather than to run indefinitely. This allows tasks to be much lighter-weight than threads. Tasks represent internal concurrency within a component and may only access that components frame. The TinyOS task scheduler uses a non-preemptive, FIFO scheduling policy. TinyOS abstracts all hardware resources as components. For example, calling the *getData()* command on a sensor component will cause it to later signal a *dataReady()* event when the hardware interrupt fires. While many components are entirely software-based, the combination of split-phase operations and tasks makes this distinction transparent to the programmer. For example, consider a component that encrypts a buffer of data. In a hardware implementation, the command

would instruct the encryption hardware to perform the operation, while a software implementation would post a task to encrypt the data on the CPU. In both cases an event signals that the encryption operation is complete.
TinyOS simulator has four key requirements:

- Scalability: The simulator must be able to handle large networks of thousands of nodes in a wide range of configurations.

- Completeness: The simulator must cover as many system interactions as possible, accurately capturing behavior at a wide range of levels. Algorithm and network protocol simulations are helpful, but the reactive nature of sensor networks requires simulating complete applications.

- Fidelity: The simulator must capture the behavior of the network at a fine grain. Capturing subtle timing interactions on a mote and between motes is important both for evaluation and testing.

- Bridging: The simulator must bridge the gap between algorithm and implementation, allowing developers to test and verify the code that will run on real hardware. Often, algorithms are sound but their implementations are not.

3.2 TOSSIM

By looking into the specific requirements and the challenges of the TinyOS, Philip Levis at.el [17] presents TinyOS Simulator, called TOSSIM.

TOSSIM captures the behavior and interactions of networks of thousands of TinyOS motes at network bit granularity. Figure 3.1 shows a graphical overview of TOSSIM. The TOSSIM architecture is composed of five parts: support for compiling TinyOS component graphs into the simulation infrastructure, a discrete event queue, a small number of re-implemented TinyOS hardware abstraction components, mechanisms

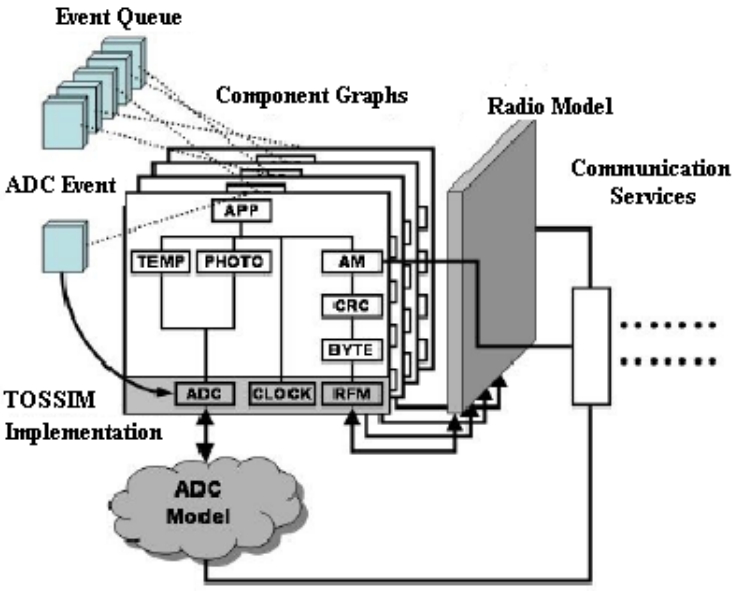


Figure 3.1: TOSSIM Architecture

for extensible radio and ADC models, and communication services for external programs to interact with a simulation. TOSSIM takes advantage of TinyOSs structure and whole system compilation to generate discrete-event simulations directly from TinyOS component graphs. It runs the same code that runs on sensor network hardware. By replacing a few low-level components (e.g., those shaded in Figure 3.1), TOSSIM translates hardware interrupts into discrete simulator events; the simulator event queue delivers the interrupts that drive the execution of a TinyOS application. The remainder of TinyOS code runs unchanged. In order to achieve its goal of scalability, each node in the simulator is connected in a directed graph where each edge has a probabilistic bit error. For perfect transmission, the bit error is 0, and can be changed for different situations. Also in the name of efficiency, every node in TOSSIM runs the same application code; all nodes are identical. The goal of scalability is successfully achieved. By default, TOSSIM captures TinyOS behavior at a very low level. It simulates the network at the bit level, simulates each individual ADC capture, and every interrupt in the system that shows the fidelity of the TOSSIM simulator.

3.2.1 TinyViz: TOSSIM Graphical User Interface

TinyViz, the TOSSIM visualization tool as shown in figure 3.2, illustrates the capabilities of TOSSIMs communication services. TinyViz is a Java based graphical user interface for TOSSIM, allowing simulations to be visualized, controlled, and analyzed. TinyViz provides visual feedback on the simulation state and mechanisms for controlling the running simulation, e.g., modifying ADC readings and radio loss probabilities. TinyViz provides a plugin interface allowing developers to implement their

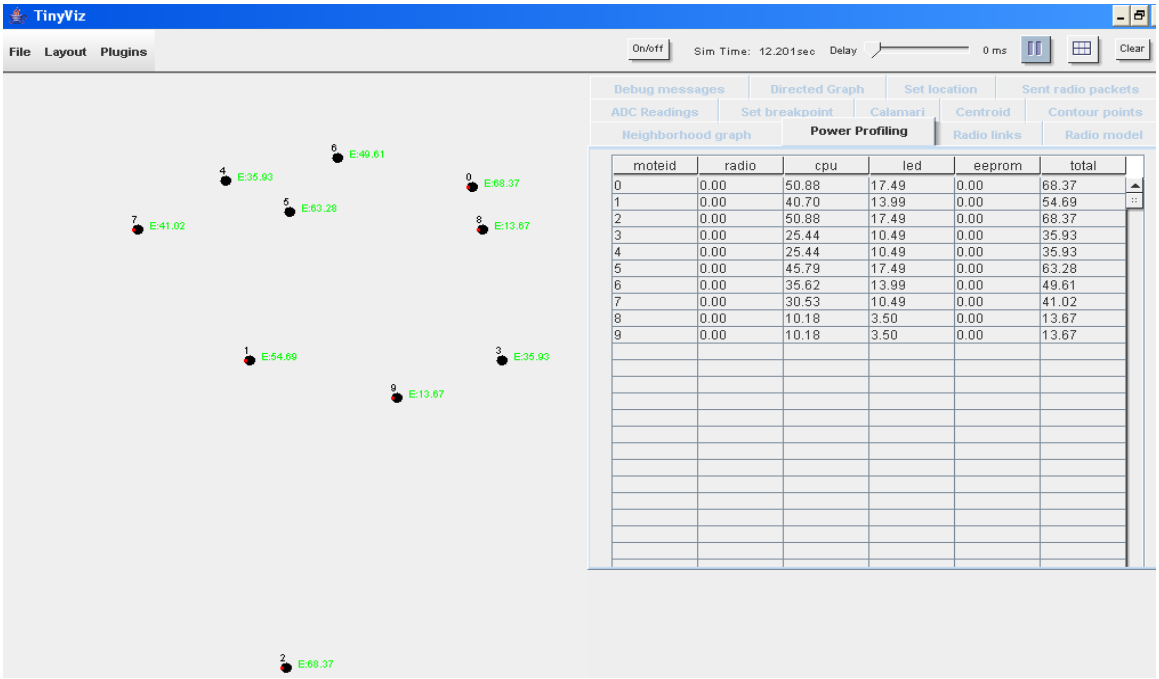


Figure 3.2: A TinyViz GUI

own application-specific visualization and control code within the TinyViz engine. Users interact with a simulation by loading plugins that provide desired functionality. The TinyViz engine publishes TOSSIM events to loaded plugins. TinyViz has a set of default plugins that provide basic debugging and analysis capabilities. A sensor plugin that displays mote sensor values in the GUI allows the user to set individual mote sensor values during simulation. TinyViz comes with a suite of built-in plugins, in the tools/java/net/tinyos/sim/plugins directory.

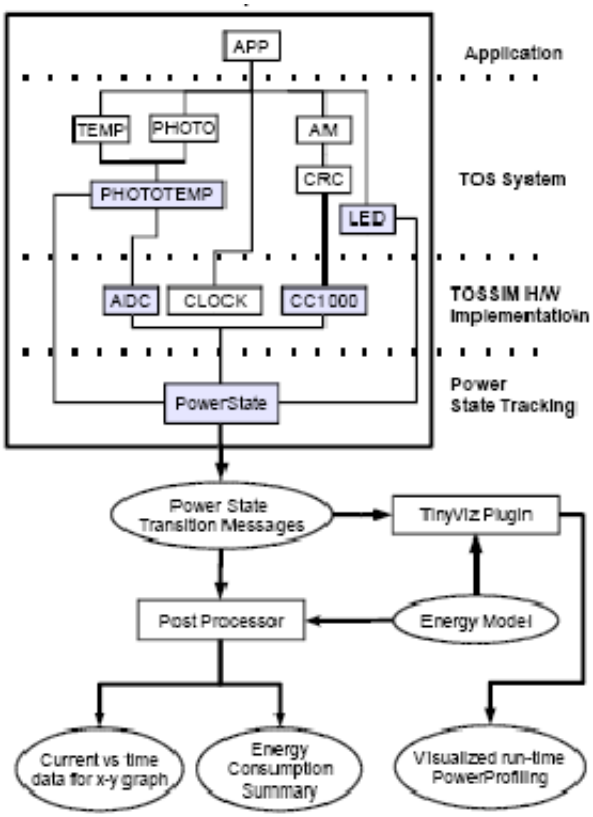


Figure 3.3: PowerTOSSIM Architecture

3.2.2 PowerTOSSIM

PowerTOSSIM is based on the TinyOS operating system and the TOSSIM simulation environment [18], which is a extension of TOSSIM simulator. Figure 3.3 illustrates the architecture of PowerTOSSIM. It makes use of the TinyOS and TOSSIM component model to instrument hardware state transitions for the purpose of tracking power consumption.

3.3 Network Simulator: NS-2

The network simulator (NS)[20], which is a discrete event simulator for networks, is a simulated program developed by VINT (Virtual InterNetwork Testbed) project group (A Collaboration among USC/ISI, Xerox PARC, LBNL, and UCB). It supports

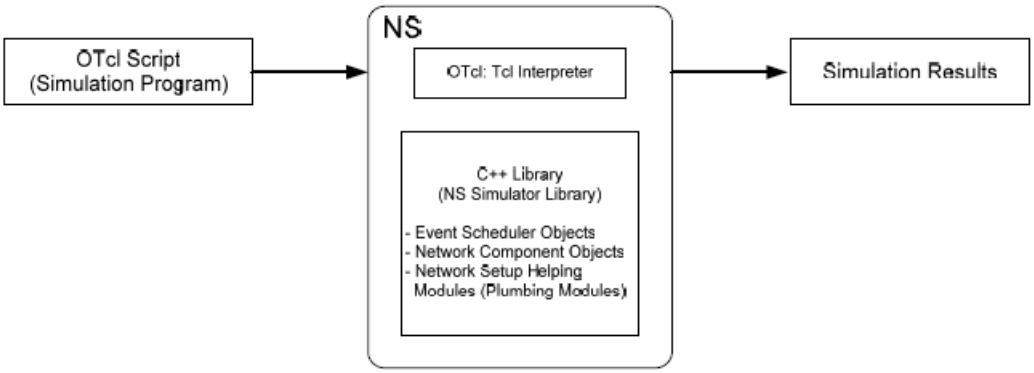


Figure 3.4: Basic structure of NS

simulations of TCP and UDP, some of MAC layer protocols, various routing and multicast protocols over both wired and wireless network etc. The basic structure of NS-2 is as shown in figure 3.4. To setup and run a simulation, a user writes an OTcl script, which is a simulation program to initiate an event scheduler, set up the network topology using the network objects and plumbing functions in the library, and to tell traffic sources when to start and stop transmitting packets through the event scheduler. When NS-2 which works as OTcl interpreter receives the OTcl script, it will set environment parameters following the received script. If a user wants to make a new network object, it will be easy to make a compound object from the object library, and plumb the data path through the object rather than write a new one. When the simulation is finished, the simulation results are produced in one or more text-based output files that contain detailed simulation data, which can be used to analyze directly.

3.4 MIT's extension to NS-2

The MIT uAMPS extensions to ns [21] add support for large scale wireless sensor networks. These extensions include models for node energy dissipation and node state, as well as several routing protocols. It describes the uAMPS additions to ns and gives details on how to use the code and run the simulator for wireless sensor

networks.

3.5 Summary

This chapter presented TinyOS operating system which is specifically designed for sensor networks and basic requirements of the TinyOS simulator to capture the real world phenomena. TOSSIM is the simulator which captures the behavior and interactions of TinyOS motes. The graphical user interface for the TOSSIM, TinyViz and PowerTOSSIM for simulating the power profile of the sensor nodes are also included. Later part of the chapter discussed about network simulator NS-2 and its extension for the implementation of LEACH routing protocol.

Chapter 4

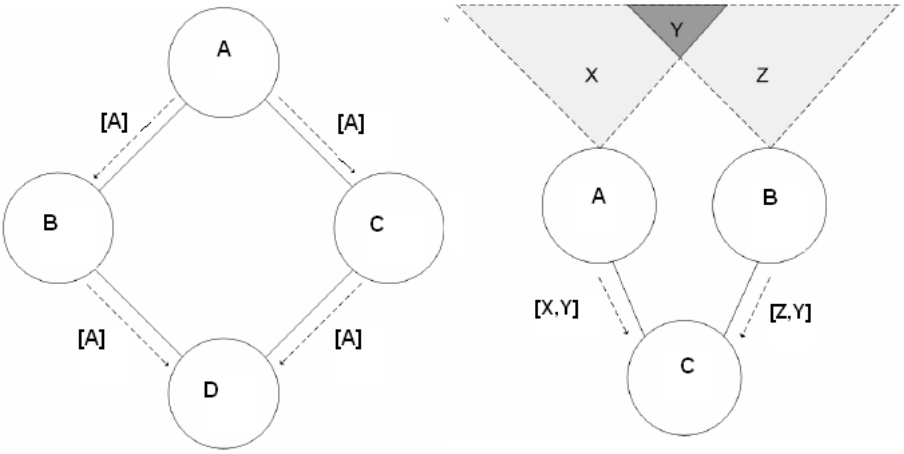
The Implementation of Flat Routing protocols using TOSSIM

This chapter will contains the simulation of following two routing protocols and its implementation specifications in TinyOS environment.

- Flooding
- Gossiping

4.1 Flooding Protocol

In the flooding protocol [2] each sensor node receiving a data or management packet repeats the packet by broadcasting it. Only packets which are destined for the node itself or packets whose hop count has exceeded a preset limit are not forwarded. The main benefit of flooding is that it requires no costly topology maintenance or route discovery. Once sent a packet will follow all possible routes to its destination. If the network topology changes sent packets will simply follow the new routes added.
Flooding does however have several problems. One such problem is implosion. Implosion is where a sensor node receives duplicate packets from its neighbors. Figure 4.1a illustrates the implosion problem. Node A broadcasts a data packet ([A]) which



(a) Implosion problem. (b) Overlap problem.

Figure 4.1: Implosion and Overlap problem in Flooding protocol.

is received by all nodes in range (nodes B and C in this case). These nodes then forward the packet by broadcasting it to all nodes within range (nodes A and D). This results in node D receiving two copies of the packet originally sent by node A. This can result in problems determining if a packet is new or old due to the large volume of duplicate packets generated when flooding. Overlap is another problem which occurs when using flooding. If two nodes share the same observation region both nodes will witness an event at the same time and transmit details of this event. This results in nodes receiving several messages containing the same data from different nodes. Figure 4.1b illustrates the overlap problem. Nodes A and B both monitor geographic region Y. When nodes A and B flood the network with their sensor data node C receives two copies of the data for geographic region Y as it is included in both packets.

4.2 Gossiping Protocol

The Gossiping protocol [2] is an extension of the flooding protocol. Instead of broadcasting each packet to all neighbors, the packet is sent to a single neighbor chosen at random from a neighbor table. Having received the packet the neighbor chooses

another random node to send to. This can include the node which sent the packet. This continues until the packet reaches its destination or the maximum hop count of the packet is exceeded. Gossiping avoids the implosion problem experienced by flooding as only one copy of a packet is in transit at any one time. However the protocol does take a long time to deliver a packet to its destination as the hop count can become quite large due to the protocols random nature.
Flooding is one of the simplest protocols available and is often used as part of other protocols. Flooding was chosen for implementation and evaluation as it provides a base protocol to compare more elaborate protocols against. Gossiping is very similar to flooding but avoids the implosion problem by forwarding packets to a random neighbor. Gossiping was chosen in order to evaluate the improvements it offers over flooding.

4.3 Implementation Environment

4.3.1 Routing Protocol Specification

Packet Format

TinyOS provides the TOS_Msg structure as the underlying packet format for network communications. The message format used for the simulation is as follows:

- sendingNode (2 bytes) the node that has sent the packet.
- originNode (2 bytes) the node that generated the packet.
- seqNo (2 bytes) the sequence number of the packet.
- hopCount (2 bytes) the number of hops the packet has travelled.
- data [21] (8 bytes) the payload field of length 21 bytes.

Flooding

The protocol operates in the following way:

- Upon receiving a packet the hopCount field is incremented by one.
- Each node maintains a list of previously seen packets. This list is of a set size containing the details of the most recently seen packets. A combination of the originNode and seqNo fields is used as a unique identifier of a packet. If the originNode and seqNo fields of a received packet match any of the values in the list the packet will be dropped as a duplicate. This approach is an attempt to limit the implosion problem.
- If the packet is new, a sensor node will forward the packet as long as the hopCount field has not reached the maximum value allowed.
- If the node is the base station the packet will be not forwarded otherwise the packet will be broadcast to every node within range.

Gossiping

The Gossiping protocol relies on a neighbor table to keep a list of currently alive neighbor nodes. In order to maintain this list each node sends an advertisement packet periodically. The packet has the following structure:

- nodeID (2 bytes) the address of the node.

The protocol operates in the following way:

- Upon receiving a packet the hopCount field is incremented by one.
- If the node is the base station the packet is not forwarded.
- Other nodes forward the packet to a random neighbor from its neighbor table so long as the maximum hop count for the packet has not been exceeded. If no alive neighbor nodes are listed in the neighbor table the packet is dropped.

Gossiping does not require the use of a sequence number as only one copy of each packet will ever be in transit at any one time.

4.3.2 Driver Application

In order to gather data about the operation of the chosen routing protocols during simulation network traffic is required. In order to generate this traffic a simple application is required which transmits data periodically using the multi-hop messaging services provided. The application chosen is a simple temperature monitor. The application periodically takes a temperature reading. This reading is then time stamped and sent to the base station using the multi-hop routing layer. The data is carried in an application packet which has the following format:

- address (2 bytes) the address of the origin node.
- timestamp (4 bytes) the time at which the packet was sent.
- reading (2 bytes) the sensor reading.

4.3.3 Components Overview

- TimeSyncM: This module provides a basic time synchronization service for the network. Nodes synchronies to a network global time which is based on the local time of node 0.
- MHEngineM: This module is a modified version of MultiHopEngineM from the TinyOS multi-hop routing component library. This module is the main component of the multi-hop messaging layer and provides the packet movement logic.
- MHFloodingPSM: This module provides the route selection logic for the flooding protocol.

- MHGossipingPSM: This module provides the route selection logic for the gossiping protocol.

- TempMonM: This module provides network traffic by sending packets containing sensor data periodically.

The whole architecture of the various Modules and Configuration components, is as shown in figure 4.2.

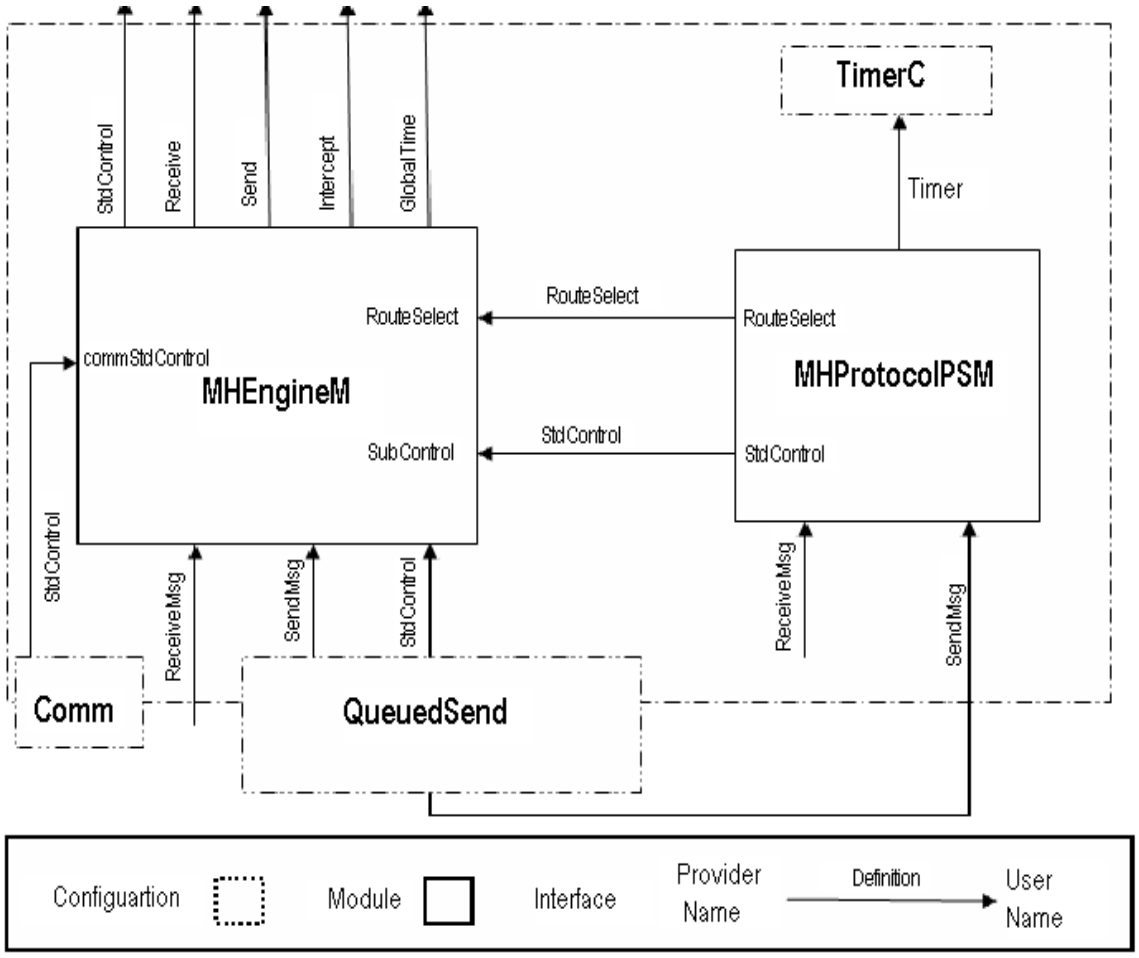


Figure 4.2: Modules and Configuration component architecture using TOSSIM.

4.3.4 Evaluation Metrics

The following metrics were chosen for evaluating the protocols:

- **Latency:** Latency may be defined as the time taken to deliver a packet to the base station from the origin node. Latency will be looked at while evaluating these two protocols. Higher latency results in more delay. Thus, lower latency is preferable to higher latency.

- **Power Usage:** The amount of power used during the simulation will be monitored and used for evaluating the protocols. Batteries have a finite amount of power and the unattended nodes die once power runs out. For this reason lower power usage is preferable to higher power usage.

- **Overhead:** This is calculated as a ratio between the average number of packets received by the node and the average number of packets generated by the node. Higher the overhead the higher is the power usage since power is unnecessarily spend in forwarding the redundant packets received by the node.

- **Scalability:** Scalability of the protocol is calculated on the basis of the above three metrics. The three metrics are carefully examined as the number of nodes in the network increases. The number of sensor nodes deployed in the sensing area are increased. Any routing scheme must be able to work with this huge number of sensor nodes.

The hardware documentation indicates that transmission of a packet requires roughly 1/3 more power than required to receive a packet[1]. Constants were defined to account for this. Receiving was considered to use 2 power units where as transmission was considered to use 3 units. Every time a packet was transmitted 3 units was added to the used power count and every time a packet was received 2 units was added to the count. In order to track the number of packets sent, forwarded and received by a node code was added to relevant methods to increment the corresponding variable

every time a packet was sent, forwarded or received. Note that the number of packets sent refers to the number of packets generated by the local node which were then sent.

In order to get the latency, track the number of packets received from each node by the base station as well as time taken for packets to reach the base station from those nodes.

Overhead is calculated based on the following ratio:

$$Overhead = \frac{Average\ \ no\ \ of\ \ packets\ \ received\ \ by\ \ the\ \ node}{Average\ \ no\ \ of\ \ packets\ \ generated\ \ by\ \ the\ \ node} \tag{4.1}$$

4.4 Testing and Analysis of results

This section includes the simulation test cases which are carried out on the TinyOS simulator (TOSSIM) and analysis of the results.

4.5 Test cases

The performed test cases are as listed below in the table.

Table 4.1: Summary of Test cases using TOSSIM

Test case No	Protocol Used	Time of Simulation (in minute)	No of sensor nodes
1	Flooding	10	25
2	Flooding	10	50
3	Flooding	10	100
4	Gossiping	10	25
5	Gossiping	10	50
6	Gossiping	10	100
7	Flooding	30	25
8	Flooding	30	50
9	Flooding	30	100
10	Gossiping	30	25
11	Gossiping	30	50
12	Gossiping	30	100

In addition to the testing carried out using debug statements, TinyViz was used to check that the protocols are visually working as expected.

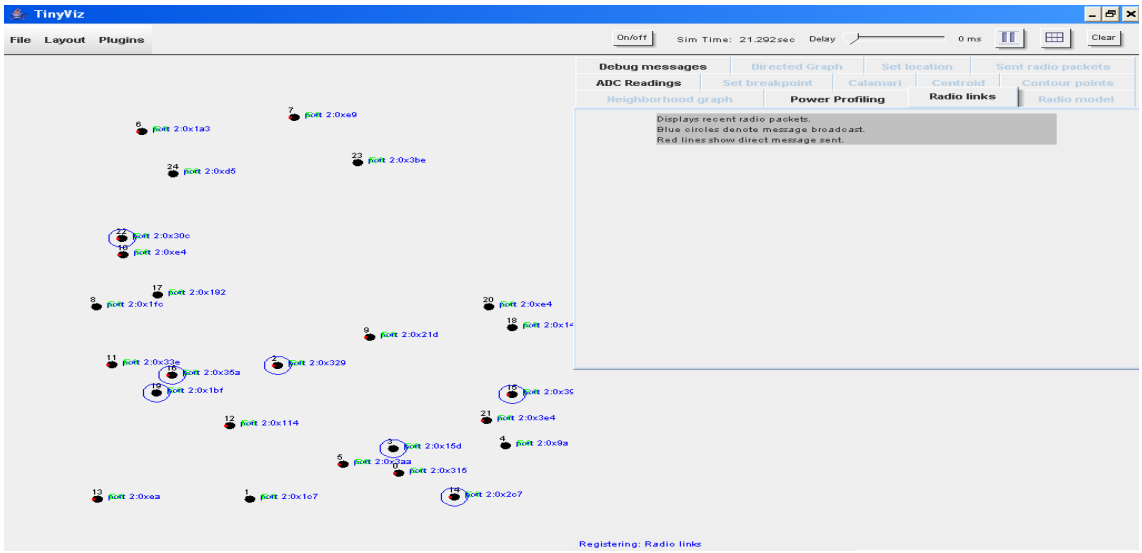


Figure 4.3: Flooding protocol displayed using TinyViz.

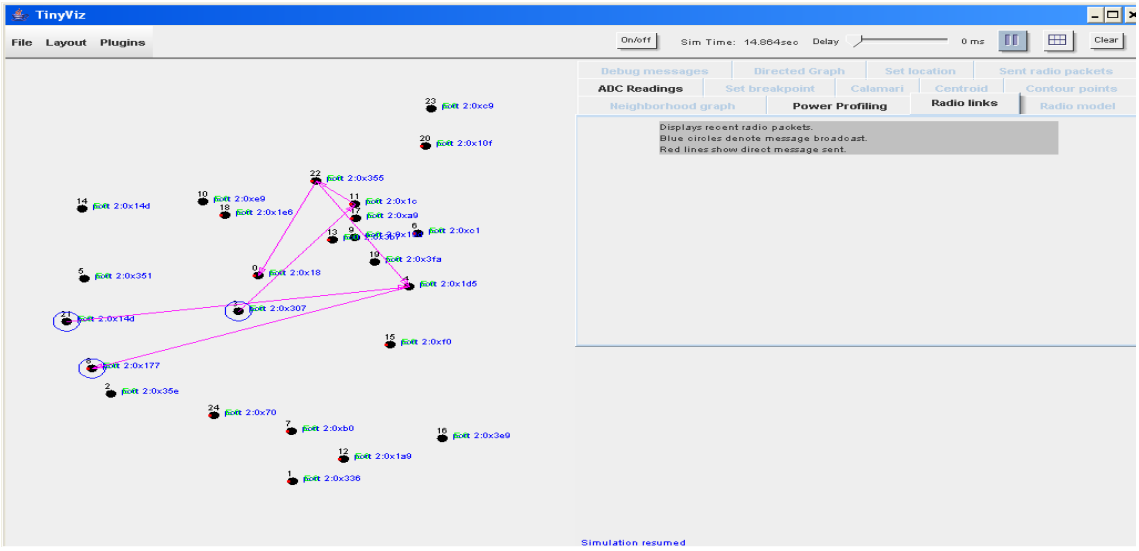


Figure 4.4: Gossiping protocol displayed using TinyViz.

The TinyViz window is shown in figure 4.3 for the flooding protocol, where blue circle around the nodes indicates the broadcasting of the packets. While figure 4.4

shows the gossiping protocol where directed arrow indicates the unicast of the packet to the particular neighbor node.

4.6 Result Analysis

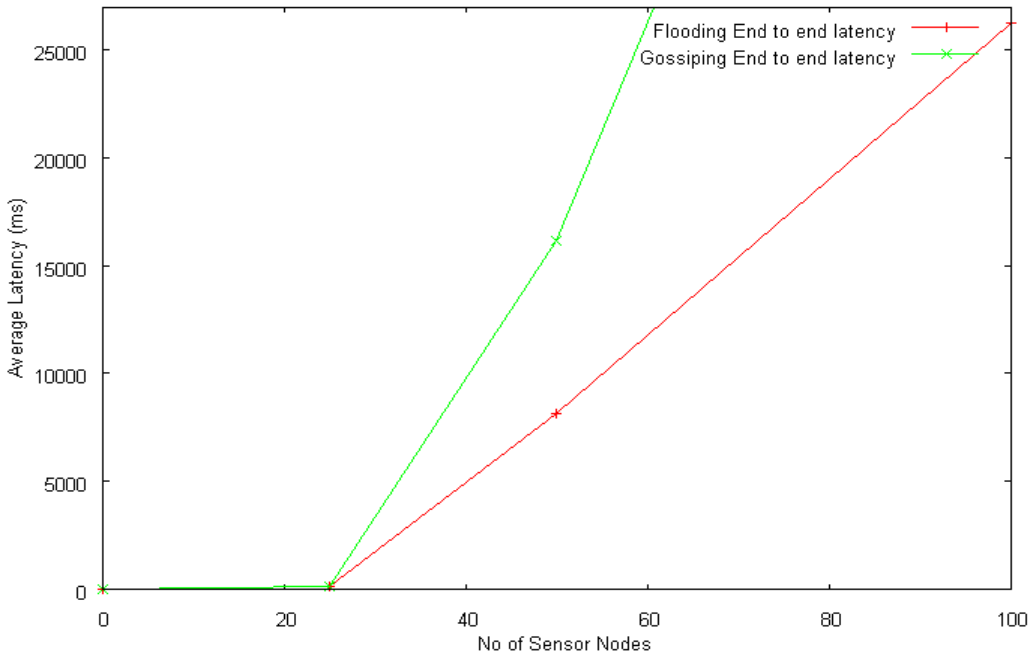
Latency:

The first metric mentioned was the latency of the protocol. In order to evaluate latency the average end to end delay and the average per hop delay was calculated for the output of each simulator run. A lower latency is better than a higher latency.

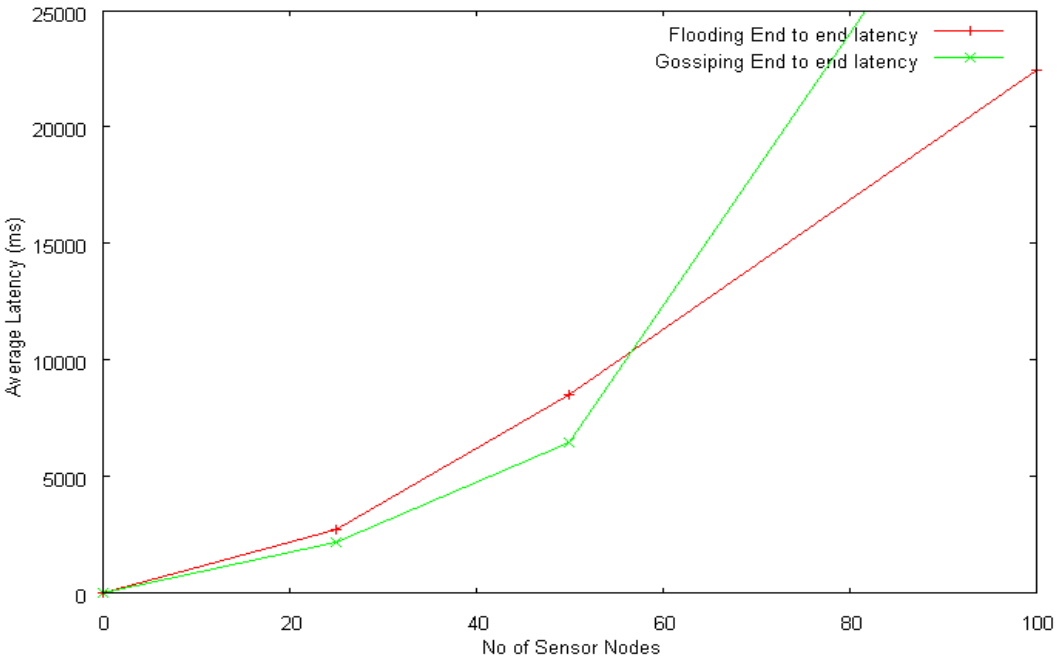
Table 4.2: Latency Measurement for Flooding and Gossiping protocol

Protocol	**Time**	**No of Sensor Nodes**	**Average Latency**	
			End to End	**Per Hop**
Flooding	10 minute	25	122.6112	122.6112
	10 minute	50	8132.8	8132.8
	10 minute	100	26230.02	26230.02
	30 minute	25	2687.92	2687.92
	30 minute	50	8482.68	8482.68
	30 minute	100	22438.54	22438.54
Gossiping	10 minute	25	82.8812	13.8832
	10 minute	50	16126.44	4279.04
	10 minute	100	67214.84	23861.48
	30 minute	25	2171.36	486.52
	30 minute	50	6437.64	2223.58
	30 minute	100	35762.36	14090.54

Table-II shows the average end-to-end and per hop latency for the each cases while figure 4.5 and figure 4.6 shows the corresponding graphs. From this results it is clear that end-to-end latency is better for flooding than the gossiping.

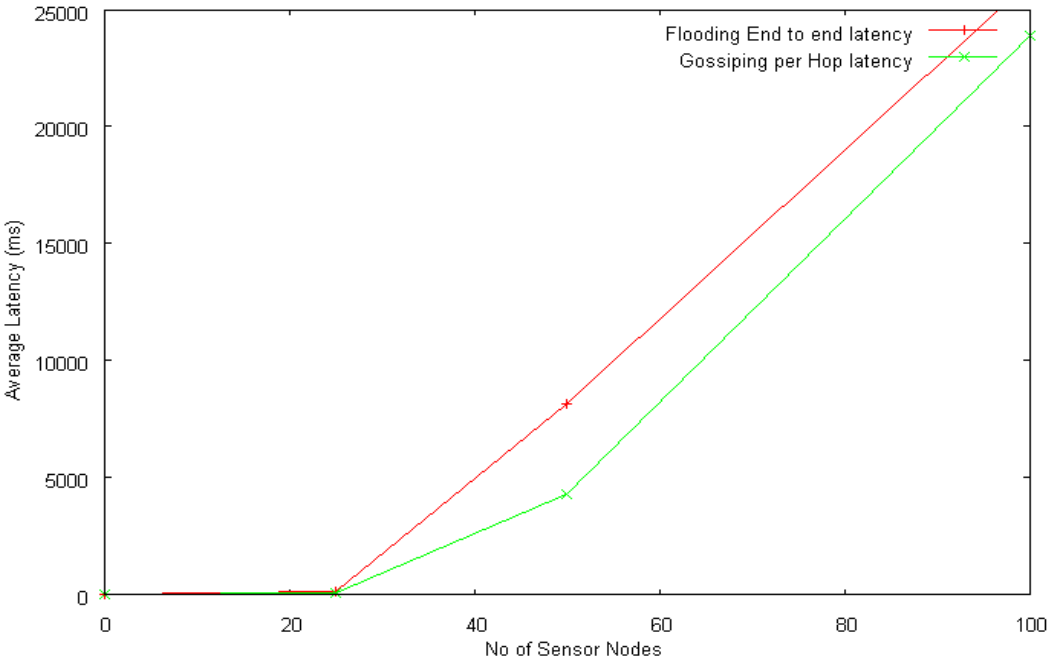


(a) For 10 minute.

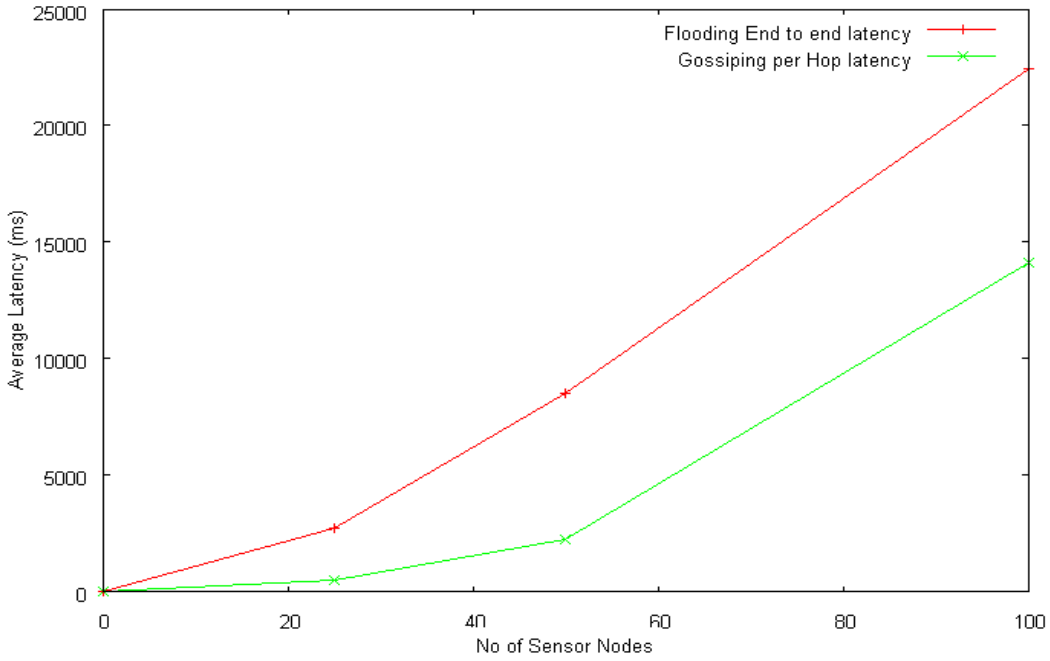


(b) For 30 minute.

Figure 4.5: End to End Latency in Flooding and Gossiping protocol.



(a) For 10 minute.



(b) For 30 minute.

Figure 4.6: Per Hop Latency in Flooding and Gossiping protocol.

Power Profile:

The second metric, power usage was measured by calculating the amount of power consumption for each sensor nodes during each simulation run. Figure 4.7 to 4.12 shows the corresponding graphs for the various test cases. From the graphs it is clear that the power consumption in the flooding is more due its broadcasting nature.

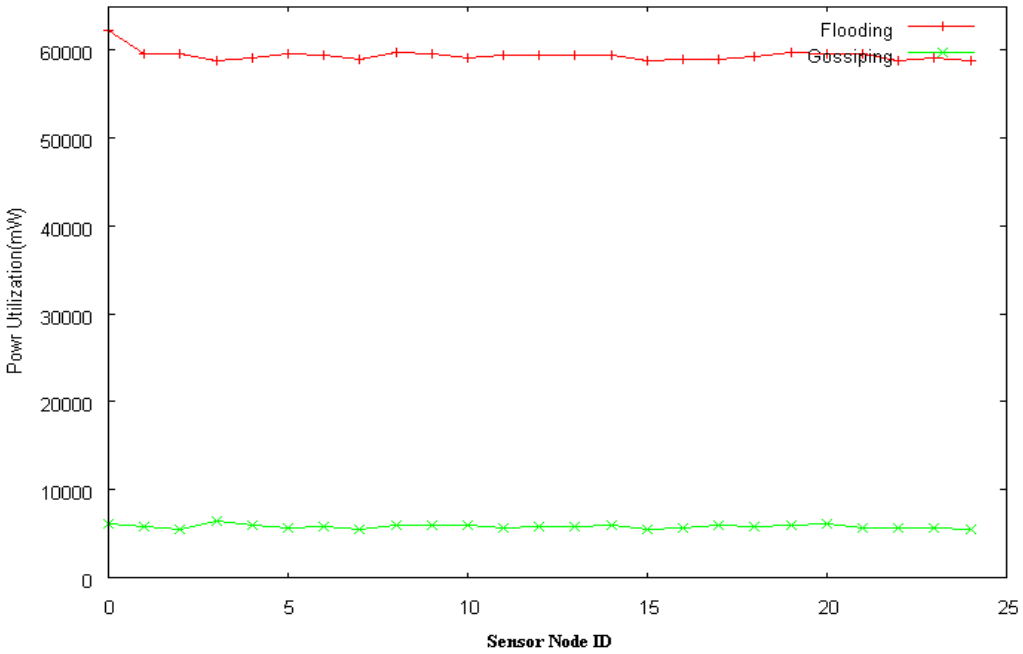


Figure 4.7: Power Profile for 25 sensor nodes in 10 minute.

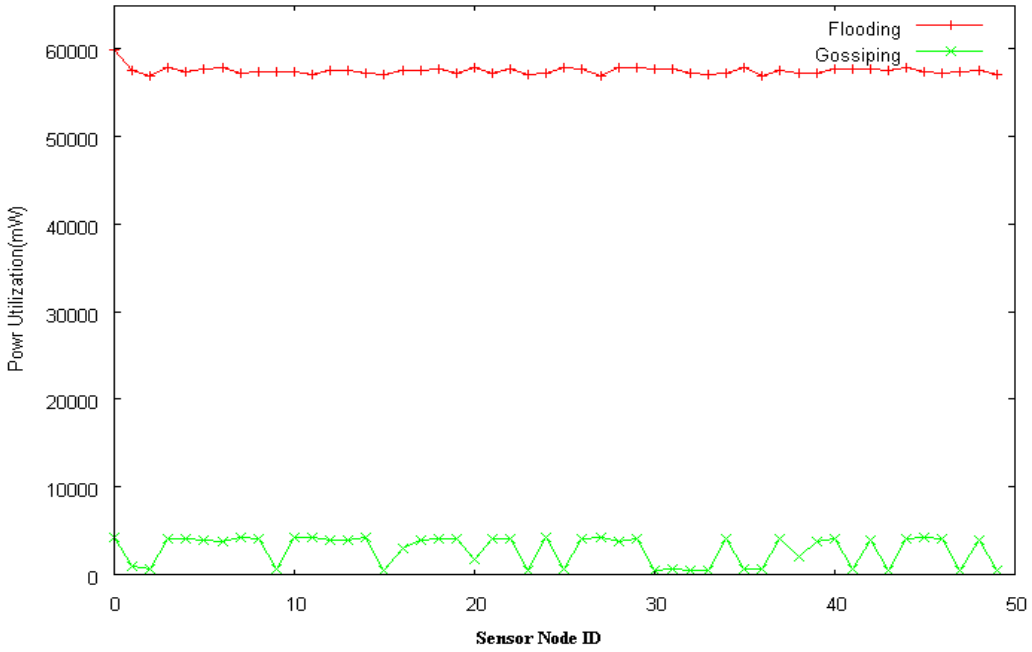


Figure 4.8: Power Profile for 50 sensor nodes in 10 minute.

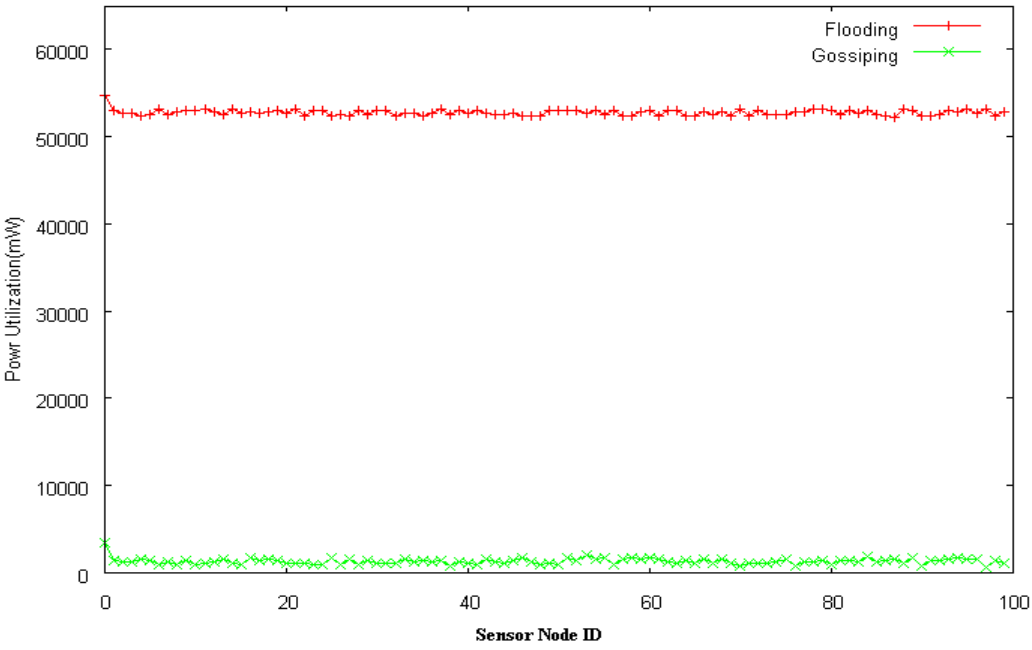


Figure 4.9: Power Profile for 100 sensor nodes in 10 minute.

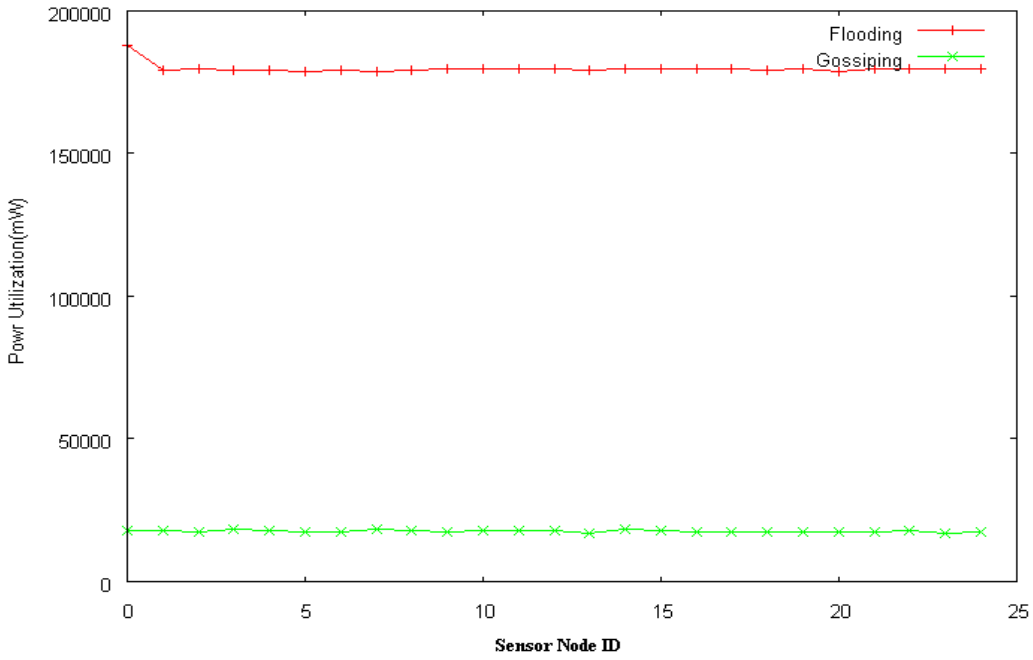


Figure 4.10: Power Profile for 25 sensor nodes in 30 minute.

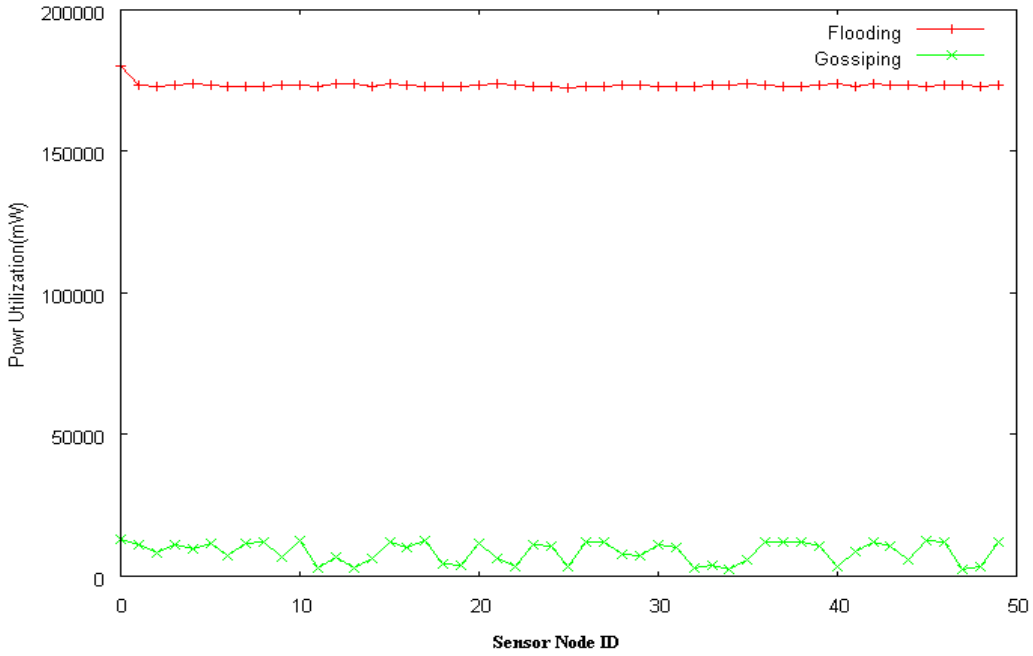


Figure 4.11: Power Profile for 50 sensor nodes in 30 minute.

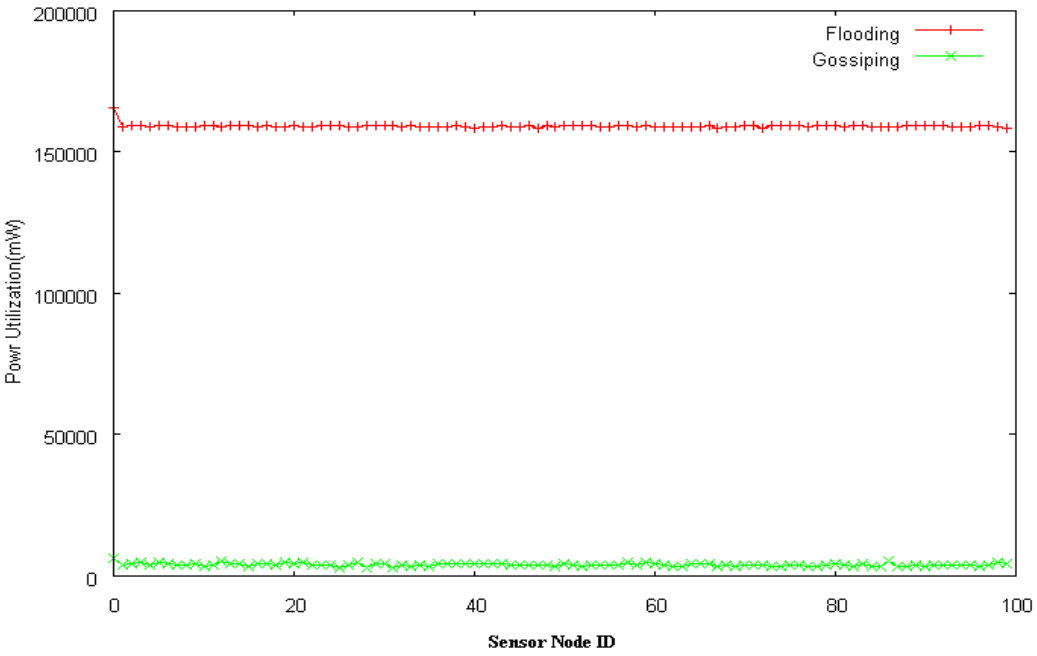


Figure 4.12: Power Profile for 100 sensor nodes in 30 minute.

Overhead:

This is calculated as a ratio between the average number of packets received by the sensor node and the average number of packets generated by the sensor node. The figure 4.13 and 4.14 shows the overhead profile for sensor nodes during 10 minute and 30 minute simulation run respectively. From the graphs it is clear that the overhead in the flooding is more.

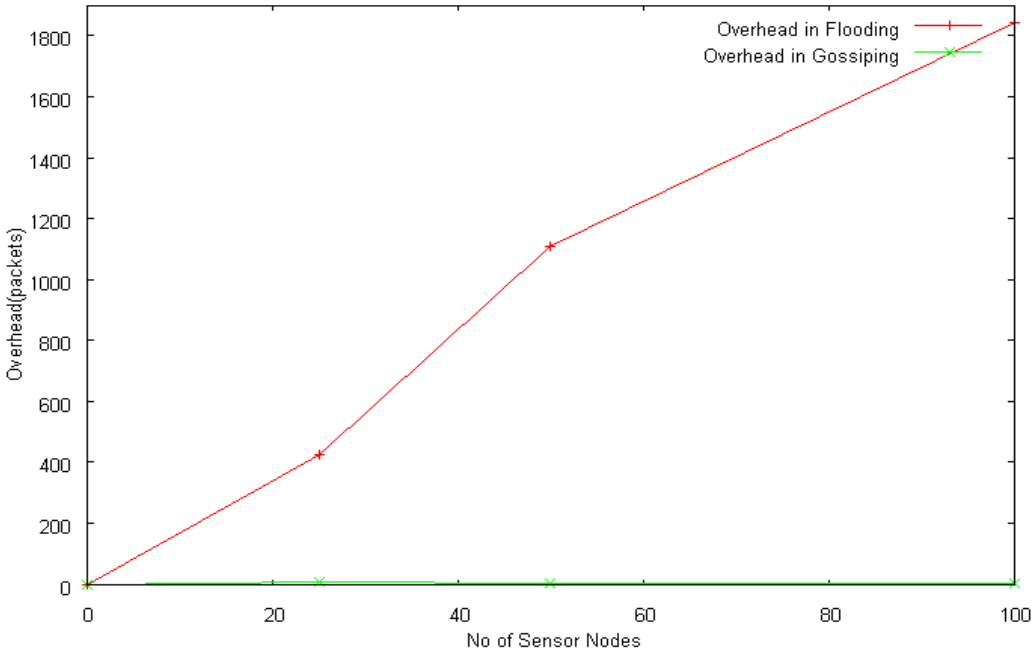


Figure 4.13: Overhead in Flooding and Gossiping protocol in 10 minute.

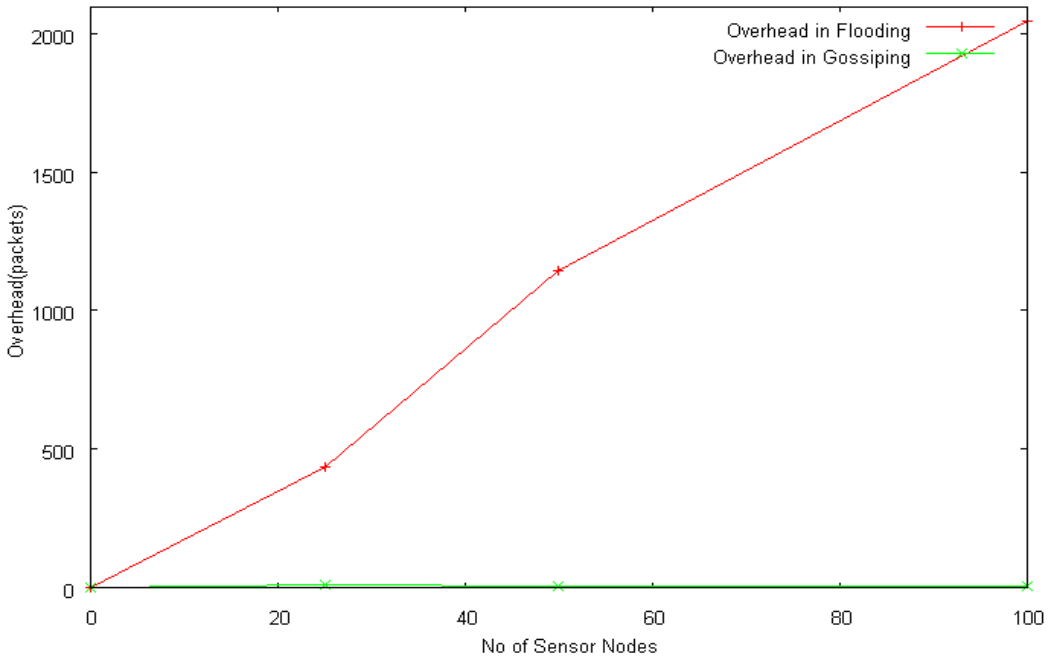


Figure 4.14: Overhead in Flooding and Gossiping protocol in 30 minute.

4.7 Summary

The chapter presented the Flooding and Gossiping operations implementation in the TinyOS environment. Flooding uses a large amount of power and would drain the batteries of the nodes quickly. Gossiping uses significantly less power than flooding. Overhead is also more in the case of flooding than the gossiping protocol while end-to-end latency is less in the case of flooding protocol than the gossiping. In conclusion it is clear that none of the protocols are really suitable for use in sensor networks as implemented. When considering multi-hop routing, flooding is of limited use. Extensions of the flooding is gossiping protocol, which overcome the problem of implosion and overlapping in sensor networks.

Chapter 5

Simulation of LEACH routing protocol using NS-2

This chapter will cover the basic simulation models like channel propagation and radio energy used for simulation of the LEACH routing protocol. It will also includes the various kind of analysis on the results obtained after the simulation of LEACH protocol. To simulate the LEACH protocol, MITs NS2 extension for LEACH simulator [21] is used.

5.1 Channel Propagation Model

In the wireless channel, the electromagnetic wave propagation can be modeled as falling off as a power law function of the distance between the transmitter and receiver. In [21], free space and two-tay ground propagation models are described. Both free space model which considers direct line-of-sight and two-ray ground propagation model which considers ground reflected signal also, were considered depending upon the distance between transmitter and receiver. If the distance is greater than $d_{crossover}$ distance, two-ray ground propagation model is used. The crossover is defined as follows:

$$d_{crossover} = \frac{4 * \pi * \sqrt{L} * h_r * h_t}{\lambda} \tag{5.1}$$

Where, L $\geq$ 1 is system loss factor. h_r is the height of the receiving antenna, h_t is the height of the transmitting antenna and λ is the wavelength of the carrier signal. The transmit power is attenuated based on following formula:

$$P_r(d) = \begin{cases} \frac{P_t * G_t * G_r * \lambda^2}{(4*\pi*d)^2 * L} & \text{if } d < d_{crossover} \\ \frac{P_t * G_t * G_r * h_t^2 * h_r^2}{d^4} & \text{if } d \geq d_{crossover} \end{cases} \tag{5.2}$$

Where, P_r is the received power at distance d, P_t is transmitted power , G_t is the gain of the transmitting antenna and G_r is the gain of the receiving antenna.

5.2 Radio Energy Model

The radio energy model describing the radio characteristics, includes energy dissipation in the transmit and receive modes. Transmitter dissipates energy to run the radio electronics and power amplifier whereas receiver dissipates energy to run the radio electronics [21]. Figure 5.1 shows the energy dissipation model. Using this radio

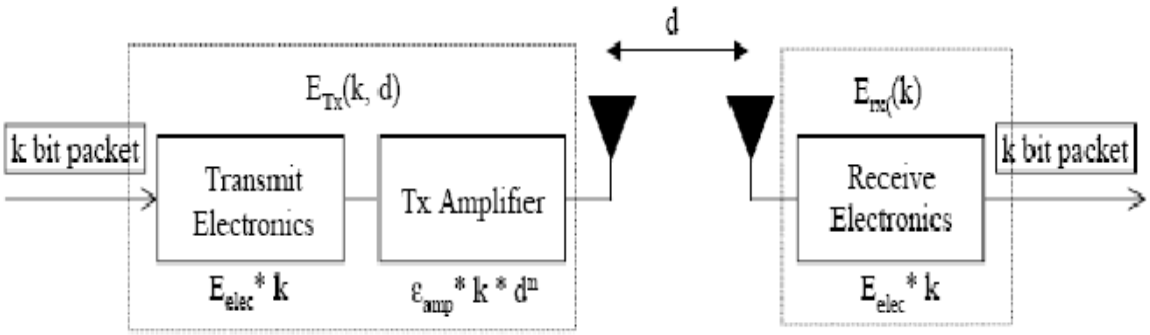


Figure 5.1: Radio energy dissipation model for LEACH protocol.

model, to transmit k-bit message at distance d the radio expends:

$$E_{Tx}(k,d) = E_{Tx-elec}(k) + E_{Tx-amp}(k,d) \tag{5.3}$$

$$E_{Tx}(k,d) = \begin{cases} E_{elec} * k + \epsilon_{friss-amp} * k * d^2 : d < d_{crossover} \\ E_{elec} * k + \epsilon_{two-ray-amp} * k * d^4 : d \geq d_{crossover} \end{cases} \quad (5.4)$$

and to receive this message, the radio expends:

$$E_{Rx}(k) = E_{Rx-elec}(k) \quad (5.5)$$

$$E_{Rx}(k) = E_{elec} * k. \quad (5.6)$$

5.3 Simulation experiments

For the simulation experiments, parameters for propagation model and energy model were assumed as per the following table 5.1:

Table 5.1: Various parameter values for LEACH simulation.

Description	Parameter	Value
Radio electronics Energy	E_{elec}	50 nJ/bit
Radio amplifier Energy	$\epsilon_{friss-amp}$ $\epsilon_{two-ray-amp}$	10 pJ/bit/m^2 0.0013 pJ/bit/m^4
Bitrate	R_b	1 Mbps
Antenna Gain factor	G_t , G_r	1
Antenna Height	h_t , h_r	1.5 m
Signal Wave length	λ	$\frac{3*10^8}{914*10^6}$ = 0.328m
System loss factor	L	1
Cross-over distance	$d_{crossover}$	$\frac{4*\pi*\sqrt{L}*h_r*h_t}{\lambda}$ = 86 m

A random test network was used having 100 nodes with base station located at (50, 175) (not shown) as shown in figure 5.2. For the experiments, each nodes was initial given 2J of energy.

In table 5.2, Simulation results shows how the performance of the sensor network using LEACH protocol varies as the percent of the nodes that are cluster-heads is changed. Here 0 and 100 percent cluster-heads is the same as direct communication. By these experiments, we concluded the optimal percent of nodes that should be

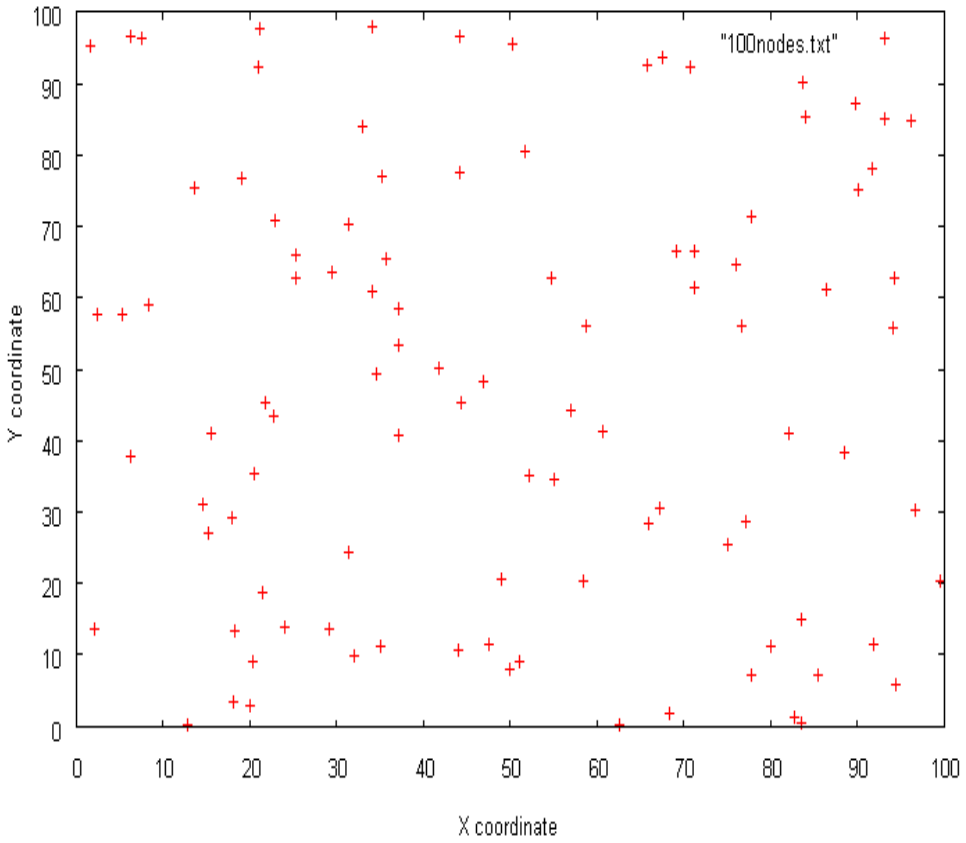


Figure 5.2: Sensor network topology with base station at (50,175)

cluster-heads.

Table 5.2: Simulation results of LEACH protocol.

% No of Clusters	Lifetime (s)	Throughput
3	285.09	32279
4	464.10	35897
5	542.30	52127
6	464.00	42041
8	181.39	8301

Figure 5.3 and 5.4 shows the network performance graphs in terms of lifetime and throughput of the network respectively.

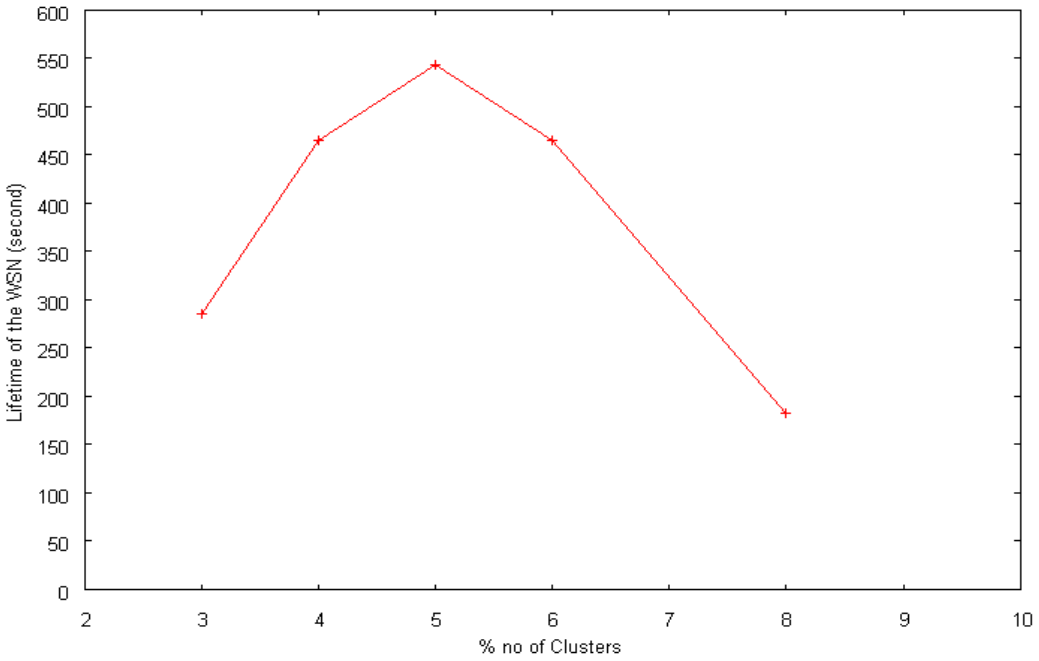


Figure 5.3: No of clusters Vs Lifetime of the network.

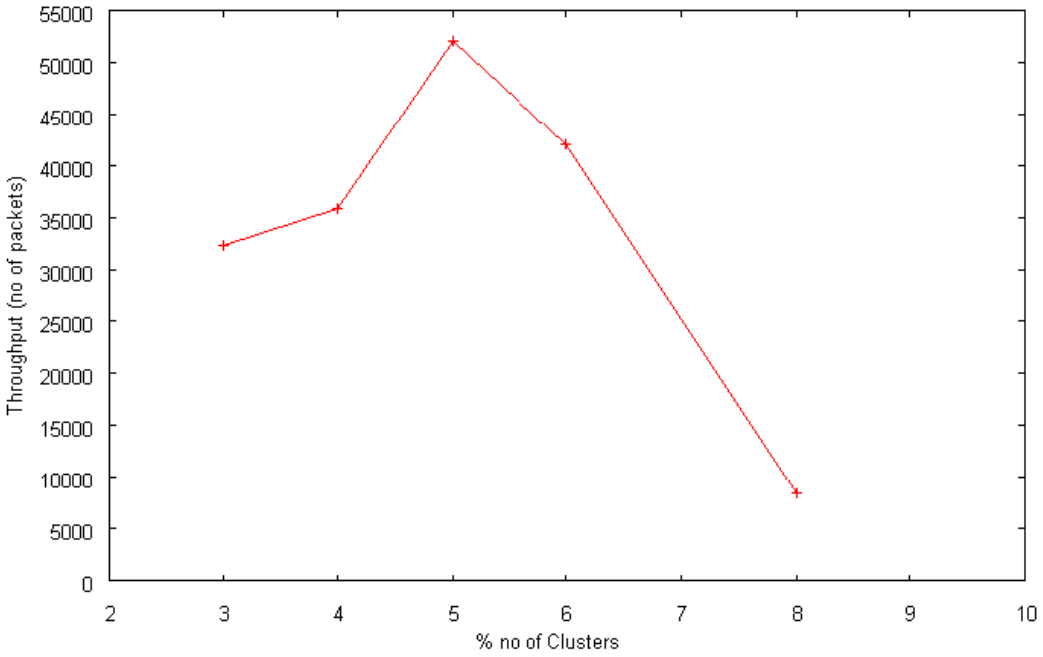


Figure 5.4: No of clusters Vs Throughput of the network.

5.4 Summary & Conclusions

The chapter discussed the propagation and energy models and its various parameters used for LEACH simulation. Experiment was performed by varying the cluster-heads numbers. From the results it is clear that performance of the sensor network is better when 5 percent of the nodes are cluster-head in the LEACH protocol. Despite the significant overall energy savings, however, the various assumptions made by LEACH protocol raise a number of issues like:

- LEACH assumes that all nodes begin with the same amount of energy and that the amount of energy a CH consumes is more than that of a non-cluster node.
- LEACH assumes that all nodes can communicate with each other and are able to reach the sink. Therefore, it is only suitable for small size networks.
- LEACH requires that all nodes are continuously listening. This is not realistic in a random distribution of the sensor nodes, for example, where cluster-heads would be located at the edge of the network.
- Finally, there is no mechanism to ensure that the elected cluster-heads will be uniformly distributed over the network. Hence, there is the possibility that most of the cluster-heads are concentrated in one part of the network, which is comparable with the problem like local minima.

Chapter 6

Proposed modification in LEACH routing protocol

This chapter will propose some modification in basic LEACH protocol in order to improve the performance of the protocol. The proposed I-LEACH (Improved LEACH) ensure that the elected cluster-heads will be uniformly distributed over the network. Hence, there is no possibility that all cluster-heads will be concentrated in one part of the network. The performance of the proposed I-LEACH protocol is evaluated mainly as per the following metrics:

- **Average Energy consumption:** The average energy consumed by the sensor nodes are measured at equal intervals.
- **Average Throughput:** The average number of packets received at the sink.
- **Life time of the network:** The total number of nodes which are alive at the end of all the cycles of the algorithms.

6.1 Proposed algorithm for I-LEACH

I-LEACH employs the distributed clustering approach as compare to LEACH protocol. The total sensor field is divided into the equal sub-region. The choice of the

cluster head (CH) from each sub-region is determined by the threshold approach as in LEACH protocol. Following is the algorithm for the I-LEACH protocol.

I-LEACH Algorithm

Let N_i or N_j denote a common node

$S(N_i) = (N_1, N_2 \ldots\ldots N_n)$ denote the set of n nodes

$E(N_i)$ denote energy in a node

N_{xyz} denote node location

C_i denote a cluster ID

$CH(N_i)$ denote a cluster head node.

d_{ij} denote distance measured from node Ni to Nj

$thresh(N_i)$ denote the threshold value of node Ni

Initialization

Create node N_i

Set node position N_{xyz}

Clusters formation

Divide the sensor field into equal sub-region R_i

Select CH from the each sub-region R_i based on threshold value.

if $N_i \in R_i$ && $thresh(N_i) < Threshold$ && $hasnotbeenCHyet$ **then**

    $N_i = CH\ (N_i)$ for sub-region R_i

else

    $N_i = N_j$ (normal node)

end if

Send Data to Base station

$CH(N_i)$ sends data to Base station

Repeat the steps 12 to 18 for different rounds

End of algorithm

T he sensor field is divided into equal sub-region as shown in figure 6.1 and 6.2 for the 100 and 200 nodes respectively for the I-LEACH protocol simulation.

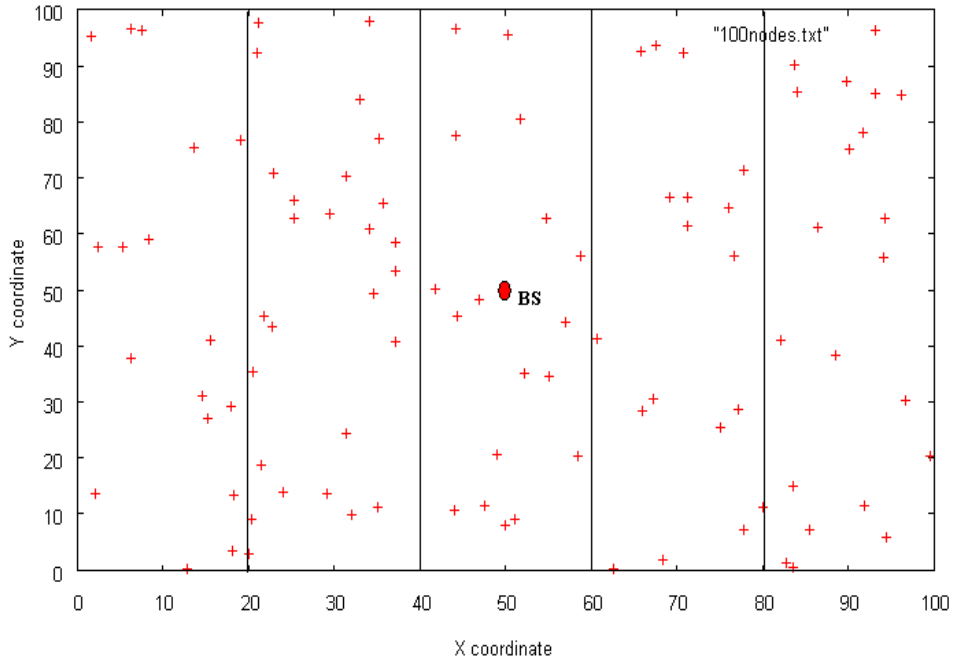


Figure 6.1: Sensor network topology for 100 nodes with base station at (50,50)

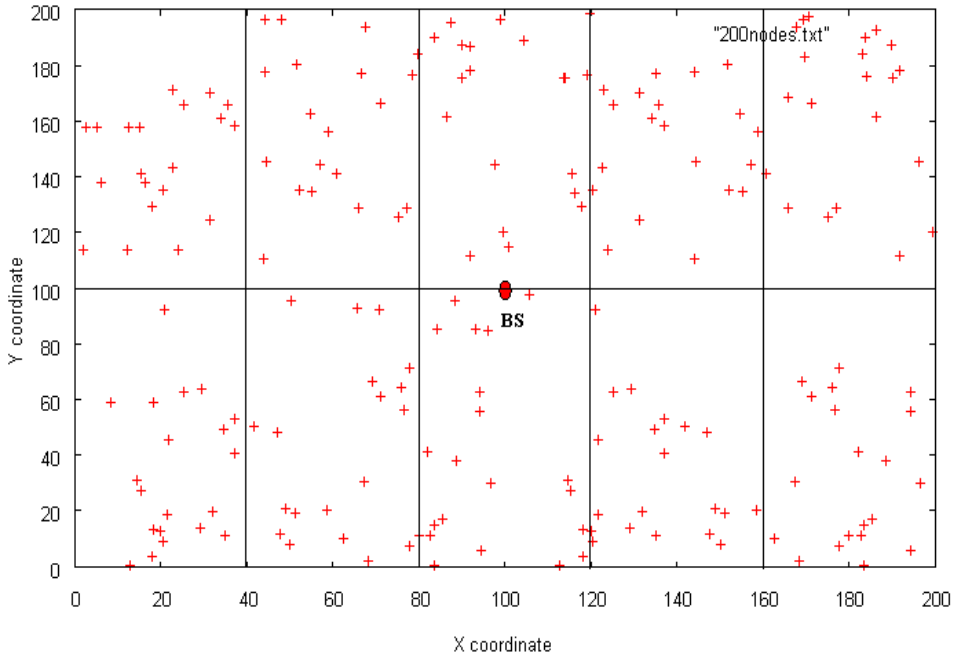


Figure 6.2: Sensor network topology for 200 nodes with base station at (100,100)

6.2 Simulation Results and Analysis

The performance of the proposed I-LEACH is compared with basic LEACH protocol in terms of Average energy consumption, Life time of the network and Average throughput. All experiment results presented in this section are average of three simulation runs in 100 and 200 nodes network size. The following table shows the simulation results at various simulation runs.

Table 6.1: Simulation Results

Network Size		**LEACH**		**I-LEACH**	
		Life Time(s)	**Throughput**	**Life Time(s)**	**Throughput**
100	1	372.30	35905	566.90	51106
	2	404.60	27889	603.50	50517
	3	433.80	44871	570.20	50563
200	1	417.90	31987	480.60	40824
	2	342.50	22166	500.80	37865
	3	250.49	17011	480.10	37644

6.2.1 Average Energy consumption

As simulation started with equal amount of energy (2J) with each sensor nodes, so total energy with the network will be 200J for 100 nodes and 400J for 200 nodes simulation. Figure 6.3 and 6.4 shows the comparison of average energy consumption at various time between LEACH and I-LEACH protocols for 100 and 200 nodes respectively.

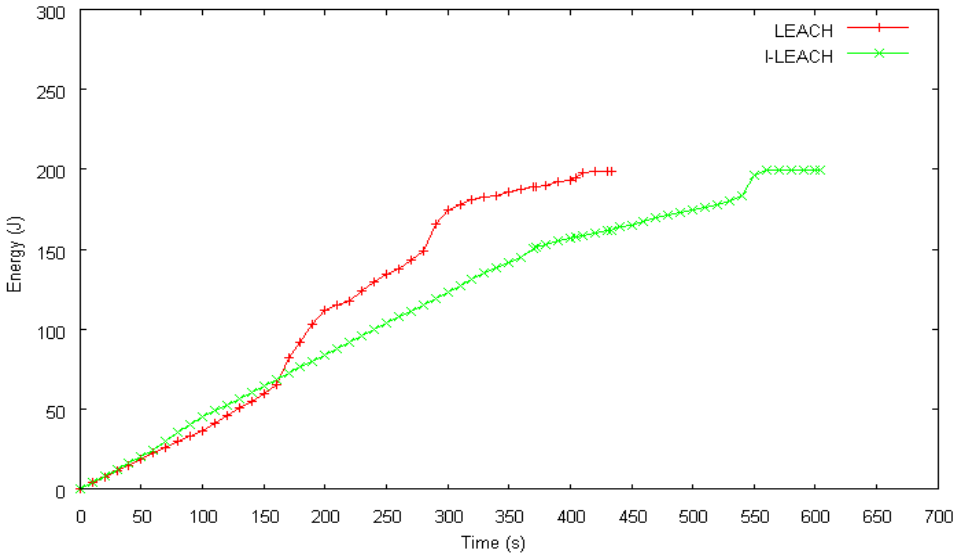


Figure 6.3: Average energy consumption comparison(100nodes).

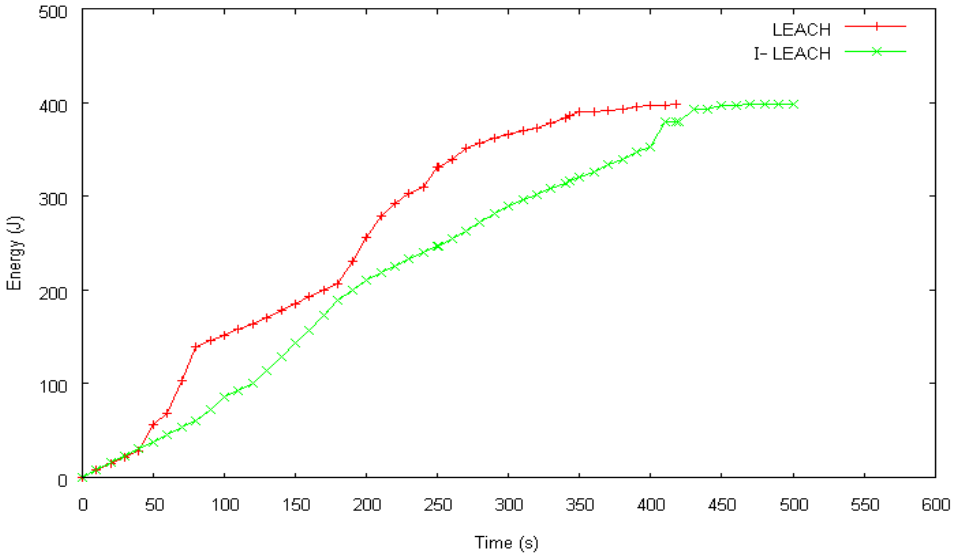


Figure 6.4: Average energy consumption comparison(200nodes).

6.2.2 Life time of the network

The total number of nodes which are alive at the end of each rounds is shown in figure 6.5 and 6.6 for the 100 and 200 nodes network respectively. The simulation will stop if total number of live nodes is less than five in the case of 100 nodes network while total number of live nodes is less than ten in the case of 200 nodes network.

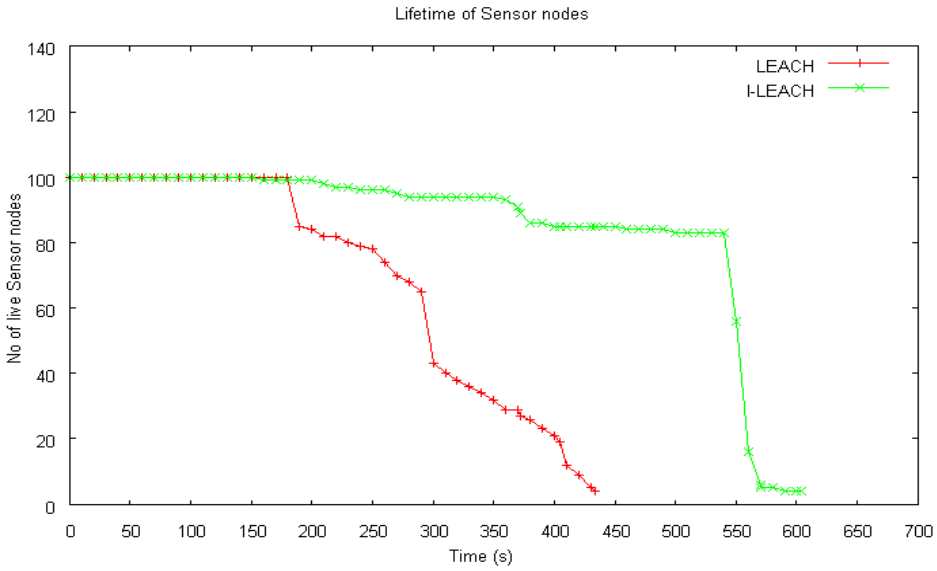


Figure 6.5: Life time comparison(100 nodes).

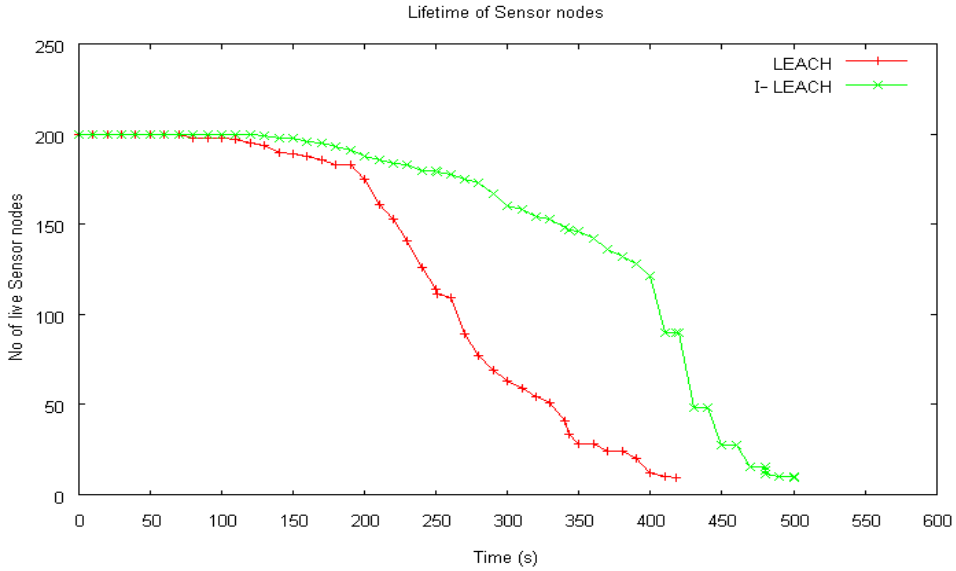


Figure 6.6: Life time comparison(200 nodes).

6.2.3 Average throughput

It will measure the average number of packets reaching at the sink (base station) node. The location of the sink is (50,50) in the 100 nodes network while (100,100) in the case of 200 nodes network. Figure 6.7 and 6.8 shows throughput achieved in the both cases respectively.

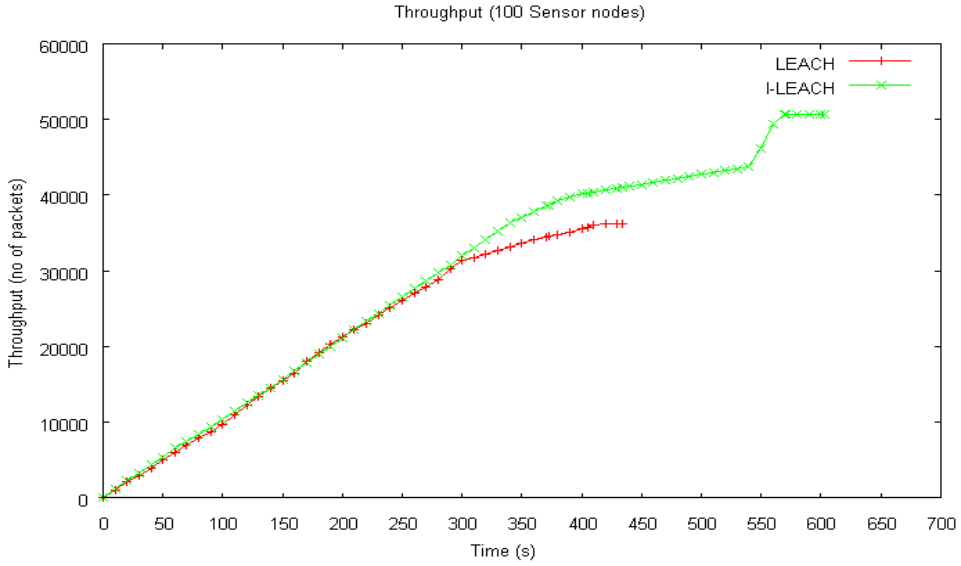


Figure 6.7: Throughput comparison(100 nodes).

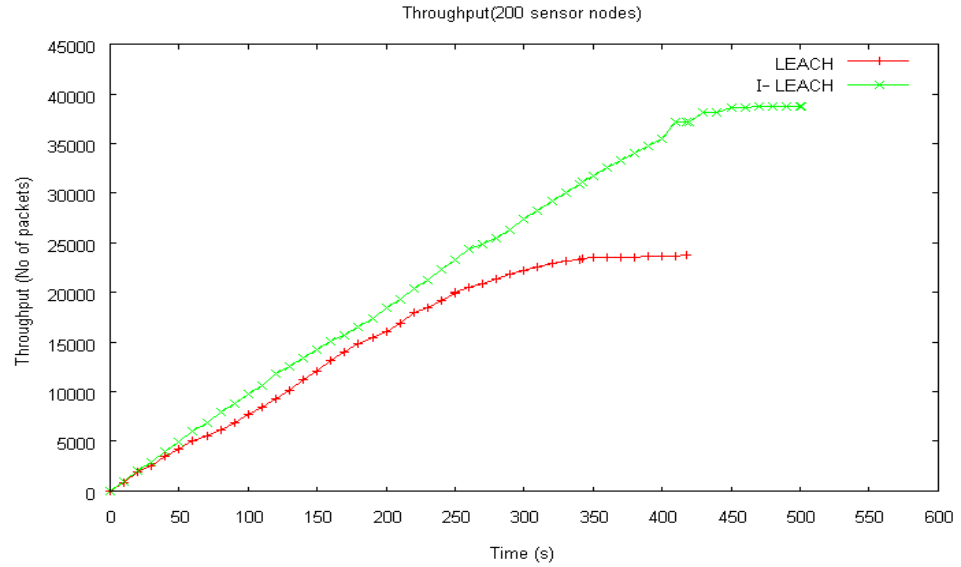


Figure 6.8: Throughput comparison(200 nodes).

6.3 Summary

The proposed I-LEACH ensure that the elected cluster-heads are uniformly distributed over the network. Hence, there is no possibility that all cluster-heads will be concentrated in one part of the network. The result of simulations conducted indicates that the proposed clustering approach is more energy efficient and scalable and hence effective in prolonging the network life time compared to LEACH. It also outperforms LEACH with respect to throughput of the network. I-LEACH improves energy consumption by around 43% and throughput by 40% in 100 nodes network size while improves energy consumption by around 44% and throughput by 63% in 200 nodes network size.

Chapter 7

Conclusion and Future Work

7.1 Conclusion

The core operation of a WSN is to gather and convey the collected data to a distant BS for further processing and analysis. Gathering information from a WSN in an energy effective manner is of paramount importance in order to prolong its life span. This calls for use of an appropriate routing protocol to ensure efficient data transmission through the network. In this research project, The basic cluster based LEACH routing protocol is improved by suggesting distributed cluster formation approach. The result of simulations conducted indicates that the proposed clustering approach is more energy efficient and hence effective in prolonging the network life time compared to LEACH.

7.2 Accomplishments

1. Implemented Flat routing algorithms in TOSSIM.
2. Implemented LEACH and I-LEACH algorithms in NS-2.
3. Programmed using nesC and tool command language.
4. Learnt to simulate a network using TOSSIM and NS-2.

5. Performed NS-2 simulations in a LINUX platform. This gave me an opportunity to learn LINUX commands

6. Benefited from the experience of research and writing the project documentation. This sharpened research and report writing skills.

7. Acquired research techniques which were the core of this project.

7.3 Future Work

Our Implementation performs better compared to the basic LEACH protocol, still there are certain optimizations possible in the protocol like Mobility consideration. In the present implementation the base station is considered to be static but in real deployment of the sensor nodes the base station could be moving also. The other consideration is the size of the network. As the network grows the location of the base station might be an issue, the cluster head may be out of the coverage of base station. In such scenario the network architecture has to be modified to multilevel clustering so every cluster head comes directly or indirectly under the communication range of base station. The next improvement possible is at the MAC layer assumptions. We have considered 802.11 for communication requirements which can be replaced with the 802.15.4(ZigBee) communications as the later has several advantages over the former. The ZigBee standard enjoys low power requirements, low latency, high reliability and larger range because of mesh networking.

Appendix A

List of websites

1. http://ceng.usc.edu/ anrg/SensorNetBib.html
2. http://www.ece.rochester.edu/ wheinzel/research.html
3. http://cs.acadiau.ca/ shussain/wsn/publications/index.html
4. http://www.mail-archive.com/tinyos-help@millennium.berkeley.edu/msg19389.html
5. http://www.tinyos.net/tinyos-1.x/doc/html/install-tinyos.html
6. http://www.tinyos.net/scoop
7. http://www.eecs.harvard.edu/ shnayder/ptossim/install.html
8. http://www.isi.edu/nsnam/dist/ns-allinone-2.27.tar.gz
9. http://www.internetworkflow.com/downloads/ns2leach/mit.tar.gz
10. http://www.mail-archive.com/ns-users@isi.edu

Appendix B

List of publication

1. Paper titled **"Survey on Energy Efficient Hierarchical Routing Algorithms for Sensor Networks"** published in '4th National Conference on Current trends in Technology' - NUCONE'09, Organized by Institute of Technology, Nirma University, Ahmedabad, India, held during 25-27 November, 2009.

2. Paper titled **"Performance Analysis of Hierarchical Routing Protocol (LEACH) for Wireless Sensor Networks"** published in 'National Conference on Advances in Wireless Communications' - NCAWC-2010, Organized by SSG College of Engineering, Shegaon, Maharashtra, India, held during 07-08 May, 2010.

References

[1] Mohammad Ilyas and Imad Mahgoub."Handbook of sensor networks : compact wireless and wired sensing systems", *Book, Chapter 1 & 6 CRC Press,* 2005.

[2] Kemal Akkaya, Mohamed Younis, "A survey on routing protocols for wireless sensor networks",*in the Elsevier Ad Hoc Network Journal, Vol. 3/3 pp. 325-349,* July, 2005.

[3] Jamal N. Al-Karaki,Ahmed E. Kamal,. "Routing Techniques In Wireless Sensor Networks: A Survey",*IEEE Wireless Communications,* December 2004.

[4] D. J. Dechene, A. El Jardali, M. Luccini, and A. Sauer, "A Survey of Clustering Algorithms for Wireless Sensor Networks",*Computer Communications, Butterworth-Heinemann Newton, MA USA,* October 2007.

[5] G.Santhosh Kumar, Lino Abraham Varghese, Jose Mathew K and K. Poulose Jacob, "Evaluation of the Power Consumption of Routing Protocols for Wireless Sensor Networks", *Ad Hoc and Ubiquitous Computing, ISAUHC '06. International Symposium IEEE,* 2006.

[6] W. R. Heinzelman, A. Chandrakasan, and H. Balakrishnan, "Energy Efficient Communication Protocol for Wireless Micro Sensor Networks," *Proceedings of the 33rd International Conference on System Sciences (HICSS00).* Jan 2000.

[7] S. Lindsey and C. Raghavendra, "PEGASIS: Power-Efficient Gathering in Sensor Information Systems," *IEEE Aerospace Conf. Proceding, vol. 3, 9-16, pp. 1125-30,* 2002.

[8] A. Manjeshwar, D.P. Agrawal, "TEEN: a protocol for enhanced efficiency in wireless sensor networks", *in: Proceedings of the 1st International Workshop on Parallel and Distributed Computing Issues in Wireless Networks and Mobile Computing, San Francisco, CA, April,* 2001.

[9] A. Manjeshwar, D.P. Agrawal, "APTEEN: a hybrid protocol for efficient routing and comprehensive information retrieval in wireless sensor networks", *in: Proceedings of the 2nd International Workshop on Parallel and Distributed Computing Issues in Wireless Networks and Mobile computing, Ft. Lauderdale, FL,*April 2002.

[10] Younis O., Fahmy S. "HEED: A Hybrid, Energy-Efficient, Distributed Clustering Approach for Ad Hoc Sensor Networks." *IEEE Transactions on Mobile Computing, 3(4), 660669.,* 2004

[11] L. Yu, N. Wang, W. Zhang, and C. Zheng, "GROUP: A Grid-Clustering Routing Protocol for Wireless Sensor Networks", *East China Normal University., IEEE ,2006.*

[12] Fan Xiangning, Song Yulin. "Improvement on LEACH Protocol of Wireless Sensor Network, *2007.*

[13] V. Loscr, G. Morabito and S. Marano. "A Two-Levels Hierarchy for Low-Energy Adaptive Clustering Hierarchy.

[14] Dissertation, Hang Zhou, Zhe Jiang and Mo Xiaoyan, "Study and Design on Cluster Routing Protocols of Wireless Sensor Networks, *2006.*

[15] W. B. Heinzelman et al., "An Application-Specific Protocol Architecture for Wireless Microsensor Networks, *PhD Thesis, MIT 2000.*

[16] M. Bani Yassein, A. Al-zoubi, Y. Khamayseh, W. Mardini, "Improvement on LEACH Protocol of Wireless Sensor Network (VLEACH), *International Journal of Digital Content Technology and its Applications Volume 3, Number 2, June 2009*

[17] Philip Levis, Nelson Lee, Matt Welsh, and David Culler, "TOSSIM: Accurate and Scalable Simulation of Entire TinyOS Applications." *in Proceedings of SenSys03, First ACM Conference on Embedded Networked Sensor Systems,* 2003.

[18] Victor Shnayder, Mark Hempstead, Bor-rong Chen, Geoff Werner, Matt Welsh, "Simulation the Power Consumption of Large-Scale Sensor Network Applications", *in Proceedings of SenSys'04, Second ACM Conference on Embedded Networked Sensor Systems,* 2004.

[19] Deved Gay, Philip Levis, Robert von Behren, "The nesC Language: A Holistic Approach to Networked Embedded Systems", *In Proceeding of Programming Language Design and Implementation (PLDI),* June 2003.

[20] Thammakit Sriporamanont,Gu Liming, "Wireless Sensor Network Simulator", *A technical report, IDE0602, January 2006.*

[21] Wendi Heinzelman, Anantha Chandrakasan and Hari Balakrishnan, "The MIT uAMPS ns Code Extensions", Massachusetts Institute of Technology Cambridge, MA02139, *A technical report, Jane 2000.*