

Analysis of Low Energy Adaptive Clustering Hierarchy (LEACH) protocol

Project Report submitted in partial fulfillment of the requirement for the degree of
Bachelor of Technology.

in

Information Technology

under the Supervision of

Dr. Pooja Jain

By

SAURABH GOEL

111438

to



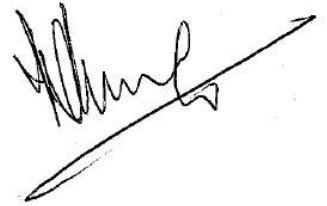
Jaypee University of Information and Technology

Waknaghat, Solan – 173234, Himachal Pradesh

Certificate

This is to certify that project report entitled “Analysis of Low Energy Adaptive Clustering Hierarchy (LEACH) protocol”, submitted by Saurabh Goel in partial fulfillment for the award of degree of Bachelor of Technology in Information Technology to Jaypee University of Information Technology, Waknaghat, Solan has been carried out under my supervision.

This work has not been submitted partially or fully to any other University or Institute for the award of this or any other degree or diploma.



Date: 22/12/2014

Dr. Pooja Jain

Assistant Professor(Senior Grade)

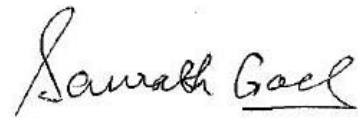
Department of CSE/IT

Acknowledgement

I would like to thank my guide and mentor, Dr. Pooja Jain, for guiding me in my path to greater knowledge. She has always helped me to understand hard concepts in a simple manner. She has immensely helped me through this project by giving me her expert guidance.

I would also like to thank all of my colleagues for helping me in my studies and all the other activities.

Lastly, I would like to thank my parents for supporting me throughout my studies and for always being my support pillar.

A handwritten signature in black ink that reads "Saurabh Goel". The signature is written in a cursive style with a prominent loop at the beginning of the first name.

Date:05/08/2015

Saurabh Goel

CONTENTS

1. INTRODUCTION.....	1
1.1 Abstract.....	1
1.1 Purpose	1
1.2 Background	14
2. PROJECT CHARACTERISTICS.....	15
2.1 Functional Requirements.....	15
2.2 Project Characteristics	15
2.21 LEACH Network Model	15
2.22 Sensor Models	16
2.23 Simulation Data.....	18
2.3 Simplifying Assumptions	18
3. PROJECT DESIGN	19
3.1 Design Flow	19
3.2 Design Timeline.....	21
4. IMPLEMENTATION	21
4.1 Design Drift	22
4.2 Redefined Requirements	22
5. CONCLUSION9	
5.1 Results.....	23
5.2 Conclusion.....	25
5.3 Future Directions	28
6. APPENDIX 12	

6.1 Glossary..... 35

7. REFERENCES 12

Chapter 1

1. Introduction

1.1 Abstract

The study and implementation of wireless sensor networks is an emerging field in computer science, with applications in fields from environmental monitoring to national defense. These networks consist of individual sensor nodes which typically collect and transmit data to a central server. Due to the distributed nature of these networks, energy consumption is one of the primary determinants in network efficiency. Algorithms that optimize this factor are an area of intensive research and the subject of this project.

The goal of this project is to evaluate one such algorithm using the open source network simulator ns2. LEACH, or Low Energy Adaptive Clustering Hierarchy, consists of a percentage of randomly chosen nodes to act as cluster heads. Once the cluster heads have been determined, they broadcast a signal notifying other nodes within range. The nodes can then calculate their optimal cluster head. These heads act as a middle layer between individual nodes and the server, relaying data and optimizing packet size. By definition, the LEACH model ensures that only two hops separate the node from the database. A network consisting of 100 randomly deployed nodes implementing the LEACH model will be used to test programs for node management.

The simulation data obtained by implementing these algorithms using ns2 will depict which is the most efficient in terms of energy consumption. These data will give researchers a better understanding of which models produce particular results, thus facilitating the goal of this project.

1.1 Purpose

The purpose of this project is to evaluate various programs for wireless sensor network management and to determine a model which optimizes power consumption.

1.2 Background

Research into the design and implementation of wireless sensor networks is a relatively recent addition to the computer sciences. Made possible by advances in the miniaturization of computer hardware and battery longevity, these networks consist of nodes capable of collecting information from their environment and transmitting this information to a centralized data collection point. Applications range from meteorology to manufacturing and national defense. Because they are composed of wireless, typically battery-powered devices, the issue of power consumption is the primary determinant of network usefulness. Programs for network management must find the optimal design to increase the overall lifetime of the wireless sensor network.

Routing protocols in WSNs

A WSN can have network structure based or protocol operation based routing protocol. Routing protocols in WSNs might differ depending on the application (Protocol-Operation-based) and network architecture (Network-Structure-based).

Depending on protocol operation WSN can be classified into, *Multipath-based routing*

It uses multiple paths rather than a single path in order to enhance network performance. For instance the fault tolerance can be increased by maintaining multiple paths between the source and destination at the expense of increased energy consumption and traffic generation.

Query-based routing

The destination nodes propagate a query for data from a node through the network. A node with this data sends the data that matches the query back to the node that initiated it.

Negotiation-based routing

This negotiation based routing is done in order to eliminate redundant data transmissions. In this communication decisions are also made based on the resources available in the network scenario.

QOS-based routing

When delivering process of data in ongoing with the help of this routing, balances the network in between energy consumption and data quality through certain QOS metrics such as delay, energy or bandwidth.

Coherent-based routing

The entity of local data processing on the nodes is being distinguished between the coherent (minimum processing) and the non-coherent (full processing) routing protocols.

Depending on the network structure Routing Protocols can also be classified into,

Flat-based routing:

In this routing protocol each node plays the same role and sensor nodes collaborate to perform the sensing task.

Hierarchical-based routing:

In this type of routing, the nodes having the higher-energy are used to process and send the information, while the nodes having the low-energy are used to perform the sensing in the proximity of the target. The process of creation of clusters and assigning special tasks to cluster heads can efficiently increase the overall system scalability, lifetime, and energy efficiency. Hierarchical routing is an efficient way to lower the energy consumption within a cluster with the help of performing data aggregation and fusion within the different clusters in order to decrease the number of transmitted messages to the sink node.

Location-based routing:

In this type of protocol sensor nodes are addressed by means of their locations. The distance between neighbouring nodes can be estimated on the basis of incoming signal strengths from the source nodes. Relative coordinates of neighbouring nodes can be obtained by exchanging such information between neighbours or by communicating with a satellite using GPS. To save energy, some location-based schemes also

suggest that nodes should go to sleep if there is no activity to perform in a definite time.

Hierarchical Routing Protocols

Among the issues in WSN the consumption of energy is one of the most important issues. Traditional routing protocols for WSN may not be optimal in terms of energy consumption [paper Energy]. Hierarchical routing protocols are found to be more energy efficient than other protocols. Hierarchical routing follows the clustering mechanisms. Clustering techniques can be efficient in terms of energy and scalability. By the use of a clustering technique they minimize the consumption of energy greatly in collecting and disseminating data. This is neither but the process of fusion and aggregation process. Hierarchical routing protocols minimize energy consumption by dividing nodes into different clusters. In each cluster, higher energy nodes i.e. the cluster head (CHs) can be used to process and send the information to the base station while low energy nodes i.e. the cluster members can be used to perform the sensing in the proximity of the target and send to its cluster head. This means that creation of clusters and assigning special tasks to cluster heads can greatly contribute to overall system scalability, lifetime, and energy efficiency, reduces the size of the routing table by localizing the route setup within the clusters, and conserves communication bandwidth of network.

LEACH (Low-Energy Adaptive Clustering Hierarchy)

LEACH is a cluster-based wireless sensor networking protocol. LEACH adapts the clustering concept to distribute the energy among the sensor nodes in the network. LEACH improves the energy-efficiency of wireless sensor networking beyond the normal clustering architecture. As a result, we can extend the life time of our network, and this is the very important issue that is considered in the wireless sensor networking field.

In LEACH protocol, wireless sensor networking nodes divide themselves to be many local clusters. In each local cluster, there is one node that acts as the base station (or we can call it “cluster-head”). Hence, every node in that local cluster will send the data to the cluster-head in each local cluster. The important technique that makes LEACH be different from the normal cluster architecture (the drain the nodes battery very quickly) is that LEACH uses the randomize technique to select the cluster-head depending on the energy left of the node.

After cluster-head is selected with some probability, the cluster-heads in each local cluster will broadcast their status to the sensor nodes in their local range by using CSMA MAC protocol. Each sensor node will choose a cluster-head that is closest to itself to join that cluster because each sensor node will try to spend the minimum communication energy with its cluster head.

After the clustering phase is set up, each cluster-head will make a schedule for the nodes in its cluster. In paper LEACH, TDMA is used. For more efficiency, each sensor node could turn-off waiting for their allocated transmission.

Cluster-heads will collect the data from the nodes in its cluster, and compresses that data before transmits the data to the base station. By following this protocol, the base station will get the data from all sensor nodes that we are interested, and ready for the end-user to access the data.

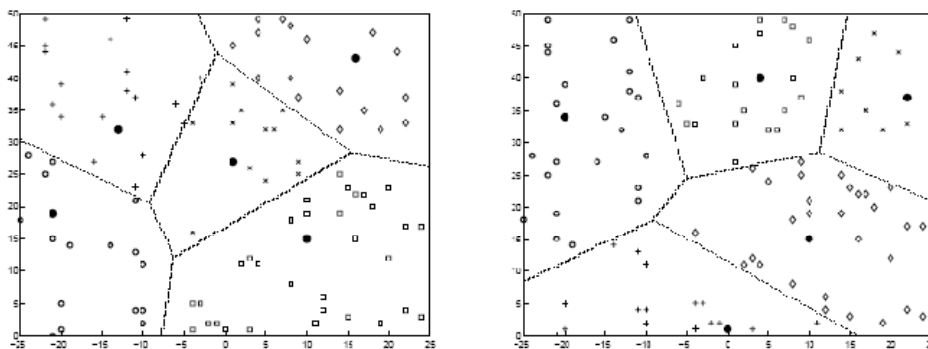


Figure. Dynamic clusters: Cluster-head nodes at time t_1 and $t_1 + d$

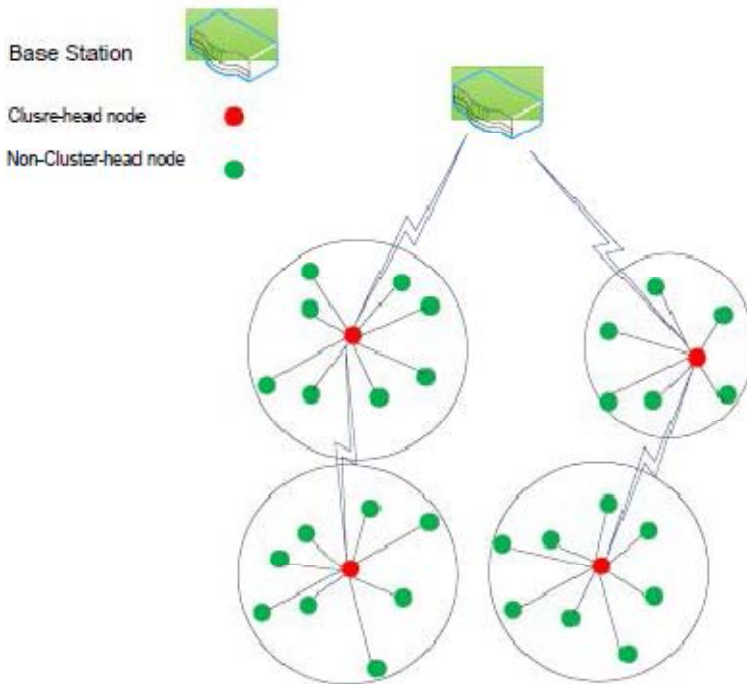
It shows that the cluster-heads of each local cluster are not fixed. At time t_1 , a set C of nodes might be the cluster-heads, and after that, at time $t_1 + d$, a set C' might be the cluster-heads. This is because we want to spread the energy dissipation among all of sensor nodes.

LEACH-C (Centralized Low Energy Adaptive Clustering Hierarchy)

It involves a centralized clustering algorithm. The steady state will remain the same whereas the setup phase of the Leach C contains each node sending information about the current location and also the energy level to the base station. The base station thus by utilizing the global information of the network produces better clusters that require less energy for data transmission. It needs GPS or the other location tracking method. The base station has to make sure that only nodes with enough energy are allowed to participate in the selection of the cluster head. The base station then broadcasts the information to all nodes in the network. Leach-C has a deterministic threshold algorithm which takes into account the amount of energy in the node and/or whether or not the node was recently a cluster head. The number of cluster head nodes and its placement cannot be guaranteed. The central control algorithm can be used to form the clusters which may produce better clusters through the distribution of the cluster head nodes throughout the network.

MULTIHOP LEACH

The distance between the cluster head and the base station is increased enormously when the network diameter is increased beyond a certain level in which the scenario is not suitable for Leach routing protocol.



The energy efficiency of the protocol can be increased by using multi-hop communication within the cluster. Multi hop-Leach is a complete distributed clustering based routing protocol. The multi hop approach is utilized inside the cluster and outside the cluster.

LEACH-F (Fixed no. of clusters Low Energy Adaptive Clustering Hierarchy)

In Leach-F, once the clusters are formed they are fixed and there is no setup overhead at the beginning of each round. It uses the same centralized cluster formation algorithm as Leach-C for deciding the clusters. In Leach-F, new nodes cannot be added to the system and do not adjust their behaviour based on nodes dying. Furthermore, the node mobility cannot be handled by the Leach-F. Only the cluster head position is rotated among the nodes within the cluster. Leach-F may or may not be provided energy saving. A stable cluster and rotating cluster head concept is used by Leach-F

in which cluster once formed is maintained stable throughout the network lifetime in order to avoid re-clustering.

LEACH-L (Energy Balanced Low Energy Adaptive Clustering Hierarchy)

Leach-L is an advanced multi hop routing protocol and considers only the distance. It is suitable for large scope wireless sensor network and the optimum hop counts are deduced. The cluster heads can communicate directly to the base station when they are located close to it. When they are located far away from the base station, they can communicate by the method of multi-hop way and the shortest transmission distance is limited. In this, the sensors are allowed to use different frequencies and gaps to communicate with base station. The clusters are re-established in each round consisting of the setup and steady state phase. And in each round new cluster heads are elected and the load is distributed and balanced among the nodes in the network. Since Leach-L makes power equally distribute among all sensors, in the pre-period, the network's activity nodes and cover areas of Leach-L are greatly larger than that of Leach-M.

LEACH-E(Energy Low Energy Adaptive Clustering Hierarchy)

LEACH-E is the enhancement of LEACH. It involves a cluster head selection algorithm which have non-uniform starting energy level among the sensors having global information about the other sensors. In order to minimize the total energy consumption, the required number of cluster heads

has to scale as the square root of the total number of sensor nodes and this can be determined by Leach-E. By making the residual energy of the sensor node as the main factor, it decides whether the sensor nodes turn into the cluster head or not in the next-round.

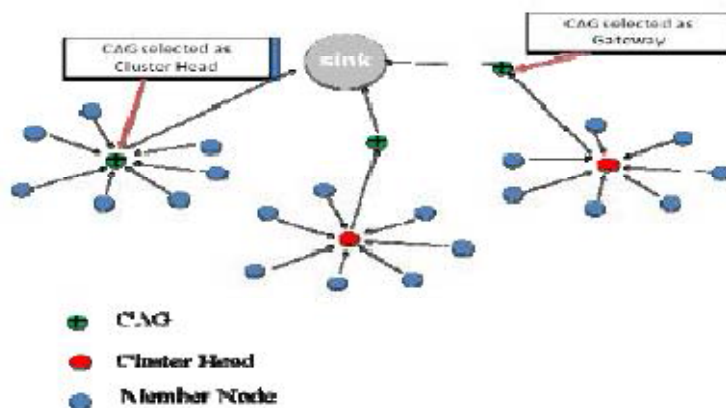
LEACH-B (Balanced Low Energy Adaptive Clustering Hierarchy)

Leach-B uses the decentralised algorithms of cluster formation where each sensor node only knows about its own position and the final receiver and does not know about the position of all the sensor nodes. Leach-B involves the following techniques:

Cluster head selection algorithm, Cluster formation and data transmission with multiple access. By evaluating the energy dissipated in the path between final receiver and itself, each of the sensor node chooses its cluster head. Efficiency of Leach-B is better than Leach.

LEACH-A (Advanced Low Energy Adaptive Clustering Hierarchy)

In Leach protocol the head node consumes more energy than others. Hence the energy saving and reliable data transfer is improved LEACH-A. In this, the data is processed using mobile agent technique based on Leach. Advanced Leach, a heterogeneous energy protocol is proposed for the purpose of decreasing the node's failure probability and for prolonging the time interval before the death of the first node which can be referred to as stability period.



By using a synchronized clock, each sensor knows the starting of each round. Let n be the total number of nodes and m be the fraction of n that are equipped with a time more energy than others. These nodes are called CAG nodes, the nodes selected as cluster heads or gateways and the rest $(1-m) * n$ as the normal nodes. The CAG nodes will become the cluster head for the data aggregation and transmit to the sink. The Leach-A protocol offers the following advantages

1. The merging of the data is done to reduce the amount of information that are transmitted to the base station.
2. More energy can be saved by using TDMA/CDMA techniques that allows hierarchy and makes clustering on several levels.
3. The CAG nodes continues to send data to the sink when all normal nodes death.

LEACH-M (Mobile - Low Energy Adaptive Clustering Hierarchy)

Mobility support is an important issue in Leach routing protocol. Leach-M is proposed to mitigate this issue. Leach-M involves the mobility of non cluster head nodes and cluster

head during the setup and steady state phase. The nodes in Leach-M are assumed to be homogeneous and have their location information through GPS. The cluster head can be selected based on the minimum mobility and lowest attenuation mode. The selected cluster heads then broadcasts their status to all nodes in transmission range.

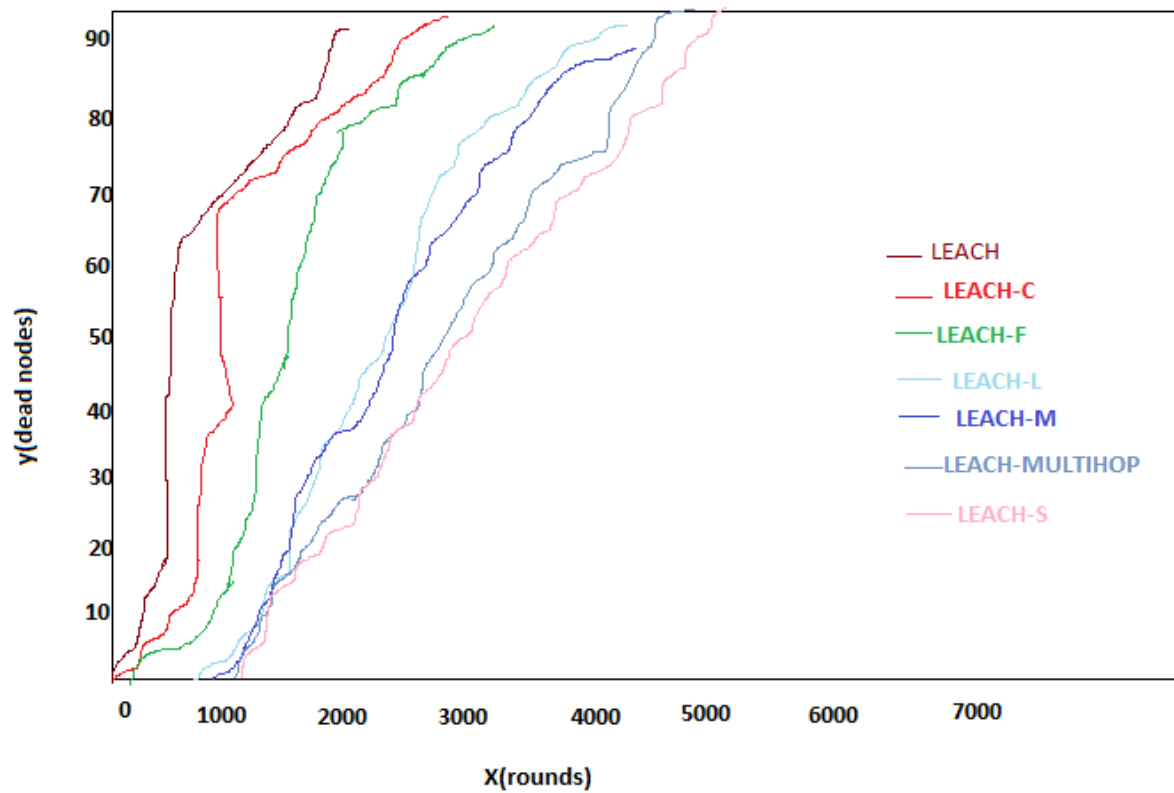
LEACH-S (SOLAR-AWARE CENTRALIZED LEACH)

In Leach-S, the base station selects the cluster head with the help of improved central control algorithm. Base station select solar powered nodes having maximum residual energy. In Leach-S, the solar status is transmitted by the nodes to the base station along with the energy and the nodes with higher energy are selected as the cluster head. When the number of solar-aware nodes is increased, the performance of sensor network is also increased. The sun duration increases the lifetime of the sensor network. The cluster head handover takes place if the sun duration is smaller.

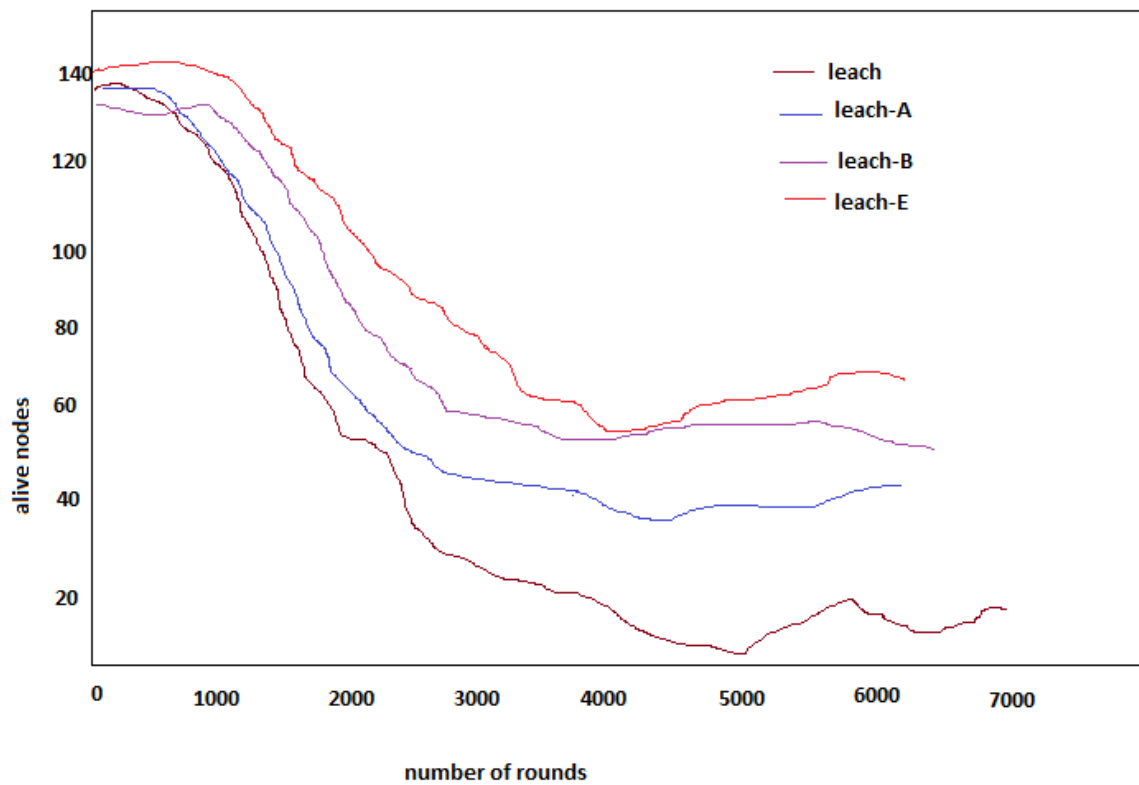
COMPARISON OF LEACH AND ITS IMPROVED VERSIONS

A number of protocols that are the enhanced version of the conventional LEACH routing protocol have been compared. All these protocols showed better performance than the conventional LEACH routing protocol.

SIMULATION RESULTS (DEAD NODES)



SIMULATION RESULTS (ALIVE NODES)



Simulation Results

According to results, Solar aware has better network lifetime than others as they have the ability to re-energise themselves. From the above analysis, it has been shown that all the descendants of the leach protocol are proved to be better than the leach and they overcome the drawbacks and issues related to leach which is the hierarchical clustering routing protocol.

PERFORMANCE COMPARISON OF DIFFERENT LEACH PROTOCOLS

Clustering Routing protocol	Classification	mobility	Scalability	Self organization	Randomized Rotation	Distributed	Centralised	Hop count	Energy efficiency	homogeneous	Use of location information	Data aggregation
LEACH	Hierarchical	Fixed BS	Limited	Yes	Yes	Yes	No	Single hop	High	Yes	No	Yes
LEACH-S	Hierarchical	Fixed BS	Good	Yes	Yes	Yes	Yes	Single hop	Very high	Yes	No	Yes
Multi-hop LEACH	Hierarchical	Fixed BS	Very good	Yes	Yes	Yes	No	Multi hop	Very high	Yes	Yes	Yes
LEACH-M	Hierarchical	Mobile BS and nodes	Very good	Yes	Yes	Yes	No	Single hop	Very high	Yes	Yes	Yes
LEACH-A	Hierarchical	Fixed BS	good	Yes	Yes	Yes	No	Single hop	Very high	No	No	Yes
LEACH-C	Hierarchical	Fixed BS	good	Yes	Yes	No	Yes	Single hop	Very high	Yes	Yes	Yes
LEACH-B	Hierarchical	Fixed BS	good	Yes	Yes	Yes	No	Single hop	Very high	Yes	Yes	Yes
LEACH-F	Hierarchical	Fixed BS	limited	No	Yes	No	Yes	Single hop	Very high	Yes	Yes	Yes
LEACH-L	Hierarchical	Fixed BS	Very good	Yes	Yes	Yes	No	Multi hop	Very high	Yes	Yes	Yes
LEACH-E	Hierarchical	Fixed BS	Very good	Yes	Yes	Yes	No	Single hop	Very high	No	Yes	Yes

Chapter 2

2. Project Characteristics

2.1 Functional Requirements

We will be implementing a network simulation using NS-2 in a linux environment. The simulation must follow the LEACH network model discussed in section 2.21. The simulation includes implementations of three specific sensor models as discussed in 2.22. The program will then write the data from the simulation to a file in order to be graphically displayed by another program. The graph must show overall energy of the network versus time in order to analyze which model is more energy efficient.

2.2 Project Characteristics

2.21 LEACH Network Model

The overall network model will be that of LEACH. It consists of a percentage of randomly chosen nodes to act as cluster heads (which are the green nodes in figure 2.1). Once the cluster heads have been determined, they broadcast a signal notifying other nodes (which are the aqua nodes in figure 2.1) within range. The nodes can then calculate their optimal cluster head, forming the regions displayed in figure 2.1. The cluster heads act as a middle layer between individual nodes and the server, relaying data and optimizing packet size. By definition, the LEACH model ensures that only two hops separate each node from the database. Although this is the model for the network itself, the research of this project is not focused on network topology but the sensor models specifically. The individual nodes in the network will be implemented using one of three different models.

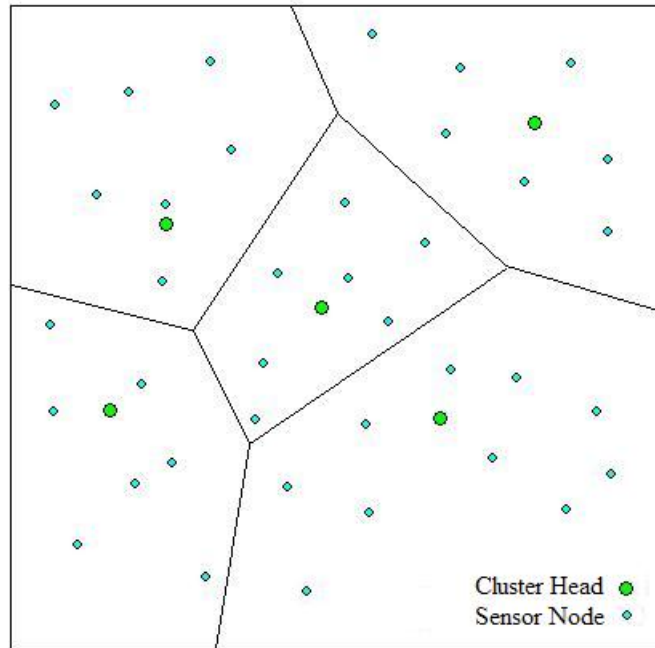


Figure 2.1: LEACH Diagram

2.22 Sensor Models

The focus of the research for this project is the sensor models. The three models discussed in this section are Active-Listening, Active-Sleeping, and Active-Listening-Sleeping which were developed by Qi Han as potential sensor models using the LEACH network model.

In the AL or Active-Listening model, the sensor is initially in the “listening” state which is where the sensor is waiting for either a source-initiated update or a consumer-initiated update. A source-initiated update occurs if the sensor value falls out of the designated range, whereas a consumer-initiated update occurs if the server requests the sensor value. Once the sensor is required to process one of these updates, it is transitioned into the “active” state where it will then transmit the data to its cluster head or to the server, directly, if the node is a cluster head itself. After transmitting, the sensor will transition back into the “listening” state. Figure 2.2 shows the sensor state diagram.

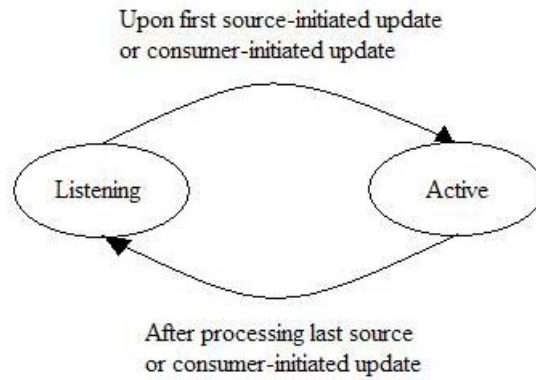


Figure 2.2: The Active-Listening Model (AL)

The AS or Active-Sleeping model is initially in the “sleeping” state. Here, the sensor is not receiving at all, which means that it cannot receive consumer-initiated updates. The cluster head, however, is never asleep and will queue the consumer-initiated updates until the target node is no longer sleeping. The two ways the sensor shifts to the “active” state are by either a source-initiated update or by a certain time T_s that the node has been “sleeping.” While “active” the node transmits the data and can also perform consumer-initiated updates. After processing the last source or consumer-initiated update, the sensor toggles back into the “sleeping” state as shown in figure 2.3.

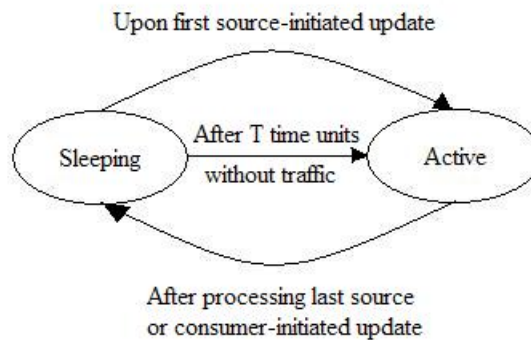


Figure 2.3: The Active-Sleeping Model (AS)

The final sensor model to be simulated is the ALS or Active-Listening Sleeping model. This model is a hybrid of the AS and AL models because it combines all three states into one design. Just like the AS model, this model is initially in the “sleeping” state. It only toggles to the “active” state if a source-initiated update has occurred or an arbitrary time T_s has passed. Once the sensor is finished transmitting data, it transitions in the “listening” state where, just like the AL model, the sensor can only go back to “active” upon a source or consumer-initiated update. If the node has been in the “listening” state for T_a time units, it will switch back to the “sleeping” state. The amount of time that a node sits in the “listening” state (T_a) will converge on the optimal time as the simulation progresses since this time is dependent on the node activity pattern. The ALS model is shown below in figure 2.4.

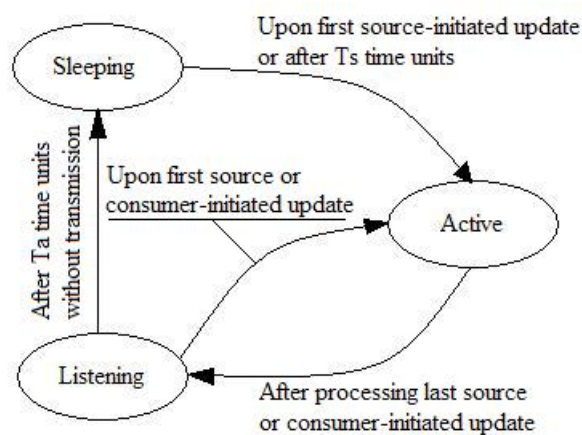


Figure 2.4: The Active-Listening-Sleeping Model (ALS)

2.23 Simulation Data

There will be two kinds of data, both random and real, used to test the three sensor models described above. As described by Qi Han, the random data will simulate temperature. The sensor nodes in the network will initialize their values by randomly and uniformly picking a value from the range [1, 100] in order to approximate temperature readings in Fahrenheit. These values perform a random walk in one dimension: every second, the values either increases or decreases by an amount sampled uniformly from the range [0, 5]. For example, a node in the network might be initialized with the value “24.” At the next time step (second) in the simulation the value might change to “25” or “22” depending on the second

random value chosen. Then the value will continue to change as the simulation runs, always within 5 of the previous value.

The real data used will be pulled from the NOAA website. This is the real-time data from moored ocean buoys. The measurements include surface winds, sea surface temperature, upper ocean temperature and currents, air temperature, and relative humidity. Samples are taken every 10 minutes.

2.3 Simplifying Assumptions

Several factors are not being included in this simulation. The first is that we are not taking into account what would happen should a node fail or die during a transmission or reception. In the simulation, a node can only die at the end of a time step. Another simplification made with regards to the data is that the real data will only use the temperature portion of the files we download. The other variables, such as sea surface wind speed and humidity will not be used.

Chapter 3

3. Project Design

3.1 Design Flow

Here is the design flow for the project. Each block will be explained in further detail below:

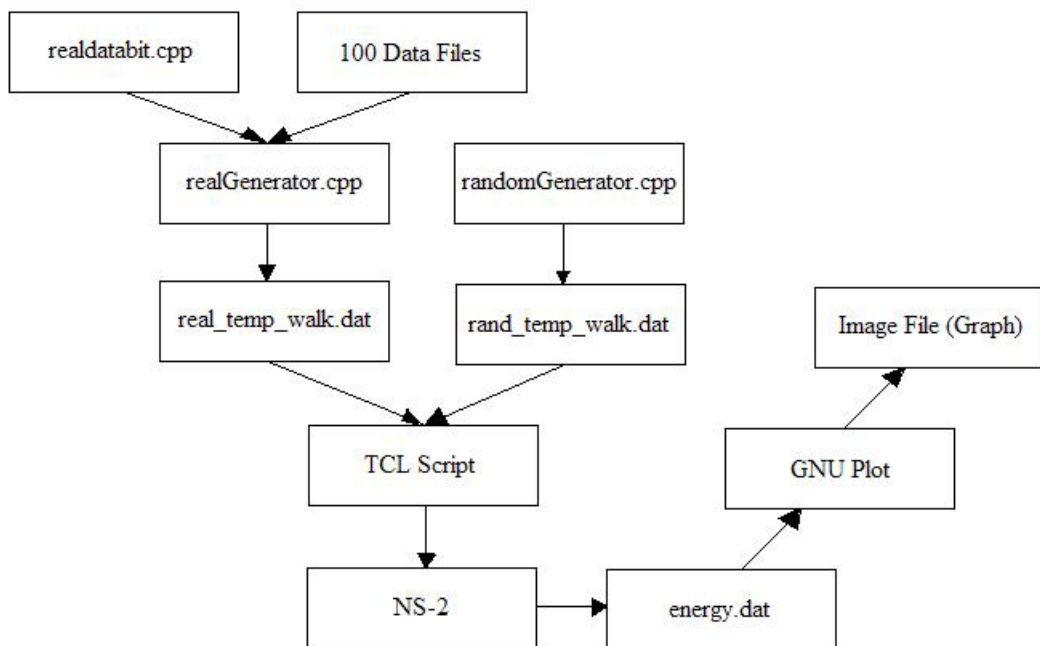


Figure 3.1: Architectural Design Flow

Data

As discussed above, there will be two kinds of data that feed into the simulation. The random data will be represented by a text file called `rand_temp_walk.dat`. This text file will be generated by a c++ file called `randomGenerator.cpp`. The real data will be created similarly but with a little more work involved. We downloaded 100 text files from the NOAA site. Each of these files is meant to represent a single node and all the values that node will hold for the entire simulation. The c++ file called `realGenerator.cpp` uses a helper file called `realdatabit.cpp`. A `realdatabit` object holds all of the data for a single time step as an array. The index of that array is the node number in the simulation. In `realGenerator.cpp`, there is an array of `realdatabit` objects that all of the data read from the files is fed into. Once all the files have been read and the data has been placed in the data structure, an output file is created in a format that the TCL Script file needs. It is important to note that these text files are generated before any simulation occurs. The simulator does not determine the value of the nodes. The text files describe what value each node will have at any given time during the simulation.

C++ Files

The LEACH topology was defined using TCL Script that MIT had generated. Because of this, the majority of the code is embedded within the TCL implementation. C++ files were used to generate the data files for both the random and real data traces. These are then read by the TCL Script to drive the simulation.

TCL Script

TCL is used here in order to make interacting with NS-2 easier. The TCL scripting language is fairly simple. All a TCL file does is create nodes and sets parameters that NS-2 can understand. This is then run using NS-2 which drives the simulation.

NS-2

NS-2 is the network simulator that the design is centered around. Because we will set the appropriate parameters in the TCL script fed into NS-2, it will produce a raw text file containing the information that describes the energy dissipation over time.

Text Files

There are a few text files in our design. Two text files will be generated with out c++ files called `rand_temp_walk.dat` and `real_temp_walk.dat`. These are read by the TCL Script in order to define what values the nodes have at any given time during the simulation. NS-2 creates an output text file that contains the raw information about the energy consumed by the network as time passed. This information will be used by GNU Plot to represent it graphically.

GNU Plo

This is an open source program that allows us to view the information in the text file that NS-2 produced in a graphical format. This will give us easy to view information on the overall results of our simulation. This graph will be stored in another file for future reference.

3.2 Design Timeline

Tasks	Week 1	Week 2	Week 3	Week 4	Week 5	Week 6
Read assigned papers	Yellow					
Install and learn NS-2	Yellow	Yellow				
Implement LEACH		Red	Red			
Setup sensor energy model		Yellow				
Setup data traces		Yellow				
Integrate LEACH and sensor energy model			Yellow	Yellow		
Display energy dissipation in real-time.				Yellow		
Setup quality-aware sensor data collection framework				Yellow		
Implement AL, AS, and ALS models					Red	
Implement sensor state management					Yellow	
Integrate ALS and sensor state management						Yellow
Work that James will do.	Yellow					
Work that Phil will do.	Red					
Work that both will do.	Yellow					

Figure 3.2: Design Timeline

Above is our design timeline. The tasks of our project are listed to the left, the times those tasks are accomplished are listed to the right, and the colors represent who completed the designated tasks.

Chapter 4

4. Implementation

4.1 Design Drift

The original design described how the majority of the code would be in c++ files in order to generate TCL script to drive NS2. The purpose of this was to minimize the amount of TCL that we had to learn while maintaining the flexibility and comfort of primarily coding directly in c++. The implementation actually drifted slightly away from this original design. The majority of the code is actually directly in TCL without first going through the c++ files. The reason for this was due to the fact that the code used to create LEACH was taken from MIT which was entirely in TCL with only one or two c++ files that did rather little comparatively.

4.2 Redefined Requirements

Because of the change in design, the project as a whole was set back substantially. All code that describes how data flows into the simulation, how the individual sensor models are implemented, and how the simulation runs overall is completely determined by the way in which we implemented LEACH. Since we did not implement LEACH ourselves, we had to learn how MIT did it by looking through the numerous files that define it. This, coupled with having to learn TCL in far more detail than originally foreseen, consumed a lot of time. This delay was significant enough to put the project's success in jeopardy. After meeting with our client, we were given a more succinct project goal which was to complete the quality aware sensor network and have a functional simulation that runs with both the real and random data defined above using the LEACH topology. A graph displaying energy dissipation over time is also a functional requirement that we will still meet.

Chapter 5

5. Conclusion

5.1 Results

The simulation produced four graphs that we will discuss here. This first graph (Figure 5.11) shows the LEACH routing protocol energy consumption without any data being sent. Each jump on the curve depicts when cluster heads are being reconfigured.

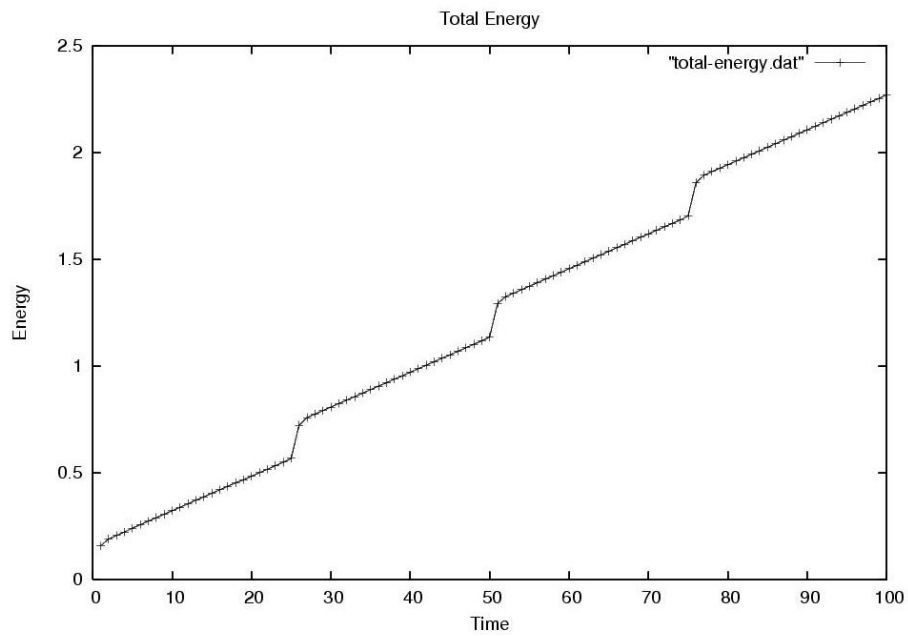


Figure 5.11

The next graph (Figure 5.12) shows the energy consumption of the three sensor energy models without having any data sent. The jumps on the curve once again represent the times when the cluster heads are being chosen. It shows Active-Sleeping to be the lowest for energy cost and Active-Listening-Sleeping to be the most expensive.

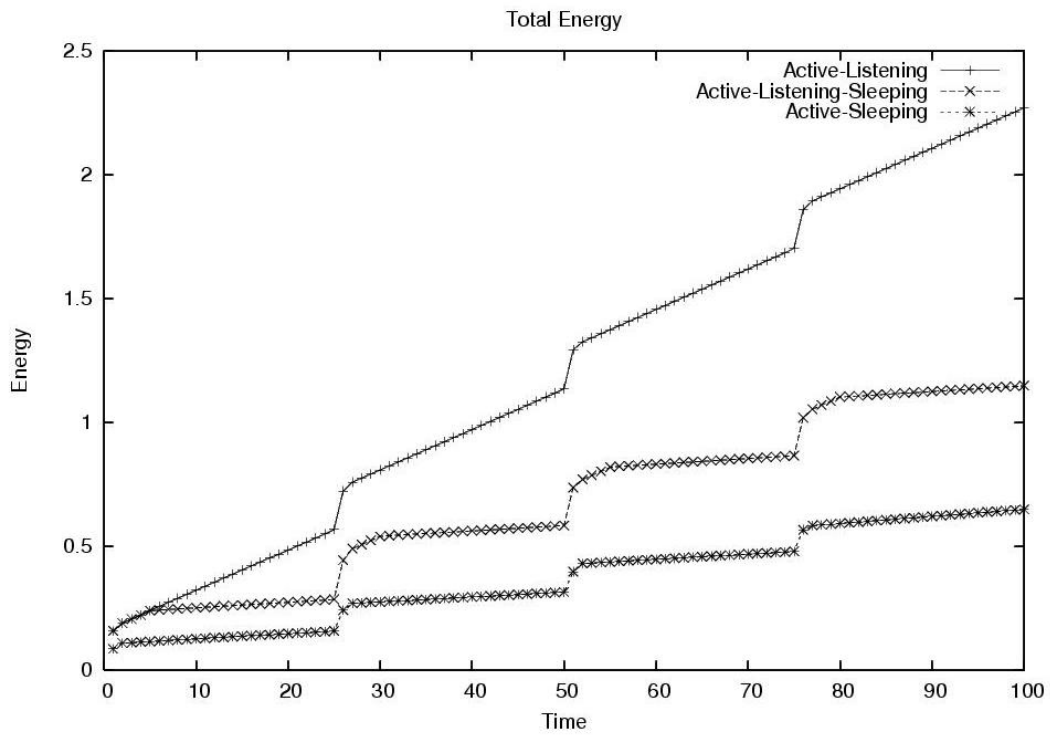


Figure 5.12

The last graph (Figure 5.13) that we have created depicts the energy consumed by the network while processing our rand_temp_walk.dat file with the Active-Listening model.

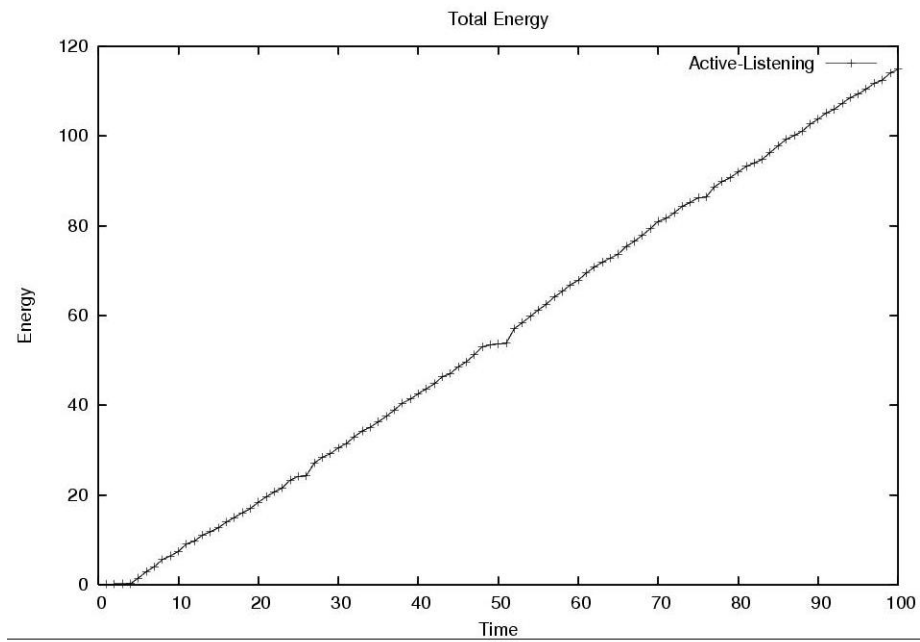


Figure 5.13

The last graph (Figure 5.14) that we have created depicts the energy consumed by the network while processing our rand_temp_walk.dat file with the Active-Sleeping model.

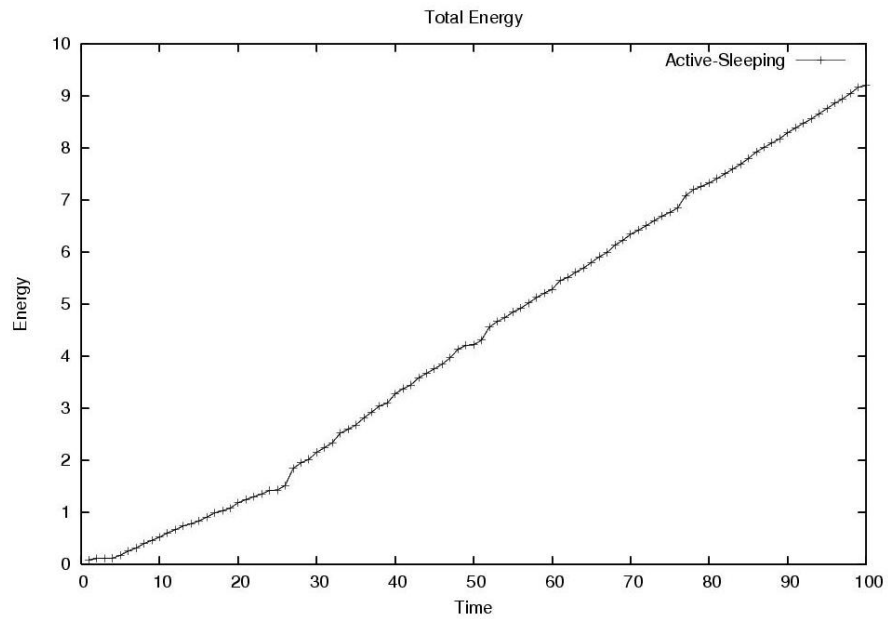


Figure 5.14

The last graph (Figure 5.15) that we have created depicts the energy consumed by the network while processing our rand_temp_walk.dat file with the Active-Sleeping model.

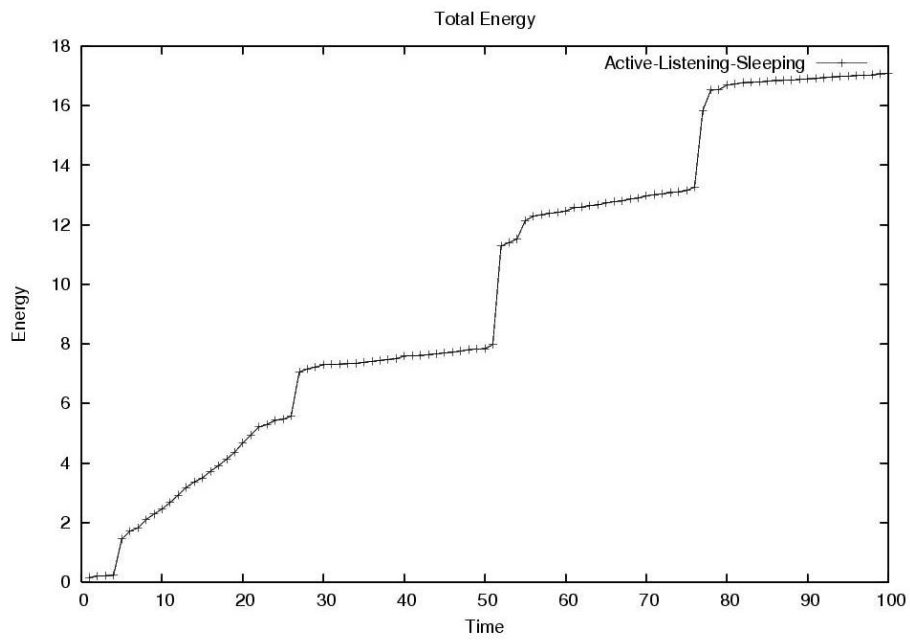


Figure 5.15

5.2 Conclusion

Overall the project as a whole was a success. Despite the unfortunate change in our design a running simulation was still possible. Our graphs show that our simulation fulfills the requirements.

5.3 Future Directions

The next step is to define functions that take the simulation variables as parameters instead of hard coding the values. Also, direct comparisons between alternate routing protocols, alternate sensor energy models, alternate topologies, and multiple base station situations should follow the work accomplished by this project.

Chapter 6

6. Appendix

6.1 Glossary

AL – Active-Listening model for nodes in the network.

ALS – Active-Listening-Sleeping model for nodes in the network.

AS – Active-Sleeping model for nodes in the network.

LEACH – Low Energy Adaptive Clustering Hierarchy. This is the network topology used for the nodes in our simulation.

NOAA – National Oceanic and Atmospheric Association. This is where the real data for the simulation comes from.

NS-2 – Network Simulator 2. This is the platform that our simulation will run on.

TCL Script – Tool Command Language Script. This is our intermediate code that will initialize and drive the simulation in NS-2.

WSN – Wireless Sensor Network. This is the kind of network that we are simulating.

Chapter 7

7. References

- [1] Q. Han, S. Mehrotra, and N. Venkatasubramanian. Sensor Data Collection with Quality Guarantees. May 14, 2006.
- [2] NOAA. Tropical atmosphere ocean project, pacific marine environmental laboratory. <http://www.pmel.noaa.gov/tao/>, June 2006.
- [3] D. Culler, D. Estrin, and M. Srivastava. Guest Editors' Introduction: Overview of Sensor Networks, IEEE Computer, 37(8). August 2004.
- [4] R. Szewczyk, E. Osterweil, J. Polastre, M. Hamilton, A. Mainwaring, and D. Estrin. Application driven systems research: Habitat monitoring with sensor networks, Communications of the ACM, Special Issue on Sensor Networks. June 2004.
- [5] K. Romer, O. Kasten, and F. Mattern. Middleware Challenges for Wireless Sensor Networks, ACM SIGMOBILE Mobile Computing and Communications Review, Vol. 6, No. 4, October 2002.
- [6] J. Zhao and R. Govindan. Understanding Packet Delivery Performance In Dense Wireless Sensor Networks, ACM SenSys, November 2003.
- [7] W. Rabiner Heinzelman, A. Chandrakasan, and H. Balakrishnan. Energy-Efficient Community Protocol for Wireless Microsensor Networks, 'Proceedings of the 33rd International Conference on Systems

