Contents

Figures

Tables

Chapter 01: Introduction

1.1 Introduction

Wireless sensor network is one of the most widely used network in today's world. This kind of network is used in different field of networking properties like as military observation, military surveillance, air traffic controls, weather forecasting, autonomous vehicles management systems and many more. Let's see how WSN network works? First of all, there are sensors are attached in different area to gather information regarding their particular objective, these kinds of sensor are act like a node in the network. When the sensor gathers different data then they pass it to a processing device.

The processing device than process the data precisely and then it presents to us. Because of the complexity of the network there are some important issue we are facing in today's world. The main problem in WSN is that its energy efficiency is not high. The energy consumption rate of sensor is pretty high. Because of that the survivability of this network is questionable. In many routing protocols there are different technique used to create an energy efficient network. But the result is not that optimistic. So, we used different "LEACH" protocol to simulate the result and find that which one is best for energy efficiency and node survivability in existing "LEACH" cluster formation algorithm. In our proposed "LEACH" protocol we find that the efficiency of energy and node survivability after a simulated round are higher than previous variation of leach protocol. This simulation result helps us to draw or make a more sufficient Wireless sensor network.

Finally, we use MATLAB simulation to evaluate different "LEACH" protocol result with our modified "LEACH".

1.2 Background

In the recent past, different WSN models were developed in terms of robustness, energy efficiency, privacy & security, scalability, heterogeneity, self-configuration & adaption, responsiveness. Among them one of the most critical fact is energy efficiency as wireless sensor networks are mobility network having limited lifespan. LEACH is an energy efficient model developed in early 2000 that has significant routing scheme. But this was more improved in further variations of LEACH such as LEACH-C. Here we are working on LEACH by implement different cluster head formation technique.

1.3 Motivation

Modern technological advances have made optimistic feasibility of Wireless sensor networks (WSNs). Energy consumption emerges as one of the most critical aspect in WSNs regarding hardware and software design, practicability. Cluster head (CH) are selected by a specific algorithm. Nodes send data to Cluster Head and Cluster Head sends to the sink. When the energy of CH tends to empty, new CH is selected. If the CH or sensor nodes energy remains long time, the network lifespan increase. The main target is to acquire minimum communication and computation costs. So, this is why it's motivated us to proposing an optimized LEACH protocol.

1.4 Problem Statement

Researcher have used different routing protocols to identify how they can build a robust and efficient cluster head selection technique but it can gain more energy efficiency and survival of the nodes or sensors in the network. The consumption of the energy in previous work is pretty much higher and also the survival rate of the sensor is pretty low.

The problem with different “LEACH" protocol is that they are more focused on cluster head management system that any node to a cluster head is a percentage of 0.1 i.e. if a node is cluster head now than it cannot be a cluster head for next 1/p rounds. So, it’s a drawback because if the ongoing cluster head have enough energy it can remain as cluster head. This process can save the energy of new cluster head formation.

In WSNs, sensor nodes need to send data in fixed time interval using TDMA by the cluster head and condition. These networks applied in several important tasks as military, health, environment etc. If they are unable to send data in crucial time, it tends to create irrecoverable situation. Also, sensors have limited energy reservation. So, this limited energy reservation needs to be properly utilized. Here we are approaching a routing protocol to route the data in a feasible and proficient way to reduce energy consumption which increase network lifetime.

1.5 Objectives

The aim is to provide an energy efficient “LEACH" routing protocol and stable the network life time.

- To provide energy efficient “LEACH" protocol.
- To provide stable network connection.
- To increase network lifespan.
- Reduce the dead nodes.

1.6 Contribution

- A modified “LEACHE" protocol has been proposed in this paper.
- Consumption rate of energy has been reduced with the applying of Optimized “LEACH” protocol.
- We simulate a comparison result between different “LEACH" protocol.
- Improvement of Cluster Head selection technique
- Hibernate sensor node scheme
- Minimum 5% better network lifetime than LEACH
- Maximum 20% better network lifetime than LEACH

Chapter 02: Literature Review

2.1 Overview

After we have known about the background, aim, contribution and objective of our research, it's a high time to evaluate literatures which are related to our work. In here we will be discussing about various kinds of wireless sensor network routing protocols and different cluster head selection techniques.

2.2 LEACH

Leach protocol is a TDMA based MAC protocol. The primary objective of this protocol is to improve the lifespan of wireless sensor networks. It achieves this work by lowering energy consumption. Leach protocol consists of two phases:

1) Set-up phase

2) Steady phase

LEACH protocol basically works in two phases with consist of several phases. Hierarchical routing protocol is a representation of Leach protocol. The main two feature of LEACH protocol is Its self-adaptive and self-organized [2]. Leach protocol uses round as unit, each round is made up of cluster set-up stage and steady state storage

1.Set-up phase

The main objective of set-up phase is to make cluster and select the cluster head for each of the cluster by choosing the sensor node with maximum energy [3].

Set-up phase has three operational steps:

1. Cluster head notification

2. Cluster set up

3. Creation of transmission timeline

Initially cluster head sends notification packet to inform the cluster nodes, when a node is elected as a cluster head then it cannot become cluster head again until all the nodes of the cluster have become cluster head once. This is useful for balancing the energy consumption. After initial phase, non-cluster head nodes receive the cluster head notification and then send join request to the cluster head. This notification informing that non-cluster head nodes are members of the cluster under that cluster head. In the meantime, this non-cluster head nodes save their energy by switch off their transmitter and turn it on only when they have something to transmit to the cluster head [2]. In the next phase, each of the chosen cluster head creates a transmission schedule for the member nodes of their cluster. Transmission schedule is created according to the number of nodes in the cluster. After that Each node then sends its data in the allocated time schedule [3].

2.Steady Phase

In steady phase, sensor nodes send their data to the cluster head. In this phase sensor nodes in each cluster can communicate only with the cluster head with the help of single hop transmission. After sensor nodes sends their data to cluster head than cluster head forwards the collected data to the base station either directly or via other cluster head along with the static route defined in the source code. After a selected time, the network again goes back to the set-up phase.

2.3 Existing works on LEACH routing protocol

1. Energy-Efficient Communication Protocol for Wireless Microsensor Networks.

   In this paper they look at communication protocols, which improve these networks overall energy dissipation. In here they find that the conventional protocols for sensor networks likes direct transmission, minimum-transmission-energy, multi-hop routing, and static clustering which are not optimal at all. So, they proposed “LEACH" which is called Low-Energy Adaptive Clustering Hierarchy. Throughout the nodes or sensors, this “LEACH" protocol is able to distribute energy dissipation evenly and in here it also doubling the system life time.

2. Efficient Cluster Head Selection Strategy for Provisioning Fairness in Wireless Sensor Networks.

   In this paper they select a fair and balanced cluster head for cross layer protocol in wireless sensor network, a novel scheme is proposed in this paper. In here they argue that “LEACH-C" and it’s most of the variants select cluster head by considering energy and distance as a crucial parameter. In this paper it’s state that this will reduce energy consumption of nudes but failed to create balanced cluster. To, improve this situation they consider residual energy, number of neighbor nodes and one hop neighbor information.

   They proposed an algorithm which will eliminate higher energy consumption but also it covers a wide range of area in a sensing manner. They evaluate the proposed protocol with two recent clustering approach, which is distributed approach and centralized approaches also showed their performance simulation against different “LEACH" routing protocol in MATLAB which show the improvement over related schemes. In their future extensions they will include adaptive intra cluster communication techniques with different traffic which will eventually reduce idle listening by cluster head.

3. MODLEACH: A Variant of LEACH for WSNs.

   In here, researchers modified one of the most used wireless sensor network routing protocol “LEACH” by introducing efficient cluster head replacement scheme and dual transmitting power level. Their modified “LEACH" outperformed it using metrics of cluster head formation, throughput and network life. After using this they used hard and soft thresholds are used in the process of modification to boost its performance.

Finally, they analysis different performance of “LEACH", “MOD-LEACH", “MOD-LEACHHT", “MOD-LEACHST" with considering metrics of throughput, network life and cluster head replacement.

4. Energy-efficiency and Route-selection of multilevel Hierarchal Routing Protocols in WSNs

   For more energy efficient and effective communication, they compared six different protocols of different environment where they represent their own schemes of energy reducing, clustering technique and selection of different route. This paper made an insight view about that which protocol is more effective and robust to create an energy efficient network.

   In here, they performed a MATLAB simulation to analyze and compare the performance of “LEACH", multi-level hierarchal “LEACH" and multi- hop “LEACH".

   Moreover, they try to develop a good protocol design which should be able to scale well both in energy heterogenous and homogenous environment and it can also meet the demand in different application scenarios and networks.

5. Design guidelines for wireless sensor network: communication, clustering and aggregation.

   A systematic cost-based analysis of both the modes and provide results that could serve as guidelines to decide which mode should be used for given settings. It determined closed form expression for the required number of cluster heads and the required battery energy of nodes for both the modes. It also proposed a hybrid communication mode which is a combination of single hop and multi-hop modes. In here it tries to find which is more cost effective than either of the two modes.

   The main problem which is address is that of determining the optimum number cluster heads, of dimensioning, and determining the optimum mode of communication in each cluster (single hop or multi-hop).

6. ANCH: A New Clustering Algorithm for Wireless Sensor Networks.

   In this paper, they worked with ANCH, a new clustering algorithm for wireless sensor networks which reduces the networks energy consumption and significantly improvements in network lifespan. Optimizing the distribution of cluster heads across the network, which achieved this improvement.

   The result of their extensive simulation shows that their method decreases the consumption of network energy and increase the network life time.

7. Energy Efficient Hybrid Multi-Hop Clustering Algorithm in Wireless Sensor Networks.

   In this paper they present two approach for two different scenarios. First of all, Energy efficient Hybrid clustering scheme (EEHCS) is used for direct hop communication for base it worked

when the base station is situated in the network region and secondly, Energy Efficient Hybrid Multi-Hop Clustering Scheme (EEHMCS) is used for multi-hop communication to the base station using other CHs if base station is not situated in the network parameter.

In here base station select energy efficient CHs depending on remaining energy and number of member nodes by centralized CH election method and broadcast the select CH message to all the nodes. It reduces the control message overhead required for distributed clustering approach.

It showed 27.63% percent performance improvement in network lifespan over LEACH-C.

8. An Application Specific Protocol Architecture for Wireless Microsensor Networks.

   This protocol architecture for microsensor networks is developed and analyze to achieved higher energy efficiency. It's a combination of energy-efficient cluster-based routing and media access together with application specific data aggregation to achieve good performance in terms of system life time, latency and application-perceived quality.

   This proposed protocol featured a new distributed cluster formation. This technique makes large number of nodes with self-organization, deploy algorithm for adaptive clustering and for evenly distribute the energy among all nodes it rotates cluster head positions. It also enables distributed signal processing to save communication resources.

9. Sensor Networks: Evolution, Opportunities, and Challenges.

   Different technical challenges and opportunities of WSN are describe briefly by researchers. Network identification methods, control segment and routing technique, collaborative signal, information processing, tasking and querying, and security.

   Localized algorithms and directed diffusion, distributed tracking in wireless ad hoc networks, and distributed classification using local agents are some recent work in wireless sensor network. Researchers are also working on different network algorithms.

10. On Efficient Clustering of Wireless Sensor Networks.

    Two different methodologies for clustering sensor networks which increase the sensor coverage and also create efficient operation for wide range of networks. Deployment of multiple gateways and partitions sensors among these gateways is the first step in this technique. The second step describe that some sensors are designated as agents for a single gateway in order to reach out-of-range nodes. The performance of both approaches is compared in a simulated environment.

    The sensitivity of the operation of the sensor network and reduce the ambiguity of the propagation model is the main objective to achieve. Which eventually allow to facilitate the management of dense networks with sheer number of sensors it enables effective monitoring of sensors nodes behavior.

2.1 Advantage and Disadvantage

There are many advantages of LEACH protocol which are:

1. The Cluster Heads collect data from sensor nodes which helps to reduce the traffic in the entire network.

2. For saving energy it uses single hop routing from nodes to cluster head.

3. It upgrade the lifetime of the sensor network.

4. For creating cluster location information of nodes is not required.

5. Control information from base station and global knowledge of the network is not required.

Besides the advantages of LEACH, it also has some disadvantages which are as follows:

1. No information about the number of cluster heads in the network.

2. When cluster head dies, the cluster will become useless.

3. Clusters are divided randomly, which results in uneven distribution of Clusters.

2.2 Summary

In this chapter we discuss about the LEACH protocol and its characteristics. We also discuss about the existing work done in the LEACH protocol. Finally, we discuss about the advantages and disadvantages of LEACH protocol.

Chapter 03: Environment Study

3.1 Overview

In this chapter we will discuss about the different tools and techniques we used to model down our proposed work. This chapter is consisting of briefly discussion and system over view.

3.2 Tools

For implement and comparing our work with other methods we used MATLAB simulation and Omnet++. These tools are vastly used as an operation platform of simulation and decision making. In this tool there are many wireless sensor network and simulation packages. We use MATLAB for comparison between different routing protocols.

3.2.1 Simulator

A simulator sets up an environment to the original device's OS, but doesn't attempt to simulate the real device's hardware. Some programs may run a little differently, and it may require other changes (like that the program be compiled for the computer's CPU instead of the device's), but it's a close enough match that you can do most of your development against the simulator.

There are different kinds of simulator exist in WSN which are NS2, NS3, OMNET++, TOSSIM etc.

3.2.2 Omnet++

Omnet++ is a library and framework used for simulation. The language is used in Omnet++ is C++ which is object oriented. The foundation of Omnet++ is based on generic architecture, because the intention is to use in many forms of application. It helps to simulate wireless communication network. Also, on-chip network, queuing networks etc. It provides domain-specific functionality. Some of the examples are support for sensor networks, internet protocols, phonetic networks, ad-hoc networks and many more. We can simulate these networks using Omnet++.

Because of that it’s gained massive popularity as a simulation platform for networks in scientific community.

In Omnet there a model is used for component architecture. It is written in C++ language. Moreover, it also has a high-level language (NED). In here, we can modify codes and models for our research purpose.

There is also a graphical user interface which is very user friendly and it supports simulation and we can include the simulation kernel of it in our application.

Features of OMNET++

- It has application usable simulation kernel.
- Various tools for plotting data.
- A graphical user interface editor for NED files.
- A Command line user interface for simulation execution.

- Compiler for NED topology description language.
- Different kinds of utilities like random number seed generation tools, make file tools etc.
- Different types of user interface for simulation execution.
- It's also has a graphical user interface.

Different File Format in Omnet++

1. NED: Network description file (NED) is a topology description language of OMNeT++. It is the one of the simple syntaxes and very powerful topologies to define chain, ring, mash, hypercube, tree structure etc.
   Features of NED file
   - It has hierarchical, typed module structure.
   - Flexible topology description.
   - Different parameters.
   - It has support for flexible partitioning of the model parallel execution.
   - Structure of NED file

   Simple Class: - Simple class is used for building basic blocks for other modules, It's a C++ class. In here parameters are declared that belong to module. Parameter can be of type double, int, bool, string and xml. Parameter get their value from NED file or from the configuration (omnetpp.ini) file.

   In here gates are the connection point of the modules. Omnet++ has three types of gates which are input, output and in-out.

   Network Class: - Network class is consisting of submodule and connection. Submodule declare the nodes architecture in the network and which types of node they are. On the other hand, connection is used to declare how nodes are transmitted data in a given time parameter. You can change network design and color in this class.

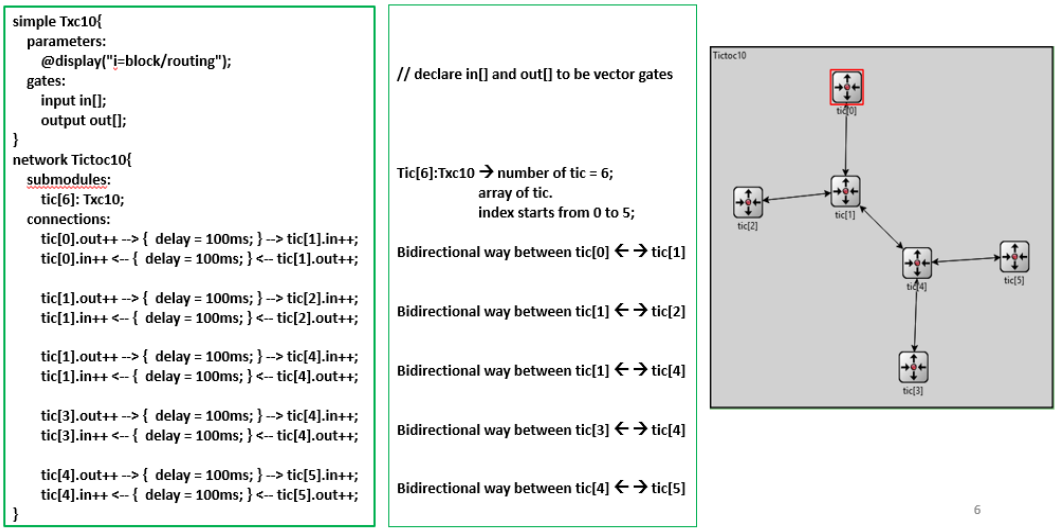

Figure 1 NED file structure

2. Header file (.h file): - Header file is used to create methods for new class. In C/C++ header file uses to create relation between source file and NED file. It's run in the background. It's consists of different library program.
3. Program file (.c file): - This file is used for writing program in C/C++. In here simple modules are programmed in C++.
4. Message file (.msg files): - Message definition or .msg file is used to define various message types and add data fields to them. Omnet++ will convert this message into C++ classes.
5. Configure file (omnetpp.ini): - When a program first started it read the NED files, then omnetpp.ini file usually called. This configuration file has settings for network.

Summary

After analysis every single simulator which related to wireless sensor network we find Omnet++ is the most suitable for our research work. The most essential part to work with any simulator in WSN is that the understanding the code formation and also the support of different existing library. In here Omnet++ standout from all of them. Where ns-2 and ns-3 is really difficult to use because those simulators are run on different operating system or environment. Which everyone is not familiar with and installing ns-2 and ns-3 is very difficult job to do. Moreover ns-2 graphical user interface is not wall established. On the other hand, ns3 is comparatively new in the platform, so that its library is not sufficient to work on WSN protocols.

3.2.3 MATLAB

MATLAB is known as "Matrix Laboratory" which is used for numerical computation and visualization. This enhanced the software for using linear algebra. It's a great tool for solving different equations, algebraic and for numerical integration. It's very popular to the scientific community. It's helps us to easily work with entire matrices rather than one number at a time.

MATLAB is very unique and more user friendly for using matrix and vector manipulation. It's comes with basic set of tools for visualizing data and for performing calculation on matrices and vectors.

Some MATLAB toolbox are given below: -

- Statistic toolbox.
- Neural Network toolbox.
- Fuzzy logic toolbox.
- Signal processing toolbox.
- Wavelet toolbox.
- Financial toolbox.
- Bio informatics toolbox.
- Database toolbox.

3.2.3.1 Features of MATLAB

- Manipulation and simplification of mathematical expression.
- Different symbolic integration, differentiation, transformation and linear algebra.
- 2D and 3D plotting for analytical equations and functions.

- ODE (Ordinary differential equation) solving mechanism.
- Different variable precision arithmetic's.
- Symbolic expression conversion to MATLAB, Simulink, Sims-cape, C, Fortran, and LaTeX.
- Different unit systems for specifying, converting and computing using SI, US and custom unit system.
- Packaging of MATLAB programs as standalone application.
- It can create Microsoft Excel add-ins for integration with excel spreadsheets.
- Application can be distributed for free.
- It has high encryption capability for intellectual property.
- MATLAB code can deploy against Hadoop and Spark.

3.3 Techniques

3.3.1 Localization Methods Taxonomy

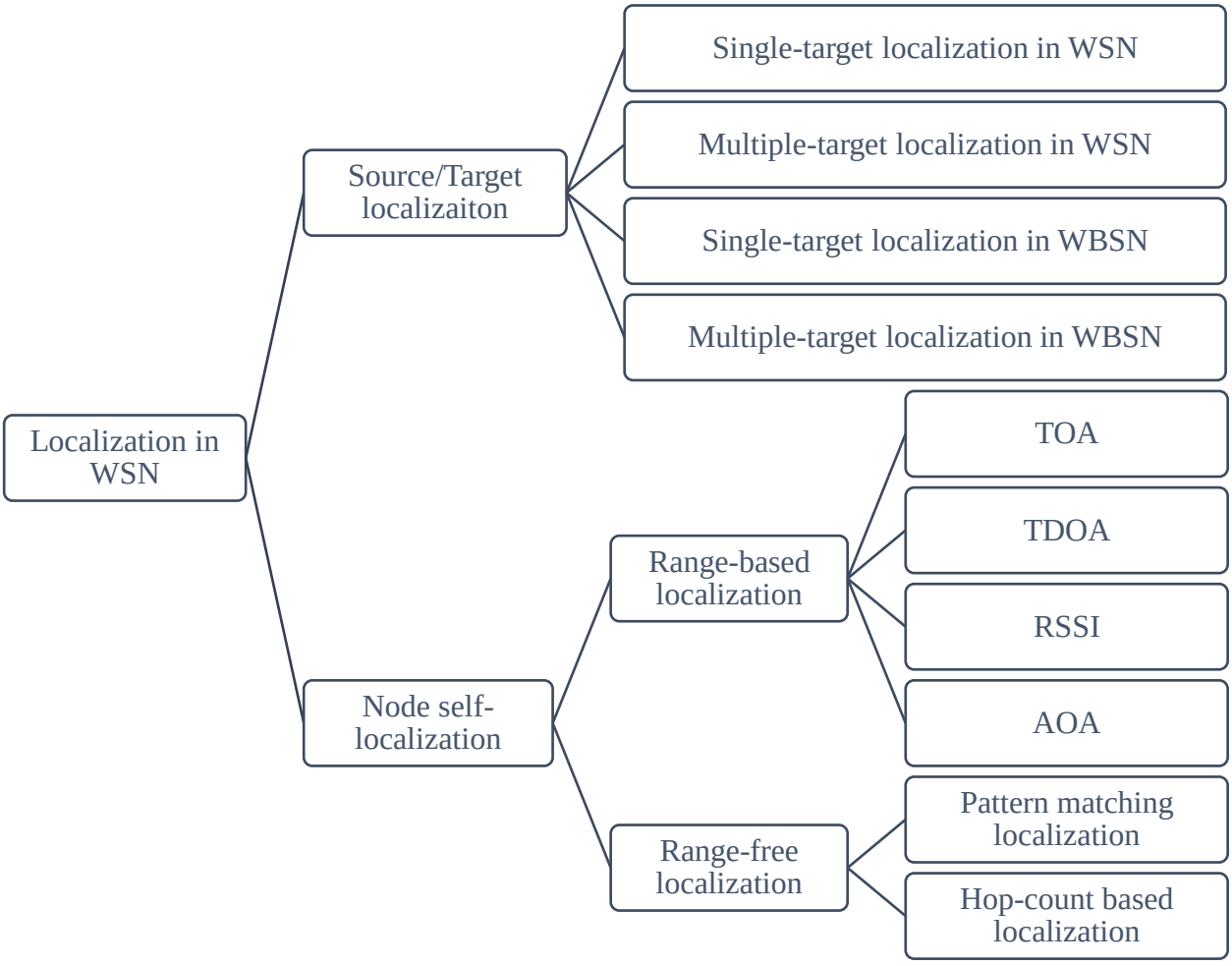

Figure 2 Localization Taxonomy

3.3.2 Cluster Formation

Wireless Sensor Networks (WSNs) are deployed over a target area to supervise certain phenomena of interest. Each node takes readings from the local environment, processes and transmits a certain number of packets, containing the sensed data, to the sink node. Two common modalities can be used to access the shared medium to communicate the data to the sink node: unscheduled and scheduled-based transmissions. In this paper, a clustered-based architecture is considered partly based on the encouraging results presented in previous works, such as. In a cluster-based architecture, there are two distinct phases:

- The cluster formation phase, where all the active nodes transmit a control packet directed to the sink node in order to be part of the cluster. Specifically, the active nodes in the supervised area transmit their control packet with probability τ in each time slot. If there is only one transmission, that is only one node transmits, the control packet is successfully received by the sink node, and the node that successfully transmitted this packet is considered to be already a member of a cluster. As such, this node no longer transmits in the cluster formation phase. The remaining nodes continue this process until all the active nodes successfully transmit their control packet. If there are two or more transmissions in the same time slot, all transmissions are considered to be corrupted, and the control packets involved in this collision have to be retransmitted in future time slots. Hence, when a collision occurs, none of the involved nodes are aggregated to a cluster.

- The steady state phase, where all the nodes in the system transmit their data packets to a cluster head (CH), which in turn transmits an aggregated data packet to the sink node.

3.3.2.1 Cluster Formation Phase

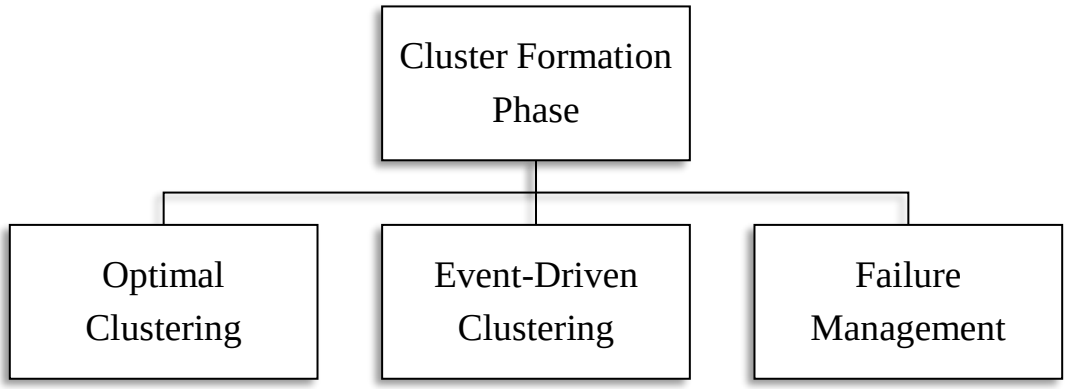

Figure 3 Cluster Formation Phase

3.3.2.2 Taxonomy of Clustering Methods

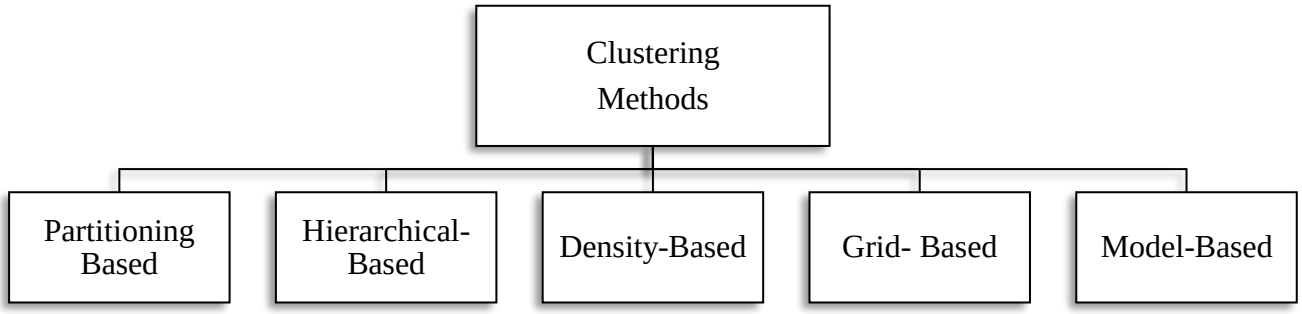

Figure 4 Clustering Methods

3.3.3 Aggregation

The main goal of data aggregation algorithms is to gather and aggregate data in an energy efficient manner so that network lifetime is enhanced. Wireless sensor networks (WSN) offer an increasingly attractive method of data gathering in distributed system architectures and dynamic access via wireless connectivity.

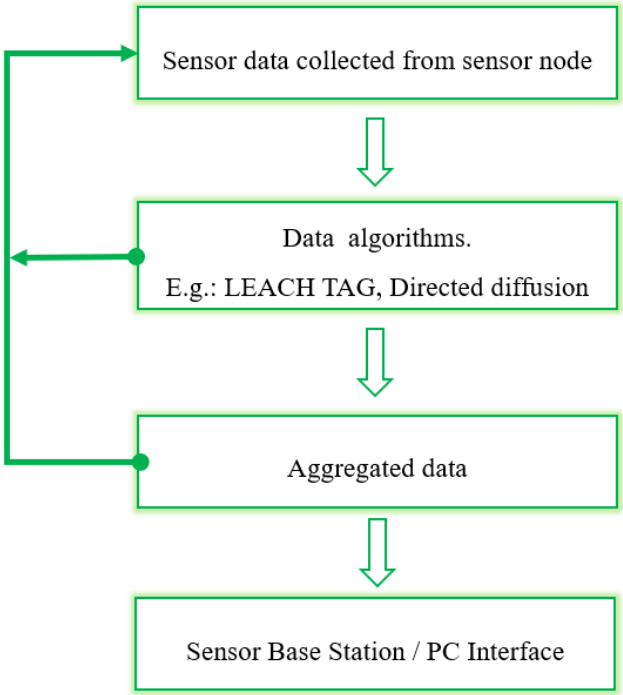

Figure 5 Data Aggregation Algorithm

Data aggregation is a process of aggregating the sensor data using aggregation approaches. The general data aggregation algorithm works as shown in the below figure. The algorithm uses the sensor data from the sensor node and then aggregates the data by using some aggregation algorithms such as centralized approach, LEACH (low energy adaptive clustering hierarchy), TAG (Tiny Aggregation) etc. This aggregated data is transfer to the sink node by selecting the efficient path.

3.3.4 Multi-hop

For the case of cellular and wireless networks, the process of wireless communication take place in the last link between the wireless end system and base station. The technique of multi-hop networks is there will be one or more intermediate nodes which will stay along the path that will receive and send the packets forward with the help of wireless links. By using Multi-hop wireless several benefits can be achieved. For example: It can improve the connectivity and extend the coverage area of the network compared to a network with single wireless links. It can also save power and energy as transmission over “short” links consumes less power than transmitting over “long” links. It makes the efficient use of the wireless medium as it enables higher data rates which result in higher throughput. It also prevents the wide deployment of cables. So, we can deploy it by which cost will be minimized. If the multi hop network is dense then several paths can be made available which can be used to increase the robustness of the network. There is a disadvantage in multi hop. The protocols that is developed for a network which is fixed like cellular as well as the internet are not optimal for multi-hop. We can use it for the case of routing protocols, where multicast, unicast and broadcast routing protocols have been developed for ad-hoc and sensor networks.

On the transport layer, the Transmission Control Protocol (TCP) is the de facto standard in the Internet and in order to allow interoperability, we have to make sure that the TCP should be supported in multi-hop wireless networks as well. However, many protocol mechanisms such as congestion control and error control based on acknowledgements do not work efficiently in multi-hop wireless networks due to some reasons for example contention and control packet overhead. Even on application level new concepts are required to support discovery of available applications and services. After investigating several application scenarios was found for multi-hop wireless networks. At first to extend the coverage area of cellular network the use of multi-hop was proposed. Recently, to provide broadband internet service without expensive cables, mesh networks is proposed. Wireless mesh networks consist of mesh routers and mesh clients, where mesh routers have minimal mobility and form the backbone of wireless mesh networks [Aky05]. They make use of heterogeneous network technology such as IEEE 802.11, 802.16, and cellular radio networks. Relaying nodes can also be mobile such as in case of vehicles. In that case the term mobile ad-hoc network is more appropriate. Vehicular networks as a special case of mobile ad-hoc networks make use of the frequently existing communication equipment in cars (either pre-installed or enabled by equipment carried by passengers). Wireless sensor networks are another emerging technology, can cover large geographical areas, and provide connectivity without having direct physical access to each sensor node. Sensor nodes can be configured and sensor data can be read using multi-hop networking. The following sections discuss the contributions from COST Action 290 in research areas discussed above. Section 5.2 investigates the performance of forwarding and relaying in multi-hop wireless networks and discusses approaches to optimize wireless resource usage. Section 5.3 investigates routing protocols for unicast, multicast, and broadcast communication in multi-hop wireless networks.

3.3.4.1 Packet Relaying in Multi-Hop Networks

In wireless multi-hop networks, nodes communicate with each other using wireless channels and do not have the need for common infrastructure or centralized control. Nodes may cooperate with each other by forwarding or relaying each other's' packets, possibly involving many intermediate relay nodes. This enables nodes that cannot hear each other directly to communicate over intermediate relays without increasing transmission power. Such multi-hop relaying is a very promising solution for increasing throughput and providing coverage for a large physical area. By using several intermediate nodes, the sender can reduce transmission power thus limiting interference effects and enabling spatial reuse of frequency bands. In ad-hoc networks, the medium is shared and nodes arrange access to the medium in a distributed way independent of their current traffic demand. In particular given standard ad-hoc routing protocols that try to minimize relaying nodes on the path, nodes closer to the network center are more likely to become a relay node. This has the inherent drawback that a node that serves as a relay node for transmissions of multiple neighboring nodes is prone to become a performance bottleneck. As it is necessary to understand performance of such relay networks, the next sub section provides an overview on performance analysis of a relay node. When multiple relays are involved across an end-to-end path, it is important to control overhead for each single packet transmission. Unfortunately, current Medium Access Control (MAC) and physical layers for Wireless Local Area Network (WLAN) based multi-hop networks impose high overhead for the transmission of small data packets, which is common for Voice over Internet Protocol (VoIP). By combining several small packets into larger ones, per packet transmission overhead can be reduced significantly. Therefore, the following subsections provide an overview on efficient packet aggregation mechanism.

3.3.4.2 Power Consumption

A wireless sensor node can only be equipped with a limited power source (<0.5Ah, 1.2V) due to the several hardware constraints described. Moreover, for most applications, replenishment of power resources is impossible. WSN lifetime, therefore, shows a strong dependence on battery lifetime. Thus, the sources that consume energy during the operation of each node should be analyzed and maintained efficiently.

In a multi-hop ad hoc sensor network, each node plays two separate and complementary roles:

- **Data originator:** Each sensor node's primary role is to gather data from the environment through the various sensors. The data generated from sensing the environment need to be processed and transmitted to nearby sensor nodes for multi-hop delivery to the sink.
- **Data router:** In addition to originating data, each sensor node is responsible for relaying the information transmitted by its neighbors. The low-power communication techniques in WSNs limit the communication range of a node. In a large network, multi-hop communication is required so that nodes relay the information sent by their neighbors to the data collector, i.e., the sink. Accordingly, the sensor node is responsible for receiving the data sent by its neighbors and forwarding these data to one of its neighbors according to the routing decisions. The operations related to each role affect how the energy is consumed in a sensor node. Moreover, as explained in Section 3.5, the failure of a few nodes can cause significant topological changes and might require rerouting of packets and reorganization of the network. Hence, power conservation and power management are integrate l parts of any communication protocol in WSNs. Consequently, the design of power-aware protocols and algorithms for WSNs is of paramount importance. The main task of a sensor node in a sensor field is to detect events,

perform local data processing and then transmit the data. Power consumption can hence be divided into three domains, sensing, communication, and data processing, which are performed by the sensors, the CPU, and the radio, respectively. A breakdown of the power consumption of a MicaZ sensor node is shown. It can be seen that, among these three, a sensor node expends maximum energy for data communication. The three sources of energy consumption are discussed next.

3.3.5 Single-hop

In a wireless sensor network most of the applications are dependent on a single-hop wireless network to send the data of a phenomena to Base station. That is the result of a measurement by a sensor are sent directly to BS. In most of the cases the applications are dependent on various kinds of sensor which are embedded into a device. For example, small sensors can be embedded into a traffic surveillance system to monitor traffic on congested roads or be used to monitor hot spots in a region or building.

Another famous example of wireless sensor network composed of embedded sensors is Health Care. The Health monitoring sensors are embedded into watches. When the patient wears this, the sensors sense the data such as pulse, blood pressure, heart beat etc. When it can sense any risk it also sends alarm messages to a nearby center using single-hop wireless sensor communication. As the sensors are powered by battery there is also used an intelligent and smart sensor power management system that helps to increase the energy efficiency and also quality of service (QoS).

3.4 TDMA

Now a days it has been considered as the best choice for an energy efficient protocol which allow multiple access in wireless sensor networks. In TDMA there exist many approaches which of increased complexity, which use various kinds of algorithm and merging information from different layers or multiple access techniques. Nowadays there are many surveys which are finding out the advantages and disadvantages of different protocols, but we have a feeling that they cannot draw the general image of TDMA techniques, which can be helpful to choose the protocol for a network. Or for someone who is trying to establish e new MAC protocol. Now analyze various algorithms which are used in TDMA protocols and present a list of characteristics that a robust TDMA protocol requires.

3.5 IEEE 802.15.4

The intention behind the development of IEEE 802.15.4 was to provide a framework, also the lower levels for low power networks which also cost low. It only provides two layers. They are the MAC layer and the PHY layer. The upper layers are developed considering the need of market.

The IEEE 802.15.4 standard might not be known widely like other higher layer standards, for example Zigbee, which is publicized massively. Nevertheless, the IEEE 802.15.4 technology and standard forms the basis and it gives a platform for their operation.

In recent days the usage of IEEE 802.15.4 technology is increasing continuously along with other famous technology like Zigbee. The importance of this standard is also increasing. But still for the massive marketing for Zigbee and other standards, the news of IEEE 802.15.4 is known less.

The IEEE 802.15.4 standard was developed to provide a framework and the lower levels for low cost, low power networks. It only provides the MAC and PHY layers, leaving the upper layers to be developed according to the market needs.

Accordingly, the IEEE 802.15.4 standard may not be as widely known as some of the higher layer standards such as Zigbee which have been widely publicized. Nevertheless, the IEEE 802.15.4 technology and standard forms the basis, underpinning their operation and providing a reliable platform for their operation.

Now with technologies such as Zigbee being used in a large way, the use of IEEE 802.15.4 technology is corresponding increasing, and it is becoming an important standard. However, with widespread marketing for Zigbee and other standards, IEEE 802.15.4 is less well known.

Chapter 04: Methodology and Modeling

4.1 Overview

Methodology includes the step by step process of how our work implemented and modeling refers to the elaborated description of methodology.

1. At first, the cluster head is selected based on the threshold value.
2. After the selection of cluster heads, we determine the hibernate nodes of cluster heads.
3. When the energy of the cluster head is less than a energy-limit value, the cluster head becomes a normal node.
4. At that time the hibernate node of that cluster head will come into play the role of Cluster Head.
5. Then again another hibernate node will be determined for the newly formed Cluster Head. Here we have ensured that the node which just played the role of Cluster Head will not be selected as Hibernate node immediately.
6. All the selected Cluster Heads and Hibernate nodes stored in a structure array.
7. This process is repeated till the all nodes of the network are dead.

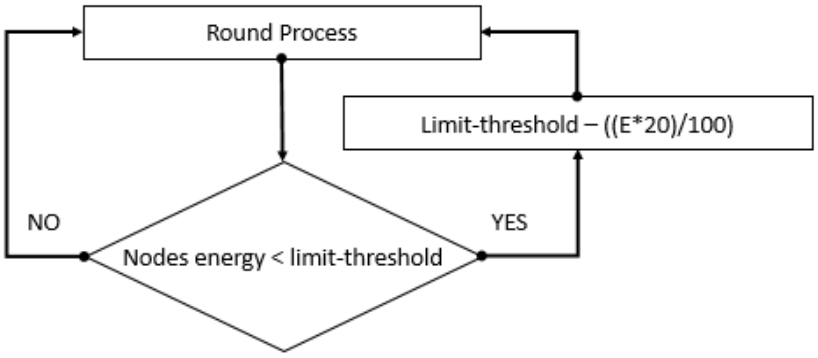

Figure 6 Round flow chart

4.2 Network Parameters

Table 1 Network Parameters for simulation

Network Parameters	Value
Network Size	200 X 200 m2
Number of Nodes	100
Channel Type	Wireless Channel
Initial Energy of Sensor Nodes	0.5 J
Transceiver idle state energy consumption	50 nJ/bit
Data Aggregation/ Fusion Energy consumption	E_{DA} = 5 nJ/bit/report
Amplification Energy (Cluster to BS) d >= do	E_{fs} = 10pJ/bit/m^2
Amplification Energy (Cluster to BS) d <= do	ε_{amp} = 0.0013 pJ/bit/m^2

Energy consumption is based on three terms that are energy for free space and multipath and energy for data aggregation. Also amplifying energy is needed to proficiently pass the data.

Individual Round Energy Consumption

- Transceiver idle state energy consumption
- Same energy cost for these following steps:

<table>
<tr><td>CH forming</td><td rowspan="3">Sent data to sink (BS)</td></tr>
<tr><td>CH remains as CH</td></tr>
<tr><td>H → CH shift</td></tr>
<tr><td colspan="2">Here energy consumption is same as transmitting energy which is stated below</td></tr>
</table>

- When CH receives data, energy consumption is same as receiving energy which is stated below.

4.3 First Order Radio Model

An illustration of the first order radio model is shown in figure below.

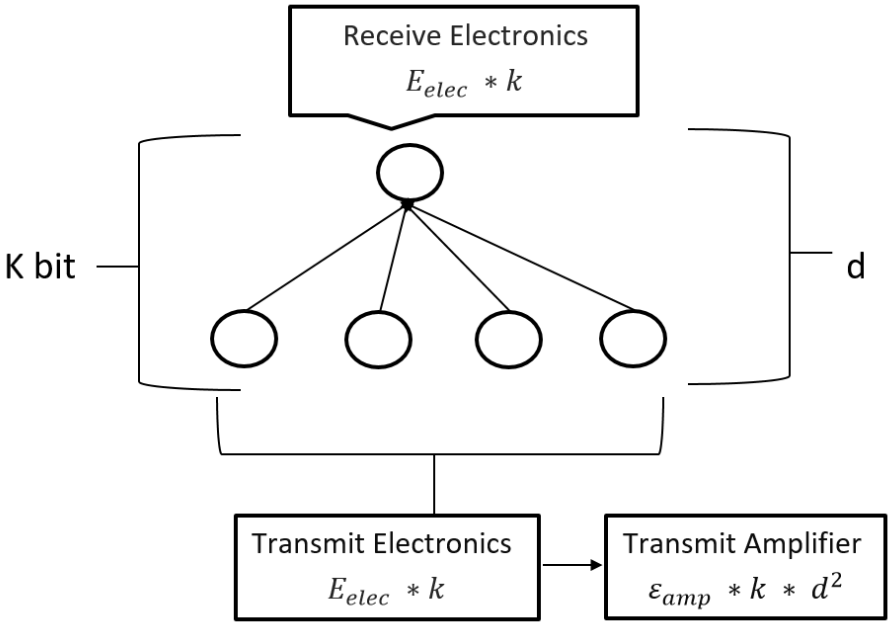

Figure 7 First Order Radio Model

The energy consumption of transmitting a k-bit packet at a distance d can be expressed by [1]:

$$E_{Tx}(k,d) = E_{elec} * k + E_{fs} * k * d^2 \qquad if(d \leq d_0) \qquad (1)$$

$$E_{Tx}(k,d) = E_{elec} * k + \varepsilon_{amp} * k * d^4 \qquad if(d > d_0) \qquad (2)$$

Receiving a k-bit packet can be calculated by:

$$E_{Rx}(k,d) = E_{elec} * k \qquad (3)$$

Where,

E_{elec} = E_{TX} + E_{DA} *or* E_{RX} + E_{DA}

E_{DA} = required energy for data aggregation

ε_{amp} = amplifying power to ensure the smooth operation

k = size of the transmitting or receiving packet

d = the distance between the sender node and receiver node

d_0 = a predefined value, which depends on the performance of the sensor node

4.4 Node Deployment

Nodes placement of WSN regulate the effectiveness of the whole network in terms of parameters such as latency, power consumption, interference etc. However, as several WSN are deterministically deployed over an area of interest, it is very difficult to perform specific node placement [16].

Node deployment is the first step to form the network. Each node has 4 attributes.

Table 2 Node Attributes

Node	S(i).xd	Node coordinate in x-axis
	S(i).ym	Node coordinate in x-axis
	S(i).E	Initial node energy
	S(i).type	Type of node (N / CH / H)

In our forming network, total number of nodes is 200. Parameter values of all nodes are setting up by a for loop where coordinates are determined by rand () function and others are default values.

Now nodes deployed in the network. Base station is formed by default base station values.

4.5 Cluster Head Selection

Let's define how we generate cluster formation and how to select cluster head in our proposed method. First of all, we will define the network at the initial phase. After that the node have all the ability by itself if it wants to be a cluster or not in the current round. In here the decision will be made by analyzing the suggested cluster head based on threshold value. n node will choose the decision by generating random number using rand(). Now we find that if the number is less than threshold T (n), the node becomes a cluster-head for the current round.

$$T(n) = \frac{p}{1 - p * \left\{ r \bmod \frac{1}{p} \right\}} \quad (4)$$

Where,

p = the desired percentage of cluster heads (e.g., P = 0.1)

r = the current round

4.6 Hibernate Node Formation

After forming cluster head matrix, we determine hibernate nodes for each cluster head. Hibernate nodes are selected based on the closest distance and greater energy value than energy-limit. Whenever a node is satisfied these conditions, it'll become hibernate node for the according cluster head. When a hibernate node turns into cluster head, again hibernate node selected for this cluster head. This process is ongoing till the condition is dissatisfied.

Hibernate node characteristics:

- After selected as hibernate, it stays in idle state till turn into cluster head
- No sensing energy consumption
- Cost only idle state energy consumption

4.7 Cluster Formation

After the formation of cluster heads and related hibernate node, cluster will be formed based on distance. Nodes of the network will be connected to the cluster head which is nearest.

4.8 Dead Node Calculation

In our simulation total number of rounds is 1500. We are storing the specific round numbers at which the first node, 50% nodes and all nodes are dead.

Pseud code 1 Dead node calculation:

```
/* dead is the dead node counting variable and r is the round number */
if dead is 1 then
   if flag_first_dead is 0 then
      first_dead=r
      flag_first_dead=1
   end
end

/* 10% node of the whole network = 0.1*n */
/* where n is the total number of nodes */
if dead is 0.1*n then
   if flag_teenth_dead is 0 then
      teenth_dead=r
      flag_teenth_dead=1
   end
end
if dead is n then
   if flag_all_dead is 0 then
      all_dead=r
      flag_all_dead=1
   end
end
```

If any node's energy is less or equal to zero then dead count is increment gradually.

1ST NODE	When the first node of the network is dead, then first_dead=round and flag_first_dead=1.
50% NODE	When 50% nodes of the WSN dead, then fifty_dead=r and flag_fifty_dead=1.
ALL NODE	When all nodes dead, then all_dead=r and flag_all_dead=1.

As we need to visualize the node survival performance, we are storing these first node, 50% nodes and all nodes dead counts in STATISTIC.DEAD and STATISTICS.ALIVE array.

4.9 Network Total Energy Calculation

Total energy is the summation of all nodes energy.

In each round, using for loop, we store total energy in the TotalNetworkEnergy variable.

Pseuducode: Total Energy Calculation

```
/* n is the total number of nodes */
/* S(i).E is the individual energy of each node */
for i=1 to n
   if S(i).E is greater than 0 then
      TotalNetworkEnergy=TotalNetworkEnergy+S(i).E;
   end
end
```

As we need to visualize the total network energy, we store the value in STATISTIC.TotalEnergy and STATISTICS.AvgEnergy array.

Chapter 05: Structure Profile

5.1 Overview

To simulate our proposed LEACH routing protocol, we used OMNET++.

OMNeT++ is a component-based, modular and open-architecture discrete event network simulator. OMNeT++ represents a framework approach

- Instead of containing explicit and hardwired support for computer networks or other areas, it provides an infrastructure for writing such simulations
- Specific application areas are catered by various simulation models and frameworks, most of them open source.

5.2 Simulation Model Building

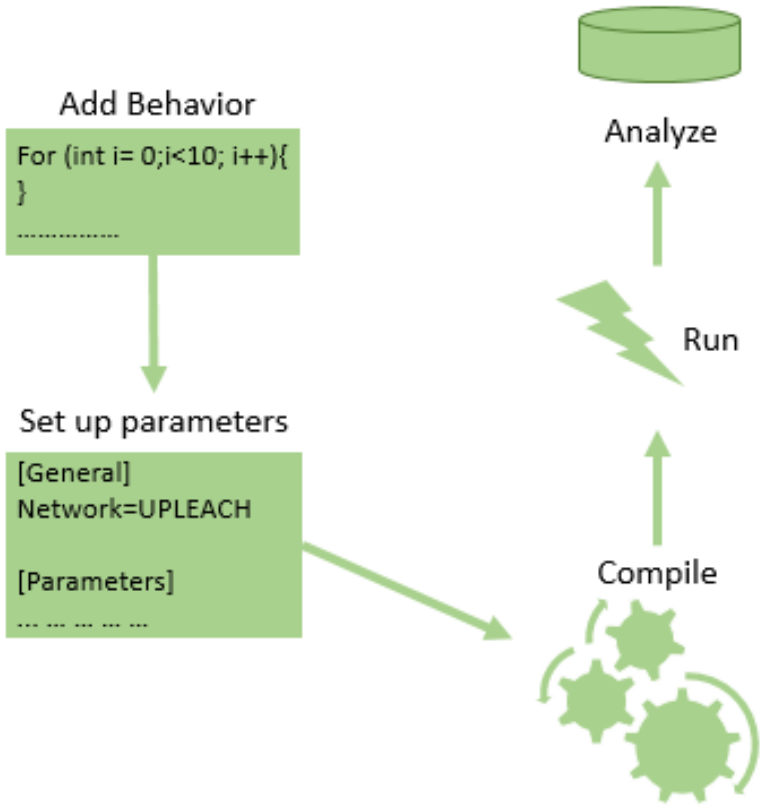

Figure 8 Simulation Model Building

5.3 Build Process

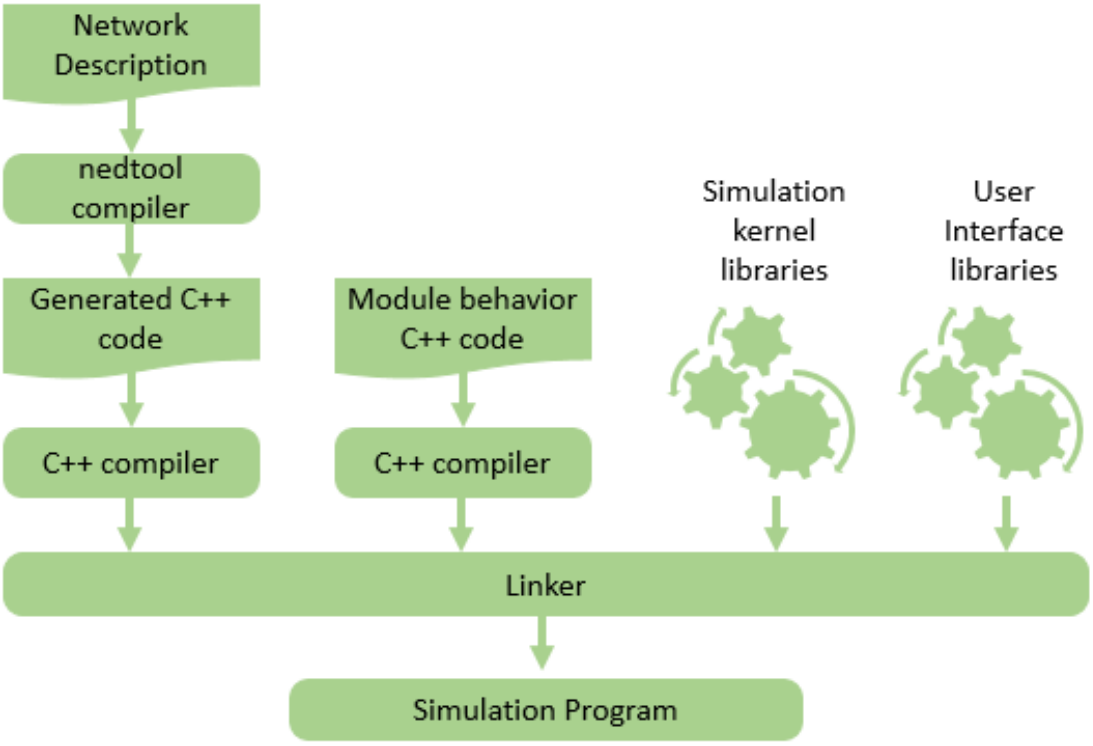

Figure 9 Build Process of Omnetpp Simulation

5.4 Simulation Phases

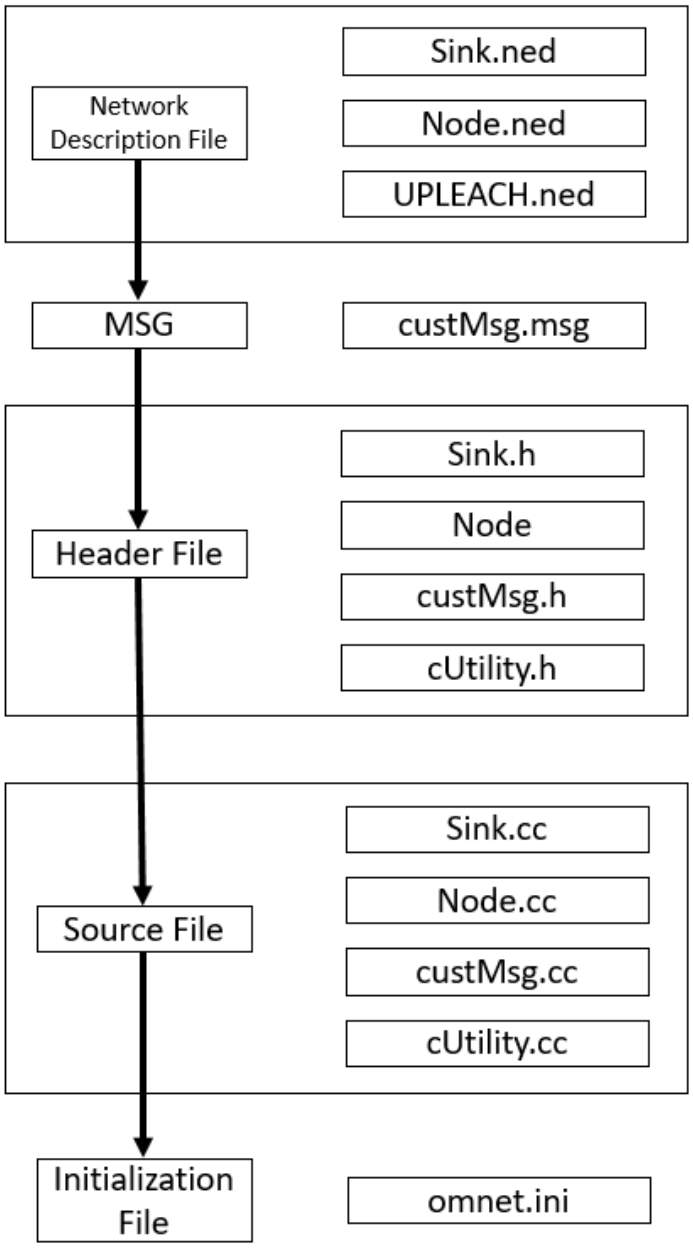

Figure 10 Simulation Phases

5.5 NED

UPLEACH.ned

This is the file for each node describes the input and output gates.

```
package UPLEACHPckg;

network UPLEACH
{
    parameters:
        int roundNumber = default(0);
        double totalRemainingEnergy = default(0);
        double avgRemainingEnergy = default(0);
        double lastRoundTime = default(0);
        double clusterHeadPercentage = default(0.1);
        int noOfCH = default(0);
        int noOfCluster = default(0);
        string lstCH = default("0");
        int sinkX = default(80); //old = 230
        int sinkY = default(20); //old = 40

      //1=Setup, 2=Data send to CH, 3=Data Send to Sink
        int networkStatus = default(0);

        int netSizeX = default(200);
        int netSizeY = default(200);
        int noOfNodeDied = default(0);
        int fstNodeDieRound = default(0);
        int tenNodeDieRound = default(0);
        int allNodeDieRound = default(0);

        //Added for throughput
        int noDataSentToCH = default(0);
        int noDataSentToSink = default(0);
        int noDataInSink = default(0);

        int noPacketSentToSink = default(0);
        int sendPacketToCH = default(0);

        int noOfWirelessNode @prompt("Number of Nodes") = default(2);
        //@display("bgb=netSizeX,netSizeY");
        //@display("bgb=800,600");
        @display("bgb=200,200");

    submodules:
        sink: Sink;
        node[noOfWirelessNode]: Node {
            @display("i=misc/node_vs,gold;p=230,140");
        }

}
```

Node.ned

This is the file for each node describes the input and output gates.

```
package UPLEACHPckg;
simple Node
{
    gates:
        input radioIn @directIn; // work as input and output
}
```

Sink.ned

This is the file for the sink describes the input and output gates.

```
package UPLEACHPckg;
simple Sink
{
    gates:
        input radioIn @directIn;
}
```

5.6 Message Definition

custMsg.msg

```
//enum PacketType{
//    RTS_DATA = 0; //request to send
//    CTS_DATA = 1; //clear to send
//};
packet custMsg{
    int sourceId;   //Initial source id of the message

    int messageId;  //Unique message id for each message

    int ackMsgId;   //Message Id, which acknowledgemnt is send

    int senderId;   //Immediate sender id of the message

    int intendedReceiverId;      //Message intended receiver id

    double packetGenerateTime;   //Initial packet or message generation time

    double packetReachToSinkTime;

    bool isRelayMsg;       //Is this message a relay message or not

    double packetReachTime;

    double overheadDelay;
};
```

5.7 Header File and Source File

A header file is a file with extension .h which contains C function declarations and macro definitions to be shared between several source files. There are two types of header files: the files that the programmer writes and the files that comes with your compiler.

Header file:

```
#include <headerFile>
Class ClassName: public inheitedModule{
        Public:
                variables;
                Constructor;
                Destructor;
        Private/protected:
                methodName (parameter);
}
```

Source file:

```
#include <headerFile>
Define_Module(ModuleName),                    //Defining user created header files.

//Definition of all methods declared in header file. Like as,
ModuleName :: methodName(Parameter){
        Method Definition
}
```

The header files and related source files that we are used to simulate our routing protocol are briefly depicted below.

Node.h

```
#ifndef NODE_H_
#define NODE_H_

#include <omnetpp.h>
#include <Sink.h>
#include <custMsg_m.h>

class Node: public cSimpleModule {

public:
    Variables

    Node();
    virtual ~Node();

private:
    void SetCoordinate();
    void ClusterHeadSelection(int roundNo);
    void ClusterFormation(int roundNo);
    void ResetG(int roundNo);

    int CalculateDistanceToBS(int nodeindex);
    custMsg* CreateCustMsg(const char *name);
    int CalculateDistance(int senderIndex, int receiverIndex);

    void SendDataToCH();
    void SendDataToSink();

    void WriteDeadNodeHistory(int noOfDeadNode, int roundNumber);
    void WriteOneTenAllNodeDeadHistory();
    void WriteNetworkEnergyHistory(int roundNumber);
    void CountSinkPacket(int roundNumber);
    void CountCH(int roundNumber, int noOfCH); //Cound CH vs Round


    void TempDataSendToCH();
    void TempDataSendToSink();
    void ThresholdCheck();

    //New added for throughput calculation
    void CountThroughput(int roundNumber);
    void ResetParam();

protected:
    virtual void initialize();
    virtual void handleMessage(cMessage *msg);
    void finish();
};
#endif /* NODE_H_ */
```

Sink.h

```
#ifndef SINK_H_
#define SINK_H_


#include <omnetpp.h>
#include <custMsg_m.h>

class Sink: public cSimpleModule {
public:
    Variables;

    Sink();
    virtual ~Sink();

private:
    void SetCoordinate();
    custMsg* CreateCustMsg(const char *name);

protected:
    virtual void initialize();
    virtual void handleMessage(cMessage *msg);
};

#endif /* SINK_H_ */
```

cUtility.h

```
#ifndef CUTILITY_H_
#define CUTILITY_H_

#include <vector>
#include <string>

class cUtility {
public:
    cUtility();
    virtual ~cUtility();
    std::string getTestMsg();

    std::vector<std::string> split(const std::string& str, const char& ch);

    std::vector<int> convertToInt( std::vector < std::string > vStr);
};
#endif /* CUTILITY_H_ */
```

custMsg_m.h

```
#ifndef _CUSTMSG_M_H_
#define _CUSTMSG_M_H_

#include <omnetpp.h>

class custMsg : public ::cPacket{
  protected:
      //Variables
  private:
    void copy(const custMsg& other);

  protected:
    // protected and unimplemented operator==(), to prevent accidental usage
    bool operator==(const custMsg&);

  public:
    custMsg(const char *name=NULL, int kind=0);
    custMsg(const custMsg& other);
    virtual ~custMsg();
    custMsg& operator=(const custMsg& other);
    virtual custMsg *dup() const {return new custMsg(*this);}
    virtual void parsimPack(cCommBuffer *b);
    virtual void parsimUnpack(cCommBuffer *b);

    // field getter/setter methods
    virtual int getSourceId() const;
    virtual void setSourceId(int sourceId);
    virtual int getMessageId() const;
    virtual void setMessageId(int messageId);
    virtual int getAckMsgId() const;
    virtual void setAckMsgId(int ackMsgId);
    virtual int getSenderId() const;
    virtual void setSenderId(int senderId);
    virtual int getIntendedReceiverId() const;
    virtual void setIntendedReceiverId(int intendedReceiverId);
    virtual double getPacketGenerateTime() const;
    virtual void setPacketGenerateTime(double packetGenerateTime);
    virtual double getPacketReachToSinkTime() const;
    virtual void setPacketReachToSinkTime(double packetReachToSinkTime);
    virtual bool getIsRelayMsg() const;
    virtual void setIsRelayMsg(bool isRelayMsg);
    virtual double getPacketReachTime() const;
    virtual void setPacketReachTime(double packetReachTime);
    virtual double getOverheadDelay() const;
    virtual void setOverheadDelay(double overheadDelay);
};
inline void doPacking(cCommBuffer *b, custMsg& obj) {obj.parsimPack(b);}
inline void doUnpacking(cCommBuffer *b, custMsg& obj) {obj.parsimUnpack(b);}

#endif // _CUSTMSG_M_H_
```

Chapter 06: Mathematical Analysis & Data Sheet

6.1 Mathematical Analysis

- In idle state:

  per round energy consumption = 0.000000005J

  1000 round energy consumption = 1000 * 0.000000005 = 0.000005J

- Suppose, Message size = 4000bit.

  d0 = 50m

  d1= 20m

  d2= 60m

- Required energy to receive:

  receiving energy $= (E_{RX} + E_{DA}) * 4000$

  $= \{(0.000000005 + 0.000000005) * 4000 \}$

  $= 0.00004J$

- Required energy to transmit:

  - for d1, if(d<=d0) then

    transmit energy $= (E_{TX} + E_{DA}) * 4000 + E_{fs} * 4000 * d^2$

    transmit energy $= \{(0.000000005 + 0.000000005) * 4000 \}$

    $+ (0.000000000001 * 4000 * 20^2)$

    $= 0.0000416J$ [Lowest energy consumption for single transmission]

  - for d2, if(d>d0) then

    transmit energy $= (E_{TX} + E_{DA}) * 4000 + \varepsilon_{amp} * 4000 * d^4$

    transmit energy $= \{(0.000000005 + 0.000000005) * 4000 \}$

    $+ (0.000000000000013*4000* 60^4)$

    $= 0.00071392J$ [Highest energy consump. for single transmission]

- Hibernate node's energy remains almost same till they turned into CH.
- Suppose, a CH remains as CH for 10 rounds. For these 10 rounds, linked hibernate node have no energy dissipation. So, its energy remains same for 10 rounds i.e. till convert to CH.

  H.E = 0.0000416 (Lowest energy dissipation per round)

  10 rounds will cost = 10 * 0.0000416 = 0.000416 J

  H.E = 0.00071392 (Highest energy dissipation per round)

  10 rounds will cost = 10 * 0.00071392 = 0.0071392 J

- If total round is 1000, then Hibernate node's energy remains same for 100times.

So, for 1000 round,

H.E = 0.000416 * 100 = 0.0416 (Lowest energy dissipation)

H.E = 0.0071392 * 100 = 0.71392 (Highest energy dissipation)

This energy is saved in 1000 rounds for single CH.

- If there are 10 CH, then

  Total lowest energy dissipation = 0.0416 * 10 = 0.416 J [0.83% of total network energy]

  Total highest energy dissipation = 0.71392 * 10 = 7.1392 J [14% of total network energy]

By this energy, the network can run another 100+ rounds.

But in simulation around 8%-10% energy saved that enhance the network lifetime.

6.2 Graphical Data

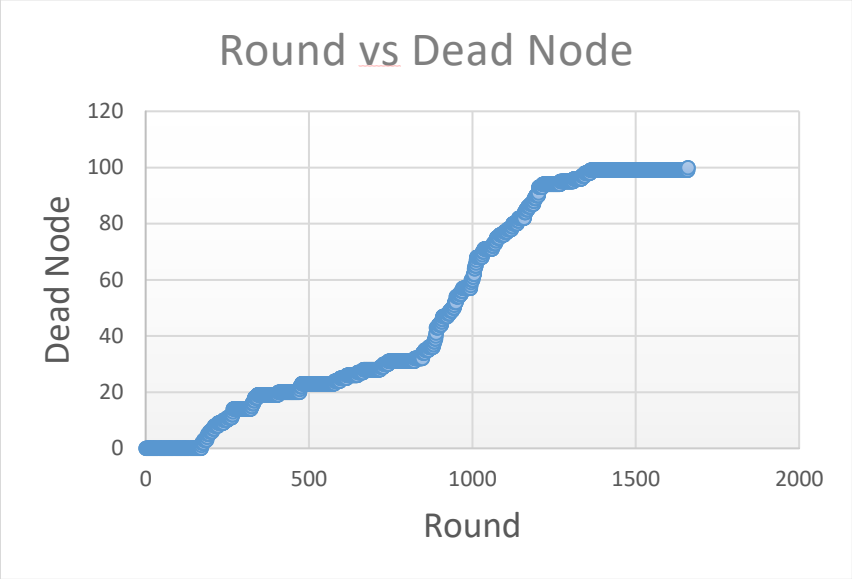

Figure 11 Dead node per round

Figure 10 shows how many nodes died per round. After all nodes are dead, the above figure formed.

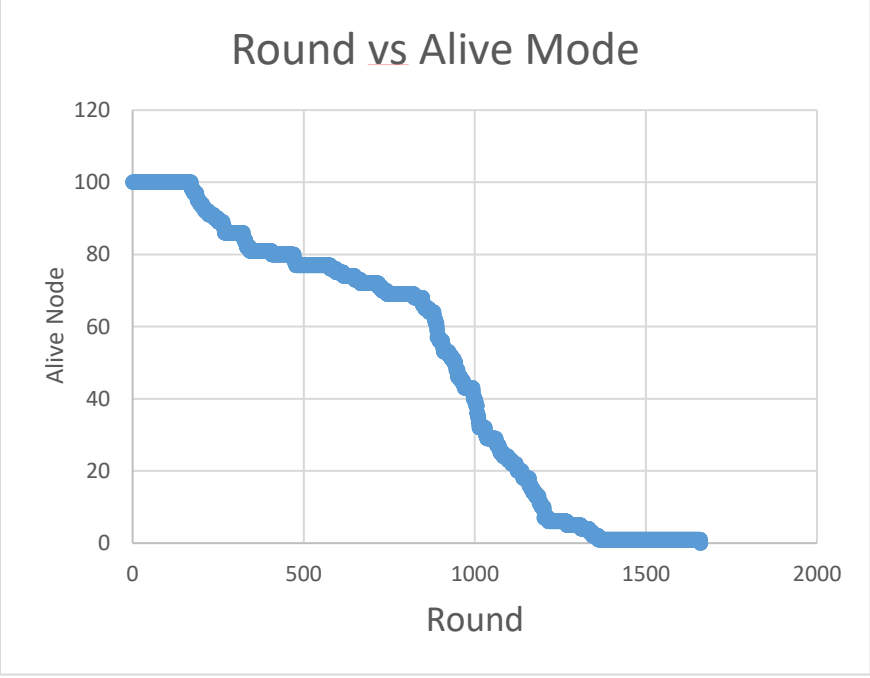

Figure 12 Alive node per round

Figure 11 shows how many nodes alive per round. After all nodes are dead, the above figure formed.

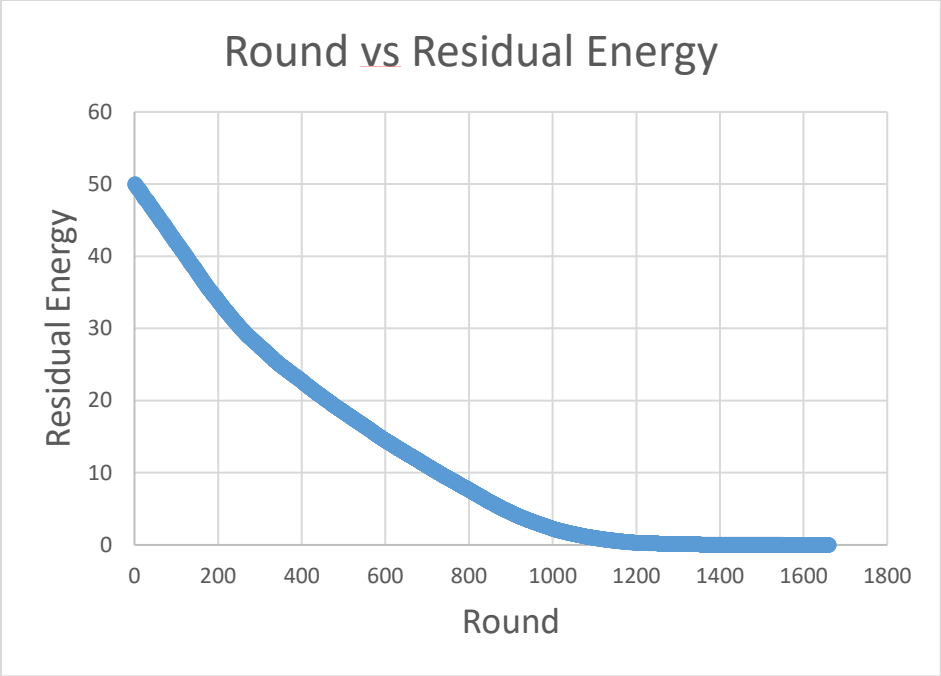

Figure 13 Residual Energy per round

Figure 12 shows that residual energy per round consist in the simulation

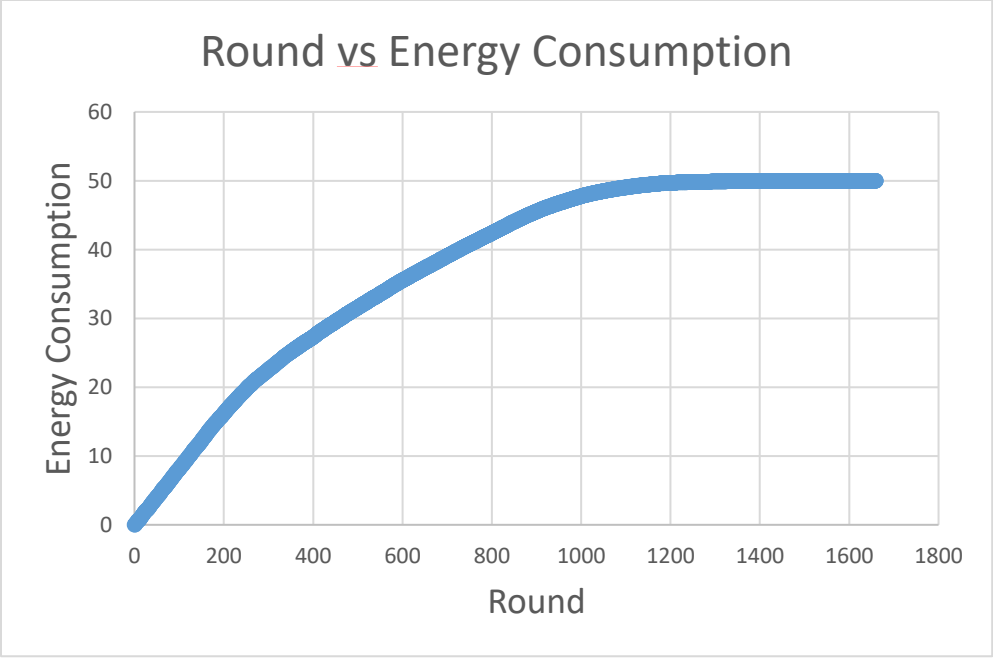

Figure 14 Alive node per round

Figure 13 shows that the energy consumption rate per round. This figure formed for energy consumption rate.

Packet Sent to BS

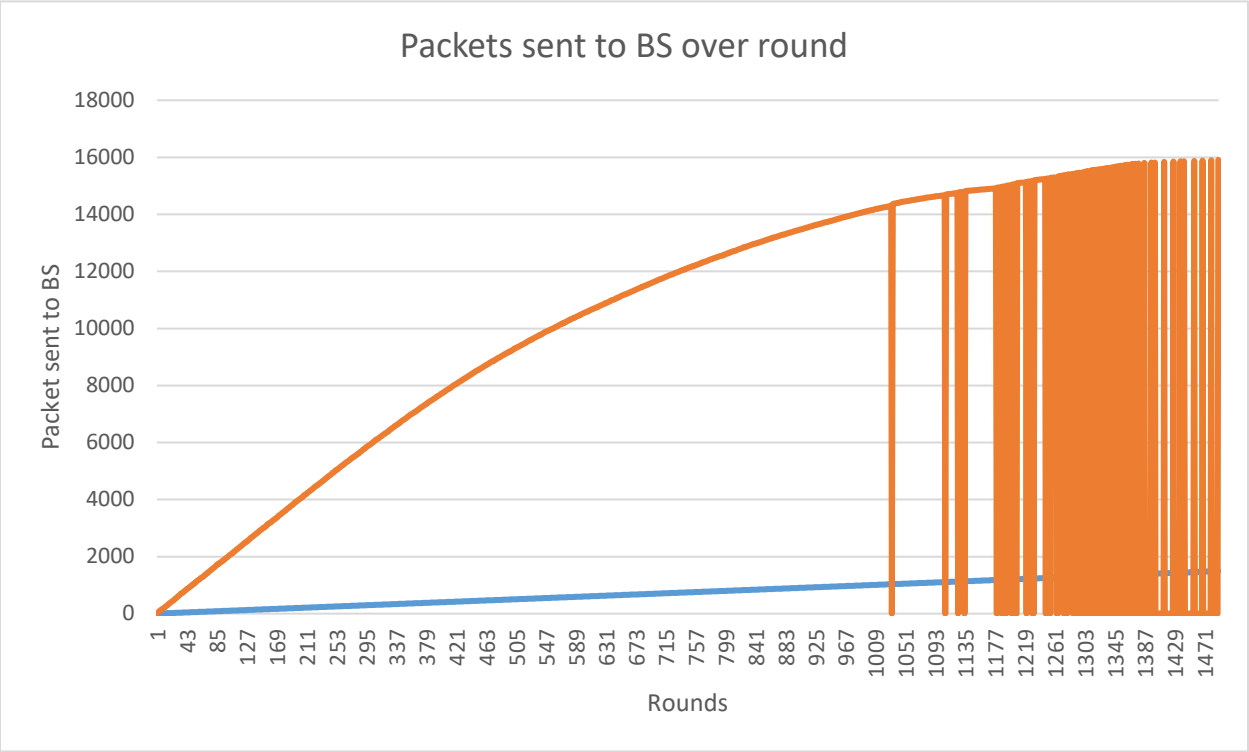

Figure 15 Packet sent to BS over round

6.3 First Iteration Cluster Head Data

Table 3 First Iteration Cluster Head Matrix

xd	Yd	distance	id
85.27281	89.30603	159.4223	12
234.9289	311.1384	116.4979	25
220.3239	368.359	169.5813	35
264.717	221.9252	68.3301	39
317.2354	302.1898	155.5214	49
272.9212	259.7236	94.25715	52
369.7098	39.90836	233.304	56
364.7339	369.4003	236.2916	58
361.7166	293.9417	187.0222	62
337.3407	361.9637	212.3552	63
397.3383	391.6698	275.0995	89
212.6977	397.4296	197.8375	92

6.4 Initial Node Data

Table 4 Initial Node Data

Node no.	xd	Yd	G	E	Type
1	102.166	3.833639	0	0.5	'N'
2	85.37524	150.8607	0	0.5	'N'
3	291.7285	242.4021	0	0.5	'N'
4	351.2713	306.8977	0	0.5	'N'
5	318.094	156.2672	0	0.5	'N'
6	309.6087	136.8943	0	0.5	'N'
7	345.8744	178.5242	0	0.5	'N'
8	75.20984	23.03572	0	0.5	'N'
9	88.1282	89.12509	0	0.5	'N'
10	267.23	369.4559	0	0.5	'N'
11	222.1021	309.7797	0	0.5	'N'
12	120.0114	341.702	0	0.5	'N'
13	6.075038	0.292134	0	0.5	'N'
14	76.15722	371.7983	0	0.5	'N'
15	364.5683	43.82225	0	0.5	'N'
16	203.1643	371.473	0	0.5	'N'
17	38.31476	351.0816	0	0.5	'N'
18	377.7863	236.5216	0	0.5	'N'
19	110.3374	74.79533	0	0.5	'N'
20	291.1436	174.0962	0	0.5	'N'
21	141.9345	302.2056	0	0.5	'N'
22	141.6649	252.9034	0	0.5	'N'
23	176.8423	123.2508	0	0.5	'N'
24	350.0036	243.0994	0	0.5	'N'
25	226.1958	239.628	0	0.5	'N'
26	247.6692	339.5051	0	0.5	'N'
27	265.8117	18.15776	0	0.5	'N'
28	307.9608	340.5657	0	0.5	'N'
29	279.1162	375.2162	0	0.5	'N'
30	271.1073	56.92309	0	0.5	'N'
31	110.3124	81.60735	0	0.5	'N'
32	86.68647	385.2044	0	0.5	'N'
33	318.3632	173.8405	0	0.5	'N'
34	328.8082	16.63628	0	0.5	'N'
35	366.538	70.71094	0	0.5	'N'
36	368.9428	107.4144	0	0.5	'N'
37	255.6308	12.69342	0	0.5	'N'
38	133.5811	356.3199	0	0.5	'N'
39	118.7306	163.0645	0	0.5	'N'
40	212.0819	268.4767	0	0.5	'N'

41	214.7827	257.8671	0	0.5	'N'
42	209.4637	197.7457	0	0.5	'N'
43	146.9751	246.3687	0	0.5	'N'
44	270.5574	101.1367	0	0.5	'N'
45	145.5633	114.5202	0	0.5	'N'
46	329.6573	89.09689	0	0.5	'N'
47	149.091	288.3692	0	0.5	'N'
48	114.9124	157.8806	0	0.5	'N'
49	279.9568	301.6306	0	0.5	'N'
50	141.2409	233.231	0	0.5	'N'
51	223.9161	1.743209	0	0.5	'N'
52	17.3795	298.039	0	0.5	'N'
53	255.6283	97.43267	0	0.5	'N'
54	300.1765	356.5316	0	0.5	'N'
55	283.7901	320.1344	0	0.5	'N'
56	259.7991	0.934927	0	0.5	'N'
57	145.0607	46.84334	0	0.5	'N'
58	258.2049	8.085621	0	0.5	'N'
59	388.3289	108.4352	0	0.5	'N'
60	99.96274	20.45498	0	0.5	'N'
61	293.06	100.8456	0	0.5	'N'
62	287.4829	141.808	0	0.5	'N'
63	110.9599	146.4771	0	0.5	'N'
64	371.5992	179.4505	0	0.5	'N'
65	122.1121	250.3741	0	0.5	'N'
66	58.86406	135.894	0	0.5	'N'
67	149.9166	47.05659	0	0.5	'N'
68	240.2595	273.6178	0	0.5	'N'
69	21.5656	36.41644	0	0.5	'N'
70	235.1618	208.726	0	0.5	'N'
71	347.2326	205.2634	0	0.5	'N'
72	3.551328	37.87739	0	0.5	'N'
73	62.19526	32.86026	0	0.5	'N'
74	300.0718	397.3195	0	0.5	'N'
75	47.41371	98.55096	0	0.5	'N'
76	231.4998	90.96634	0	0.5	'N'
77	252.1529	162.691	0	0.5	'N'
78	338.7678	198.8427	0	0.5	'N'
79	233.3454	179.9087	0	0.5	'N'
80	146.4537	131.8463	0	0.5	'N'
81	321.0696	68.5957	0	0.5	'N'
82	86.727	270.41	0	0.5	'N'
83	81.73611	358.0016	0	0.5	'N'
84	18.08999	273.7694	0	0.5	'N'
85	133.2797	148.7171	0	0.5	'N'

86	4.419726	231.5909	0	0.5	'N'
87	388.3822	140.5773	0	0.5	'N'
88	118.5303	359.3251	0	0.5	'N'
89	319.7728	146.8171	0	0.5	'N'
90	201.0603	381.2835	0	0.5	'N'
91	212.1098	297.752	0	0.5	'N'
92	180.8382	37.95115	0	0.5	'N'
93	100.8692	59.97844	0	0.5	'N'
94	50.5104	393.211	0	0.5	'N'
95	160.3906	131.8866	0	0.5	'N'
96	250.496	339.632	0	0.5	'N'
97	297.7151	115.8513	0	0.5	'N'
98	246.8287	320.5072	0	0.5	'N'
99	198.5763	118.2512	0	0.5	'N'
100	186.0584	340.3087	0	0.5	'N'

Chapter 07: Result

Total Performance Evaluation

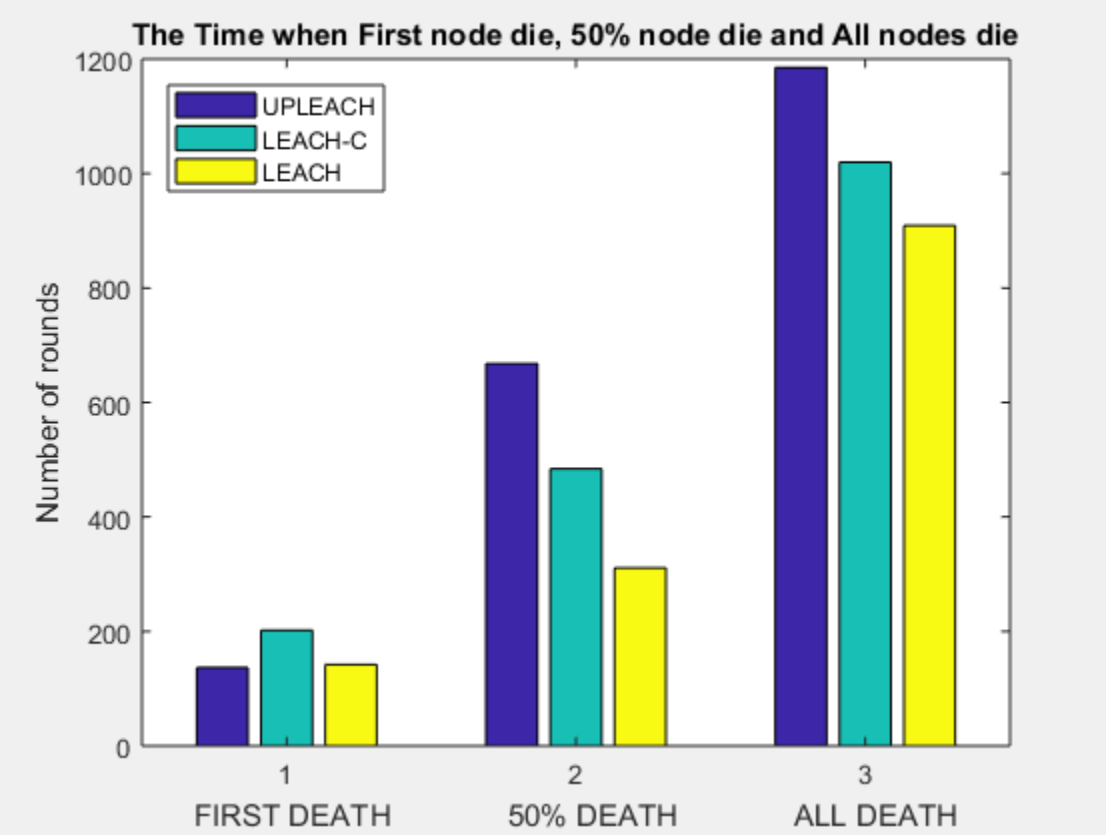

Figure 16 Comparison of 1st, half and all node death among LEACH distributions

Fig 10, it shows that our leach protocol performs well when we compare with "LEACH "and "LEACH-C" routing protocol. 50% nodes are dead around 300, 500 and 700 rounds in LEACH, LEACHC and UPLEACH that 14% better than LEACHC and 23% better than LEACH.

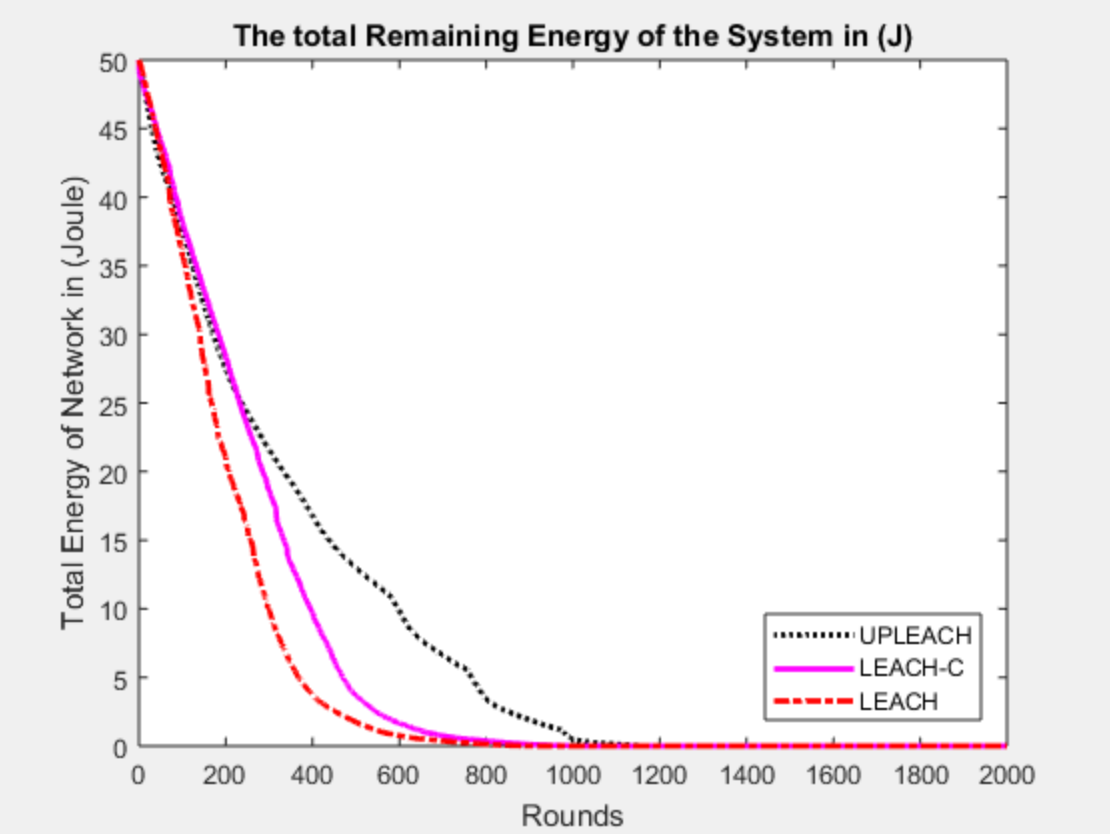

Figure 17 Network residual energy per round

Fig 11, shows the residual energy falling per round. LEACH and LEACHC total network energy is empty around 800 round where UPLEACH total network energy is empty around 1100 round.

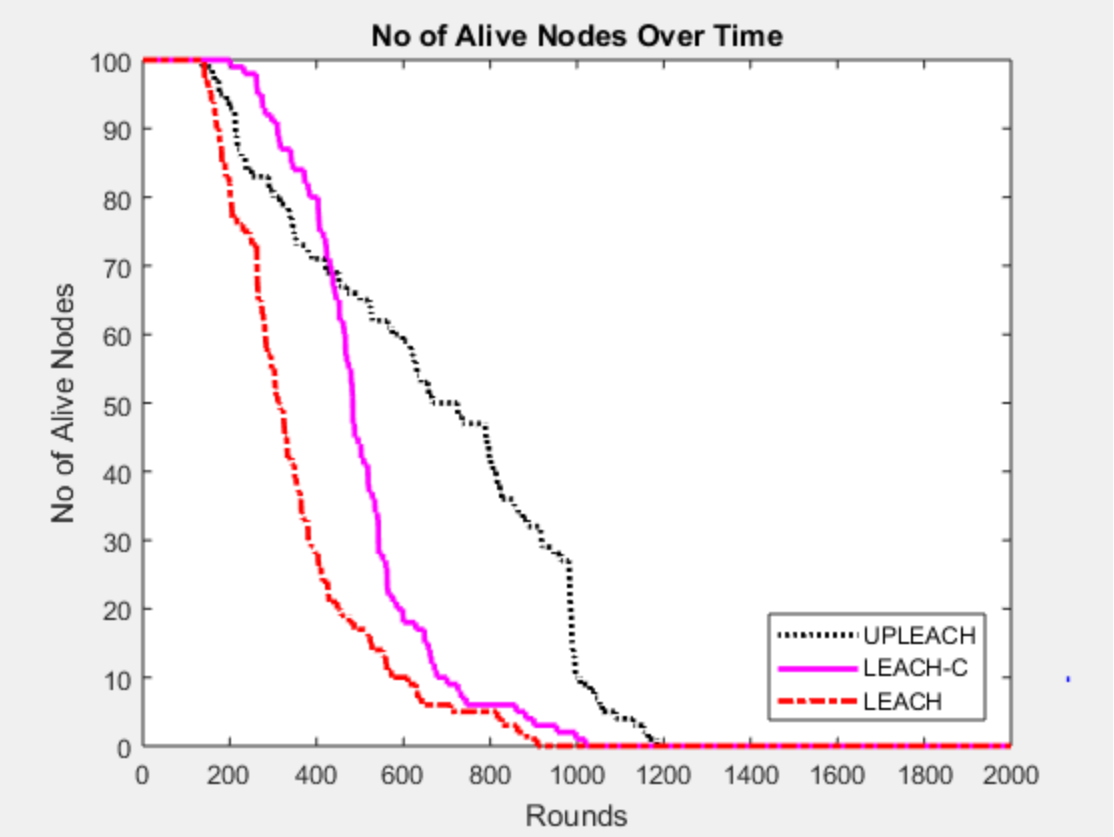

Figure 18 Alive node decreasing per round

The fig 12, discriminates the alive nodes per round rates among LEACH, LEACHC and UPLEACH that after 30 nodes death, UPLEACH outperformes the survivallity till the end.

Chapter 08: Conclusion

In our thesis, we have shown a different approach of cluster head formation techniques. By upgrading the leach protocol, we can see the efficiency in the result. In here our "LEACH" protocol perform higher in terms of energy efficiency and nodes survival ratio. Both the analytical and simulation result show that the extended life time of WSN can be achieved with this novel approach. However, our goal is to achieve an efficient wireless node charging module to enhance the life time of sensor node in future. Moreover, we intend to develop an energy station which recharges the energy nodes that lead to an immortal WSN.

References

[1] W. Heinzelman, A. Chandrakasan, and H. Balakrishnan."Energy-Efficient Communication Protocols for Wireless Microsensor Networks". *In Proceedings of Hawaiian International Conference on Systems Science,* January 2000.

[2] Amit Karmaker; Md. Mahedee Hasan; y,Shafika Showkat Moni and Mohammad Shah Alam."An Efficient Cluster Head Selection Strategy for Provisioning Fairness in Wireless Sensor Networks" *IEEE International WIE Conference on Electrical and Computer Engineering (WIECON-ECE),* December 2016, AISSMS, Pune, India

[3] D. Mahmood, N. Javaid1, S. Mahmood, S. Qureshi, A. M. Memon, T. Zaman, "MODLEACH: A Variant of LEACH for WSNs", *Eighth International Conference on Broadband and Wireless Computing, Communication and Applications*

[4] Fareed, M.S.; Javaid, N.; Ahmed, S.; Rehman, S.; Qasim, U.; Khan, Z.A., "Analyzing Energy-Efficiency and Route-Selection of Multi-level Hierarchal Routing Protocols in WSNs", Broadband, Wireless Computing, Communication and Applications (BWCCA), *2012 Seventh International Conference,* vol., no., pp.626,631, 12-14 Nov. 2012.

[5] V. Mhatre and C. Rosenberg, "Design Guidelines for Wireless Sensor Networks: Communication, Clustering and Aggregation", Ad Hoc Networks, 2(1), 2004, pp.45-63.

[6] M. M. Zanjireh, A. Shahrabi, and H. Larijani," ANCH: A New Clustering Algorithm for Wireless Sensor Networks," *In 27th International Conference on Advanced Information Networking and Applications Workshops (WAINA),* pp. 450-455, 2013.

[7] A. Patra and S. Chouhan," Energy efficient hybrid multihop clustering algorithm in wireless sensor networks," *In International Conference on Communication, Networks and Satellite,* pp. 59-63 Dec. 2013.

[8] W. B. Heinzelman, A. P. Chandrakasan, and H. Balakrishnan, "AnApplication-Specific Protocol Architecture for Wireless Microsensor- Networks," *IEEE Transactions on Wireless Communications,*vol.1, no. 4, pp. 660-670, 2002.

[9] C. Y. Chong and S. P. Kumar, "Sensor Networks: Evolution, Opportunities and Challenges", Proceedings of the IEEE, 91, No. 8, pp. 1247-1256, Aug 2003.

[10] M. Younis, P. Munshi, G. Gupta and S. M. Elsharkawy, "On Efficient Clustering of Wireless Sensor Networks", *Second IEEE Workshop on Dependability and Security in Sensor Networks and Systems,* 2006, pp. 78-91.

[11] S. V. K. a. A. Pal, "Assisted-Leach (A-Leach) Energy Efficient Routing Protocol for Wireless Sensor Networks,*" International Journal of Computer and Communication Engineering,* vol. Vol. 2, 4, July 2013.

[12] M. S. A. A. K. SHAW, "Transmission Time and Throughput analysis of EEE LEACH, LEACH and Direct Transmission Protocol: A Simulation Based Approach," *Advanced Computing: An International Journal (ACIJ),* vol. Vol.3, p. 5, September 2012.

[13] B. N. N. A. W. A. K. Salim EL KHEDIRIa, "A New Approach for Clustering in Wireless Sensors Networks Based on LEACH," *International Workshop on Wireless Networks and Energy Saving Techniques (WNTEST)*, 2014.

[14] M. A. G. Leena Y.Barai, "Performance Evaluation of LEACH Protocol for Wireless Sensor Network," *International Journal of Innovative Research in Advanced Engineering (IJIRAE)*, vol. vol 1, no. 6, 2014.

[15] X. Y. Xiaowen Maa, "Improvement on LEACH Protocol of Wireless Sensor Network," *Trans Tech Publications, Switzerland*, Vols. 47-350, 2013.

[16] G. Horvat, D. Zagar and D. Vinko, "Influence of node deployment parameters on QoS in large-scale WSN," in *3rd Mediterranean Conference on Embedded Computing (MECO)*, Montenegro , 2014

[17] Amit Parmar, and Ankit Thakkar,"An improved modified LEACH-C algorithm for energy efficient routing in Wireless Sensor Networks", *Nirma University Journal of Engineering and Technology*, VOL. 4, N0. 2, JUL-DEC 2015