A Novel Technique, Adaptive Intra Cluster Routing For Wireless Sensor Networks

Developed By:
Raja Adeel Akhtar (242-FAS/MSCS/F05)

Supervised By:
Dr. Abid Ali Minhas

Department of Computer Science
Faculty of Basic and Applied Sciences
International Islamic University Islamabad
2009

International Islamic University, Islamabad

Dated: ----------

Final Approval

This is to certify that we have read the thesis submitted by **Raja Adeel Akhtar, Reg # 242-FAS/MSCS/F05**. It is our judgment that this project is of standard to warrant its acceptance by the International Islamic University, Islamabad, for the Degree of **MS in Computer Science**.

Project Evaluation Committee

External Examiner:
Prof. Dr. Muhammad Abdul Qadir
Dean, Faculty of Engineering and Applied Sciences, Mohammad Ali Jinnah University, Islamabad.

Internal Examiner:
Dr. Muhammad Sher
Department of Computer Science
Faculty of Basic and Applied Sciences
International Islamic University, Islamabad.

17-6-09

Supervisor:
Engr. Dr. Abid Ali Minhas
Associate Professor
(CS&Engg. Deptt.)
Bahria University, Islamabad.

A Dissertation Submitted To

Department of Computer Science,

Faculty of Basic and Applied Sciences,

International Islamic University, Islamabad

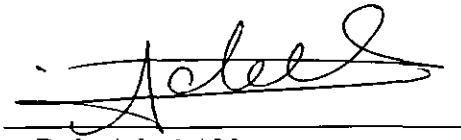
As a partial Fulfillment of Requirements for the Award of the

Degree of

MS in Computer Science

Declaration

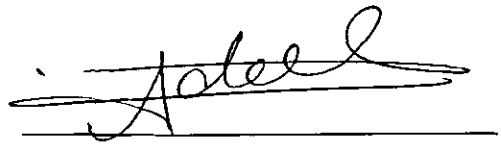
I hereby declare that this Thesis **"Adaptive Intra Cluster Routing for WSN"** neither as a whole nor as a part has been copied out from any source. It is further declared that I have done this research with the accompanied report entirely on the basis of my personal efforts, under the proficient guidance of my teachers especially my supervisor Dr. Abid Ali Minhas. If any of the system is proved to be copied out of any source or found to be reproduction of any project from any of the training institute or educational institutions, I shall stand by the consequences.



Raja Adeel Akhtar
Reg # 242-FAS/MSCS/F05

Acknowledgement

At very first, we bestow all praises, acclamation and appreciation to Almighty ALLAH, the most merciful and compassionate. The most Gracious and Beneficent, Whose bounteous blessings enable us to pursue and perceive higher ideals of life, All praises for His Holy Prophet MUHAMMAD (SAW) Who enabled us to recognize our ALLAH and creator and brought to us a real source of knowledge from ALLAH, The QURAN, Who is role model for us in every aspect of life. Secondly I must mention that it was mainly due to my family's moral and financial support during my entire academic career that enabled me to complete my work dedicatedly. I would like to thanks to my teacher's and especially consider it a proud privileged to express my gratitude and deep sense of obligation to my reverend supervisor Dr. Abid Ali Minhas for his dexterous guidance and kind behavior during the project. I would like thank my brother, and sisters who encouraged me at those moments when I got exhausted. I also would like to say thanks to my truly friends that helped me in every difficulty. I once again would like to admit that I owe all my achievement to my most loving parents who mean most to me, for their prayers are more precious then any treasure on earth.



Raja Adeel Akhtar
Reg # 242-FAS/MSCS/F05

Project In Brief

Project Title:	**Adaptive Intra Cluster Routing for Wireless Sensor Networks**
Organization:	**International Islamic University, Islamabad (IIUI)**
Undertaken By:	**Raja Adeel Akhtar (242-FAS/MSCS/F05)**
Supervised By:	**Engr. Dr. Abid Ali Minhas** **Associate Professor** **(CS&Engg. Deptt.)** **Bahria University Islamabad**
Start Date:	**June 2007**
Completion Date:	**March 2009**
Tools & Technologies:	**TinyOS** **TOSSIM** **MS Office** **TinyViz** **Mote View**
Operating System:	**Linux Red Hat9** **Windows XP**
System Used:	**Intel(R) Pentium(R) M Processor 1.86 GHz** **RAM 1 GB** **80 GB Hard Disk**

Abstract

Scientists have listed down ten emerging technologies that will change the world's future and Wireless Sensor Network (WSN) is one of them. A lot of work is in progress to make this technology more effective by having research in this field. This network is composed of tiny nodes called motes which are battery operated in general. As these devices are energy limited so energy efficiency is one of the most critical parts of this network. To have energy efficient network, efficient routing plays very important role. In my research work, Adaptive Intra Cluster Routing (AICR) is proposed which is the best solution to the problem offered by traditional routing algorithms. Quantitative analysis reveals its energy efficiency over the previously published work. It has also been shown that network lifetime is increased by implementing AICR.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

1. Introduction

Wireless Sensor Network (WSN) is a new technology of ad-hoc networks and is a hot topic for research. As it is a new technology so lot of research is required for its technical growth. Our research work is a knowledge contribution towards the research of this area. In this chapter the topic has been briefly introduced.

1.1 Wireless Sensor Network

"A wireless network consists of tiny wireless devices distributed in the environment having sensors attached with them to monitor the environment is called ***Wireless Sensor Network****".*

In start wireless sensor network was developed for a military based application called battlefield surveillance. But now a day wireless sensor network is used for the development of industrial as well as civilian applications.
Some common applications of WSN are:

- Intruder detection [1]
- Forest fire detection [2]
- Cattle herd [3]
- Battle field [4]
- Fire rescue [5]
- Environmental monitoring [6]
- Condition based maintenance [7]
- Airconditioning sensors [8]

WSN has some essential design issues, where research work is required to make this technology more reliable. These different issues include:

- Routing
- Security
- QoS
- Real time computing
- Hardware development

This network is composed of tiny nodes called *motes* that are capable of performing some processing, gathering sensed information and communicating with other connected nodes in the network. These motes are battery operated and one of the most important factors is energy consumption for such type of network.

Our work is related to routing or network layer based protocol development. The proposed idea is energy efficient protocol for cluster based routing for WSN.

1.2 Problem Statement

As mentioned above energy efficiency is always required for such type of network. So we can say that an energy efficient routing protocol is always required for such type of network. At present direct routing is adopted for small networks and for large network multihop routing is adopted to get energy efficient routing in WSN. But in large network when multihop routing is applied some problem occurs e.g. for large network the load of whole network is pushed towards the closest nodes to the base station so these nodes have more than enough data to transmit towards the base station. As we know that network life time is calculated with the death of first node so in this case the network life time is very less because the closest respective node dies very soon because of having lot of data to transmit towards the base station. On the other hand if in such case direct routing is adopted then the node at the boundary of the network has to put in more energy to transmit its data towards the base station. So there is an esteem need of such protocol that avoids this problem and prolongs the network lifetime.

1.3 Contribution of this Dissertation

In our research work a new intra cluster based routing protocol has been proposed that is energy efficient and prolongs the network lifetime as well. The protocol is "Adaptive Intra Cluster Routing". We have implemented this protocol for intra cluster routing and got positive results which are mentioned in chapter 6.

1.4 Dissertation Organization

This thesis is organized as follows. Chapter 2 describes the Wireless Sensor Networks in detail, respective hardware and software. In Chapter 3, related work in same area has been presented. Chapter 4 gives the detailed description of proposed solution for the problem identified. Chapter 5 presents description of TOSSIM simulator and its implementation details. In chapter 6, the proposed algorithm has been evaluated with the help of comparison with existing techniques by considering different performance metrics. Finally in Chapter 7, conclusions and future work have been mentioned.

Chapter 2

Routing in WSN

2. Wireless Sensor Network

Wireless Sensor Network (WSN) is an emerging technology and is very hot research area for wireless communications. At an abstract level function WSN can be divided into two main portions, the sensing and the computing. In sensing side there are small, cheap and efficient nodes called sensor nodes. These tiny nodes sense the area of interest and get required information for the user. On computing side there is a Base Station (BS) which is very high profile system that has high processing capabilities, fast computing memory and centrally powered system. The whole scenario is depicted in Fig1. The sensor nodes are wireless and battery powered, which make them very complex to handle. The wireless nature makes routing as a real challenging task and the limited power adds more in its complexity. So the research is going on to develop such protocols that provide robust wireless communications, which are energy efficient and provide low latency.

2.1 Architecture

The sensing nodes are deployed in the area of interest and they sense the required information like temperature etc and continuously send that sensed data to the BS. On the BS different analytical and statistical formulas are applied on the collected data and finally required information is extracted out of it. So we can say that the main challenges in WSN are the communication among the nodes with least energy consumption and efficient routing. Different protocols have been developed to get efficient solutions for these challenges.

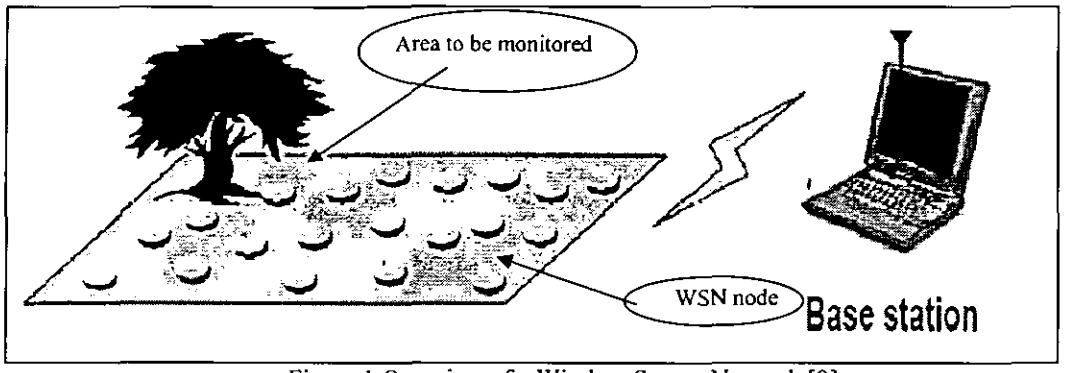


Figure 1 Overview of a Wireless Sensor Network [9]

2.1.1 Sensor Hardware

A sensor node consists of five components, including *sensing hardware*, *processor*, *memory*, *power supply* and *transceiver* [10]. A typical node with its components is presented in Fig 2.

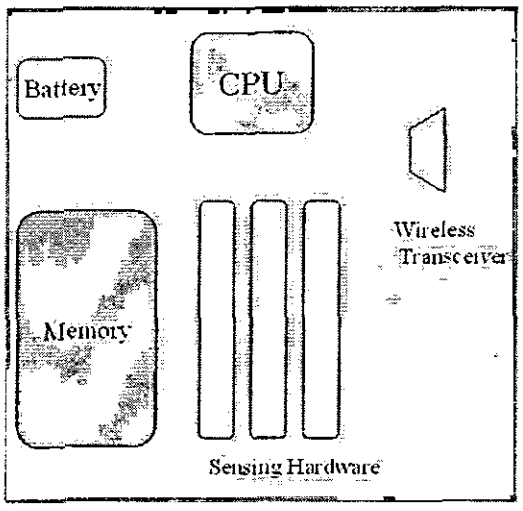


Figure 2 Components of Sensor Node [11]

2.1.1.1 Sensor Node

Now we will discuss the components of a WSN node in detail. As mentioned above the five components of a typical node now we will see these components one by one in detail.

Microcontroller

Microcontroller plays a role of a CPU in a sensor node. Microcontroller is used as a central controller in a node, other options for a central controller are: General Purpose Microprocessors, Field Programmable Gate Array (FPGA), Digital Signal Processors (DSP) and Application Specific Integrated Circuit (ASIC). But the Microcontroller is the best choice for a sensor node because of its flexibility of connection (with other devices), less power consumption and it is easily programmable. One of the big advantages is of its power awareness. These devices go to sleep mode rather than keeping itself in an idle state. The power consumption of general purpose Microprocessor is more so it is not suitable for sensor node as energy is one of the major constraints of this type of network (WSN). FPGA is also reprogrammable but it takes more energy and time that are also big constraints for this device (sensor node). For a broadband wireless network DSP is the right choice but in WSN the wireless communication is much simpler, easy modulation is involved and signal processing of sensed data is not very complicated. ASIC is used for a customized processor design therefore it provides a hardware development facility which is not required for WSN, here a software solution is required.

Transceiver

Transceiver is a dual purpose device used for wireless communication. As it is named "tran" means transmitter and "sceiver" means receiver. So a Transceiver is that device which transmits and receives the signal in a wireless communication. First we see how

many types of wireless transmission media are available. Radio Frequency (RF), Optical Communication (LASER) and infrared are different types of wireless transmission media. Laser requires less energy but for successful transmission line of sight is required and it is also a sensitive media which means that atmospheric conditions effect the transmission as well. On the other hand infrared needs no line of sight but the broadcasting range is limited so it is only useful for small coverage area. For WSN the most appropriate media for communication is RF. The frequency range is from 433 MHz to 2.4 GHz. The Transceiver has four modes of operation namely Transmit, Receive, idle and sleep.

External Memory

One of the most important components of a node is memory, where the data is stored. While talking about the memory we should consider the cost and energy constraints in front of us as well. On the basis of energy constraint the different kinds of memories are: on-chip memory, off-chip RAM and Flash. Flash memory is widely used due to its low cost and huge data storage capacity. The storage of data in memory is of two types, for the storage of user data "User memory" and for the storage of programming data "Program memory" is available.

Power Source

A sensor node has to perform three basic tasks which are sensing, processing and communication with other nodes. So the maximum power consumption is done in these three tasks. The more power consuming task is done by the transceiver of the node while communicating with other nodes. On the other hand energy consumption is not as much in sensing and processing tasks. This is justified by calculating the power consumption by all three tasks and according to [12] "energy utilization for transmitting 1 Kb at distance of 100 m is same as for executing 3 million instructions by 100 million instructions per second". Different sources of power are available, it can be battery or capacitor based. For sensor node battery is the main source of power supply. The latest model "micaz" motes have two 1.5V AA batteries as power source. Batteries are either chargeable or non-rechargeable. The different electrochemical based electrodes are available like NiCd, Ni2n, Nmh and Lithium-Ion, on the basis of which classification of the batteries is done. The latest models of batteries are capable of recharging from solar or vibration energy (from air). In energy constraint based devices there must be energy saving mechanism. Two types of energy saving policies are available: Dynamic Power Management (DPM) and Dynamic Voltage Scaling (DVS) [12]. In DPM those parts are shutdown which are not currently in use. In DVS on the basis of non-deterministic work load the power levels varies. To get the quadratic reduction for power consumption the voltage with frequency is varied by this scheme.

Sensors

The major objective of WSN is to sense the required information, process it and transmit it. So one of the major tasks of this network is sensing. Here small hardware devices

called sensors (usually called sensor board) are used, continuously producing measurable information in the area of interest. This sensed analog signal is first digitized by Analog-to-Digital converter that is sent to microcontroller for processing. Sensor node should be small, energy aware, be autonomous, unattended and be adaptive to any change.

2.1.1.2 Mote Hardware

The size of a sensor node is very small. There are different types of nodes used for research e.g. mica2, micaz, micadot etc. The *mica2 mote* have a shape of rectangular block, with size 2.25 × 1.25 × 0.25 (in inches), weights about 18 grams as shown in Fig 3, the *mica2dot mote* have circular shape with size 1.0 × 0.25 (in inches) and weights 3 grams [10].

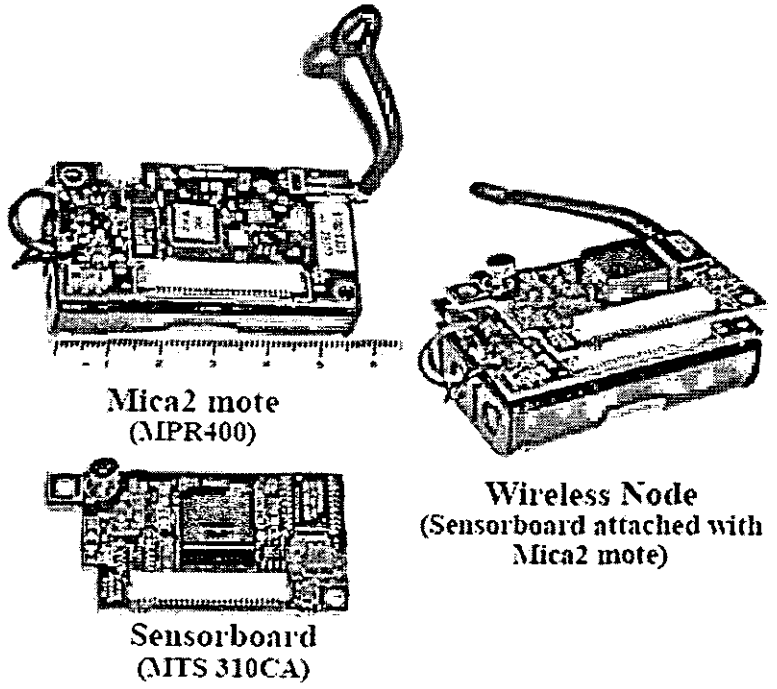


Figure 3 Mica2 mote, its sensor board and their combination [10]

Mote hardware is developed by Berkeley and now one can get proper series of sensor nodes developed by Berkeley. Fig 4 shows some well known models of sensor nodes. The latest two are Micaz and MicaDot motes. Fig 4 shows the complete specifications of mote models [12].

Sensor Node Name	Microcontroller	Tranceiver	Program+Data Memory	External Memory	Programming	Remarks
BTnode	Atmel ATmega 128L (8 MHz @ 8 MIPS)	Chipcon CC1000 (433-915 MHz) and Bluetooth (2.4 GHz)	64+180 K RAM	128K FLASH ROM, 4K EEPROM	C and nesC Programming	BTnut and TinyOS support
Dot	ATMEGA163		1K RAM	8-16K Flash	weC	
Eyes	MSP430F149	TR1001		8 Mbit		PeerOS Support
IMote	ARM core 12 MHz	Bluetooth with the range of 30 m	64K SRAM	512K Flash		TinyOS Support
IMote 1.0	ARM 7TDMI 12-48 MHz	Bluetooth with the range of 30 m	64K SRAM	512K Flash		TinyOS Support
KMote	TI MSP430 microcontroller	250 kbit/s 2.4 GHz IEEE 802.15.4 Chipcon Wireless Transceiver	10k RAM	48k Flash		TinyOS and SOS Support
Mica	Atmel ATMEGA103 4 MHz 8-bit CPU	RFM TR1000 radio 50 kbit/s	128+4K RAM	512K Flash	nesC Programming	TinyOS Support
Mica2	ATMEGA 128L	Chipcon 868/916 MHz	4K RAM	128K Flash		TinyOS, SOS and MantisOS Support
MicaZ	ATMEGA 128	802.15.4/ZigBee compliant RF transceiver	4K RAM	128K Flash	nesC	TinyOS, SOS, MantisOS and Nano-RK Support
Rene	ATMEL8535	916 MHz radio with bandwidth of 10 kbit/s	512 bytes RAM	8K Flash		TinyOS Support
SenseNode	MSP430F1611	Chipcon CC2420	10K RAM	48K Flash	C and NesC programming	GenOS and TinyOS Support
T-Mote Sky	Texas Instruments MSP430 microcontroller	250 kbit/s 2.4 GHz IEEE 802.15.4 Chipcon Wireless Transceiver	10k RAM	48k Flash		Contiki, TinyOS, SOS and MantisOS Support
FireFly	Atmel ATmega 1281	Chipcon CC2420	8K RAM	128K FLASH ROM, 4K EEPROM	C Programming	Nano-RK RTOS Support
Iris Mote	Atmel ATmega 1281	Atmel rf230	8K RAM	128K FLASH ROM, 4K EEPROM	C and NesC Programming	TinyOS Support

Figure 4 List of Well Known Sensor Nodes or motes [12]

2.1.2 Sensor Software

Up till now we have seen the hardware architecture of WSN. In this section we will discuss the operating systems available for WSN.

2.1.2.1 Operating System

Different operating systems are available to implement and develop routing algorithms for WSNs. Dulman and Havinga [13] have proposed an operating system for a European project named as *Energy Efficient Networks (EYES)*. They have included the characteristics of small size, power awareness, distribution and reconfiguration in their project.

Another operating system *"Contiki"* for WSNs is presented by Dunkels *et al.* [14] implemented in the C language [15]. This project is developed with the contribution of European Research Consortium for Informatics and Mathematics (ERCIM) [16]. Contiki is open source operating system for embedded systems, it is portable and supports multi-tasking. It has been tested for a platform *Embedded Sensor Board (ESB)*. When this OS was released it was having priority over TinyOS [18] in a way that, "in TinyOS, nesC language is used which connects different components together to build a system. These components are statically linked with kernel. Once linking is done system modifications are not allowed. On the other hand Contiki allows the modifications at run time [17]. *Contiki network simulator* has also been developed by the authors.

Another operating system *Sensor Network Operating System (SOS)* is presented by Han *et al.* in 2005 [18]. According to [18], TinyOS was having a problem of memory protection with it. Authors have also mentioned that some interrupt concurrency bugs are caught by the nesC compiler. Some improvements in this regard have done in their work. TinyOS is widely used if compared with SOS. Further details of SOS e.g. presentation, description etc. can be found from its web address [19]. In TinyOS latest version lot of improvements have been made. In [20] Lin Gu has proposed a new technique for the development of an operating system with low cost and high performance perimeters using t-kernel.

In our research work we have used the TinyOS because of its flexibility. TinyOS is open source software freely available on internet for application development of wireless sensor networks [17].

Chapter 3

Literature Survey

3. Literature Survey

Since last few years lot of research has been done in the field of WSN and still the research is going on to make this field more and more effective. We have studied some of latest work done published in this area. In this chapter we discuss the literature which has been studied to get the updated knowledge of this area.

3.1 Previous Work

WSN has many domains for research but the main domain of research in WSN is routing. Researchers have worked a lot for the development of routing protocols for wireless sensor networks. In [21] authors have discussed the different types of routing models for WSN that are one hop model, multihop model and cluster-based model. Researchers have presented routing protocols for all of these models. The proposed protocol is related to cluster-based model. A little description of one hop, multihop and cluster based routing is as follows:

One Hop Model
This is a simple model that uses direct sending. The sensor nodes send the data directly to the base station.

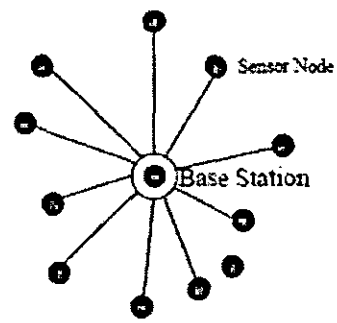


Figure 5 One hop model [21]

Multi-Hop Model
In this model nodes do not send data directly to the base station but they chose their neighbors to forward data toward the base station. In this model less energy is utilized depending upon the topology of WSN.

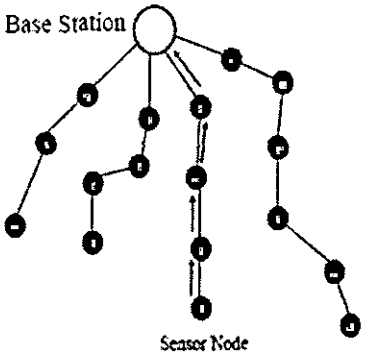


Figure 6 Multi-hop model [21]

Cluster-Based Model

In this model whole network is grouped into clusters. Each cluster has a cluster head that gets the sensed data from cluster member nodes and then sends it to other cluster heads or direct to the base station

.

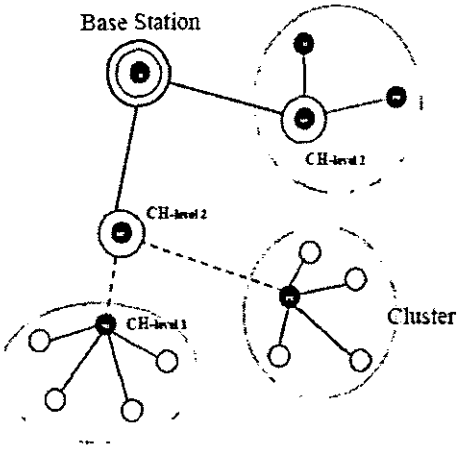


Figure 7 Cluster based model [21]

Researchers have developed many routing protocols for above mentioned models but still there are lots of advancements to be done.

A very early and classic routing protocol named LEACH (Low-Energy Adaptive Clustering Hierarchy) is developed by Heinzelman et al [22]. In this paper, different routing protocols are discussed and their energy utilization is analyzed. Then a new technique for the selection of Cluster Head is proposed. Finally this has been proved by statistical analysis that by rotation of CH role in side a cluster the energy utilization is reduced and network lifetime is increased as well. So by evenly distributing the CH role

in a cluster load balancing is achieved. Simulation shows that LEACH is capable of having factor 8 reduction in energy consumption compared with other protocols.
Another cluster formation technique is presented by Bandyopadhyay [23] in which author has presented a distributed, randomized clustering algorithm to organize the sensors in a wireless sensor network into clusters [23]. Author has saved the energy by limiting the CH advertisement to k hops thus unnecessary advertisements have been avoided in this work.

Another research work is carried out in [24], in which authors considered hybrid sensor network. A hybrid network is composed of two types of nodes: basic static nodes and mobile CH. In this research work cluster routing is adopted. At lower layer basic static nodes sense the data and send it to mobile CHs. The hybrid network is much better than homogenous network for energy utilization. To maximize the lifetime of the network a dynamic positioning technique for position the CH is proposed and is showed that by getting better location for a CH prolongs the lifetime and causes a balanced network. An overview of two layered hybrid sensor networks is depicted in Fig 8.

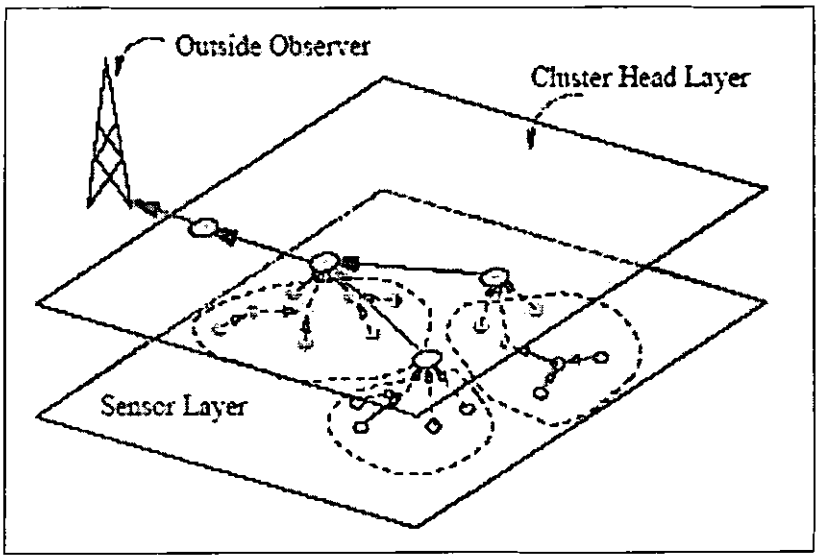


Figure 8 A two Layered Hybrid Sensor Network [24]

A novel distributed data gathering algorithm for wireless networks has been proposed by Goyeneche et al [25]. A technique for data gathering is presented for WSN, in which each node sends its data to its neighbors and so on until data reaches to such node that has already sent its data to the BS. That node will become CH for the sending node.

Israr in [26] proposes a new cluster based routing technique that covers the energy consumption and load balancing issues of WSN. In this algorithm the node whose coverage area is covered by its neighbors becomes their temporary CH. Thus by using these temporary CHs two layered communication is achieved. At bottom layer nodes send data to respective temporary CH and at top layer the temporary CHs adopt multihop routing to deliver the data to the BS. Performance analyses show that the proposed

technique is more energy efficient and also performs the load balancing in the network. Fig 9 shows the process of formation of temporary cluster head on the basis of coverage area.

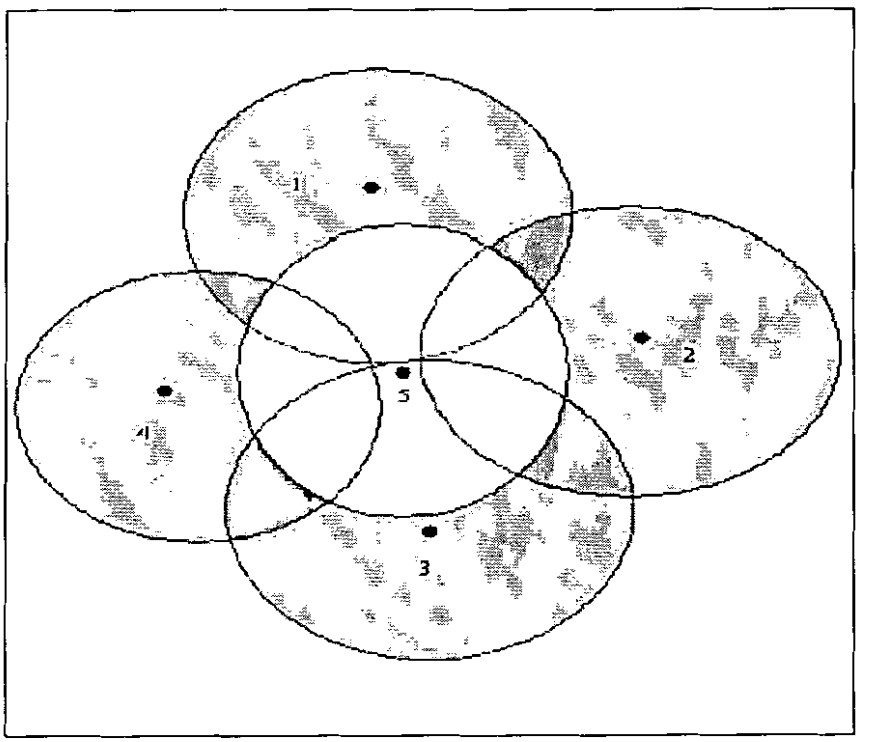


Figure 9 Formation of Temporary CH [26]

Another energy conservation model is given by Anu Tuan Hoang and Mehul Motani [27] in which they have considered the broadcast nature of sensor nodes. When ever data is transmitted the nodes in coverage area hears it even if it is not for them. By using this property one node can compress its own data by overhearing other's transmission. Thus energy conservation is achieved. A flow of collaborative broadcasting and compression in WSN is shown in Fig 10.

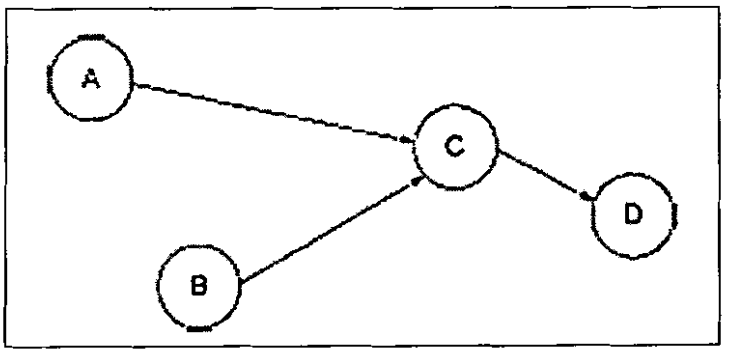


Figure 10 Collaborative Broadcasting and Compression in WSN [27]

LiLi in [28] proposes an Energy Efficient Clustering Routing (EECR) algorithm for WSN. Cluster routing is adopted in this research work. The algorithm creates clusters and selects the CHs on the basis of weight values. This leads to a balanced network with less energy consumption routing that causes increase in the lifetime of the network.

Tao Wu and Subir Biswas [29] have presented a Self-Reorganizing Slot Allocation (SRSA) mechanism. Cluster routing is adopted but this research work is related to MAC layer protocol. The aim of this work is to reduce inter cluster TDMA interference. Secondly a mechanism is presented which is feedback based adaptive reorganization of slot allocation on MAC layer that also reduces the inter cluster interference. The proposed idea is simulated and compared with TDMA-over-CDMA, TDMA with random slot allocation and CSMA MAC protocols.

A new adaptive cluster routing algorithm has been presented by Irfan Ahmed et al [30]. Authors have presented a routing protocol CIDRSN (Cluster ID based Routing in Sensor Networks). Cluster routing is adopted in this research work. Cluster ID based routing is adopted and cluster size adaptation is proposed in it. In this algorithm Cluster ID is used as next hop rather than CH-ID in routing table. This eliminates the cluster formation process in each round. Cluster formation is only carried out in start thus reduces the energy consumption and increases the network lifetime to about 16%. The results are compared with traditional hierarchical routing protocols and got the better results.

3.2 Limitations in Literature Survey

After going through all of the above research works, we found a common problem or deficiency that needs to be solved. Researchers have done work either on multihop routing or direct hop routing in all of the above research works and in some cases both techniques are not feasible to be adopted. In case of using the traditional routing technique like Direct or MultiHop in large network size the energy and network lifetime are badly effected.

3.2.1 Limitations of Direct Routing

While considering large network size, if mode of routing is selected as direct routing then all nodes will send their data directly to CH. The node at maximum distance to CH will have to put in more energy to send its data to CH. So after sending very small amount of data packets its death will occur which leads to small network lifetime. So adopting direct routing in large network size is not feasible.

3.2.2 Limitations of MultiHop Routing

If the mode of routing is selected multihop in large network size, then network lifetime will be less. In multihop routing the nodes closer to the CH will get maximum load of the network because they have to send their own data as well as data from other nodes. If this scenario is considered in large network size, then huge number of data packets will be

routed through these closer nodes. This will drain the energy levels of the closer nodes very quickly. So multihop routing in large network size is also not feasible.

3.3 Objectives

It means that there is a room to work on adaptive routing. So an adaptive routing protocol is required that will fill the existing space in this field. Our major objectives for the development of our new adaptive routing technique will be:

Development of such routing technique that:

1. Will be suitable for large network size.
2. Should be adaptive routing.
3. Should be better then traditional routing techniques.
4. Should be energy efficient.
5. Should prolong the network lifetime.

Chapter 4

Proposed Intra Cluster Routing Algorithm for WSN

4. Proposed Adaptive Intra Cluster Routing

The proposed routing technique is a complete routing solution for the above stated problem for wireless sensor networks. The proposed technique contains all the essentials of relevant field in it. This chapter describes the design goals and proposed technique.

4.1 Design Goals

We have designed "Adaptive Intra Cluster Routing" while keeping in mind the following goals:

Development of such routing technique that:

1. Will be suitable for big network size.
2. Should be adaptive routing.
3. Should be better then traditional routing technique.
4. Should be energy efficient.
5. Should prolong the network lifetime.

4.2 Adaptive Intra Cluster Routing

It has been discussed that for WSN, the network life time is a major challenging issue. Keeping this point in consideration a solution for above mentioned problem is proposed in this research work. The solution is "Adaptive Intra cluster Routing". In this algorithm each node decides the mode of routing e g. whether to route data using direct routing or by multihop routing. Finally by adapting this technique the network lifetime is prolonged, energy consumption is reduced and number of packet sent in the network is increased. This is a solution for large networks, because in small networks we can easily use direct routing. The algorithm works in three phases that are:

1. Cluster Head Selection
2. Cluster formation
3. Intra Cluster routing

4.2.1 Cluster Head Selection

The scope of this thesis is intra cluster communication. For CH selection and Cluster formation any energy efficient algorithm can be employed. In our work CH selection is carried out in a way that in start all nodes send their energy levels and their respective IDs to the BS (Base Station). The BS selects the CHs (Cluster Heads) on the basis of energy levels of the nodes, the ID and the geographical location. The BS is well known to the topology and about the geographical area. We are assuming that the network deployment is manually that's why BS is well informed about the locations of the nodes.

Then this selection is informed to the respective CH by unicast messages. CH selection flow diagram is presented in Fig 11.

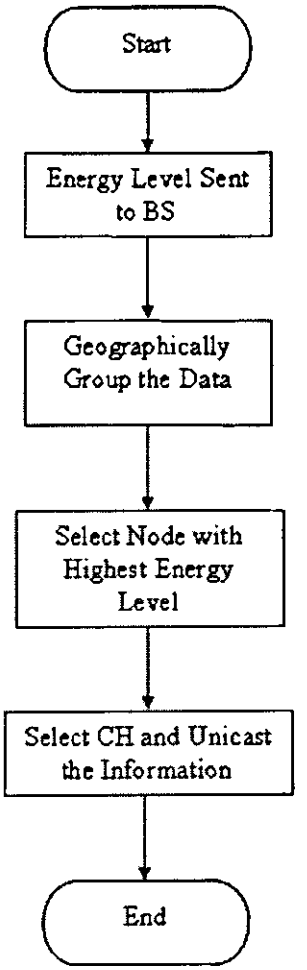


Figure 11 Flow diagram of CH selection

4.2.2 Pseudo Code for Cluster Head Selection

START

```
INITIALIZE
        Destination ID=Destination Address
        Source ID=Source Address
        Energy=Energy Level of the Node
        Data=usable information
        High Level=Max Energy Level
```

```
        Least ID
```

// End of Initialization

BS Function

```
    //Query Nodes about Hello Packets
    Call Node Function

    Group Data according to Geographical Locations
    IF (Energy = High Level AND Source ID = Least ID)
        CH = Source ID
        Data="you are selected as CH"
    ELSE
        Discard Data
ENDIF
    //Unicast Information to Selected CH
    Destination ID = Source ID
    Send Data to Destination ID

END
```

Node Function

```
    Read Energy Level
    Energy= Energy Level
    Calculate Destination ID
    Destination ID= BS
    Data = (Energy Level + Source ID)
    Send Data to Destination Address
```

4.2.3 Cluster Formation

Selected CHs broadcast their status as CH to the network. The nodes after receiving this broadcast beacon calculate their respective distances on the basis of RSSI (Received Signal Strength Indicator) value and select their respective CH. The nodes then send a join request to their CH. The CHs select the nodes for their clusters. In this way clusters are formed. Fig 12 shows the flow diagram for cluster formation process.

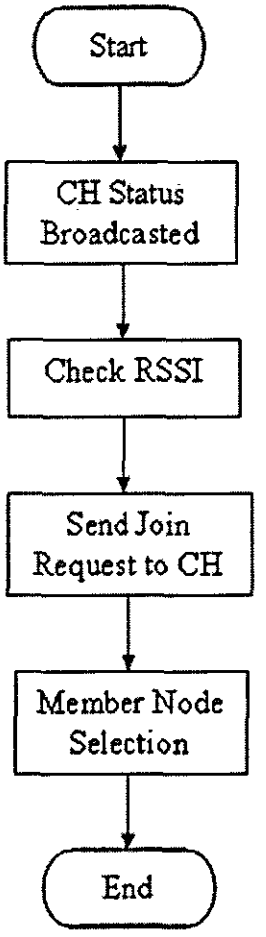


Figure 12 Flow diagram of Cluster Formation

4.2.4 Pseudo Code for Cluster Formation

START

```
INITIALIZE
        Source ID=Source Address
        Destination ID=Destination Address
```

```
        High Data=High Value
        Load= Number of Nodes in a Cluster
```

// End of Initialization

```
IF (Source ID = CH) THEN
     Broadcast Info "I am CH join me"
     Call CH Function
ELSE
     Call Node Function

ENDIF
```

NODE FUNCTION

```
Get Broadcast info from CH
Calculate RSSI Value

IF (RSSI >= High) THEN
     Destination ID=CH
     Send ACK to Destination ID

ELSE
     Discard Data

ENDIF
```

CH Function

```
Get Join Request from Source ID
Check Load of Cluster

IF (load is Less) THEN
     Include Source ID in Cluster
     Destination ID=Source IS
     Send ACK to Destination ID
ELSE
     Send "Load in Cluster is High"


ENDIF
```

END

4.2.5 Intra Cluster Routing

Each member node knows its distance with CH based on calculated RSSI value. Now here each node takes a decision whether to route data using direct sending or to use multihop routing. If the RSSI value is greater that means distance is less, on the other hand is RSSI value is less that means the distance is greater. If distance is less than pre determined value then the node adopts direct routing and if distance is greater than a pre determined value, the node adopts multihop routing. Each node does this calculation in start and then start sending its respective data towards CH. On each node this check is carried out to select the mode of routing. The energy level of each node is also considered while routing. So after sending each packet towards the CH the energy level is checked. As the energy consumed by the transceiver of the node thus energy decrement is carried out by the power of the transceiver at which it is sending the packets. The nodes are equipped with dynamic transceivers. That means the power of transceiver is set dynamically according to the distance the signal travels. If the parent node toward which the packet is being sent is at large distance, the more energy will be consumed and if parent/receiving node is near so comparatively less energy will be reduced.

The parent node selection procedure is itself a big task in multihop routing. In our algorithm the shortest path is selected towards the CH. For shortest path selection each node maintains a neighbor table and by using this information each node calculates the shortest path toward the CH. The neighbor table is updated after each 10 seconds.

So the direct routing is started as the simulation starts but the nodes adopting multihop routing starts packet sending after some time, first they get the neighbor information and when this information becomes mature then neighbor table is consulted for routing decision of packet transmission. This is why while simulating, in start the nodes using direct routing start data transmission immediately but on the other hand, nodes using multihop routing start data transmission after some interval of time.

Implementing and simulating this algorithm the overall network life time is increased, energy consumption is decreased and number of packet sent in the network is increased as well. The flow diagram of the proposed idea is depicted in Fig 13. In the flow diagram the value of x is pre determined value as a threshold value of distance.

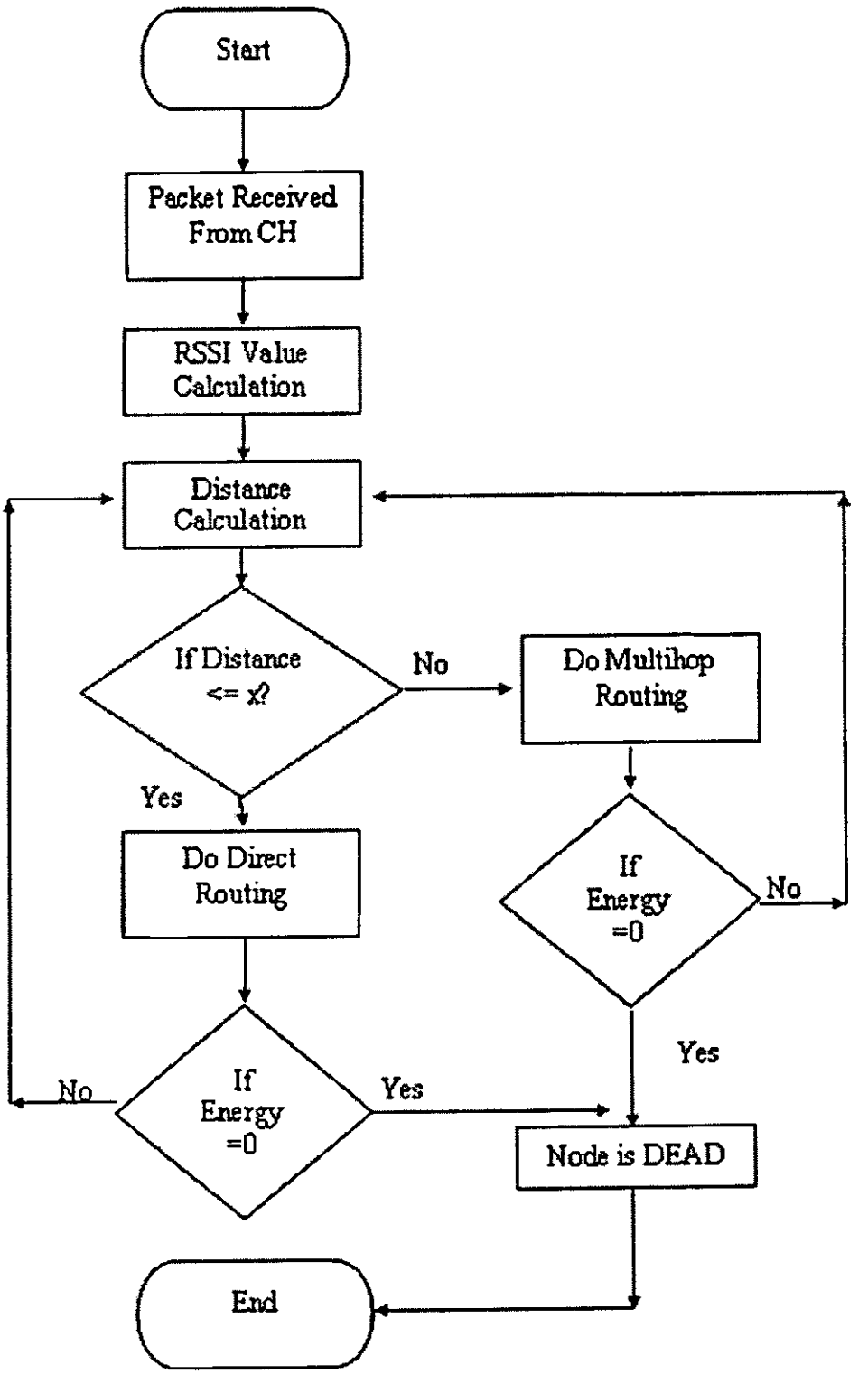


Figure 13 Flow Diagram for Adaptive Intra Cluster Routing

The flow diagram in Fig 14 shows the flow of our algorithm at abstract level. In start the CH sends the beacon packet to the member nodes of the cluster, on the basis of RSSI value the node calculates the distance. If the RSSI value is less that means the signal strength of received packet is less so the distance is greater from the CH. On the other hand if the RSSI value is large then the distance is less. Hence we can say that the RSSI value is inversely proportional to the Distance, the greater the RSSI value smaller the distance or vice versa.

$$\text{Distance} \propto 1/\text{RSSI}$$

The calculated distance value is then compared with a pre determined value threshold value for distance. In our case that threshold value is set to 25 for simulation. This value is obtained after going through a number of simulations. If the calculated distance value is less than or equals to 25 then the mode of routing will be direct routing and if the calculated distance value is greater than 25 then the mode of routing will be multihop routing. Here the concept of adaptive routing arises, the nodes will perform the routing decision at their own.

While routing when ever the energy level reaches to a critical level the node stops sending data and will be considered to be dead. At this point the number of packets sent to the CH is calculated. Network lifetime is calculated with the death of first node in the network.

4.2.6 Pseudo Code for Intra Cluster Routing

```
START

    INITIALIZE
            Source ID= Source Address
            Destination ID= Destination Address
            Parent ID= Parent Address
            Data, Energy Level
// End of Initialization
    Get Sensed Info
    Data= Info
    Select CH

    IF (Distance < 25) THEN
        Call Direct Routing
```

```
     ELSE
          Call MultiHop Routing
```

Direct Routing

```
     Sense required info
     Data =Info
     Destination ID= CH

     WHILE (Energy Level ≠ 0)
          Send Data to Destination ID

     END WHILE

     Broadcast "Node is Dead"
```

MultiHop Routing

```
     Sense required Info
     Data =Info
     Calculate Parent ID for data Delivery
     Destination ID= Parent ID

WHILE (Energy Level ≠ 0)
          Send Data to Destination ID

END WHILE

     Broadcast "Node is Dead"


END
```

Chapter 5

Simulation Details

5. TinyOS implementation details

In this chapter the implementation details of our research work has been discussed. We have used TinyOS as working environment for WSN. TOSSIM simulator is available with TinyOS operating system.

5.1 TOSSIM Simulator

In this section we discuss the TOSSIM simulator in detail. A simulator is "a device, instrument, or piece of equipment designed to reproduce the essential features of something" [31]. Before going toward real networks one should do the simulation process because by using simulation we can easily debug the application. In case of WSN, the real motes are very costly and therefore the availability of this hardware is not very common for a research work in our environment. Very few universities are having real motes for research work. The price of a micaz mote is RS.50, 000 and if we need a WSN of 50 nodes then you can calculate the total cost of such network. Then the only option is simulation of one's idea. There are many simulators available for wireless networks for example *Avrora* [32], *ATEMU* [33], *EmStar* [34], *SWANS* [35], *QualNet* [36], *OPNet* [37] and SENSE [38] but *TOSSIM* [39] is preferred due to its excellent features.

"To simulate an application built for TinyOS, TOSSIM simulator is available" [40]. Instead of compiling any application on real motes one should first compile and run it in TOSSIM. The major advantage of TOSSIM is that you can simply burn your code in motes without any change after compiling it in TOSSIM. The main reason of using TOSSIM is that user can easily debug the code which is nearly impossible without TOSSIM.

TH-5611

TOSSIM is a highly profiled and powerful simulator for TinyOS. We can simulate up to a network of 1000 nodes easily. As mentioned above the cost of a single mote is RS 50, 000 so TOSSIM is really a cost effective solution for a WSN simulation. TOSSIM provides high level fidelity simulations. TOSSIM mainly focuses on simulation rather than simulating real world. TOSSIM like other simulators makes some assumptions and focuses some special behaviors. Some of the characteristics of TOSSIM are briefly discussed below.

Fidelity

Fidelity means how much a device is accurate. A reliability of any simulator is gauged by its fidelity. TOSSIM simulates the behaviors of TinyOS at very low level. At bit level the network is simulated. Every interrupt and ADC is captured by TOSSIM.

Time

In TOSSIM, time is kept same as CPU clock of mica platform that is 4 MHz. The code in TOSSIM runs instantaneously without any preemption.

Models

TOSSIM is a flexible simulator. In some scenarios it does not provide some functionality, means some real world models are not provided by it. But by connecting some external tools required models can be achieved and used for the perfection of a given simulation. There is no effect on the efficiency of the simulator by connecting to any other external tool. Some models are discussed below with their respective tools.

Radio

Radio model is not provided is TOSSIM, but "LossyBuilder" a java based tool, can be used to generate the desired radio model. This will generate an input file for TOSSIM which will be given at run time, will cause TOSSIM to add radio model while simulating the specific application. Propagation, delay and bit corruption can be achieved by this tool just like real world. The loss probability of each packet is independent.

Power/Energy

The energy consumption is not directly modeled by TOSSIM. But energy consumption parameter can be simply added for any component, and after the simulation using that parameter the over all energy consumption of that component can be calculated.

Building

TinyOS implementation of any protocol is required for its simulation because TOSSIM directly builds from the code. Apparently it is difficult task to do but one can simply run that code on real motes with no change.

Imperfections

Nothing is 100% in this world and TOSSIM also has some limitations with it. As discussed before TOSSIM run the simulation instantaneously so no preemption is involved during the simulation. But in real motes there can be preemption by any interrupt occurrence, so the simulation never stops due to no interrupt handling in TOSSIM.

Networking

40Kbit RFM networking stack is simulated by TOSSIM. It also includes MAC, timing, encoding and acknowledgements.

Authority

Performance analysis of the algorithms can be done through TOSSIM. But the results of respective simulation are not so much authoritative. TOSSIM can tell you high data loss between two algorithms but like real mote the data loss scenario can not be perfect.

5.2 How to Compile an Application

The code written in nesC language can be compiled in any version of TOSSIM. For compiling an application open cygwin shell, go to the application directory and write "make" command in the shell. It will compile the application with all its error and warnings. You have to specify the platform for which you want to compile the application as shown in Fig 14. The valid platforms are *mica, mica2, mica2dot* and *pc*. For the compilation of simulation different keyword are used, in TinyOS 1.x "*make pc*" but in TinyOS 2.x "*make sim*" is used.

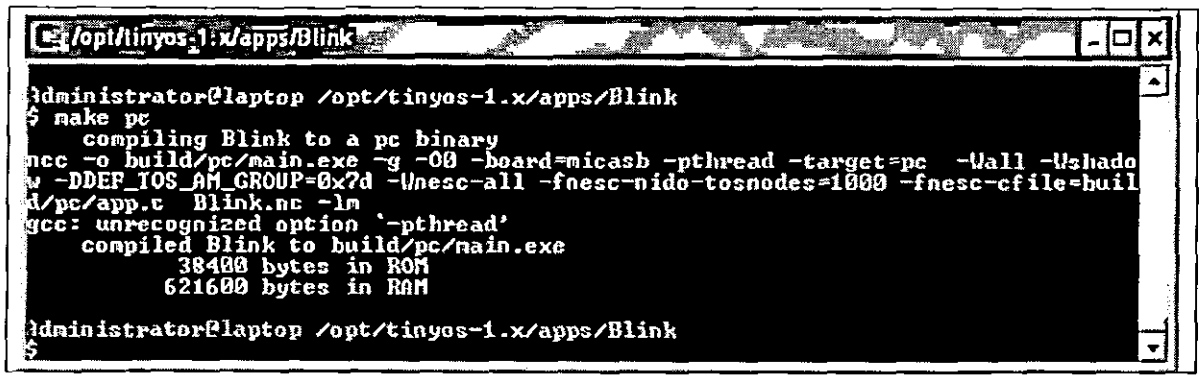


Figure 14 Compiling an Application

TOSSIM can be configured in Linux and Windows. Basically it is a Linux based application but if one wants to run it in windows then first install a Linux emulator then configure TinyOS, and then TOSSIM will be available in windows platform. Cygwin is a Linux emulator that we have used it in our work.

After the compilation of any application a new directory *build* will be created in side the application direction as shown in Fig 15. This directory contains all the compiled files in it.

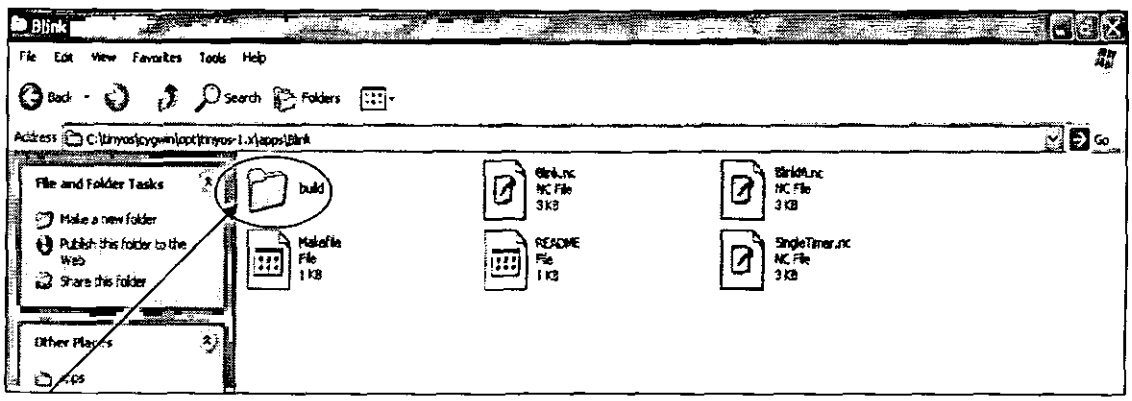


Figure 15 Directory Structure

Inside this directory there will be directories according to the platforms you have compiled for, for example for "*make pc*" there will be a directory named "*pc*" and for "*make mica*" there will be "*mica*" directory respectively. Fig 16 shows the *pc* directory created after the "*make pc*" command.

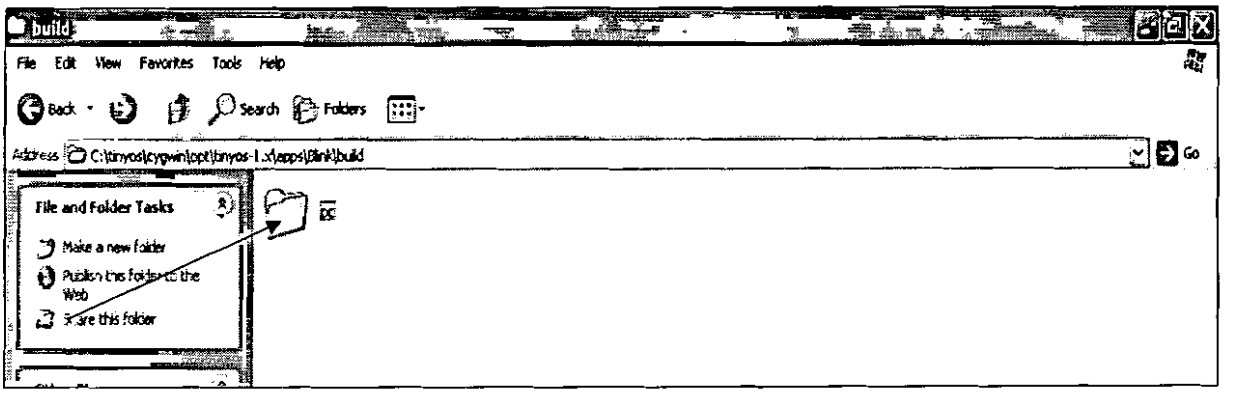


Figure 16 Directory Structure

This directory actually contains the compiled files in it. "main.exe" is the file which is an executable compiled file. Fig 17 shows the compiled files in *pc* directory.

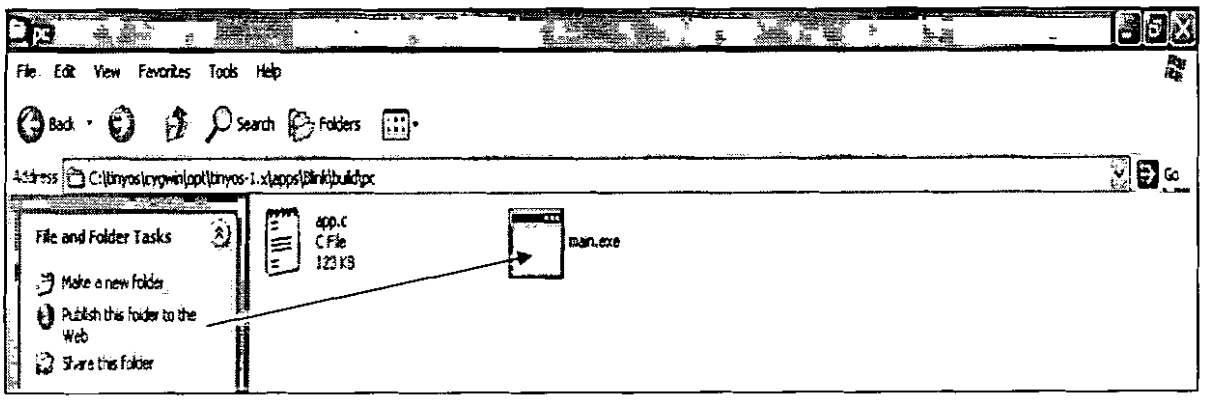


Figure 17 Main .exe File Location

So if we conclude the whole compilation process we will say that by "make" command with desired platform a directory will be created inside the application directory. In build directory all the platform dependent directories are generated. Finally inside the specified platform directory a "*main.exe*" file is created which is the actual compiled file of respective application.

5.3 How to Run an Application

Up till now we have compiled the application and we have understood the directory structure as well. Now we will see the steps how to run a pre compiled application.

To run any application first open the cygwin shell (in windows for Linux just open the terminal). Go to the application's directory which you want to run by "*cd*" command. Here give a command "build/pc/main.exe 5" to run this application for a network of 5 nodes as shown in Fig 18. This will start the simulation.

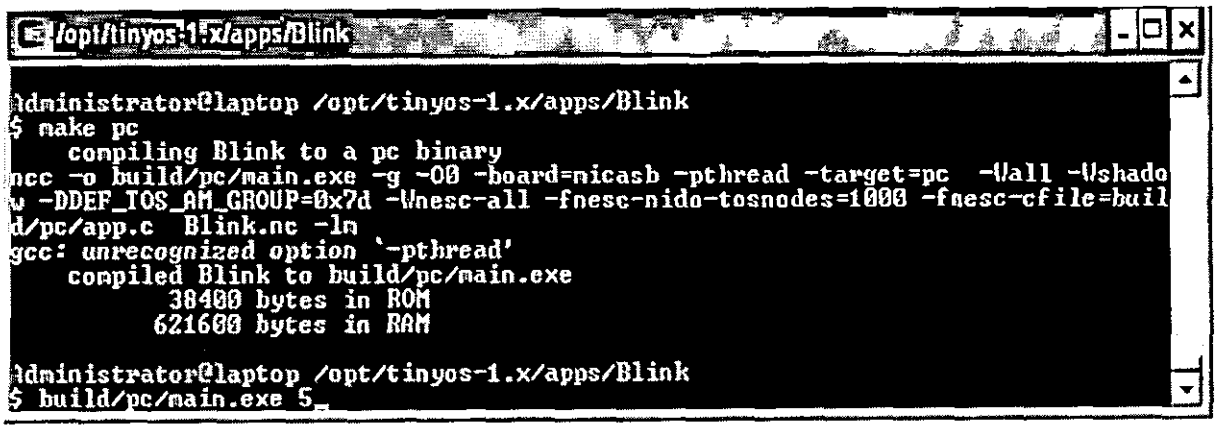


Figure 18 Running an Application

But before running the application you have to specify the dbg variables first, otherwise it will enable all the dbg variables and you will get a huge output and will not be able to understand any thing. Fig 19 shows an application that is run without specifying the DBG modes.

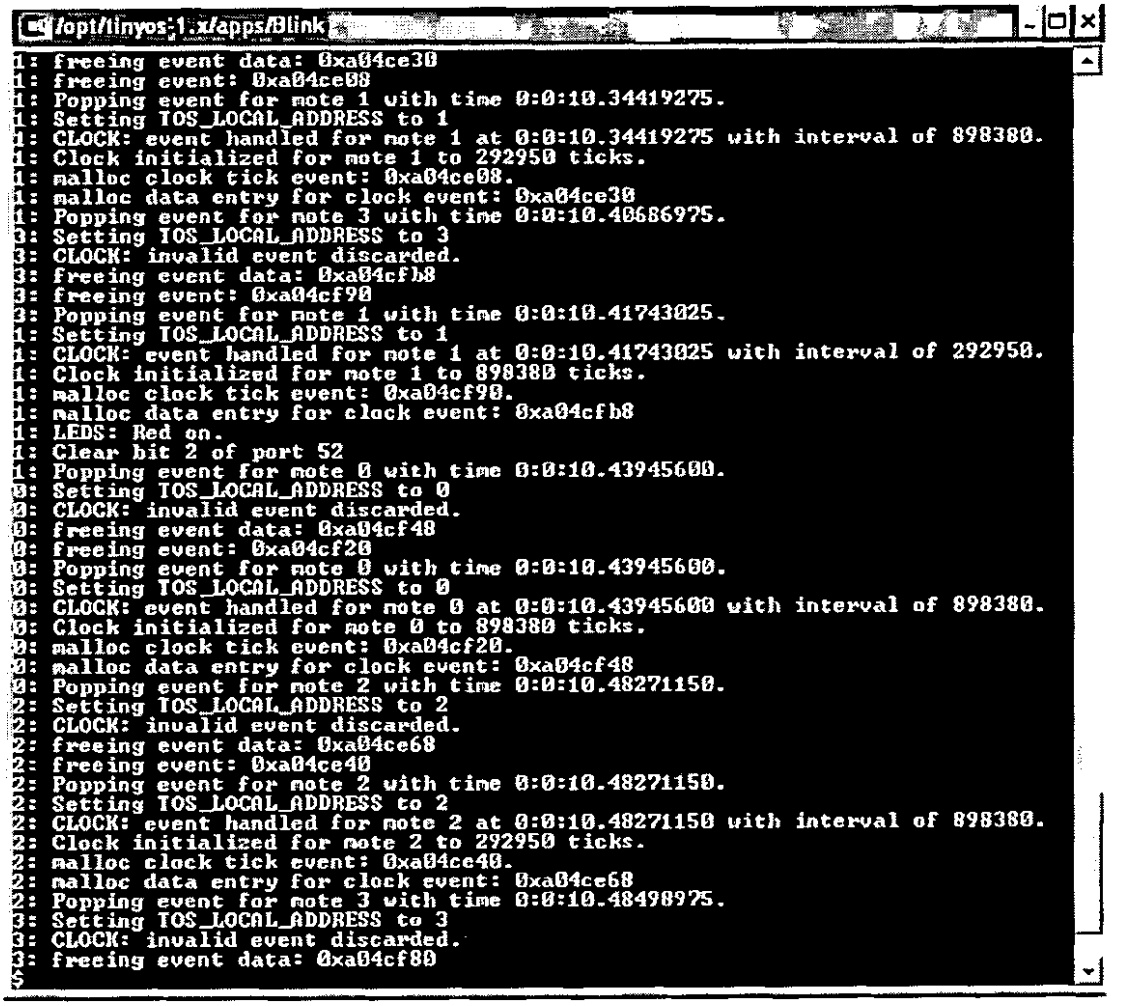


Figure 19 Snapshot of an output

But if you specify the DBG modes before running the application then it will be easy to understand the output. As mentioned in Fig 20, only DBG=led is enabled. So this will only show the status of LEDs.

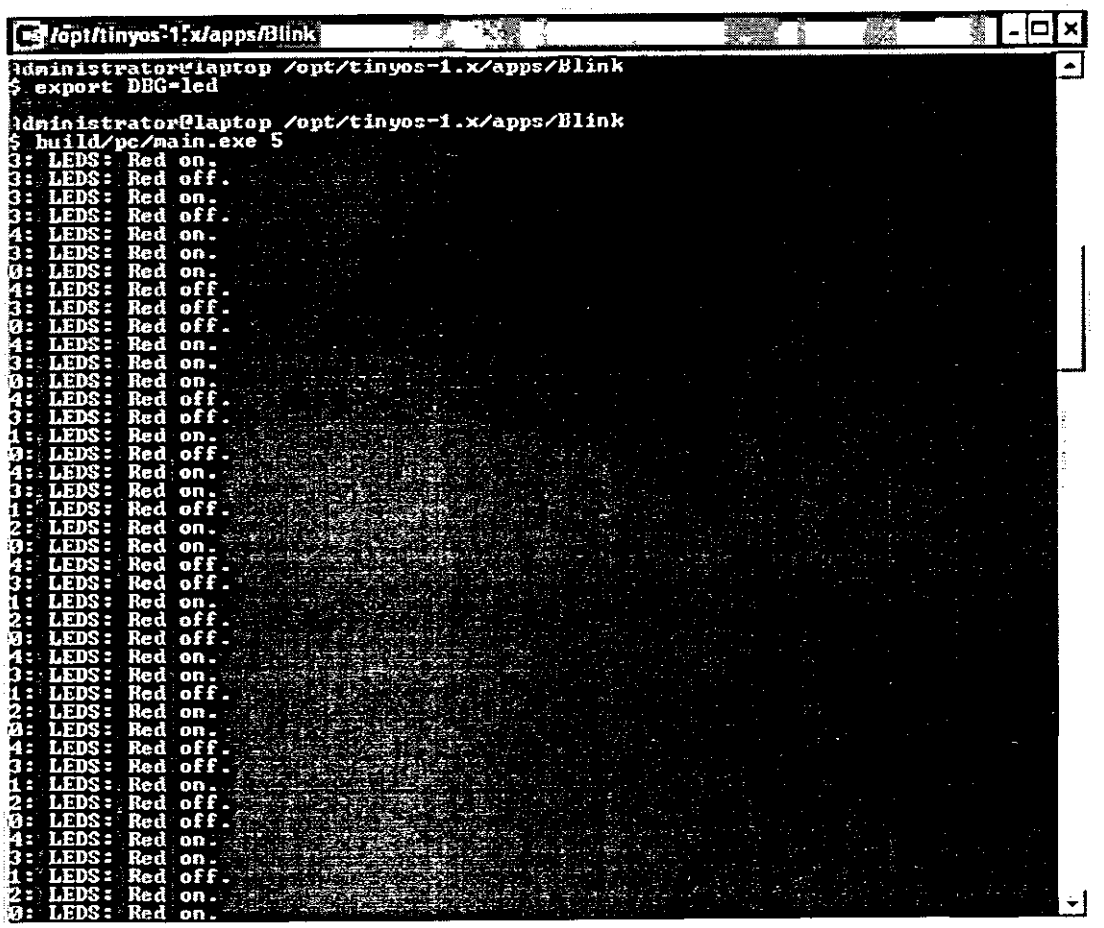


Figure 20 Snapshot of an output

The details of running options are given in Fig 21.

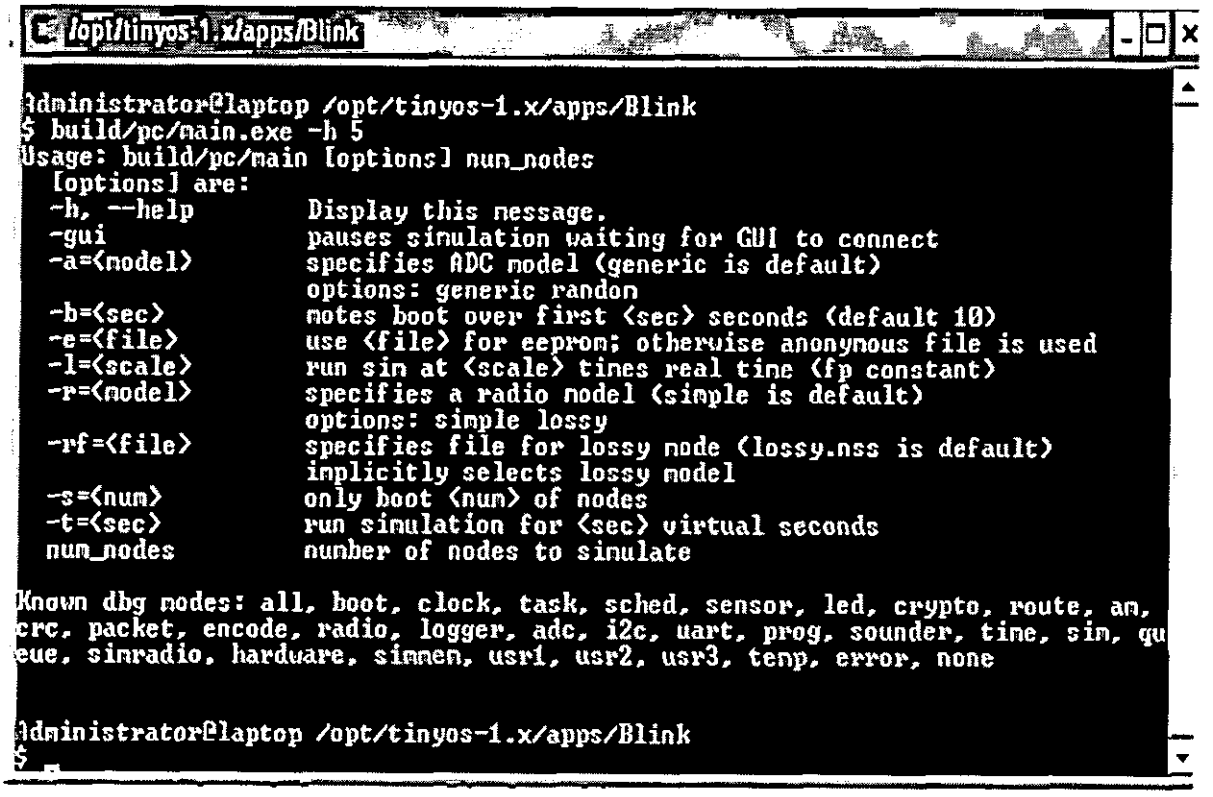


Figure 21 Snapshot of Help command

-h or –help prints the help page as shown in Fig21.
-gui option suspends the execution until the gui tool is connected. The GUI tool is TinyViz which will be discussed in coming section.
-a=<model> ADC models can be specified here. Generic and Random are two available model.
-b=<sec> here the booting time of nodes can be specified.
-ef=<file> EEPROM file input is given here, By default it uses anonymous file.
-l=<scale> here scale of time at which the simulation has to run is mentioned. TOSSIM runs the simulation at the rate according to real time given. For example, -i=2.0 means to run the simulator twice the real time.
-r=<file> to specify the radio file as an input.
-rf=<file> a topology file is given as input by using this option, TOSSIM places the nodes according to this topology file.
-s=<num> from specified network only those number of nodes will be booted which are mentioned here.
-t=<sec> to run the simulation for specific interval of time.
num_nodes to run the simulation for a specific number of nodes.

5.4 DBG Modes

Debugging environment is provided at run time in TOSSIM. DBG flags are set for each debug statement in code. At run time DBG modes should be specified otherwise all the

dbg modes will be enabled and you will get huge number of statements which will be very complex to understand. To enable specific mode the command is:

export DBG=usr1

We can enable multiple modes in same command like:

export DBG=usr1,adc,boot

The available dbg modes are mentioned in help command shown in Fig 22.

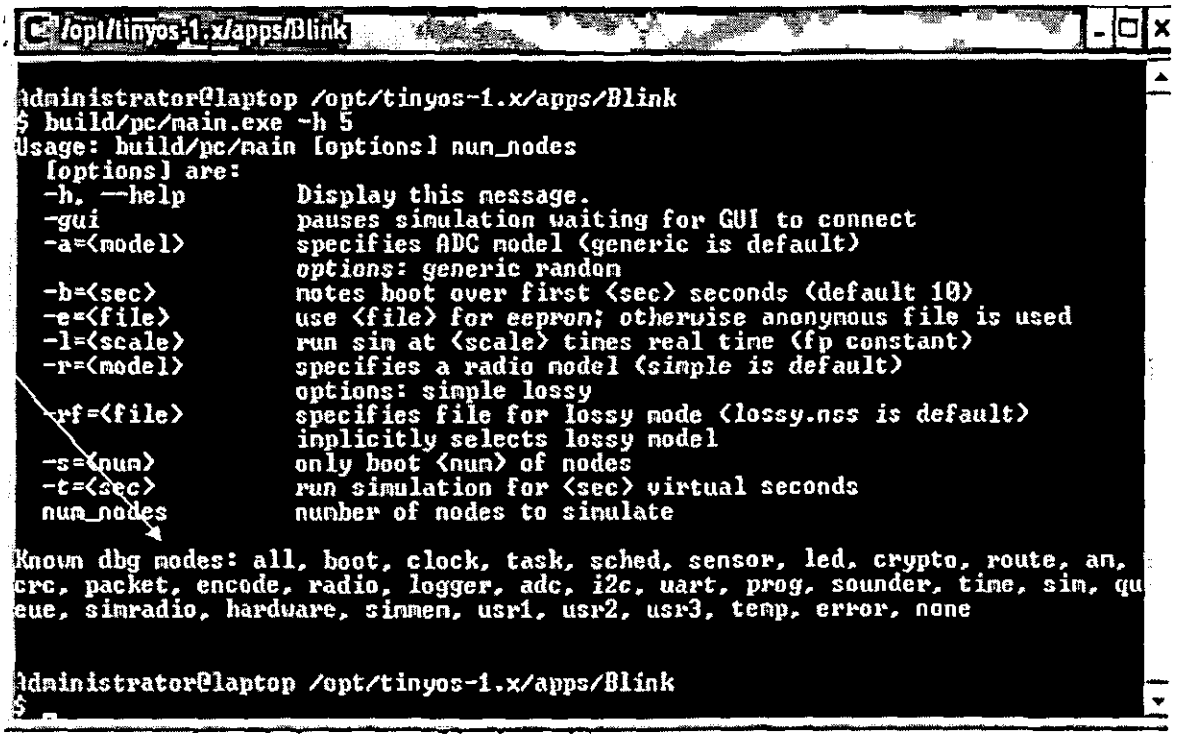


Figure 22 Snapshot of DBG Modes List

Let us see a little bit detail of each of the mode shown in Fig 22:

all: Enable all available messages

boot: Simulation boot and StdControl

clock: The hardware clock

task: Task enqueueing/dequeueing/running

sched: The TinyOS scheduler

sensor: Sensor readings

led: Mote leds

crypto: Cryptographic operations (e.g., TinySec)

route: Routing systems

am: Active messages transmission/reception

crc: CRC checks on active messages

packet: Packet-level transmission/reception

encode: Packet encoding/decoding

radio: Low-level radio operations: bits and bytes

logger: Non-volatile storage

adc: The ADC

i2c: The I2C bus

uart: The UART (serial port)

prog: Network reprogramming

sounder: The sounder on the mica sensor board

time: Timers

sim: TOSSIM internals

queue: TOSSIM event queue

simradio: TOSSIM radio models

hardware: TOSSIM hardware abstractions

simmem: TOSSIM memory allocation/deallocation (for finding leaks)

usr1: User output mode 1

usr2: User output mode 2

usr3: User output mode 3

temp: For temporary use

DBG modes are basically some flags that are set in the code and being manipulated at run time. The code debugging is done by using these dbg modes. The user can define his/her own flags using four modes which are usr1, usr1, usr3 and temp. Other than these four modes are known as system modes.

5.5 TinyViz

GUI tool for TOSSIM is TinyViz. As mentioned above, the –gui option at run time enables this tool. When we give command using this option the simulator suspends itself

until TinyViz will be connected. To connect TinyViz you have to open a new Cygwin shell and give this command:

java net.tinyos.sim.TinyViz

This will run a java based tool called TinyViz. Fig 23 shows the two cygwin shells used to run TinyViz.

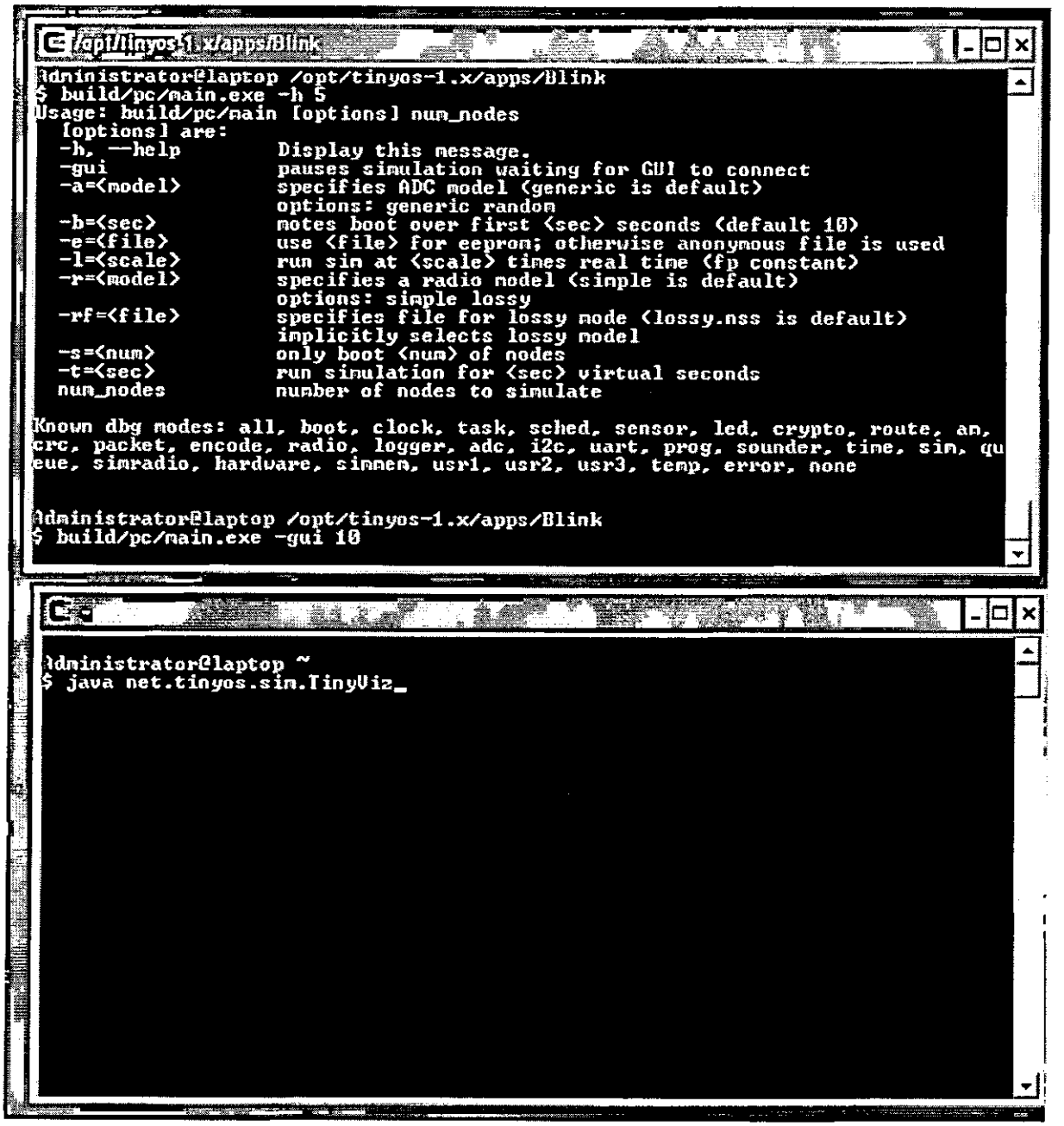


Figure 23 Running TinyViz

After this command TinyViz tool will run. Fig 24 shows the front end of TinyViz tool.

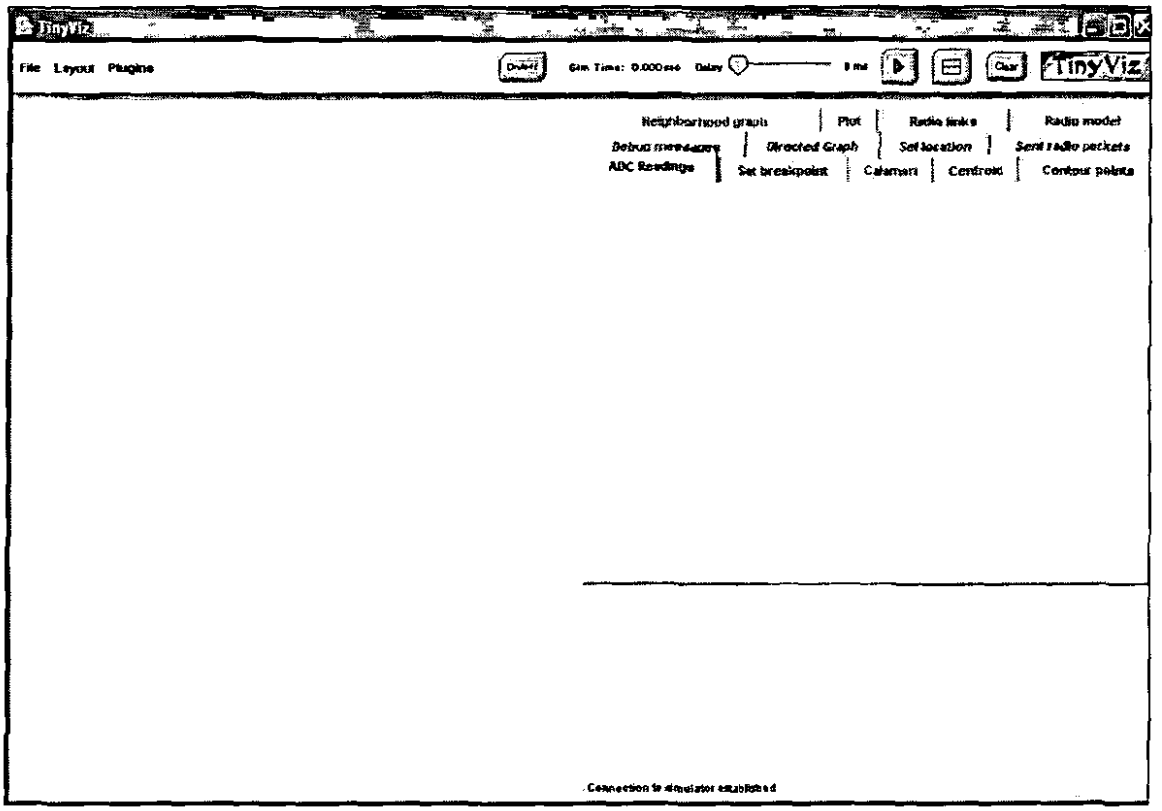


Figure 24 Snapshot of TinyViz GUI

In TinyViz different Plugins are available. You can simply enable them to visualize the specified effect. For example, Debug Plugin will enable the debug messages in this tool at left side of the screen. There are some other options like on/off button will quit the simulation, Sim Time shows the simulation time, Delay is for controlling the speed of the simulation, Play button starts the simulation and stops it. Fig 25 shows the list of Plugins available in TinyViz.

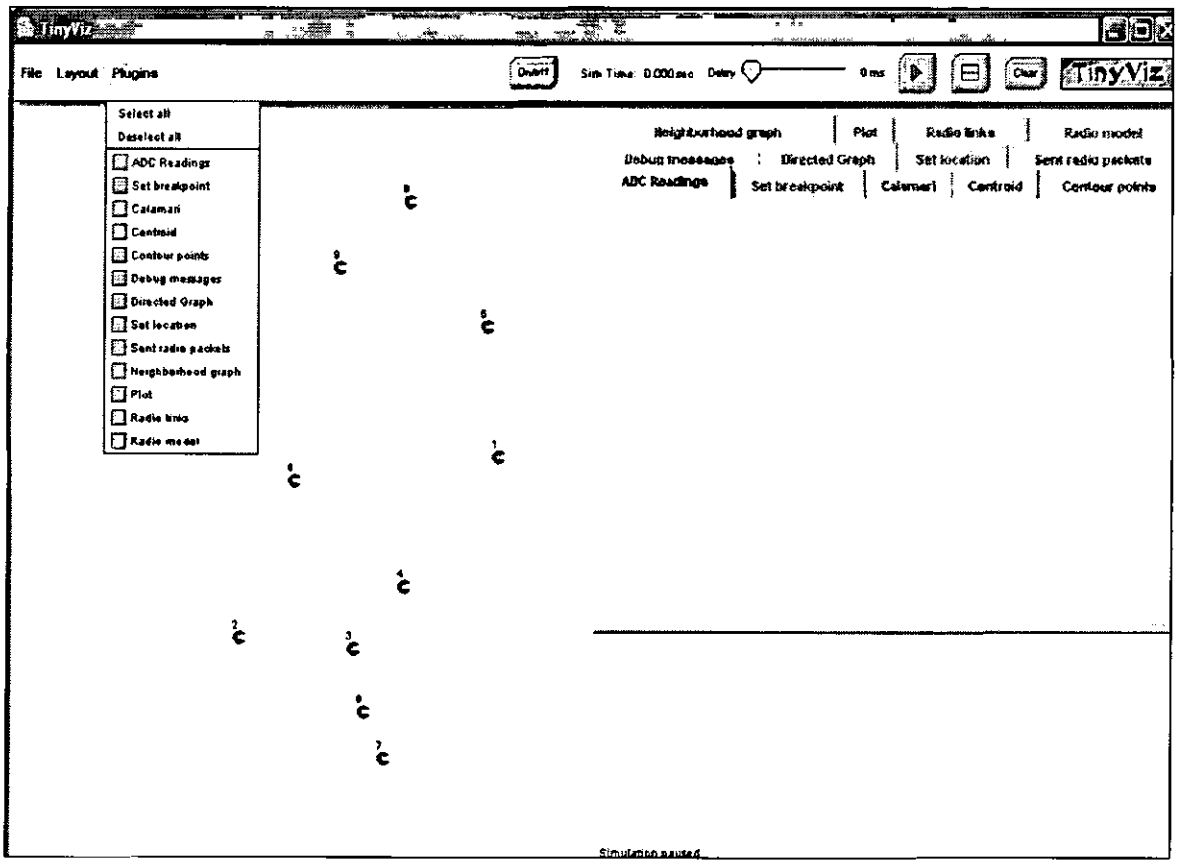


Figure 25 Snapshot of TinyViz Plugins List

5.6 NesC Language

For embedded systems a new language nesC is adopted. NesC is used for component based application development. TinyOS system and applications are written in nesC. NesC is a programming language having C like syntax. NesC stands for "Network Embedded Systems C".

5.6.1 Basic Concepts Behind nesC

- In nesC applications are developed out of components, having bidirectional interfaces. NesC is a programming tool for the development of application programs in TinyOS. For the installation of nesC compiler we have to get its rpm file and install it using cygwin shell and then configure it by setting the class path.
- In nesC there are components with interfaces as ports for the communication of components with others. So a component can only be used/accessed through its interfaces. The component may provide or use interface that is why the interfaces are called bidirectional.

- Interfaces are basically the functionality of a component. One component is accessed by the other component through its interfaces. Interfaces can be accessed by key word *uses* or *provides*. So a component may be an interface provider or an interface user.
- Once an interface is used it must be implemented by the user component. This shows that interfaces make complex interaction between the components, for example after using the interface Send the Send.send instance can't be called until and unless Send.sendDone instance is implemented.

NesC compiler is installed in TinyOS environment but you have to set the class paths for its configuration. For any type of application we have to create two files for nesC programming. One for wiring the interfaces and other is for actual implementation. NesC file extension is *.nc*. The file containing the wiring of the components is called configuration file and the other file with code is called Module file.

5.6.1.1 Configuration File

The configuration file tells us how different components are bind together to perform the collective activity for specific application. The code below is taken from an application written in nesC. Here we will see that how components are bound together through their interfaces. The bounding is denoted by an arrow -> character. The component at left side of the arrow uses the interface and the component at right side of the arrow provides the interface.

```
Main.StdControl -> BlinkM.StdControl;
```

Here Main component is using the interface StdControl which is provided by the component BlinkM. The configuration file is enclosed in a block named same as the file name.

```
configuration Blink {
}
```

5.6.1.2 Module File

As mentioned above actual implementation of an application is written in module file. Like configuration file, in module file the whole code is enclosed in a block named same as file name. That is why a capital M is written with the file name for identification.

```
module BlinkM{
}
```

Both files should be saved by the name as same as the configuration and module block names. In module file interfaces used or provided are mentioned by the key word uses and provides. As mentioned above Main component is using the StdControl interfaces provided by component BlinkM. In module file this will be shown as:

```
module BlinkM {
  provides {
interface StdControl;
  } }
```

This shows that the component BlinkM is providing the interface StdConrol.
Now we see how dbg flags are set in nesC code. For the purpose of debugging we can set the DBG flags in code and at run time we can easily debug the application by enabling the specific DBG mode.

```
dbg(DBG_USR1, "Timer fired:  \n");
```

Here the DBG flag USR1 is set and given a value that contains a string "Timer fired". Now at runtime when we will enable this mode then this output will be shown at screen. We can get different variable values and can easily manipulate them for complex calculations using this facility.

TinyOS application is actually configured in layered structure because we know that layered structure is easy to handle for compiler. When these layers collaborate with each other a complete application is formed. Fig 26 shows a complete picture of layered application of TinyOS.

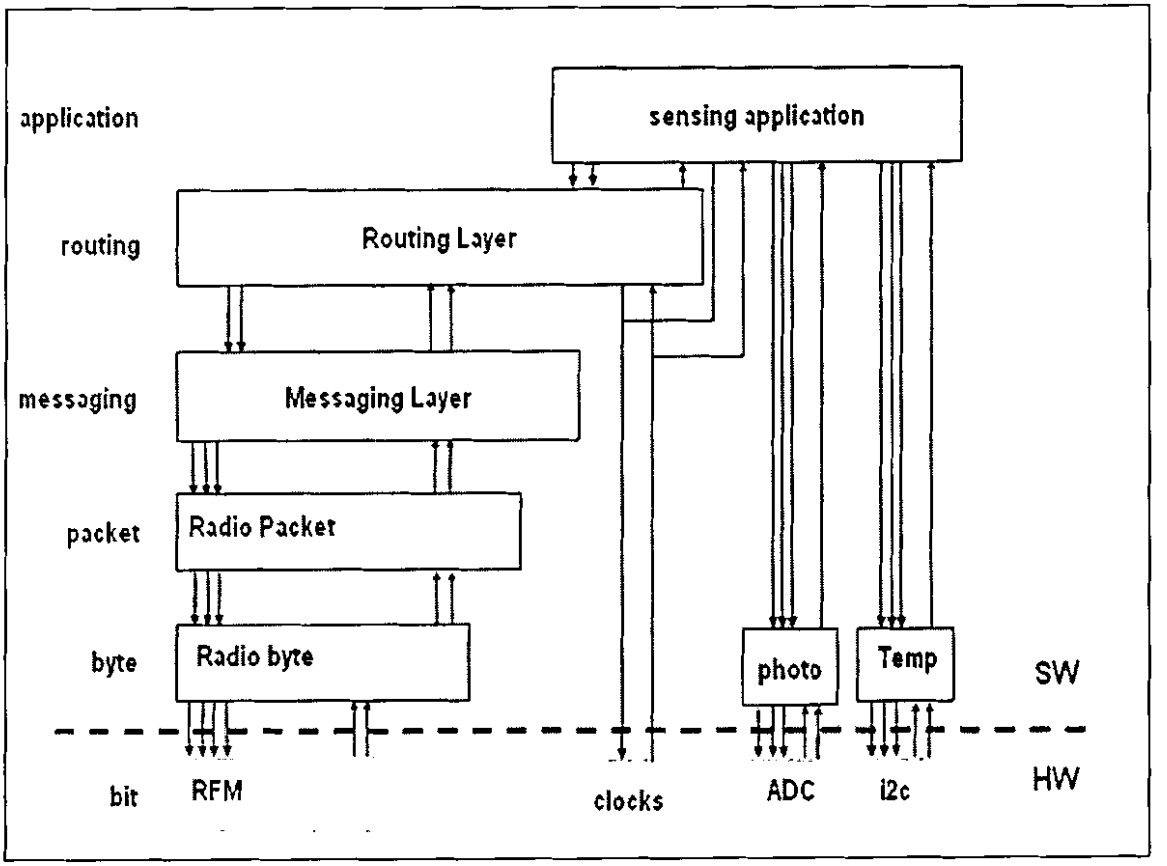


Figure 26 A complete TinyOS application [41]

5.7 Simulation Setup

This section describes the simulation setup used for the implementation of proposed algorithm. The simulation topology is shown in Fig 27. Thirty nodes are used with node 0 as CH. The algorithm runs for intra cluster having 29 cluster member nodes. The algorithm first calculates the distance of each node from CH, if the distance is less than 25 feet the mode of routing is selected as Direct Routing. On the other hand nodes at distance of more than 25 feet will adopt Multihop Routing. So nodes 1,2,6,7,8,12 and 13 will adopt direct routing and others will adopt multihop routing for data packet routing.

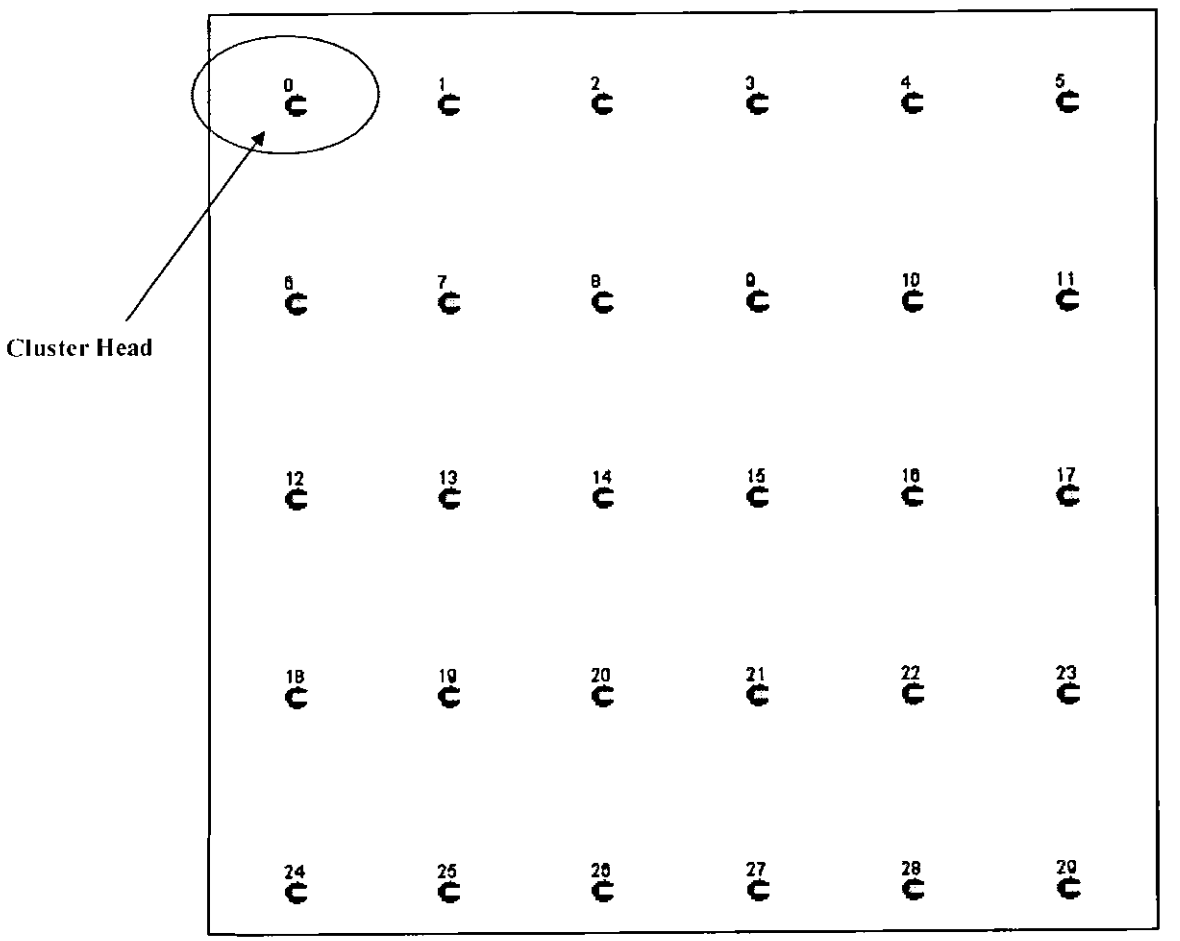


Figure 27 Topology considered for simulation

The nodes within distance of 25 feet are considered as in close region of the topology. Fig 28 shows the nodes in close region and nodes out of close region. It is clear from the figure that nodes 1,2,6,7,8,12 and 13 are within the close region and all others nodes are out of the close region.

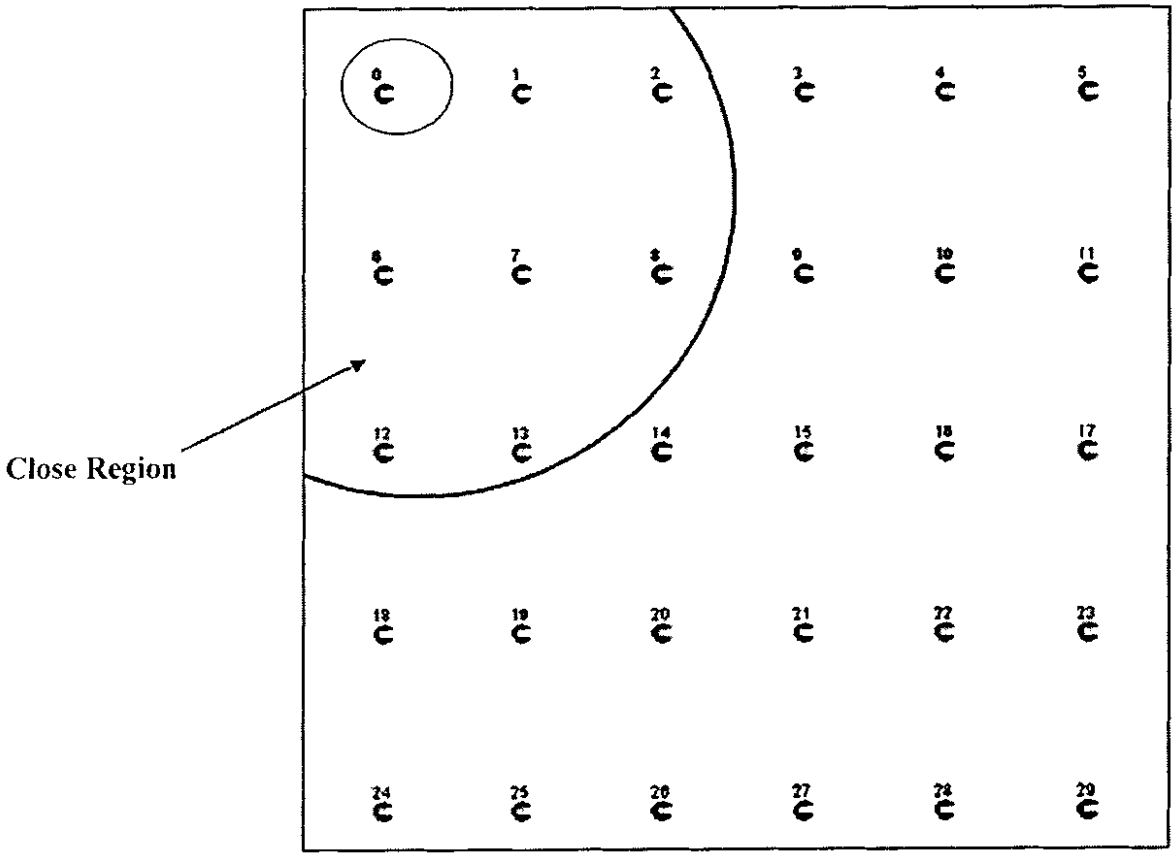


Figure 28 Topology considered for simulation

5.8 Simulation Parameters

We have performed three different experiments by simulating different scenarios. The topology taken for the simulation is grid and in a grid of 5x6 the network is established. Node to node distance is 10 feet. A java based application "Lossy Builder" is used to set the distance parameter for the simulation. Different experiments have been done using different parameters and number of results have been achieved to analyze the performance of algorithm.

5.8.1 Experiment No 1

To check the energy consumption of the algorithm the energy level of each node is considered 300 mJ. This simulation is run for 30 nodes, it is mentioned in future works that same algorithm can be checked on large network size. The nodes having less than 25

feet distance from the CH will adopt direct routing. Different distance factors have been checked for same topology but optimum results have been achieved for 25. The simulation is run for approx 15 minutes. Results have been shown in next chapter while considering the parameters as discussed. Table 1 shows the list of parameters considered for the simulation.

Table 1 Simulation Parameters for Experiment no 1

Parameters	Value
Topology	Grid
Simulation Time	15 min
Maximum Energy level	300 mJ
Distance Unit	Feet
MAC Scheduling	S-MAC
Network Size (Nodes)	30
Area of Close Region	25 Feet
Cluster Head	Node 0
Broadcast ID	65535
Geographical Area	2000 Sq Feet

5.8.2 Experiment No 2

The simulation time for this experiment is taken 30 minutes and the initial energy level of each node is taken 600 mJ. All the other parameters are kept constant. Table 2 shows the simulation parameters considered for this experiment.

Table 2 Simulation Parameters for Experiment no 2

Parameters	Value
Topology	Grid
Simulation Time	30 min
Maximum Energy level	600 mJ
Distance Unit	Feet
MAC Scheduling	S-MAC
Network Size (Nodes)	30
Area of Close Region	25 Feet
Cluster Head	Node 0
Broadcast ID	65535
Geographical Area	2000 Sq Feet

5.8.3 Experiment No 3

Table 3 is showing list of parameters taken for this experiment. For this experiment, initial energy level of each node is taken 900 mJ and simulation is run for 45minutes. All the other parameters are kept constant.

Table 3 Simulation Parameters for Experiment no 2

Parameters	Value
Topology	Grid
Simulation Time	45 min
Maximum Energy level	900 mJ
Distance Unit	Feet
MAC Scheduling	S-MAC
Network Size (Nodes)	30
Area of Close Region	25 Feet
Cluster Head	Node 0
Broadcast ID	65535
Geographical Area	2000 Sq Feet

Fig 29 shows a screen shot of Adaptive Routing simulation taken from a live simulation running in TOSSIM simulator. Fig 30 shows that the nodes in close region that are 1 and 12 are sending the packets directly to node 0 but on the other hand nodes 18 and 20 are sending the data packets toward CH through node 11 by adopting multihop routing. This is clear that adaptive routing technique has shifted the network load far from closer nodes thus enhanced the network lifetime.

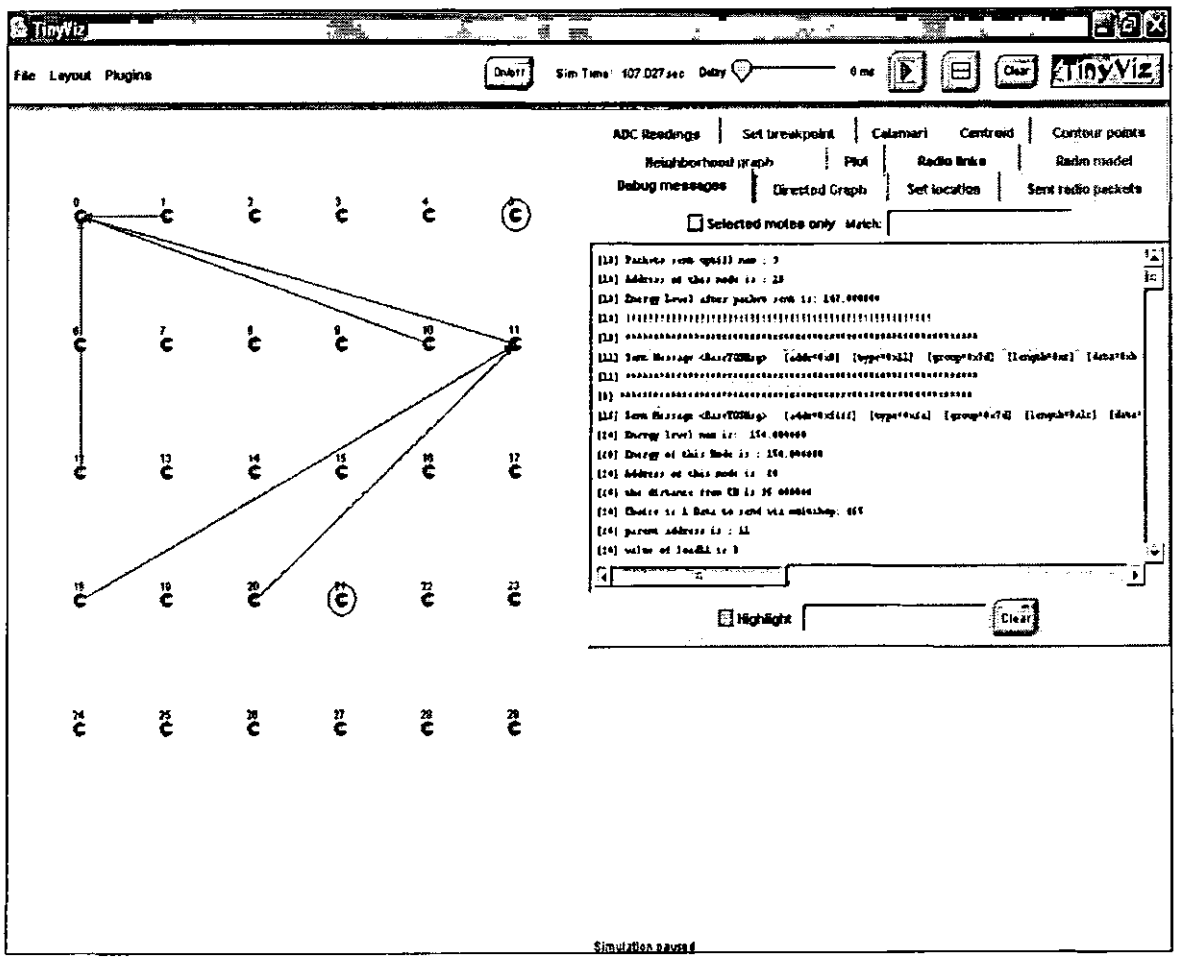


Figure 29 Screen shot of Adaptive Routing

Chapter 6

Results

6. Results

In this chapter we have described the simulation topology, simulation assumptions and simulation results which show the effectiveness of proposed protocol. The results also show that stated goals have been achieved which were described in previous chapter.

For simulation we have used TOSSIM with TinyOS. The results have been compiled and compared with existing techniques but proposed technique is more energy efficient than the others. The simulation of proposed Adaptive Routing is run for 30 nodes and is compared with well known routing protocol, multihop router that performs multihop routing. Numbers of simulations have been carried out to get the optimal solutions. Then both techniques are compared to show the performance of proposed scheme using different parameters.

6.1 Performance Metrics

The performance metrics considered are:

1. Packets sent in the network
2. Energy consumed by the network
3. Energy remaining of network (at time t)
4. Network life time
5. Total number of packets sent to Cluster Head

6.2 Simulation Results

In Fig 30 both the techniques are compared and a combined graph has been plotted that is showing that Adaptive technique is capable of sending more data packets in the network as compared with Multihop routing. In our algorithm the nodes at a distance of 25 feet are considered as close nodes and the mode of routing for these nodes is Direct routing. Being closed to the CH less power will be required to send the data, so less energy will be consumed. No load of others nodes have to bear these close nodes. Due to this fact their packet sending capability will be increased. On the other hand nodes at large distance have to put in more energy to transmit their data towards node 0 which is CH as the topology of the simulation is shown in Fig 27.

The graph plotted in Fig 30 shows that the packet delivery rate is very high in close nodes in Adaptive Routing algorithm. The close nodes are 1,2,6,7,6,12 and 13. It is clear from the graph that the packet delivery ratio is very high in these nodes because they do not have any load of extra packet to send with them. On the other hand in Multihop routing the nodes close to the CH have to send their own data with additional load of other nodes. By having extra delivery of data through them causes more energy consumption, resulting the less number of packet delivery in the network. So we can say that the through put of proposed algorithm is more than the traditional technique.

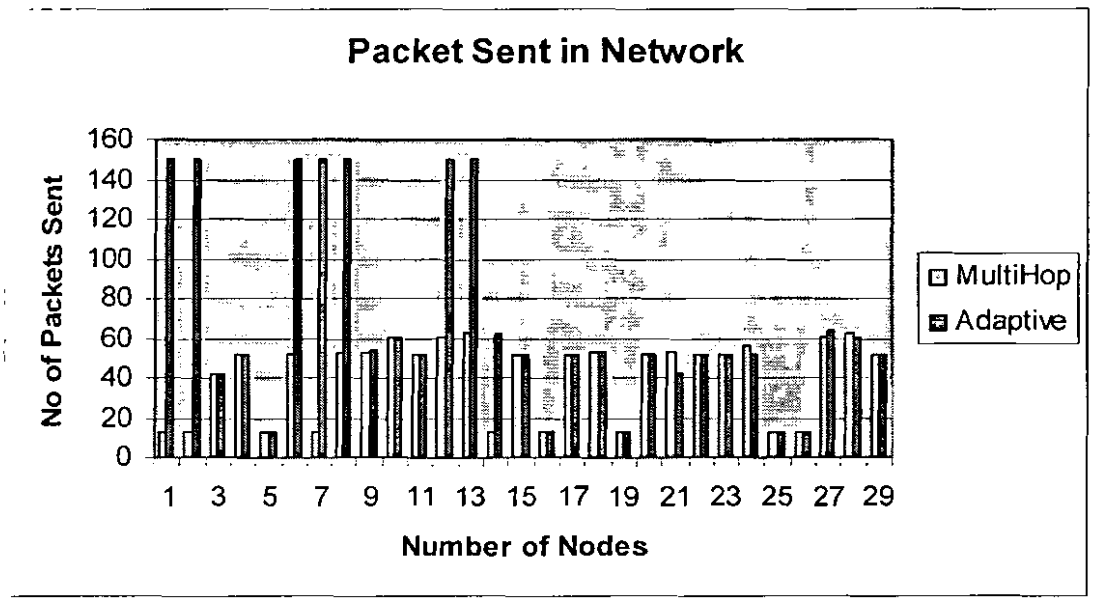


Figure 30 Packet Sent in the Network

Different experiments have been done while considering different energy levels to get the overall performance of the algorithms. Fig 31 shows a comparison of average number of packet sent in the network. It is clear form the graph that while increasing the initial energy level, the proposed algorithm is capable of sending more data packets towards the CH if compared with multihop routing algorithm.

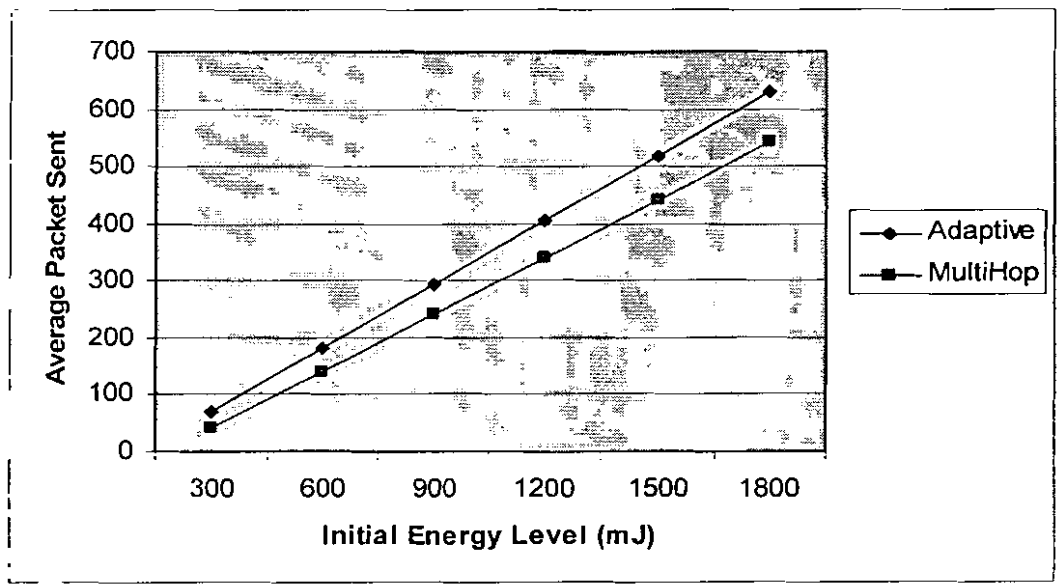


Figure 31 Average Packet Sent in the Network

Fig 32 shows the comparison of remaining energy levels of nodes for one simulation. It is clear from the graph the proposed technique has more remaining energy level than the other one. If we put a deep look in the graph we will see that the nodes which are closed to the CH (while adopting direct routing) as mentioned above are having more energy levels than the nodes adopting multihop routing. This is a normal behavior of the nodes because the nodes are putting less energy for data transmission will have more energy as remaining power source. This check is performed after 4 minutes of simulation run and at that point energy levels are checked and this is clear from the graph that overall energy consumption is less in adaptive Routing. So this leads to an energy efficient routing for WSN.

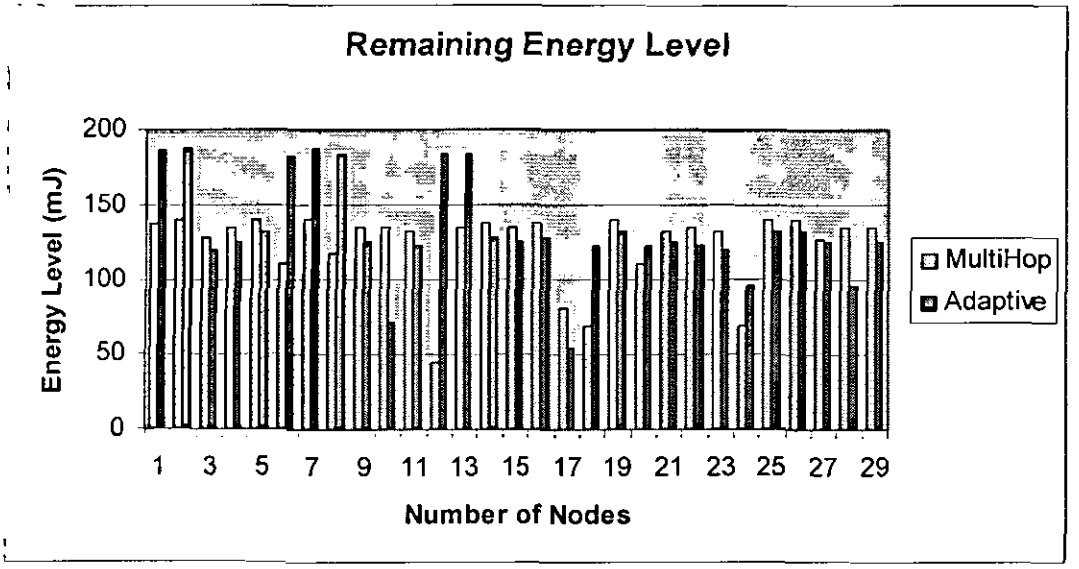


Figure 32 Remaining Energy of Network

Fig 33 shows the energy consumed by the nodes at time t. This is clear that Adaptive technique is more energy efficient and the energy consumption is less in this technique as compared with multihop routing. After some time of simulation run, this check is performed to get the behavior of the network. Same behavior is also shown by the nodes in this graph. The nodes adopting direct routing will definitely consume less energy for the transmission of their respective data. On the other hand nodes far away from the CH will consume more energy for data transmission. Here *energy efficient routing* factor is also achieved.

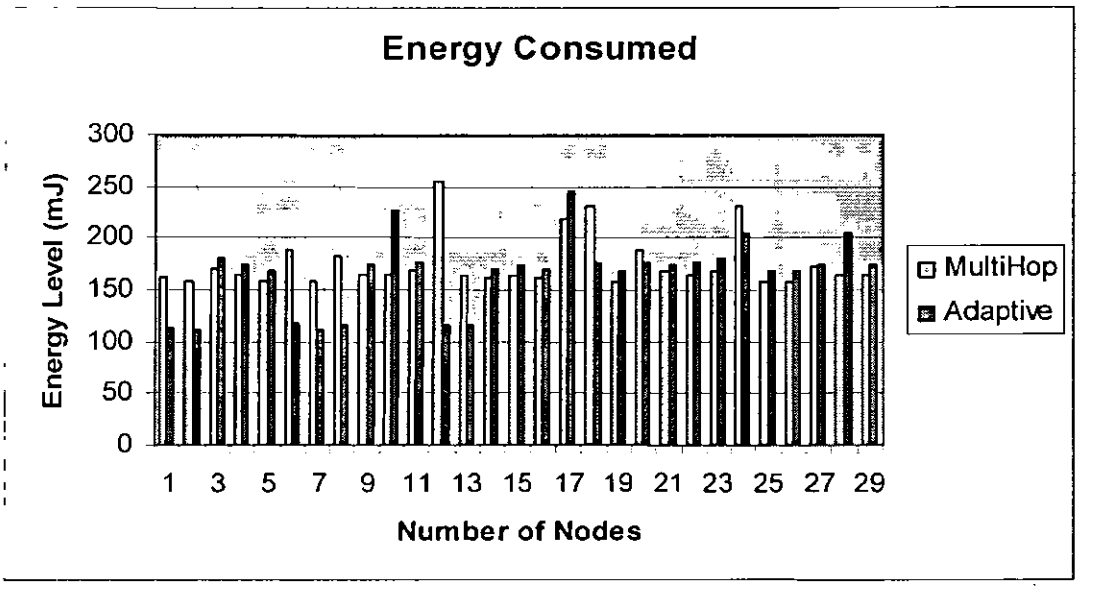


Figure 33 Energy Consumed at Time t

In Fig 34 the comparison of total packets sent in the network is shown and the graph is showing that Adaptive routing is capable of sending more packets towards the CH. If we compare both the techniques we will see that Adaptive routing is sending 68% more packets towards CH than multihop routing. It is very obvious that such result has been achieved because we have seen above, if the technique is sending more packets to the CH, the remaining energy level is high at any time while routing and the energy consume by the nodes is also less. Then it is obvious that total number of packets sent in the network will be more. This is another proof of energy efficient routing technique that by having same level of power source the Adaptive Routing technique is capable of sending more packets toward the respective CH.

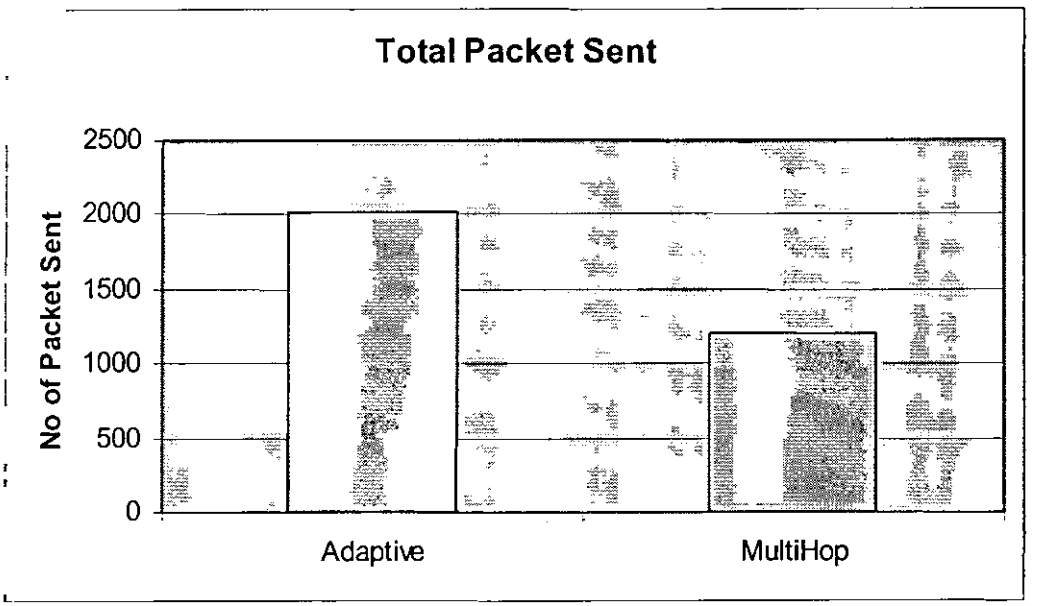


Figure 34 Total Packet Sent in the Network

One of the most important factors is network lifetime for WSN, which has been increased by proposed technique. Fig 35 shows the comparison of network lifetime in both techniques. The graph is showing that while using multihop routing the node death occurred earlier than in adaptive routing algorithm thus increased the network lifetime. Network lifetime of any network is calculated with the death of first node in the network. In proposed technique the death of first node occurred after the death of first node in multihop routing technique. This is one of the major achievements of our proposed technique.

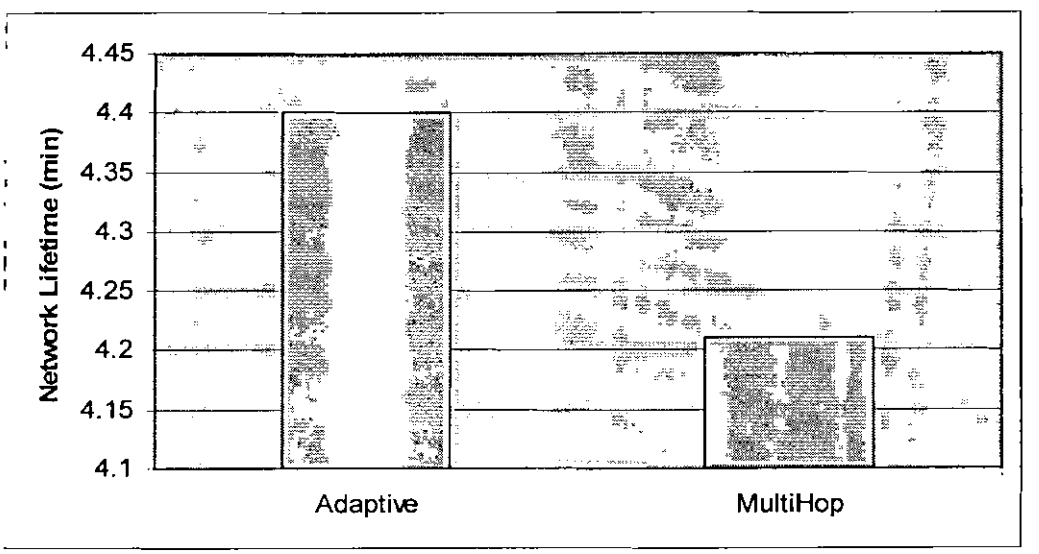


Figure 35 Network Lifetime

Number of experiments have been taken by considering different simulation parameters. Fig 36 shows comparison of network lifetimes, by considering different initial energy levels of nodes. It is clear from the graph that as the energy level is increased the difference in network lifetime for Adaptive routing and Multihop routing also increases, and Adaptive routing is achieving higher lifetime than the other one.

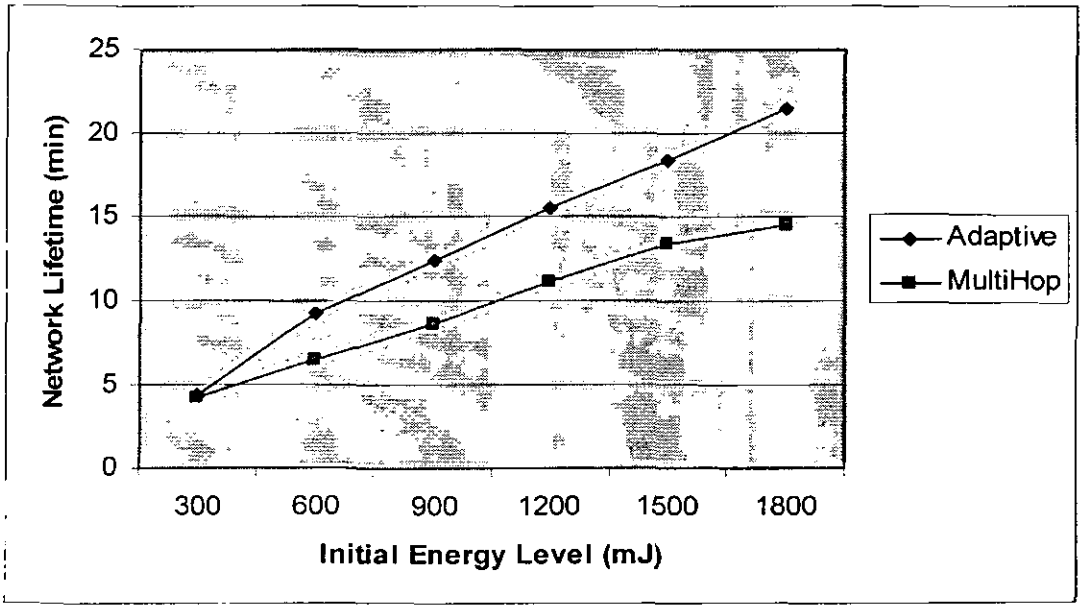


Figure 36 Network Lifetimes for Different Energy Levels

The results of our proposed algorithm are showing that set goals have been achieved in this research work after going through number of simulations.

Chapter 7

Conclusions and Future Work

7.1 Conclusions

In this research work, we have proposed a new technique for intra cluster routing which is more energy efficient than a well known routing protocol, multihop router that performs multihop routing. We have proved our idea by simulating a network of 30 nodes in TOSSIM. While justifying our idea through results of our simulation we have considered the parameters that include: number of packets sent in the network, energy consumed by the network, remaining energy level of nodes at specific time and network lifetime of the network. By using proposed technique we have increased the network lifetime and number of packet sent in the network. Finally it has been shown that overall energy consumption for network is reduced. Simulation showed that Adaptive Routing increased 68% in number of packets sent in the network. 4% increase has been achieved in lifetime of the network. If the simulation time is increased these factors will give more commendable results for the proposed algorithm.

7.2 Future Work

We have developed a new routing protocol that is energy efficient than the traditional protocol but there is still room to work in this area to make WSN more reliable. This work can be extended in different dimensions that include:

7.2.1. Real Motes

Currently our idea is implemented and tested in a simulation using TOSSIM simulator. This work can be tested on real motes and then while considering real environment more mature results can be achieved.

7.2.2. Inter Cluster routing

At present the proposed idea is implemented and tested for intra cluster routing but this work can be extended to inter cluster routing. This will involve the issues of inter cluster communication as well.

7.2.3 Simulation time

Currently the simulation time is very short but still positive results have been achieved. This can be run for long time and the results after increased in simulation time will be more effective. Because if Adaptive routing technique is sending more packets in less time then more packets will be sent while running the simulation for a long time.

7.1.4 Network Size

The proposed routing algorithm can be tested in a large network size. As in our work we have considered the cluster size of 30 nodes, still this is a large cluster size but this can be checked on a cluster size of more than 50 nodes. Because in real networks dense deployment of the nodes is highly required.

References

1. Yuan Yuan Li; Parker, L.E, "Intruder detection using a wireless sensor network with an intelligent mobile robot response", in Southeastcon, 2008. IEEE, Volume 3-6, April 2008 Page(s):37 - 42.

2. Liyang Yu; Neng Wang; Xiaoqiao Meng, " *Real-time forest fire detection with wireless sensor networks*", in International Conference Wireless Communications, Networking and Mobile Computing, 2005. Proceedings. 2005, Volume 2, Sept. 2005 Page(s): 1214 – 1217.

3. Wietrzyk, B.; Radenkovic, M.; Kostadinov, I, "*Practical MANETs for Pervasive Cattle Monitoring*", in Conference on Seventh International Networking, 2008. ICN 2008, Volume, 13-18 April 2008 Page(s):14 – 23.

4. Tatiana Bokareva, Wen Hu, Salil Kanhere, Branko Ristic, Neil Gordon, Travis Bessell, Mark Rutten and Sanjay Jha, "*Wireless Sensor Networks for Battlefield Surveillance*", in Proceedings of The Land Warfare Conference 2006 (LWC 2006), October, Brisbane, Queensland, Australia.

5. Yeon-sup Lim; Sangsoon Lim; Jaehyuk Choi; Seongho Cho; Chong-kwon Kim; Yong-Woo Lee, "*A Fire Detection and Rescue Support Framework with Wireless Sensor Networks*", in International Conference on Convergence Information Technology, 2007. Volume , 21-23 Nov. 2007 Page(s):135 – 138.

6. Goldsmith D, Liarokapis F, Malone G and Kemp J, "*Augmented Reality Environmental Monitoring Using Wireless Sensor Networks*", in 12th International Conference on Information Visualisation, 2008..

7. Fredrik Osterlind, Erik Pramsten, Daniel Roberthson, Joakim Eriksson, Niclas Finne and Thiemo Voigt, "*Integrating Building Automation Systems and Wireless Sensor Networks*", 2007 IEEE.

8. Wall, J.; Platt, G.; James, G.; Valencia, P, "*Wireless Sensor Networks as Agents for Intelligent Control of Distributed Energy Resources*" in 2nd International Symposium on Wireless Pervasive Computing, 2007. ISWPC, Volume, 5-7 Feb. 2007.

9. Energy Aware Routing in Wireless Sensor Networks By Ashish Babbar2008.

10. Dr Abid Ali Minhas *"Power Aware Routing Protocols for Wireless ad hoc Sensor Networks"* Phd thesis Graz University of Technology, Graz, Austria March 2007.

11. http://www.ewh.ieee.org/r2/baltimore/Chapter/Comm/WSN-IEEE-Nov2005-v2.ppt.

12. http://Sensor node - Wikipedia, the free encyclopedia.htm

13. Stefan Dulman and Paul Havinga, "*Operating system fundamentals for the eyes distributed sensor networks*", University of Twente, Department of Computer sciemce Enschede, the Netherlands, [http://www.eyes.eu.org/publications/], October 2002.

14. Adam Dunkels, Bjorn Gronvall, and Thiemo Voigt, "*Contiki - a lightweight and flexible operating system for tiny networked sensors*", in 29th Annual IEEE International Conference on Local Computer Networks, 16.

15. http://www.sics.se/contiki/.

16. http://www.ercim.org/.

17. http://www.tinyos.net.

18. Chih-Chieh Han, Ram Kumar, Roy Shea, Eddie Kohler, and Mani Srivastava," *A dynamic operating system for sensor nodes*" in MobiSys '05: Proceedings of the 3rd international conference on Mobile systems, applications, and services, pages 163–176, New York, NY, USA, 2005. ACM Press.

19. http://nesl.ee.ucla.edu/projects/sos-1.x/publications/.

20. Lin Gu," *Virtualizing operating system for wireless sensor networks*", University of Virginia Charlottesville, VA, USA . Pages: 176 Year of Publication: 2006.

21. Jamil Ibriq and Imad Mahgoub, *"Cluser-Based Rouitng in Wireless Sensor Networks: Issues and Challenges "* Department of Computer Science and Engineering, Florida Atlantic University.

22. Wendi Rabiner Heinzelman, Anantha Chandrakasan, and Hari Balakrishnan, "*Energy-Efficient Communication Protocol forWireless Microsensor Networks*", in Proceedings of the 33rd Hawaii International Conference on System Sciences 2000.

23. Seema Bandyopadhyay and Edward J. Coyle, "*An Energy Efficient Hierarchical Clustering Algorithm for Wireless Sensor Networks*", in IEEE INFOCOM 2003.

24. Ming Ma and Yuanyuan Yang, "*Clustering and Load Balancing in Hybrid Sensor Networks with Mobile Cluster Heads*", Department of Electrical and Computer Engineering State University of New York, Stony Brook, NY 11794, USA. ACM Third International Conference in Quality Of Service In Heterogeneous Wired/Wireless Networks 2006.

USA. ACM Third International Conference in Quality Of Service In Heterogeneous Wired/Wireless Networks 2006.

25. M. Goyeneche, J. Villadangos, J.J. Astrain, M. Prieto, A. C'ordoba, "*A Distributed Data Gathering Algorithm for Wireless Sensor Networks with Uniform Architecture*", in PE-WASUN'06, October 6, 2006, Torremolinos, Malaga, Spain.

26. Nauman Israr and Irfan Awan, "*Multihop clustering algorithm for load balancing in wireless sensor networks*" University of Bradford, UK.

27. Anh Tuan Hoang and Mehul Motani, "*Collaborative Broadcasting and Compression in Cluster-Based Wireless Sensor Networks*", in ACM Transactions on Sensor Networks, Vol. 3, No. 3, Article 17, Publication date: August 2007.

28. Li Li, Dong Shu-song, Wen Xiang-ming, "*An energy efficient clustering routing algorithm for wireless sensor networks*", in the journal of china universities of posts and telecommunications Volume 13, Issue 3, September 2006.

29. Tao Wu and Subir Biswas, "*Reducing Inter-cluster TDMA Interference by Adaptive MAC Allocation in Sensor Networks*", in Proceedings of the Sixth IEEE International Symposium on a World of Wireless Mobile and Multimedia Networks 2005.

30. Irfan Ahmed, Mugen Peng, Prof. Wenbo Wang, "*A Unified Energy Efficient Cluster ID based Routing Scheme for Wireless Sensor Networks – A more Realistic Analysis*", in Third International Conference on Networking and Services 2007.

31. *Microsoft ENCRATA World English Dictonary*

32. Ben L. Titzer, Daniel K. Lee, and Jens Palsberg, "*Avrora: scalable sensor network simulation with precise timing*", in Fourth International Symposium on Information Processing in Sensor Networks (IPSN), pages 477 – 482, UCLA, Los Angeles, CA, USA, April 2005.

33. Manish Karir, Jonathan Polley, Dionysys Blazakis, Jonathan McGee, Dan Rusk, and John S. Baras, "*Atemu: a fine-grained sensor network simulator*", in First Annual IEEE Communications Society Conference on Sensor and Ad Hoc Communications and Networks, pages 145 – 152, October 2004.

34. Jeremy Elson, Lewis Girod, and Deborah Estrin, "*Emstar: Development with high system visibility*", in Wireless Communications, pages 70 – 77, December 2004.

35. Rimon Barr, "*Swans: Scalable wireless ad hoc network simulation*", in Handbook on Theoretical and Algorithmic Aspects of Sensor, Ad hoc Wireless, and Peer to Peer networks. Ch. 19, CRC Press, pages 297–311, (http://jist.ece.cornell.edu/docs.html), September 2005.

36. http://www.scalable networks.com/.

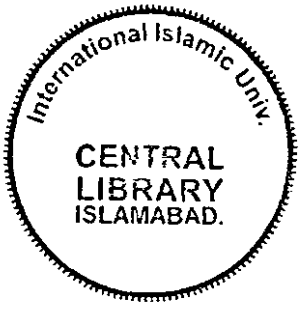
37. http://www.opnet.com/.

38. http://www.cs.rpi.edu/ cheng3/sense/.

39. Philip Levis, Nelson Lee, Matt Welsh, and David Culler, "*Tossim: Accurate and scalable simulation of entire tinyos applications*", in Proceedings of the First ACM Conference on Embedded Networked Sensor Systems (SenSys 2003). (http://webs.cs.berkeley.edu/users/nestfr/nestfr_paper_display.php), November 2003.

40. "TOSSIM: A Simulator for TinyOS Networks", Philip Levis and Nelson Lee. UC Berkeley September 17, 2003.

41. http://www.ee.duke.edu/~romit/courses/s07/material/lecture-sensor-ddiffusion-leach.ppt.



Appendix A

List of Abbreviations

WSN:	Wireless Sensor Network
AICR:	Adaptive Intra Cluster Routing
CH:	Cluster Head
BS:	Base Station
TOSSIM:	TinyOS Simulator
NesC:	Network Embedded System C
QoS:	Quality of Service
CPU:	Central Processing Unit
FPGA:	Field Programmable Gate Array
DSP:	Digital Signal Processor
ASIC:	Application Specific Integrated Circuit
DPM:	Dynamic Power Management
DVS:	Dynamic Voltage Scaling
LEACH:	Low-Energy Adaptive Clustering Hierarchy
TOSSIM:	TinyOS Simulator
NesC:	Network Embedded System C
DBG:	Debug
LASER:	Light Amplification by Stimulated Emission of Radiation
GHz:	Giga Hertz
ROM:	Read Only Memory
RAM	Random Access Memory
EEPROM:	Electronically Erasable Programmable ROM
EYES:	Energy Efficient Networks
FU:	Free University
SOS:	Sensor Network Operating System
EECR:	Energy Efficient Clustering Routing
AM:	Active Messages
CIDRSN:	Cluster ID based Routing in Sensor Networks
RSSI:	Received signal Strength Indicator
CRC:	Cyclic Redundancy Check
ADC:	Analog to Digital Converter
GUI:	Graphical User Interface
IEEE:	Institute of Electrical and Electronics Engineers

Appendix B

Semantic of Proposed Algorithm

Semantic of proposed algorithm is discussed in this section. The Fig B1 shows how components and interfaces are collaborated with each other in the application. As discussed in section 5.6.1 that in TinyOS different components are connected via interfaces to perform collective functionality of an application. In proposed algorithm, component *Main* is initiating the application. Component *Main* is connected to other components through the available interface *StdControl*. *SurgeM* is the core component of this application which is performing the major task for this application while collaborating with other available components through respective interfaces. As in this application light data is sensed, so component *Photo* is used. For time based computing component *TimerC* is used.

Another important component in this application is *MultiHopRouter*. For multihop routing parent selection and neighbor table updating for routing is done by his component. Component *SurgeM* is using this component while adopting multihop as mode of routing. The edges in this graph show the interfaces and nodes show the component. Sending and receiving of the data towards other nodes in the network is done by *GenericCommPromiscuous* component.

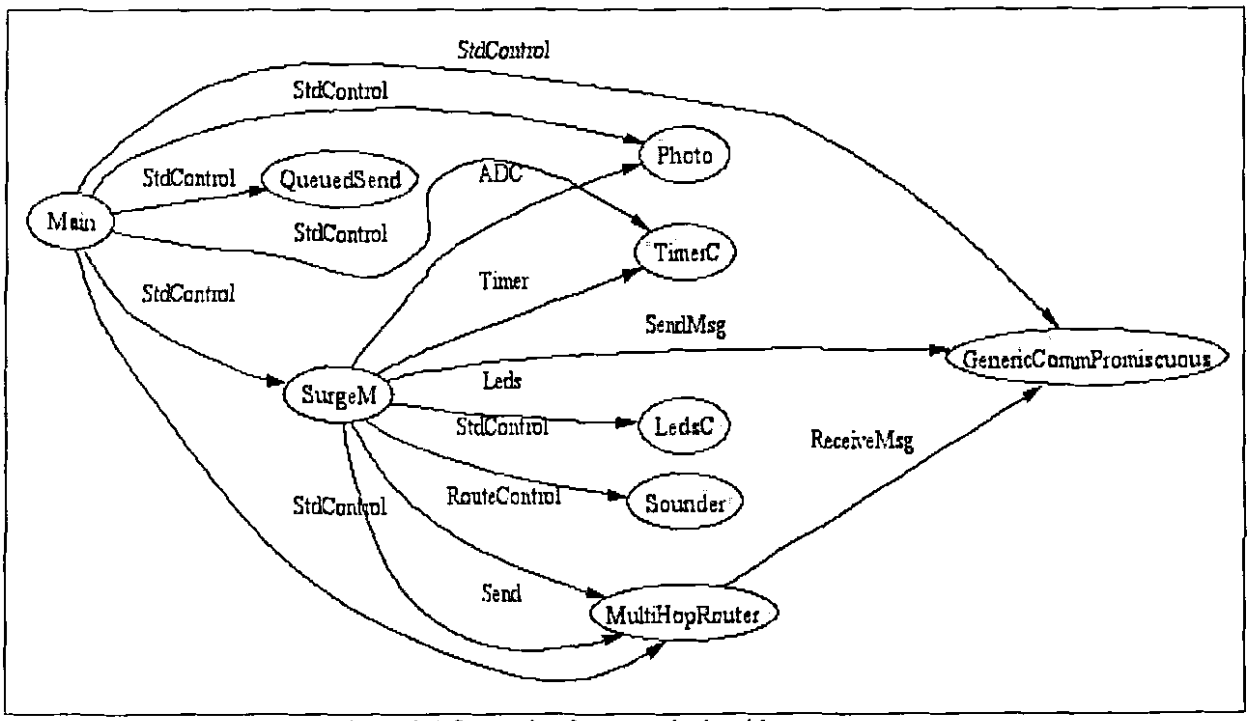


Figure B 1 Semantic of Proposed Algorithm

Different components are involved in multihop routing. *MultoiHopRouter*, *MultiHopEngineM* and *MultiHopLEPSM* are three major components involved in multihop routing. The implementation uses the shortest-path –first algorithm with node 0 as ultimate destination node. Fig B2 shows a semantic of multihop routing application.

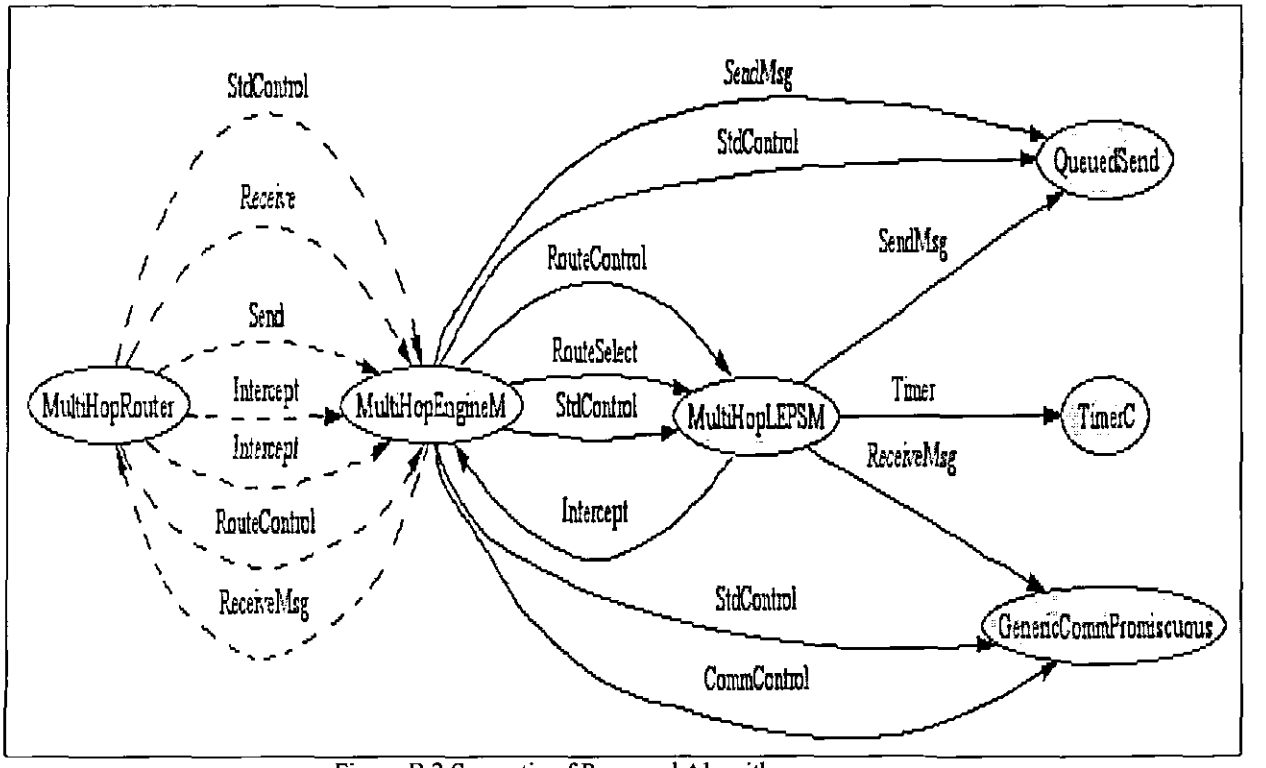


Figure B 2 Semantic of Proposed Algorithm

For multi-hop routing the overall packet movement logic is provided by *MultiHopEngineM*. Next hop path is selected using its *RouteSelect* interfaces.

Link Estimation and Parent Selection (LEPS) mechanisms for the multi-hop implementation is provided by MultiHopLEPSM. Next hop destination is carried by this component based on shortest path semantic. The ultimate destination is identified as the node 0.

MultiHopEngineM and *MultiHopLEPSM* are connected with other necessary components using *MultiHopRouter* configuration. This configuration is providing the overall wiring of the multihop routing algorithm. The dotted lines shows that *MultiHopRouter* is a configuration part (file) of the application as mentioned in section 5.6.1.1. For queuing outbound packets the *SendMsg* port of *MultiHopEngineM* is wired with *QueuedSend* component. For single hop route information exchange ReseiveMsg and SendMsg interfaces of *MultiHopLEPSM* are wired with *AM_MULTIHOPMSG* parameter.