Sudan University of Science and Technology

College of Graduate Studies

كلية الدراسات العليا

Performance Evaluation of Ant Colony Optimization for Routing in Wireless Sensor Networks

تقييم أداء تعزيز مستعمرة النمل للتوجيه في شبكات المحسسات اللاسلكية

A thesis submitted in partial fulfillment of the requirements for the degree of M.Sc. in Electronics Engineering (Computer and Networks Engineering)

By:

Tagwa Musa Hassan Musa.

Supervisor:

Dr. Fath Elrahman Ismael Khalifa Ahmed.

Dec - 2018

DEDICATION

This thesis is dedicated to my beloved family and friends
who have always been constant sources of support
and encouragement.
To my lovely mom who has always supported me
and stand with me.

ACKNOWLEDGEMENTS

I express my gratitude towards The Almighty God for His blessings upon me. I owe my profound gratitude to my thesis supervisor Dr.Fath Elrahman Ismael Khalifa for his valuable guidance, supervision and persistent encouragement due to the approach adopted by him in handling my thesis. It is extremely hard to find words that express my gratitude to my parents, my sibling and my friends for their invaluable help over this year. I wish them all good luck in their future plans. They gave me courage and strength whenever I needed and supported me in every possible way throughout these years.

Tagwa Musa

المستخلص

شبكات الإستشعار اللاسلكية تتكون من عدد كبير من نقاط الوصول و المحسسات. في شبكات الإستشعار اللاسلكية تكمن المشكلة الأساسية في محدودية عمر بطاريات الشحن المستخدمة بواسطة المحسسات و تحرك نقاط الوصول في الشبكة. لتجنب هذه المشكلة، تم طرح مقترح بروتوكولات نقاط الوصول المتحركة. تساعد هذه البروتوكولات في تحقيق توازن و توحيد الطاقة المستهلكه عبر الشبكة. يهدف هذا البحث في التركيز علي المتطلبات المتغيره لنقاط الوصول المتحركة بإلقاء نظرة عامه علي بروتوكولات نقاط الوصول المتحركة وإجراء مقارنة للأداء بين خوارزمية مستعمرة النمل و بروتكول تكيف تقليل الطاقة في العنقود الهرمي. هذه البروتوكولات تم تنفيذها بواسطة برنامج محاكاة الشبكات النسخة الثانية. هذا البرنامج يقارن بروتوكولات التوجيه عبر استخدام بروتوكول تحكم الإرسال و بروتوكول مخطط المستخدم مع حساب معايير أداء مختلفة. أداء هذين البروتوكولين تم تحليلهما عبر خمسة معايير: الإنتاجية، زمن التأخيرالكلي، معدل إيصال الحزم، إستهلاك الطاقة وزمن عمر الشبكة. أظهرت التجربة لعدد 80 نقطة إنه لخوارزمية مستعمرة النمل أعلي إنتاجية بنسبة 0.8% من بروتوكول تكيف تقليل الطاقة في العنقود الهرمي (تحت بروتوكول تحكم الإرسال). وسُجل زمن التأخير ب 0.007 مللي ثانية في خوارزمية مستعمرة النمل (تحت بروتوكول مخطط الإستخدام)، أفضل قراءه سُجلت كانت 0.005 مللي ثانية عند 20 نقطة. و سُجل معدل 92.20% كمعدل إيصال للحزم في خوارزمية مستعمرة النمل كما سجل بروتوكول تكيف تقليل الطاقة في العنقود الهرمي معدل 51.20% (تحت بروتوكول مخطط الإستخدام). و سجلت خوارزمية مستعمرة النمل أقل قراءه لإستهلاك الطاقة بمعدل 5.5% بينما سجل برتوكول تكيف تقليل الطاقة في العنقود الهرمي معدل 9.62% (تحت بروتوكول تحكم الإرسال). زمن عمر الشبكة سجل أعلي قراءه له بعدد النقاط الحيه مقارنة بالزمن، في خوارزمية مستعمرات النمل بعدد 59 نقطه حيه عند الزمن 2000 ثانية، وسجل بروتوكول تكيف تقليل الطاقة في العنقود الهرمي عدد 7 نقاط حيه فقط.

ABSTRACT

A wireless sensor network is composed of a large number of sensor nodes and a sink. In the wireless sensor networks the main problem is limited battery life used by sensor nodes and the mobility of the sink nodes in the network. To avoid this problem, protocols with mobile sinks were proposed. They helped in achieving load balancing and uniform energy consumption throughout the network. This research aimed to concentrate on the dynamic requirements of the mobile sink by providing an overview of mobile sink protocol concerns and performance comparison between Ant Colony Optimization (ACO) and Low-Energy Adaptive Clustering Hierarchy (LEACH) protocol. The routing protocols are implemented using NS-2.34. The simulation compares the routing protocols with using transmission control protocol (TCP) and user datagram protocol (UDP) with different performance metrics. The performance of these routing protocols analyzed using five metrics: throughput, end to end delay, packet delivery ratio, energy consumption and network lifetime. Simulation results of 80 nodes showed that the ACO has high throughput with 0.8% than LEACH under TCP traffic. Average end to end delays record the value of 0.007 ms in ACO under UDP traffic (best record was 0.005 ms with 20 nodes). PDR was 92.20% in ACO than LEACH with 51.20% under UDP traffic. ACO recorded lowest energy consumption with 5.50% and 9.62% for LEACH under TCP traffic. Network lifetime was recorded high number of alive node per time in ACO had 59 alive nodes at round 2000, where LEACH had only 7 nodes at that time.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF EQUATIONAS

LIST OF SYMBOLS

Ptk_r	Successfully Delivered Packets
Ptk_q	Required Packets
Tr_i	Time of Received Packet
Ts_i	Time of Send Packet
Ptk_N	Total No. of Packets Received
Ptk_{size}	Packet Size
T	Total Duration of Simulation
E_{ik}	Initial Energy Level of Node
E_{fk}	Final Energy Level of Node
N	Number of Nodes in Simulation
E_i	Initial Energy
E_w	Expected Wasted Energy
E_c	Energy Consumption of Network
E_s	Energy Consumed by Sensor

LIST OF ABBREVIATIONS

ACO	Ant Colony Optimization
CBR	Constant Bit Rate
CH	Cluster Head
CHR	Cluster-Head Relay Routing
ECRA	Enhanced Connection Resolution Aloha
FTP	File Transfer Protocol
GAF	Geographic Adaptive Fidelity
GEAR	Geographic and Energy-Aware Routing
GloMoSim	Global Mobile Information Systems Simulation
HEED	Hybrid Energy-Efficient Distributed Clustering
IDSQ	Information-Driven Sensor Query
LEACH	Low-Energy Adaptive Clustering Hierarchy
MAC	Media Access Control
MBS	Mobile Base Station
MDC	Mobile Data Collector
MECN	Minimum Energy Communication Network
MMSPEED	Multi-path Multi-speed SPEED
MWSN	Mobile Wireless Sensors Network
NAM	Network Animator
NS2	Network Simulator 2
OMNET++	Operation and Maintenance New Equipment Training
OPNET	Optimized Network Engineering Tool
OTCL	Object oriented Tool Command Language
PDR	Packet Delivery Ratio

PEGASIS	Power-Efficient Gathering in Sensor Information System
SAR	Sequential Assignment Routing
SEP	Stable Election Protocol
SPIN	Sensor Protocols for Information via Negotiation
TCLCL	TCL Classes
TCP	Transmission Control Protocol
TDMA	Time Division Multiple Access
TEEN	Threshold sensitive Energy Efficient sensor
TTDD	Two-Tier Data Dissemination
UDP	User Datagram Protocol
VM	Virtual Machine
WSN	Wireless Sensors Network

CHAPTER ONE

INTRODUCTION

Chapter One
Introduction

1.1 Preface

Wireless Sensor Networks (WSNs) consist of many sensor nodes, which are usually distributed across areas difficult to be accessed in order to collect and send the data to the main sink location. Despite the fact that a number of protocols have been proposed for routing and energy management, WSNs still face problems in selecting the best path with efficient energy consumption and successful delivery of the packets. In particular, these problems occur when WSNs are subjected to critical situations such as node or link failure, and it is even more critical in sensitive applications such as nuclear and healthcare [1].

Wireless Sensors Networks (WSN) consists of spatially distributed autonomous sensors to monitor physical or environmental conditions, such as temperature, sound, vibration, pressure, motion or pollutants, and to cooperatively pass their data through the network. The more modern networks are bi-directional, enabling also to control the activity of the sensors. The development of wireless sensor networks was motivated by military applications such as battlefield surveillance; today such networks are used in many industrial and consumer application, such as industrial process monitoring and control, machine health monitoring, environment and habitat monitoring, healthcare applications, home automation, and traffic control. [2]

The main problem in WSN, sensor node have limited battery life because the sensor nodes size is small so battery size, processor, storage space for data, these all are little as sensor nodes. In WSN a bundle of sensed information and routing information has to be sending which after have some time constraints so that information can be employable before any disaster occurs like manufacturing monitoring, apparatus monitoring etc. In WSN the power influence utilization is much superior data communication then internal processing. Energy preservation in WSN is need to the address. Wireless sensor is flat to node breakdown due to power hammering. In order to provide a consistent service through network, the network should be personality adjusting and must have flexible properties as requisite from time to time. A restricted access node may encounter failure due to imperfect battery existence. In such case the network protocol should be intellectual enough to such collapse to keep the network operational many techniques are proposed for energy discount, clustering is single of them. The cluster heads are voted sporadically such that members of a cluster can communicate with their cluster heads. These cluster heads send data acknowledged from its members to a base station. The multi clustering can also be used. The cluster head should have to be rotated for the complementary of energy and then there will be equivalent weight on every node. The energy expenditure can be condensed [3].

1.2 Problem Statement

The main problem in WSN is the limitation off energy sources of the nodes which is to be kept very low; sensors consumed lot of energy in routing which decrease the alive time of nodes in the network.

1.3 Proposed Solution

In this work, the performance evaluation is carried out in order to determine the best routing protocol which takes less energy consumption under the different sizes of network. This thesis compares performance analysis of ACO and LEACH Routing Protocols for WSN using NS-2 based on performance metrics such as throughput, packet delivery ratio, energy consumption, network lifetime and end to end delay.

1.4 Aims and Objectives

The aim of this thesis is performance evaluation of ACO and LEACH protocols in WSNs. The outcome of this study is in the form of quantitative results of the efficiency of the routing protocols with performance metrics. These results can be used as the baseline for selecting routing protocols in a variety of situations.

The objectives of this thesis are:

- To implement different network scenarios using the NS2 simulator for different routing protocols.
- To analyze and compare the performance of TCP/FTP and UDP/CBR traffic in ACO and LEACH routing protocol generally implemented in WSN environment with different performance metrics such as throughput, packet delivery ratio, energy consumption, network lifetime and end to end delay.

1.5 Methodology

- Network Simulator 2 is chosen as a simulation environment to create experiment scenarios and conduct the simulations.
- Scenarios are generated by TCP and UDP traffics with varying the numbers of nodes.
- The detailed trace files created by each run are stored on disk, and analyzed using a script-routine (written in awk script).
- The NS2 simulator gives two files as output; NAM file, which is used for graphical visualization and trace file which is used for calculating the results.

1.6 Thesis Outlines

The thesis includes five chapters, Chapter One provides introduction to the WSN, the problem statement and objectives while Chapter Two covers background study of WSN routing protocols for mobile sink and literature review. In Chapter Three the methodology section, where the framework of the simulator, routing metric, implementation and simulation environment are defined while Chapter Four presents performance evaluation results. And Chapter Five includes the conclusion and recommendation for future work.

CHAPTER TWO

LITERTURE REVIEW

Chapter Two

Literature Review

2.1. Background

Wireless Sensor Networks (WSN) consists of much small energy constrained nodes. Each node consists of a sensing or actuating unit, a processing unit(CPU, micro controller or DSP kit), memory(to store data or program),an RF transceiver(usually with an Omni directional antenna) and power(battery or solar). Desirable functionality of sensor nodes in a WSN include: ease of installation, self-indication, self-diagnosis, reliability, time awareness for coordination with other nodes, some software functions and DSP, and standard control protocols and network interfaces. Route maintenance and data dissemination are the two processes that consume energy in WSNs. Frequent network maintenance and node replacement isn't possible in WSNs due to the nature of the environment where they are deployed. Hence WSN must be fault tolerant. Multipath routing is an option to continue the networking in the presence of faults. But this requires more control overhead [4].

2.1.1 Mobility in WSN

In WSN, the two categories of sink are static sink and mobile sink. The static sink is stationary on a particular position while mobile sink can move across the network. Due to the increasing demand for energy efficient routing and reliable data delivery, mobile sink's concept became prevalent. As the nodes can survive for more time and give better results. In the case of a mobile sink with fixed paths the choice of the mobility path influences energy efficiency [5].

Recent researches have shown that introducing mobility in wireless sensor network is advantageous as the mobile nodes can relocate after initial deployment to achieve the desired density requirement and reduce the energy holes in the network thereby increasing the network life time [4].

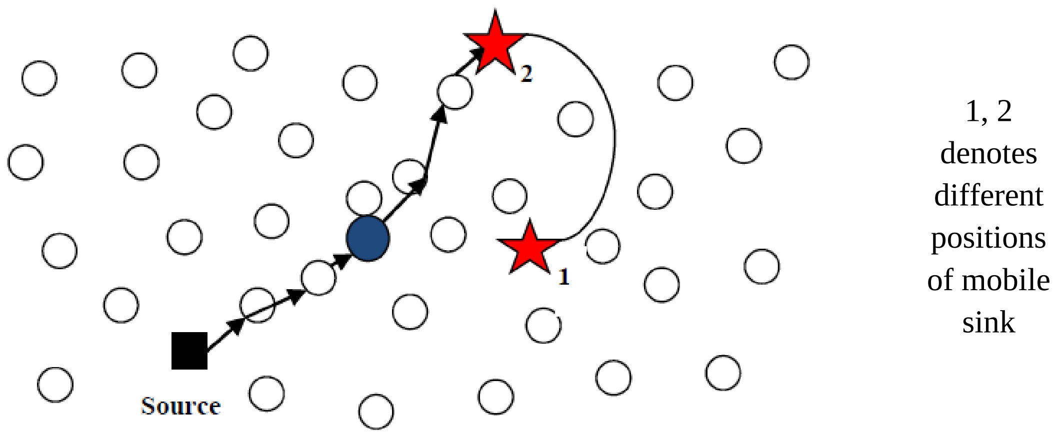

Figure 2.1: Mobile Sink

2.1.2 Sink Mobility Patterns

Depending on the requirement of the application, it could be seen from two perspectives [5].

Sink's perspective- It reflects the true motion pattern of the sink. Continuous: In this, the sink nodes follow a particular pattern for e.g. cyclic, straight line etc. Nomads: Here, the mobile sink doesn't follow a particular pattern, they move here and there like nomads.

Sensor’s perspective- It tells about the sink mobility with respect to the sensor node’s limited knowledge.

Table 2-1 Node Mobility in WSN

<table>
<tr><td rowspan="12">Node Mobility</td><td rowspan="10">Sensor’s Perspective</td><td rowspan="7">Heterogeneous Mobility</td><td rowspan="3">Geographic Model</td><td>Mobile Base Station (MBS)</td></tr>
<tr><td>Mobile Data Controlled (MDC)</td></tr>
<tr><td>Rendezvous Based Solution</td></tr>
<tr><td rowspan="2">Random Model</td><td>Pause Period</td></tr>
<tr><td>Motion Period</td></tr>
<tr><td colspan="2">Controlled Model</td></tr>
<tr><td colspan="2">Predictable Model</td></tr>
<tr><td rowspan="3">Homogeneous Mobility</td><td rowspan="2">Random Model</td><td>Partially Random</td></tr>
<tr><td>Totally Random</td></tr>
<tr><td colspan="2">Controlled Model</td></tr>
<tr><td rowspan="2">Sink Perspective</td><td colspan="3">Continuous</td></tr>
<tr><td colspan="3">Nomads</td></tr>
</table>

Introducing mobility to some or all nodes in a WSN, improves the network lifetime. It also provides more channel capacity and enhances coverage and targeting. The basic architecture of a three tier Mobile Wireless Sensor Network, the sensor nodes are deployed randomly in the network. These nodes can communicate with each other and the mobile agents. The mobile agents can move anywhere and at any time and they are responsible for collecting the sensed data and forward them to the fixed network consisting of Access Point [4].

There are various approaches for studying the mobility for data collection in WSNs. Mobility pattern of various nodes in the network must be modeled, which could be incorporated for various routing protocols. According to ref a survey of mobility models for WSN, the mobility model can be mainly categorized into two:

The homogeneous mobility model is the one in which a group of mobile sensor nodes move according to the same model in the given deployment area. They can be further categorized into two - Random model and Controlled model. The Random mobility model can be further categorized into Partially Random and Totally Random models. In Partially mobility model, the mobile nodes depend on each other to specify the movement direction in the network. Totally Random mobility model will allow the group of mobile nodes moving in a random direction. The Controlled mobility model will allow a set of nodes to move in a specified direction [4].

In heterogeneous mobility model, the mobile nodes will move independently without depending on any other node in the network. Various nodes in the network will move according to their adopted mobility model. Thus in a network, various mobility models are adopted. Heterogeneous mobility model can be further categorized into four categories - Random mobility model, Controlled mobility model, Predictable mobility model and Geographic mobility model. The Random mobility model will divide the motion of the mobile node into pause period and motion period. They will allow the nodes to move in the network in a random pattern. In Controlled mobility model, the mobile nodes would visit the sensor nodes based on the predefined schedule that is built based on the sampling rate of the sensors and event occurrence rate. The next classification of mobility model, Predictable mobility model where the sensor nodes know the path in which the mobile sinks will use. Until the predicted time of data transfer, the sensor nodes will be in sleep mode, thus saving a large amount of energy. After that, the sensor nodes go to active mode and will start sending data to the mobile sink. The geographic mobility model is the one in which the mobile nodes movement can be restricted according to the geographic nature of the environment in which a mobile node or sink is deployed such as: Mobile base station (MBS)-based, Mobile data collector (MDC)-based, Rendezvous based solutions [4].

In a classic WSN, where all the nodes are static, lot of energy gets depleted for the node close to the sink. This excessive energy expenditure is due to the continuous transmission and response by the sensor node close to the sink. The primary aim of MBS based solutions is increasing the lifetime of the network by evenly distributing the energy consumption. In case of MDC based solutions, the data are gathered from sensors by visiting them individually. Based on the mobility pattern of the MDCs, there can be Random mobility, Predictable mobility, and Controlled mobility [4].

2.1.3 Advantage of Employ Mobile Sink over Static

Experimental result shows that the sensor node which are near or one hop away from base station drains their energy level faster than the other nodes of the network. Nodes which are one hop away from base station need to forward own message as well as forward message originating from other nodes. In doing so, sensor nodes drain their energy level and became inactive or dead. As a result many sensor nodes will be dead and unable to forward the message to base station and network communication become dead. To increase the life time of sensor network, there is need to deploy multiple base stations and periodically change their locations [6].

2.1.4 Routing Protocols for Mobile Sink WSN

The routing protocols for MWSN can be mainly classified based on the network structure, state of information, mobility and energy efficient techniques. Based on the mobility of nodes, various routing protocols have been studied. The major challenge for defining the routing protocol in MWSN occurs due to the fact that the topology of the network is periodically changing. Due to the mobility of the sink, the set of sensors located near the sink changes over a time. This would help in balancing the energy consumption and thereby prolonging the network life time. The sink can follow three types of mobility patterns in MWSN: [7]

Random mobility: In this mobility, sink can select the random path in the sensor field to collect the data. There are two strategies for collecting the data which are push and pull strategy. The sink uses the pull strategy for collecting the data. In pull strategy, data can be forwarded from the node only when sink initiates a request to collect the data otherwise node waits until sink initiates. The maximum energy consumption will reduce in random sink mobility when compared with static sink. Even though single hop data collection reduces the energy consumption but it also results in incomplete data collection because there is no determined path in random mobility. Other important issue is the coverage time and energy dissipation. If multiple mobile sink moves randomly in a field, coverage time gets reduced. A path coordination method in which sinks leaves the trail in its path. When other sinks come across this trail it will changes its path and reach other direction. This will increase the coverage of the network but this system will increase the overhead and additional energy dissipation [8].

Fixed mobility: In fixed mobility, sink follows a fixed path in a round robin manner. The path fixed is predetermined and it cannot be changed by the WSN in any situation. Hence the coverage can be achieved more by determining the routing path for data packets through fixed path. The most interesting feature of this is whether the sink can able to predict its future position or not. This method can forward the data by means of pull strategy based on a request message by sink. Hence mobility path is fixed so the method can achieve the energy dissipation very low and also sink easily predict its future position [8].

Controlled mobility: In controlled mobility, mobility of the sink is controlled by some parameters of interest such as residual energy, predefined events. In this mobility sink is placed in any mobile entity for example sink is placed in mobile robot or any moving vehicle. The mobile entity is integral part of the network and it can be controlled fully [8].

2.1.5 Classification of Routing Protocols for Mobile Sink WSN

Many routing algorithms were developed for wireless networks. All foremost routing protocols planned for WSNs may be divided into five categories as shown in figure 2.2 [10].

Hierarchical protocols: In this segment, we evaluation a sample of hierarchical-based routing protocols for WSNs. Hierarchical protocols is a cluster based protocols. Clustering is an energy-efficient communication protocols that can be used by the sensors to account their sensed data to the base station [10].

Data-Centric protocols: Data-centric protocol is used to control the redundancy it happens because sensor node does not have global identification number which contains uniquely, so data is transmitted to each node with significant redundancy of data .In data centric routing, the sink request for data by sending the query so the nearest sensor node transmits the data selected understands from the query [10].

Heterogeneity-based Protocols: Heterogeneity sensor networks are of two types of sensors namely line-powered and battery powered. In this segment we talk about uses of heterogeneity based protocols in WSNs to expand the network lifetime [10].

Location-Based protocols: Sensors nodes are pointed by mean their positions. Position information for sensor nodes is required for the sensor networks by the majority of the routing protocols to add the distance between two exacting nodes so that energy expenditure can be reduced [10].

Qos-based protocols: In addition to diminish energy utilization, it is also important to consider quality of service (Qos) condition in terms of delay, reliability, and fault tolerance in routing in WSNs [10].

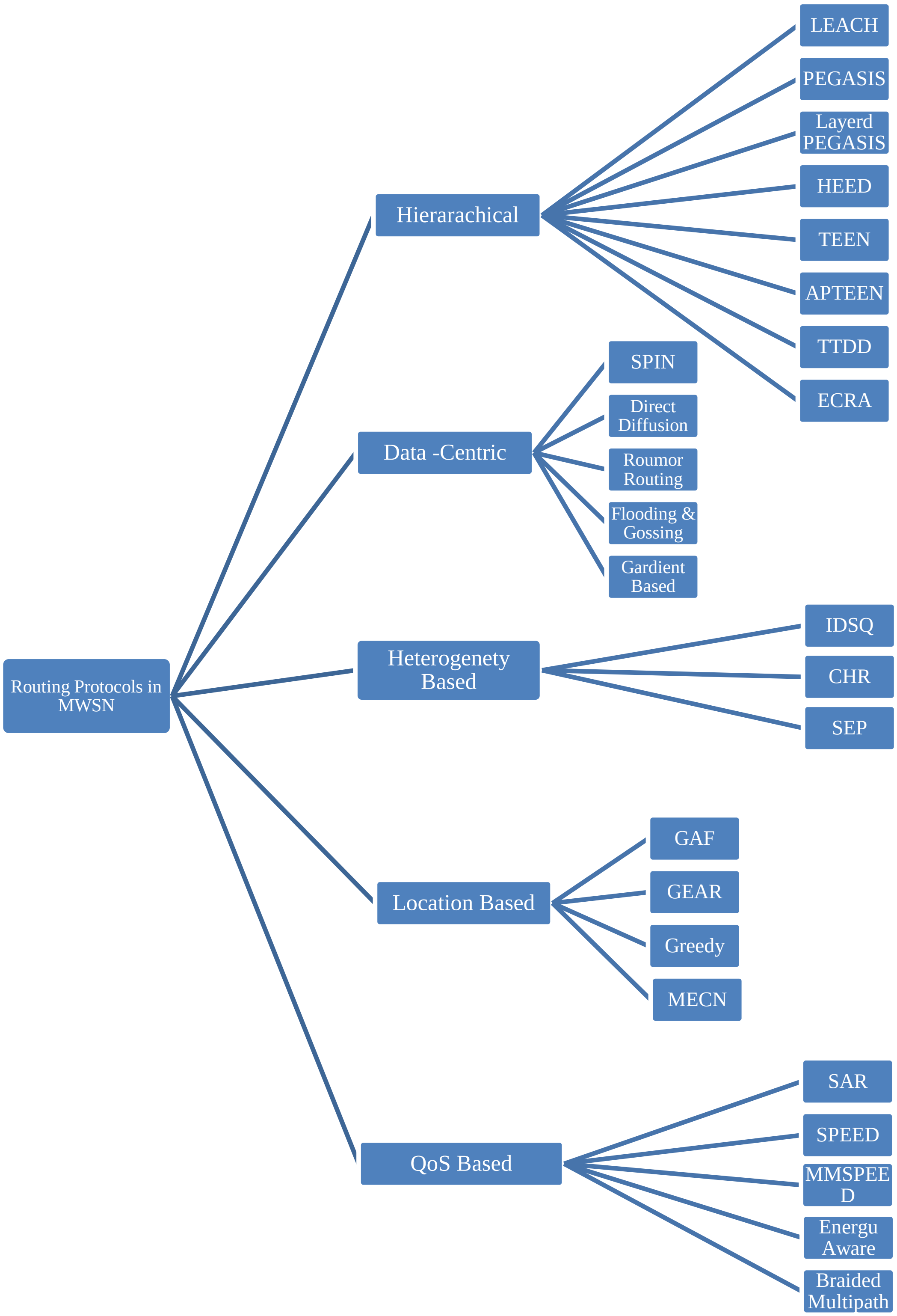

Figure 2.2: Routing Protocols for Mobile

2.1.6 Ant Colony Optimization (ACO)

ACO Routing comes under the optimization problem for finding the shortest path from source to the destination. Ants have been there for millions of years and they detect shortest path between the food sources and nest. An ant secretes a volatile chemical substance called pheromone that helps in converging over the shortest path among multiple paths. Ants secrete pheromone on the ground while moving and follow the path with maximum pheromone concentration. This mechanism helps the ant to choose the best path and the same has been proved to be to generate optimal path from among multiple paths [9].

The ant behavior is mapped in electronic devices for solving various combinatorial problems. While traversing through the phases of the problem, asynchronous agents are included to produce partial results. Greedy approach is followed for arriving at the solution in incremental way in each phase. The same approach is used for routing the data packets from the source to destination in the wireless sensor networks. Amount of network packet transmission, energy efficiency and increasing the network lifetime are very important in a wireless sensor network.

An ACO algorithm has two phases where the capability of the algorithm is enhanced which is trail evaporation and daemon actions. To hold the unlimited accumulation of trail over specific component trail evaporation is done. To implement the actions that cannot be performed by a single ant, the Daemon actions are used. [9].

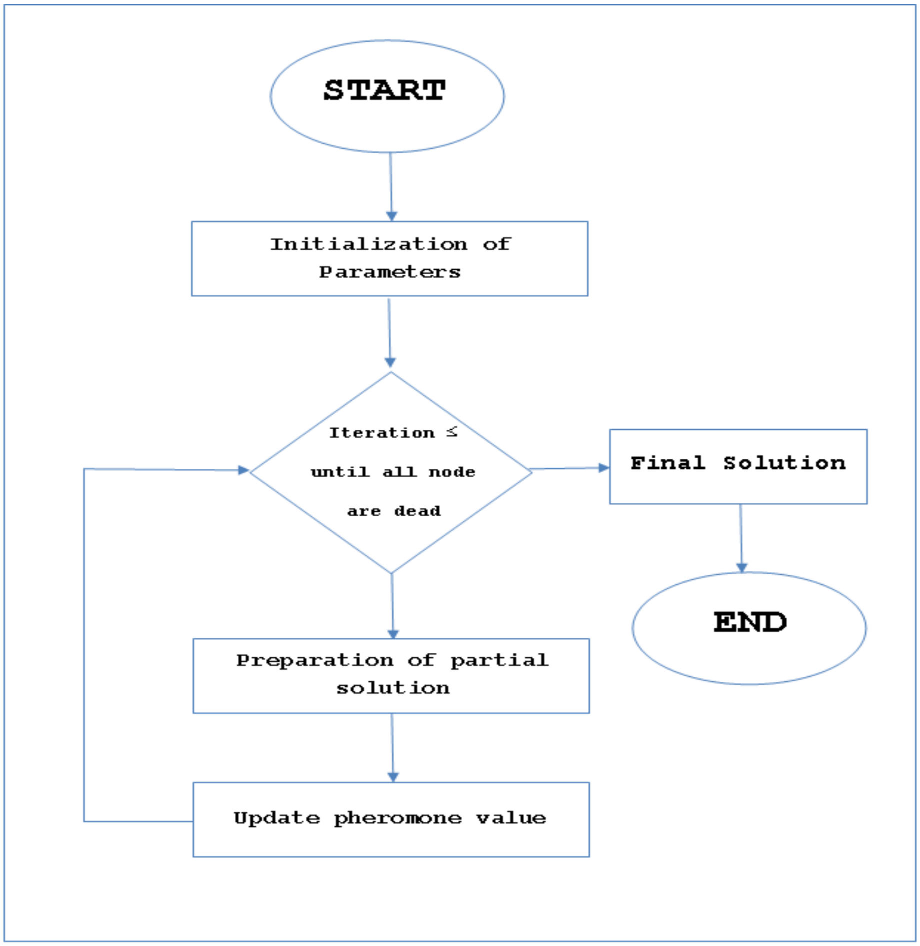

Figure 2.3: Flow chart of Ant Colony Optimization

2.1.7 Low-energy adaptive clustering hierarchy (LEACH)

Low Energy Adaptive Clustering Hierarchical Protocol, It is a routing protocols and also known as cluster based protocols. LEACH protocol provides communication between two sensor nodes in WSN. LEACH is most commonly used protocol in WSN [10].

Limitation in LEACH protocol is chosen of CH [Cluster Head] randomly is main problem of LEACH protocol because when CH [Cluster Head] is selected in randomly way then there is no record account for energy consumption. So a node with low energy has same probability as node of high energy. If node with low Energy is chosen as CH [Cluster Head] then this node will die soon due to which WSN cannot exist for a long time. There are some modified versions of LEACH protocol like V-LEACH protocol, M-LEACH protocol, E-LEACH protocol, and O-LEACH protocol [10].

In LEACH the role of the cluster head is periodically transferred among the nodes in the network in order to distribute the energy consumption. The performance of LEACH is based on rounds. Then, a cluster head is elected in each round. For this election, the number of nodes that have not been cluster heads and the percentage of cluster heads are used. Once the cluster head is defined in the setup phase, it establishes a TDMA schedule for the transmissions in its cluster. This scheduling allows nodes to switch off their interfaces when they are not going to be employed. The cluster head is the router to the sink and it is also responsible for the data aggregation. The cluster head controls the sensors located in a close area and the data aggregation performed by this leader permits to remove redundancy [10].

A centralized version of this protocol is LEACH-C. This scheme is also based on time rounds which are divided into the set-up phase and the steady-phase. In the set-up phase, sensors inform the base station about their positions and about their energy level. With this information, the base station decides the structure of clusters and their corresponding cluster heads. Since the base station possess a complete knowledge of the status of the network, the cluster structure resulting from LEACH-C is considered an optimization of the results of LEACH [11].

Leach is adaptive clustering routing protocol that minimizes the energy drain in wireless sensor network. Leach is clustering based protocol that defines a whole WSN in form of different clusters. LEACH divides a network into finite number of cluster, in each cluster there are some members and one cluster head or coordinator exists. Cluster head collect the data from source node and send to sink. Leach protocol has a several round and each round contains two phase. First phase is cluster formation and head selection phase and in second phase data dissemination occur between source node to cluster head and cluster head to mobile sink. Leach uses randomization approach to balance the load on sensor nodes [6].

2.2 Related Works

Several performance evaluations of MWSN routing protocols using TCP and UDP traffic have been done by considering various parameters such as mobility, network load and network lifetime.

In [12] the authors compared ACO and LEACH by using two traffic generators namely residual energy and End to end delay for N-cluster and common node over WSN and analyzed the behavior of the routing protocols using NS2 as simulation environment. Simulation scenario setup was implemented using 25, 50, 75 ~ 300 sensor nodes with n-cluster and common nodes (c-nodes). Simulator was defined a multi-path route between the sensor nodes and cluster head (CH) and single –hop routing between cluster head and base station (BS) for transmission of data packets. ACO is shown better performance than LEACH in residual energy (Remaining energy) for both N-cluster and common node and end to end delay metric had it is minimum value in ACO than LEACH for both N-cluster and common node. The chosen of two matrices is determine as limitation for more performance matrices to be apply and analyzed the protocols based on.

The authors of [13] evaluated proposed Ant Colony Optimization (ACO) algorithm for routing of data packets in wireless sensor networks with LEACH protocol in terms of higher energy efficiency, prolonged network lifetime, enhanced stability period, and the higher amount of data packets transmitted to base station in a densely deployed wireless sensor network.

They were simulated the two algorithm in NS2, the performance of LEACH algorithm is compared with the ACO algorithm. Sensor nodes were been distributed randomly in $100*100$ square. All the nodes have same transmission range. The nodes have their horizontal and vertical coordinates located between 0 and maximum value of the dimension which is 100. Simulation results depict that ACO has higher energy efficiency in dense environment than the existing LEACH protocol. In LEACH when the number of nodes increases from 100 to 200 or 300, the performance of LEACH protocol decreases while the performance of ACO either increases or remains stable. In this algorithm, cluster heads are selected randomly in homogeneous environment with static sensors which made limitation for mobile sensors (mobile sink) needed to be evolution.

CHAPTER THREE

SIMULATION STEPS

Chapter Three
Simulation Steps

3.1 Overview

This section covers techniques, tools, performance metrics which are chosen for evaluating the performance of protocols. The importance of performance evaluation and simulation are also described in this section.

3.2 Importance of Performance Evaluation and Simulation

In a computer system performance is a key factor. All the software and hardware design go through the performance tests again and again before implementation. Integration of computer system in almost every walk of life demands a reliable computer network system. It is therefore considerd necessary for all computer professionals, researchers and system engineers to acquire basic knowledge of performance evaluating technique. Performance can be evaluated via measurement or modeling and simulation [14]. The simulation technique is suitable for testing models especially in research areas and educational centers. Potential advantages of the simulation are, it saves time, cost and provides detail results and a good understanding of event's occurrence.

3.3 Network Simulator

Network Simulator (Version 2), widely known as NS2, is simply an event driven simulation tool that has proved useful in studying the dynamic nature of communication networks. Both wired and wireless network functions and protocols (e.g., routing algorithms, TCP, UDP) can be simulated using NS2.

In general, NS2 provides users with a method of specifying such network protocols and simulating their corresponding behaviors. There are many simulators such as OPNET, NetSim, GloMoSim, NS3, OMNET++ and NS2 etc. NS2 is used for simulation due to it is free, open source , support different types of networks such as wired Network, wireless ad-hoc mode, wireless managed mode and wired cum wireless [15] . Also NS-2 comes closer to reality than other simulators, NS-2 has the rich collection of models than others simulators and A good simulation design, good results can be achieved with NS-2.

3.4 Architecture of NS2

Fig 3.1 shows the basic architecture of NS2. NS2 provides users with an executable command NS which takes on input argument, the name of a Tcl simulation scripting file. Users feed the name of a Tcl simulation script (which sets up a simulation) as an input argument of an NS2 executable command ns. In most cases, a simulation trace file is created, and is used to plot graph and/or to create animation. Following are the steps for writing a script in ns-2:

- Create a new simulator object.
- Turn on tracing [Open your own trace files].
- Create network (physical layer).
- Create link and queue (data-link layer).
- Define routing protocol.
- Create transport connection (transport layer).
- Create traffic (application layer).
- Insert errors.

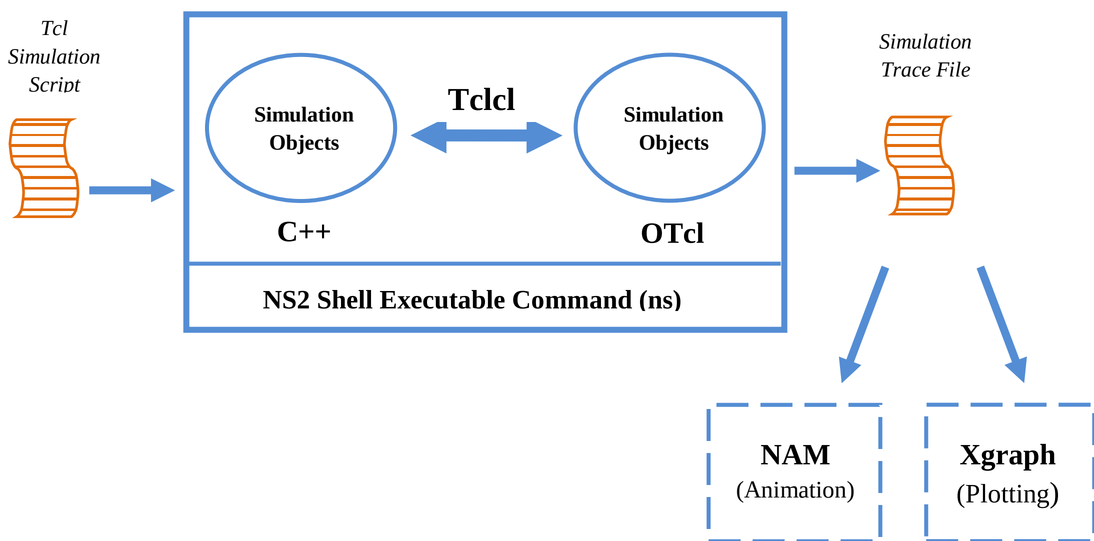

Figure 3.1: Basic architecture of NS.

NS2 consists of two key languages: C++ and Object-oriented Tool Command Language (OTcl). While the C++ defines the internal mechanism (i.e., a backend) of the simulation objects, the OTcl sets up simulation by assembling and configuring the objects and scheduling discrete events (i.e., a frontend). The C++ and the OTcl are linked together using Tclcl. Mapped to a C++ object, variables in the OTcl domains are sometimes referred to as handles. Conceptually, a handle (e.g., n as a Node handle) is just a string (e.g., _o10) in the OTcl domain, and does not contain any functionality. Instead, the functionality (e.g., receiving a packet) is defined in the mapped C++ object (e.g., of class connector). In the OTcl domain, a handle acts as a frontend which interacts with users and other OTcl objects. It may define its own procedures and variables to facilitate the interaction. The member procedures and variables in the OTcl domain are called instance procedures (instprocs) and instance variables (instvars), respectively. It is important to the readers are to learn about C++ and OTcl languages to get a better understanding of these architecture [16].

NS2 provides a large number of built-in C++ objects. It is advisable to use these C++ objects to set up a simulation using a Tcl simulation script. However, advance users may find these objects to be insufficient. They need to develop their own C++ objects, and use OTcl configuration interface to put together these objects. After simulation, NS2 outputs either text-based or animation-based simulation results. To interpret these results both graphically and interactively, tools such as NAM (Network AniMator) and XGraph are used. To analyze a particular behavior of the network, users can extract a relevant subset of text-based data and transform it to a more conceivable presentation [16].

3.5 Performance Metrics

In the evaluation of routing protocols different performance metrics are used. They show different characteristics of the whole network performance. In this performance comparison, we evaluate the Packet Delivery Ratio, Throughput, Network Lifetime, Energy Consumption, Packet Delivery Ratio and End to End Delay of selected protocols in order to study the effects on the whole network.

3.5.1 Packet Delivery Ratio

This represents the ratio between the number of data packets that are sent by the source and the number of data packets that are received by the sink [17]. The Average packet Delivery Ratio is calculated from the following Eq. (3.1)

$$\boldsymbol{PDR} = \frac{Ptk_r}{Ptk_q} \tag{3.1}$$

Where Ptk_r is Successfully Delivered Packets and Ptk_q is Required Packets

3.5.2 Average End-to-End Delay

It is the average delay between the sending of the data packet by the source and its receipt at the corresponding receiver including the acquisition, buffering and processing at intermediate nodes, and retransmission delays at the MAC layer, etc. if the value delays due to route of End-to-end delay is high then it means the protocol performance is not good due to the network congestion [17]. The Average Delay can be calculated form the following Eq. (3.2)

$$\boldsymbol{Avg.Delay} = \frac{\sum_{i=0}^{n} Tr_i - Ts_i}{Ptk_N} \tag{3.2}$$

Where Tr_i is Time of Received Packet Ts_i is Time of Send Packet and Ptk_N Total No. of Packets Received.

3.5.3 Throughput

The ratio of total data received by a receiver from a sender for a time the last packet received by receiver measures in bit/sec and byte/sec [17]. It can be expressed mathematically as the Eq. (3.3)

$$\boldsymbol{Throughput\ (bit/s)} = \frac{Ptk_r \times Ptk_{size} \times 8}{T} \tag{3.3}$$

Where Ptk_r is No. of Delivered Packets Ptk_{size} is Packet Size and T is Total Duration of Simulation.

3.5.4 Average Energy Consumption (Ea)

The average energy consumption is calculated across the entire topology. It measures the average difference between the initial level of energy and the final level of energy that is left in each node [17]. The Average energy consumption is calculated from the following Eq. (3.4)

$$\boldsymbol{E}_{\mathrm{a}} = \frac{\sum_{k=1}^{n}(E_{ik} - E_{fk})}{N} \tag{3.4}$$

Where E_{ik} is The Initial Energy Level of Node, E_{fk} is The Final Energy Level of Node and N Is Number of Nodes in Simulation.

This metric is an important because the energy level of network uses is proportional to the network's lifetime. The lower the energy consumption the longer is the network's lifespan.

3.5.5 Network life time

It is a time interval from starting operation of the sensor network until the death of the first alive node. Network lifetime is the time span from the deployment to the instant when the network is considered nonfunctional. It can be, for example, the instant when the first sensor dies, a percentage of sensors die, the network partitions, or the loss of coverage occurs [17]. The average Network life time of the Network is measured as the following Eq. (3.5)

$$\boldsymbol{Avg.\ lifetime} = \frac{E_i - E_w}{E_c - E_s} \tag{3.5}$$

Where E_i is Initial Energy, E_w is Expected Wasted Energy, E_c is Energy Consumption of Network and E_s is Energy Consumed by Sensor.

3.6 Evaluation Technique

The simulation software NS-2.34 has been used for performance assessment of ACO and LEACH based on various performance metrics. NS-

2.34 is an open source network simulator that is widely used for networking research. Performance evaluation of different routing protocol is done on NS2 which is installed on virtual machine (VM) under the Linux platform (Ubuntu 12.04). The simulation environment consists of an area of 100m x 100m, where randomly 20 to 100 mobile nodes are placed. A source and a destination are selected randomly. Data sources generate data according to CBR (Constant Bit Rate) and FTP (File Transfer Protocol) traffic pattern. Source and destination pairs are spread randomly over the network. By observing the performances of the network under mobility it can be test the stability of design. The simulation parameters are shown in Table 3-1.

Table 3-1 Simulations Parameters

Parameters	Value
Traffic Agent Type	FTB / CBR
Data Type	TCP / UDP
MAC	802.11
Channel	Wireless
Network Size	100 × 100
Routing Protocol	ACO, LEACH
Number of Nodes	20, 40, 60, 80, 100
Simulation Time	200 sec
Packet Size	512 bit
Initial Energy	500 J

In NS2, the steps for getting trace and NAM files after the simulation are as follows:

- Writing of the program in OTCL: OTCL is used to write the program for generate a network, network environment, and trajectory of mobile nodes.
- Run the **.tcl** file on the terminal under the Linux mint platform.
- NS2 trace analyzer is use to analyses trace file obtained during simulation and according to trace file generate the respective graphs.

3.7 Simulation Steps

We installed as first step Ubuntu 12.04 as an environment for NS-2.34, Ubuntu can be installed directly or in virtual machine (VM) like (VMWare Work Station).

3.7.1 AntHocNet Installation

To test the ACO algorithm NS-2.34 with anthocnet packets was built on Ubuntu. Used these commands in sequences on terminal:

```
>> tar xvf ns-allinone-2.34_gcc5.tar.gz
>> cd ns-allinone-2.34/
>> patch –p0 < anthocnet_ns234.patch
>> ./install
>> sudo make install
>> cp ns ns-ant
>> sudo cp ns-ant /usr/local/bin/
```

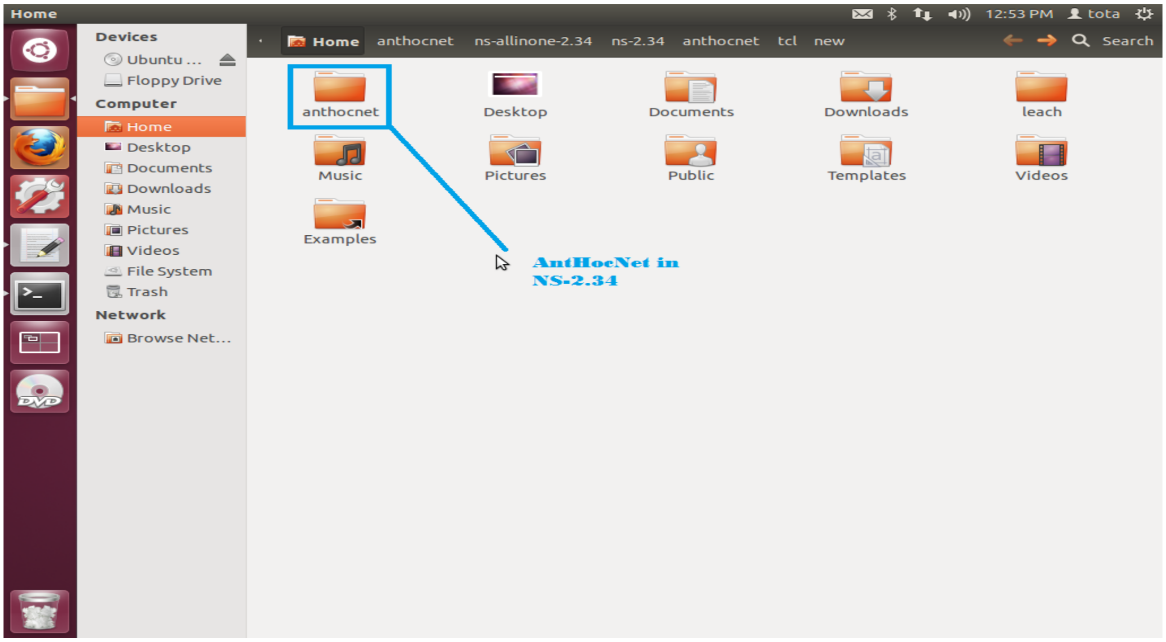

Figure 3.2: AntHocNet in NS-2.34

Running the Newant.tcl script on terminal after installing anthocnet.

```
>> ns-ant Newant.tcl anthocnet
```

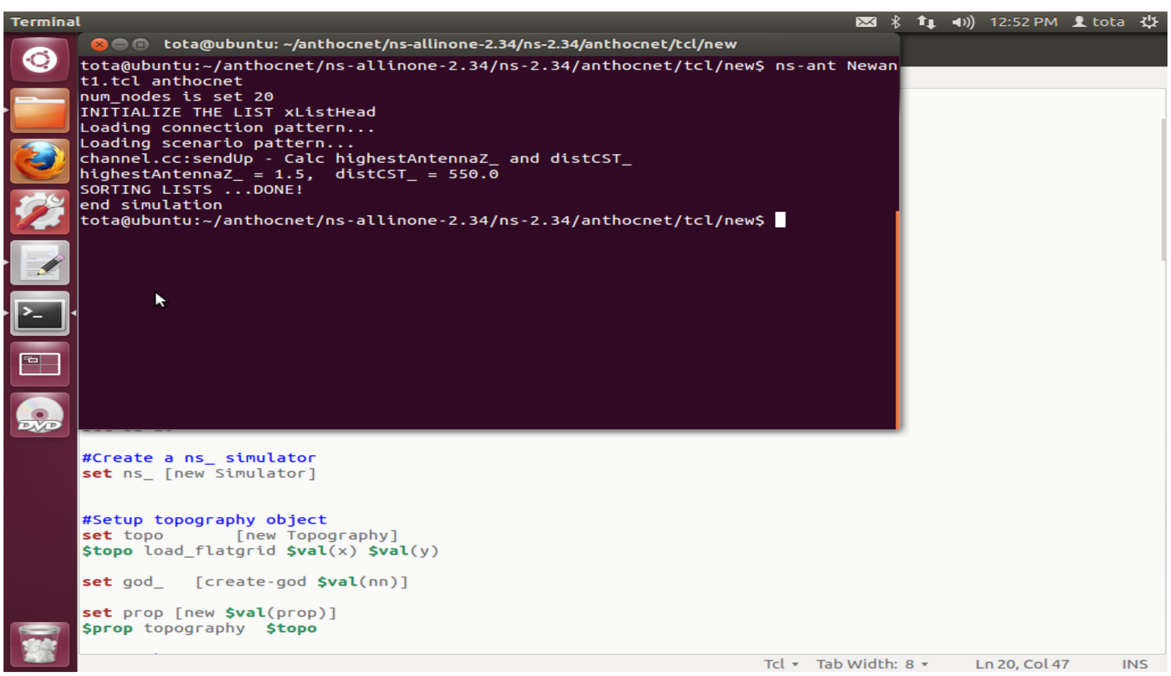

Figure 3.3: Newant1.tcl run in terminal

3.7.2 LEACH Installation

To test the LEACH algorithm NS-2.34 with LEACH packets was built on Ubuntu. Used these commands in sequences on terminal:

```
>> tar xvf ns-allinone-2.34.tar.gz
>> cd ns-allinone-2.34/
>> tar xvf leach+pegasis-ns234.tar.gz
>> patch -p0 < otcl_ns234_gcc-4.4.patch
>>export CC=gcc-4.4 CXX=g+++-4.4 && ./install
>> cd ns-234
>> sudo make install
```

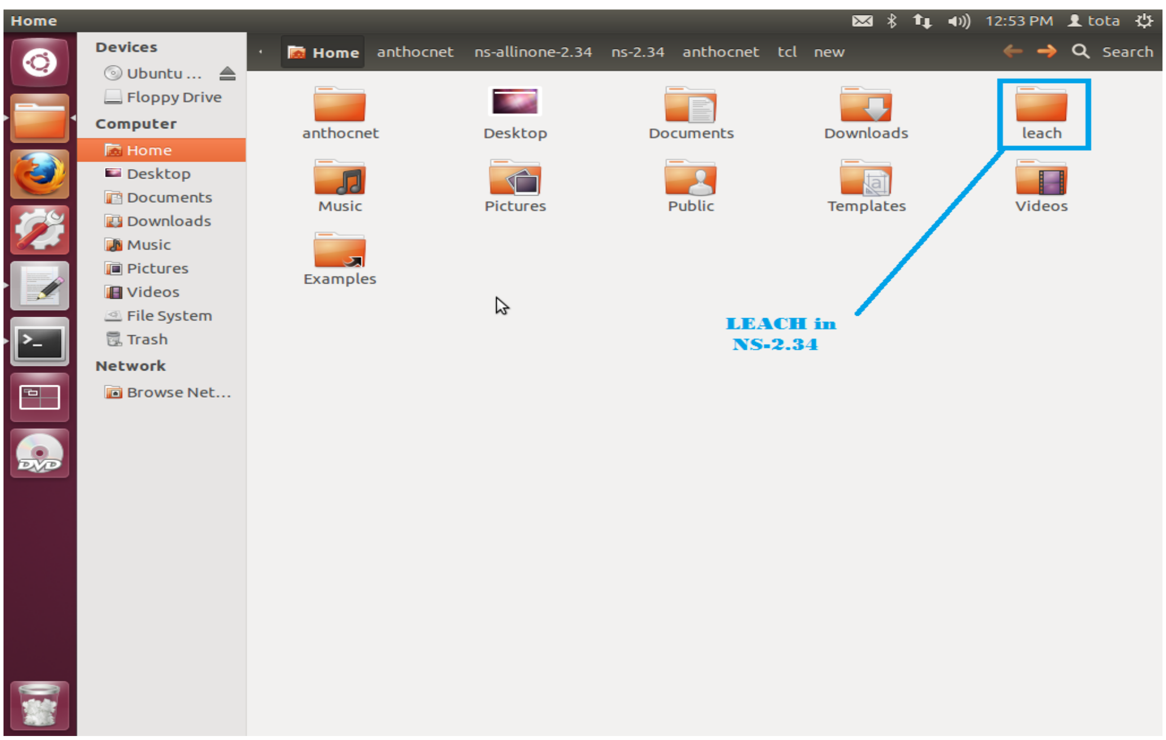

Figure 3.4: LEACH in NS-2.34

Running the ./test script on terminal after installing LEACH.

```
>> ./test
```

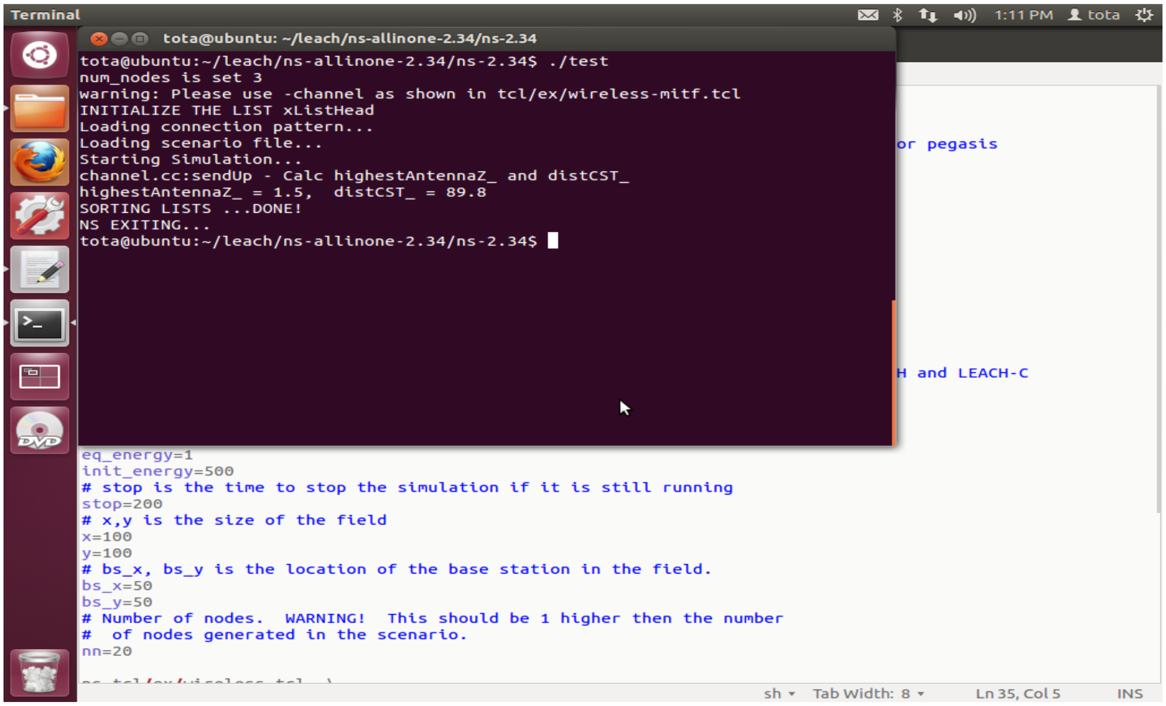

Figure 3.5: ./leach_test run in terminal

3.7.3 Running and Output

To calculate the performance metrics .awk script is used and run after simulation end the data was stored in traced files.

- In AntHocNet

Use .awk script to calculate performance metrics

```
>> awk -f avg_thoughput.awk antnet_trace.tr
```

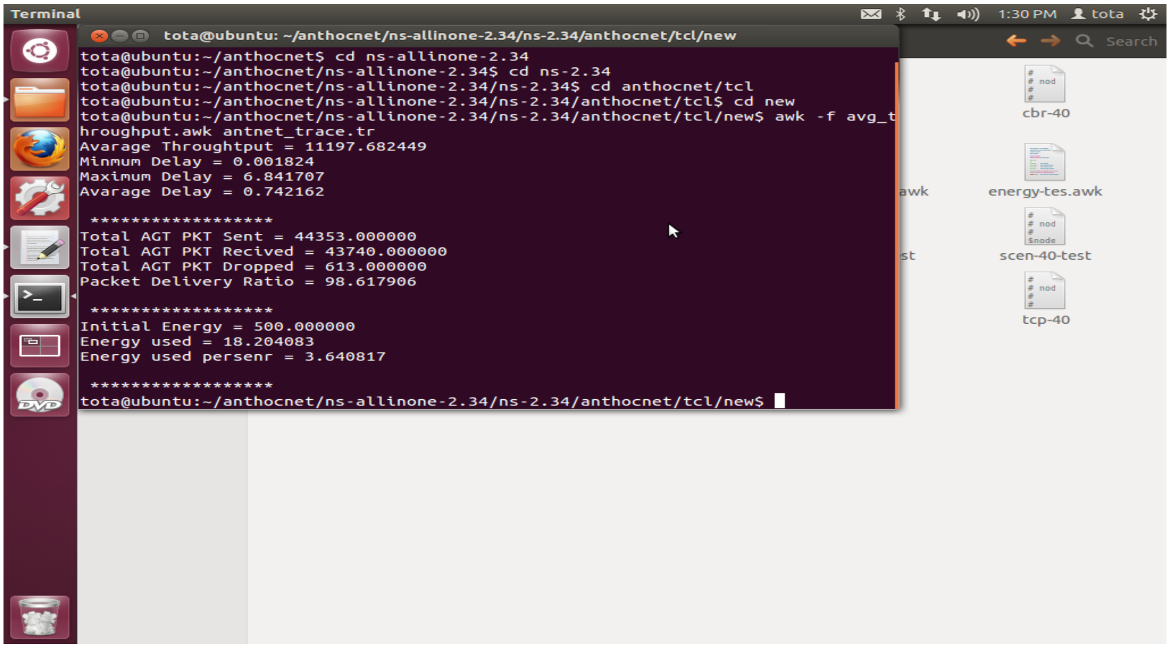

Figure 3.6: calculating metrics in Antnet

Use Network Animator NAM to simulate the node movement and behavior

```
>> nam antnet.nam
```

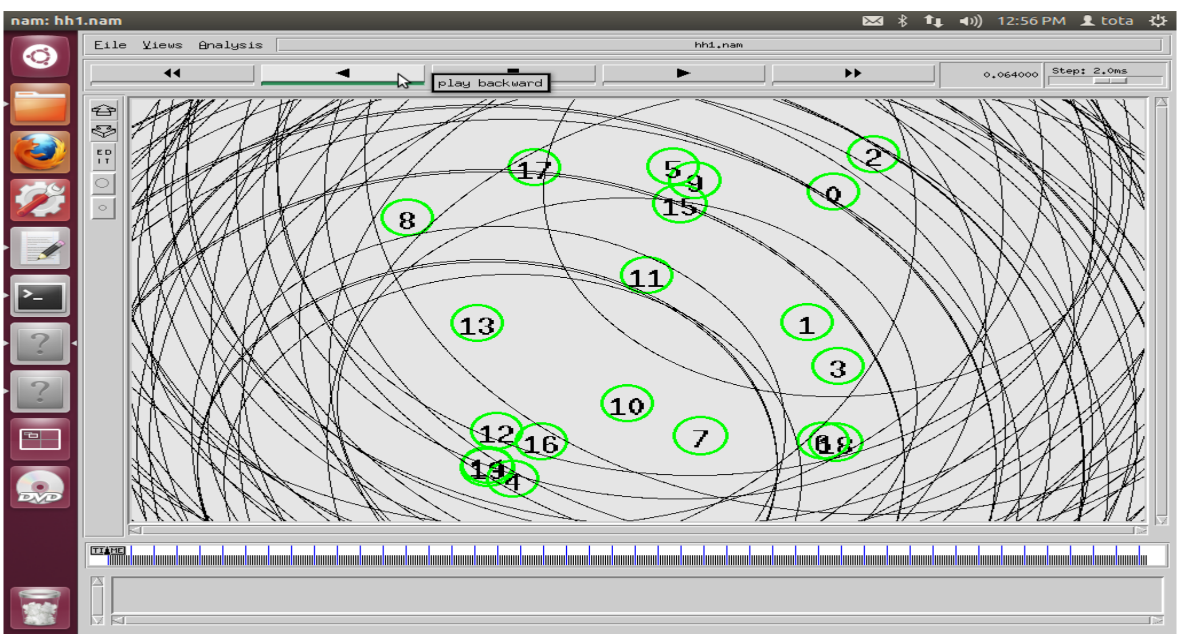

Figure 3.7: NAM for Newant1.tcl with 20 node

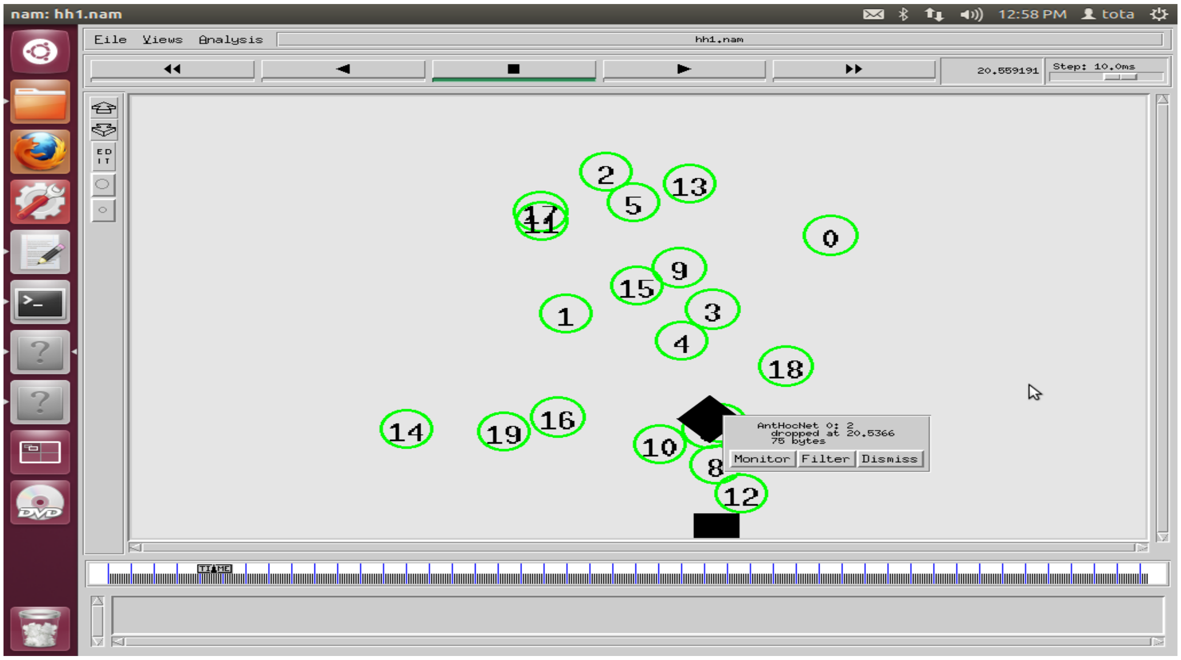

Figure 3.8: AntHocNet Agent

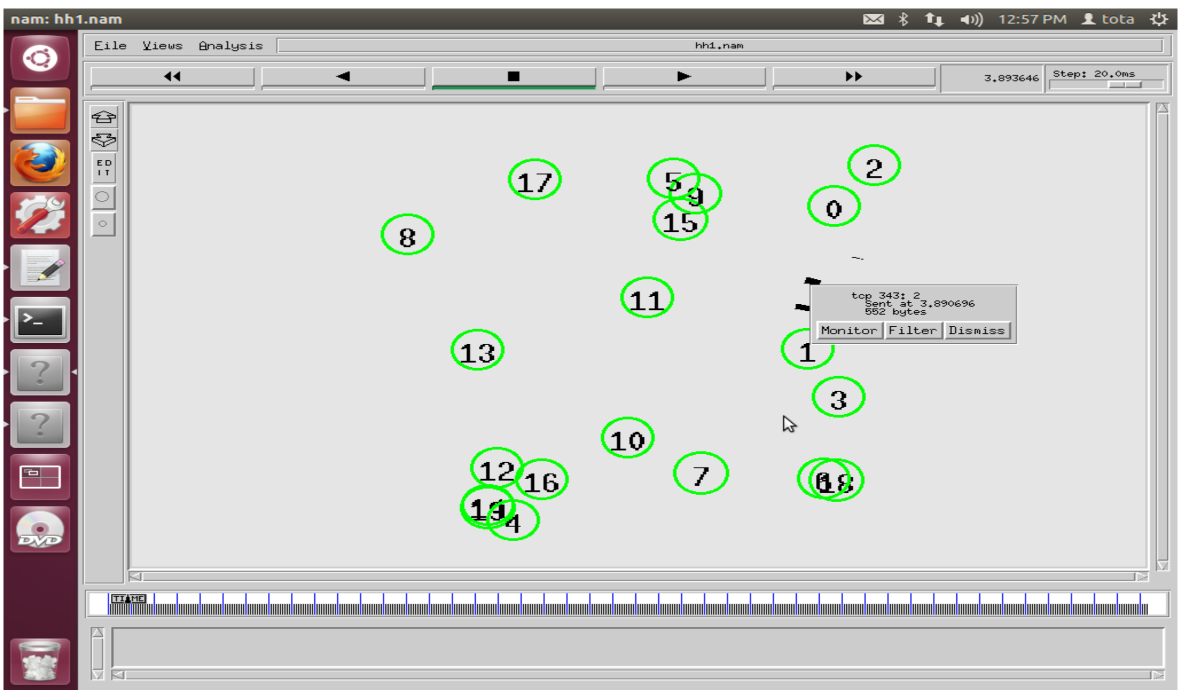

Figure 3.9: TCP packets send

- In LEACH

Use .awk script to calculate performance metrics

```
>> awk -f avg_thoughput.awk leach.tr
```

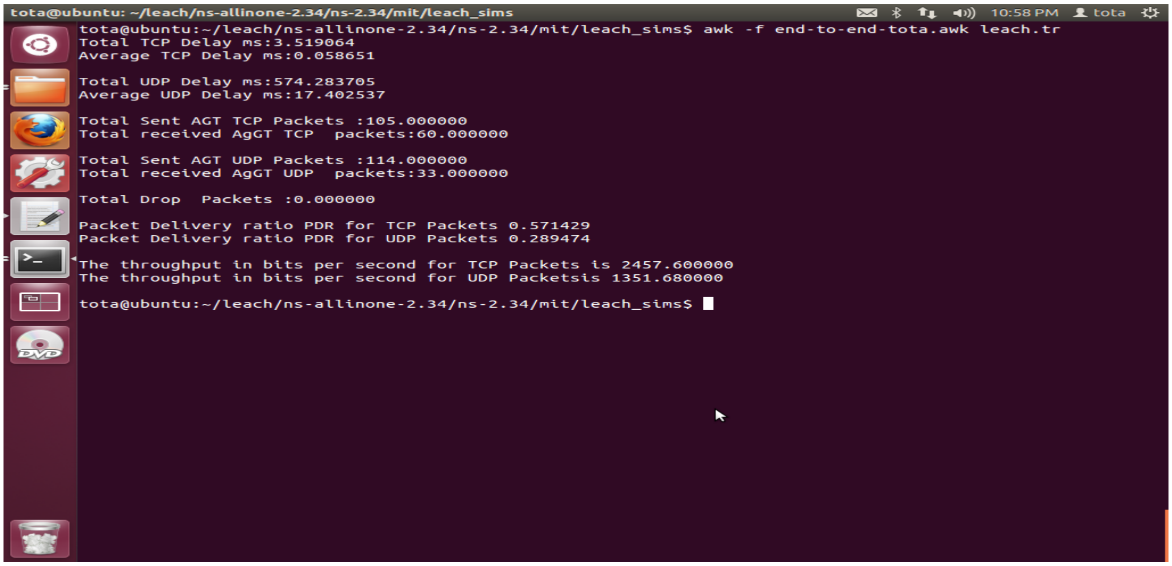

Figure 3.10: LEACH output

Use .awk script to calculate performance metrics

```
>> awk -f tes_energy.awk leach.energy
```

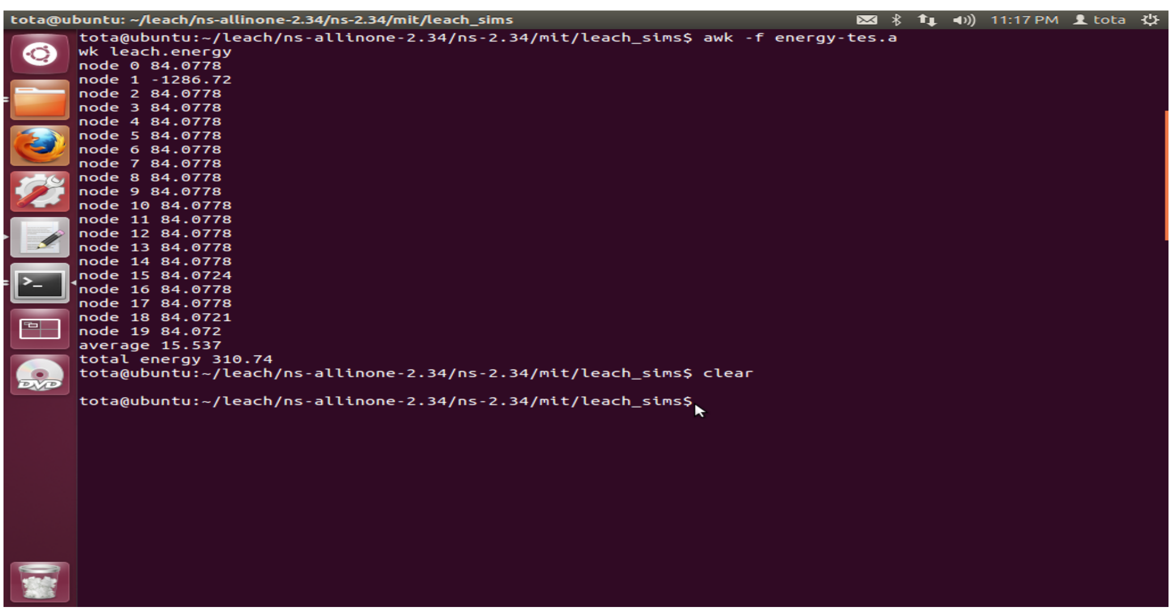

Figure 3.11: LEACH energy output

3.8 Summary

Performance evaluation of different routing protocol is done on NS2 by considering different scenarios. The metrics to measure and compare the performance of the protocols are throughput, end to end delay, average energy consumption, network life time and packet to delivery ratio.

CHAPTER FOUR

RESULT AND DISCUSSION

Chapter Four

Result and Discussion

4.1 Overview

The following tables show the observations taken for the various configurations, and their effect on the five performance metrics for TCP/FTP, and UDP/CBR separately for ACO and LEACH. The results are provided through graphs plotted as Performance metrics vs. numbers of nodes. Then they are compared between TCP/FTP and UDP/CBR for each protocol by using 80 nodes.

4.2 TCP/FTP Traffic

The Table 4.1 specifies the values of parameters used for TCP traffic.

Table 4.1 Observations for varying number of nodes for TCP traffic

No. of Nodes	Throughput		End 2 End Delay		PDR		Energy Cons.	
	ACO	**LEACH**	**ACO**	**LEACH**	**ACO**	**LEACH**	**ACO**	**LEACH**
20	11.20	4.46	0.71	0.30	98.6	57.1	3.64	4.50
40	6.07	3.28	0.92	0.50	90.8	39.2	4.21	5.10
60	3.83	2.82	0.95	0.50	85.6	39.5	4.80	8.64
80	2.30	2.28	1.12	0.70	77.2	20.0	5.55	9.62
100	1.63	1.40	1.20	0.90	76.1	19.9	6.13	10.20

4.2.1 Throughput for FTP Traffic

Figure 4.1 shows the response of throughput expressed in kb/s against the number of nodes for the two protocols taken from Table 4.1.

Throughout is directly related to the packet drops. Packet drops typically happens because of network congestion or for lack of route. Figure 4.1 depicts the variation in throughput by increasing number of nodes. On an average throughput decreases as network density increases due to congestion and collision in the networks. The throughput of ACO and LEACH are decreased with the number of node increased. At 20 nodes ACO has a good throughput by 60.17% of LEACH throughput. With the increasing of nodes both of two protocols has almost the same throughput with 2.30 Kb/s. But at all the throughput of ACO is better than LEACH protocol.

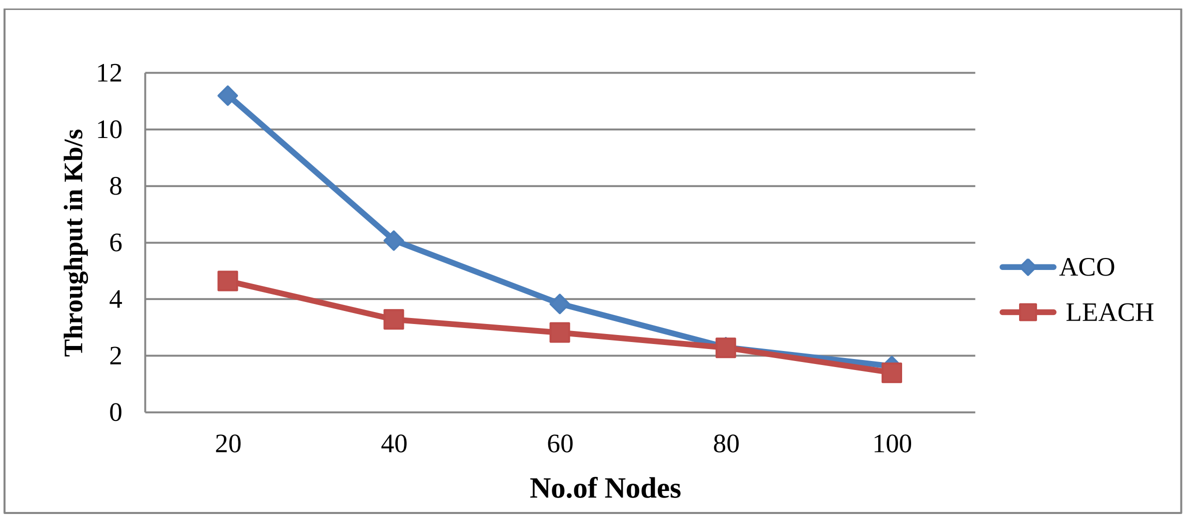

Figure 4.1: throughput vs. number of nodes for TCP traffic

4.2.2 End to End Delay for FTP Traffic

Based on the observations of Table 4.1, the response of end to end delay in second against varying number of nodes is shown in Figure 4.2.

Refer to Figure 4.2 LEACH has 45.50% low end to end delay compared to ACO in all the simulation scenarios, although the two protocols has increasing in end to end delay with the increasing of nodes. Due to in ACO protocol, routes to every destination were always calculated on demand and update with optimal path. The LEACH achieves low end-to-end delay due to its cluster head CH routing methodology.

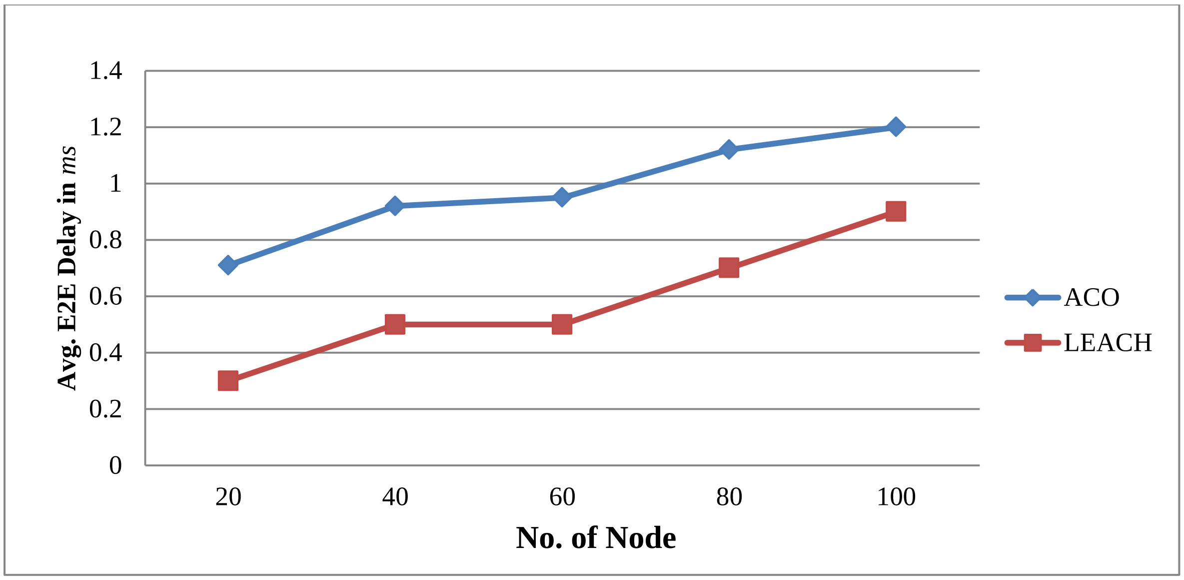

Figure 4.2: end to end delay vs. number of nodes for FTP traffic

4.2.3 Packet Delivery Ratio for FTP Traffic

Based on the observations of Table 4.1, the response of packet delivery ratio against varying number of nodes is shown in Figure 4.3.

Refer to Figure 4.3, PDR decreases with increasing number of nodes as congestion in network increases resulting in more dropped packets due to collisions. ACO has recorded good values with 45 ~ 47% than LEACH values. ACO has highest PDR in 20 nodes by 98.6%; Highest PDR value indicates the good performance.

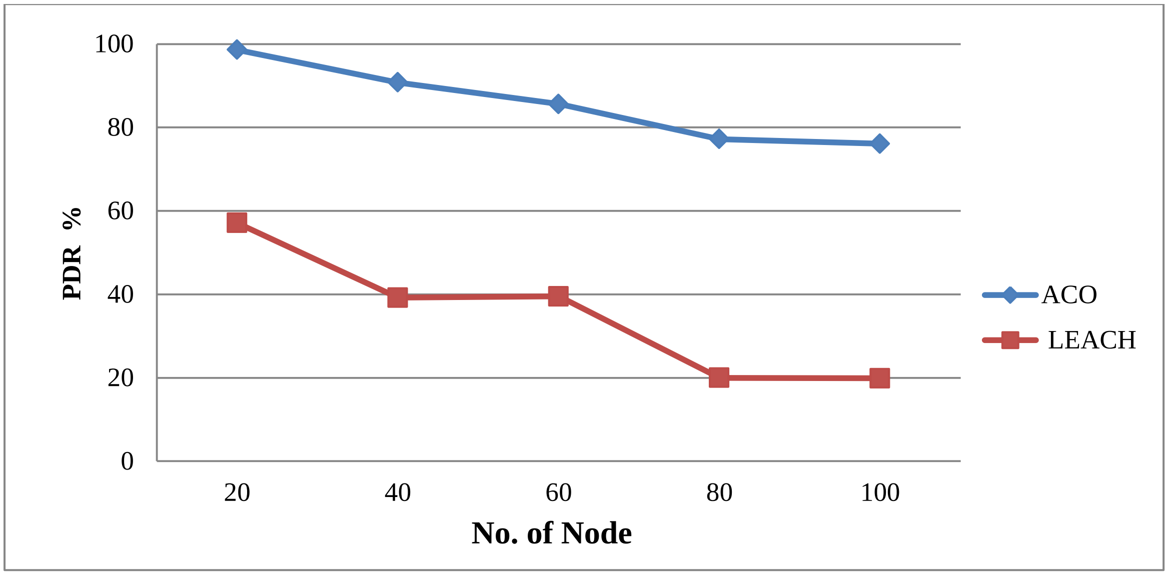

Figure 4.3: packet delivery ratio vs. number of nodes for FTP traffic 1

4.2.4 Energy Consumption for FTP Traffic

Based on the observations of Table 4.1, the response of average energy consumption against varying number of nodes is shown in Figure 4.4.

Refer to Figure 4.4, energy consumption increases with increasing number of nodes as congestion in network increases resulting in more dropped packets due to collisions. At 20 nodes both two protocols has low energy consumption and increased with increasing number of nodes. LEACH has the highest record with 10.20% at 100 nodes, while ACO recorded only 6.13% at 100 nodes. LEACH has high energy consumption due to cluster overhead in CH; lowest energy consumption value increase network lifetime and network performance.

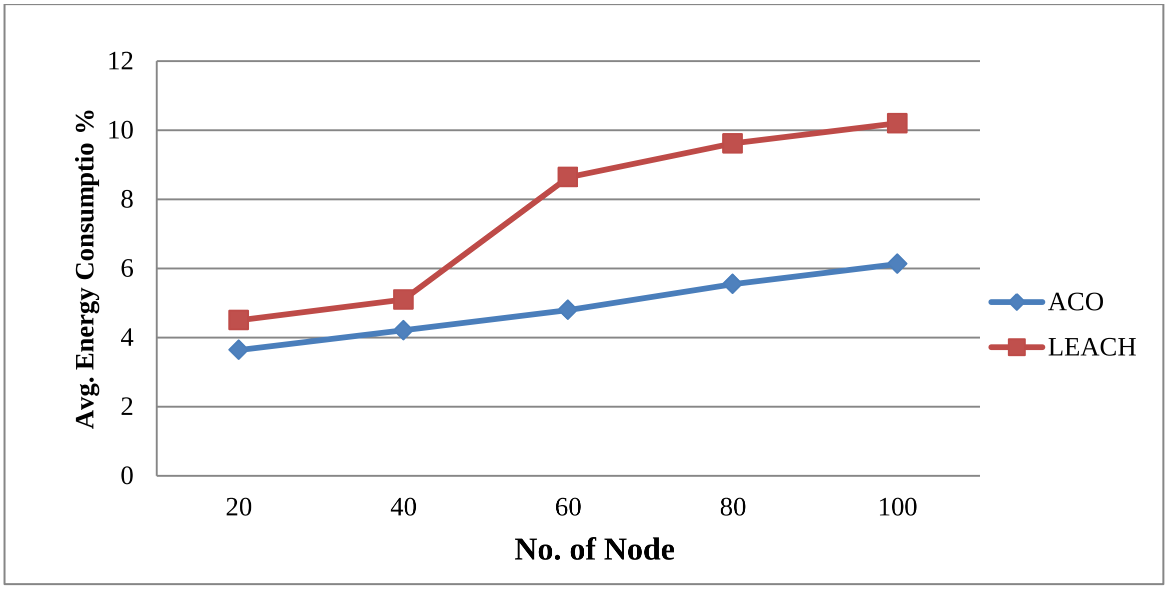

Figure 4.4: average energy consumption vs. number of nodes for FTP traffic

4.3 UDP/CBR Traffic

The following Table 4.2 specifies the values of parameters used for UDP traffic

.

Table 4.2 Observations for varying number of nodes of UDP traffic

No. of Nodes	Throughput		End 2 End Delay		PDR		Energy Cons.	
	ACO	LEACH	ACO	LEACH	ACO	LEACH	ACO	LEACH
20	0.56	0.48	0.005	0.11	90.00	27.00	37.76	32.65
40	0.57	0.43	0.006	0.19	89.96	38.50	36.57	47.26
60	0.56	0.30	0.006	0.11	90.50	44.00	35.66	65.16
80	0.54	0.22	0.007	0.34	92.20	51.20	34.85	42.41
100	0.55	0.19	0.007	0.36	88.00	54.30	34.21	44.49

4.3.1 Throughput for CBR Traffic

The following Figure 4.5 shows the response of throughput expressed in kb/s against number of nodes for the two protocols obtained by Table 4.2.

It can be seen that at Figure 4.5, ACO has more throughput as compared to LEACH. At 20 nodes both of the two protocols has throughput ~ 50 Kb/s. with the increasing of nodes, ACO has better throughput with 60% than LEACH. Throughput in case of LEACH decreases with increasing number of nodes because LEACH is hierarchical routing protocol with CH and Base station nodes require more control overhead to maintain the route to every other node. LEACH works efficiently under small scale networks. Since, it consumes less bandwidth owing to the less frequent broadcasting of update packets. Here ACO routing protocol showing best throughput with increasing number of node because in ACO routing protocol, routing table is established at every node, so there is no need to carry entire route information along with data packet that will decrease the control overhead.

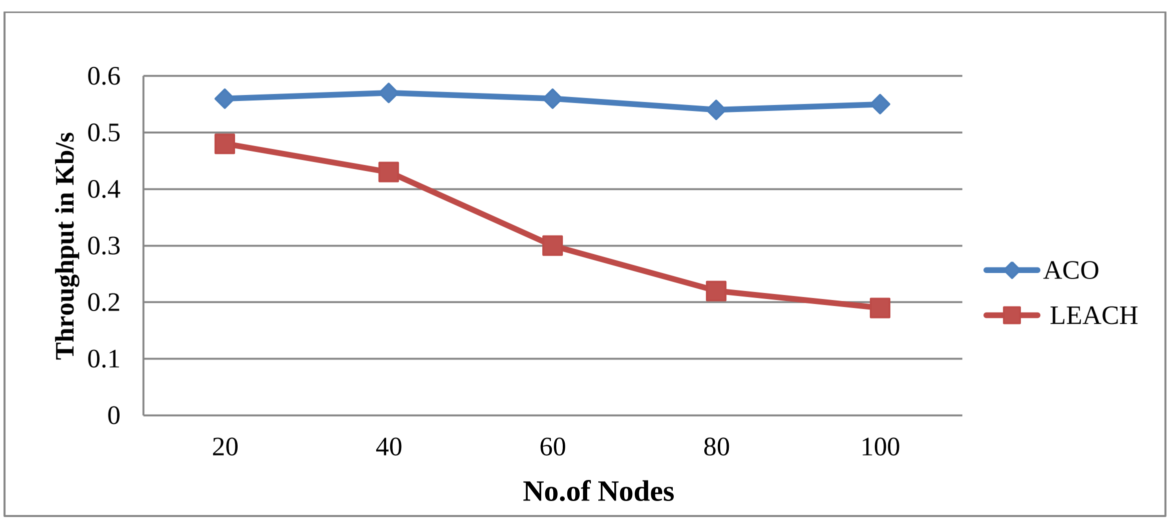

Figure 4.5: Throughput vs. number of nodes for CBR traffic

4.3.2 End to End Delay CBR Traffic

Based on the observations of Table 4.2, the response of end to end delay in second against varying number of nodes is shown in Figure 4.6.

Refer to Figures 4.6; ACO has low end to end delay with 0.005 ms at 20 nodes compared to LEACH with 0.11 ms at 20 nodes in the simulation scenario because updated route to the intended node is always available whenever any node wishes to send the data to any other node when number of nodes increases, LEACH takes more time to deliver the packets to the destination. So, the delay of LEACH increases when increasing number of nodes. For the large network, the route discovery process consumes more time to find the short hop count path to the destination. It causes the link failure often and it leads to the repeated route recovery process therefore it introduces a large delay in the network. The increased mobility causes more routing packet generation to find the fresh route. If the valid route is known under the route discovery process, data packets are forwarded to the destination; otherwise, data packets are buffered until the route is discovered, which makes delay in the data transmission.

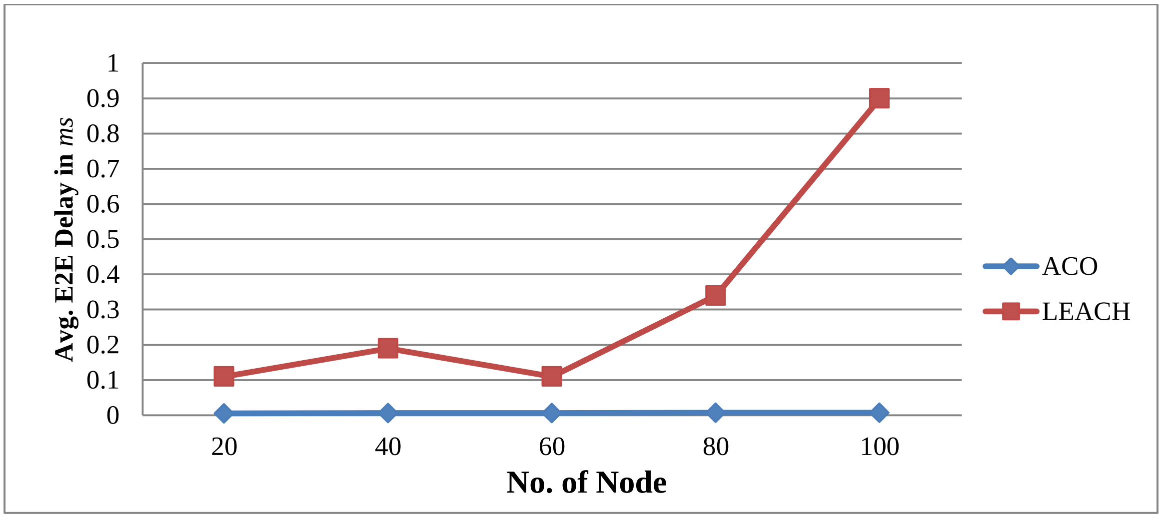

Figure 4.6: End to end delay vs. number of nodes for CBR traffic

4.3.3 Packet Delivery Ratio CBR Traffic

Based on the observations of Table 4.2, the response of packet delivery ratio against varying number of nodes is shown in Figure 4.7.

The packet delivery ratio of ACO decreased in stable level and LEACH protocol increased, when the number of node increases. This is because; it is difficult to maintain the routing information under a large scale network. ACO has highest PDR with 90% at 20 nodes which is better with 60% than LEACH PDR at 20 nodes, because ACO develop multipath in one route discovery process that means the chance of dropping decreases.

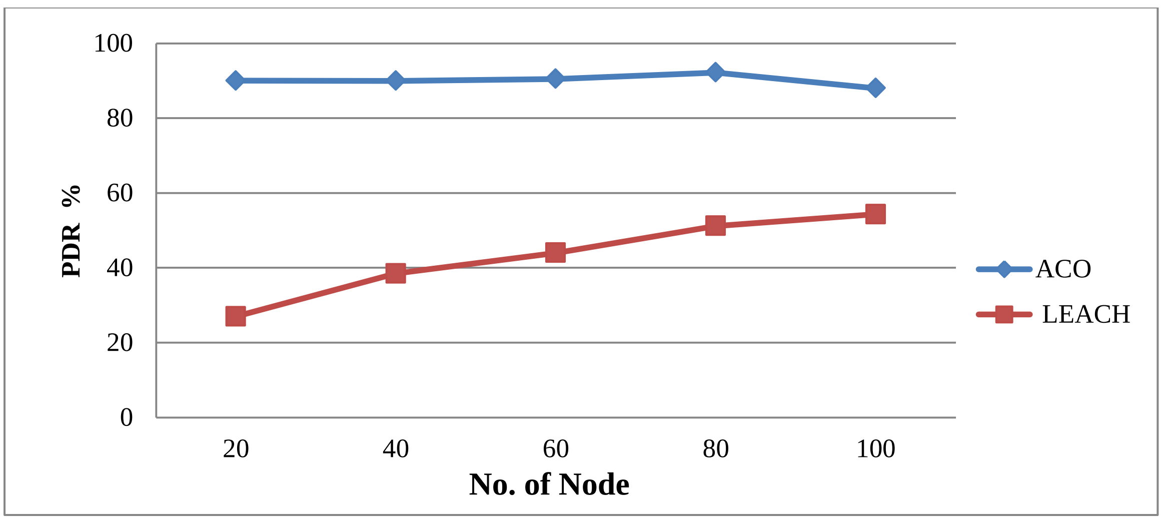

Figure 4.7: Packet delivery ratio vs. number of nodes for CBR traffic

4.3.4 Energy Consumption for CBR Traffic

Based on the observations of Table 4.2, the response of average energy consumption against varying number of nodes is shown in Figure 4.8.

Refer to Figure 4.8, energy consumption decreased with increasing number of nodes. LEACH has high energy consumption due to cluster overhead in CH. ACO had lowest energy consumption value with 34.21% at 100 nodes which is better with 23% of LEACH energy consumption at 100 nodes; which means increasing in network lifetime and network performance.

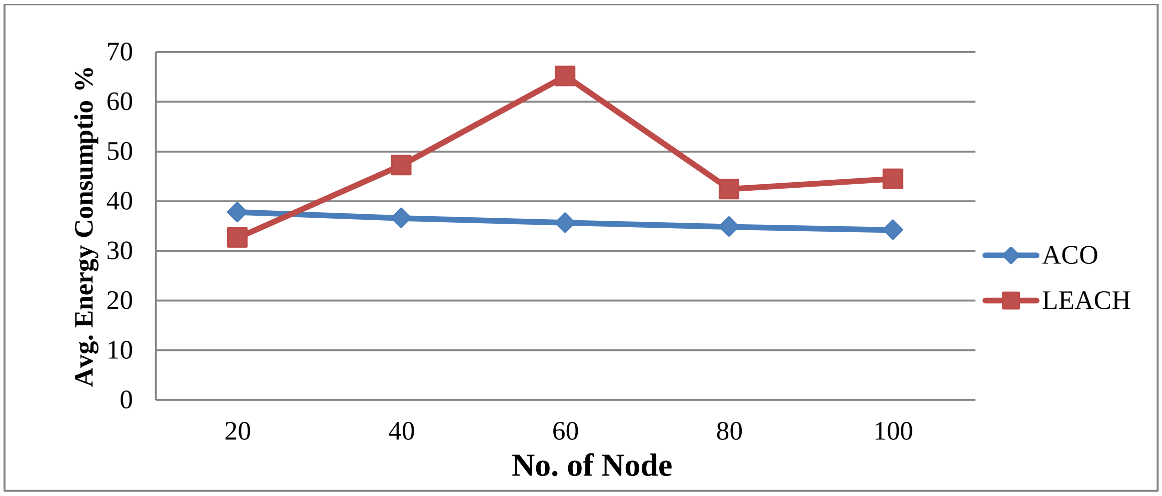

Figure 4.8: average energy consumption vs. number of nodes for CBR traffic

4.4 Network Lifetime for ACO and LEACH

Based on simulation, the response of network lifetime against varying number of nodes is shown in Figure 4.9.

Refer to Figure 4.9; number of alive nodes decreased with increasing in rounds (time) which is means increasing in network lifetime. Simulation start with 100 node and wait till first node drain (dead), here ACO show better lifetime than LEACH protocol with 88% at round 2000 sec.

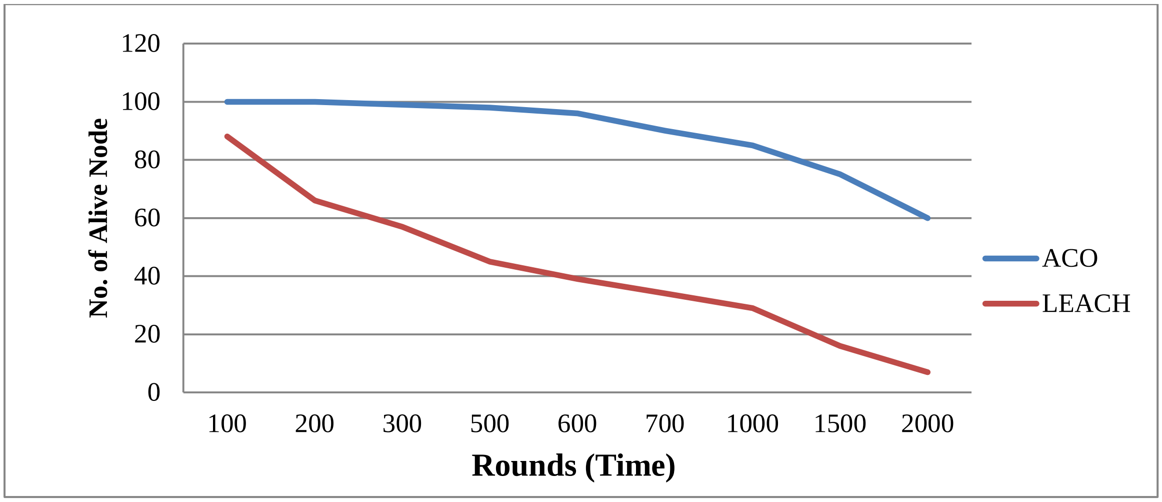

Figure 4.9: Network lifetime vs. number of nodes

4.5 TCP/FTP and UDP/CBR Traffic

The following Table 4.3 specifies the parameters used for TCP and UDP traffics by using 80 nodes for ACO and LEACH

Table 4.3 UDP and TCP traffic at 80 nodes

No. of Nodes	Throughput		End 2 End Delay		PDR		Energy Cons.	
	ACO	**LEACH**	**ACO**	**LEACH**	**ACO**	**LEACH**	**ACO**	**LEACH**
TCP	2.30	2.28	1.12	0.70	77.20	20.00	5.50	9.62
UDP	0.54	0.22	0.007	0.34	92.20	51.20	34.85	42.41

4.5.1 Throughput

From Figure 4.10, Out of the two traffic types i.e. TCP/FTP and UDP/CBR, the TCP provides far better performance than the UDP. ACO has better throughput with 0.8% under TCP traffic and with 60% under UDP than

LEACH. This proves that the network working with ACO and LEACH provide better efficiency with TCP/FTP than UDP/CBR.

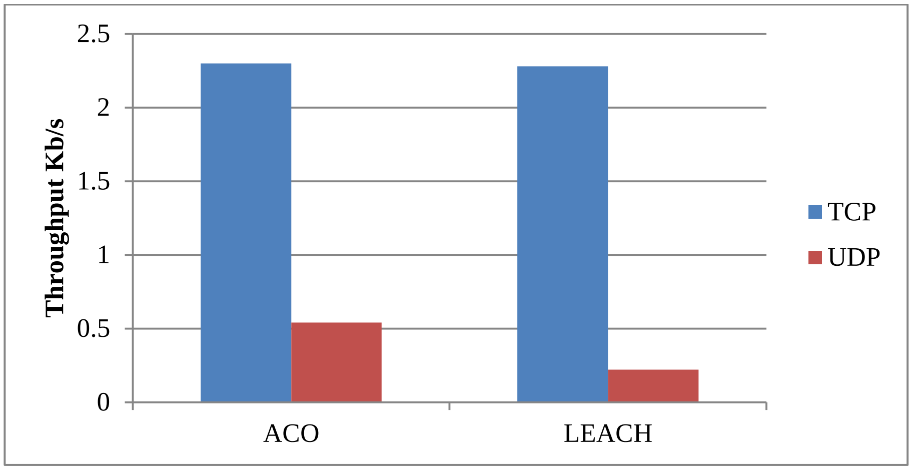

Figure 4.10: Throughput with using TCP and UDP

4.5.2 End to End Delay

The UDP/CBR offers lesser, end to end delay, than TCP/FTP, but as an exception in ACO the end to end delay with UDP traffic is almost the lowest with 0.007 ms.

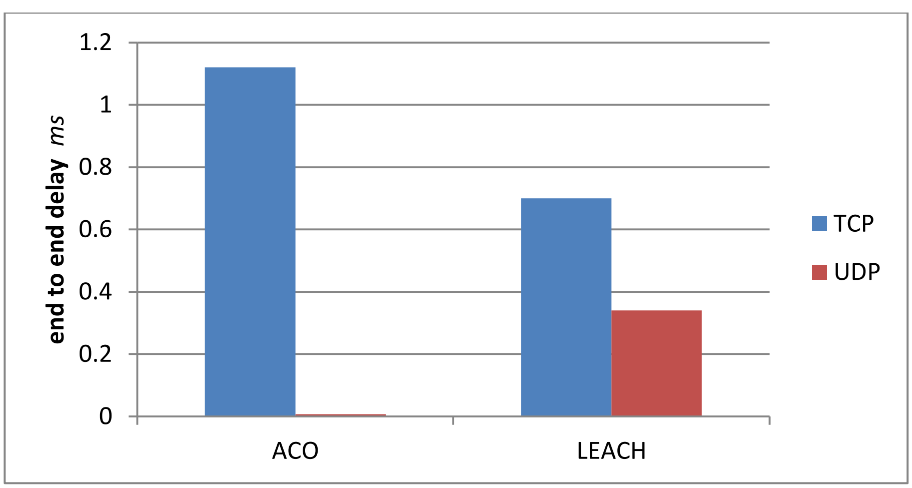

Figure 4.11: End to End delay with using TCP and UDP

4.5.3 Packet Delivery Ratio

With UDP traffic all protocols had high PDR as compared to TCP traffic. Therefore, UDP/CBR is more reliable than TCP/FTP. Due to the UDP protocol is used there is a "guaranteed delivery”. PDR had value of 92.20% in ACO under UDP traffic which is better by 44.5% than LEACH with 51.20% under UDP traffic.

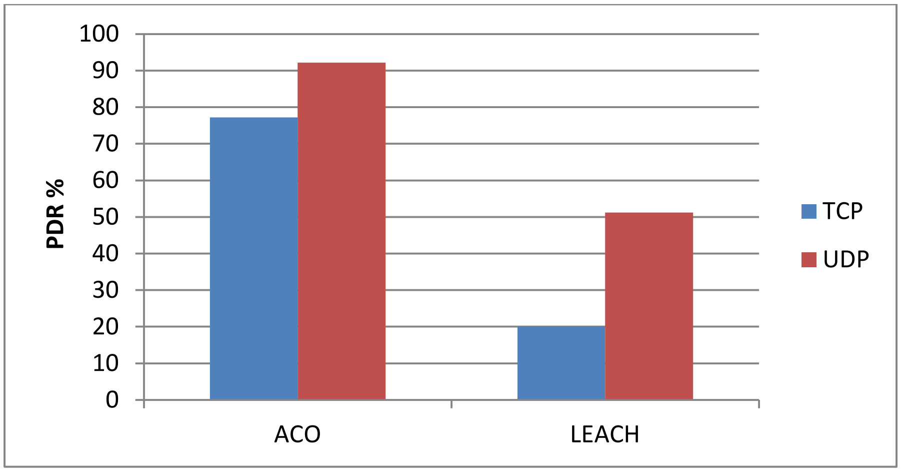

Figure 4.12: Packet delivery ratio with using TCP and UDP

4.5.4 Energy Consumption

With UDP traffic all protocols had high energy consumption as compared to TCP traffic. Therefore, TCP/FTP is more efficient energy consumption than UDP/CBR**.** ACO recorded lowest energy consumption with 5.50% under TCP traffic and 9.62% in LEACH under TCP traffic

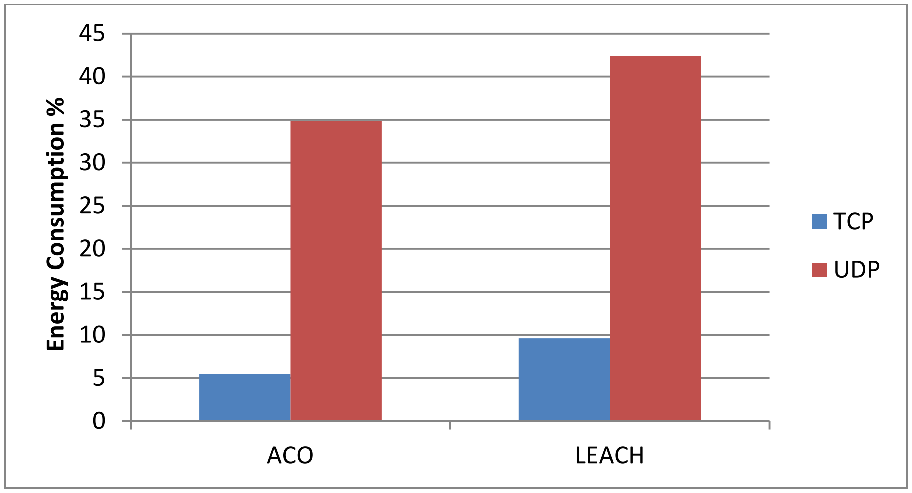

Figure 4.13: Energy Consumption with using TCP and UDP

4.6 Summary

From the previous results is clear that ACO had a higher performance in throughput and energy consumption with TCP traffic. With UDP traffic the ACO has high performance in end to end delay and packet delivery ratio. While LEACH had a good end to end delay, although ACO performance is better than the LEACH protocol.

CHAPTER FIVE

CONCLUSION AND FUTURE WORK

Chapter Five
Conclusion and Future Work

5.1 Conclusion

The various conclusions drawn from various experiments, observations, and analysis done in the thesis are as follows: Throughput: for TCP traffic, the network working with ACO provides better efficiency by 0.8% than LEACH. For UDP traffic, ACO has more throughputs as compared to LEACH by 60%. Average End to End Delay: With TCP and UDP traffics, the end to end delay was increasing. The ACO offers lesser end to end delay in UDP traffic with 0.007 ms, and LEACH offer lesser in TCP traffic with 37.5% than ACO. Packet Delivery Ratio (PDR): Although the PDR of ACO has greater values than LEACH, It is around 92.20% when using UDP traffic with 44.5% better than in LEACH. In TCP traffic, ACO has high PDR by 74.1% of LEACH. Average Energy Consumption: As general Both ACO and LEACH consume little energy in TCP traffic than UDP traffic.ACO offered lower energy consumption with 42.8% lesser than in LEACH in TCP traffic, while in UDP ACO recorded lesser with 18% than LEACH. Network lifetime: for both of two protocols the number of alive node in network decrease with increasing in number of rounds (time). Although ACO had better network lifetime than LEACH, ACO show better lifetime than LEACH protocol with 88% at round 2000 sec. It can also be concluded from the simulation results that the efficiency of ACO better than LEACH. Generally the ACO and LEACH offer good performance of throughput and energy consumption in TCP traffic, and good performance of end to end delay and PDR in UDP traffic.

5.2 Future Work

A future study could be conducted on comparison the performance of these two protocols when the traffic generator is other than FTP and CBR like TELNET and HTTP because these traffic generators are the representatives of the traffic in the real scenario and expanding the study towards hybrid routing protocols, considering more metrics and more complex scenarios.

REFERENCES

1. Khoshkangini, R., Zaboli, S. and Conti, M., “Efficient routing protocol via ant colony optimization (ACO) and breadth first search (BFS)”, IEEE International Conference on Internet of Things (iThings), and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) (pp. 374-380), 2014.
2. Gómez Blas, N., Mingo López, L.F.D., Aslanyan, L. and Ryazanov, V., “Self-organizing Routing Algorithm fo Wireless Sensors Networks (WSN) using Ant Colony Optimization (ACO) with Tinyos”, International Journal on Information Technologies and Knowledge, pp.142-151, 2011.
3. Sharma, T., Singh, H. and Sharma, A., “A comparative review on routing protocols in wireless sensor networks”, International Journal of Computer Applications, 2015.
4. Anwar, A. and Sridharan, D., “A survey on routing protocols for wireless sensor networks in various environments”, International Journal of Computer Applications, 112(5), 2015.
5. Sethi, D. and Bhattacharya, P.P., “A Comparative Analysis of Various Routing Protocols and Performance Comparison of Clustered Routing Protocols in Mobile Sink Scenario”, Majlesi Journal of Electrical Engineering, 12(3), 2018.
6. Raghuvanshi, A., Kumar, A. and Yadav, G.K., “Analysis of Routing Protocols for Mobile Sink in Wireless Sensor Networks: A Survey”, 2015.

7. Keerthika, A. and Hency, V.B., “A survey of routing protocols of wireless sensor network with mobile sinks”, ARPN Journal of Engineering and Applied Sciences, 2016.
8. Meera, G. and Sekhar, P., “A Survey on routing protocols for mobile sink based WSN”, International Research Journal of Engineering and Technology, 3(01), pp.895-901, 2016.
9. Srividhya, R. and Kathiresan, I.D.V., “Ant Colony Optimization Algorithm and its Applications”, Journal of Network Communications and Emerging Technologies (JNCET) www. jncet. org, 7(9), 2017.
10. Sharma, T., Singh, H. and Sharma, A., “A comparative review on routing protocols in wireless sensor networks”, International Journal of Computer Applications, 123(14), 2015.
11. García Villalba, L., Sandoval Orozco, A., Trivino Cabrera, A. and Barenco Abbas, C., “Routing protocols in wireless sensor networks”, sensors, 9(11), pp.8399-8421, 2009.
12. Brahm P., Shaveta R. and Paramjeet S., “Swarm Intelligence Techniques to improve lifetime of Wireless Sensor Networks”, International Journal of Computer Sciences and Engineering (ICSE), Vol.6,(9), 2018.
13. Kim, J.Y., Sharma, T., Kumar, B., Tomar, G.S., Berry, K. and Lee, W.H., “Intercluster ant colony optimization algorithm for wireless sensor network in dense environment”, International Journal of distributed sensor networks, 10(4), p.457402, 2014.
14. Usop, N.S.M., Abdullah, A. and Abidin, A.F.A., “Performance evaluation of AODV, DSDV & DSR routing protocol in grid environment”, IJCSNS International Journal of Computer Science and Network Security, 9(7), pp.261-268, 2009.

15. Gupta, S.G., Ghonge, M.M., Thakare, P.D. and Jawandhiya, P.M., "Open-source network simulation tools: An overview", International Journal of Advanced Research in Computer Engineering & Technology (IJARCET), 2(4), pp.pp-1629, 2013.
16. Chapter 6: simulation and performance metrics, last visit Dec. 2018, *www.verypdf.com*.
17. Anitha, R.U. and Kamalakkannan, P., "EEDBC-M: enhancement of leach-mobile protocol with energy efficient density-based clustering for mobile sensor networks (MSNs)", International Journal of Computer Applications, 74(14), 2013.

Appendix A

ACO code in NS-2.34

- ## Antnet Codes:

```
>>ns-ant Newant1.tcl anthocnet
```

```
#====================================
#     Simulation parameters setup
#====================================
    set val(chan)   Channel/WirelessChannel
    set val(prop)   Propagation/TwoRayGround
    set val(netif)  Phy/WirelessPhy
    set val(mac)    Mac/802_11
    set val(ifq)    Queue/DropTail/PriQueue
    set val(ll)     LL
    set val(ant)    Antenna/OmniAntenna
    set val(ifqlen) 50
    set val(nn)     20
    set val(rp)     AntHocNet
    set val(x)      100
    set val(y)      100
    set val(stop)   2500.0
    set val(cp)     "./tcp-20"
    #set val(cp)    ""
    #set val(sc)    ""
    set val(sc)     "./scen-20-test"
    set val(energymodel)   EnergyModel
    set val(initialenergy) 500
    set val(seed)          1.0


#====================================
#        Initialization
#====================================
    #number of nodes
       set sz 20

    #Create a ns_ simulator
       set ns_ [new Simulator]


    #Setup topography object
       set topo       [new Topography]
       $topo load_flatgrid $val(x) $val(y)

       set god_   [create-god $val(nn)]

       set prop [new $val(prop)]
```

```
   $prop topography  $topo

#Open the Trace file
   set tracefd [open antnet_trace.tr w]
   $ns_ trace-all $tracefd

#Open the NAM trace file
   set namfile [open hh1.nam w]
   $ns_ namtrace-all-wireless $namfile $val(x) $val(y)

   set chan_  [new $val(chan)];#Create wireless channel


#====================================
#     Mobile node parameter setup
#====================================
$ns_ node-config
      -adhocRouting    $val(rp) \
      -llType          $val(ll) \
      -macType         $val(mac) \
      -ifqType         $val(ifq) \
      -ifqLen          $val(ifqlen) \
      -antType         $val(ant) \
      -propType        $val(prop) \
      -phyType         $val(netif) \
      -channel         $chan_ \
      -topoInstance    $topo \
      -agentTrace      ON \
      -routerTrace     ON \
      -macTrace        OFF \
      -energyModel     $val(energymodel) \
      -idlePower       1.0 \
      -rxPower         0.01 \
      -txPower         0.01 \
      -sleepPower      0.000001 \
      -transitionPower 0.2 \
      -transitionTime  0.005 \
      -initialEnergy   $val(initialenergy) \
      -movementTrace   ON


#====================================
#        Nodes Definition
#====================================

#Create Antnet agents

for {set i 0} {$i < $sz} { incr i} {
   set node_($i) [$ns_ node]
   $god_ new_node $node_($i)
}
```

```
for {set i 0} {$i < $sz} {incr i} {
    set   ant_($i)  [new Agent/AntHocNet $i]
}

# Source the Connection and Movement scripts

if { $val(cp) == "" } {
    puts "*** NOTE: no connection pattern specified."

} else {
    puts "Loading connection pattern..."
    source $val(cp)
}

# Source the Scenario scripts

if { $val(sc) == "" } {
    puts "*** NOTE: no scenario pattern specified."
    set $val(sc) "none"
} else {
    puts "Loading scenario pattern..."
    source $val(sc)
}

#==================================

# Define node initial position in nam
for {set i 0} {$i < $val(nn)} { incr i } {
# 30 defines the node size for nam
    $ns_ initial_node_pos $node_($i) 10
}

# Telling nodes when the simulation ends
for {set i 0} {$i < $val(nn) } { incr i } {
    $ns_ at $val(stop) "$node_($i) reset";
}

# ending nam and the simulation
    $ns_ at $val(stop) "$ns_ nam-end-wireless $val(stop)"
    $ns_ at $val(stop) "stop"
    $ns_ at 199.91 "puts \"end simulation\" ; $ns_ halt"
proc stop {} {
    global ns_ tracefd namfile
    $ns_ flush-trace
    close $tracefd
    close $namfile
}

    $ns_ run
```

Appendix B

LEACH code in NS-2.34

➢ LEACH Codes:

```
>>./leach_test
```

```
# Copyright (c) 1997 Regents of the University of California.
# All rights reserved.
#
# Redistribution and use in source and binary forms, with or without
# modification, are permitted provided that the following conditions
# are met:
# 1. Redistributions of source code must retain the above copyright
#       notice, this list of conditions and the following disclaimer.
# 2. Redistributions in binary form must reproduce the above copyright
#       notice, this list of conditions and the following disclaimer in the
#     documentation and/or other materials provided with the distribution.
# 3. All advertising materials mentioning features or use of this software
#    must display the following acknowledgement:
#          This product includes software developed by the Computer Systems
#      Engineering Group at Lawrence Berkeley Laboratory.
# 4. Neither the name of the University nor of the Laboratory may be used
#    to endorse or promote products derived from this software without
#    specific prior written permission.
#
# THIS SOFTWARE IS PROVIDED BY THE REGENTS AND CONTRIBUTORS ``AS IS'' AND
# ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
# IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
# ARE DISCLAIMED.  IN NO EVENT SHALL THE REGENTS OR CONTRIBUTORS BE LIABLE
# FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL
# DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS
```

```
# OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION)
# HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT
# LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY
# OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF
# SUCH DAMAGE.
#
# $Header: /usr/src/mash/repository/vint/ns-2/tcl/ex/wireless.tcl,v 1.2 1999/02/24 23:27:34 haldar Exp $
#
# Ported from CMU/Monarch's code, nov'98 -Padma.

# ======================================================================
# Default Script Options
# ======================================================================
    set opt(chan)       Channel/WirelessChannel
    set opt(prop)       Propagation/TwoRayGround
    #set opt(netif)     NetIf/SharedMedia
    set opt(netif)      Phy/WirelessPhy
    #set opt(mac)       Mac/802_11
    set opt(mac)        Mac/802_11
    set opt(ifq)        Queue/DropTail/PriQueue
    set opt(ll)         LL
    set opt(ant)        Antenna/OmniAntenna

    set opt(x)          0     ;# X dimension of the topography
    set opt(y)          0     ;# Y dimension of the topography
    set opt(cp)         "/home/tota/leach/ns-allinone-2.34/ns-2.34/tcl/mobility/scene/cbr-20" ;# connection pattern file
    #set opt(cp)        ""
    set opt(sc)         "/home/tota/leach/ns-allinone-2.34/ns-2.34/tcl/mobility/scene/scen-20-test" ;# scenario file

    set opt(ifqlen)     50         ;# max packet in ifq
    set opt(nn)         20         ;# number of nodes
    set opt(seed)       0.0        ;# nodes speed
    set opt(stop)       200.0      ;# simulation time
    set opt(tr)         out.tr     ;# trace file
    set opt(rp)         dsdv       ;# routing protocol script
    set opt(lm)         "on"       ;# log movement

#======================================================================

    set AgentTrace          ON
    set RouterTrace         ON
    set MacTrace            OFF

    LL set mindelay_        50us
    LL set delay_           25us
```

```
    LL set bandwidth_       0    ;# not used
    LL set off_prune_       0    ;# not used
    LL set off_CtrMcast_    0    ;# not used

    Agent/Null set sport_        0
    Agent/Null set dport_        0

    Agent/CBR set sport_         0
    Agent/CBR set dport_         0

    Agent/TCPSink set sport_     0
    Agent/TCPSink set dport_     0

    Agent/TCP set sport_         0
    Agent/TCP set dport_         0
    Agent/TCP set packetSize_    1460

    Queue/DropTail/PriQueue set Prefer_Routing_Protocols    1

    # unity gain, omni-directional antennas
    # set up the antennas to be centered in the node and 1.5 meters above it
    Antenna/OmniAntenna set X_    0
    Antenna/OmniAntenna set Y_    0
    Antenna/OmniAntenna set Z_    1.5
    Antenna/OmniAntenna set Gt_   1.0
    Antenna/OmniAntenna set Gr_   1.0

    # Initialize the SharedMedia interface with parameters to make
    # it work like the 914MHz Lucent WaveLAN DSSS radio interface
    Phy/WirelessPhy set CPThresh_ 10.0
    Phy/WirelessPhy set CSThresh_ 1.559e-11
    Phy/WirelessPhy set RXThresh_ 3.652e-10
    Phy/WirelessPhy set Rb_       2*1e6
    Phy/WirelessPhy set Pt_       0.2818
    Phy/WirelessPhy set freq_     914e+6
    Phy/WirelessPhy set L_        1.0

# ==========================================================

    proc usage { argv0 }  {
       puts "Usage: $argv0"
       puts "\tmandatory arguments:"
       puts "\t\t\[-x MAXX\] \[-y MAXY\]"
       puts "\toptional arguments:"
       puts "\t\t\[-cp conn pattern\] \[-sc scenario\] \[-nn nodes\]"
       puts "\t\t\[-seed seed\] \[-stop sec\] \[-tr tracefile\]\n"
    }
```

```
proc getopt {argc argv} {
   global opt
   lappend optlist cp nn seed sc stop tr x y

   for {set i 0} {$i < $argc} {incr i} {
         set arg [lindex $argv $i]
         if {[string range $arg 0 0] != "-"} continue

         set name [string range $arg 1 end]
         set opt($name) [lindex $argv [expr $i+1]]
   }
}


proc cmu-trace { ttype atype node } {
   global ns_ tracefd

   if { $tracefd == "" } {
         return ""
   }
   set T [new CMUTrace/$ttype $atype]
   $T target [$ns_ set nullAgent_]
   $T attach $tracefd
         $T set src_ [$node id]

         $T node $node

   return $T
}


proc create-god { nodes } {
   global ns_ god_ tracefd

   set god_ [new God]
   $god_ num_nodes $nodes
}

proc log-movement {} {
    global logtimer ns_ ns

    set ns $ns_
    source           /home/tota/leach/ns-allinone-2.34/ns-2.34/tcl/mobility/timer.tcl
    Class LogTimer -superclass Timer
    LogTimer instproc timeout {} {
   global opt node_;
   for {set i 0} {$i < $opt(nn)} {incr i} {
       $node_($i) log-movement
   }
   $self sched 0.1
```

```
        }

        set logtimer [new LogTimer]
        $logtimer sched 0.1
    }

# ==========================================================
# Main Program
# ==========================================================
    getopt $argc $argv

#
# Source External TCL Scripts
#
    source /home/tota/leach/ns-allinone-2.34/ns-2.34/tcl/lib/ns-mobilenode.tcl

    #if { $opt(rp) != "" } {
       source /home/tota/leach/ns-allinone-2.34/ns-2.34/tcl/mobility/$opt(rp).tcl
       #} elseif { [catch { set env(NS_PROTO_SCRIPT) } ] == 1 } {
       #puts "\nenvironment variable NS_PROTO_SCRIPT not set!\n"
       #exit
    #} else {
       #puts "\n*** using script $env(NS_PROTO_SCRIPT)\n\n";
             #source $env(NS_PROTO_SCRIPT)
    #}
    source /home/tota/leach/ns-allinone-2.34/ns-2.34/tcl/lib/ns-cmutrace.tcl

# do the get opt again incase the routing protocol file added some more
# options to look for
    getopt $argc $argv

    if { $opt(x) == 0 || $opt(y) == 0 } {
       usage $argv0
       exit 1
    }

    if {$opt(seed) > 0} {
       puts "Seeding Random number generator with $opt(seed)\n"
       ns-random $opt(seed)
    }

#
# Initialize Global Variables
#
    set ns_       [new Simulator]
    set chan[new $opt(chan)]
```

```
    set prop[new $opt(prop)]
    set topo[new Topography]

    # setup output trace file
    #set tracefd  [open $opt(rp).tr w]
    #set tracefd  [open leach.tr w]
    set tracefd   [open $opt(tr) w]

    # try for setup output nam file
    set nam_vystup          [open $opt(rp).nam w]
    $ns_ namtrace-all-wireless $nam_vystup $opt(x) $opt(y)
    # end

    $topo load_flatgrid $opt(x) $opt(y)

    $prop topography $topo

#
# Create God
#
    create-god $opt(nn)


    #
    # log the mobile nodes movements if desired
    #
    if { $opt(lm) == "on" } {
        log-movement
    }

#
#  Create the specified number of nodes $opt(nn) and "attach" them
#  the channel.
#  Each routing protocol script is expected to have defined a proc
#  create-mobile-node that builds a mobile node and inserts it into the
#  array global $node_($i)
#
    if { [string compare $opt(rp) "dsr"] == 0} {
       for {set i 0} {$i < $opt(nn) } {incr i} {
            dsr-create-mobile-node $i
       }
    } elseif { [string compare $opt(rp) "dsdv"] == 0} {
       for {set i 0} {$i < $opt(nn) } {incr i} {
            dsdv-create-mobile-node $i
       }
    } elseif { [string compare $opt(rp) "leach"] == 0} {
       for {set i 0} {$i < $opt(nn) } {incr i} {
            leach-create-mobile-node $i
       }
```

```
    } elseif { [string compare $opt(rp) "leach-c"] == 0} {
       for {set i 0} {$i < $opt(nn) } {incr i} {
             leach-create-mobile-node $i
       }
    } elseif { [string compare $opt(rp) "stat-clus"] == 0} {
       for {set i 0} {$i < $opt(nn) } {incr i} {
             leach-create-mobile-node $i
       }
    } elseif { [string compare $opt(rp) "mte"] == 0} {
       for {set i 0} {$i < $opt(nn) } {incr i} {
             leach-create-mobile-node $i
       }
    } elseif { [string compare $opt(rp) "pegasis"] == 0} {
       for {set i 0} {$i < $opt(nn) } {incr i} {
             leach-create-mobile-node $i
       }
    }
#
# Source the Connection and Movement scripts
#
    if { $opt(cp) == "" } {
       puts "*** NOTE: no connection pattern specified. - wireless.tcl"
             set opt(cp) "none"
    } else {
       puts "Loading connection pattern...- wireless.tcl"
       source $opt(cp)
    }

    if { $opt(sc) == "" } {
       puts "*** NOTE: no scenario file specified. - wireless.tcl"
             set opt(sc) "none"
    } else {
       puts "Loading scenario file... - wireless.tcl"
       source $opt(sc)
       puts "Load complete... - wireless.tcl"
    }

#
# Tell all the nodes when the simulation ends
#
    for {set i 0} {$i < $opt(nn) } {incr i} {
        $ns_ at $opt(stop).000000001 "$node_($i) reset";
    }

# original end $ns_ at $opt(stop).00000001 "puts \"NS EXITING...\" ; $ns_ halt"

# new end
    $ns_ at $opt(stop).000000001 "finish"
```

```
    $ns_ at $opt(stop).000000002 "puts \"NS EXITING...\" ; $ns_ halt"

# Change for finish
    proc finish {} {
       global ns_ nam_vystup
       $ns_ flush-trace
       close $nam_vystup
       #exit 0
       #exec nam out_aodv_big_auto.nam &
    }
# end of change

    puts $tracefd "M 0.0 nn $opt(nn) x $opt(x) y $opt(y) rp $opt(rp)"
    puts $tracefd "M 0.0 sc $opt(sc) cp $opt(cp) seed $opt(seed)"
    puts $tracefd "M 0.0 prop $opt(prop) ant $opt(ant)"

    puts "Starting Simulation... - wireless.tcl"
    $ns_ run
```