

P-2592

**SERVICE COORDINATION IN
MANET**



A PROJECT REPORT

Submitted by

RANJITHAM.E	71205205046
RUBHA.A.K	71205205048
SHARADHA.R	71205205053

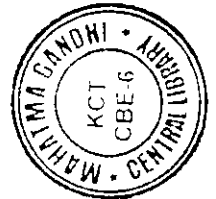
in partial fulfilment for the award of the degree

of

BACHELOR OF TECHNOLOGY

IN

INFORMATION TECHNOLOGY



KUMARAGURU COLLEGE OF TECHNOLOGY, COIMBATORE

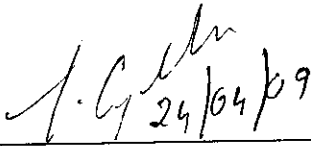
ANNA UNIVERSITY : CHENNAI 600 025

APRIL 2009

ANNA UNIVERSITY : CHENNAI 600 025

BONAFIDE CERTIFICATE

Certified that this project report “**SERVICE COORDINATION IN MANET**” is the bonafide work of “**RANJITHAM.E, RUBHA.A.K, SHARADHA.R**” who carried out the project work under my supervision.



SIGNATURE

Mrs.J.Cynthia, M.E,
SENIOR LECTURER
Dept of Information Technology,
Kumaraguru College of Technology,
Coimbatore – 641 006.



SIGNATURE

Dr.Thangasamy,B.E(Hons),Ph.D.,
DEAN
Dept of Computer Science and
Engineering,
Kumaraguru College of Technology,
Coimbatore – 641 006.

The candidates with University Register No 71205205046,
71205205048 and 71205205053 were examined by us in the project viva-
voice examination held on **.28.04.09**

INTERNAL EXAMINER



EXTERNAL EXAMINER

DECLARATION

We,

RANJITHAM.E

Reg.No: 71205205046

RUBHA.A.K

Reg.No: 71205205048

SHARADHA.R

Reg.No: 71205205053

hereby declare that the project entitled “**SERVICE COORDINATION IN MANET**”, submitted in partial fulfillment to Anna University as the project work of Bachelor of Technology (Information Technology) degree, is a record of original work done by us under the supervision and guidance of Department of Information Technology, Kumaraguru College of Technology, Coimbatore.

Place: Coimbatore

Date:

E. Ranjitham

A. K. Rubha

Sharadha R

[RANJITHAM.E]

[RUBHA.A.K]

[SHARADHA.R]

Project Guided by

J. Cynthia M.E
24/04/09

[Ms. J.Cynthia M.E]

ACKNOWLEDGEMENT

We express our sincere thanks to our Chairman **Padmabhushan Arutselvar Dr. N. Mahalingam B.Sc., F.I.E.**, Vice Chairman **Dr.K. Arumugan B.E., M.S., M.I.E.**, Correspondent **Shri.M.Balasubramaniam and Joint Correspondent Dr.A.Selvakumar** for all their support and ray of strengthening hope extended. We are immensely grateful to our Vice Principal **Prof.R.Annamalai**, for his invaluable support to the outcome of this project.

We are deeply obliged to **Dr.S.Thangasamy B.E(Hons)., Ph.D**, Dean Department of Computer Science and Engineering for his valuable guidance and useful suggestions during the course of this project.

We also extend our heartfelt thanks to our project co-ordinator **L.S Jayashree M.E., Ph.D.**, Asst.Prof, Department of Information Technology for providing us her support which really helped us.

We are indebted to our project guide and extend our gratitude toward **Mrs.J.Cynthia M.E**, Senior Lecturer, Department of Information Technology for her helpful guidance and valuable support given to us throughout this project.

We thank the teaching and non-teaching staffs of our Department for providing us the technical support during the course of this project. We also thank all of our friends who helped us to complete this project successfully.

ABSTRACT

ABSTRACT:

Efficient routing and service provisioning in Mobile adhoc networks is a big research challenge. Conventional schemes such as tree based and mesh based unicast and multicast routing protocols are satisfactory but are very difficult to scale due to their overhead in their routing schemes. Conventional service provisioning schemes have limited scalability and efficiency. Most preferred protocols nowadays are the geographic unicast and multicast protocols, which are less scalable when combined with the conventional service provisioning schemes. Hence our aim is to propose a service coordination scheme which possesses the features of scalability, efficiency, robustness and adaptability. In this scheme we analyze three different leader election algorithms, namely zone id based, hash based and voting algorithm, through which moderated virtual zone construction, proficient service management, competent tracking of services and packet delivery with lesser overhead can be accomplished.

TABLE OF CONTENTS

CHAPTER NO	TITLE	PAGE NO
	ABSTRACT	
1.	INTRODUCTION	1
2.	LITERATURE REVIEW	
	2.1 SERVICE PROVISIONING	4
	2.1.1 SERVICE PROVISIONING WITH REACTIVE ROUTING PROTOCOL	4
	2.1.2 SERVICE PROVISIONING WITH PROACTIVE ROUTING PROTOCOL	5
	2.1.3 SERVICE PROVISIONING WITH HYBRID ROUTING PROTOCOL	6
	2.1.4 SERVICE PROVISIONING WITH LOCAL DIRECTORY	7
	2.1.5 SERVICE PROVISIONING WITH WITH CENTRALIZED AND DISRIBUTED DIRECTORY	8
	2.2 EXISTING SYSTEM	
	2.2.1 TREE BASED MULTICASTING PROTOCOL	9
	2.2.1.1 DVMRP	10
	2.2.1.2 MAODV	11
	2.2.2 MESH BASED MULTICASTING PROTOCOL	11
	2.2.2.1 ODMRP	12

	2.2.3 HYBRID MULTICASTING PROTOCOL	12
	2.3 PROPOSED SYSTEM	
	2.3.1 GEOGRAPHIC LOCATION SERVICES	13
	2.3.2 SCALABLE POSITION BASED MULTICAST	13
	2.3.3 SERVICE CO-ORDINATOR ELECTION	
	2.3.3.1 HRPM	14
3.	METHODOLOGY	
	3.1 VIRTUAL ZONE CONSTRUCTION	16
	3.2 SERVICE MANAGEMENT	16
	3.3 LEADER ELECTION	17
	3.3.1 ZONE ID BASED LEADER ELECTION	17
	3.3.2 HASHING BASED LEADER ELECTION	18
	3.3.3 VOTING ALGORITHM BASED LEADER ELECTION	20
	3.4 SERVICE PROVISIONING	21
4.	SIMULATION ENVIRONMENT AND SCENARIO	
	4.1 SIMULATION ENVIRONMENT	
	4.1.1 NETWORK SIMULATOR	22

4.1.2	COMPONENTS OF NS	22
4.1.2.1	NAM, THE NETWORK ANIMATOR	22
4.1.2.2	PRE-PROCESSING	22
4.1.2.3	POST-PROCESSING	22
4.1.3	GOALS OF NS	23
4.1.4	TWO LANGUAGES	23
4.1.5	IMPORTANT NS-2 PROGRAMMING LANGUAGES	24
4.1.6	WHY TWO LANGUAGES	25
4.1.7	NAM (NETWORK ANIMATOR)	25
4.1.8	XGRAPH	26
4.1.9	CREATION OF TRACE FILES	26
4.2	SIMULATION SCENARIO	27
5.	PERFORMANCE EVALUATION	
5.1	THROUGHPUT ANALYSIS	32
5.2	DATALOSS ANALYSIS	33
6.	CONCLUSION	36
7.	FUTURE ENHANCEMENTS	37

8.	APPENDICES	
	8.1 SOURCE CODE	38
	8.2 SCREEN SHOTS	63
9.	REFERENCES	67

INTRODUCTION

1. INTRODUCTION

Adhoc networks are self organizing, self healing, distributed networks which most often employ wireless transmission networks. A MANET is a dynamic self configurable wireless network which has no fixed infrastructure or central administration. These characteristics make MANETs suitable for mission critical applications such as disaster recovery, crowd control, search and rescue and automated battlefield communications. Group communications is important in MANET. Group communications is also very important in supporting multimedia applications such as gaming and conferencing. With a one-to-many or many-to-many transmission pattern, multicast is an efficient method to realize group communications and plays an important role in MANET. Nodes can move arbitrarily and the network topology can change frequently and unpredictably, and the band width and battery power are limited. For these reasons the development of routing protocols for MANET is extremely challenging. As a result multicast routing [6] has become a research focus recently, and various multicasting protocols in MANET have been proposed.

The conventional MANET multicast protocols can be divided into two main categories, tree-based and mesh-based. [6] The tree based concept is borrowed from the multicasting protocols in wired networks. Since efficiency can be achieved and robustness is not a critical issue in the stable wired network, most multicast methods are tree based, either source- or shared-tree-based. To let these multicasting products work in MANET, some modification and extension should be made. The tree based protocols (e.g. LAM, MAODV, AMR) construct a tree for the multicast delivery and the tree structure is known for its efficiency in utilizing the network resource optimally. However, maintaining tree structure in

these conventional protocols is very difficult, and the tree connection is easy to be broken and the construction is not reliable.

The mesh based protocols are proposed to enhance the robustness by providing redundant paths between the source and destination pairs at the cost of higher forwarding overhead. That is, when a route fails, which is common in mobile ad hoc networks, there should be another route to deliver the data. It is the redundancy of the routes that provides the fault tolerance. Furthermore, these conventional multicast protocols generally do not have good scalability due to the overhead for route searching, group membership management, and tree/mesh structure creation and maintenance over the dynamic topology of MANET.[5]

Service provisioning is closely related to routing. However, the current service provisioning protocols are normally built on or simply extended from the existing topology-based MANET routing protocols. Consequently, they inherit their limitations (e.g., limited scalability, relatively high control overhead and unreliability). In MANET routing, there is a trend to develop position-based routing schemes which are more robust and efficient than the traditional topology-based routing schemes. In this work, we make use of the position information in our service provisioning mechanism to increase its scalability and efficiency. Instead of using complicated schemes to manage network topologies, only the positions of service nodes need to be tracked.

A service provision framework [1] will support the following functions:

- 1) Service Discovery:** Locating the services based on user's requests.
- 2) Service Delivery:** Delivering relevant service data and control messages. Unicast routing is needed for the delivery between peer devices, while multicast routing would be required to support efficient group communications.

3) Service coordination: A service request may need to be satisfied by several candidate providers. To enable coordination, the service provision framework should be able to track a group of service providers and their services. Hence efficient and scalable service and membership management is required to facilitate the selection of appropriate providers and the collaboration among multiple providers. All the above functions are very important for a complete service provision framework, while current work normally focuses on only a subset of the problems (e.g., service discovery and/or service delivery), without fully considering or efficiently supporting the other functions. Instead, our work is aimed to support all these functions.

LITERATURE REVIEW

2. LITERATURE REVIEW

2.1 Service provisioning

Service related protocols in MANET are mainly focused on service discovery and this discovery schemes are normally integrated with different kinds of routing protocols namely reactive, proactive, hybrid, unicast, multicast and anycast protocols.

2.1.1 Service provisioning with reactive routing protocols:

Service discovery architectures follow two mechanisms.

- Information about services offered on the network is stored on one or few centralized zones, referred to as service coordinators(SCs)
- Information about each service is stored on each node offering the service.

A solution that is only based on first mechanism is referred to as a service coordinator based architecture while a solution based on the second mechanism is referred to as distributed query based architecture. Finally a solution based on a mixture of both the first and the second mechanism is referred to as hybrid architecture. [7]

It looks a user friendly solution that gives a high level of service availability, low discovery delay and so forth. At the same time, it wants a network friendly solution i.e with low messaging overhead and with little additional complexity added to the network. To a certain degree it is also possible to increase the user friendliness at the cost of introducing more messaging.

However in a dynamic topology with network entries and departures the service coordinators of the hybrid architecture have the disadvantage of sometimes providing the user agents with ‘false positives’ i.e with outdated bindings of servers that have already left the network. Moreover the hybrid approach may call for a separate complicated mechanism for electing service coordinators which might require a substantial amount of network resources. This was not taken into account of the analysis. The design of a light weight, dynamic mechanism for election of service coordinators would also require further research.

2.1.2 Service provisioning with proactive routing protocol

This method presents a light weight service discovery mechanism for the mobile adhoc pervasive environment applying cross layer design to reduce the infrastructure and protocol overhead and to improve service accessibility. Integrated with the network routing layer, the proposed mechanism automatically identifies the proper service discovery models for the current network configuration.[4]

When a directory is available in the network the service nodes register with the directory about their service records and refresh the records periodically. If no directory is known a node replies directly to a query if it has a matching service. A service in this case also has an option to periodically advertise in the network unless it is set to limit the network traffic load.

When not knowing any directory a client sends the service query to the entire network. If having the information of a directory server, the client sends the query to the directory server. When the directory fails to reply to the query the client would resend the query with a flag to the network asking the service to directly respond. [7]

If no directory responds to this query, the directory is marked then as not available by all the nodes that hear the flagged query and the response message from the service node instead from a directory. Thus this scheme automatically switches between the client service and the client service directory models, adapting to the dynamics in the MANET.

2.1.3 Service provisioning with hybrid routing protocol

This protocol provides a service provisioning scheme that can be greatly enhanced in terms of efficiency regarding service discoverability and energy consumption by piggybacking service information in routing layer messages. A node requesting a service in addition to discovering that service it is simultaneously informed of the route to the service provider. It extended the zone routing protocol in order to encapsulate service information in its routing messages.

In this case a hybrid approach is developed in which a node uses

- Proactive routing (eg. DSDV)
- Reactive routing

2.1.4 Service provisioning with local directory

Introduced a scalable service discovery protocol for MANET, which is based on the homogeneous and dynamic deployment of cooperating directories within the network. Scalability of our protocol comes from the minimization of the generated traffic, and the use of compact directory summaries that enable to efficiently locate the directory that most likely caches the description of a given service.

The architecture composed of up to a hundred of nodes, and further supporting bridging with other networks, either ad hoc or infrastructure-based. While decentralized service discovery is a priori more suited to the specifics of MANET by not assigning any specific role to given nodes, it is too costly in terms of traffic. But, centralized service discovery based on service directory (i.e.) either makes the protocol too much dependent on specific node(s) or induces too much resource consumption in the networks.

This solution relied on the dynamic and homogeneous deployment of backbone cooperating service directories within the MANET. The MANET is structured around a directory in which directory coverage is defined with respect to a given number of hops, which is set according to the nodes density, as known from the underlying routing protocol.[4]

Dynamic employment of directories is then carried out so that directories cover distinct areas. Scalability of our solution further comes from the fact that:

- Protocol minimizes the generated traffic using broadcasting.

- The directory content is summarized using a Bloom filter, which provides a highly compact summary that may be exchanged with other directories, and is used to accurately locate the directory that holds the description of a given service.

The deployment feature for implementing the protocol as a middleware service is highly required to support the composition of Web services deployed over mobile nodes in a hybrid network which was not discussed.

2.1.5 Service Provisioning with Centralized and Distributed Directory

Ref developed a distributed service discovery [7] architecture, which relies on a topology. It consists of two independent components:

- Formation of a virtual backbone and
- Distribution of service registrations, requests, and replies.

The first component creates a mesh structure from a subset of a given network graph that includes the nodes acting as service brokers and a subset of paths (which we refer as virtual links) connecting them. Service broker nodes constitute a dominating set, i.e. all the nodes in the network are either in this set or only one-hop away from at least one member of the set.

The second component establishes sub-trees rooted at service requesting nodes and registering servers for efficient dissemination of the service discovery probing messages. Simulation results were provided for comparison of performance measures, i.e. latency, success rate, and control

message overhead, when different architectures and network support mechanisms are utilized in service discovery.

2.2 Existing system

Group communication is important in Mobile Ad Hoc Networks (MANET). Group communications is also very important in supporting multimedia applications such as gaming and conferencing. With a one-to-many or many-to-many transmission pattern, multicast is an efficient method to realize group communications. The high dynamics of MANET, however, makes the design of routing protocols much more challenging than that for wired network.

The conventional MANET multicast protocols can be divided into two main categories. They are:

- Tree based Multicasting Protocols[5]
- Mesh based Multicasting Protocols[5]

2.2.1 Tree based Multicasting Protocol

Most multicast methods are tree-based, either source- or shared tree based. The former one will construct a multicast tree among all the member nodes for each source node; usually this is a shortest path tree. This kind of protocol is more efficient for the multicast, but has too much routing information to maintain and has less scalability. The later one constructs only one multicast tree for a multicast group including several source nodes. Every source uses this tree to do multicast. Usually the shared tree constructed is a minimum spanning tree. Since the path between a sender

and a receiver is not necessarily the shortest path, the shared-tree-based protocol in doing multicast, but it reduces the overhead greatly by maintaining less routing information. The following two protocols are developed for MANET.

- DVMRP (Distance Vector Multicast Routing Protocol)
- MAODV (Multicast O-Demand Distance Vector Routing Protocol)

2.2.1.1 DVMRP (Distance Vector Multicast Routing Protocol)

This protocol uses reverse path algorithm to construct a source based multicast tree in a multicast tree in a multicast group for each source node. In DVRMP there are two main phases to construct a multicast tree [6]. A source node will broadcast an advertise message at the first phase. Every intermediate node will forward this message if the link from which it gets the message is on the shortest path to the source. Therefore the parent - child relationship is established and a shortest path tree rooted on the source node among all the nodes is established. The second phase is pruning. Every leaf node if it's not group member will send to its parent node. The intermediate parent node will send a prune message to its parent if there is no group member in the sub tree rooted by it. In this way the sub trees without member nodes are pruned by being deleted from their parents forwarding table. Thus a source based multicast tree is established. Later when a non member node wants to join this multicast group, it will send a graft message to its parent. The parent will forward the graft message upstream until the message reaches an on-tree node. In this way, this branch is grafted on the source-base multicast tree.

2.2.1.2 MAODV(Multicast On-Demand Distance Vector Routing Protocol)

This protocol uses broadcast to find the route in a demand way and constructs a shared routing tree. The node that wants to join a multicast group or has data to send will broadcast a Route Request (RREQ) message. This message will be rebroadcast by all the intermediate nodes until it reaches an on-tree node. This on-tree node can then reply a Request Response (RREP) message by unicast along the reverse path to the sender. The sender node may get more than one RREP, if so, it will select the best one based on sequence number and hop count, then unicast an activation message along this selected path.[5] Every intermediate node on this path will be a forwarding node. It sets up entry in its routing table to add the sender and itself on the tree. In this way, the multicast tree has only a single path to any tree node. This protocol uses hard state in its routing table, that is to say, the state information is updated when failure occurs, contrary to soft state, in which routing table is updated periodically. When a link failure occurs, it will be detected and some kind of repair will be done.

2.2.2 Mesh based Multicasting Protocols:

The mesh based method is much more suited for MANET, which demands more robustness of the protocol. That is, when a route fails, which is common in mobile ad hoc networks, there should be another route to deliver the data. It is the redundancy of the routes that provides the fault tolerance.



2.2.2.1 ODMRP (On-Demand Multicast Routing Protocol)

This protocol uses the concept of ‘forwarding node’ to do the multicasting. It finds some nodes to be ‘forwarding node’ in the whole network, and only these will forward multicast messages. [5] The source on-demand establishes the routes by broadcasting the JoinData message with TTL. This message is periodically generated to refresh both the membership and routes. Every intermediate node will add the upstream node’s ID in its own routing table upon receiving this message. The message will be forwarded until it group member then creates a JoinTable message and broadcasts this message to all its neighbors.

2.2.3 Hybrid Multicasting Protocols

These protocols are a combination of tree-based and mesh-based methods to seek both efficiency and robustness. It has two main procedures. One is mesh creation, the other is tree creation. It first creates the virtual mesh links among the group members based on the physical links. A logic core will be selected from the members in this procedure. It then uses this mesh to establish the multicast tree. The logical core will initiate the tree creation. The tree can stay unchanged even if the topology of the network changes, as long as the links between the core node and tree members still work with the help of virtual mesh. The neighbors on the multicast tree are connected by the underlying unicast tunnels which have the responsibility to deal with dynamic network topology. Both the tree and mesh are quite static.

2.3 Proposed system

2.3.1 Geographic location services

GLS [6] is a new distributed location service which tracks mobile node locations. GLS combined with geographic forwarding allows the construction of adhoc mobile networks that scale to a larger number of nodes than possible with previous work. Each mobile node periodically updates a small set of other nodes with its current location. A node sends its position updates to its location servers without knowing their actual identities, assisted by a predefined ordering of node identifiers and a predefined geographic hierarchy. Queries for a mobile node's location also use the predefined identifier ordering and spatial hierarchy to find a location server for that node.

Furthermore GLS tolerates node failures well. Each failure has only a limited effect and query performance degrades peacefully as nodes fail and restart. The query performance of GLS is also relatively insensitive to node speeds. Simple geographic forwarding combined with GLS compares favorably with Dynamic Source Routing (DSR).

With the help of the GPS i.e. the Global Positioning System, each node is assumed to know the location of every other node present in its network and hence tracking of resource requests, data packet deliveries are made easy.

2.3.2 Scalable position based multicast

SPBM [8] is a multicast routing protocol for adhoc networks. SPBM uses the geographic position of nodes to provide a highly scalable group membership scheme and to forward data packet with a very low overhead. It bases

its multicast forwarding decision on whether there are group members located in a given direction or not, allowing for a hierarchical aggregation. Because of aggregation the overhead for group membership management is bounded by a small constant while it is independent of the number of multicast senders for a given multicast group. The group management scheme is responsible for the dissemination of the membership information for multicast groups, so that forwarding nodes know in which direction receivers are located. The multicasting forwarding algorithm is executed by a forwarding node to determine the neighbors that should receive a copy of the given multicast. This decision is based on the information based on the information by the group management scheme.

In general location services are broadly connected into two namely flooding based and quorum based. Flooding based is further divided into proactive and reactive. And quorum is divided into explicit and implicit methods. The explicit and implicit methods are further subdivided into hierarchical and flat services respectively

2.3.3 Service coordinator election

2.3.3.1 Hierarchical rendezvous point membership management

To join a hierarchically decomposed multicast group, a node first generates the hashed location for the RP (rendezvous point) and sends a JOIN message to the RP, same as in the flat scenario. After receiving the value of the current decomposition index d of the hierarchy from the RP, the joining node invokes the hashing function with d and its current location to compute the hashed location of the AP of its cell. The node then starts periodically sending the LOCATION UPDATE packets to it. Such location updates are soft state and serve as a subgroup

membership update i.e. if an AP stops receiving location update from a member, it assumes the member has migrated to another cell.

The state the RP needs to keep about the group is just a bit vector of d^2 bits with each bit representing whether a member exists in a particular or not. Thus the RP can easily encode a large number of AP's. The frequency of location update determines the accuracy of the knowledge at the RP/APs and consequently the accuracy of the multicast tree. We use threshold based updates where each node initiates a LOCATION UPDATE whenever it moves 100m from the location of the last update.

METHODOLOGY

3. METHODOLOGY

3.1 Virtual zone construction

The given geographic area is divided into a number of zones. Each zone consists of a number of nodes which can offer different services. The length of a side of the zone square is defined as *zone size*. [1] Each virtual zone has a zone ID (zID) to help identify and locate a zone. Each node in every zone is provided with a node ID. zID for each node is denoted as (a,b) the value of (a,b) can be calculated from its position coordinates such that

$$a = [x_j / x_0 / (\text{zone size})]$$

$$b = [y_j / y_0 / (\text{zone size})],$$

where (x₀; y₀) is the position of the virtual origin and (x,y) is its position coordinates. For simplicity, we assume that all zone IDs are positive. Each zone has a zone centre denoted by (x_c,y_c). For a zone with zID (a,b), the position of its center (x_c; y_c) is:

$$x_c = x_0 + (a + 0.5) / \text{zone size}$$

$$y_c = y_0 + (b + 0.5) / \text{zone size}.$$

A packet destined to a zone will be forwarded towards its centre. [8] In general the zones further divide themselves into sub zones when their network density increases. When the number of nodes in a particular zone increases, the zone divides itself into two separate zones. One divides into two, two into four and so on. Each of the sub divided zone will be self sufficient. For simplicity reasons we have not considered the problem of zone division in this project.

3.2 Service management

Some nodes in every zone provides a particular service. When the nodes move from one zone to the other the services of the new zones are also advertised

to this node and they are acquired by the newly entered node if they are in need of that particular service.

A node in a particular zone may have a number of services. Initially the division of the zones might also happen based on the services provided. Nodes offering similar services are grouped in a particular zone.

The way in which the services are managed will play an important role in the resource tracking. Based on the efficiency with which the services are managed, the need for the services are properly tracked and service provisioning is done.

3.3 Leader election

For each zone (hierarchy level) the leader is elected to handle the service provisioning functions to all nodes in the respective zones. Regional coordinator is elected to coordinate all the local coordinators. Three types of leader elections are:

- Zone id based leader election
- Hash based leader election
- Voting algorithm based leader election

3.3.1 Zone id based leader election

An access point (AP) is termed as the zone leader and electing it based on ZONE ID is one of the most conventional methods. In electing a global coordinator that is a rendezvous point (RP) the node with the highest zone ID is made the leader. When a node appears in the network it sends out a beacon announcing its existence. And then it waits for a $\text{intval}(\text{min})$ period for the beacons from other nodes. Every $\text{intval}(\text{min})$ [1] a node will check its neighbor table and

decides its leader under different cases: 1) The neighbor table contains no other zNodes and it will announce itself as the leader. 2) All the zNodes flags are unset that means no zNode has announced the leadership role. If the node is closer to the zone center than other nodes it will announce its leadership role through beacon message. 3) More than one zNodes have their flags set, the one with the largest node ID is selected. If the node's own flag is set before the checking but another node wins as zLD, the node will deliver its multicast table to the elected zone leader. 4) Just one flag is set for its zNodes with flagset is zLD[1].

So in simple terms for electing a zone leader that is an access point the node, with the highest node ID is selected.

3.3.2 Hashing based leader election

The concept of group management is introduced assuming a flat geographic domain. We then introduce hierarchical domain decomposition of a multicast group and describe how to apply RPGM [3] recursively in a hierarchy of sub domains. Rendezvous point group management allows multicast group members to leverage geographic hashing group management. Any node that wants to join a multicast group first hashes the group identifier to obtain the RP's location in the physical domain of the network using a hash function.

$$H(\text{GID}) = \{x, y\} \text{ where } x, y \in \text{MANET region}$$

This hashing function takes as input the group identifier and outputs a location (x- and y- coordinates) contained in the region. Note that we assume that this is a well known hash function that is known by nodes that enter the network through external means or using some resource discovery process [3].

After obtaining the hashed RP location for the group it wants to join, the node sends a JOIN message addressed to this hashed location. This JOIN message is routed by geographic forwarding to the node that is current closest to the hashed location in the network. This node is the designated RP at this time. Since there is only one such node at any given time, the JOIN messages from all the group members converge at a single RP in a distributed fashion without global knowledge.

Note that computing the hashed location assumes that all nodes know the approximate geographic boundaries of the network. Such boundary information may be pre-configured at nodes before deployment or discovered using some simple protocol.

To join a hierarchically decomposed multicast group a node first generates the hashed location for the RP and sends a JOIN message to the RP, same as in a flat domain scenario. After receiving the value of the current decomposition index d of the hierarchy from the RP, the joining node invokes the hashing function with d and its current location to compute the hashed location of the AP of its cell [1]. The node then starts periodically sending LOCATION UPDATE packets to its AP. Such location updates are soft state and serve as a subgroup membership update i.e. if an AP stops receiving location update from member, it assumes the member has migrated to another cell.

Upon receiving a location update[4] from each member, the AP summarizes the membership inside its cell as non empty and further propagates to the RP whenever the membership switches between empty and non empty. The cells in which no group members exists do not have any active APs and consequently no updates from these cells are sent to the RP.

3.3.3 Voting algorithm based leader election

Initially individual mobile users are ranked based on the predefined hierarchy and geographic domain of the organization. Based on the ranking mobile users are placed at respected levels. The top hierarchy is initially the global coordinator.

The global coordinator in turn delivers multicasting contents [5] to the local coordinator and sometimes routed through zonal coordinators at their intermediate level. The local coordinator delivers multicasting content to the service nodes in the respective geographic region.

The idea of the voting algorithm is fetched from the cluster head election [2] in clustering algorithm. In this, different parameters such as computational ability, battery power, the node ratio, transmission power, node mobility and node positioning are considered. The eligibility factor called EF of the node I that is utilized to serve as a CH at a particular time t is calculated as,

$$EF_i(t) = a_1 e^{-v_i(t)} + a_2 B_i(t) + a_3 (1 - E_i(t))$$

Where $v_i(t)$ is the mobile nodes average speed at time t and $B_i(t)$ is the remaining battery power in node I at time t. $E_i(t)$ is the Euclidean distance of the node I to the cluster centre calculated at time t and a_1, a_2, a_3 are the weighting factors that reflect the importance of each parameter. The node with has the highest value of EF will elect itself as the CH in the cluster.

The same idea is followed for the election of AP and RP in our case except that for reducing the complexity we consider only three factors namely battery power, average speed of the nodes and the Euclidean distance from the zone centre. Battery power should be high for a node to stay alive longer and

provide service. The average speed of the node should be less for it to stay in a particular zone for a longer time without showing faster switches between zones. The Euclidean distance between the node and its zone centre [1] should be less to receive packets quicker since the packets forwarded to a zone are forwarded towards its centre. The node which moderately excels in all the three factors is made the leader.

The leader election happens from time to time at fixed intervals. The RP and AP are elected at the start of the simulation. The leader election then happens at regular intervals such as 0 seconds, 7th second and fourteenth seconds based on the three algorithms zone ID based, hash based and voting algorithm based respectively. The RP leader election happens at the 0 and 9th seconds.

3.4 Service provisioning

Once an AP for a zone is elected all the data are routed through it. It maintains the record of all the nodes and the services present in its network. It also has prior information about the leaders of its neighboring zones. So when a node requests a service present in its own zone or from another zone, the request is made through the leader [7]. If the service is present in the same zone, the leader routes the QUERY MESSAGE to the SP and the SP responds to the SR by means of a HIT MESSAGE. [8]

***SIMULATION ENVIRONMENT
AND SCENARIO***

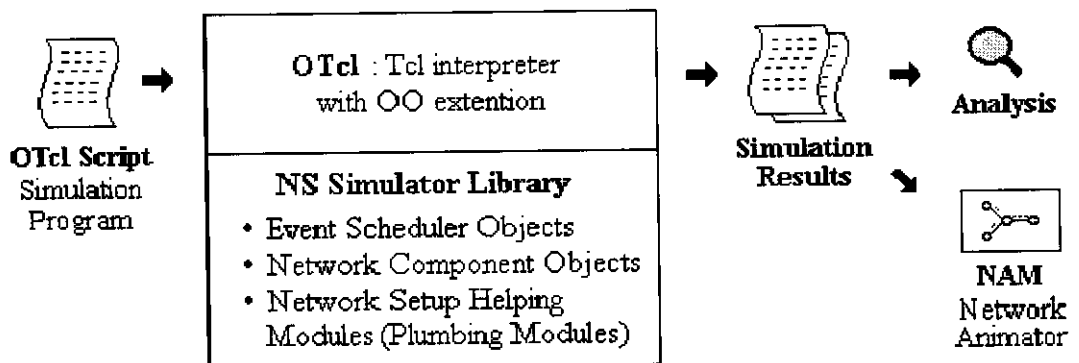
4. SIMULATION ENVIRONMENT AND SCENARIO

4.1 Simulation environment

4.1.1 Network simulator (NS)

Ns-2 stands for Network Simulator Version 2. Ns-2 is a discrete event simulator

targeted at network research. Focused on modeling network protocols.



4.1.2 Components of NS

4.1.2.1 Nam, the Network AniMator

- Visualize ns (or other) output
- GUI input simple ns scenarios

4.1.2.2 Pre-processing

- Traffic and topology generators

4.1.2.3 Post-processing

- Simple trace analysis, often in Awk, Perl, or Tcl

4.1.3 Goals of NS

- Support networking research and education
 - Protocol design, traffic studies, etc.
 - Protocol comparison
- Provide a collaborative environment
 - Freely distributed, open source
 - Share code, protocols, models, etc.
 - Allow easy comparison of similar protocols
 - Increase confidence in results
 - Models provide useful results in several situations
- It covers multiple layers
 - Application layer, transport layer, network layer and link layer.
- Supports the simulation of Intserv/diffserv, Multicast, Transport, Applications Wireless(fixed, mobile, satellite)

4.1.4 Two languages

- C++:
 - Detailed protocol simulations require systems programming language

- Byte manipulation, packet processing, algorithm implementations
 - Run time speed is important
 - Turn around time(Run simulation, find bug, fix bug, re-compile) is slower
- **Tcl:**
 - Simulations of slightly varying parameters or configurations
 - Quickly exploring a number of scenarios
 - Iteration time(change the model and re-run) is more

4.1.5 Important NS-2 Programming Languages

- **NS-2**

It is an object oriented simulator, written in C++, with an OTcl(Object Tool

Command Language) interpreter as a front-end.

- **Back-end C++**

- Defining new agents, protocols and framework.
- Manipulations at the byte/bit levels.

- If you have to change the behavior of an existing C++ class in ways that weren't

anticipated

- **Front-end Otcl**

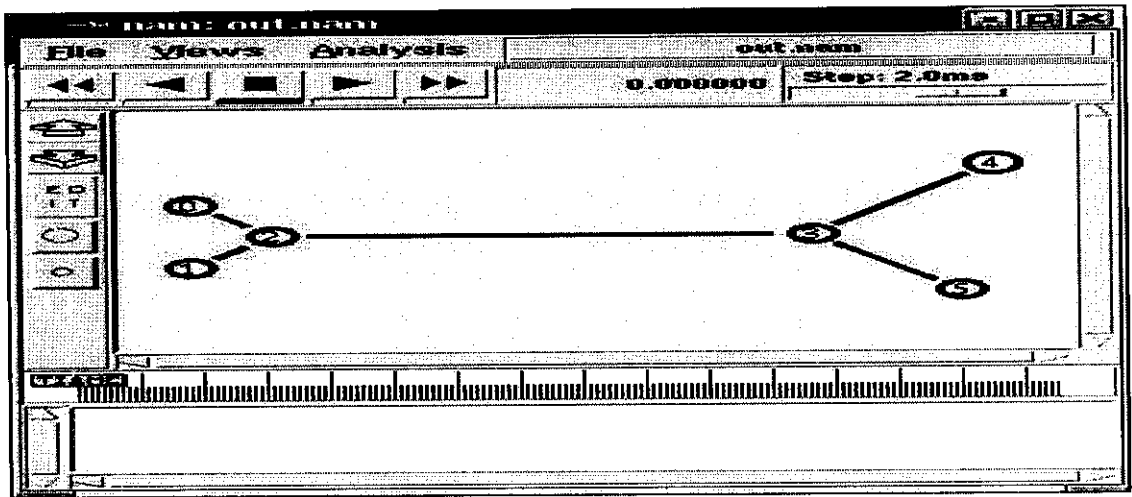
- Topologies, scenarios, simulations, ...
- Script language (easy topology modifications)
- If you can do what you want by manipulating existing C++ objects.

4.1.6 Why Two Languages

- Simulator had two distinct requirements
 - Detailed simulation of Protocol (Run-time speed)
 - Varying parameters or configuration (Change mkodel & rerun)
- C++ is fast to run but slower to change
- Otcl runs much slower but can be changed quickly.

4.1.7 Nam (Network AniMator)

“Nam is a Tcl/TK based animation tool for viewing network simulation traces and real world packet traces.”



4.1.8 Xgraph

Convert trace output into xgraph format

Commandline start : "xgraph"

4.1.9 Creation of Trace files

Used to trace packets on all links

Set tracefd [open simple.tr w]

\$ns_trace-all \$tracefd

- Format:

Event	Time	From node	To nodetype	Pkt size	Flags	Fid	Src addr	Dst addr	Seq num	Pkt id
-------	------	-----------	-------------	----------	-------	-----	----------	----------	---------	--------

r : receive (at to_node)

+ : enqueue (at queue)

- : dequeue (at queue)

d : drop (at queue)

```
r 1.086667 3 5 cbr 1000 ----- 2 1.0 5.0 1 1
r 1.111227 2 3 tcp 40 ----- 1 0.0 4.0 0 2
+ 1.111227 3 4 tcp 40 ----- 1 0.0 4.0 0 2
- 1.111227 3 4 tcp 40 ----- 1 0.0 4.0 0 2
```

4.2 Simulation scenario

The simulation was performed using the network simulator ns-2. The network field size is 1500mx 1500m, containing 15-20 mobile nodes. All the nodes follow the random waypoint mobility model with AODV as the default routing protocol with TCP traffic.

The experimentation is particularly interested in the scalability of multicast routing protocols and service provisioning. The geographic multicasting algorithm is used to deliver the data packets to the group member. The input to this algorithm is service packets with sources and destinations. The output of this algorithm is to deliver the service packet data to the group member.

- **Defining wireless options**

```

set val(chan)      Channel/WirelessChannel      ;#Channel Type
set val(prop)      Propagation/TwoRayGround     ;# radio-propagation
                                                           model
set val(netif)     Phy/WirelessPhy             ;# network interface type
set val(mac)       Mac/802_11                  ;# MAC type
set val(ifq)       Queue/DropTail/PriQueue     ;# interface queue type
set val(ll)        LL                           ;# link layer type
set val(ant)       Antenna/OmniAntenna         ;# antenna model
set val(ifqlen)    50                          ;# max packet in ifq
set val(nn)        16                          ;# number of mobilenodes
set val(rp)        DSDV                        ;# routing protocol
set val(x)         1500
set val(y)         1000

```

```

set ns_            [new Simulator]
set tracefd       [open tree.tr w]
$ns_ trace-all $tracefd
set namtrace      [open tree.nam w]
$ns_ namtrace-all-wireless $namtrace $val(x) $val(y)

```

- **set up topography object**

```

set topo          [new Topography]
$topo load_flatgrid $val(x) $val(y)

```

- **Create God – General Operations Descriptor**

```
create-god $val(nn)
```

- **Configure node:**

```
$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) \  
-topoInstance $topo \  
-agentTrace ON \  
-routerTrace ON \  
-macTrace ON \  
-movementTrace OFF
```

- **creation of nodes**

```
for { set i 0 } { $i < $val(nn) } { incr i } {  
    global n  
    set node_($i) [$ns_ node]  
}
```

- **To set the random motion**

```
for { set i 0 } { $i < $val(nn) } { incr i } {  
    $node_($i) random-motion 0  
}
```

- **To set the size of the nodes**

```
for { set i 0 } { $i < $val(nn) } { incr i } {  
    $ns_ initial_node_pos $node_($i) 60  
}
```

- **Adding nodes and setting positions**

```
$node_(1) set X_ 185.0  
$node_(1) set Y_ 550.0  
$node_(1) set Z_ 0.0
```

- **Adding agents and traffic source**

```
set tcp [new Agent/TCP]  
$tcp set class_ 1  
set sink [new Agent/TCPSink]  
$ns_ attach-agent $node_(1) $tcp  
$ns_ attach-agent $node_(2) $sink  
$ns_ connect $tcp $sink  
set ftp [new Application/FTP]
```

```
$ftp attach-agent $tcp1
$ns_ at 1.2 "$ftp start"
$ns_ at 2.5 "$ftp stop"
```

- **Tell nodes when the simulation ends**

```
for {set i 0} {$i < $val(nn)} {incr i} {
    $ns_ at 10.0 "$node_($i) reset";
}
$ns_ at 10.02 "stop"
$ns_ at 10.03 "puts \"NS EXITING...\" ; $ns_ halt"
proc stop {} {
    global ns_ tracefd
    $ns_ flush-trace
    close $tracefd
    puts "running nam..."
    exec nam tree &
}
puts "Starting Simulation..."
$ns_ run
```

PERFORMANCE EVALUATION

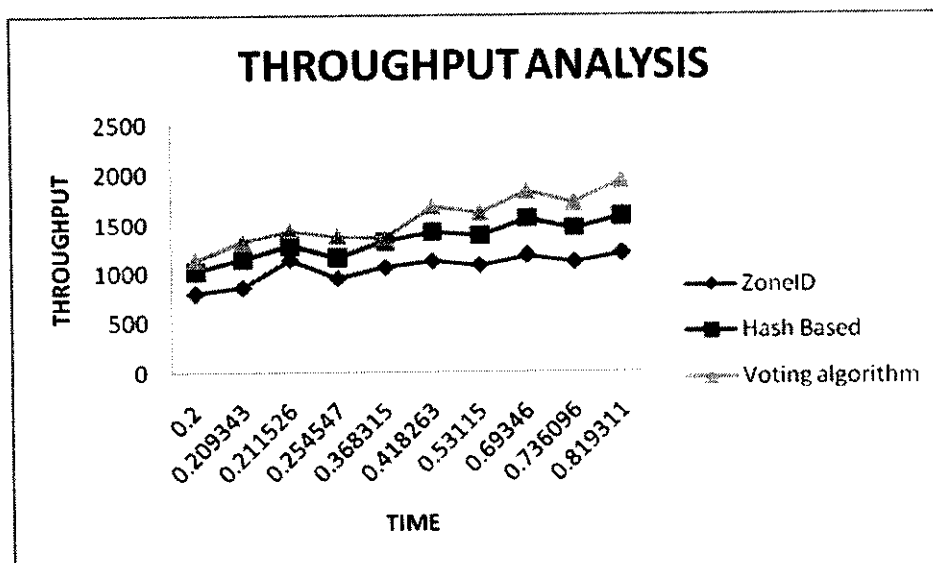
5. PERFORMANCE EVALUATION

5.1 Throughput analysis

The performances of the three algorithms namely zoneID based, hash based and voting algorithm are evaluated based on the throughput and the data loss.

The throughput values are calculated by extracting the values of the packet sizes from the trace file. The throughput value of a node at different time intervals are recorded such as 0-7 seconds for the zone ID based leader, 7-14 seconds for the hash based leader and after that by the voting algorithm based leader.

The throughput values of a particular node when packets are routed through the different leaders elected by different algorithms are plotted in a graph wherein the throughput are marked along the Y axis and the time scale is marked along the X axis



Inference:

From the graph it is inferred that the throughput increases by time as the mode of selection of leader changes the efficiency of the leader.

At first the leader is elected by nodeID based algorithm wherein no constraints are considered except that the node has the highest nodeID. So the leader elected through this method was less efficient compared to the other two.

Secondly the leader elected through hash based algorithm considered the constraints of the nodes location and its group identity along with a random hash function

$$H(\text{GID},d,\text{myloc})=\{x,y\} \text{ where } x,y \in \text{cell region}$$

and hence comparatively more efficient than the previous algorithm. The throughput values are found higher than the nodeID algorithm and lesser than the voting algorithm.

The third best algorithm is the voting algorithm which elects an efficient leader which has recorded the highest throughput of all the three in our analysis. It is based on the formula

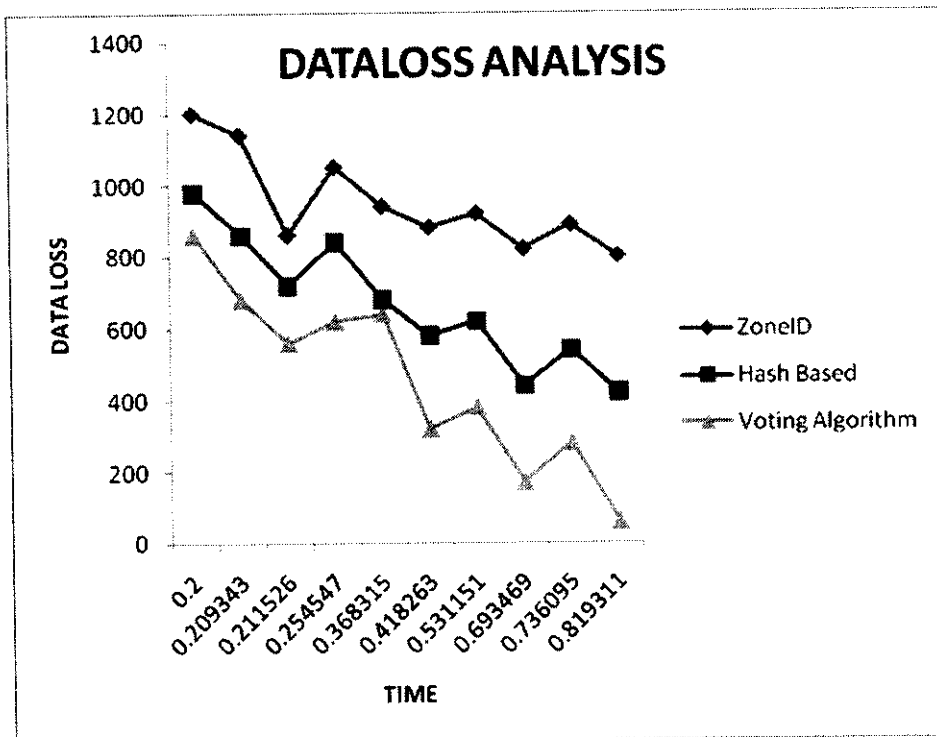
$$EF_i(t) = a_1 e^{-v_i(t)} + a_2 B_i(t) + a_3 (1 - E_i(t))$$

Due to higher battery power, lesser average speed and closer distance to the zone centre the leader elected stays in the zone for longer, routes data effectively and receives data packets efficiently.

5.2 Data loss analysis

The value of data loss that occurs in the packets that are transferred are extracted from the trace file and similar to the throughput analysis, the data loss

incurred on a node when three leaders are elected based on the three different algorithms namely zone ID based, hash based and voting algorithm are tabulated. The data loss is calculated by the difference in packet size of the sent and received packet. The Tabulated values are plotted in a graph.



Inference:

From the graph it is inferred that the data loss occurring in the nodes decreases as the efficiency of the leader increases owing to the changes in their mode of selection.

In the first method the leader is elected only based on its node ID and hence efficiency becomes lesser and data loss is heavy.

The hash based method is comparatively better because the leader is elected based on its position and group ID . this algorithm somehow returns a prominent value of a nodes position and a leader node is effectively functioning always and hence data loss becomes minimum.

The third and efficient method is the voting algorithm wherein the nodes transfer data packets efficiently because of their better battery power supporting competent data transfer, lesser average speed owing to their longer time of stay in the zone, and closer to the zone centre contributing to better routing of packets.

CONCLUSION

6. CONCLUSION

The system has developed a scalable and efficient service provisioning scheme for MANET. The scheme is adaptable to the changing topology and management requirements of the network. The system also accomplishes a scalable resource tracking, service membership management mechanism to support timely and coordinative service provisioning.

It has compared and analyzed the three algorithms for the service coordinator election and has arrived at the conclusion justifying the efficiency of the voting algorithm and the proficient routing of data packets through the leader elected by that algorithm. The justification is supported evidently by the performance evaluation of the throughput and data loss. The throughput has consistently increased and the data loss has completely decreased.

Thus the project has completely accomplished the design of an efficient service provisioning scheme and hence can comprehensively support data packet transfers through the dynamic adhoc environment.

FUTURE ENHANCEMENTS

7. FUTURE ENHANCEMENTS

The system is more scalable in terms of number of multicast groups compared to other protocols. Still the future research can be made in the following directions:

- (a) More schemes for service discovery to be considered supporting better resource tracking.
- (b) Intend to consider more characteristics while implementing the voting algorithm to elect an efficient leader.
- (c) Intend to compare the effect of group size, effect of packet size ,scalability issues and delay characteristics with other multicast protocols and service provisioning schemes.
- (d) Intend to maintain a proper group membership managemt:
- (e) Intend the service provisioning scheme with an efficient multicast tree construction

APPENDICES

8. APPENDICES

8.1 Source code

```
set val(chan)      Channel/WirelessChannel    ;#Channel Type
set val(prop)      Propagation/TwoRayGround  ;# radio-propagation model
set val(netif)     Phy/WirelessPhy          ;# network interface type
set val(mac)       Mac/802_11               ;# MAC type
set val(ifq)       Queue/DropTail/PriQueue   ;# interface queue type
set val(ll)        LL                       ;# link layer type
set val(ant)       Antenna/OmniAntenna      ;# antenna model
set val(ifqlen)    50                      ;# max packet in ifq
set val(nn)        16                      ;# number of mobilenodes
set val(rp)        AODV                    ;# routing protocol
set val(x)         1500
set val(y)         1000
set val(seed)      2.0
```

```
# Initialize Global Variables
```

```
set ns_            [new Simulator]
```

```
set tracefd       [open tree.tr w]
```

```
$ns_ trace-all $tracefd
```

```
set namtrace [open tree.nam w]
```

```
$ns_ namtrace-all-wireless $namtrace $val(x) $val(y)
```

```
# set up topography object
```

```
set topo          [new Topography]
```

```
$topo load_flatgrid $val(x) $val(y)
```



```

# Create func1
create-god $val(nn)
$ns_ color 0 red

# Create channel #1 and #2
set chan_1_ [new $val(chan)]
set chan_2_ [new $val(chan)]
# configure node, please note the change below.
$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) \
    -topoInstance $topo \
    -agentTrace ON \
    -routerTrace ON \
    -macTrace ON \
    -movementTrace OFF \
    -channel $chan_1_

$ns_ node-config \        channel $chan_2_
#~~~~~NODE
DECLARATION~~~~~

```

```
# creation of nodes
```

```
for { set i 0 } { $i < $val(nn) } { incr i } {
```

```
  global n
```

```
  set n($i) [$ns_ node]
```

```
}
```

```
# To set the random motion
```

```
for { set i 0 } { $i < $val(nn) } { incr i } {
```

```
  $n($i) random-motion 0
```

```
}
```

```
# To set the size of the nodes
```

```
for { set i 0 } { $i < $val(nn) } { incr i } {
```

```
  $ns_ initial_node_pos $n($i) 60
```

```
}
```

```
$n(1) color "green"
```

```
# Provide initial (X,Y, for now Z=0) co-ordinates for mobilenode
```

```
$n(0) set X_ 600.0
```

```
$n(0) set Y_ 900.0
```

```
$n(0) set Z_ 0.0
```

\$n(1) set X_ 250.0

\$n(1) set Y_ 450.0

\$n(1) set Z_ 0.0

\$n(2) set X_ 270.0

\$n(2) set Y_ 650.0

\$n(2) set Z_ 0.0

\$n(3) set X_ 150.0

\$n(3) set Y_ 520.0

\$n(3) set Z_ 0.0

\$n(4) set X_ 200.0

\$n(4) set Y_ 350.0

\$n(4) set Z_ 0.0

\$n(5) set X_ 350.0

\$n(5) set Y_ 475.0

\$n(5) set Z_ 0.0

\$n(6) set X_ 750.0

\$n(6) set Y_ 450.0

\$n(6) set Z_ 0.0

store the initial positions of the nodes

set x(0) 600

set y(0) 900

set x(1) 250

set y(1) 450

set x(2) 270

set y(2) 650

set x(3) 150

set y(3) 520

set x(4) 200

set y(4) 350

set x(5) 350

set y(5) 475

set x(6) 750

set y(6) 450

Now produce some simple node movements

```
puts "-----"  
puts "          SERVICE COORDINATION IN MANET "  
puts "-----"
```

\$ns_ at 0.0 "\$n(0) setdest 605.0 905.0 30.0"

\$ns_ at 0.0 "\$n(1) setdest 255.0 455.0 30.0"

\$ns_ at 0.0 "\$n(2) setdest 275.0 655.0 30.0"

```
$ns_ at 0.0 "$n(3) setdest 155.0 525.0 30.0"  
$ns_ at 0.0 "$n(4) setdest 205.0 355.0 30.0"  
$ns_ at 0.0 "$n(5) setdest 355.0 485.0 30.0"  
$ns_ at 0.0 "$n(6) setdest 755.0 455.0 30.0"
```

```
set ser(1) 1  
set ser(2) 1  
set ser(3) 1  
set ser(4) 1  
set ser(5) 1
```

```
puts " Node0 is elected as Regional co-ordinator "  
puts " Services offered "  
puts " NODE    SERVICE "  
for {set i 1} {$i<$val(nn)} {incr i} {  
    puts "$i    $ser($i) "  
}
```

```
for {set i 1} {$i < $val(nn)} { incr i} {  
    set gz [ expr $x($i) * 2 ]  
    if { $gz < 1000 } {  
        set grp($i) 1  
    }  
    if { $gz > 1000 && $gz < 2000 } {  
        set grp($i) 2  
    }  
    if { $gz > 2000 } {
```

```
set grp($i) 3
```

```
}
```

```
}
```

```
puts " LEADER ELECTION: 3 METHODS "
```

```
puts " -> HASH BASED LEADER ELECTION "
```

```
puts " -> ZONE ID BASED LEADER ELECTION "
```

```
puts " -> VOTING ALGORITHM BASED LEADER ELECTION "
```

```
#~~~~~LEADER ELECTION BASED ON HASH ID AND  
DATA TRANSMISSION~~~~~
```

```
proc hasselect1 { i1 i2 i3 i4 i5 i6 i7 i8 } {
```

```
puts "-----"
```

```
puts "    HASH FUNCTION BASED LEADER ELECTION    "
```

```
puts "-----"
```

```
global ns_
```

```
global n
```

```
global x
```

```
global y
```

```
global grp
```

```
for { set i 1 } { $i < 6 } { incr i } {
```

```
    set x1 $x($i)
```

```
    set y1 $y($i)
```

```
    set d 3
```

```
    set has($i) [ expr ($x1 * 2) + ($y1 * 2) + $grp($i) + $d ]
```

```

}
if { $has(1) > $has(2) && $has(1) > $has(3) && $has(1) > $has(4) && $has(1) >
$has(5) } {
    set p1 1
}
if { $has(2) > $has(1) && $has(2) > $has(3) && $has(2) > $has(4) && $has(2) >
$has(5) } {
    set p1 2
}
if { $has(3) > $has(1) && $has(3) > $has(2) && $has(3) > $has(4) && $has(3)
> $has(5) } {
    set p1 3
}
if { $has(4) > $has(1) && $has(4) > $has(2) && $has(4) > $has(3) && $has(4)
> $has(5) } {
    set p1 4
}
if { $has(5) > $has(1) && $has(5) > $has(2) && $has(5) > $has(3) && $has(5)
> $has(4) } {
    set p1 5
}

if { $p1==1 } {
puts " LEADER    ZONE "
puts " 1        1 "
set k111 1
set l111 2

```

```
set m111 3
set z111 4
set o111 5
}
if { $p1==2 } {
puts " LEADER    ZONE "
puts " 2        1 "
set k111 2
set l111 1
set m111 3
set z111 4
set o111 5
}
if { $p1==3 } {
puts " LEADER    ZONE "
puts " 3        1 "
set k111 3
set l111 1
set m111 2
set z111 4
set o111 5
}
if { $p1==4 } {
puts " LEADER    ZONE "
puts " 4        1 "
set k111 4
set l111 1
```



```
set m111 2
set z111 3
set o111 5
}
if { $p1==5 } {
puts " LEADER   ZONE "
puts " 5       1 "
set k111 5
set l111 1
set m111 2
set z111 3
set o111 4
}

$ns_ initial_node_pos $n($k111) 120
set tcp1 [new Agent/TCP]
$tcp1 set class_ 1
set sink1 [new Agent/TCPSink]
$sink1 set class_ 1
$ns_ attach-agent $n($k111) $tcp1
$ns_ attach-agent $n($l111) $sink1
$ns_ connect $tcp1 $sink1
set ftp1 [new Application/FTP]
$ftp1 attach-agent $tcp1
$ns_ at $i1 "$ftp1 start"
$ns_ at $i2 "$ftp1 stop"
```

```
$ns_ at $i1 "$ns_ trace-annotate \"SERVICES ARE ADVERTISED TO NODE  
$l111 BY NODE $k111 \\""
```

```
#~~~~~LEADER ELECTION BASED ON NODE ID AND DATA  
TRANSMISSION~~~~~
```

```
proc elect1 { a1 a2 a3 a4 a5 d1 d2 d3 d4 d5 d6 d7 d8 } {  
  puts "-----"  
  puts "      ZONE ID BASED LEADER ELECTION      "  
  puts "-----"  
  global ns_  
  global n  
  global k11  
  set nid($a1) $n($a1)  
  set nid($a2) $n($a2)  
  set nid($a3) $n($a3)  
  set nid($a4) $n($a4)  
  set nid($a5) $n($a5)  
  
  if { $nid($a1) > $nid($a2) && $nid($a1) > $nid($a3) && $nid($a1) > $nid($a4)  
    && $nid($a1) > $nid($a5) } {  
    set p4 1  
  }  
  if { $nid($a2) > $nid($a1) && $nid($a2) > $nid($a3) && $nid($a2) > $nid($a4)  
    && $nid($a2) > $nid($a5) } {  
    set p4 2
```

```
}  
  if { $nid($a3) > $nid($a1) && $nid($a3) > $nid($a2) && $nid($a3) > $nid($a4)  
&& $nid($a3) > $nid($a5) } {  
    set p4 3  
  }  
  if { $nid($a4) > $nid($a1) && $nid($a4) > $nid($a2) && $nid($a4) > $nid($a3)  
&& $nid($a4) > $nid($a5) } {  
    set p4 4  
  }  
  if { $nid($a5) > $nid($a1) && $nid($a5) > $nid($a2) && $nid($a5) > $nid($a3)  
&& $nid($a5) > $nid($a4) } {  
    set p4 5  
  }
```

```
if { $p4==1 } {  
  puts " LEADER  ZONE "  
  puts " $a1      1 "  
  set k11 $a1  
  set l11 $a2  
  set m11 $a3  
  set z11 $a4  
  set o11 $a5  
}  
if { $p4==2 } {  
  puts " LEADER  ZONE "  
  puts " $a2      1 "
```

```
set k11 $a2
set l11 $a1
set m11 $a3
set z11 $a4
set o11 $a5
}
if { $p4==3 } {
puts " LEADER  ZONE "
puts " $a3      1 "
set k11 $a3
set l11 $a1
set m11 $a2
set z11 $a4
set o11 $a5
}
if { $p4==4 } {
puts " LEADER  ZONE "
puts " $a4      1 "
set k11 $a4
set l11 $a1
set m11 $a2
set z11 $a3
set o11 $a5
}
if { $p4==5 } {
puts " LEADER  ZONE "
puts " $a5      1 "
```

```
set k11 $a5
set l11 $a1
set m11 $a2
set z11 $a3
set o11 $a4
}
```

```
$ns_ initial_node_pos $n($k11) 120
```

```
set tcp1 [new Agent/TCP]
```

```
$tcp1 set class_ 1
```

```
set sink1 [new Agent/TCPSink]
```

```
$sink1 set class_ 1
```

```
$ns_ attach-agent $n($k11) $tcp1
```

```
$ns_ attach-agent $n($l11) $sink1
```

```
$ns_ connect $tcp1 $sink1
```

```
set ftp1 [new Application/FTP]
```

```
$ftp1 attach-agent $tcp1
```

```
$ns_ at $d1 "$ftp1 start"
```

```
$ns_ at $d2 "$ftp1 stop"
```

```
$ns_ at $d1 "$ns_ trace-annotate \"SERVICES ARE ADVERTISED TO NODE  
$l11 BY NODE $k11 \"/>
```

```
#~~~~~LEADER ELECTION BASED ON VOTING ALGORITHM  
AND DATA TRANSMISSION~~~~~
```

```
proc elect4 { aa ab ac ad ae t1 t11 t2 t22 t3 t33 t4 t44 } {
```

```
global ns_
```

```
global n
```

```
global k
```

```
puts "-----"
```

```
puts "          VOTING ALGORITHM BASED LEADER ELECTION          "
```

```
puts "-----"
```

```
set bp($aa) [ expr round ( 100 * ( srand([clock seconds])*7 ) ) ]
```

```
set bp($ab) [ expr round ( 120 * srand([clock seconds]*10) ) ]
```

```
set bp($ac) [ expr round ( 115 * srand([clock seconds]*16) ) ]
```

```
set bp($ad) [ expr round ( 120 * srand([clock seconds]*14) ) ]
```

```
set bp($ae) [ expr round ( 132 * srand([clock seconds]*5) ) ]
```

```
set av($aa) [ expr round ( 25 * srand([clock seconds]*12) ) ]
```

```
set av($ab) [ expr round ( 35 * srand([clock seconds]*5) ) ]
```

```
set av($ac) [ expr round ( 45 * srand([clock seconds]*13) ) ]
```

```
set av($ad) [ expr round ( 55 * srand([clock seconds]*14) ) ]
```

```
set av($ae) [ expr round ( 65 * srand([clock seconds]*9) ) ]
```

```
set s($aa) [expr $bp($aa) + $av($aa) ]
```

```
set s($ab) [expr $bp($ab) + $av($ab) ]
```

```
set s($ac) [expr $bp($ac) + $av($ac) ]
```

```
set s($ad) [expr $bp($ad) + $av($ad) ]
```

```
set s($ae) [expr $bp($ae) + $av($ae) ]
```

```
if { $s($aa) > $s($ab) && $s($aa) > $s($ac) && $s($aa) > $s($ad) && $s($aa) > $s($ae) } {
```

```
set p7 1
```

```
}
```

```
if { $s($ab) > $s($aa) && $s($ab) > $s($ac) && $s($ab) > $s($ad) && $s($ab) > $s($ae) } {
```

```
set p7 2
```

```

}
if { $s($ac) > $s($aa) && $s($ac) > $s($ab) && $s($ac) > $s($ad) && $s($ac) >
$s($ae) } {
set p7 3
}
if { $s($ad) > $s($aa) && $s($ad) > $s($ab) && $s($ad) > $s($ac) && $s($ad) >
$s($ae) } {
set p7 4
}
if { $s($ae) > $s($aa) && $s($ae) > $s($ab) && $s($ae) > $s($ac) && $s($ae) >
$s($ad) } {
set p7 5
}

if { $p7==1 } {
puts " LEADER    ZONE "
puts " $aa      1 "
set k $aa
set l $ab
set m $ac
set z $ad
set o $ae
}
if { $p7==2 } {
puts " LEADER    ZONE "
puts " $ab      1 "
set k $ab

```

```
set l $aa
set m $ac
set z $ad
set o $ae
}
if { $p7==3 } {
puts " LEADER  ZONE "
puts " $ac      1 "
set k $ac
set l $aa
set m $ab
set z $ad
set o $ae
}
if { $p7==4 } {
puts " LEADER  ZONE "
puts " $ad      1 "
set k $ad
set l $aa
set m $ab
set z $ac
set o $ae
}
if { $p7==5 } {
puts " LEADER  ZONE "
puts " $ae      1 "
set k $ae
```



```

set l $aa
set m $ab
set z $ac
set o $ad
}
set tcp1 [new Agent/TCP]
$tcp1 set class_ 1
set sink1 [new Agent/TCPSink]
$sink1 set class_ 1
$ns_ attach-agent $n($k) $tcp1
$ns_ attach-agent $n($l) $sink1
$ns_ connect $tcp1 $sink1
set ftp1 [new Application/FTP]
$ftp1 attach-agent $tcp1
$ns_ at $t1 "$ftp1 start"
$ns_ at $t11 "$ftp1 stop"
$ns_ at $t1 "$ns_ trace-annotate \"SERVICES ARE ADVERTISED TO NODE $l
BY NODE $k \""

```

~~~~~~~SERVICE PROVISIONING~~~~~~~

```

proc service { h1 } {
global ns_
global n
global a l1
global k l1
global u l1
global a

```

```

global k
global u
set flag 0
puts "-----"
puts "          SERVICE PROVISIONING"
puts "-----"
puts " Enter the service requestor "
gets stdin sr
puts " Enter the service required "
gets stdin serv

if { $h1 >= 14.0 && $h1 < 21.0 } {
  if { $sr == 1 || $sr == 2 || $sr == 3 || $sr == 4 || $sr == 8 } {
    set lead $k
  }
  if { $sr == 5 || $sr == 6 || $sr == 7 || $sr == 9 || $sr == 13 } {
    set lead $a
  }
  if { $sr == 10 || $sr == 11 || $sr == 12 || $sr == 14 || $sr == 15 } {
    set lead $u
  }
  set sp(1) 4
  set sp(2) 5
  set sp(3) 4
  set sp(4) 2
  set sp(5) 6
  set sp(6) 9
}

```

```
set sp(7) 5
set sp(8) 6
set sp(9) 7
set sp(10) 11
set sp(11) 13
set sp(12) 11
set sp(13) 14
set sp(14) 12
set sp(15) 14
if {$serv==1 && $sr==1 } {
  set flag 2
}
if {$serv==1 && $sr==2 } {
  set flag 2
}
if {$serv==1 && $sr==3 } {
  set flag 2
}
if {$serv==1 && $sr==4 } {
  set flag 2
}
if {$serv==1 && $sr==8 } {
  set flag 2
}
if { $serv==2 && $sr==5 } {
  set flag 2
}
}
```

```
if { $serv==2 && $sr==6 } {  
  set flag 2  
}  
if { $serv==2 && $sr==7 } {  
  set flag 2  
}  
if { $serv==2 && $sr==9 } {  
  set flag 2  
}  
if { $serv==2 && $sr==13 } {  
  set flag 2  
}  
if { $serv==3 && $sr==10 } {  
  set flag 2  
}  
if { $serv==3 && $sr==11 } {  
  set flag 2  
}  
if { $serv==3 && $sr==12 } {  
  set flag 2  
}  
if { $serv==3 && $sr==14 } {  
  set flag 2  
}  
if { $serv==3 && $sr==15 } {  
  set flag 2  
}
```

```
if {$flag==2} {  
  puts "Service Requestor:$sr"  
  puts "Service:$serv"  
  puts "Service Provider:$sp($sr)"  
  
  set tcp17 [new Agent/TCP]  
  $tcp17 set class_ 1  
  set sink17 [new Agent/TCPSink]  
  $ns_ attach-agent $n($serv) $tcp17  
  $ns_ attach-agent $n($lead) $sink17  
  $ns_ connect $tcp17 $sink17  
  set ftp17 [new Application/FTP]  
  $ftp17 attach-agent $tcp17  
  $ns_ at $h1 "$ftp17 start"  
  $ns_ at [expr $h1 + 0.5 ] "$ftp17 stop"  
  $ns_ at $h1 "$ns_ trace-annotate \"SERVICE$serv IS REQUESTED BY THE  
  NODE $sr \""
```

```
set tcp18 [new Agent/TCP]  
$tcp18 set class_ 1  
set sink18 [new Agent/TCPSink]  
$ns_ attach-agent $n($lead) $tcp18  
$ns_ attach-agent $n($sp($sr)) $sink18  
$ns_ connect $tcp18 $sink18  
set ftp18 [new Application/FTP]
```

```
$ftp18 attach-agent $tcp18
$ns_ at [expr $h1 + 0.55] "$ftp18 start"
$ns_ at [expr $h1 + 0.75] "$ftp18 stop"
$ns_ at [expr $h1 + 0.55] "$ns_ trace-annotate \"SERVICE PROVIDER$sp($sr) IS
QUERIED FOR SERVICE AVAILABILITY \""
if {$flag==0} {
if { $serv==1 } {
    set ap2 $k
    set spp2 4
}
if { $serv==2 } {
    set ap2 $a
    set spp2 7
}
if { $serv==3 } {
    set ap2 $u
    set spp2 11
}
puts "Service Requestor:$sr"
puts "Service:$serv"
puts "Service Provider:$spp2"
set tcp17 [new Agent/TCP]
$tcp17 set class_ 1
set sink17 [new Agent/TCPSink]
$ns_ attach-agent $n($sr) $tcp17
$ns_ attach-agent $n($lead) $sink17
$ns_ connect $tcp17 $sink17
```

```
set ftp17 [new Application/FTP]
$ftp17 attach-agent $tcp17
$ns_ at $h1 "$ftp17 start"
$ns_ at [expr $h1 + 0.5 ] "$ftp17 stop"
$ns_ at $h1 "$ns_ trace-annotate \"SERVICE$serv IS REQUESTED BY THE
NODE $sr \""
```

```
set tcp18 [new Agent/TCP]
$tcp18 set class_ 1
set sink18 [new Agent/TCPSink]
$ns_ attach-agent $n($lead) $tcp18
$ns_ attach-agent $n($ap2) $sink18
$ns_ connect $tcp18 $sink18
set ftp18 [new Application/FTP]
$ftp18 attach-agent $tcp18
$ns_ at [expr $h1 + 0.55] "$ftp18 start"
$ns_ at [expr $h1 + 0.75] "$ftp18 stop"
$ns_ at [expr $h1 + 0.55] "$ns_ trace-annotate \" QUERYING FOR SERVICE TO
THE LEADER OF THE SERVICE PROVIDER ZONE \""
```

```
proc hashpre {} {
global x
global y
global grp
for { set i 1 } {$i < 16 } {incr i} {
set x1 $x($i)
set y1 $y($i)
```

```

set d 3

set has($i) [ expr ($x1 * 2) + $grp($i) + $d ]
}

puts "[ expr $i - 1] is elected as regional co-ordinator"
}

# Tell nodes when the simulation ends
for {set i 0} {$i < $val(nn)} {incr i} {
    $ns_ at 21.05 "$n($i) reset";
}

for {set r 1} {$val($r) < 6} {incr r} {
    $ns_ at 0.2 "hashelect1 0.2 0.28 0.29 0.34 0.35 0.42 0.43 0.5 "
    $ns_ at 7.0 "elect1 1 2 3 4 5 7.2 7.28 7.29 7.34 7.35 7.38 7.39 7.5 "
    $ns_ at 14.0 "elect5 5 6 7 9 13 14.1 14.3 14.35 14.4 14.5 14.6 14.9 15.2 "
}

$ns_ at 8.2 "service 8.3 "
$ns_ at 18.0 "service 18.5 "
$ns_ at 25.02 "stop"
$ns_ at 9.0 "hashpre "
$ns_ at 25.03 "puts \"NS EXITING...\"; $ns_ halt"

proc stop {} {
    global ns_ tracefd
    $ns_ flush-trace
    close $tracefd
    puts "running nam..."
    exec nam tree &
}

```

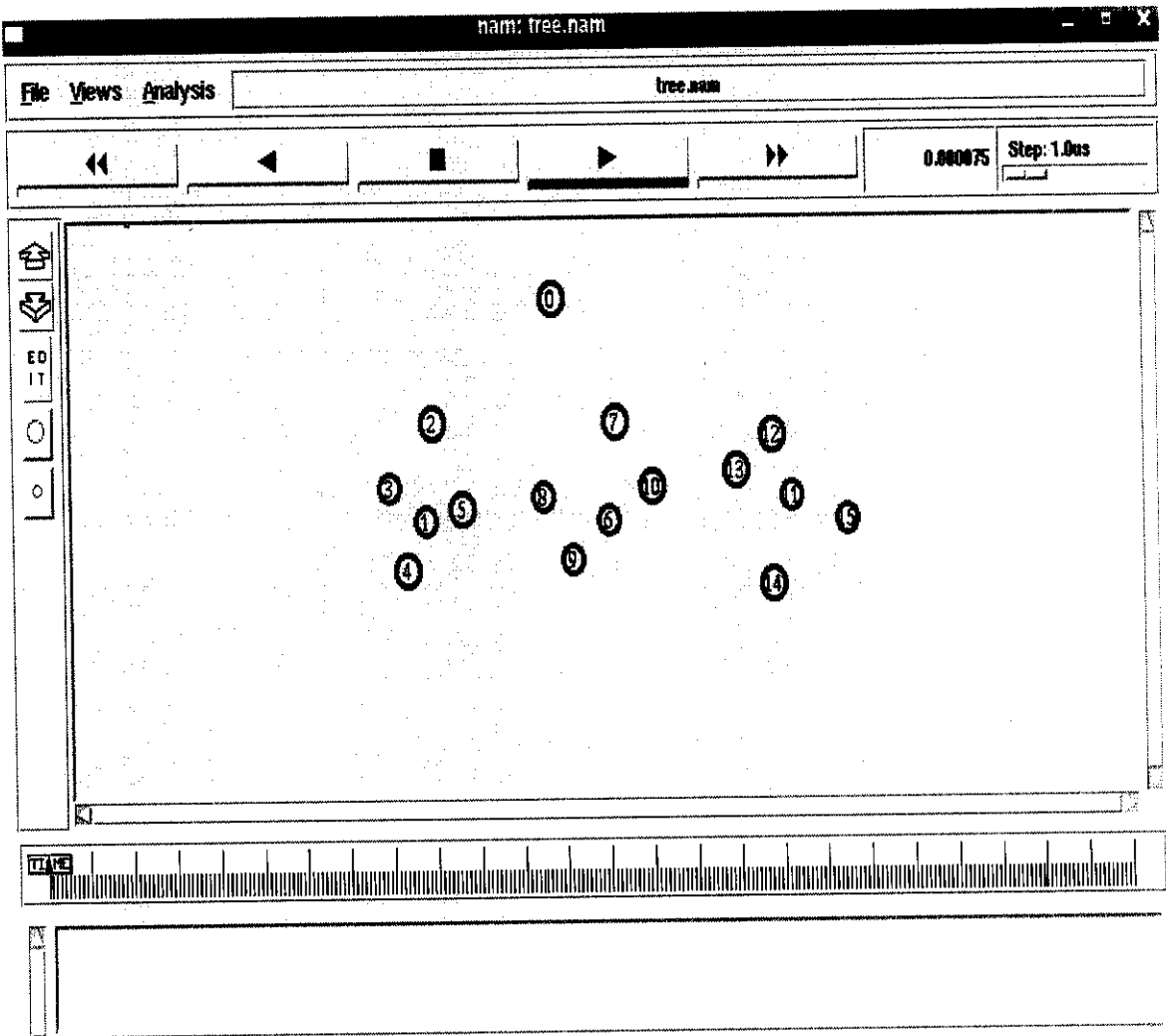


puts "Starting Simulation..."

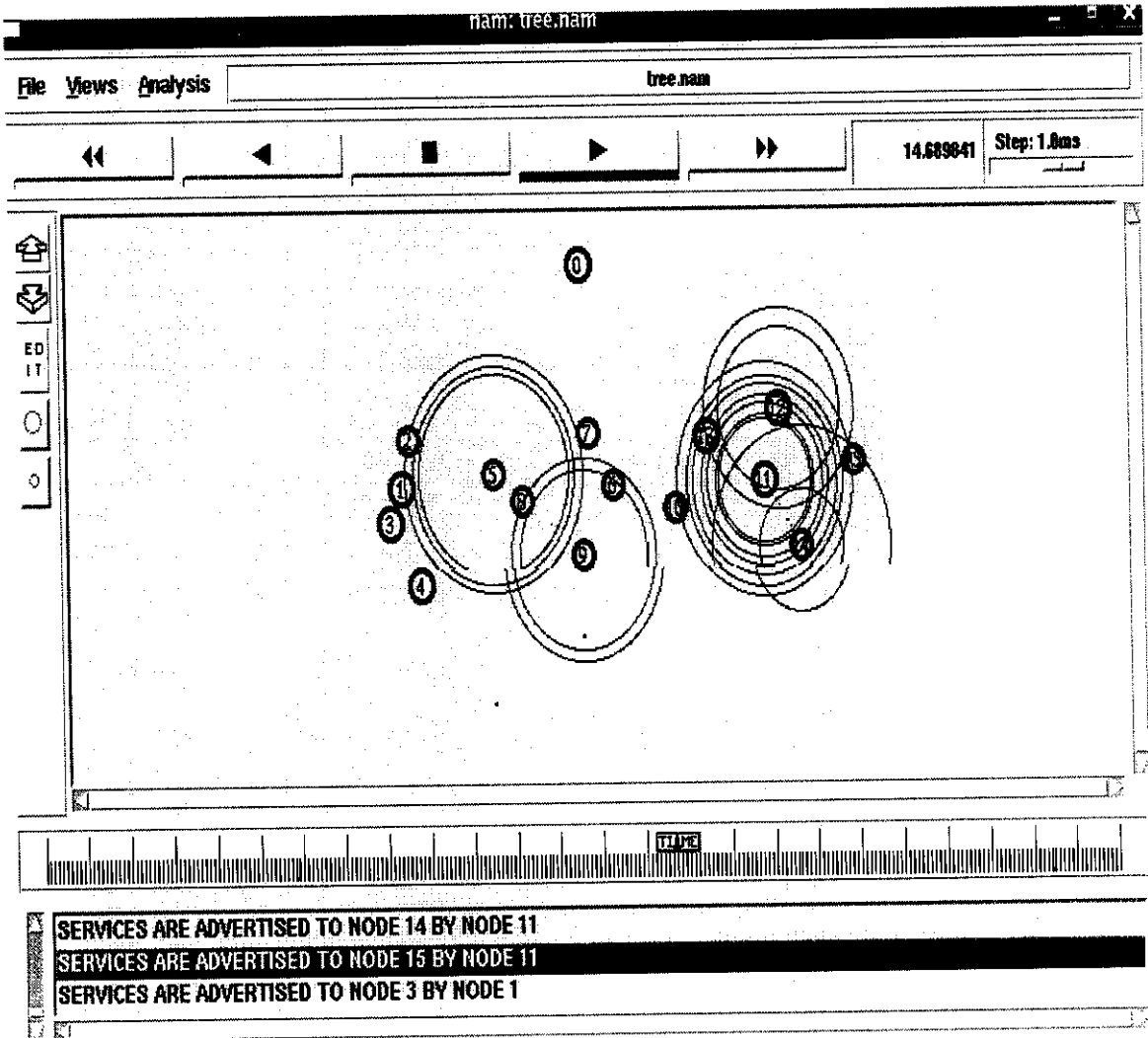
\$ns run

## 8.2 Snap shots

### INITIAL POSITION OF THE NODES



# SERVICE ADVERTISEMENT



# SERVICE REQUEST AND QUERYING

nam: tree.nam

File Views Analysis tree.nam

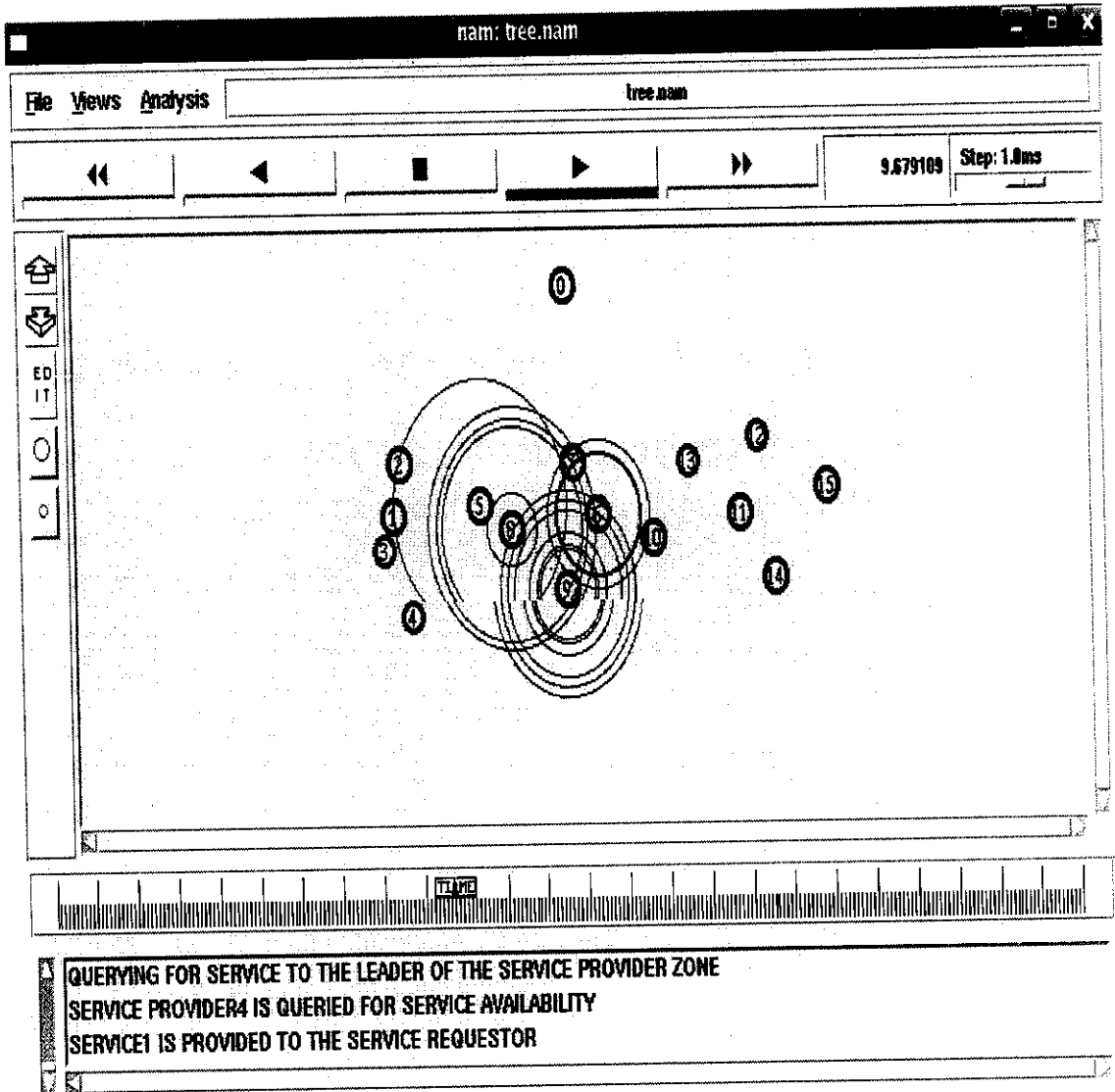
9.077109 Step: 1.0ms

ED IT

TIME

SERVICE1 IS REQUESTED BY THE NODE 6  
QUERYING FOR SERVICE TO THE LEADER OF THE SERVICE PROVIDER ZONE  
SERVICE PROVIDER4 IS QUERIED FOR SERVICE AVAILABILITY

# SERVICE DELIVERY:



## ***REFERENCES***

---

## 9. REFERENCES

- [1] Xiaojing Xiang and Xin Wang ‘ A Scalable Geographic Service provision framework for Mobile Adhoc Networks ‘ in PerCom ’07 IEEE,2007
- [2] Guo Song ,Gerard Parr and Bryan Scotney‘ A conceptual Framework for bandwidth Protection in Mobile Adhoc Networks ‘ in MOBICOM IEEE,2006
- [3] Saumitra M.Das,Himabindu Pucha and Y.Charlie Hu ‘Distributed Hashing for Scalable Multicast in Wireless Adhoc Networks’ in WowMom’06 IEEE,2006
- [4] Roy Friedman and Gabriel Kliot ‘Location Services in Wireless AdHoc and Hybrid Networks: A survey’ in Technion conference, 2006
- [5] Shuhui Yang and Jie Wu ‘New Technologies of Multicasting in MANET ‘ in IEEE,2004
- [6] Li j ‘ A scalable location service for geographic adhoc routing’ in ACM/IEEE MOBICOM, August 2000
- [7] Cheng L. ‘Service advertisement and discovery in mobile adhoc networks’ in Proc of CSCW, November 2002
- [8] Xiaojing Xiang and Xin Wang ‘An efficient Geographic Multicast Protocol for Mobile Adhoc Networks’ in WoWMoM’06 IEEE,2006