SCALABLE TEAM MULTICAST IN AD HOC NETWORKS

Supervised by
Dr. Tauseef-ur-Rehman

Developed by
Afsheen Khurram
Shafaq Naz

Department of Computer Science,
International Islamic University, Islamabad.
(2004)

بسم الله الرحمن الرحيم

International Islamic University Islamabad

Department of Computer Science

Date: ________________

Final Approval

It is certified that we have read the project report submitted by Shafaq Naz (91-CS/MS/02) and Afsheen Khurram (84-CS/MS/02) and it is our judgement that this project is of sufficient standard to warrant its acceptance by the International Islamic University, Islamabad for the MS Degree in Computer Science.

External Examiner
Dr. Qasim Rind
Ex Director General
Establishment Division
Islamabad

[illegible] ________________

Internal Examiner
Dr. Sikander Hayat Khiyal
Head,
DES, IIU,
Islamabad

[illegible] ________________

Supervisor
Dr. Tauseef-ur-Rehman
Head,
Department of Telecommunication Engineering,
International Islamic University,
Islamabad, Pakistan

[illegible] ________________

A dissertation submitted to the

DEPARTMENT OF COMPUTER SCIENCE

INTERNATIONAL ISLAMIC UNIVERITY, ISLAMABAD

as a partial fulfillment of the requirements

for the award of the degree of

Master of Science.

Dedicated to

Allah Almighty

Last Holy Prophet Muhammad (SAW)

And

Our Parents

who are very kind to us.

DECLARATION

We hereby declare that this project report, neither as a whole nor as a part thereof has been copied out from any source. It is further declared that we have completed our work on the basis of our personal efforts and under the sincere guidance of our teachers. If any part of our work is proved to be copied out from any source or found to be reproduction of someone else, we shall standby the consequences. No portion of the work presented in this report has been submitted in support of any application for any other degree or qualification of this or any other university or institute of learning.

Shafaq Naz
91-CS/MS/02
Afsheen Khurram
84-CS/MS/02

ACKNOWLEDGEMENT

We would like to take this opportunity to pay our humble gratitude to Almighty Allah, Who enabled us to complete this project. We are very thankful to our supervisor, Dr. Tauseef-ur-Rehman who provided full assistance in our work

Afsheen Khurram
Shafaq Naz

PROJECT IN BRIEF

Project Title:	Scalable Team Multicast in Mobile Ad hoc Networks
Undertaken By:	Shafaq Naz and Afsheen Khurram
Supervised by:	Prof. Dr. Tauseef-ur-Rehman Head of Telecommunication Faculty of Applied Sciences International Islamic University, Islamabad.
Tools Used:	Network Simulator 2 Rational Rose 2002
Operating System:	Windows 2000
System Used:	Intel Pentium III
Date Started:	2003
Date Completed:	May 30th, 2004

ABSTRACT

A new model for team multicast for large scale mobile ad hoc networks is proposed. This project exploits Team Multicast and Team Dynamics. In Team Multicast the nodes which share common interest belong to one team. All members in a team participate in same multicast group. Our project includes Motion Affinity in which Team members which share common interest have coordinated motion. In this model each team is treated as a logical subnet .In each logical subnet a node called Landmark Node is dynamically elected and landmark node is the representative of the team. The address of the elected landmark and there paths are propagated throughout the whole network. In Mobile Ad hoc Networks large membership size and network size leads to the problem of scalability. Our proposed M-LANMAR is a new protocol designed for large scalable MANETS which is based on pre-existing LANMAR.

LIST OF CONTENTS

CHAPTER 1

1. INTRODUCTION

Wireless access networks are rapidly becoming a part of our everyday life. The widespread availability of miniature wireless devices such as PDAs, cellular phones, Pocket PCs, small fixtures on buildings, and sensors are one step towards making the vision of anywhere, anytime pervasive access and computing a reality. But we are still a long way off from the goal of seamless wireless operation where any wireless device would be able to connect to any other wire line/wireless device at any time, in any place, and while satisfying the requirements of the user of the device. An important area that has to be focused on to make this vision a reality is that related to Ad hoc networks.

Technology under development for wireless Ad hoc networks is making important steps toward this end goal possible. Wireless ad-hoc networks usually cannot depend on traditional infrastructure found in enterprise environments such as dependable power sources, high bandwidth, continuous connectivity, common network services, well-known membership, static configuration, system administration, and physical security. Finally, a very interesting and challenging problem arises. Wireless ad-hoc networks will remain on the drawing board even if the other problems associated with them are solved. A wireless ad hoc network is a collection of two or more devices/ nodes or terminals with wireless communications and networking capability that communicate with each other without the aid of any centralized administrator. The network topology is in general dynamic, because the connectivity among the nodes may vary with time due to node mobility, node departures and new node arrivals. Hence, there is a need for efficient routing protocols to allow the nodes to communicate.

The importance in mobile ad hoc networking is to support robust and efficient operation in mobile wireless networks by incorporating routing functionality into mobile nodes. Such networks are forecasted to have dynamic, sometimes rapidly changing, random, multihop topologies, which are likely composed of relatively bandwidth-constrained wireless links. The set of applications for MANETs is diverse, ranging from small, static networks that are constrained by power sources, to large-scale, mobile, highly dynamic networks. The design of network protocols for these networks is a complex issue

To solve this problem, a new model for team multicast for large scale mobile ad hoc networks is proposed in this thesis. This model exploits Team Multicast and Team Dynamics. In Mobile Ad-hoc Networks (MANETS) large membership size and network size leads to the problem of scalability. Our proposed M-LANMAR is a new protocol designed for large scalable MANETS which is based on pre-existing LANMAR protocol.

1.1 Mobile Ad-hoc Networks (MANET)

One particularly challenging environment for multicast is a mobile ad-hoc network (MANET). A MANET consists of a dynamic collection of nodes with sometimes rapidly changing Multi-hop topologies that are composed of relatively low-bandwidth wireless links. Since each node has a limited transmission range, not all messages may reach all the intended hosts. To provide communication through the whole network, a source-to-destination path could pass through several intermediate neighbor nodes. Unlike typical wire line routing protocols, Ad-hoc routing protocols must address a diverse range of the network topology that can change randomly and rapidly, at unpredictable times. Since wireless links generally have lower capacity, congestion is typically the norm rather than the exception. The majority of nodes will rely on batteries, thus routing protocols must limit the amount of control information that is passed between nodes. This is for consideration of energy efficiency. The majority of applications for the MANET technology are in areas where rapid deployment and dynamic reconfiguration are necessary and the wire line network is not available. These include military battlefields, emergency search and rescue sites, classrooms, and conventions where participants share information dynamically using their mobile devices. These applications lend themselves well to multicast operation. In addition, within a wireless medium, it is even more crucial to reduce the transmission overhead and power consumption. Multicasting can improve the efficiency of the wireless link when sending multiple copies of messages by exploiting the inherent broadcast property of wireless transmission.

1.1.1 Characteristics of Ad hoc networks

The silent characteristics of Ad hoc networks are:

- **Dynamic Topologies:** The rapid and unpredictable movement of the nodes and fast-changing propagation conditions, network information leads to the frequent network reconfigurations and frequent exchanges of control information over the wireless medium.
- **Asymmetric link characteristics:** In a wireless environment, communication between two nodes may not work equally well in both directions. In other words if a node is in transmission range of another node, the reverse may not be true.
- **Multihop communications:** Each node in the Ad hoc network will act as a transmitter, a receiver, or a relay station. So packets from a transmitter node (source) may reach the receiver node (destination) in multiple hops through several intermediate relay nodes.
- **Decentralized operation:** The Ad hoc networks need not to rely on preexisting infrastructure or centralized control. In Ad hoc networks, since there is no preexisting infrastructure, the centralized entities (e.g. BSs, MSCs and the HLR in cellular networks) do not exist. Thus the lack of these entities in Ad hoc networks requires more sophisticated distributed algorithms to perform equivalent functions.
- **Bandwidth-constrained variable-capacity links:** Wireless links will continue to have lower capacity than their hardwired counter parts. Throughput of wireless communications is often much les because of the effects of multiple access, fading, noise and interference conditions.
- **Energy-constrained operations:** In Ad hoc networks mobile nodes rely on batteries or other exhaustible means for their energy.

Characteristics of Ad hoc networks create a set of performance concerns for protocol design that extend beyond those guiding the design of protocols for conventional networks with preconfigured topology [1].

1.1.2 Affinity Team Model

In MANETS nodes are often organized in teams with different tasks corresponding different functional and operational characteristics. Nodes in same team have coordinated motion. This model is referred as Affinity Team Model. Few assumptions about Team Multicast are made that Team is based on some nodes that have common interest and thus all the members in team participate in same multicast group. As multicast dynamics are on a per team basis so the entire team joins or leaves from a multicast group. Depending upon different needs the two or more teams merge into one or they split up into sub-teams.

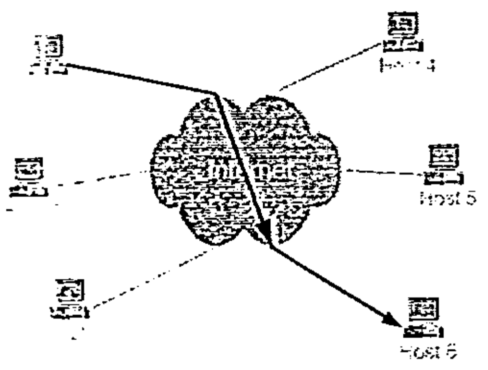
1.2 The concept of Unicast, Broadcast and Multicast

There are three main types of transmissions Unicast, Broadcast and Multicast transmission.

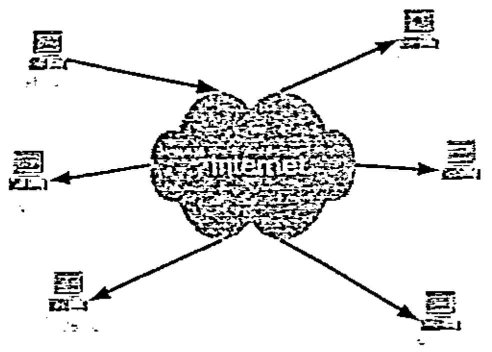
- **Unicast:** In unicast routing, traffic is routed through the network along a single path from the source to the destination host. A unicast router does not really care about the source address—only the destination address and how to forward the traffic towards that destination. The router scans through its routing table and then forwards a single copy of the unicast packet out the correct interface in the direction of the destination

- **Broadcast:** In broadcast routing, traffic is routed along all paths from the source to all connected hosts throughout the network. In a Broadcast application one host sends and all other connected hosts receive.

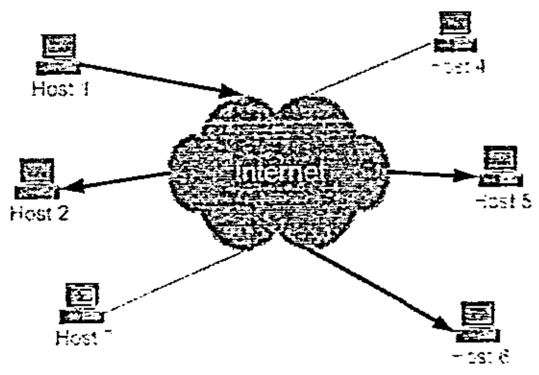
- **Multicast:** In multicast routing, the source is sending traffic to an arbitrary group of hosts that are represented by a multicast group address. The multicast router must determine which direction is upstream (towards the source) and which direction (or directions) is downstream. If there are multiple downstream paths the router will replicate the packet and forward it down the appropriate downstream paths, which is not necessarily all paths.



(a) Unicast



(b) Broadcast



(c) Multicast

Figure 1.1: (a) Unicast, (b) Broadcast, (c) Multicast

1.3 Multicasting

Multicasting is the transmission of datagram to a group of hosts identified by a single destination address. Multicasting is intended for group-oriented computing. The multicast service is critical in applications characterized by the close collaboration of teams (e.g. rescue patrol, battalion, scientists, etc) with requirements for audio and video conferencing and sharing of text and images. The use of multicasting within a network has many benefits. Multicasting reduces the communication costs for applications that send the same data to multiple recipients. Instead of sending via multiple unicast, multicasting minimizes the link bandwidth consumption, sender and router processing, and delivery delay. Maintaining group membership information and building optimal multicast trees is challenging even in wired networks. However, nodes are increasingly mobile.

1.3.1 Why Multicast?

Multicast is an efficient way of delivering one-to-many communications. Its benefits over the unicast delivery paradigm for this type of communication are well-chronicled. The explosive growth of multimedia content on the Internet has highlighted the need for a ubiquitous wide-scale deployment of native multicast across the Internet. In the example of Internet radio, unicast requires that each listener must make a separate connection to the server that is the source of the data. This results in tremendous load on the server and congestion across expensive WAN links as the number of listeners increases. With multicast, one stream is sent by the server to the network and a distribution tree forms. Interested listeners simply add a branch to the tree. Routers replicate packets at each branch in the tree. In this way, no packets are ever duplicated in the network, and the server never has to send more than one stream of data.

It is no longer inconceivable to predict that all of television and radio will eventually be delivered primarily over the Internet someday. Accepting that, it cannot be denied that the unicast method of delivery simply cannot scale to support this vision. Conversely, multicast is designed explicitly to provide this functionality. Also, multicast's benefits do not end with audio and video applications. File transfer, network management, stock tickers and any other application that requires one-to-many delivery is ideal for multicast.

During the 80s and early 90s, the multicast world had been confined to a tunneled overlay network of routers and UNIX servers known as the MBone. It was primarily used by research institutions as a hobbyist toy. However, with the recent creation and standardization of some protocols, along with the willingness of some service providers to provide a scalable architecture, a ubiquitous wide-scale deployment of non-tunneled (native) multicast across the

Internet is a reality today. The protocols used to do this are not solely designed for the enterprise networks of small- and medium-sized businesses. These protocols are available and in use today and will scale to support native deployments across the entire Internet.

1.3.2 Multicast Group Concept

Multicast is based on the concept of a group. An arbitrary group of receivers expresses an interest in receiving a particular data stream. This group does not have any physical or geographical boundaries—the hosts can be located anywhere on the Internet. Hosts that are interested in receiving data flowing to a particular group must join the group using Internet Group Management Protocol (IGMP). Hosts must be a member of the group to receive the data stream.

1.3.3 IP Multicast Addresses

IP Multicast addresses specify a "set" of IP hosts that have joined a group and wish to receive traffic sent to this group. A multicast stream is first assigned an address within class D range. Any host that wishes to receive the stream places that stream's class D IP address on which ever interface it uses for IP. Because all clients of the stream would have the same class D address, the multicast is sent to one address and many clients.

The Internet Assigned Numbers Authority (IANA) controls the assignment of IP Multicast Addresses. IANA has assigned the old Class D address space to be used for IP Multicast. This means that all IP Multicast-group addresses will fall in this range:

224.0.0.0 - 239.255.255.255

The IEEE LAN specifications made provisions for the transmission of broadcast and/or multicast packets. In the 802.3 standard, bit 0 of the first octet is used to indicate a broadcast and/or multicast frame. Figure 2 shows the location of the Broadcast/Multicast bit in an Ethernet frame.

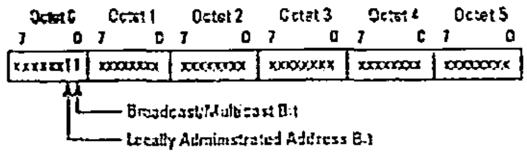


Figure 1.2: IEEE 802.3 MAC Address Format

This bit indicates that the frame is destined for an arbitrary group of hosts or all hosts on the network (in the case of the broadcast address, 0xFFFF.FFFF.FFFF). IP Multicast makes use of this capability to transmit IP packets to a group of hosts on a LAN segment.

1.4 Multicast Distribution Trees

Multicast capable routers create distribution trees that control the path which IP Multicast traffic takes through the network in order to deliver traffic to all receivers. The two basic types of multicast distribution trees are source trees and shared trees.

1.4.1 Source Trees

The simplest form of a multicast distribution tree is a source tree with its root at the source and branches forming a spanning tree through the network to the receivers. Because this tree uses the shortest path through the network, it is also referred to as a shortest path tree (SPT).

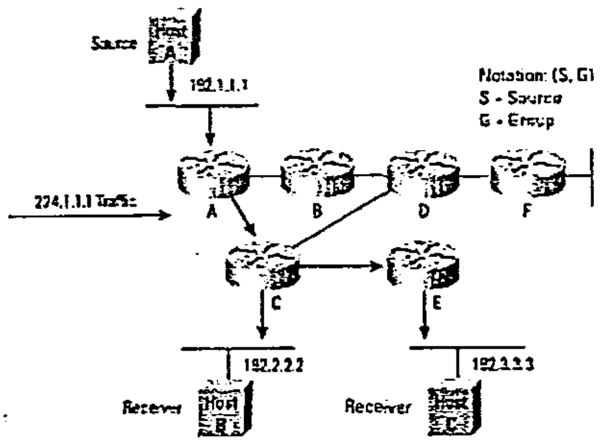


Figure 1.3: Host A Shortest Path Tree

The diagram above shows an example of an SPT for group 224.1.1.1 rooted at the source, Host A, and connecting two receivers, Hosts B and C. The special notation of (S,G), pronounced "S comma G", enumerates an SPT where S is the IP address of the source and G is the multicast group address. Using this notation, the SPT for the example in the figure above would be (192.1.1.1, 224.1.1.1). The (S,G) notation implies that a separate SPT exists for each individual source sending to each group—which is correct. For example, if Host B is also sending traffic to group 224.1.1.1 and Hosts A and C are receivers, then a separate (S,G) SPT would exist with a notation of (192.2.2.2, 224.1.1.1).

1.4.2 Shared Trees

Unlike source trees that have their root at the source, shared trees use a single common root placed at some chosen point in the network. This shared root is called a Rendezvous Point (RP).

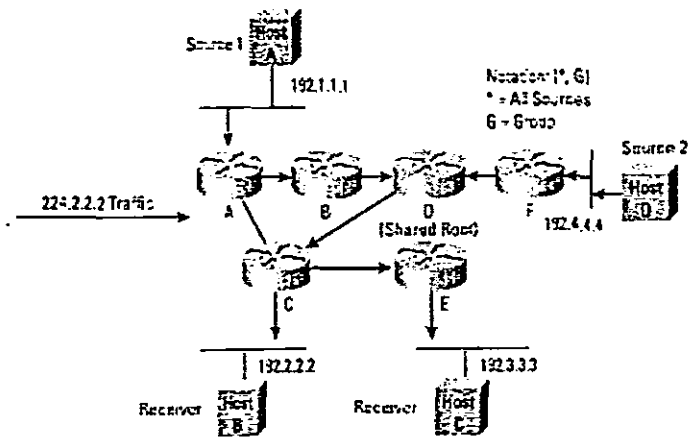


Figure1.4: Shared Distribution Tree

Figure above shows a shared tree for the group 224.2.2.2 with the root located at Router D. When using a shared tree, sources must send their traffic to the root and then the traffic is forwarded down the shared tree to reach all receivers. In this example, multicast traffic from the sources, Hosts A and D, travels to the root (Router D) and then down the shared tree to the two receivers, Hosts B and C. Since all sources in the multicast group use a common shared tree, a wildcard notation written as (*, G), pronounced "star comma G", represents the tree. In this case, * means all sources, and G represents the multicast group. Therefore, the shared tree shown in the figure above would be written as (*, 224.2.2.2).

1.4.3 Source Trees Verses Shared Trees

Both SPTs and Shared Trees are loop-free. Messages are replicated only where the tree branches. Members of multicast groups can join or leave at any time; therefore the distribution trees must be dynamically updated. When all the active receivers on a particular branch stop requesting the traffic for a particular multicast group the routers prune that branch from the distribution tree and stop forwarding traffic down that branch. If one receiver on that branch becomes active and requests the multicast traffic the router will dynamically modify the distribution tree and start forwarding traffic again.

Shortest Path Trees have the advantage of creating the optimal path between the source and the receivers. This will guarantee the minimum amount of network latency for forwarding multicast traffic. This optimization does come with a price. The routers must maintain path information for each source. In a network that has thousands of sources and thousands of groups this can quickly become a resource issue on the routers. Memory consumption from the size of the multicast routing table is a factor that network designers must take into consideration.

Shared Trees have the advantage of requiring the minimum amount of state in each router. This will lower the overall memory requirements for a network that only allows shared trees. The disadvantage of shared trees is that under certain circumstances the paths between the source and receivers might not be the optimal paths—which might introduce some latency in packet delivery. Network designers must carefully consider the placement of the RP when implementing a shared tree only environment.

1.5 Modes of Multicast Routing

Multicast uses one of the two spanning tree technologies dense mode or sparse Mode to get streaming media to its destinations.

1.5.1 Dense Mode

Dense mode assumes that group members are in dense pockets of the network. This is used for large distributions, such as Web cast concerts or corporate presentations, where many receivers on the same subnet or network are getting their broadcasts from the same location. Dense mode also assumes that bandwidth is high enough to handle the broadcasts. Routing protocols such as DVMRP (Distance Vector Multicast Routing Protocol), PIM-DM (Protocol Independent Multicast Dense Mode) and MOSPF (Multicast Open Shortest Path First) are used to distribute data in a dense-mode network.

DVMRP uses a flooding method of getting the multicast data to its destinations. DVMRP relies on the shortest path for propagation. DVMRP does use broadcast technology to update all the routers on the network. This can result in a lot of traffic if routers are spread apart or sending to sparsely connected members, and therefore DVMRP is good only for dense transmissions on high-bandwidth connections. DVMRP also must keep a large amount of state information about all the connections it maintains. Because a DVMRP tree is created for every group ID sent, this information can be overwhelming -- another reason DVMRP works only in dense distributions where the router can keep its connections as tightly packed as possible.

The PIM-DM routing protocol is similar to DVMRP in its overall operation, but it offers a distinct advantage over DVMRP in that it doesn't rely on any one unicast routing protocol. PIM-DM is the dense-mode version of PIM, a standardized, scalable multicast routing protocol.

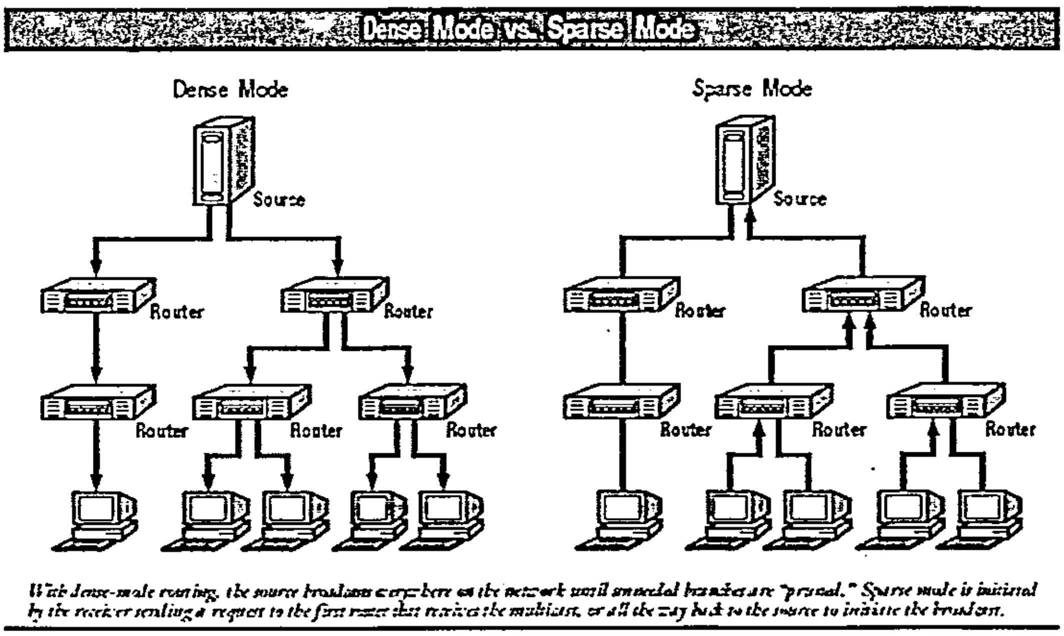
The third routing protocol used in dense-mode networks, MOSPF, extends OSPF to handle multicast traffic instead of only unicast. MOSPF routes data over the lowest-cost connection with available bandwidth and the shortest hop count is used to determine the best path. Using this method, routes over heavily congested connections can be avoided by creating a higher cost for them.

1.5.2 Sparse Mode

Sparse mode might sound as if it's designed for desert multicast sessions, but its purpose is to find efficient means of getting data to many people spread over wide areas. While dense mode assumes there are group members in every corner of the network, sparse mode realizes that for specialized transmissions, members are in small pockets, or clusters, of the network. It's unnecessary for data to be transmitted to every corner of the network when the data needs to reach only a few areas, or areas spread out over long distances. Sparse-mode protocols also are designed to work well over congested connections (for example, in trying to reach a few users scattered across the Internet) and lower-bandwidth connections (such as a low-bandwidth corporate WAN).

Two sparse-mode protocols exist: CBT (Core-Based Trees) and PIM-SM (Protocol Independent Multicast Sparse Mode). Both build routing trees by requiring the routers to participate in creating the tree. Sparse-mode routers effectively ask to join a multicast session when a downstream member requests admission. While dense-mode routers create different trees for each multicast group, CBT simplifies the approach by creating one tree that's used by all groups. CBT employs a tree structure based on a core router from which all data flows, regardless of the source. This is advantageous in that the link-state information is lessened because all groups use the same tree. Conversely, the core router becomes overloaded as more members join. Using multiple core routers is an option, but this solution has not yet been widely adopted.

PIM-SM is similar to CBT in topology, but it's more flexible. Instead of a core router, PIM-SM uses an RP (rendezvous point), where downstream routers "gather." This lets PIM-SM create a shared tree, or one based on the shortest path. Thus, each group can have a different tree structure based on what works best. CBT keeps the amount of link-state information down but may not provide the best path to a member. If low latency is needed, PIM-SM can create better paths.



With dense-mode routing, the source broadcasts everywhere on the network until unneeded branches are "pruned." Sparse mode is initiated by the receiver sending a request to the first router that receives the multicast, or all the way back to the source to initiate the broadcast.

Figure 1.5: Dense Mode vs. Sparse Mode

1.6 Multicast Protocols

ODMRP, MAODV, CAMP multicast protocols exist in which each node in a team acts as an individual unit without effecting group mobility feature. But these protocols are efficient and effective with the small sized multicast groups i.e. less than 100 nodes and suffers with large amount of communication overhead because of flooding of control packets in large–sized networks with large number of multicast groups. The characteristics of ad hoc networks (dynamic topology, limited bandwidth, unreliable transmissions, limited energy supply, etc.) make routing algorithm design particularly challenging, especially if the network grows to thousands of nodes, as is often the case in sensor networks and in battlefield scenarios.

1.7 Landmark Ad hoc Routing Protocol (LANMAR)

Multicasting in large scale mobile and ad hoc networks is proposed which also exploits Team Motion Affinity. Team is dynamically created. The base of our team multicast scheme is Landmark Ad hoc Routing Protocol (LANMAR). LANMAR provides an efficient proactive routing platform which effects teams' mobility. Each team is assigned a unique subnet address by LANMAR and a Landmark node is elected by each team. Team address and the multicast group address is propagated by the landmark into the network by the proactive routing algorithm such as DSDV which have a correct view of network topology at all the times. Routing information is known before hand through continuous route updates. This is how each node in the network has the updated routing entry the landmark of each team.

1.8 Network Simulators

Although there are several network simulators, available for developing different simulations for different types of networks. For the testing and validation of our results the idea was to perform simulations that compare different aspects of the performance of M-LANMAR in comparison with LANMAR.

1.8.1 Simulation programs

As there is a wide range of simulation programs available so we have performed a survey of the commonly used simulatorsNS2, GloMoSim, QualNet and OPNET in order to determine which one is the most suitable simulator for our scenario. While choosing the simulator, we kept the following points in mind:

- How user-friendly is it?
- Is it compatible with our operating system?
- Which protocols does it support?
- Whether it shows the simulation results according to our requirement?

- Is it easy to install?
- How frequent is the simulator used in research-papers regarding MANETs?
- How much it is proved to be successful regarding previous simulations?

1.9 Network Simulator 2 (NS2)

NS2 stands for Network Simulator (version 2). NS is a discrete event driven simulator targeted at networking research. NS provides substantial support for simulation of TCP, routing, and multicast protocols over wired and wireless (local and satellite) networks.

1.9.1 History ofNS2

NS began as a variant of the REAL network simulator in 1989 and has evolved substantially over the past few years. In 1995NS development was supported by DARPA through the VINT project at LBL, Xerox PARC, UCB, and USC/ISI. The wireless code from the UCB Daedelus and CMU Monarch projects and Sun Microsystems, have added the wireless capabilities toNS2.Several versions of NS are NS1 and NS2. NS version2 is a discrete-event driven and object-oriented network simulator. The most recent version ofNS2 isNS-2.26 which is released the 26 of February 2003 and supports AODV, DSDV, DSR and TORA.

1.9.2 Benefits of using NS2

NS2 is mainly used to simulate various networking protocols and architectures. (E.g.: TCP, IP in the wired domain, and, AODV, DSDV in the wireless domain). NS2 can also be used to write applications like FTP, ECHO etc. In NS2 it's possible to alter and write your own code to make it more suitable for our own scenarios.

1.9.3 Flaws of NS2

NS2 cannot be efficiently used by applications that "happen" to be network aware. (E.g.: Applications that are AI based which "happen" to use the network). NS2 cannot be easily used to illustrate the working of multimedia based applications. (E.g.: Realplayer displaying a movie trailer). The software is for the larger part text-based and might therefore be a bit complicated to use if you aren't familiar to Unix-commands

1.9.4 Why we chose NS2 for our simulations?

There are a lot of components (models) that can be chosen for specific simulations. But NS2 can simulate wired networks, wireless networks, satellite networks, etc. with various protocols at the packet level.

- Ns-2 is a popular simulator.
- Support a lot of protocols.
- Free to download.
- Source code available.
- Can be complied on different platforms. E.g. UNIX and Windows.
- An extensive manual and tutorials for the installation and use of the software available on theNS2 homepage.
- Many users to communicate.
- Some parts are managed with GUIs, which makes it easier to understand what's happening.
- A bug report is available for reporting problems and bugs.

1.9.5 OTcl Linkage

Ns is an object oriented simulator, written in C++, with an OTcl interpreter as a front-end. The simulator supports a class hierarchy in C++ (also called the compiled hierarchy), and a similar class hierarchy within the OTcl interpreter (also called the interpreted hierarchy). The two hierarchies are closely related to each other; from the user's perspective, there is a one-to-one correspondence between a class in the interpreted hierarchy and one in the compiled hierarchy. The root of this hierarchy is the class TclObject. Users create new simulator objects through the interpreter; these objects are instantiated within the interpreter, and are closely mirrored by a corresponding object in the compiled hierarchy. The interpreted class hierarchy is automatically established through methods defined in the class TclClass. User instantiated objects are mirrored through methods defined in the class TclObject. There are other hierarchies in the C++ code and OTcl scripts; these other hierarchies are not mirrored in the manner of TclObject.

Ns uses two languages because simulator has two different kinds of things it needs to do. On one hand, detailed simulations of protocols require a systems programming language which can efficiently manipulate bytes, packet headers, and implement algorithms that run over large data sets. For these tasks run-time speed is important and turn-around time (run simulation, find bug, fix bug, recompile, re-run) is less important. On the other hand, a large part of network research involves slightly varying parameters or configurations, or quickly exploring a number of scenarios. In these cases, iteration time (change the model and re-run) is more important. Since configuration runs once (at the beginning of the simulation), run-time of this part of the task is less important. NS meets both of these needs with two languages, C++ and OTcl. C++ is fast to run but slower to change, making it suitable for detailed protocol implementation. OTcl runs much slower but can be changed very quickly (and interactively), making it ideal for simulation configuration. NS (via tclcl) provides glue to make objects and variables appear on both languages.

OTcl is used:

- For configuration, setup, and "one-time" stuff.
- If you can do what you want by manipulating existing C++ objects.

And C++ is used:

- If you are doing anything that requires processing each packet of a flow.

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

For example, links are OTcl objects that assemble delay, queuing, and possibly loss modules. There are certainly grey areas in this spectrum: most routing is done in OTcl (although the core Dijkstra algorithm is in C++).

1.9.5.1 Code Overview

The term "interpreter" is synonymous with the OTcl interpreter. The code to interface with the interpreter resides in a separate directory, tclcl. The rest of the simulator code resides in the directory,NS-2. The notation *~tclcl/* (file) is used to refer to a particular (file) in the Tcl directory. Similarly, the notation, *~ns/* (file) is used to refer to a particular (file) in theNS-2 directory.

There are a number of classes defined in *~tclcl/*. The six that are used in NS are:

- The Class Tcl contains the methods that C++ code will use to access the interpreter.
- The class TclObject is the base class for all simulator objects that are also mirrored in the compiled hierarchy.
- The class TclClass defines the interpreted class hierarchy, and the methods to permit the user to instantiate TclObjects.
- The class TclCommand is used to define simple global interpreter commands.
- The class EmbeddedTcl contains the methods to load higher level built-in commands that make configuring simulations easier.
- the class InstVar contains methods to access C++ member variables as OTcl instance variables.

1.9.5.1.1 Class Tcl

The class Tcl encapsulates the actual instance of the OTcl interpreter, and provides the methods to access and communicate with that interpreter. The class provides methods for the following operations:

Obtain a reference to the Tcl instance.

- Invoke OTcl procedures through the interpreter.
- Retrieve, or pass back results to the interpreter.
- Report error situations and exit in a uniform manner.
- Store and lookup "TclObjects".
- Acquire direct access to the interpreter.

1.9.5.1.2 Class TclObject

Class TclObject is the base class for most of the other classes in the interpreted and compiled hierarchies. Every object in the class TclObject is created by the user from within the interpreter. An equivalent shadow object is created in the compiled hierarchy. The two objects are closely associated with each other.

An object is referred t as a TclObject. This particular object can be either in the class TclObject, or in a class that is derived from the class TclObject. If it is necessary, it is explicitly qualified whether that object is an object within the interpreter, or an object within the compiled code.

1.9.5.1.3 Class TclClass

This compiled class (class TclClass) is a pure virtual class. Classes derived from this base class provide two functions: construct the interpreted class hierarchy to mirror the compiled class hierarchy; and provide methods to instantiate new TclObjects. Each such derived class is associated with a particular compiled class in the compiled class hierarchy, and can instantiate new objects in the associated class.

1.9.5.1.4 Class TclCommand

This class (class TclCommand) provides just the mechanism forNS to export simple commands to the interpreter that can then be executed within a global context by the interpreter. There are two functions defined in ~*n*s/misc.cc: ns-random and ns-version. These two functions are initialized by the function init_misc(void), defined in ~*n*s/misc.cc; init_misc is invoked by Tcl_AppInit(void) during startup.

1.9.5.1.5 Class EmbeddedTcl

Ns permit the development of functionality in either compiled code, or through interpreter code, that is evaluated at initialization. For example, the scripts ~tclcl/tcl-object.tcl or the scripts in ~ns/tcl/lib. Such loading and evaluation of scripts is done through objects in the class EmbeddedTcl. The easiest way to extend NS is to add OTcl code to either ~tclcl/tcl-object.tcl or through scripts in the ~ns/tcl/lib directory. In the latter case, NS sources ~ns/tcl/lib/ns-lib.tcl automatically, and hence the programmer must add a couple of lines to this file so that their script will also get automatically sourced by NS at startup.

Three points to note with EmbeddedTcl code are that firstly, if the code has an error that is caught during the evaluation, then NS will not run. Secondly, the user can explicitly override any of the code in the scripts. In particular, they can re-source the entire script after making their own changes. Finally, after adding the scripts to ~ns/tcl/lib/ns-lib.tcl, and every time thereafter that they change their script, the user must recompile NS for their changes to take effect. The user can source their script to override the embedded code.

1.9.5.1.6 Class InstVar

This section describes the internals of the class InstVar. This class defines the methods and mechanisms to bind a C++ member variable in the compiled shadow object to a specified OTcl instance variable in the equivalent interpreted object. The binding is set up such that the value of the variable can be set or accessed either from within the interpreter, or from within the compiled code at all times. There are five instance variable classes: class InstVarReal, class InstVarTime, class InstVarBandwidth, class InstVarInt, and class InstVarBool, corresponding to bindings for real, time, bandwidth, integer, and boolean valued variables respectively.

CHAPTER 2

2. LITERATURE REVIEW

Before starting research on splitting technique we have searched many research papers, web sites and books. We collected a lot of material to thoroughly understand our research.

2.1 Classification of Routing protocols in Ad hoc Wireless networks

Milenko Petrovic in his research paper "Routing Protocols for Ad hoc Networks" has discussed different routing protocols for Ad hoc networks. In Ad hoc networks, the routing problem has two components; route discovery and route maintenance. For these purposes, routing protocols are needed which uses routing update packets to reflect the changes to all other routers in the network. These protocols are categorized in two types.

Ad hoc Routing Protocols can be classified in three types.

2.1.1 Proactive Routing Protocols

These protocols try to have a correct view of network topology at all times. Any changes in topology are propagated through the network, so that all nodes know of the change in the topology as it happens. This type of protocol operation is considered pro-active, since it tries to determine before they are needed. E.g. DSDV, WRP, CBR, CGSR, FSR. Some of these protocols discussed in his research are:

2.1.1.1 FSR (Fisheye State Routing)

FSR is based on link state routing. The goal of FSR is to reduce flooding used in disseminating link state information when connectivity changes. A node periodically broadcasts link state information. It does not broadcast any information on events (e.g. link failure, changes in topology, etc.), but only periodically. Different update periods are used for nodes at different distances. Nodes with the same update period are said to be within one update scope. Closer nodes are updated more frequently. The consequence of this is that nodes that are further away may have incorrect routes. As packets travel through the network and get closer to their destination, the more accurate the route gets (since nodes closer to the destination are updated more frequently). This method reduces number of broadcast packets.

2.1.1.2 LANMAR (Landmark Routing Protocol)

Milenko Petrovic in the paper states that LANMAR is a combination of link state and distance vector protocols. It borrows from FSR and Landmark routing protocols. It is best suited to networks where group mobility applies. LANMAR uses concepts of landmarks to reduce size of routing tables and effectively handle changes in topology resulting from node mobility. A group of nodes that are in close proximity of each other (i.e. direct communication is possible) have a designated landmark. The landmark node has its Landmark flag set to ON. Nodes that have the same landmark are within the same scope. Routing table at each node contains only routes to nodes in the same scope, and routes to all landmarks. This makes LANMAR suitable for large networks. Note that nodes that are in the same scope have the same subnet address. It is assumed that when nodes move, they do not go beyond their scope (this restriction is relaxed below). It is also possible for all the nodes within the same scope to move together as a group. There is no restriction on group movement. Drifters can cause problem but in the worst case, drifters cause 30% increase in routing table entries. It is also possible that some nodes become isolated. Isolated nodes are subnets of size 1. If number of such nodes is small, then they can be handled just like regular landmarks. If number of isolated nodes is large, then LANMAR routing becomes FSR routing (when all nodes become isolated nodes).

2.1.2 Reactive Routing Protocols (On-Demand Protocol)

These protocols only try to keep valid routing information to the destination that they need. In other words, network topology is detected as needed (On-Demand). These protocols usually initiate topology discovery at a time when traffic requires it. E.g. DSR, AODV, TORA, ABR, SSAR.

2.1.2.1 AODV (Ad-hoc On-Demand Distance Vector)

Operation of the protocol can be divided in two functions – route discovery and route maintenance. During protocol initialization, neighbors are discovered. A node sends Hello message on its interface and receives Hello messages from its neighbors. This process repeats periodically to determine neighbour connectivity. When a route is needed to some destination, the protocol starts route discovery. The source sends Route Request Message to its neighbours. If a neighbour has no information on the destination, it will send message to all of its neighbours and so on. Once request reaches a node that has information on the destination, then that node sends Route Reply Message to the Route Request Message initiator. When Route Reply Message reaches the initiator, the route is ready, and the initiator can start sending data packets. If one of the links on the forward path breaks, the intermediate node just above the link that failed sends new Route Reply Message to all the sources that are using the forward path to inform them of the link failure. It does this by sending the message to all neighbours using the forward path. In turn, they will send to their neighbours until all upstream nodes that use forward path are informed. The source nodes can then initiate new route request procedures if they still need to route packets to the destination.

2.1.2.2 DSR (Dynamic Source Routing)

DSR uses a modified version of source routing. Operation of the protocol can be divided in two functions –route discovery and route maintenance. Route discovery operation is used when routes to unknown hosts are required. Route maintenance operation is used to monitor correctness of established routes and to initiate route discovery if a route fails. When a node needs to send a packet to a destination it does not know about, the node will initiate route discovery. The node sends route discovery request to its neighbours. Neighbours can either send a reply to the initiator or forward the route request message to their neighbours after having added their address to the request message

2.1.3 Hybrid Routing Protocols

There also exists routing protocols that contain both a pro-active and an on-demand component. Such protocols are termed hybrid. E.g. ZRP

2.1.3.1 ZRP (Zone Routing Protocol)

ZRP is a framework for combining operations of on-demand and proactive protocols into a singe routing protocol. ZRP tries to get the best of what on-demand and proactive protocols have to offer. It also tries to improve on their weaker points that arise when only proactive or on-demand protocols are used on their own. . ZRP framework has three major components - IERP (IntEr-zone Routing Protocol), IARP (IntrA-zone Routing Protocol) and BRP (Border cast Resolution Protocol). These protocols are not full protocols but are specifications of how a protocol used in their place should operate. The idea behind ZRP is to partition network into zones. [1]

Amitava Mukherjee, Somprakash Bandyopadhyay and Debashis Saha have discussed multicasting in "Location Management and Routing in Mobile Wireless Networks". Multicast communication in the context of Ad hoc networks is a very useful and efficient means of supporting group-oriented applications, where the need for one-to-many data dissemination is quite frequent in critical situations such as disaster recovery and battle field scenarios. Instead of sending data via multiple unicast, multicasting reduces the communication costs by minimizing the link bandwidth consumption and delivery delay [2].

2.1.4 Tree-based vs. mesh-based

In another paper "Scalable Team Multicast in Wireless Ad hoc Networks Exploiting coordinated Motion", Yunjung Yi, Xiaoyan Hong and Mario Gerla have discussed two other categories of multicast routing protocols. Multicast routing protocols developed specifically for MANET, some are tree-based, and some are based on a mesh structure. A tree based protocol uses a multicast tree structure with a high data forwarding efficiency and low robustness e.g. MAODV (Multi-cast Ad hoc On demand Distance Vector), AM Route (Ad hoc Multicast Routing), LAM (Light weight Adaptive Multicast), LGT (Location Guided Tree) and ARMIS (Ad hoc Multicast Routing protocol utilizing Increasing id number). Whereas in mesh based protocols enhances multicast mesh structure and allows redundant paths between source and a group member with a better robustness and higher forwarding overhead because of increased network load e.g. ODMRP (On Demand Multicast Routing Protocol), MCEDAR (Multicast Core-Extraction Distributed Ad hoc Routing), FGMP (Forwarding Group Multicast Protocol) and CAMP (Core-Assisted Multicast Protocol) .

The key difference between a mesh and a tree structure is how data packets are accepted to be processed. A router is allowed to accept unique packets coming from any neighbor in the mesh, as opposed to trees where a router can only take packets coming from routers with whom a tree branch has been established. Therefore, keeping the branch information updated is one extra challenge protocols based on trees have to face in a mobility scenario [3].

2.2 Performance Analysis of studied Routing Protocols

2.2.1 AODV

Three evaluations of AODV performance are studied. The results cover almost all aspects of RFC 2501. AODV does not support sleeping nodes. All connections are assumed bi-directional. Network connectivity is not reported. Topological rate of change is not reported, although it can be inferred indirectly from the node mobility. The topological change happens uniformly across the network, since mobility (speed, direction) of nodes is also selected uniformly.

In Ad hoc on-demand distance vector routing [4], it is concluded that the protocol performs well for small number of sources that generate traffic. It has > 95% throughput up to 100 nodes. Throughput drops to 85% at 500 nodes and 70% at 1000 nodes. The authors of [5] note that ADOV Route Request Messages propagate to all nodes of the ad hoc network. This is a problem with large number of nodes. In [6], similar observations are made, but since only up to 100 nodes are used, the throughput does not fall below 95%. In [4] and [6], they notice that higher node speed with moderate to high network load causes decrease in throughput since control overhead becomes larger.

AODV performs well in the networks up to 100 nodes with packet delivery rate > 98% regardless of node mobility and network load. For networks with more than 100 nodes and low load, performance is still good regardless of mobility. For networks with more than 100 nodes and high load, end-to-end delay increases as mobility decreases. It was observed that under high load, high node mobility has effect of load balancing. Even though the aim of the protocol is scalability, AODV has problems scaling to very large ad hoc networks (> 1000 nodes).

2.2.2 DSR

Three performance evaluations are studied. DSR performs well with low node mobility where caching optimization has the greatest impact [7], [5]. It has throughput of >90% even under high network load. If the network load is low, it performs well regardless of mobility. At high mobility and moderate to high load, the throughput drops to 60-70%. Under low network load, DSR has very low end-to-end delay, but with moderate to high network load, delay increase 5-7 times. Distributed network load at moderately high mobility rates results in lower end-to-end delay. At highest and lowest mobility, delay is the largest

[5]. Routing overhead reported in [5] is low. In [7] it is reported that when clusters occur, DSR keeps trying to discover routes. The back-off optimization helps, but part of bandwidth is still consumed.

DSR performs quite well. Performance of DSR has been evaluated well for networks up to 100 nodes. At low network load and regardless of mobility, DSR performs well, with throughput of > 90%. At higher mobility, routing overhead increases. Since DSR uses source routing, it sends only a few protocols specific packets, but each data packet has some overhead due to routing information carried. Authors of the protocols suggest that even though overhead in bytes is substantial, it is till more desirable to have overhead in data packets rather than in routing protocol packets, which have to compete with data packets for the shared medium, which is more expensive than Optimization techniques used improve performance of DSR considerably. They also require all as network load increases, performance of DSR decreases monotonically with network load. Coupled with high mobility, DSR performance can suffer dramatically in large (~100 nodes) Distributed network load effect has been observed for high network load and medium mobility.

2.3.4 FSR

Two performance analysis are studied. [8] reports that increasing number of update scopes decreases number of packets sent by flooding during link state update. This also results in more inaccurate routing, which decreases throughput. End-to-end delay increases slowly with increasing network load. Since FSR uses periodic broadcasts, routing overhead is constant regardless of load and mobility. Thus, with increased load, the routing overhead decreases. This makes FSR scalable. On the other hand, with slow changing topologies, there is still a considerable routing overhead, since periodic broadcasting is done even if there are no changes. The authors also note that increasing number of scopes beyond three does not result if further significant reduction in routing overhead for this simulation setup.

In [9] we can see that at low mobility (<1 m/s) and high network FSR performs quite well, delivering more that 80% of packets. As soon as mobility increases beyond 1m/s, throughput drops considerably (40% at 2m/s). From this point, the throughput decreases more slowly so that at 10m/s it is at 20%. At low network load, similar behavior is observed, but throughput is somewhat better, but it is still less then 50% for speeds greater than 2m/s. Packet delay increases monotonically with network load, regardless of node mobility. This study also analyzes storage overhead for FSR. This is important for link state protocols, since their routing tables contain directions to every node in the network. Routing table size can be seen as a potential draw back for FSR scalability, especially since we are dealing with mobile devices where all components can be very expensive.

FSR performs well at low node mobility. Routing overhead decreases with increase load. At node mobility increases, throughput drops considerably (~40%) for nodes moving 2m/s or faster. More comparative performance analysis is needed.

2.3.5 LANMAR

One performance analysis is studied. When there is no mobility LANMAR performs well, regardless of network load. At low mobility (<1m/s) the protocol still performs well. At higher mobility (>1m/s) the throughput drops considerably (<40%), regardless of network load. The protocol does show improvement over FSR (on which it is based). The authors contributed this to more accurate nodes to landmarks vs. less accurate nodes to destinations that FSR provides. Average packet delay does not vary with mobility. It increases linearly with network load. Routing overhead for LANMAR depends only network topology and not on number of active nodes. LANMAR routing tables are kept small by use of landmarks. There is a trade-off between routing table size, subnet size and throughput. Larger subnet sizes make routing tables small, but link state updates are more expensive. Smaller subnet size increases routing table sizes, but link state updates are fewer. The authors also note that good LANMAR performance is expected when traffic is evenly distributed across all subnets event when the network load is high.

More comparative performance analysis is needed. Exhibits good scalability where group mobility applies. Similar performance results as for FSR.

2.3.6 DSDV

[5] Selects main parameters for DSDV simulation without an explanation. The parameters include periodic update interval, periodic updates missed before link declared broken, initial triggered update weighted settling time, weighted settling time weighting factor, route advertisement aggregation time and maximum packets buffered per node per destination. Their simulation reveals the following about DSDV. Routing overhead is constant, which is good in terms of scalability. DSDV fails to converge at high mobility. Only up to 70% of packets are delivered. Routing overhead is not large, as compared to on-demand protocols, where with high network load and mobility, overhead is much larger than for DSDV. DSDV finds routes that are very close to optimal. For slower rate of movement (1m/s), delivers more than 98% of packets.

DSDV performs poorly when nodes are mobile. Only for very slow changing network topologies does it perform well. DSDV poor performance contributed to slow reaction to changing network topology which results n many packets being dropped.

2.3.7 ZRP

[10] and [11] show that ZRP performance depends mainly on one parameter - zone radius. Performance of ZRP is evaluated for various zone radii. When radius = 1 then ZRP reduces to simple flooding since every zone contains only immediate neighbours, so every neighbour is also a border node. As radius increases, number of IERP packets decreases all else being the same. This is because numbers of border routers decreases, so IERP messages do not need to be propagated to every node. When radius = 2, ZRP performs best in terms of overhead. With larger radius, zones become bigger, so overhead gets smaller, but routing

tables become bigger since there are more nodes within each zone. [10] and [11] conclude the following. If number of queries (i.e. route discovery) is small and if nodes are moving fast, smaller zones are better. Smaller zones are less sensitive to non-local node movement. If number of queries is large (larger network load) and nodes are moving slower then larger zones are better (since nodes are moving slower less changes to topology are happening, so we might benefit by increasing zone size where routing is done proactively).

ZRP performance is tightly related to zone radius. Networks of different size will require adjustment to zone radius. Authors do not mention if this adjustment can be performed by the protocols itself. More comparative performance studies are needed.

2.4 Research Papers Review

Mobile ad hoc networks (MANETs) are self-organizing networks that do not need a wired/wireless infrastructure. Two nodes communicate directly if they are in the transmission range of each other. Otherwise, they reach via a multi-hop route. Each MANET node must therefore be able to function as a router, forwarding data packets on behalf of other nodes. In many MANET scenarios (e.g., warfront activities, search and rescue, disaster relief operations, etc.), the mobile nodes are often organized in teams with different tasks and, correspondingly, different functional and operational characteristics. In particular, nodes in the same team will have coordinated motion. We call this model the "affinity team model". Moreover, multicast dynamics are on a per team basis - and entire team joins or withdraws from a multicast group. Two or more teams may merge into one; or a team may split in sub-teams, depending on the operational needs. Since MANETs function under severe constraints such as limited bandwidth and energy, group communications should be performed efficient and at low control overhead cost. Several MANET multicast protocols already exist (e.g., ODMRP, MAODV, CAMP). Those protocols work effectively with small-scale multicast groups (e.g., less than 100 nodes). However, they suffer from severe communications overhead caused by control packet floods (e.g., Join Query or Request packet flooding in ODMRP and MAODV) in a large-scale network with a large number of multicast groups.

LANMAR provides an efficient proactive routing platform which efficiently exploits team mobility. LANMAR Landmark Ad Hoc Routing) protocol is a proactive routing [6]. It uses the notion of landmarks to keep track of logical subnets. Such a logical subnet consists of nodes that have a common interest and move together as a "group". A representative of the subnet, i.e., a "landmark" node, is dynamically elected in each subnet. M-LANMAR (Multicast-enabled Landmark Ad hoc Routing) protocol is a proactive scheme, where group member-ship and multicast routes are updated proactively [3].

Xiaoyan Hong, Mario Gerla, Li Ma showed their research about LANMAR protocol in their paper "Multiple-Landmark Routing for Large Groups in Ad Hoc Networks". They stated that previous work in Landmark Ad Hoc Routing (LAN-MAR) has attempted to address the problem of scalability by utilizing the group motion pattern. LANMAR identifies logical sub-nets in which the members have a commonality of interests and are likely to move as a"group". A"landmark" is dynamically elected in each logical subnet and directs packets to its group. However, when a logical subnet grows large in size or acquires an arbitrary, irregular shape, the local routing scope of the landmark may not cover all the nodes in the group. The nodes which are uncovered are treated through registration (to the landmark of its subnet) and packet redirection (from the landmark). Thus, the Landmark forwards the packet to the intended destination. The scheme works well in Small/moderate group sizes. However, too many drifters will increase the routing overhead and lead to performance degradation. We propose a routing scheme ("Multiple-Landmark" Ad Hoc Routing) using multiple landmarks in each logical group. Over-head from using multiple landmarks is minimized as only one land-mark of each group is propagated over the entire network. M-LANMAR dynamically elects multiple landmarks in each subnet and re-elects them when topology changes. Each landmark has direct routing information for nodes within its scope. The union of the multiple landmarks' ranges covers the entire group [12].

Xiaoyan Hong and Mario Gerla in another paper "Dynamic Group Discovery and Routing in Ad Hoc Networks";have discussed their idea for mobile Ad Hoc networks. In some applications of large scale Ad Hoc networks, for example, advanced battlefield scenarios, the assumption that different sets of nodes move as groups is extremely helpful in achieving efficient and scalable routing. In some applications, the groups are known in advance. In other applications, however, groups form very dynamically. The group based LANMAR routing protocol performs equally well with dynamically discovered groups and preformed groups. Once groups are discovered, one can take the advantage of the 2-level Landmark Ad Hoc Routing (LANMAR) routing hierarchy to achieve scalable routing. Alternatively, On-Demand routing schemes such as AODV and DSR can be used to establish routes on demand between groups. As a result, each node maintains two routing tables: local routing table and landmark table which maintain direct routes to near by destinations and routes to all the landmarks from all the subnets respectively [3].

CHAPTER 3

3. PROBLEM DOMAIN

In the next generation of wireless communication systems, there will be a need for the rapid deployment of independent mobile users. Significant examples include establishing survivable, efficient, dynamic communication for emergency/rescue operations, disaster relief efforts, and military networks. Such network scenarios cannot rely on centralized and organized connectivity, and can be conceived as applications of Mobile Ad Hoc Networks. Since the nodes are mobile, the network topology may change rapidly and unpredictably over time. The network is decentralized, where all network activity including discovering the topology and delivering messages must be executed by the nodes themselves, i.e., routing functionality will be incorporated into mobile nodes.

The set of applications for MANETs ranges from small, static networks to large-scale, highly dynamic networks. Therefore, the design of network protocols for these networks is a complex issue. Regardless of the application, MANETs need efficient distributed algorithms to determine network organization, link scheduling, and routing. A lapse in any of these requirements may degrade the performance and dependability of the network.

3.1 Problem Definition

It is a very complex issue to design an efficient routing protocol for MANETS. One of the main challenges of MANET protocol design is the fact that unlike in Internet nodes are moving continuously. In particular, it is difficult to keep track of individual node movements and to route packets to them especially when the network grows large.

Since the performance of MANETs is affected by many factors such as bandwidth and energy as a result group communication is also effected and should be performed efficiently and at low control overhead cost. Many MANET multicast protocols such as ODMRP, MAODV, and CAMP etc already exist but the main problem with these protocols is scalability. MANET multicast protocols can be divided into two categories, tree-based protocols and mesh-based protocols. MAODV, AMRoute and ARMIS are tree-based protocols and are based on a multicast tree structure whereas ADMRP, MCEDAR and CAMP are mesh-based protocols and use a multicast mesh structure. Mesh-based protocols allow redundant paths between a source and a group member.

MANETS are *often used in group-oriented applications* e.g., warfront activities, search and rescue, disaster relief operations, etc. Multicast transmission is intended for group communication.

All of these protocols perform good and effectively with small sized groups but they are affected by communication overhead when large network size results in large number of multicast groups. So the main problem is to design MANET multicast protocol which is scalable to large membership size as well as network size. Nodes are moving continuously so it is difficult to keep track of individual node movements and to route packets to them especially when the network grows large.

3.2 Proposed Solution

In many MANET scenarios like warfront activities, search and rescue, disaster relief operations, etc the mobile nodes are often organized in teams with different tasks and, correspondingly, different functional and operational characteristics. In particular, nodes in the same team will have coordinated motion. We call this model the "affinity team model". For example, attendees of a major conference can be subdivided into teams based on their topic interests for the purpose of organizing birds of a feather sessions; various units in a division can be organized into companies and then further partitioned into task forces based on their as-assignments in the battlefield.

The "affinity team" model considerably simplifies the mobility management problem and allows us to design a routing protocol that scales. In fact, it suffices for a source to know the path to one of nodes in the team (say, a landmark) in order to route a packet to any other destination within that team.

Our basic work is a new approach to multicast in large-scale mobile Ad hoc networks. The base of our team multicast scheme is Landmark Ad hoc Routing (LANMAR) protocol. LANMAR is an effective routing platform that efficiently exploits team mobility. Keeping LANMAR as the foundation, another protocol named M-LANMAR multicast enabled LANMAR protocol is proposed. M-LANMAR improves the scalability of multicast protocols in team environment. It also improves the reliability and congestion control properties of multicast protocol and also achieves low maintenance cost because of underlying hierarchical routing LANMAR. It also best utilizes affinity team model.

So, the domain of our problem is propose a new multicast paradigm in the coordinated movement of teams to obtain a highly scalable, efficient, robust multicast distribution based on the election of landmarks and to investigate a mesh-structure like ODMRP between subscribed landmarks to improve scalability and efficiency.

We will also develop a Simulator to evaluate M-LANMAR and to compare it with other protocols such as ODMRP. In our simulator, we will calculate two metrics for our performance study, delivery ratio and normalized control overhead.

CHAPTER 4

4. SYSTEM DESIGN

The purpose of design is to create architecture for the involving implementations. Object oriented design is the method of designed encompassing the process of objects oriented decomposition and notation for depicting logical and physical as well as static and dynamic models of system under design.

The design phase focuses on defining the software to implement the application. The design object is to produce a model of the system which can be used latter to build the system. The designed goal is to find the best possible design within the limitations imposed by the requirement and the physical social environment in which the system will operate.

4.1 Network Topology

The basic components of our network topology are nodes and links between the nodes. Both nodes and links are creates in NS 2 simulator. There are well defined classes of each of them in OTcl. These are described below:

4.1.1 Node Basics

The Node itself is a standalone class in OTcl. However, most of the components of the node are themselves TclObjects. The typical structure of a (unicast) node consists of two TclObjects: an address classifier and a port classifier. The function of these classifiers is to distribute incoming packets to the correct agent or outgoing link. All nodes contain at least the following components:

- an address , monotonically increasing by 1 (from initial value 0) across the simulation namespace as nodes are created,
- a list of neighbors,
- a list of agents ,
- a node type identifier , and

- a routing module

By default, nodes in are constructed for unicast simulations. In order to enable multicast simulation, the simulation should be created with a multicast option. When a simulation uses multicast routing, the highest bit of the address indicates whether the particular address is a multicast address or an unicast address. If the bit is 0, the address represents a unicast address; else the address represents a multicast address.

4.1.1.1 Configuring the Node

Procedures to configure an individual node can be classified into:

- Control functions
- Address and Port number management, unicast routing functions
- Agent management
- Adding neighbors

Each of the function is described in the following paragraphs.

4.1.1.2 The Classifier

The function of a node when it receives a packet is to examine the packet's fields, usually its destination address, and on occasion, its source address. It should then map the values to an outgoing interface object that is the next downstream recipient of this packet. This task is performed by a simple classifier object. Multiple classifier objects, each looking at a specific portion of the packet forward the packet through the node. A node uses many different types of classifiers for different purposes.

A classifier provides a way to match a packet against some logical criteria and retrieve a reference to another simulation object based on the match results. Each classifier contains a table of simulation objects indexed by slot number. The job of a classifier is to determine the slot number associated with a received packet and forward that packet to the object referenced by that particular slot.

4.1.1.2 Multicast Classifiers

The multicast classifier classifies packets according to both source and destination (group) addresses. It maintains a (chained hash) table mapping source/group pairs to slot numbers. When a packet arrives containing a source/group unknown to the classifier, it invokes an OTcl procedure to add an entry to its table. This OTcl procedure may use the method to add a new (source, group, slot) 3-tuples to the classifier's table.

4.1.2 Links

This is the second aspect of defining the topology. Links are created to connect the nodes and complete the topology. Links supports a variety of other media, including an emulation of a multi-access LAN using a mesh of simple links, and other true simulation of wireless and broadcast media. The CBQlink is derived from simple links and is a considerably more complex form of link

As with the node being composed of classifiers, a simple link is built up from a sequence of connectors. There are instance procedures that operate on the various components defined by some of the connectors

The Link../ns-2/ns-link.tcl is a standalone class in OTcl that provides a few simple primitives. The SimpleLink../ns-2/ns-link.tcl provides the ability to connect two nodes with a point to point link.

Five instance variables define the link:

- **Head:** Entry point to the link, it points to the first object in the link.

- **Queue:** Reference to the main queue element of the link. Simple links usually have one queue per link. Other more complex types of links may have multiple queue elements in the link.

- **Link:** A reference to the element that actually models the link, in terms of the delay and bandwidth characteristics of the link

- **Ttl:** Reference to the element that manipulates the ttl in every packet

- **Drophead:** Reference to an object that is the head of a queue of elements that process link drops.

Note however, that if the user enable tracing multiple times on the link, these instance variables will only store a reference to the last elements inserted. Other configuration mechanisms that add components to a simple link are network interfaces (used in multicast routing), link dynamics models, and tracing and monitors.

4.1.3 Agents

4.1.3.1 UDP Agents

A UDP agent accepts data in variable size chunks from an application, and segments the data if needed. UDP packets also contain a monotonically increasing sequence number and an RTP time stamp. Although real UDP packets do not contain sequence numbers or time stamps, this sequence number does not incur any simulated overhead, and can be useful for trace file analysis or for simulating UDP based applications.

4.1.3.2 TCP Agents

There are two major types of TCP agents: One-Way agents and a Two-Way agent. One-way agents are further sub divided into a set of TCP senders (which obey different congestion and error control techniques) and receivers ("sinks"). The two-way agent is symmetric in the sense that it represents both a sender and receiver. It is still under development.

One-Way TCP senders attempt to capture the essence of the TCP congestion and error control behavior, but are not intended to be faithful replicas of real-world TCP implementations. They do not contain a dynamic window advertisement, they do segment number and ACK number computations entirely in packet units, there is no SIN/FIN connection establishment/teardown, and no data is ever transferred (e.g. no checksums or urgent data).

4.2 M-LANMAR Protocol

M-LANMAR (Multicast-enabled Landmark Ad hoc Routing) protocol is a proactive scheme, where group membership and multicast routes are updated proactively. With the aid of an underlying unicast protocol, the sources maintain the multicast routes to only landmarks of joined teams instead of individual paths to each member.

4.2.1 Join Multicast Group

In LANMAR, each node keeps fresh routes to all landmarks in the network by periodic landmark updates. Using the landmark updates, a team maintains its membership to M- multicast group(s). A landmark of a team that wishes to join the multicast group(s) implicitly advertises "Join Request" to the sources by piggybacking the targeting multicast group ID(s) (address(es)) on landmark broadcast packet. Upon receiving the "implied" Join Request, each node in the network updates respective landmark entry with the subscribed multicast group IDs. Thus, the Join Request will be propagated into the sources in a few landmark table exchanges. Membership is constantly refreshed, as each landmark includes subscribed multicast addresses to all outgoing landmark update packets.

4.2.2 Leave Multicast Group

When a team who is a part of multicast group wants to leave, the landmark removes the ID of that multicast group from its subscribed multicast groups list. Thus, the landmark will stop advertising the group. The landmark's entry at other nodes will be updated accordingly.

4.2.3 Data Propagation

The source nodes look up their landmark table to find the landmark addresses of the subscribed teams. For each landmark that subscribes to this multicast group, the source creates a "virtual link", i.e., a tunnel, to the landmark and sends encapsulated multicast data. Upon reception of the encapsulated data, each landmark initiates flooding within the subnet so that each member can receive the data. With an assumption of restricted size of the subnet ("x" hops from the landmark to all nodes), local flooding is used with initial TTL "x+1". Each node in the team accepts incoming multicast data.

CHAPTER 5

5. DEVELOPMENT

The goal of development and implementation phase is to create the logic of the system design that will translate into the code in the specific language. For a given design the aim in this phase is to implement the design in the best possible manner. The development and implementation phase effects both testing and maintenance profoundly. A well written algorithm and logic can reduce the testing and maintenance effort. An important concept that helps the understandability of program is structured programming and object oriented programming. The goal of the structured programming is too linear in the control flow in the program where as object oriented programming based on events, functions and state of the object. The design of this project is object oriented so coding will perform in object oriented programming.

5.1 Tools

NS is a discrete event simulator targeted at networking research. NS provides substantial support for simulation of TCP, routing, and multicast protocols over wired and wireless (local and satellite) networks.

NS began as a variant of the REAL network simulator in 1989 and has evolved substantially over the past few years. NS is open source package that has always included substantial contributions from other researchers.

5.1.1 NS Installation:

- **Version:**

NS evolves through version 1, version 2, and the most up-to-date version is 2.1b8a. The

versions available to download on NS page are from 2.1b3 to 2.1b8a for all-in-one package. You may download any version that is best fit your need. The newer the version
is, the more modules and features it has. But it does not mean the newer, the better -- you may not need the newly added parts at all.

For NS source core, you could get as early as version 2.0(which is rarely used now). Link to download version 1 of NS is also there in case some researchers may need to use that version.

- **Platform:**

NS supports Unix (FreeBSD, SunOS, Solaris), Linux, and Windows-95/98/2000/NT. Unix is highly preferred, since you will experience less problems in using NS2 on Unix than on Windows.

- **Components:**

The main components of NS-2 are as below:

- Tcl release 8.3.2 (required component)
- Tk release 8.3.2 (required component)
- Otcl release 1.0a7 (required component)
- TclCL release 1.0b11 (required component): simulation interface
- NS release 2.1b8a (required component): simulation code core
- Nam release 1.0a10 (optional component): animation tool
- Xgraph version 12 (optional component): graphic tool

- **Requirements:**

To build NS you need a computer and a C++ compiler. NS is fairly large. The all-in-one versions available to download on NS page are from 2.1b3 to 2.1b8a for all-in-one package. You may download any version that is best fit your need. The newer the version is, the more modules and features it has. But it does not mean the newer, the better -- you may not need the newly added parts at all.

For NS source core, you could get as early as version 2.0(which is rarely used now). Link to download version 1 of NS is also there in case some researchers may need to use that version.

- **Platform:**

NS supports Unix (FreeBSD, SunOS, Solaris), Linux, and Windows-95/98/2000/NT. Unix is highly preferred, since you will experience less problems in using NS2 on Unix than on Windows.

- ***Components:***

The main components of NS-2 are as below:

- Tcl release 8.3.2 (required component)
- Tk release 8.3.2 (required component)
- Otcl release 1.0a7 (required component)
- TclCL release 1.0b11 (required component): simulation interface
- NS release 2.1b8a (required component): simulation code core
- Nam release 1.0a10 (optional component): animation tool
- Xgraph version 12 (optional component): graphic tool

- **Requirements:**

To build NS you need a computer and a C++ compiler. NS is fairly large. The all-in-one

The disadvantage to have 2 languages is long learning curve and hard to debug.

5.2 Activity Diagram

It gives the pictorial representation of algorithm. Activity Diagram is used to represent activities. Basic need is that we want to make procedural design in UML. Operations in sequence are represented in activity diagram. Activity diagrams are useful when we want to describe a behavior which is parallel or when we want to show how behaviors in several use cases interact.

- Figure 3.1 describes the process of Join Group.
- Figure 3.2 describes the process of Leave Group.
- Figure 3.3 describes the process of Data Propagation.

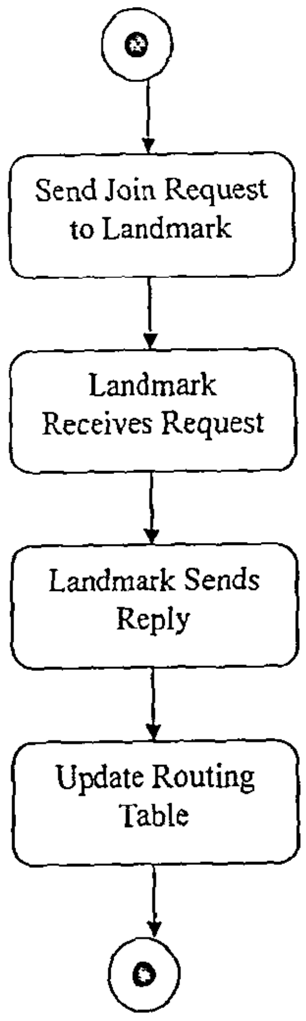


Figure 5.1: Activity Diagram of Join Group

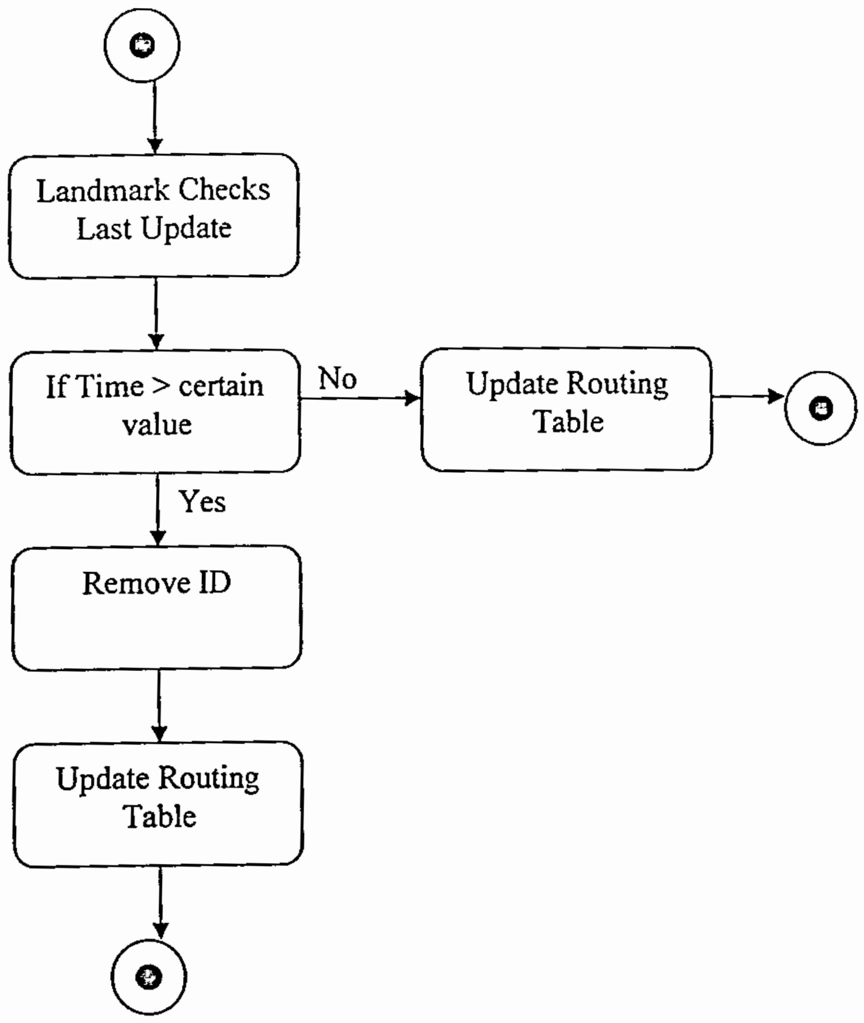


Figure 5.2: Activity Diagram of Leave Group

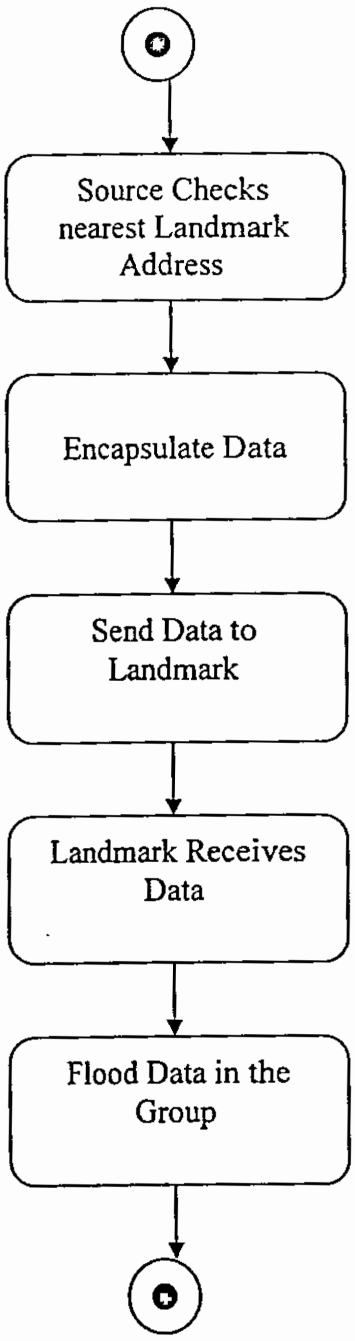


Figure 5.3: Activity Diagram of Data Propagation

5.3 Network Topology in Ns:

This script defines our topology of nodes, and agents, UDP agents with a CBR traffic generators, and a join group and leave group commands. The output is two trace files, out.tr and out.nam. When the simulation completes, it will attempt to run a nam visualization of the simulation on your screen.

While working on the simulation in NS 2, these points must be kept in mind.

a. Which section predefines tracing?

b. How many nodes are considered in the topology?

c. How is the topology defined? Draw it.

d. What is the (bandwidth, delay) associated with each link connecting each pair of nodes?

e. What is the queueing policy for packets at each node?

f. Which application runs from one node to another node?

g. What is the duration of the simulation?

h. Run the simulation. What do the quantitative results represent? How many trace files have been generated?

The script involves:

Initialize the simulation

```
set ns [new Simulator]
```

Predefine tracing

```
set f [open out.tr w]
$ns trace-all $f
set nf [open out.nam w]
$ns namtrace-all $nf
```

Creating nodes

```
set n0 [$ns node]
set n1 [$ns node]
set n2 [$ns node]
set n3 [$ns node]
```

Creating links

```
$ns duplex-link $n0 $n2 5Mb 2ms DropTail
$ns duplex-link $n1 $n2 5Mb 2ms DropTail
$ns duplex-link $n2 $n3 1.5Mb 10ms DropTail
```

Some Agents.

```
set udp0 [new Agent/UDP] ;# A UDP agent
$ns attach-agent $n0 $udp0 ;# on node $n0
```

A CBR traffic generator agent attached to the UDP

```
set cbr0 [new Application/Traffic/CBR]
$cbr0 attach-agent $udp0 agent
```

Creating Groups

```
set group0 [Node allocaddr]
```

Joining Group

```
set rcvr4 [new Agent/LossMonitor]
```

```
Sns attach-agent $n4 $rcvr4
Sns at 1.2 "$n4 join-group $rcvr4 Sgroup0"
```

Leaving Group

```
Snode($k) leave-group $rcvr($k) Sgrp
Sns detach-agent $node($k) $rcvr(Sk)
delete Srcvr(Sk)
```

CHAPTER 6

6. TESTING

Testing is an important phase during software development life cycle and shows the scalability of the software. It also helps in comparing the final software with the objectives. Software testing is a critical element of software quality assurance and represents the ultimate review of specification, design and coding.

Testing should focus upon the systems external behavior. Another purpose of testing is to check how the system fails under certain conditions. The design and implementation both must be tested. The basic objective of testing is to find the maximum number of errors in minimum amount of effort.

6.1 Object Oriented Testing Strategies

Testing begins with unit testing then progress towards integration testing and ends with system testing. In unit testing single modules are tested first. Once they are tested individually they are integrated into a program structure and tested again to find errors due to interfacing of different modules. Finally the system as a whole is tested.

6.2 Types of Testing

We conducted the following type of testing to make the software stable and error free.

- **Code Inspection**

  Review and walk through
- **Unit Testing**

  All the modules of the project are first tested individually.
- **Integration Testing**

  After all the modules tested individually they are combined to form the final product. All the links and paths were tested. This testing should be done at several levels. E.g. tests of two or three objects, dozens of objects and thousands of them are all needed.

- **Black Box testing**

  The software was checked for graphical user interface and measures taken that expected output is generated.

- **System Testing**

  The software was checked as a whole.

- **Beta Testing**

  Used by outsiders rather than developers often makes up for lack of imagination about possible error paths by testers.

- **Portability Testing**

  Test should be applied across the range of systems on which the software may execute. Tests may employ suspected non-portable constructions at the compiler, tool, language, operating system or machine level.

6.3 Evaluation

Evaluation of the software is carried out to check the stability and usability of the product being developed. We took measures to ensure that the developed software becomes effective and our research work is a new paradigm in research world. Some of the features of the software are given below:

- **Efficiency and Effectiveness**

  The product developed is effective and efficient.

- **Accuracy and Reliability**

  The simulator provides reliable and accurate results.

- **Scalability**

  The product is scalable.

6.4 Test Cases

Following are the test cases:

Test Case: 1
Objective: To check Nodes Creation
Test Engineer: Afsheen Khurram
Date: 28th April 2004
Result: Nodes are created successfully.

☒ Pass ☐ Fail ☐ Not Executed

Test Case: 2(a)
Objective: Nodes Configuration.
Test Engineer: Afsheen Khurram.
Date: 30th April 2004
Result: Problem occurred

☐ Pass ☐ Fail ☒ Not Executed

Test Case: 2(b)
Objective: Nodes Configuration
Test Engineer: Afsheen Khurram
Date: 1st May 2004
Result: Nodes configured successfully.

[✓ Pass] ☐ Fail ☐ Not Executed

Test Case: 3
Objective: To check Links Creation
Test Engineer: Shafaq Naz
Date: 1st May 2004
Result: Links successfully created

[✓ Pass] ☐ Fail ☐ Not Executed

Test Case: 4
Objective: To check Group Creation
Test Engineer: Shafaq Naz
Date: 2nd May 2004
Result: Groups Created Successfully

[✓ Pass] ☐ Fail ☐ Not Executed

Test Case: 5(a)
Objective: To check Multicasting Parameters.
Test Engineer: Afsheen Khurram.
Date: 2nd May2004
Result: Large amount of Bugs occurred

☐ Pass [✓ Fail] ☐ Not Executed

Test Case: 5(b)
Objective: To check Multicasting Parameters
Test Engineer: Afsheen Khurram
Date: 4th May 2004
Result: Multicasting parameters not applied properly.

☐ Pass ☐ Fail [✓ Not Executed]

Test Case: 5(c)
Objective: To check Multicasting Parameters.
Test Engineer: Shafaq Naz
Date: 5th May 2004
Result: Multicasting Parameters applied successfully.

[✓ Pass] ☐ Fail ☐ Not Executed

Test Case: 6(a)
Objective: To check Group Joining
Test Engineer: Shafaq Naz
Date: 5th May 2004
Result: Not all the nodes were joining their respective groups.

☐ Pass ■ Fail ☐ Not Executed

Test Case: 6(b)
Objective: To check Group Joining.
Test Engineer: Afsheen Khurram.
Date: 6th May 2004
Result: Nodes successfully join their respective groups.

■ Pass ☐ Fail ☐ Not Executed

Test Case: 7
Objective: To check color change of nodes on joining the group
Test Engineer: Afsheen Khurram
Date: 6th May 2004
Result: Nodes successfully changing colors on each group joining.

[✓ Pass] ☐ Fail ☐ Not Executed

Test Case: 8
Objective: Landmark sending colored packets to their respective members.
Test Engineer: Shafaq Naz
Date: 7th May 2004
Result: Landmark successfully sending packets.

[✓ Pass] ☐ Fail ☐ Not Executed

CHAPTER 7

7. RESULTS AND DISCUSSION

7.1 Comparison of Ad hoc Routing Protocols

We surveyed routing protocols for ad hoc networks that have been submitted to IETF as internet-drafts. Two types of protocols have been review – proactive and on-demand. The conclusions drawn were that on-demand protocols are far superior in energy usage then proactive protocols. This conclusion was based on DSDV that uses periodic broadcasts, even when there are no changes in network topology. As we have seen there are proactive protocols that avoid broadcasts when there are no changes in topology (e.g. ZRP, TBRPF, FSR). These protocols are probably likely to improve their energy usage dramatically. More comparison studies are needed between on-demand and proactive protocols to determine which performs better and under what circumstances [2].

The two on-demand protocols MAODV and ODMRP share certain salient characteristics. In particular, they both discover multicast routes only in the presence of data packets to be delivered to a multicast destination. Route discovery in either protocol is based on request and reply cycles where multicast route information is stored in all intermediate nodes on the multicast path. However, there are several important differences in the dynamics of the two protocols, which may give rise to significant performance differences. First, MAODV uses a shared bi-directional multicast tree while ODMRP maintains a mesh topology rooted from each source. Second, ODMRP broadcasts the reply back to the source while MAODV unicast the reply. Third, MAODV does not activate a multicast route immediately while ODMRP does [7].

Existing studies show that tree-based on-demand protocols are not necessarily the best choice. In a harsh environment, where the network topology changes very frequently, mesh-based protocols seem to outperform tree-based protocols, due to the availability of alternative paths, which allow multicast datagrams to be delivered to all or most multicast receivers even if links fail. Much room still exists to improve protocol performance (as measured by the packet delivery ratio) while reducing the associated overhead.

7.2. Simulation

We evaluated M-LANMAR and also compared it with a robust ad hoc multicast protocol, ODMRP (On-Demand Multicast Routing Protocol). In [18], it is showed that ODMRP generally performs well in a mobile environment compared with other MANET multicast protocols such as CAMP, AMRoute and ARMIS. By keeping this in mind, we limit ourselves to the comparison with ODMRP and with flooding (the latter being the most reliable scheme in a lightly loaded, mobile network). The performance of flooding, obviously, degrades as the offered load (given multicast traffic) increases. We use the following metrics for our performance study:

- Delivery ratio: The ratio of the number of delivered packets to each member versus the number of supposedly received packets by each member.

- The normalized control overhead: the total number of sent control packets (e.g., Join Query/Reply in ODMRP, local/landmark routing table exchanges in M-LANMAR) is divided by the total number of delivered packets to members.

7.3 Simulation Environments

We used NS2 simulator, a packet level simulator targeted at networking research, which provides substantial support for simulation of TCP, routing, and multicast protocols over wired and wireless (local and satellite) networks. We used default parameters provided by NS2. In our simulation, each source generates data in a CBR (Constant Bit Rate) fashion with UDP (User Data Protocol). The transmission range of each node is 376m and bandwidth of the device is 2Mbits/sec. In the network, 20 nodes are uniformly placed and grouped into 4 multicast groups. Each node represents a team. The average number of neighbors for each node is 4 and the average hop count from the landmark node to each node in the logical subnet is 1. For maintaining the routing structures, ODMRP uses 2 seconds interval for each Join Query and M-LANMAR uses 1-second interval for landmark updates and 2.3 seconds period for local routing table exchanges.

Two cases are studied in our simulation study: static and mobile scenario. Each simulation executes for 10 seconds with randomly chosen multicast source and destination team(s). Throughout our simulation study, we use only one source node and 4 teams on average for each multicast group. The source sends out one packet every 2 second with 512 bytes packet size as default.

7.4 Simulation Results

The first experiment uses a static network to examine the scalability of the proposed idea as the number of multicast group's increases. For each multicast group, four randomly selected teams join in.

Fig. 7.1 shows the delivery ratio of three protocols. This graph clearly demonstrates that the performance of ODMRP considerably drops as the number of multicast groups increases. While, M-LANMAR and FLOODING show a stable, consistently high delivery ratio. The main bottleneck of ODMRP is the excessive control overhead due to periodic maintenance messages such as Join Query and Join Reply. In MANET scenarios, due to the shared medium and limited bandwidth, the number of control overhead is extremely important and, in fact, superfluous control packets can considerably impair the delivery ratio of data. Indeed, a scalable protocol should reduce protocol overhead as the offered load increases.

Fig. 7.2 shows the normalized control overhead of ODMRP and M-LANMAR. This result demonstrates that the normalized control overhead of ODMRP slightly increases as the offered load becomes heavy (i.e., the number of multicast group increases). In fact, the total control overhead of ODMRP is proportional to the number of multicast groups. On the other hand, in M-LANMAR, nodes exchange their local routing table and landmark table periodically regardless of actual offered load (i.e., M-LANMAR aggregates multicast group maintenance packets). Thus, the control overhead of M-LANMAR decreases as the actual offered load increases.

The following set of experiments addresses the mobile network shown in Fig. 7.3 and 7.4. Here, we vary the transmission rate with 1 packet/sec and 4 packets/sec for each multicast source. We can observe three striking facts.

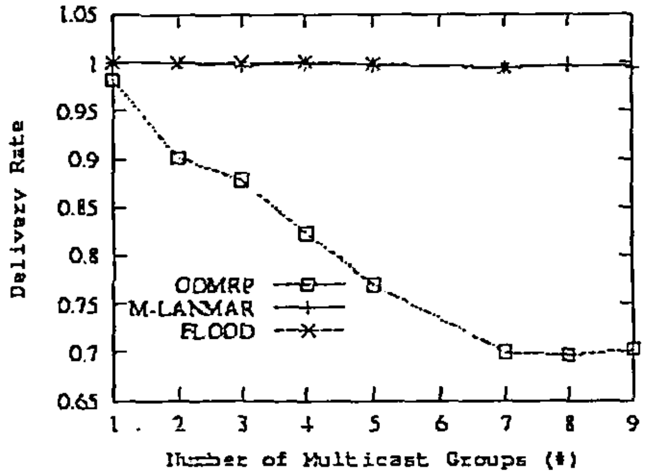


Figure 7.1 shows the Delivery Ratio in the Static Network

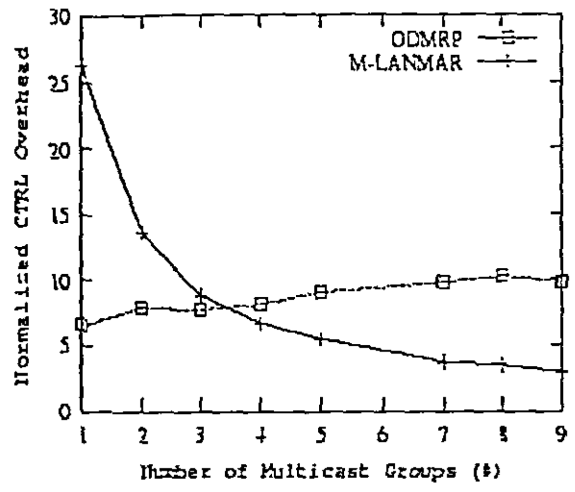


Figure76.2 shows Normalized Control OH in the Static Network

First, Fig.7.3 illustrates that ODMRP outperforms MLANMAR when the offered load is low. This is because with a mesh multicast structure, ODMRP provides redundant paths from the source to a destination and thus enhances the chance of packet delivery to a member even when the primary route fails. On the other hand, since MLANMAR depends on the primary route only, the delivery ratio of M-LANMAR is considerably impaired by the route failures from the source to a landmark due to node mobility. More importantly, this failure of data transmission from the sources to a landmark leads to packet loss at all nodes in the team. This effect is similar to the case when the root of a sub tree in the multicast tree fails. As a result, a single failure significantly reduces the total number of delivered packets. We also note that M-LANMAR provides, in most cases, reliable delivery to all team members once the packet has reached the landmark. In other words, if the landmark of a joined team receives the multicast data, then all members can hear the data with very high probability because M-LANMAR uses flooding within the team. In future work, we will investigate multiple paths (redundant paths) to enhance the route availability between source and landmarks. One possible solution is to construct a mesh structure (like ODMRP) among the sources and subscribed landmarks instead of unicast tunneling.

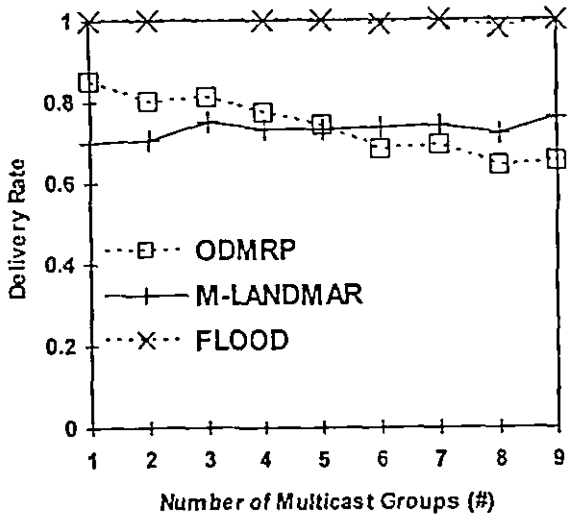
mobility. ODMRP suffers from heavy contention and collision due to the increase of control overhead and the number of relayed packets. Remarkably, for all scenarios, M-LANMAR shows stable delivery ratio regardless of the given offered load. All these observations put together indicate that M-LANMAR provides a scalable team multicast solution.

Lastly, the analysis of flooding shows that the delivery ratio in flooding drops with heavy offered load as shown in Fig. 4. We could not even complete the execution of the flooding runs with a large number of multicast groups (≥ 4) due to heavy memory requirements. In [19], the authors introduce the "broadcast storm" problem, where flooding results in heavy contention and collision in MANET scenarios. Indeed, with small size networks and low offered load, flooding can improve the reliability via redundant packet forwarding.

However, flooding becomes inefficient due to heavy overhead in the dense and large network with high offered load. For that reason, we use restricted scope flooding in each team to exploit the advantage of flooding scheme (such as high reliability) but without paying the huge overhead. In summary, through our extensive simulation studies, we learn that, in realistic scenarios where high offered load and large number of multicast groups are given, M-LANMAR provides an efficient and reliable team multicast solution.

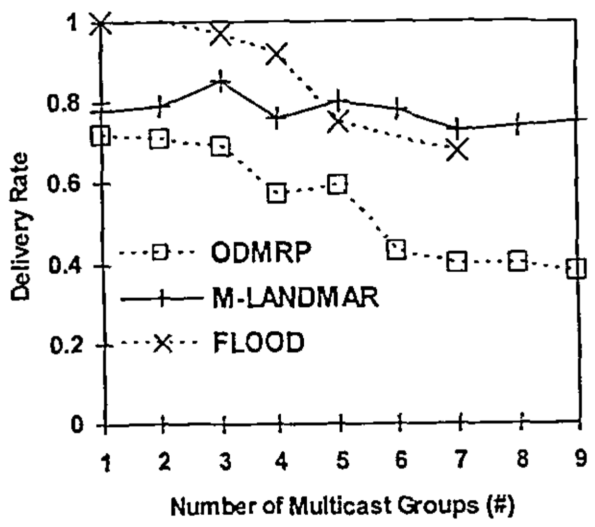
7.5 Conclusion and Future Works

In this thesis, we propose a new multicast paradigm, namely team multicast. We exploit the coordinated movement of teams to obtain a highly scalable multicast distribution based on the election of landmarks. As a starting point, we have implemented M-LANMAR, a landmark based scheme that uses tunneling from the source to the landmark in each team and then flooding within the team. We study the performance of M-LANMAR and compared it with ODMRP and FLOOD.



Delivery Ratio in Mobile Network with
1pkt/sec Transmission

Figure 7.3 showing Delivery Ratio in Mobile Network with 1pkt/sec Transmission



Delivery Ratio in Mobile Network with
4pkts/sec Transmission

Figure 7.4 showing Delivery Ratio in Mobile Network with 4pkts/sec Transmission

Secondly, Fig. 7.4 shows that, offered load become heavier (up to four times that of the scenarios in Fig. 3), MLANMAR does performs better than ODMRP even in presence of

Obtaining the results following three facts are noticed. First, a "flat" Multicast protocol that does not exploit the affinity team model has scalability limitations. While, a team multicast protocol is well scaled as the offered load increases. Secondly, in presence of node mobility, redundant paths provided by a mesh topology can considerably enhance the delivery ratio. However, such multiple paths also exacerbate the contention and collision. Finally, team multicast not only outperforms the conventional schemes but also provides the opportunity for several enhancements such as the support of reliable delivery (via UDP), and congestion control, and resource discovery.

Since M-LANMAR uses separate tunneling from a source to each landmark, with large number of joined teams, MLANMAR may be inefficient i.e., waste bandwidth. Thus, we will further investigate a mesh-structure like ODMRP between subscribed landmarks to improve the efficiency, robustness and scalability.

BIBLIOGRAPHY AND REFERENCES

Bibliography and References:

[2] "Location Management and Routing in Mobile Wireless Networks"; Amitava Mukherjee, Somprakash Bandyopadhyay and Debashis Saha; Artech House Publishers; U.S.A; 2003.

[1] "Routing Protocols for Ad hoc Networks"; Milenko Petrovic; 2001

[3] Scalable Team Multicast in Wireless Ad hoc Networks Exploiting coordinated Motion"; Yunjung Yi, Xiaoyan Hong and Mario Gerla;

[4] Ad hoc on-demand distance vector routing. Perkins, C. Royer, E. Proceedings of the second IEEE Workshop on Mobile Computing Systems and Applications. 1999.

[5] A Performance Comparison of Multi-Hop Wireless Ad Hoc Routing Protocols. Broch, J. et al. Proceedings of ACM/IEEE MOBICOM. 1998.

[6] Performance comparison of two on-demand routing protocols for ad hoc networks, Das, S. Perkins, C. Royer, E Proceedings of the IEEE Conference on Computer Communications. 2000.

[7] Dynamic Source Routing in Ad Hoc Wireless Networks. Mobile Computing. Johnson, D. Maltz, D. Kluwer Academic Publishers. 1996.

[8] Fisheye State Routing: A Routing Scheme for Ad Hoc Wireless Networks, Pei, G, Gerla, M. Chen, T, Proceedings of the IEEE International Conference on communications. 2000.

[9] "LANMAR: Landmark routing for large scale wireless ad hoc networks with group mobility"; M. Gerla, X. Hong, and G. Pei.; Proceedings of IEEE/ACM MobiHOC; 2000.

[10] On the Performance of a Routing Protocol for the Reconfigurable Wireless Networks, Haas, Z, VTC'98. 1998.

[11] The Performance of a New Routing Protocol for the Reconfigurable Wireless Networks. Haas, Z. Pearlman, M. ICC'98. 1998.

[12] "Multiple-Landmark Routing for Large Groups in Ad Hoc Networks"; Xiaoyan Hong, Mario Gerla, Li Ma.

[13] "Dynamic Group Discovery and Routing in Ad Hoc Networks"; Xiaoyan Hong and Mario Gerla.

List of Figures

RESEARCH PAPER

A New Scalable Team Multicast Model for Mobile Ad Hoc Networks with the *Comparison of Pre-Existing Multicast Protocols*

*Shafaq Naz** *Afsheen Khurram** *Tauseef Ur Rehman***

ABSTRACT

A new model for team multicast for large scale mobile ad hoc networks is proposed in this paper. This proposal exploits Team Multicast and Team Dynamics. In Team Multicast the nodes which share common interest belong to one team. All members in a team participate in same multicast group. Our approach includes Motion Affinity in which Team members which share common interest have coordinated motion. In this model each team is treated as a logical subnet .In each logical subnet a node called Landmark Node is dynamically elected and landmark node is the representative of the team. The address of the elected landmark and there paths are propagated throughout the whole network. In MANETS large membership size and network size leads to the problem of scalability. Our proposed M-LANMAR is a new protocol designed for large scalable MANETS which is based on pre-existing LANMAR.

1. INTRODUCTION

Multicasting is the transmission of datagram to a group of hosts identified by a single destination address. Multicasting is intended for group-oriented computing. The multicast service is critical in applications characterized by the close collaboration of teams (e.g. rescue patrol, battalion, scientists, etc) with requirements for audio and video conferencing and sharing of text and images. The use of multicasting within a network has many benefits. Multicasting reduces the communication costs for applications that send the same data to multiple recipients. Instead of sending via multiple unicasts, multicasting minimizes the link bandwidth consumption, sender and router processing, and delivery delay. Maintaining group membership information and building optimal multicast trees is challenging even in wired networks. However, nodes are increasingly mobile [1].

One particularly challenging environment for multicast is a mobile ad-hoc network (MANET). A MANET consists of a dynamic collection of nodes with sometimes rapidly changing Multi-hop topologies that are composed of relatively low-bandwidth wireless links. Since each node has a limited transmission range, not all messages may reach all the intended Hosts. To provide communication through the whole network, a source-to-destination path could pass through several intermediate neighbor nodes. Unlike typical wire line routing protocols, Ad-hoc routing protocols must address a diverse range of the network topology can change randomly and rapidly, at unpredictable times. Since wireless links generally have lower capacity, congestion is typically the norm rather than the exception. The majority of nodes will rely on batteries, thus routing protocols must limit the amount of control information that is passed between nodes. The majority of applications for the MANET technology are in areas where rapid deployment and dynamic reconfiguration are necessary and the wire line network is not available. These include military battlefields, emergency search and rescue sites, classrooms, and conventions where participants share information dynamically using their mobile devices.

* Department of Computer Science, Faculty of Applied Sciences, International Islamic University, Islamabad, Pakistan.
** Department of Telecommunication, Faculty of Applied Sciences, International Islamic University, Islamabad, Pakistan.

These applications lend themselves well to multicast operation. In addition, within a wireless medium, it is even more crucial to reduce the transmission overhead and power consumption. Multicasting can improve the efficiency of the wireless link when sending multiple copies of messages by exploiting the inherent broadcast property of wireless transmission [2].

In MANETS nodes are often organized in teams with different tasks corresponding different functional and operational characteristics. Nodes in same team have coordinated motion. This model is referred as *Affinity Team Model*. Few assumptions about Team Multicast are made that Team is based on some nodes that have common interest and thus all the members in team participate in same multicast group. As multicast dynamics are on a per team basis so the entire team joins or leaves from a multicast group. Depending upon different needs the two or more teams merge into one or they split up into sub-teams [2].

ODMRP [3], MAODV [4], CAMP [5] multicast protocols exist in which each node in a team acts as an individual unit without effecting group mobility feature. But these protocols are efficient and effective with the small sized multicast groups i.e. less than 100 nodes and suffers with large amount of communication overhead because of flooding of control packets in large–sized networks with large number of multicast groups. The characteristics of ad hoc networks (dynamic topology, limited bandwidth, unreliable transmissions, limited energy supply, etc.) make routing algorithm design particularly challenging, especially if the network grows to thousands of nodes, as is often the case in sensor networks and in battlefield scenarios.

Multicasting in large scale mobile and ad hoc networks is proposed which also exploits Team Motion Affinity. Team is dynamically created. The base of our team multicast scheme is Landmark Ad hoc Routing Protocol (LANMAR) [6]. LANMAR provides an efficient proactive routing platform which effects teams' mobility. Each team is assigned a unique subnet address by LANMAR and a Landmark node is elected by each team. Team address and the multicast group address is propagated by the landmark into the network by the proactive routing algorithm such as DSDV which have a correct view of network topology at all the times. Routing information is known before hand through continuous route updates. This is how each node in the network has the updated routing entry the landmark of each team [2].

2. MANET MULTICAST PROTOCOLS

On the foundation of LANMAR, MLANMAR (multicast enabled LANMAR) protocol is proposed. In the multicast group, multicast source sends multiple copies of the packet to the landmark which further floods that packet to the associated team. Many multicast routing protocols have been proposed for ad-hoc networks. Comparing these protocols is typically done based on extensive simulation studies. Multicast routing protocols developed specifically for MANET, some are tree-based, and some are based on a mesh structure. A tree based protocol uses a multicast tree structure with a high data forwarding efficiency and low robustness e.g. MAODV (Multi-cast Ad hoc On demand Distance Vector)[4], AM Route (Ad hoc Multicast Routing)[7], LAM (Light weight Adaptive Multicast), LGT (Location Guided Tree) and ARMIS (Ad hoc Multicast Routing protocol utilizing Increasing id number) [8]. Whereas in mesh based protocols enhances multicast mesh structure and allows redundant paths between source and a group member with a better robustness and higher forwarding overhead because of increased network load e.g.

ODMRP (On Demand Multicast Routing Protocol) [3], MCEDAR (Multicast Core-Extraction Distributed Ad hoc Routing)[9], FGMP (Forwarding Group Multicast Protocol) and CAMP (Core-Assisted Multicast Protocol) [5]. With a *multicast* mesh, members are permitted to receive multicast packets from any their forwarding neighbors, instead of from the only one node (parent) in a multicast tree. Thus, a mesh topology improves the connectivity of a multicast structure and the availability of multicast routes in the presence of dynamic topology changes. A third category also exists which combines advantages of both tree and mesh based protocols called Hybrid protocol by which better performance can be achieved e.g. ZRP (Zone Routing Protocol) [10].

The key difference between a mesh and a tree structure is how data packets are accepted to be processed. A router is allowed to accept unique packets coming from any neighbor in the mesh, as opposed to trees where a router can only take packets coming from routers with whom a tree branch has been established. Therefore, keeping the branch information updated is one extra challenge protocols based on trees have to face in a mobility scenario.

2.1 OVERVIEW OF PROTOCOLS

The MAODV (*Multicast* Ad-hoc On-Demand Distance Vector) routing protocol discovers multicast routes on demand using a broadcast route-discovery mechanism. A mobile node originates a Route *Request* (RREQ) message when it wishes to join a multicast group, or when it has data to send to a multicast group but it does not have a route to that group. Only a member of the desired multicast group may respond to a join RREQ. A node receiving a RREQ first updates its route table to record the sequence number and the next hop information for the source node. This reverse route entry may later be used to relay a response back to the source. The responding node updates its route and multicast route tables by placing the requesting node's next hop information in the tables, and then unicasts a Request *Response* (RREP) back to the source node. As nodes along the path to the source node receive the RREP, they add both a route table and a multicast route table entry for the node from which they received the RREP, thereby creating the forward path. When a source node broadcasts a RREQ for a multicast group, it often receives more than one reply. The source node keeps the received route with the greatest sequence number and shortest hop count. The next hop, on receiving this message, enables the entry for the source node in its multicast route table. This process continues until the node that originated the RREP (member of tree) is reached. The activation message ensures that the multicast tree does not have multiple paths to any tree node. Nodes only forward data packets along activated routes in their multicast route tables. The first member of the multicast group becomes the leader for that group. The
Multicast group leader is responsible for maintaining the multicast group sequence number and broadcasting this number to the multicast group. This is done through a Group Hello message. The Group Hello contains *extensions that indicate* the multicast group IP address and sequence numbers (incremented every Group Hello) of all multicast groups for which the node is the group leader. Nodes use the Group Hello information to update their request table. Links in the tree are monitored to detect link breakages
When a link breakage is detected, the node that is further from the multicast group leader
(downstream of the break) is responsible for repairing the broken link. MAODV

maintains multicast tree structure using prune messages [4].

AMRoute presents a novel approach for robust IP Multicast in mobile ad hoc networks by exploiting user-multicast trees and dynamic logical cores. It creates a bidirectional, shared tree for data distribution using only group senders and receivers as tree nodes. Unicast tunnels are used as tree links to connect neighbors on the user-multicast tree. Thus, AMRoute does not need to be supported by network nodes that are not interested or capable of multicast, and group state cost is incurred only by group senders and receivers. Also, the use of tunnels as tree links implies that tree structure does not need to change even in case of a dynamic network topology, which reduces the signaling traffic and packet loss. Thus AMRoute does not need to track network dynamics; the underlying unicast protocol is solely responsible for this function. AMRoute does not require a specific unicasts routing protocol; therefore, it can operate seamlessly over separate domains with different unicasts protocols. Certain tree nodes are designated by AMRoute as logical cores, and are responsible for initiating and managing the signaling component of AMRoute, such as detection of group members and tree setup. Cores periodically send Join Requests and members send Join Reply. And, ARMIS, without depending on underlying unicasts routing protocol, maintains a tree structure. A new member can join the multicast group by sending a unicasts Join Query packet to the potential parent. If this join fails, then the new member incrementally broadcasts Join Request until succeeding in joining [7].

ODMRP (On-demand Multicast Routing Protocol) is mesh based, and uses a forwarding group concept (only a subset of nodes forwards the multicast packets). No explicit control message is required to leave the group. In ODMRP, group membership and multicast routes are established and updated by the source on demand. When a multicast source has packets to send, but no route to the multicast group, it broadcasts a Join-Query control packet to the entire network. This Join-Query packet is periodically broadcast to refresh the membership information and update routes. When an intermediate node receives the Join-Query packet, it stores the source ID and the sequence number in its message cache to detect any potential duplicates. The routing table is updated with the appropriate node ID (i.e. backward learning) from which the message was received for the reverse path back to the source node. If the message is not a duplicate and the Time-To-Live (TTL) is greater than zero, it is rebroadcast. When the Join-Query packet reaches a multicast receiver, it creates and broadcasts a "Join Reply" to its neighbors. When a node receives a Join Reply, it checks if the next hop node ID of one of the entries matches its own ID. If it does, the node realizes that it is on the path to the source and thus is part of the forwarding group and sets the FG_FLAG (Forwarding Group Flag). It then broadcasts its own Join Table built upon matched entries. The next hop node ID field is filled by extracting information from its routing table. In this way, each forward group member propagates the Join Reply until it reaches the multicast source via the selected path (shortest). This whole process constructs (or updates) the routes from sources to receivers and builds a mesh of nodes, the forwarding group [3].

Another mesh-based multicast protocol, CAMP extends the basic approach of the core-based tree (CBT) protocol for the creation of multicast structure with an allowance of one or multiple cores. "Cores" can limit control overhead for members to join the multicast groups. CAMP is based on underlying unicasts routing protocol (bellmen-ford routing scheme) to get the correct distance between nodes. A new member that is not part of multicast mesh first sends a Join Request to the nearest core if none of its neighbors are joined to the targeting group. Otherwise, this node advertises membership to neighbor nodes. Without the reachable core, a new member broadcasts Join Request using incremental flooding (i.e.. increase the initial TTL upon retransmission of Join Request packet). Any member node can send Join Reply to the new requested member. and the new member chooses the shortest path among multiple replies. With deploying one or multiple Cores for each multicast mesh, CAMP can reduce flooding overhead for Join Request packets. However, CAMP has a limitation of scalability due to the underlying proactive unicasts routing protocol. In large scale network, a "flat" proactive routing scheme such as bellman-ford scheme results the huge memory requirement and heavy routing overhead [5].

2.2 COMPARISON OF MANET MULTICAST PROTOCOLS

A performance evaluation and comparative analysis of multicast protocols for ad hoc networks are presented. The results have been obtained for a broad range of parameters including mobility, number of senders multicast group size and traffic load. In a mobile scenario a general conclusion is that mesh based protocols outperformed tree based protocols. The availability of alternate routes provided robustness to mobility. AMRoute performed well under no mobility, but it suffered from loops and inefficient trees even for low mobility. CAMP showed better performance when compared to tree protocols but with mobility, excessive control overhead caused congestions and collisions that resulted in performance degradation. ODMRP was very effective and efficient in most of the scenarios. However, the protocol showed a trend of rapidly increasing overhead as the number of senders increased.

All of these protocols work well with small sized networks but when the network size grows with the increasing number of groups as well as group size these protocols suffer from severe communication overhead caused by the flooding of control packet which also effects overall network performance. So a new scalable multicast protocol is needed to exploit the group affinity model. LANMAR works effectively with affinity team model.

3. LANMAR PROTOCOL

LANMAR (Landmark Ad hoc Routing) protocol is a combination of link state and distance vector protocols. It is a proactive routing scheme [6]. LANMAR has attempted to address the problem of scalability in ad hoc networks. LANMAR uses the concept of Landmarks to keep track of logical subnets. It assigns a unique address to each logical subnet. All nodes in the logical subnet share the same interest and are likely to move as a 'group'. LANMAR relies on the notion of group mobility i.e. a logical group moves in a coordinated fashion. The existence of such logical group can be efficiently reflected in the addressing scheme. An IP address consists of a group ID (or subnet ID) and a host IS, i.e. <Group ID, Host ID> [11]. A Landmark is dynamically elected in each group. The route to a landmark is propagated throughout the network using Distance Vector mechanism. Separately, each node in the network uses a "scoped" routing algorithm (e.g., FSR) to learn about routes within a given (max number of hops) scope. To route a packet to a destination outside its scope, a node will direct the packet to the landmark corresponding to the group ID of such destination. Once the packet is within the scope of the landmark, it will typically be routed directly to the destination. Remote groups of nodes are "summarized" by the corresponding landmarks. The solution to drifters (i.e., nodes outside of the scope of their landmark) is also handled by LANMAR. Landmark dynamic election enables LANMAR to cope with mobile environments. Thus, by using the truncated local routing table and the "summarized" landmark distance vector, LANMAR dramatically reduces routing table size and update overhead in large nets. LANMAR is well suited to provide an efficient and scalable routing solution. It uses two routing schemes for its routing function; one is "myopic" proactive routing scheme. It works up to the local scope of a node and exchange route information up to a limited number of nodes. Second is "long haul" distance vector routing scheme. It propagates the information about the landmark throughout the whole network that is the address of the elected landmark and the route to it. So each node will maintain two routing tables i.e. local and landmark routing tables. In local routing table each node maintains the direct route to the node in its local scope and landmark routing table maintains the route to all the landmarks to all subnets. A node having more than threshold value such as N which is the practical size for grouping proclaims itself as a landmark and broadcast this information to all of its neighbors. When two or more nodes proclaims themselves as a landmark node then the node having maximum number of grouping will be chosen as the elected landmark. If tie occurs then the lowest ID is elected [2].

4. M-LANMAR PROTOCOL

M-LANMAR protocol is a proactive scheme, where group membership and multicast routes are updated proactively. It is an extension of LANMAR with multiple landmarks in one logical subnet. LANMAR works well with small sized networks having small sized groups but when a logical subnet grows in size only one landmark in a subnet may not cover all the nodes in a group. As a result routing overhead increases. So using multiple landmarks in M-LANMAR in each logical subnet this problem is solved as the union of local scopes of multiple landmarks will guarantee full coverage. [12]

4.1 JOIN MULTICAST GROUP

In LANMAR, each node keeps fresh routes to all landmarks in the network by periodic landmark updates. Using the landmark updates, a team maintains its membership to multicast groups. A landmark of a team that wishes to join the multicast group implicitly advertises "Join Request" to the sources. Upon receiving the "implied" join request. each node in the network updates respective landmark entry with the subscribed multicast group IDs. Thus, the join request will be propagated into the sources in a few landmark table exchanges. Membership is constantly refreshed, as each landmark includes subscribed multicast addresses to all outgoing landmark update packets.

4.2 LEAVE MULTICAST GROUP

When a team who is a part of a multicast group wants to leave, the landmark removes the ID of that multicast group from its subscribed multicast groups list. Thus, the landmark will stop advertising the group. The landmarks entry at other nodes will be updated accordingly.

4.3 DATA PROPAGATION

M-LANMAR uses a local routing table for nearest destinations and landmark distance vector for remote ones. A data packet directing to a remote destination is first propagated to the closest landmark of the destination group. On its way if the route to the destination is found in local routing tables then the packet is directly forwarded to it. The source nodes look up their landmark table to find the landmark addresses of the subscribed teams. For each landmark that subscribes to the group, the source creates a virtual link i.e. a point to point tunnel to the destination landmark. The initiating landmark encapsulates the data packet, make multiple copies and send each copy to each landmark in the group. When the packet is reached to the destination landmark, it is decapsulated. The packet is then flooded within the subnet and destination is searched in the local routing tables.

5. SIMULATION

We evaluated M-LANMAR and also compared it with a robust ad hoc multicast protocol, ODMRP (On-Demand Multicast Routing Protocol). In [13], it is showed that ODMRP generally performs well in a mobile environment compared with other MANET multicast protocols such as CAMP, AMRoute and ARMIS. By keeping this in mind, we limit ourselves to the comparison with ODMRP and with flooding (the latter being the most reliable scheme in a lightly loaded, mobile network). The performance of flooding, obviously, degrades as the offered load (given multicast traffic) increases [14]. We use the following metrics for our performance study:

- Delivery ratio: The ratio of the number of delivered packets to each member versus the number of supposedly received packets by each member.
- The normalized control overhead: the total number of sent control packets (e.g., Join Query/Reply in ODMRP, local/landmark routing table exchanges in M-LANMAR) is divided by the total number of delivered packets to members.

5.1 SIMULATION ENVIRONMENTS

We used NS2 simulator, a packet level simulator targeted at networking research, which provides substantial support for simulation of TCP, routing, and multicast protocols over wired and wireless (local and satellite) networks. We used default parameters provided by NS2. In our simulation, each source generates data in a CBR (Constant Bit Rate) fashion with UDP (User Data Protocol). The transmission range of each node is 376m and bandwidth of the device is 2Mbits/sec. In the network, 20 nodes are uniformly placed and grouped into 4 multicast groups. Each node represents a team. The average number of neighbors for each node is 4 and the average hop count from the landmark node to each node in the logical subnet is 1. For maintaining the routing structures, ODMRP uses 2 seconds interval for each Join Query and M-LANMAR uses 1-second interval for landmark updates and 2.3 seconds period for local routing table exchanges.

Two cases are studied in our simulation study: static and mobile scenario. Each simulation executes for 10 seconds with randomly chosen multicast source and destination team(s). Throughout our simulation study, we use only one source node and 4 teams on average for each multicast group. The source sends out one packet every 2 second with 512 bytes packet size as default.

5.2 SIMULATION RESULTS

The first experiment uses a static network to examine the scalability of the proposed idea as the number of multicast group's increases. For each multicast group, four randomly selected teams join in.

Fig. 5.1 shows the delivery ratio of three protocols. This graph clearly demonstrates that the performance of ODMRP considerably drops as the number of multicast groups increases. While, M-LANMAR and FLOODING show a stable, consistently high delivery ratio. The main bottleneck of ODMRP is the excessive control overhead due to periodic maintenance messages such as Join Query and Join Reply. In MANET scenarios, due to the shared medium and limited bandwidth, the number of control overhead is extremely important and, in fact, superfluous control packets can considerably impair the delivery ratio of data. Indeed, a scalable protocol should reduce protocol overhead as the offered load increases.

Fig. 5.2 shows the normalized control overhead of ODMRP and M-LANMAR. This result demonstrates that the normalized control overhead of ODMRP slightly increases as the offered load becomes heavy (i.e., the number of multicast group increases). In fact, the total control overhead of ODMRP is proportional to the number of multicast groups. On the other hand, in M-LANMAR, nodes exchange their local routing table and landmark table periodically regardless of actual offered load (i.e., M-LANMAR aggregates multicast group maintenance packets). Thus, the control overhead of M-LANMAR decreases as the actual offered load increases.

The following set of experiments addresses the mobile network shown in Fig. 5.3 and 5.4. Here, we vary the transmission rate with 1 packet/sec and 4 packets/sec for each multicast source. We can observe three striking facts.

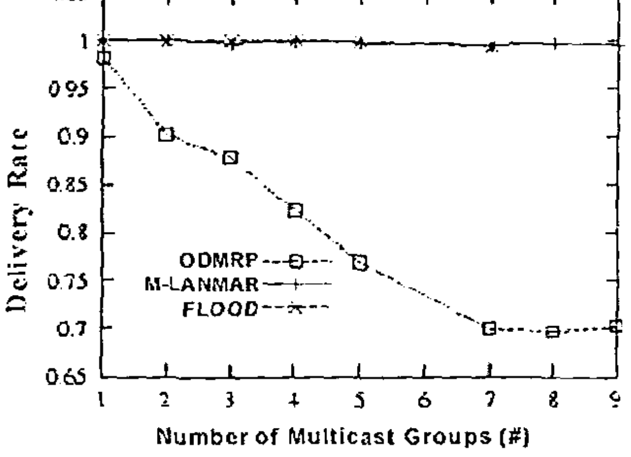


Figure 5.1 shows the Delivery Ratio in the Static Network

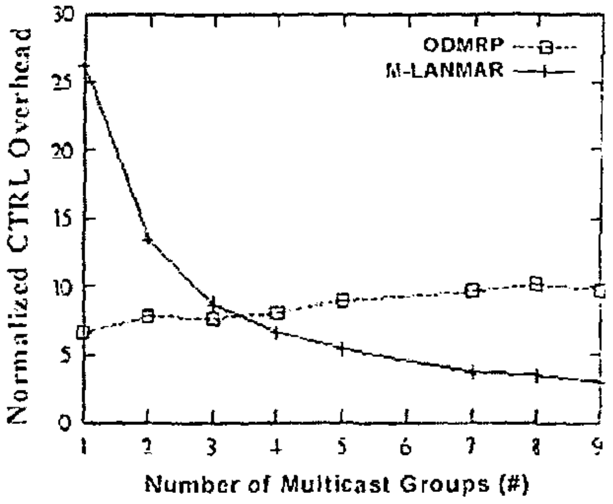


Figure5.2 shows Normalized Control OH in the Static Network

First, Fig.5.3 illustrates that ODMRP outperforms MLANMAR when the offered load is low. This is because with a mesh multicast structure, ODMRP provides redundant paths from the source to a destination and thus enhances the chance of packet delivery to a member even when the primary route fails. On the other hand, since MLANMAR depends on the primary route only, the delivery ratio of M-LANMAR is considerably impaired by the route failures from the source to a landmark load. For that reason, we use restricted scope flooding in each team to exploit the advantage of flooding scheme (such as high reliability) but without paying the huge overhead. In summary, through our extensive simulation studies, we learn that, in realistic scenarios where high offered load and large number of multicast groups are given, M-LANMAR provides an efficient and reliable team multicast solution.

5.3 CONCLUSION AND FUTURE WORKS

In this thesis, we propose a new multicast paradigm, namely team multicast. We exploit the coordinated movement of teams to obtain a highly scalable multicast distribution based on the election of landmarks. As a starting point, we have implemented M-LANMAR, a landmark based scheme that uses tunneling from the source to the landmark in each team and then flooding within the team. We study the performance of M-LANMAR and compared it with ODMRP and FLOOD.

Obtaining the results following three facts are noticed. First, a "flat" Multicast protocol that does not exploit the affinity team model has scalability limitations. While, a team multicast protocol is well scaled as the offered load increases. Secondly, in presence of node mobility, redundant paths provided by a mesh topology can considerably enhance the delivery ratio. However, such multiple paths also exacerbate the contention and collision. Finally, team multicast not only outperforms the conventional schemes but also provides the opportunity for several enhancements such as the support of reliable delivery (via UDP), and congestion control, and resource discovery.

Since M-LANMAR uses separate tunneling from a source to each landmark, with large number of joined teams, MLANMAR may be inefficient i.e., waste bandwidth. Thus, we will further investigate a mesh-structure like ODMRP between subscribed landmarks to improve the efficiency, robustness and scalability.

6. REFERENCES:

[1]. "On-Demand Multicasting in Ad-Hoc Networks: Comparing AODV and ODMRP"; Thomas Kunz and Ed Cheng; Carleton University; 2002.

[2]. "Scalable Team Multicast in Wireless Ad Hoc Networks Exploiting Coordinated Motion"; Yunjung Yi, Xiaoyan Hong and Mario Gerla; 2001.

[3] "On-Demand Multicast Routing Protocol"; S.-J. Lee, M. Gerla, and C.-C. Chiang: Proceedings of IEEE WCNC; 1999.

[4] "Multicast Operation of the Ad-hoc On-Demand Distance Vector Routing Protocol"; E. M. Royer and C. E. Perkins; ACM/IEEE MOBICOM; 1999.

[5] "Multicast Routing Protocol for Ad-hoc Networks"; J.J.Garcia-Luna-Aceves and E. L. Madruga; IEEE INFOCOM; 1999.

[6]. "LANMAR: Landmark Routing for Large Scale Wireless Ad Hoc Networks with Group Mobility"; M. Gerla, X Hong and G. Pei; Proceedings of IEEE/ACM MobiHOC; 2000.

[7] "Amroute: Ad-hoc Multicast Routing Protocol"; E. Bommaiah, M. Liu, A. McAuley, and R. Talpade; Internet-draft, draft-talpade-manet-amroute-00.txt; 1998.

[8] "A Multicast Protocol for Ad-hoc Wireless Networks"; C.-W. Wu and Y. Tay; MILCOM; 1999.

[9] "Mcedar: Multicast Core-Extraction Distributed Ad-hoc Routing"; P. Sinha, R. Sivakumar, and V. Bharghavan; IEEE WCNC: 1999.

[10]. "Location Management and Routing in Mobile wireless networks"; Amitava

Mukherjee, Somparkash Bandyopadhyay and Debashis Sasa; Artech House Publishers; USA: 2003.

[11]. "Dynamic Group Discovery and Routing in Ad Hoc Networks"; Xiaoyan Hong and Mario Gerla; 2001.

[12]. "Multiple-Landmark Routing for Large Groups in Ad Hoc Networks"; Xiaoyan Hong, Mario Gerla, Li Ma: Computer Science Department, University of California. Los Angeles; 2002.

[13] "A Performance Comparison Study of Ad-hoc Wireless Multicast Protocols"; S.-J. Lee. W. Su. J. Hsu, M. Gerla. and R. Bagrodia: IEEE INFOCOM; 2000.

[14] "*Flooding for Reliable Multicast in* Multi-hop Ad-hoc Networks"; C. Ho. K. Obraczka. G. Tsudik. and K. Viswanath:. ACM DIALM: 1999.

[15] "The Broadcast Storm Problem in a Mobile Ad-hoc Networks": S.-Y. Ni. Y.-C. Tseng. Y.-S. Chen. and J.-P. Seu; IEEE/ACM MOBICOM: 1999.