Télécom Paris (ENST)
Institut Eurécom

THÈSE

Pour Obtenir le Grade de Docteur

de l'Ecole Nationale Supérieure

des Télécommunications

Spécialité: **Réseaux et Informatique**

Shiyi WU

Protocoles de diffusion dans les Réseaux Ad Hoc Sans fil

Date de Soutenance: 13/10/2004

Acknowledgment

First and foremost, I would like to thank my supervisor Christian Bonnet for his encouragement and support over the past years. He gave me a lot of freedom with my research as well as the possibility to present my work in several international conferences.

I am grateful to the members of my doctoral committee for their time, support and useful comments.

My work with Christian Bonnet, Navid Nikaein, Houda Labio, Masato Hayashi, and Yukihiro Takatani gave me the chance to gain a valuable experience.

I also want to thank all the staff in the Mobile Communications Department and Institut Eur´ecom for their warm reception. We have spent so much wonderful time together. I will never forget them.

I owe a special gratitude to my family. I want to thank my sister, my father and mother who support me all the time.

Finally, my wife Liu Xiang gave me her infinite support and love. I am grateful to her and my son Wu Haotian for enriching my life.

Abstract

The advances in portable computing and wireless technologies are opening up exciting possibilities for the future of wireless mobile networking. A Mobile Ad hoc NETwork (MANET) is a collection of wireless mobile nodes forming dynamical and temporary network without the use of any existing network infrastructure or centralized administration. Its capability of providing rapidly deployable communication makes it an ideal choice for consumer, company and public uses. Most applications are characterized by a close degree of collaboration. Multicasting could prove to be an efficient way of providing necessary services for these kinds of applications. However, due to the limited transmission range of wireless network interfaces, multiple network "hops" may be needed for one node to exchange data with another one across the network. Consequently, the extra challenges such as frequent topology change and limited network resources are introduced in multicasting protocol design.

In this dissertation, we first examine different techniques and strategies which are used by current multicast routing protocols for MANETs. Then, we present our proposition, multicast routing protocol with dynamic core (MRDC), to provide best effort multicast routing. This protocol addresses the issue of how to optimize packet delivery success rate and overhead. This protocol gives a trade-off between forwarding overhead and routing overhead but also an optimization between delivery success rate and overhead regarding application requirement and network situation. For the applications which require 100% percent packet delivery, we study a reliable multicasting protocol which activates intermediate nodes to assist retransmission. All these works have the same goal: optimize packet delivery ratio and overhead to satisfy application requirement with good utilization of network resources especially bandwidth.

This dissertation also includes our experiences in implementing a mobile ad hoc testbed. We developed this testbed by implementation of DDR, a unicast routing protocol and MRDC so that the testbed can support both one-to-one communications and many-to-many communications. This testbed will allow us to analyze the performance of routing protocols in real network. It can also be used to study protocols and new MANET applications.

Résumé

Les avanc´ees dans le domaine de l'informatique personnelle et des technologies sans fil ouvrent des possibilit´es passionnantes pour le futur de la gestion des r´eseaux mobiles. Les r´eseaux mobiles "ad-hoc" sont cr´e´es par un ensemble de terminaux sans fil qui communiquent entre eux. Les nœuds d'un r´eseau "ad hoc" forment dynamiquement un r´eseau à façon sans utilisation de quelconque infrastructure existante ou administration centralis´ee. Ses capacit´es à fournir rapidement et flexiblement des moyens de communication font des r´eseaux "ad hoc" un choix id´eal pour certaines applications personnelles, publiques ou d'entreprise. Beaucoup de ces applications sont caract´eris´ees par un degr´e ´etroit de collaboration. Le multicast peut s'av´erer être une manière efficace de fournir les services n´ecessaires pour ce genre d'application. En raison de la limitation de la couverture radio de l'interface sans fil, le relayage par sauts multiples peut être nècessaire pour qu'un nœud puisse ´echanger des donn´ees avec les autres à travers le r´eseau. En cons´equence, les d´efis suppl´ementaires tels que le changement fr´equent de topologie et les ressources limit´ees de r´eseau sont à relever dans la conception de protocole multicast.

Dans cette dissertation, nous examinons d'abord les techniques qui sont employ´ees par des protocoles courants de routage de multicast. Ensuite, nous pr´esentons en d´etail notre proposition, Multicast routing protocol with dynamic core (MRDC), pour fournir un routage de multicast de "best effort". Ce protocole adresse le problème de comment optimiser le taux de succès de la livraison de paquets tout en r´eduisant le coût de signalisation du protocole. Il donne une compromis entre les surcharges li´ees au routage et les surcharges de transmission mais ´egalement une optimisation entre le taux de succès de la livraison et les surcharges en regardant des exigences des applications et les conditions du r´eseau. En outre, pour les applications qui exigent la livraison fiable de paquets (cent pour cent de r´eussite), nous proposons un protocole fiable de multicast, Active Reliable Multicast Protocol with Intermediate node support (ARMPIS), qui active des nœuds interm´ediaires pour aider les retransmissions. Tous ces travaux ont le même but : optimiser le taux de livraison de paquets pour r´epondre aux exigences des applications avec la bonne utilisation des ressources du r´eseau et notamment la bande de passante.

Cette dissertation inclut ´egalement notre experience de la construction et de la validation d'un banc de test de r´eseau ad-hoc. Nous avons d´evelopp´e ce banc de test par l'implementation de DDR, d'un protocole de routage d'unicast et de MRDC de sorte que le banc de test puisse supporter des communications point-à-point et aussi des communications multipoint. Ce banc de test nous permettra d'analyser les performances de MRDC dans un vrai r´eseau. Il peut ´egalement être employ´e pour ´etudier des protocoles et de nouvelles applications de r´eseaux ad hoc san fil.

Contents

List of Figures

List of Tables

Glossary

ACK	Positive acknowledgment
ABAM	Associativity-Based Ad hoc Multicast
ADMR	Adaptive Demand-driven Multicast Routing
AMRIS	Ad hoc Multicast Routing protocol utilizing Increasing id-numberS
AMRoute	Ad hoc Multicast Routing protocol
API	Application Interface
AQL	Average Queue Length
ARQ	Automatic Repeat Request
BGMRP	Border Gateway Multicast Routing Protocol
CAMP	Core-Assisted Mesh Protocol
CBR	Constant Bit Rate
CBT	Core-Based Tree
CSMA/CA	Carrier Sense Multiple Access with Collision Avoidance
DCF	Distributed Coordination Function
DDM	Differential Destination Multicast
DVMRP	Distance Vector Multicast Routing Protocol
FGMP-RA	Forwarding Group Multicast Protocol - Receiver Advertising
FGMP-SA	Forwarding Group Multicast Protocol - Sender Advertising
ID	Identifier
ICMP	Internet Control Message Protocol
IGMP	Internet Group Management Protocol
IP	Internet Protocol
LAM	Lightweight Adaptive Multicast Algorithm
MAC	Medium Access Control
MANET	Mobile Ad Hoc Network
MAODV	Multicast operation of Ad-hoc On-demand Distance Vector routing protocol
MCEDAR	Multicast Core Extracti on Distributed Ad hoc Routing
MOR	Medium Occupation for Reception
MOSPF	Multicast Open Shortest Path First
MRDC	Multicast Routing Protocol with Dynamic Core
NACK	Negative Acknowledgment
NSMP	Neighbor Supporting ad hoc Multicast routing Protocol
ODMRP	On-Demand Multicast Routing Protocol
OSI	Open System Interconnection
PIM-DM	Protocol-Independent Multicast - Dense Mode
PIM-SM	Protocol-Independent Multicast - Sparse Mode
RGMP	Receiver-initiated Group-membership Protocol
RTS/CTS	Request to Send/Clear to Send
TCP	Transport Control Protocol
TTL	Time To Live
UDP	User Datagram Protocol

Chapter 1

Introduction

The advances in wireless communication and economical, portable computing devices have made mobile computing possible. The mobile terminal (Portable PC, PDA) as well as mobile telephone becomes indispensable not only in business scene but also in our daily life. People require more from the communications network. Communications with anybody anytime anywhere in whatever forms - data, voice or video - is being envisaged, and automatic roaming of hosts in the network also seems not so "distant". However, the liberty of communication is limited by the network infrastructure. Whenever anyone wants to send information to someone else, he must use network infrastructure. If any entity of a communication pair is out of the coverage range of the network infrastructure, the communication cannot be established even if they are in face-to-face distance from each other. While, seamless coverage is expensive and sometimes impossible. With the above background, one research issue has recently attracted more and more interest. That is the design of mobile ad hoc networks (MANET)[1]. A MANET is an autonomous system formed by a collection of mobile nodes equipped with wireless interface. These nodes communicate with each other without the intervention of any existing network infrastructure or centralized administration. In such a network, each node acts as a host, and may act as a router if it volunteers to carry traffic. If two nodes are out of their radio transmission range, the network is able to establish communication between them with the help of some other nodes. That is why the literature sometimes uses the term "multihop networks" for MANETs.

MANETs, having self-organizing capability, can provide rapidly deployable communication in the place where network connectivity is not attainable or expensive. They are considered as suitable systems to support a number of applications [2]. Some of them are listed below and more can be found in [3].

- Virtual classrooms,
- Military communications,
- Emergency search and rescue operations,

- Data acquisition in hostile environments,
- Audio/Video conference,
- File distribution and
- Internet games etc.

These applications cover consumer uses, company uses and also public uses. Gaming is one of the typical applications for consumer usage where two or several players gather and form a game group somewhere. Email and file transfer are also considered easily deployable within an ad hoc network environment for personal or business uses. There is no need to emphasize the wide range of military and public applications possible with ad hoc networks since the technology was initially developed with them in mind, such as emergency rescue operations after an earthquake wherein existing network infrastructures are completely destroyed.

By analyzing the properties of the potential applications of MANETs, [3] indicates that most of the examples mentioned in that paper include one-to-one, one-to-many and many-to-many communication model. For example, an Internet game can contain only two players (chess) or several players (card games). In the former case, game information has a unique destination - the other player. In the later case, any player is a potential information sender which sends game information to all the other players in which case many-to-many communication is required. This kind of application can also be characterized as group oriented since information exchange takes place among a group of users. One-to-one communication is well studied through unicasting all kinds of networks including wired, wireless and ad hoc networks. However, supporting one-to-many and many-to-many communication, which requires ransmitting packet(s) to multipoint, the research is far from being well done in mobile environment.

In general, networks have three methods to transmit a packet addressed to multiple receivers: unicasting, broadcast and multicasting. In unicast, the sender duplicates the packet and sends separately a copy to every receiver. The same copy will appear on some links. In broadcast, the sender and intermediate nodes send a copy to each outgoing link. As a result, even nodes that do not require a copy will get the packet. Multicasting can efficiently support them. In multicast, the sender makes only one transmission. At each intermediate node, copies are made and sent to outgoing links as required. At most, one copy is required on each physical link. Therefore, multicasting is able to deliver packets to multipoint in an efficient and scalable way, which is more important in MANET where bandwidth is scarce. A protocol is called scalable if it works efficiently even as the size of the network increases. Without surprise, current multicasting protocols for MANET are evolved from that for Internet.

The first Internet multicast paradigm, the “host group” model, is proposed in 1989 [4]. In this model, a single class D IP address identifies the hosts participating in the same multicast session form a host group . A host may join and leave the group at any time and may belong to more than one group at a time. To send datagrams to a group, a host need not know the membership of the group, or be a member of the group. Data delivery in the host group model is best effort. Senders multicast to and receivers receive from their local links and it is the multicast routers that have the responsibility of delivering the multicast datagrams. The Internet multicast architecture is largely evolved from this model. It consists of the group management protocols, the IP multicast routing protocols, and multicast transport protocols. The group management protocols (IGMP [5], [6], [7] and RGMP [8]) are used for group member hosts to report their group information to the multicast routers on the subnet. Multicast routing protocols on the Internet deal with the problem of efficiently transmitting multicast datagrams from the source(s) to the destinations. Although multicast routing protocols provide best effort delivery of multicast datagrams on the Internet, many multicast applications have requirements beyond this. Therefore, various multicast transport protocols are proposed on top of the multicast routing protocols to meet the needs of different applications. Multicast transport protocols serve two major functions, namely, providing reliability and performing flow and congestion control.

In ad hoc network environment, the “host group” model should be slightly modified since nodes act as host and router at the same time. The router to which a group member host should report its group information is probably the node itself. And it is the nodes themselves that have the responsibility of delivering the multicast datagrams. That makes multicast routing a more important issue on providing multicasting for ad hoc networks. This dissertation focuses on the problem of multicast routing in mobile ad hoc networks

1.1 Motivation and Objectives

Multicast routing protocols have twin design goals: high delivery success rate and low overhead. The former is important because it is the principle aim of multicasting - transmitting packets to their receivers. The overhead is very critical since the efficient utilization of network resources is concerned. In fact, there are two kinds of overhead in multicasting: routing overhead and forwarding overhead. Control overhead is generated during routing information collection and update. Forwarding overhead is the result of delivering multicast packets. In the Internet, the network topology is usually stable, few control messages are needed to update routing information. Furthermore, the lower layer protocols in use in the network can efficiently support multicast transmission. These factors permit researchers to focus their interest on the problem of reducing forwarding overhead. They use either a source-based tree or group-shared tree as routing structure to provide best effort multicast delivery. Multicast tree contains the unique path from source(s) to receivers to guarantee transmission efficiency. As for the problem of packet loss during transmission, which is usually caused by congestion, researchers prefer to resolve it in transport protocols. The transport protocols make congestion control in order to reduce packet loss or retransmit lost packets to provide delivery guarantee. However, providing multicasting in mobile ad hoc networks is a more challenge task than that in wired networks due to frequent changes in network topology as well as the nature of the network wireless interface. While, the frequent changes in network topology implies a short validation time of routing information, wireless nature of the interface implies the limited bandwidth capacity. Thus, mobile and wireless environments exhibit opposite requirements. On one hand, node's mobility requires a high degree of routing information updates in order to maintain connectivities among senders and receivers. On the other hand, the wireless medium has low capacity and hence cannot be used for additional control traffic that is needed to continually update stale information. These properties make multicast routing protocols for Internet not adapt for MANET environment and introduce special challenges in multicasting protocol designing. Facing to the challenges of bandwidth limitation and frequent topology changes, most researchers study how to reduce control overhead to maintain the connection of multicast routing structure in MANET environment. They usually reach their goals in three ways: (a) Try to limit the effect of topology change in a small range by doing local repair, (b) Introduce more routes into the routing structure in order to make the structure robust against topology changes, (c) Reduce as many as possible the routing information which should be maintained for multicast routing. Based on these ideas, several multicast routing protocols are proposed recently.

However, multicasting is far from being well established for MANET. First, the principle idea of these three ways is to reduce the control overhead with the cost of transmission efficiency. However, we cannot consecrate too much transmission efficiency to reduce control overhead. An extreme example is to deliver multicast packets by flooding them in the network. In method, the control overhead is equal to zero since there is no routing structure, but forwarding overhead is very important. Therefore, a certain trade-off should be found between control overhead and forwarding overhead in order to reduce total overhead for multicast routing. Another important issue that usually ignored during multicast routing protocol designing is the cooperation with the other layers. For example, if the MAC layer protocol in use cannot efficiently support multicast transmission, there will be significant packets lost during delivery. Bad network situation can aggravate this problem. In the ad hoc networking scenario, it is difficult to maximize delivery success rate and minimize overhead simultaneously. Thus, there is also a trade-off between them. From the point view of applications, they do not always require the maximize delivery success rate. Some applications can tolerate a certain degree of packet loss but require short transmission delay while some others are not sensitive to transmission delay but do expect that receivers can receive as many as possible packets. Different strategies can be applied according to the diverse requirement of these applications in order to find the compromise between multicast delivery and total overhead. Thus, we give another definition of "best effort" multicasting in MANET: to deliver multicast packets with the lowest cost and as close as possible to the application's requirement.

Therefore, our research goal is to design a best-effort multicast routing protocol. This protocol is able to provide an optimal trade-off between efficient delivery and total overhead while at the same time meet the requirements of most multicast applications. Considering some applications of MANET do require a guarantee of transmitting packets to receivers, we also discuss the reliable multicasting in MANET. We believe with these efficient multicasting protocols, it will be possible to realize a number of envisioned group-oriented applications in MANET environment.

1.2 Structure of Dissertation

This Dissertation focuses on providing multicasting in mobile ad hoc network in an efficient way. Multicasting is divided into two sub issues: delivery structure management and multicast packet forwarding. The remainder of this dissertation is organized as follows.

Chapter 2 provides an overview of providing multicasting in MANET. It includes the design challenges and different techniques aiming to resolve these problems. We also present our contribution and compare it with other well-known multicast routing protocols.

Chapter 3 describes our multicast routing protocol with dynamic core (or MRDC in abbreviation) in detail. This protocol contains two plans: control plan and forwarding plan. We introduce our strategy to construct and maintain a delivery structure on traffic demand in control plan. The forwarding plan delivers multicast packets in a *best-effort* way. Considering that IEEE802.11 DCF, a widely used MAC layer protocol in MANET, does not transmit multicast packets in a suitable way, we integrate our solution in the forwarding plan. This solution chooses a suitable transmission method to deliver multicast packets by taking network situation and application requirement into account.

Chapter 4 analyzes the performance of MRDC in a network simulator ns2 [9]. The simulation first tests the correctness and robustness of this delivery structure under different environment (group configuration, network load, node's mobility). And then we demonstrate that our multicast forwarding mechanism is adaptive and can achieve better performance in network load.

Chapter 5 presents our scalable reliable multicasting protocol to offering multicast delivery guarantee in MANET. This protocol distributes multicast packet cache and retransmission tasks among intermediate nodes that overhear multicast packets. Simulation results show that our reliable multicasting protocol has a packet delivery rate close to 100% and maintains a low bandwidth consumption facing to frequent topology change.

Chapter 6 introduce our experiences in implementation an ad hoc testbed. We implemented not only MRDC but also a unicast routing protocol in the testbed. This testbed allow us to evaluate the performance of our multicasting protocol in real world. It can also be employed to concept new MANET applications since it can support one to one and many to many communications.

Finally in chapter 7, we outline remarks and summaries of our work and contribution. We discuss the general experience learned about on designing multicasting protocols for mobile ad hoc networks and outline some directions for future research.

Chapter 2

Studies of Multicast Routing Protocols for Ad Hoc Network

Multicasting is the transmission of datagrams to a group of hosts identified by a single destination address [10]. To realize this mode of transmission, three types of protocols are needed. They are group management protocols, the IP multicast routing protocols and multicast transport protocols. Since the first Internet multicast paradigm appeared in the later 1980's, numerous multicast routing protocols were well designed to offer efficient multicasting service in conventional wired networks. These multicast routing protocols, having been designed for stationary networks, may be unsuitable for mobile ad hoc networks due to the special properties of MANETs. Facing to the design challenges, several multicasting routing protocols are proposed during the last few years. These protocols coped with group membership dynamic and topology dynamic by using diverse strategies and techniques. One thing is clear that none of them is suitable in all cases. Therefore, before designing a new multicast routing protocol for MANET, it is important to:

1. Understand the properties of MANETs,
2. Study the common design challenges of providing multicast in networks and the dedicate challenges in MANET,
3. Analyze the advantage and the inconvenience of each strategy and technique and
4. Choose suitable strategies and techniques.

This chapter covers these above steps.

2.1 Issues in providing multicasting in MANETs

In this section, we first outline the specific properties of mobile ad hoc networks. After a short discussion of the multicast routing protocols for Internet, we list the design challenges for providing multicast routing for this type of networks.

2.1.1 Mobile Ad hoc Network properties

Compared to other networks, ad hoc networks has following special features that need to be studied for routing protocol designing:

Infrastructureless **-** There is no fixed backbone infrastructure for network management and packet relay. Therefore, mobile nodes become the potential network infrastructure and they must act cooperatively to handle network functions. This property yields some other special properties of ad hoc networks such as frequent topology change and limited resources.

Frequent topology change **-** The network topology may change randomly and rapidly over time since nodes are free to move in an arbitrary manner. Furthermore, networks may be partitioned and merged from time to time because of node's movement and environment change.

Wireless communication **-** Nodes use wireless interface to communicate. Wireless communication implies limited bandwidth capacity in comparison with wired communication, and differs from those by the respect that electromagnetic waves propagate in free air instead of inside cables. Many issues, which emerge from this fact such as multipath, pathloss, attenuation, shadowing, noise and interference on the channel, make the radio channel a hostile medium whose behavior is difficult to predict. Therefore, wireless communication potentially has low capacity, high collision probability, and high bit error rate.

Limited node resources **-** Node resources, which include energy, processing capacity, and memory, are relatively abundant in the wired networks, but may be limited in ad hoc networks and must be preserved. For instance, limited power of the mobile nodes and lack of fixed infrastructure restrict the transmission range and create the need for effective multihop routing in mobile ad hoc networks.

These features and their associated challenges make the multicast routing protocol design a very difficult task in such an environment. The most important characteristics which should be considered during multicasting protocol design are broadcast capacity, topological changes, high message loss rate and limited bandwidth and node resources.

2.1.2 The Design Challenges of Multicast routing for Ad hoc networks

The primary goal of multicast routing is to direct and transport packet through the network from the source node(s) to the destination node(s). To realize this operation, it contains following functionalities:

- Multicast translation - which discovers all receivers behind a multicast address in the network.

- Routing structure construction and maintenance - which establishes and maintains routes among group members.

- Packet forwarding - which transports packets from sender to these receiver(s).

Multicast routing protocols for the Internet address the issue of routing structure which connects source(s) and destinations and forward data packets using this structure. Multicast translation is accomplished during structure construction when group receivers join the structure. Through this way, these protocols focus on the problem of the minimum cost structure for packet forwarding. Here, the cost could be distance, delay, and so on. A natural routing structure for multicasting is a tree. The multicast routing protocols differ in how the multicast trees are constructed and what IP unicast routing algorithms are used when constructing the trees. Currently, there are mainly two kinds of multicast trees: source-based shortest path tree and group-shared tree. DVMRP [11], MOSPF [12] and PIM-DM [13] [14] use shortest path trees rooted at source, while CBT [15], BGMP [16] and PIM-SM [17] use group-shared tree. The shared tree in PIM-SM can be switched to a shortest path tree when needed.

Multicast routing protocols do not perform well in wireless ad hoc networks because tree structures are fragile and must be readjusted as connectivity changes. These methods generally assume that the topology of networks is stable and routing informations only accounts for a small portion of the network bandwidth. These protocols may fail to keep up with topology changes in a MANET. The frequent exchange of routing information triggered by continuous topology changes yields excessive channel and processing overhead. For example DVMRP meets *data flooding overhead* problem when it is in ad hoc networks [18]. However the limited bandwidth and node resources on one side prevent routing protocol sending too much control messages and on the other side demand protocols to well chosen routers to reduce resource consumption and fairly use node's resource. On the other hand, some protocols especially those protocols based on group shared tree use a single special node (called core or Rendez-Vous node) to construct tree. These protocol may not correctly operate in MANET due to single node failure problem which is usually caused by network partitions, node turned off, and so on.

Therefore, multicast routing for mobile ad hoc network should efficiently to deal with both group member dynamic and topology dynamic. The construction and maintenance of routing structure should be done with a reasonable routing overhead in both low and high mobility networks, while providing efficient data transmission . It is also desirable that a routing protocol to be simple, distributed, adaptive, and dynamic. In brief, in order to achieve transmission efficiency, the design challenges of providing multicast routing for ad hoc networks cover not only group membership changes, data transmission efficiency, which exist in wired networks too, but also frequent topology changes, high message loss rate, limited bandwidth and node resources, which are relative to the properties of MANETs. The transmission efficiency means to consume as less as possible network resource while the results of multicast delivery meet as much as close to the application's requirement.

2.2 Techniques in providing multicast in a MANET

Designing a new multicast routing protocol may necessitate examining the main strengths and weaknesses of each approach in mobile ad hoc network environment and comparing the different existing approaches [10], [19] according to the properties of MANETs. These approaches are around four issues:

- Protocol driven issue, the moment to run protocol;
- Multicast Structure issue, the form to connect group members;
- Routing Philosophy issue, the dependence on a unicast routing protocol;
- Forwarding issue, how to forward multicast packet

2.2.1 Protocol Driven issue

This issue replies the question of which triggers the running of multicast routing protocol operations. All researchers declare that their multicast routing protocols are on demand ([20],[21] [22], [23], [24], ...). However they are on demand in two different fashions, which we call on group demand and on traffic demand respectively. In on group demand fashion, so far as group members exist in the network, routing protocol runs to handle membership and/or topology changes. Because group members are kept connect, there is small discovery latency when a sender begins a new packet transmission. However, control messages are injected into network when there is no multicast packet being sent. That could be considered inconvenient in MANET where bandwidth is limited. On the other hand, on traffic demand fashion reduces this overhead by in the way that the operation of the protocol is driven by the presence of packets being sent. In this fashion, nodes keep in silent when they become group receivers. When a sender begins to transmit multicast packet, routing protocol starts to discover group receivers and then deliver packet. Once the transmission terminates, routing protocol stops too. Through this way, routing overhead is limited around traffic. But on traffic demand fashion introduces discovery latency. Due to the distances from sender to receivers are various, it is difficult to decide when to begin packet delivery. If the delivery starts too early, some receivers may not receive first packets for being far from sender. There are some mechanisms to alleviate this problem. For example some protocols broadcast the first packet instead of waiting all receivers are discovered.

2.2.2 Multicast Structure issue

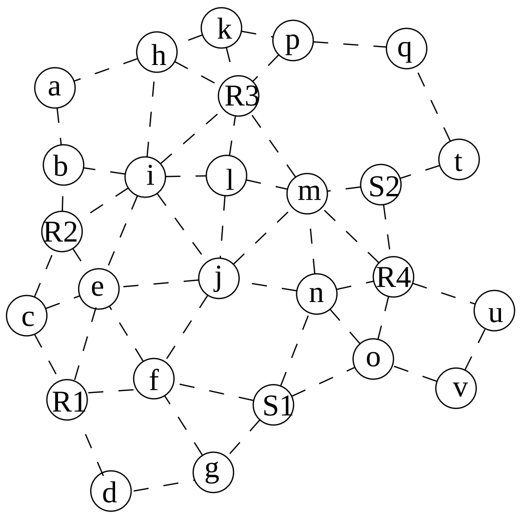
This issue discuss which type of structure is used to connect group members. Up to now there are two types of structure: tree and mesh. Tree structure is well used in wired network. It involves as less as possible intermediate nodes and contains unique path between sender and receiver pair. This structure can thus provide high data forwarding efficiency. There are two kinds of multicast trees: source-based tree and group shared tree. Source-based tree means for each group, source pair there is a multicast tree which is rooted at the source and normally constructed by the shortest from the source to the destinations. This type of tree is efficient in data transmission since packets are delivered along the shortest path to the destinations. On the other hand, a core node serves as the root of the group shared tree. Group shared tree is not efficient in packet transmission since usually source should first transmit packets to core node then core node deliver them to group receivers. However this type of tree is more efficient in the point view of routing. Multicast source just needs to explore the route to the core instead of to all group receivers in source-based tree. Only one tree need to be maintained, control overhead is relatively low. While, the idea of using the least connectivity for multicast delivery from tree structure is not necessarily best suited for multicast in a MANET. In such an environment, nodes may move in an unpredictable way which causes network topology changes frequently. Because no alternative path between a sender and a receiver, every link broken takes place on tree trigger a reconfiguration. Compared to tree, mesh structure is robust against topology changes. Mesh is a subset of graph that may have multiple paths between any source and receiver pair. These alternative paths allow multicast packets to be delivered to the receivers even if links fail. Link failure may not trigger a reconfiguration. In brief, tree-based approaches provide high data forwarding efficiency at the expense of low robustness, whereas mesh-based approaches provide robustness and low routing overhead at the expense of higher forwarding overhead and increased network load.

Figure2.1 illustrates how tree and mesh connect group members in a MANET. This group contains two senders (S1 and S2) and four receivers (R1, R2, R3 and R4) distributed in a MANET. In Figure2.1(b) a tree rooted at S1 consists of 4 intermediate nodes connects these six group member. And in Figure2.1(c), a mesh which contains the shortest path between any pair of sender and receiver involves 7 nodes to connect group members. If any node among four intermediate nodes of tree fails, tree should be reconfigured. While, mesh has not this problem. If we delete anyone of these 7 nodes, mesh keeps connecting.

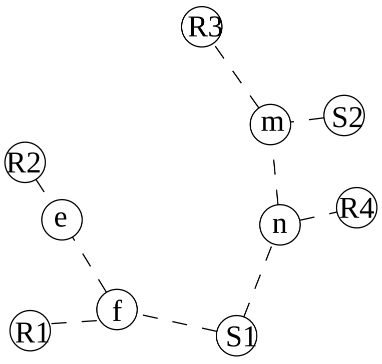
During structure construction and maintenance, intermediate nodes are explicitly invited to join structure. While for leaving they have two solutions, using soft state, in which if the structure membership is not updated before a timer out, node leaves structure automatically, or using hard state, in which node deactivates its structure membership upon receiving certain control packets. Control packets might be lost during their transmission. Thus, a more general way is to use the combination of these two solution for deactivating structure membership

2.2.3 Routing Philosophy issue

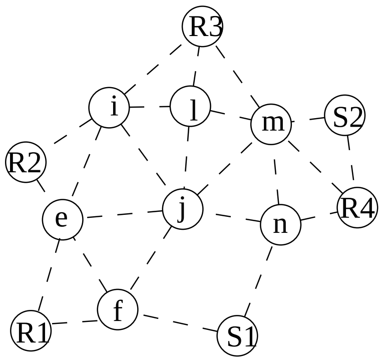
Multicasting in MANET should face to membership dynamic and topology dynamic. Different to the case of Internet, where unicast routing protocols have been well studied before multicast conception was proposed, in MANET the hot research of these two kinds of routing begins at nearly the same time. That is why we



(a) A multicast group in a MANET



(b) Tree, rooted at S1



(c)Mesh, contains shortest path

Figure 2.1: Example of tree and mesh to connect group members

can find some protocols rely on unicast routing protocol hoping this protocol suitable in most case while others are independent. According to the dependence on an underlying unicast routing protocol, we can classify multicast routing protocols to simple multicasting (full dependent), median multicasting (median dependent) and all-in-one multicasting (independent).

The idea of simple multicasting is that multicast routing protocol discovers group receivers and unicast routing protocol provides the routes to concerned receivers in intermediate nodes. Aggregating these routes, multicast protocol deliver packet. This solution simplifies the protocol design in terms that multicast protocols focus on group membership dynamic and requires no multicast routing information other than group state information to be maintained in network.

On the contrary of simple multicasting, all-in-one multicasting is completely independent of any unicast routing protocol and realize all functionalities by multicast routing protocol itself. When discovering group members, protocol can probe the route to these members and construct a delivery structure at the same time. Thus the control overhead can be reduced via aggregating messages of these two functionalities. Instead of needing unicast routes to certain destinations, intermediate nodes just needs to know its delivery structure neighbors or its state on delivery structure. Then, the multicast packet is forwarded on this delivery structure. Topology changes on one side cause link failure which multicast protocol should repair, and on the other side create better routes which multicast protocol should include into structure to get better performance. For this aim, this solution requires that multicast protocols periodically probe network topology. However, periodical probe reacts slowly to topology changes and cannot adapt to various frequency of topology change. Furthermore, some link failures generate important effect on multicast delivery. Local recovery mechanism is studied to overcome this shortcoming. In this mechanism, structure members survey the links to its structure neighbors and immediately repair a link failure in local area.

A compromise of simple multicasting and all-in-one multicasting can be found in median multicasting. This solution needs unicast routing protocol to construct and maintain delivery structure, while packet forwarding is done via delivery structure. This solution requires less unicast routing information than first solution in intermediate nodes (normally only route to core or sender). This solution is simpler than the second solution since unicast routing protocol shares routing function. It just needs to modify structure according to route changes (detected by unicast routing protocol) and membership change. However, the protocol's performance is greatly depends on the correctness and efficiency of the unicast routing protocol in use.

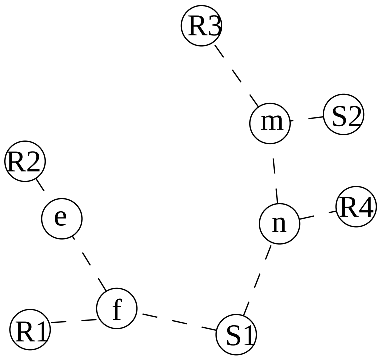
2.2.4 Forwarding issue

This issue addresses how to deliver multicast packets by using the multicast structure or the unicast routing protocol. In a tree structure, the general way is that a node takes packets from nodes with whom a tree branch has been established and forwards it to other branches than the incoming one. On the contrary packets are normally flooded in mesh in the way that a mesh member can accept unique packets coming from any neighbor in the mesh and them send them. Unique packets mean the packets that a node never received before. We will explain the mechanism to judge unique packet later. As we explained in section 2.2.2, tree structure generates less forwarding overhead while is not robust against topology changes and mesh can resist to link failure but is not efficient in data forwarding. Some techniques emerge to improve packet delivery in these two structures respectively. For example, one method to improve robustness of tree structure is to flood packet on tree to benefit broadcast nature of wireless interface. The idea is, instead of limiting packet forwarding along tree branches, that nodes forward packet on broadcast and tree members accept unique packets from any neighbor node to achieve alternative path. Figure2.2 compares these two forwarding method on tree. Flooding on tree implicitly adds three links into the tree: link between node m and R4, between S2 and R4 and between node e and R1. If the link between node n and R4 breaks, due to the link between node m and R4 and link between S2 and R4, R4 can still receive packet from S1and S2. While for mesh structure, the question is how to reduce forwarding overhead. Thus one possible way is to select routes on the mesh to form a sub graph and then send packet in this sub graph. Figure2.3 illustrates this idea. In this example, four nodes (node e, f, m and n) are chosen to forward traffic sent by S1 and nodes e, i, l and m are invited to transmit packet generated by S2. Thus, only four intermediate nodes are needed instead of seven nodes in the case of using mesh directly.

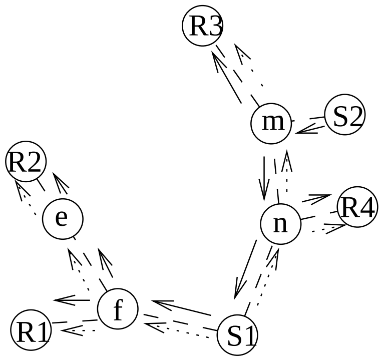
When forwarding multicast packets in MANET, one difficulty is to assure the packet in process is unique because multicast packet forwarding in MANET is different to that in traditional networks. In traditional networks, router receives multicast packet from one network interface and sends them to another interface(s). This is not the case with MANET. One node in MANET can use the same interface talking to any neighbor on the same wireless channel. This property imposes another method to avoid sending same packets multiple times. This is more important when protocol floods packet in delivery structure. The general method is to utilize sequence number for duplication detection. When sender generates a new packet, it assigns a sequence number to the packet. This sequence number along with source and destination identifications uniquely identify a packet in the network. A node register these information of packets it sends to detect duplication.

2.3 Current ad hoc multicast routing protocols

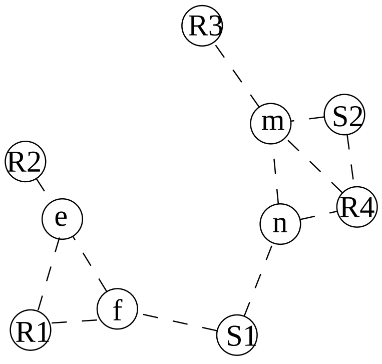
Although the advent of Defense Advanced Research Projects Agency (DARPA) packet radio networks addressed in the early 1970s [25], multicasting for mobile ad hoc networks becomes a topic of active research only during last five years. The main reason appears to be a popular belief that similar to the evolution of Internet routing multicast routing in MANET will be built on top of the unicast routing



(a) Tree

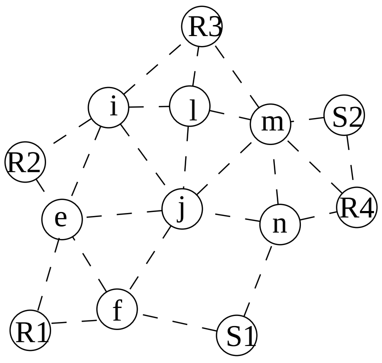


(b) Forwarding restrict on tree

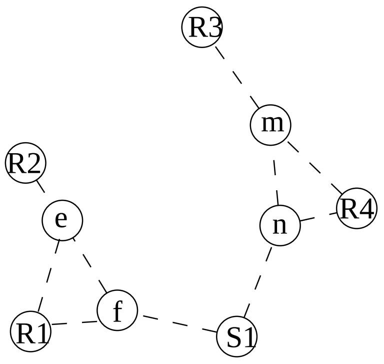


(c) Flooding on tree

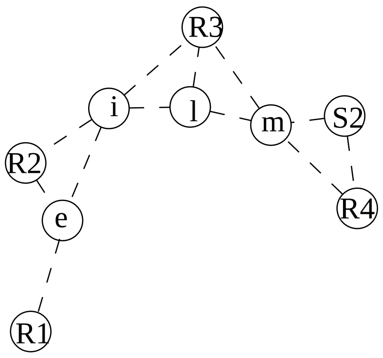
Figure 2.2: Packet forwarding in tree-based approaches



(a) Mesh, base graph



(b) Subgraph for S1



(b) Subgraph for S2

Figure 2.3: Example of subgraphs of mesh

protocols. For this reason, most research has focused on solving the unicast routing issues in mobile ad hoc networks and then, based on these unicast routing infrastructures, multicast routing protocols are proposed. They are Ad hoc Multicast Routing protocol (AMRoute)[26], Core-Assisted Mesh Protocol (CAMP) [20], [27] Differential Destination Multicast (DDM)[21] and Lightweight Adaptive Multicast Algorithm (LAM)[28].

However, other researchers believe that because of the broadcast capacity of wireless nodes, mobile ad hoc networks are better suited for multicast, rather than unicast, routing and, that it is more effective to solve the multicast routing problem separately [10]. They designed their mutlicast protocols independent of any unicast routing protocol. They either extends the principle ideas of unicast routing protocols for MANET, which is the case of Associativity-Based Ad hoc Multicast (ABAM) [29] (from Associativity-Based Routing (ABR) [30]), Multicast operation of Ad-hoc On-demand Distance Vector routing protocol (MAODV) [23] (from Ad-hoc On-demand Distance Vector routing protocol (AODV) [31]) and Multicast Core Extraction Distributed Ad hoc Routing (MCEDAR) [32] (from Core Extraction Distributed Ad hoc Routing (CEDAR) [33] [34]), or develop a new multicast routing protocol, which is the case of Adaptive Demand-driven Multicast Routing (ADMR) [35], Ad hoc Multicast Routing protocol utilizing Increasing id-numberS (AMRIS)[22], Forwarding Group Multicast Protocol - Receiver Advertising and Sender Advertising (FGMP-RA, FGMP-SA) [18], On-Demand Multicast Routing Protocol (ODMRP) [24], [36] and Neighbor Supporting ad hoc Multicast routing Protocol (NSMP) [37].

2.4 Our Contributions

A new multicast routing protocol Multicast Routing protocol with Dynamic Core (MRDC) [38], is proposed. In this protocol, a hybrid multicast tree is constructed on the demand of traffic. By rooting tree at the first source of a multicast session, the multicast tree becomes hybrid: in single source group, it is source-based tree while in multiple sources group, it is group-shared tree. Tree structure faults are temporary tolerated and periodical tree refreshing removes these errors and adapts tree to current topology. Through this design principle, the control overhead to construct and maintain tree structure remains reasonable low, while the transmission efficiency of multicast tree is maintained. As a result low total potential bandwidth consumption for multicast delivery could be obtained.

MRDC use an adaptive mechanism to forward multicast packets according to network situation and application requirements. In fact two transmission modes are defined in MRDC. One transmission mode is similar to flooding in the tree. In this mode, MRDC considers tree structure as mesh and the interior nodes forms forwarding group. Then, multicast packets are flooded in the structure without respecting the tree structure. In the second transmission mode, multicast packets are transmitted along tree edges with certain degree of reliability and with the cost of bandwidth and transmission delay. The former mode is suitable in congested networks and applications which are sensible to transmission delay but can tolerate transmission errors such as voice conference. The later mode is preferable in non-congested networks for applications which are not sensible to transmission delays such as news group. In this way, MRDC provides the best effort multicast routing by considering both application requirements and network condition.

As for the applications that require a multicast delivery guarantee, a reliable multicasting protocol, named active reliable multicast protocol with intermediate node support (ARMPIS) [39], [40], is designed on the top of MRDC. This protocol distributes the responsibility of multicast packet storage and retransmission to group receivers as well the multicast routers.

2.5 Classification

Table 2.1 classifies the current multicast routing protocol according to the criteria proposed in Section 2.2. Because the third issue, routing philosophy issue, discusses how to construct and maintain multicast structure, we employed the term (Re)Configuration in the table, which abbreviates configuration and reconfiguration. As demonstrated by this classsification, none technique can outperform than others and adapt to all situation. Most protocols are designed with a predefined situation (for example: node's mobility and multicast group size) and then choose or develop suitable techniques corresponding to that situation. However, as the definition of ad hoc network, the network situation can be changed arbitrarily and the same as for group configuration. We think it is important to study an optimal multicasting protocol which can smartly choose techniques to adapt to most situations.

Protocols	Driven	Structure	(Re)Configuration	Forwarding
ABAM	Traffic	Source Tree	Independent	Not mentioned
ADMR	Traffic	Source Tree	Independent	Flooding in Tree
AMRIS	Group	Group Tree	Independent	On tree
AMRoute	Group	Group Tree	Full-dependent	On tree
CAMP	Group	Mesh	Demi-dependent	Flooding in Mesh
DDM	Traffic	Source Tree	Full-dependent	broadcast
FGMP-RA	Group	Mesh	Independent	Flooding in Mesh
FGMP-SA	Traffic	Mesh	Independent	Flooding in Mesh
LAM	Group	Group Tree	Demi-dependent	On tree
MCEDAR	Group	Mesh	Independent	Forwarding tree
MAODV	Group	Group Tree	Independent	On tree
NSMP	Traffic	Mesh	Independent	Flooding in Mesh
MRDC	Traffic	Hybrid Tree	Independent	Adaptive
ODMRP	Traffic	Mesh	Independent	Flooding in Mesh

Table 2.1: Classification of current multicast routing protocols

Chapter 3

Multicast Routing Protocol with Dynamic Core (MRDC)

We study a multicast routing protocol called multicast routing protocol with dynamic core (MRDC) for MANETs. MRDC consists of two plans, a control plan and a forwarding plan. The control plan constructs and maintains a structure to connect multicast group members on traffic demand. The forwarding plan transmits multicast packet using this structure to provide best effort delivery. Here, the best effort delivery means that routing protocol delivers packets as close to the application requirements as possible regarding network situation. Since nodes could act as host and traffic forwarder as well, in the sequel, we define a router as a node which transmits the traffic packets generated by itself or other nodes. A multicast group member may be at the same time a multicast router for that group.

This chapter is organized as follows: Section 3.1 introduces our assumptions during protocol design. Section 3.2 presents the protocol architecture and the main design principles of MRDC. Section 3.3 describes in detail the control plan of MRDC including the creation and maintenance of a multicast tree. In section 3.4, we discuss some problems of multicasting in IEEE802.11 and propose some solutions. Then, Section 3.5 presents in detail the forwarding plan of MRDC with a special emphasis on an adaptive forwarding mechanism for IEEE802.11 ad hoc networks. Finally, Section 3.6 closes this chapter with concluding remarks.

3.1 System model

During the design process, we will suppose that nodes in MANET are uniquely identified by some kind of identification such as their IP address. All nodes communicate on the same shared wireless channel and wireless links among nodes are symmetric, which means that the transmission characteristics of the transmitter and receiver of data on the link are identical. For instance, if node a can hear node b, node b can also hear node a.

Moreover, each multicast source binds on a communication port for transmission and assigns a reference number to each multicast packet it sends. This reference number, including source address and port number, can be considered as a packet's unique identifier in the network.

3.2 MRDC Architecture and Design Principles

3.2.1 MRDC Architecture

Contrarily to most multicast routing protocols which combine multicast packet forwarding with delivery structure construction and maintenance, Multicast routing protocol with dynamic core (MRDC) is divided into a control plan and a forwarding plan, as shown in Figure 3.1. The control plan deals with the construction and maintenance of multicast delivery structures, while the forwarding plan copes with how to forward multicast packets generated by the node itself or by other nodes. This architecture allows us to concentrate on studying an optimal routing strategy to reduce global bandwidth consumption while adapting to network topology changes, and then design an adaptive multicast transmission policy regarding network situation and application requirements. The control plan works in a passive fashion and is driven by the forwarding plan. In fact, the forwarding plan triggers the control plan to collect and update multicast routing information. Thanks to this routing information, the forwarding plan is able to deliver multicast packets generated by any node to their final destinations. The control plan is somewhat lower layer independent in the sense that physical layer and MAC layer have little influence on the result of delivery structure. Conversely, the question of how to forward multicast packets hop by hop to their receivers is closely relative to the MAC layer in use. Considering IEEE 802.11 [41] is preferred by MANETs, we develop an adaptive multicast forwarding mechanism in the forwarding plan of MRDC that provides a best effort multicast delivery in IEEE 802.11 ad-hoc networks.

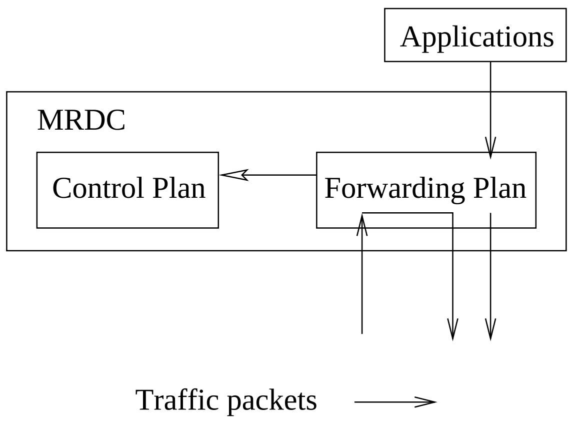


Figure 3.1: MRDC architecture

3.2.2 Control Plan Design Principles

The bandwidth consumption of a multicast routing protocol comes from routing overhead and transmission overhead. The aim of the MRDC control plan is to find a trade-off between routing overhead and forwarding overhead in a way to improve bandwidth utilization efficiency. The optimal strategy to get to this goal is to use the most efficient data transmission technique since traffic packets are usually bigger and more numerous than control messages and consequently consume much more bandwidth. Therefore, we try to reduce the routing overhead of this technique at a limited cost of traffic transmission efficiency.

In terms of traffic transmission efficiency, which means generating as little multicast forwarding overhead as possible, MRDC uses a tree structure in a way to limit the number of routers involved in the delivery structure. Extra nodes are invited to join the delivery structure only when the topology changes require the protocol to do so. The tree is periodically reconfigured to better fit to current topology.

MRDC uses a group-shared multicast tree on which the root, which we call core in this dissertation, is the first source of the multicast session or the source which wins the core competition. Group-shared tree is less efficient than source-based tree in terms of multicast packet delivery but needs less routing overhead for group members to join a multicast group and the maintenance of the delivery structure. Multicast tree is constructed on traffic demand so that control overhead is limited around the traffic. Another factor in reducing multicasting control overhead is that MRDC tolerates temporary fault in the multicast tree.

MRDC uses core to limit control overhead. However, the core concept used in MRDC differs from other group-shared tree-based multicast routing protocols for MANETs, which also use this concept (AMRIS [22], CAMP [20] [27], LAM [28] and MAODV [23]). In our protocol, core is initially the first sender of a multicast session and then transferred to another sender in some situations. This choice on one side guarantees that tree is constructed and maintained on traffic demand, core is a router (it should at least send multicast packets generated by itself) and on the other side provides flexibility facing to the number of sources in a multicast group and adapts to network topology. In a single-sender multiple-receiver session, it creates a source-oriented tree. For a multiple-sender multiple-receiver application, it offers a group-shared tree. If MRDC detects that core has ended the transmission or any other cases (e.g. battery level of core is too low to allow it to relay multicast packets), MRDC can transfer the core role to another sender. Therefore even when the network is partitioned, receivers can always receive packets sent by the senders in the same partition because MRDC designates a sender as core in each partition. In this way, MRDC can also prevent single node failure problems. Once these partition merge into one, the corresponding trees also merge into one tree which root is designated through a core competition among senders.

Most tree-based multicast protocols demand that tree structure be kept coherent and loop-free so that transmission efficiency can be preserved. This requirement implicitly increases control overhead. However, MRDC proposes to temporarily tolerate faults in the multicast tree as a way to reduce such routing overhead. For example, in CBT and MAODV, a branch break results in the dissolution of the concerned sub-tree and all receivers in the sub-tree should rejoin the multicast tree individually. In MRDC, when such a break is detected, routers try to find another route to replace the broken one without taking care of fault forming in the tree or losing a little transmission efficiency compared to the mechanism used in CBT and MAODV. In a tree structure, fault usually means tree fragmentation or loop. If the tree is logically fragmented but is still physically connected, multicast forwarding can continue on it. As for duplicate forwarding caused by loops, it can be avoided through a duplication table. Furthermore, a periodical tree refresh mechanism removes faults through destroying old trees and constructing new ones.

3.2.3 Design Principles of the Forwarding Plan

The aim of the forwarding plan or in a more general term, the multicasting protocol is to deliver multicast packets in a best effort way. Yet, this does not depend only the delivery structure but also on the MAC layer protocol in use.

Most mobile ad hoc networks give preference to IEEE 802.11 as MAC sublayer protocol since it is regarded as a standardized protocol which allows terminals to share the wireless channel through ad hoc configuration. This protocol primarily targets unicast communications and, up to this time, does not efficiently support multicast transmission. According to the IEEE 802.11 specification, MAC protocol broadcasts multicast packets. The multicast sender simply listens to the channel and then transmits its data packet when the channel remains free for a period of time. There is neither MAC-level acknowledgment nor recovery procedure for a multicast packet. As a result, once a collision occurs due to problems such as hidden terminal, etc., the packet cannot be recovered, which degrades the performance of multicast routing protocol even when the network load is low.

In order to improve multicast packet transmission using current IEEE 802.11 standard, one method is that members of a delivery structure unicast multicast packets when it is possible. By doing so, the network layer transmits multicast packets point-to-point to selected neighbors using RTS/CTS option in order to avoid the hidden terminal problem. However, this method consumes more bandwidth since a node sends multiple copies instead of a single one in the broadcast way. This method is useful when the medium is not congested and with the condition that it should not create congestion. Therefore, a mechanism is needed to smartly choose the forwarding method. Following the above ideas, we design an adaptive data forwarding mechanism in forwarding plan for IEEE 802.11 MANETs. Two forwarding modes are defined in this mechanism: unicast mode and broadcast mode. Unicast mode consists in treating one multicast packet as multiple unicast packets and sending them with IEEE 802.11's RTS/CTS option, thus avoiding the hidden terminal problem. The broadcast mode is to pass multicast packets directly to IEEE 802.11 layer to reduce bandwidth consumption. This forwarding mechanism is adaptive because it makes a choice between two forwarding modes according to network load and tries to avoid congestion. To achieve this goal, our approach primarily uses the Average Queue Length (AQL) as its mode selection metric. AQL is the mean MAC queue length, it represents the difficulty of sending packets into the network. Because queue increase is usually the result of high network load and since mode change has an important effect on queue length, using AQL can control but cannot prevent congestion. The forwarding mechanism employs another metric called Medium Occupation for Reception (MOR) using some MAC layer statistics provided by most 802.11 implementations. MOR is defined as the percentage of the time where MAC is busy in reception over a period. Accordingly, MOR does not include the traffic generated by a node itself in order to avoid the impact of any forwarding mode change. Thus MOR serves as the network load criterion and AQL as congestion level criterion.

3.3 MRDC Control Plan Description

In this section, we introduce in detail the control plan of MRDC. MRDC adopts an on-traffic-demand tree to connect group members. A multicast tree is rooted at the first source of a multicast session. The control part of MRDC consists of two aspects: Tree construction and Tree maintenance. Tree construction is the aspect by which a core is selected and advertised to the network. Nodes that are interested in the multicast session are able to join the tree. Tree maintenance is the aspect where tree members detect broken branches and attempt to repair them in order to continue multicast traffic delivery in multicast tree. Multicast tree maintenance also takes care of receivers eager to leave the group. Nodes use MRDC messages to exchange routing information, which is stored in MRDC tables.

3.3.1 Messages and Tables

The messages used by MRDC to exchange multicast routing information among nodes have the format shown in Figure 3.2. Thus, we will consider them as MRDC message. A MRDC message contains five fields: the type of the MRDC message (Type), a reserved field (Reserved) for future use, reference number (REF), group ID number (GID) and node ID number (NID). The field $< Type >$ indicates which kind of MRDC control message the packet carries. The field $< Reserved >$ permits different type of MRDC message has their own usage or for future extension. The field $< REF >$ contains a reference number which is assigned by the sender and used for duplication detection if the message is a broadcast one. GID represents the ID number of the group that this message concerns. NID is the ID number of some node, which depends on the message type.

Each node in MANET possesses four tables: multicast routing table (denoted as MRTable), unicast routing table (denoted as URTable), duplication table and active neighbor table. The multicast routing table stores multicast routing information. It contains six fields: group ID number (GID), core ID number (CID),

Type	Reserved	REF	GID	NID

Figure 3.2: Structure of a MRDC message.

reference number (REF), upstream node ID number (UID), downstream node ID numbers (DIDs), state and last update time (LUT). GID represents the ID number of the group that this entry concerns. CID is the ID number of the core. REF stores the last reference number assigned by the core. UID is the ID number of the direct upstream node on the tree. DIDs saves the ID number of direct downstream node(s) on the tree. The state field indicates the current state of a multicast routing entry. It commands the behavior of that node for multicast forwarding. A multicast routing entry has three states: on-tree, tree-fault and non-forwarder. If a node has an on-tree state multicast routing entry, it means that the node belongs to the multicast tree and all branches to its direct tree neighbors are correct. Tree-fault also means that the node is a multicast tree member but some branches might contain contain errors. Non-forwarder state signifies that the node is not a multicast tree member. This is the default state of a multicast routing entry. For a non-tree-member node, the UID and DIDs fields are empty. The LUT represents the last update time of a multicast routing entry. It is used to purge any expired information out of the table. The multicast routing table of node x is shown in Table 3.1, and it is denoted by $MRTable_x$. A multicast routing entry of group G is denoted by $MRTable(G)$

GID	CID	REF	UID	DIDs	STATE	LUT

Table 3.1: Multicast Routing Table

A unicast routing table maintains the routing information necessary to reach other nodes in the network. Initially, it is used to forward some unicast MRDC messages to their destination. However if any other unicast routing protocols for ad hoc networks ([31], [42], etc.) are present, they are also given read and write rights in order to share routing information. This table holds three fields: destination address (dst@), next hop address (hop@) and last update time (LUT). Similar to the MRTable, LUT is used to purge the expired information out of the table. Unicast routing tables are generally modified and used by MRDC.

dst@	hop@	LUT

Table 3.2: Unicast Routing Table

A duplication table, illustrated in Table 3.3, saves the packet header information for duplication detection during broadcast and multicast packet forwarding.

It contains three fields: source address (src@), port number (port #) and reference number. The reference number is assigned by the source before sending the packet. These three fields uniquely identify a multicast packet in the network. As for MRDC messages, only source address and reference number are enough since only one MRDC process runs on each node.

src@	port #	REF

Table 3.3: Duplication Table

An active neighbor table is used to restore one hop neighbor nodes which are currently active. This table has two fields (see Table 3.4): a neighbor node address (nid@) and a last update time (LUT). When a node receives a control message or a traffic packet from a neighbor node (which means the neighbor node is active to send, receive and/or forward), it updates the corresponding entry and set the LUT as the current time. If an entry is not updated in a given time, the entry is removed from the active neighbor table.

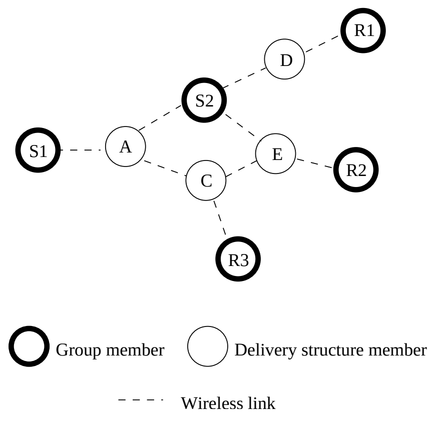
nid@	LUT

Table 3.4: Active Neighbor Table

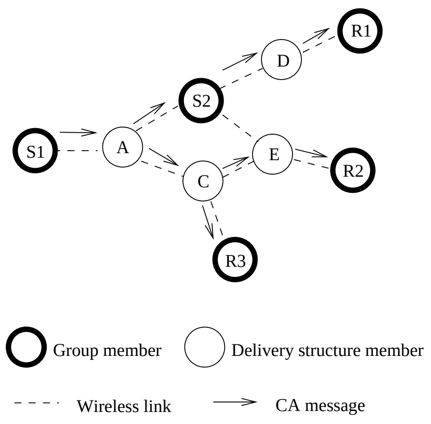
3.3.2 Tree construction

Multicast tree construction is initiated when the first source of a multicast session appears in the network. This source becomes core and broadcasts a Core Advertisement (CA) message to the network. Upon receiving this message, other group members send back a Route Active Request (RAR) message through the reverse path. When receiving RAR message, core or an active tree member replies to the request with a Route Active Acknowledge (RAA) message. When RAA message is transmitted to its destination, the nodes on the route become active tree members. In this way, the multicast tree is constructed.

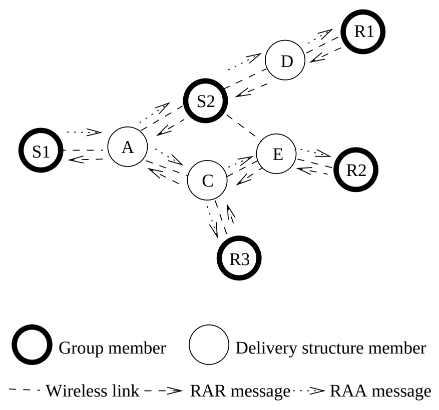
Figure 3.3 gives an example of multicast tree construction under MRDC. Initially there are two multicast sources and three multicast receivers in the network (Figure 3.3(a)). Source S1 appears earlier than source S2. Therefore S1 becomes core and broadcasts a CA message to the network as shown in Figure 3.3(b). Upon receiving the CA message, group members (S2, R1, R2 and R3) sends back RAR message and core replies with RAA messages to activate the corresponding links. This procedure is illustrated in Figure 3.3(c). In fact, during RAR message forwarding this type of message is aggregated at nodes S2, C and A to reduce bandwidth consumption. Finally the multicast tree is constructed as demonstrated in Figure 3.3(d).



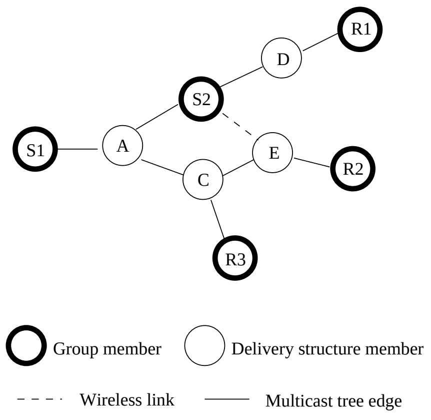
(a) Multicast group



(b) CA message propagation



(c) RAR and RAA messages transmission



(d) Final multicast tree

Figure 3.3: Multicast tree construction

CA, RAR and RAA messages are MRDC messages. A CA message is a MRDC message in which the $< Type >$ field is set to CA. The $< Reserved >$ field is not used in this type of message and consequently is set to zero. The REF field contains the reference number assigned by the core. The GID field is the ID number of the concerned group and the NID field contains the ID number of the core. The reference number, together with the core ID number, uniquely identifies a CA message in the network. RAR and RAA messages are similar to CA messages with the $< Type >$ field set to RAR and RAA respectively. The difference lies in the REF field and the NID field. The REF field of RAR messages and RAA messages is the reference number stored in the multicast routing entry $MRTable(GID)$. In other words, the last reference number assigned by core that the node is aware of. The NID field of these two types of MRDC messages is the ID number of the node which generated the message or the last node which processed the message.

A multicast routing entry in the MRTable means the presence of multicast traffic and multicast tree for the group GID. Its existence determines the behavior of a node when this node activates the membership of group GID. There are four possible cases:

1. A node becomes group receiver when the corresponding entry does not exist. In this case the node remains silent remains silent until the corresponding entry is created.

2. A node becomes group source when the corresponding the entry does not exist. In this case, this node considers that it is the first source of the multicast session. It becomes core and initializes the multicast tree construction.

3. A node becomes group receiver when the corresponding entry exists.

4. A node becomes group source when the corresponding entry exists.

In the two later cases, the node sends a RAR message to join the multicast tree.

When a multicast source of group GID becomes core, it generates a reference number and creates an entry $MRTable(GID)$ in the MRTable with the necessary information. Then core initiates a CA message and broadcasts this message to the network. Upon receiving a CA message, any node first passes the core ID number and reference number through the duplication table in order to test whether the message is a duplicate or an original. In the former case, the node discards the duplicated CA message. In the later case, the duplicate table stores the core address and the sequence number. Then the node creates an entry $MRTable(GID)$ in its MRTable with all essential information such as Core ID number and reference number. It records the ID number of the node from which it received the CA message as the next hop node towards the core in the unicast routing table. This route would be used to send or forward RAR messages. Then it propagates the CA message to its neighbors. Therefore, other nodes are able to hear of the creation of a multicast session and are able to establish a reverse path to the core.

When it is a group member that receives a CA message, this group member generates a RAR message with the NID field set to its ID number. Then it sends this message to the next hop on the reverse route to the core. When receiving a RAR, a node first compares the reference number of the RAR message to that in the corresponding multicast routing entry $MRTable(GID)$. The difference of reference numbers indicates incoherent routing information and the node will stop processing the RAR message in order to avoid the formation of an erroneous tree. On the contrary, if these reference numbers are identical, the node adds NID into the DIDs field of the multicast routing entry to register this potential downstream node. Then, it replaces the NID field of RAR message with its ID number and sends this RAR message to the next hop towards the core. In this way, a RAR message is forwarded hop by hop till reaches the core or a active multicast tree member. As a RAR message propagates through the network towards the core, a potential multicast tree branch is formed. This type of branch begins at a potential multicast tree leaf that is a group member and ends by either the core, or an active multicast tree member, or a node which is waiting for active acknowledgment.

When the core or a multicast tree member receives a RAR message, it processes the message as previously described. However, instead of forwarding the RAR message, it replies with a RAA message to activate route entries of nodes belonging to the potential branches. When a node receives a RAA message while it was waiting for such acknowledgment, it changes the state of $MRTable(GID)$ from *non-forwarder* to *on-tree*. It takes the node from which it received the RAA message as its upstream node by recording the ID number of that node in the UID field of $MRTable(GID)$. Then the node replaces the NID field of the RAA message with its ID number and forwards the RAA message to all nodes registered in the $< DIDs >$ field of $MRTable(GID)$. At last the RAA message arrives at the potential multicast tree leaf. Thus the multicast delivery tree is constructed and the source can use this tree to transmit packets.

The state change of multicast routing entry in the core is different from other nodes since it does not receive RAA message. Core uses the first received RAR message to activate the corresponding entry. However instead of doing it immediately, core waits for a while before changing the state of $MRTable(GID)$ from *non-forwarder* to *on-tree*, when it receives the first RAR message. This period of time permits multicast tree to be completely constructed before the beginning of multicast packet delivery.

In order to aggregate RAR messages, after sending a RAR message, each node starts a timer and waits for a RAA message. The node does not send or relay any further RAR messages before this timer expires. However it still processes the RAR messages by recording their NID fields into the $< DIDs >$ field of $MRTable(GID)$. The timer is stopped when the node receives a RAA message. Due to unpredictable reasons, for example RF interference, congestion, topology change during control message propagation, RAR and RAA messages might be lost. In this case, the node has not received any RAA message when the timer expires. Then if the node is not a group member, it resets the timer to be ready to forward another RAR message. As for a group member, it makes a second attempt by re-sending a RAR message. If the attempt still fails, the member will wait for the next incoming CA to join the multicast tree.

3.3.3 Tree maintenance

MRDC tree maintenance is composed of four parts: local tree recovery, periodical multicast tree refresh, group members departures and core competition. Local tree recovery reacts quickly to link failures and demands little routing overhead. On the other hand, periodical multicast tree refresh overcomes the link failures that cannot be solved by local tree recovery and gives an optimal tree structure. Periodical tree refresh gives members a chance to implicitly leave group. However they can explicitly leave by sending a message. Core competition deals with multiple core existing in the network which is normally a result of network partitions disappearance.

Local tree recovery

Multicast tree members may become disconnected from their upstream node or from some downstream node when nodes move in the network or when wireless transmission conditions change. Once a disconnection is detected, tree members execute local tree recovery to resume interrupted multicast delivery. In the local tree recovery procedure, the upstream node broadcasts a Joining Invitation (JI) message to *n-hops* away to discover a recovery route to the lost downstream node. Then the lost downstream node uses this route to rejoin multicast tree when receiving the JI message.

Tree members use *active neighbor table* to detect disconnection. After having forwarded a multicast packet, a tree member checks whether its upstream node and downstream nodes are all in its active neighbor table. If a tree neighbor is not in the table, a disconnection is detected. Recall that an entry of the active neighbor table is updated when a node receives a traffic packet or a control message from the corresponding neighbor. If an entry is not updated for a given time, it is removed. Since traffic usually flows from the root to the leaves in a tree structure, a tree member just needs to notify its presence to the upstream node. For this aim, every NEIGHBOR_HELLO period, tree members check whether they have broadcast some packet or successfully sent at least one packet to their upstream nodes during the last period. If it is not the case, they broadcast a hello message to make their upstream nodes aware of their presence.

When a broken tree edge is detected, a node runs a different sort of local tree recovery to handle this problem depending on its level in the tree structure.

If an upstream node detects a downstream link broken, this node sends a JI message to explore a route through which the lost downstream node can rejoin the multicast tree. A *Joining Invitation* (JI) message is a MRDC message with the $< Type >$ field set to JI and the NID field set to the lost downstream node's ID number. It is then broadcast *n-hops* away using an Expanded Ring Search (ERS) procedure [31] to control the message propagation. Initially, n is set to 2. This means only nodes within two hops away from the sender will receive the JI message. After a certain elapsed time, if the node does not hear from the addressed node, it increases n by one and rebroadcast the JI message. The node repeats this procedure until either it receives a recovery message from the addressed node or n reaches a maximum value, called *Greatest-Range*. Similar to the CA message, when a JI message propagates through the network, a reverse path to the message's sender is established. Upon reception of the first JI message, the lost downstream node replaces the upstream field (UID) of the respective multicast routing entry with the next hop on the reverse path, sends a recovery message and discards subsequent JIs message. The recovery message addresses the JI message's sender and its NID field is set to the lost downstream node's ID number. When a node receives a recovery message, it extracts the next hop information and compares it with the upstream field of the correspond entry $MRTable(GID)$. There are three cases. The first case happens when the node is not a multicast tree member (the state of $MRTable(GID)$ is *non-forwarder*), it therefore adds the ID number stored in the NID field of the recovery message into the $< DIDs >$ field, sets the UID field to the next hop node and activates the entry as *on-tree* state. The second case takes place when the node is a tree member and the upstream node coincides with the next hop node, the node simply adds NID into the $< DIDs >$ field. In the last case, when the node is a multicast tree member but the upstream node is not equal to the next hop node, the node sets the entry's state to *tree-fault* and changes the type of the MRDC message to *recovery_fault* type. Upon reception of a *recovery_fault* MRDC message, further nodes will only set their state of $MRTable(GID)$ to *tree-fault* and will not add the NID of the MRDC message to the $< DIDs >$ field. Finally, the node replaces the NID with its ID number and forwards the message to the next hop. This step is repeated until the recovery message arrives at its destination.

If it is the downstream node that detects at first the link failure, it triggers a timer before taking any action. If it does not receive any JI message before the expiration of the timer, this tree member removes the upstream node from the multicast routing entry and sets the state to *tree-fault*.

Let's see why we need the *tree-fault* state. The discovered routes which would be used to replaces broken tree edges can be classified into three cases, as illustrated in Figure 3.4.

- The first case (Figure 3.4 (a)) is when no node belong to the current multicast. In this case, MRDC can safely add the route to the multicast tree by setting the state of these node to *on-tree*.

- The second case (Figure 3.4 (b)) is when some nodes on the route are also tree members but none of them belong to the sub-tree rooted at the downstream node of the broken edge.

- The third case (Figure 3.4 (c)) is when at least one node is a member of the sub-tree.

In the two later cases, instead of sending extra control message to maintain a correct tree structure, MRDC tolerates these errors by setting the state of corresponding nodes to *tree-fault* and not adding the corresponding nodes to the downstream node list of their upstream node. The multicast tree is logically fragmented but physically connected with the *tree-fault* state since they are still within coverage range of their upstream nodes, therefore allowing multicast delivery to continue.

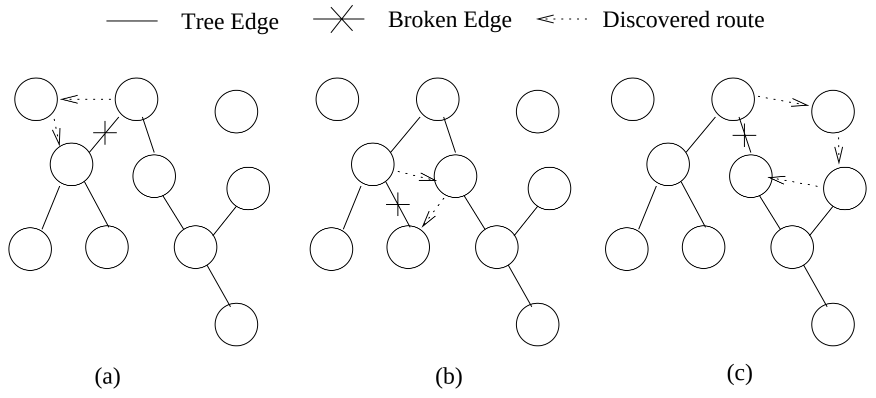


Figure 3.4: Possible local recovery scenarios

Periodical multicast tree refresh

Local tree recovery could provide non-optimal routes or may fail if either the lost downstream node is farther away than the *Greatest-Range* of the upstream node, or if the recovery message is lost. MRDC periodically reconfigures multicast trees to remove these kinds of problems and also give a chance to better adapt the tree to the current topology. The core is in charge of initiating a periodical multicast tree refresh. Every period (called *PERIOD_REF*), the core changes the state of the entry $MRTable(GID)$ to *non-forwarder,* computes a new reference number and broadcasts a CA message. This message will refresh the multicast tree and at the same time, the corresponding reverse path to the core. Once a node receives a non-duplicated CA message, it updates the corresponding fields of the multicast routing entry $MRTable(GID)$, empties the $< DIDs >$ field and sets the state of the entry to *non-forwarder*. In this way, the multicast tree is destroyed and group members run a RAR/RAA procedure to construct another multicast tree.

Group member departures

This periodical multicast tree refresh procedure gives multicast group members the possibility to quietly leave the group. These nodes can first check whether the next multicast tree refresh time will come soon. If so, they do not run the RAR/RAA procedure upon receiving the CA message so that the branch(es) will be pruned automatically. Otherwise, they might choose to explicitly leave the tree by sending a message to their upstream node. In the case when the core ends the transmission and wants to leave the group, it checks whether there is another source in the multicast tree that could become the new core. If it is the unique source in the group, it dismisses the tree. Otherwise the new core will be in charge of sending periodical CAs.

Core competition

Because of network partition, or any other reason, a source may choose to become a core without hearing any CA message from the core. Hence, there may be more than one core node existing in the network. After the network converges, a core can hear the CA messages of other core(s). These cores use a core competition algorithm to decide which source is the winner and it should continue acting as core. Cores can use their IP addresses, identification or any other information to compete. The losers will stop sending periodical CA messages and will not react to group member join packets such that their trees will be quietly dismissed after the multicast route entries have expired.

3.3.4 MRDC control overhead discussion

The control overhead of MRDC comes from the periodical tree refresh and local tree repair procedures. In periodical tree refresh, CA messages are broadcast by flooding. Each node sends at least once the CA message. Then, every group member except the core sends a RAR message and should receive a corresponding RAA message. Thus, if a network consists of n nodes and a multicast group contains m members, the control overhead of the periodical tree refresh per seconds is

$$\frac{n + 2 * (m - 1 + x)}{PERIOD\_REF}, \tag{3.1}$$

where x is the number of non group member nodes on the tree. The number of non group member tree node, x, is determined by the distribution of group members in the network. In the ideal case, where all the group members are within the coverage range of core, x reaches it minimum value, zero. On the opposite, in the worst case where group members are distributed at the bound of the network and multicast tree contains all nodes in the network, $x = n - m + 1$.

Consequently, the total control message rate of MRDC per second is:

$$\frac{n + 2 * (m - 1 + x) + y}{PERIOD\_REF}, \tag{3.2}$$

where y is the amount of control messages involved in local tree repairs during a period. The parameter y is function of both node distribution and topology change speed. In a stable network where there is no topology change, y is zero. While in an extremely dynamic network, y could reach a significant value.

To summaries, the control overhead of MRDC depends on the network size, the group size, group member distribution and the topology change frequency. It is not smaller than $\frac{n+2*(m-1)}{PERIOD\_REF}$ for a given tree refresh period *PERIOD_REF*.

3.4 IEEE 802.11 Background and Multicast in IEEE 802.11

Before introducing the forwarding plan of MRDC in detail, we would like to discuss multicasting in IEEE 802.11 and then justify that an adaptive forwarding mechanism is indeed necessary

IEEE 802.11 Distributed Coordination Function (DCF) is often used as MAC sub-layer protocol for mobile ad hoc networks [43], [44], [45], since DCF is the mode which allows terminals to share the wireless channel in an ad hoc configuration. Unfortunately, until now, IEEE 802.11 DCF is almost identical to the basic Carrier Sense Medium Access/Collision Avoidance (CSMA/CA) when it comes to send multicast packets. The multicast sender simply listens to the channel and then transmits its data frame when the channel becomes free for a period of time. The sender does not receive any transmission acknowledgment message from the receivers.

Multicast transmission not only suffer from wireless interface problems but also from the well known hidden terminal problem in the CSMA/CA protocol [46], [47]. Hidden terminals appear when the network simultaneously transmits different multicast packets. For example, Figure 3.5 shows a simple IEEE 802.11 ad hoc network. In this network, source S wants to send multicast packet to three receivers R1, R2 and R3 through five delivery structure (either mesh or tree) members (nodes A, B, C, D and E). If the source generates packets at a high rate, inter-packets collision might happen. Supposing that multicast source S generates packet p_n when node B or C is sending packet p_{n-1}, S sends packet p_n and node A cannot get this packet. The same problem exists when multiple multicast/broadcast traffic is present in the network at the same time. This kind of hidden terminal problem is generally related to traffic load in the network. The more multicast/broadcast traffic flows in the network, the more hidden terminals will be generated. Yet, there exists another kind of hidden terminal in multicast communications which has no relationship with traffic load. Hidden terminal problem may occur even when delivering one multicast packet. In the same network, let us suppose that source S has only one multicast packet to send. Among delivery structure members, node C and node B cannot hear each other while node E is unfortunately placed in the coverage range of both node B and C. When receiving the multicast packet from node A, node B and C might begin to transmit the same packet simultaneously or before the other node finishes transmission because they do not have any information from the other side. In other words, nodes B and C becomes hidden terminal for each other when they transmit the multicast packet to node E. The packet coming from node B and C collide at the wireless interface of node E prevents this node from receiving this packet from either node B or node C. As a result, the multicast transmission fails at node E. We name this kind of packet collision as ”identical packet collision”. The frequency of *inter-packet collision* is a function of network load while *identical packet collision* rate is independent of network load and is determined by the delivery structure. That also can be used to explain why all multicast routing protocols analyzed in [19] except AMROUTE [26] multicast protocols do not achieve 100 percent packet delivery ratio in stable and low load networks.

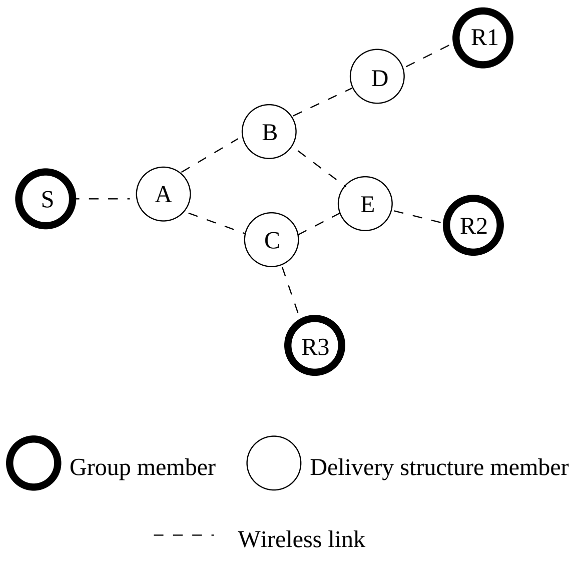


Figure 3.5: Multicast packet delivery in a simple IEEE 802.11 ad hoc network

To reduce multicast packet delivery failure, one potential solution is inspired from the mechanism used in unicast packet transmission. For unicast packets, the specific access scheme of IEEE 802.11 DCF is CSMA/CA with acknowledgments. To mitigate collisions caused by hidden terminals, the nodes can send a unicast packets with RTS/CTS option that is based on two control frames: Request-To-Send and Clear-To-Send for virtual carrier sensing. RTS/CTS option uses a four-way RTS/CTS/DATA/ACK exchange. Before sending a unicast packet, a node first sends an RTS (Request To Send) packet to the destination. If the destination believes that the medium is idle, it responds with a CTS (Clear To Send). The sender then transmits the data packet, and waits for an ACK (Acknowledgment) from the receiver. If a node overhears an RTS or CTS, it knows the medium will be busy for some time, and avoids initiating new transmissions or sending any CTS packets. Thus for multicast communication using current IEEE802.11 MAC protocol, the simplest way is to treat a multicast packet as multiple unicast packets and send them individually to each direct delivery structure neighbors, which we call unicasting multicast packets or **unicast fashion**. On the other side, we define the way of using CSMA/CA to forward multicast packets as broadcasting multicast packets or **broadcast fashion**.

Unicasting multicast packets introduces extra forwarding overhead and transmission delay. In a multicast tree, the forwarding overhead of the ”broadcast fashion” is the number of interior nodes on the tree and that of ”unicast fashion” is the number of edges on the tree. Depending on the multicast delivery structure, the difference between these two fashions is the number of leaf nodes on the tree which varies from 0 (chain) to n-1 (star). Therefore, the unicast fashion is preferable when the network load is not high and at the same time this fashion does not generate congestion. The transmission delay is defined as the difference between the moment that a packet is generated and the moment an application receives it. In broadcast fashion, all delivery structure neighbors simultaneously receive the packet. Thus, the transmission delay of each destination depends mostly on the distance to the source. In unicast fashion, transmission delay is not only a function of the distance but also of the number of branches on the path from source to destination, since tree members send multicast packets to theirs direct tree neighbors one after another. Compared to the broadcast fashion the delay could be increased by $(n-2) * T$ if the delay of one hop transmission is T, and without considering extra delay due to RTS/CTS/ACK and retransmission.

Therefore, broadcast fashion is suitable for applications which are sensitive to transmission delays and jitter but can tolerate packet loss such as voice. Conversely, for applications which are not sensitive to delays (for example, newsgroup), unicast fashion can be used in low load network to achieve a better delivery success rate. Moreover, when the network load increases or congestion appears, a node can switch to broadcast fashion to alleviate congestion so as to provide a best effort delivery. Bearing this idea in mind, we developed the forwarding plan of MRDC.

3.5 MRDC Forwarding Plan Description

In this section, we introduce in detail the forwarding plan of MRDC. The forwarding plan triggers the functionalities of control plan and delivers multicast packets to their destinations. In this section, we assume that all the nodes communicate on IEEE802.11 in ad-hoc mode. If the MAC layer protocol used differs from IEEE 802.11, the forwarding plan supposes the MAC layer support efficient multicast transmissions and will simply pass multicast packets to the MAC layer. The MAC layer accepts all multicast packets and forwards them to the network layer. Then, network layer decides whether to drop the packets or to forward the packet.

In the adaptive multicast forwarding mechanism, we defined two transmission modes: unicast mode and broadcast mode. In the unicast mode, forwarding mechanism treats a multicast packet as multiple unicast packets and sends a copy to every delivery structure neighbor with the RTS/CTS option. In the broadcast mode, multicast packets are sent directly using CSMA/CA protocol. A structure neighbor list, which contains a node list and a broadcast flag, is defined in order to permit the forwarding plan to obtain essential routing information from the control plan.

This mechanism contains two procedures: mode selection and data forwarding. The mode selection procedure is executed periodically. In this procedure, nodes compute two metrics and select a suitable transmission mode. The data forwarding procedure transmits multicast datagrams in the selected mode. The following sections present these two procedures in detail.

3.5.1 Mode Selection

If the underlying MAC layer protocol is IEEE 802.11, nodes initially operate in unicast mode and periodically (every MS_PERIOD seconds) calculate two metrics: Average Queue Length (AQL) and Medium Occupation for Reception (MOR) and compare them respectively to their thresholds: Queue LENgth threshold (QLEN) and INcoming occupation ThReshold (INTR). The comparison result defines which transmission mode will be used in the next period. Figure 3.6 shows the mode selection procedure at time t. The mode selection procedure contains two steps: metrics computation and mode selection. We will present this procedure in detail.

```
Algorithm Mode_selection(t)
Notation: AQL: Average Queue Length
          MOR: medium occupation for reception
          QL: Queue Length
          #B(t): Number of bytes received till moment t
          C(t): Transmission speed at moment t
          QLEN: Queue LENgth threshold
          INTR: INcoming occupation ThReshold
Begin
  Get QL(t), #B(t) and C(t);
  calculate AQL(t);
  calculate MOR(t);
  if mode==unicast then
    if AQL>QLEN or MOR>INTR then
      mode=broadcast;
  else
    if AQL<QLEN/2 and MOR<INTR then
      mode=unicast;
  store AQL(t) and #B(t) for the calculation of next period
End
```

Figure 3.6: Mode_selection() at time t

When the current IEEE 802.11 specification is applied to transmit multicast packets, the hidden terminal problem creates severe transmission failures. In order to reduce them, one potential solution is to deliver multicast packets point-to-point to selected neighbors with RTS/CTS option. However, the obvious disadvantage of unicasting multicast packet is that this method generates more forwarding traffic than CSMA/CA. The extra forwarding traffic becomes annoying as network load increases. Therefore, unicasting multicast packets is feasible only in low load networks. Hence, we chose the Average Queue Length (AQL) as the key metric to choose transmission mode. AQL is the mean number of packets in queue that are awaiting transmission by the network interface. Because queue lengths change greatly as transmission mode and/or traffic pattern changes, the following formula is used to calculate current average queue length AQL(t):

$$AQL(t) = \alpha * AQL(t-1) + (1-\alpha) * QL(t) \tag{3.3}$$

where $QL(t)$ is the queue length at time t and AQL(t-1) is the average queue length of the last period. This metric reflects the node's difficulty to send packets into the network. At the beginning where there is no traffic in the network and consequently no packet waits in the queue, the initial average queue length $AQL(0) = 0$. As traffic increases in the network, nodes have more packet to send or forward. Then their average queue length increase. When AQL exceeds a certain threshold, called Queue LENgth threshold (QLEN), the broadcast mode should be employed in order to reduce bandwidth consumption. After a node switches from unicast mode to broadcast mode, the length of the queue will significantly get reduced because it sends less copies and broadcast packets are sent faster (no four handshakes at the MAC layer for example). Therefore to avoid ping-pong switch, nodes should wait until their AQL become smaller than another threshold (e.g. half of QLEN) before they can switch back to unicast mode.

Furthermore, some implementations of 802.11 provide some statistics such as the number of bytes received or sent during the last period, etc. These counters can be utilized together with AQL to make a smarter choice in avoiding congestion. Here we define a metric called Medium Occupation for Reception (MOR) to reflect network load. MOR is defined as the MAC busy for receiving over a period. With this metric, a node estimates the bandwidth occupied by its neighbors and consequently the bandwidth it can use. Nodes use the statistics of received bytes provided by the underlying IEEE802.11 layer to calculate MOR as follows: The MAC layer counts the number of bytes it receives until time t: $\#B(t)$. Then the nodes are able to compute MOR using the formula:

$$MOR(t) = (\#B(t) - \#B(t-1))/(C * MS\_PERIOD) \tag{3.4}$$

where C represents the MAC layer transmission rate during the last period or at time t. The accuracy of this metric depends on how many neighbor nodes use unicast mode to transmit multicast packets since the estimation does not take into account RTS, CTS, ACK and retransmission. Thus, the more neighbors operate in broadcast mode, the more accurate MOR a node can obtain. The forwarding mechanism can estimate the available bandwidth for the node through the formula:

$$(1 - MOR(t)) * TOTAL\_BANDWIDTH \tag{3.5}$$

When MOR is bigger than a threshold, called INcoming occupation ThReshold (INTR), a node considers that it is in hot spot and probably has not enough bandwidth to unicast multicast packets and should switch to the broadcast mode. Otherwise, the node is not in hot spot and can use unicast mode if it does not create congestion. In case the MAC layer counters are not available, the algorithm always has $\#B(t) = \#B(t-1) = ... = 0$, which leads to $MOR = 0$ and consequently dis-activates this metric automatically.

In brief, if any one of AQL and MOR is superior to their respective thresholds (QLEN for AQL and INTR for MOR), the node is considered either in hot spot or having sent too many packets. It will choose broadcast mode as the multicast transmission mode to be used in the next MS_PERIOD. On the other hand, if AQL is smaller than half of QLEN and MOR is inferior to INTR, the node considers that the traffic it sent is relatively low and the medium occupation rate permits to send more traffic. It can therefore use unicast mode to transmit multicast packets in the next period. Multicast routers will forward multicast data packets according to the current transmission mode in use.

3.5.2 Multicast Data Forwarding

Routers deliver multicast packets according to the transmission mode in use and the state of correspond multicast routing entry.

When the state of the multicast routing entry is set to *on-tree*, sources begin to send their multicast packets. Intermediate routers should detect duplications before relaying a multicast packet. When a node receives a multicast packet, it consults its Duplication Table to see if the packet has been processed before. If so, it discards the packet. Otherwise, it updates the Duplication table to reflect the packet header information (source address, port number and reference number). After ensuring that the packet is non-duplicate, the forwarding plan asks the control plan to fill the structure neighbor list of group G. It decides to forward the packet, drop the packet or start any action depending on the content of this list.

If the multicast routing entry does not exist, the control plan returns an error. The forwarding plan checks whether the packet's source is the node itself. If it is the case, which means a new session begins and the node is the first source, the node should act as a core. The forwarding plan triggers a multicast tree construction. Otherwise, the forwarding plan simply drops the packet.

From now on, let us consider the case where the multicast routing entry exists. As mentioned in Section 3.3.1, a multicast routing entry has three states: *on-tree*, *tree-fault* and *non-forwarder*. While non-forwarder state indicates that a node is not a delivery structure member and as a result returns an empty list and does not set the broadcast flag, the other two states are reserved for multicast tree members. In these cases, when receiving a packet for some group G, the control plan fills in the structure neighbor list of group G with the list of all direct tree neighbors stored in the multicast routing entry and sets an indication of broadcast requirement if the entry's state is tree-fault. The control plan takes out of the list the node from which the packet has been received before passing the result to the forwarding plan. If the structure neighbor list does not contain any node and the broadcast flag is not set, the forwarding plan drops the multicast packet. Otherwise, the forwarding plan sends the packet according to the transmission mode.

If a node is in broadcast mode, the forwarding plan passes the packet directly to the MAC layer and this packet will be sent with CSMA/CA mechanism without acknowledgment. On the other hand, if a node is in unicast mode, the forwarding plan sends a copy of the packet to each member in the list. Then, if the broadcast flag is set, forwarding plan broadcasts the packet.

Let us now see how the unicast mode works in the former example (Section 3.4). S explicitly sends a multicast packet (or encapsulates the multicast packet in a unicast packet and then sends it) to node A, then node A duplicates the multicast packet and sends a copy to node B and C. This process continues until the multicast packet reaches R1, R2 and R3. As a result, the multicast transmission failure caused by hidden terminal can be greatly reduced thanks to four-way handshaking.

3.5.3 Forwarding overhead discussion

The forwarding overhead of MRDC at network layer depends on the multicast tree structure and the forwarding mode of the routers contained in the tree. If all nodes operate in broadcast mode, the number of forwarding events is equal to the number of interior nodes on the tree:

$$\#offorwarding = \#\ of\ interior\ nodes \tag{3.6}$$

On the other hand, if routers use unicast mode to transmit multicast packets, the number of forwarding events is the number of edges on the tree, which is equivalent to the number of nodes on the tree (interior nodes plus leaf nodes) minus one.

$$\#offorwarding = \#\ of\ nodes\ on\ tree - 1 \tag{3.7}$$

Therefore, the forwarding difference of these two modes is the number of leaf nodes minus one.

3.5.4 Related Works

To efficiently support multicast transmission, a few multicast MAC protocols [48], [49], [50] have been proposed to extend the IEEE 802.11 broadcast/multicast protocol with RTS/CTS handshaking. In [48], once a sender gains access to the medium, it transmits an RTS packet to its neighbors and waits for CTS packet for WAIT_FOR_CTS time units. If a node receives an RTS packet when it is not in the YIELD phase, it sends back a CTS and then waits for the data packet for WAIT_FOR_DATA time units. If the sender does not receive any CTS packet before its WAIT_FOR_CTS timer expires, it backs off and enters the contention phase again to retransmit the broadcast/multicast data packet. If the sender receives any

CTS packet before its WAIT_FOR_CTS timer expires, it transmits the data packet and waits for WAIT_FOR_NAK time units for any possible transmission problem reported by the neighboring nodes. If a receiver does not receive the data packet after it has transmitted the CTS packet for WAIT_FOR_DATA time units, it transmits NAK packet. If the sender does not receive any NAK packet before its WAIT_FOR_NAK timer expires, the broadcast/multicast service is complete. Otherwise, the sender backs off and enters the contention phase again to retransmit the data packet. Broadcast Medium Window (BMW) [49] mainly considers supporting reliability for broadcast but it can also support multicast. The basic idea of BMW is that a node reliably transmits a broadcast packet to each of its neighbors in a round robin fashion. The neighbor list is obtained by both periodical HELLO messages and overhearing. Once a packet other than HELLO message is transmitted by a node, the node will suppress its next HELLO message, assuming neighbor nodes can know its presence by overhearing this packet. The main drawback of the BMW is that it uses at least m rounds of unicasts for a broadcast/multicast packet addressed to its m neighbors, which does not only introduce at least m rounds of contention phases, but also makes no use of the broadcast nature of the wireless channel. This protocol is very similar to our unicast forwarding mechanism except that it operates at the MAC layer while ours is located in Network layer.

Considering the problem of BMW, Batch Mode Multicast MAC protocol (BMMM) [50] proposes to consolidate the m contention phases into a single one and transmits the data packet only one time before the ACK collecting. To reliably transmit a multicast packet in BMMM, a sender first uses its RTS packets to request intended receivers one by one to reply with a CTS. If the sender receives at least one CTS, it transmits the data packet. After the data packet transmission, it uses a new control packet called RAK (Request for ACK) to request ACKs from the intended receivers one by one. In case of missing ACKs, the sender will do retransmission. All the intervals between the above sequence of packets are set to a value less than DIFS, so once the sender grabs the channel, the reliable multicast operation will not be interrupted by other transmissions. It introduces m rounds of RTS/CTS exchange and RAK/ACK exchange, in which any of the $4m$ packets missing will cause retransmission.

BMMM uses less medium than the unicast transmission mode of our adaptive forwarding mechanism. However, it obtains a certain reliability at the cost of bandwidth consumption and bigger transmission delays. This protocol uses extra bandwidth for channel reservation and transmission acknowledgment compared to the plain CSMA/CA mechanism defined in the IEEE 802.11 specification. Furthermore, if these protocols operate alone, they might provide unnecessary reliability in some cases. For example, in Figure 3.2, node E is a multicast receiver for both node B and C at the MAC layer point of view . Thus nodes B and C could reliably send multicast packets to node E since reliable transmission from one node is enough. However, if BMMM cooperates with MRDC by replacing the unicast transmission mode, we can achieve a much better multicasting performance. Suppose that the MRDC constructs a multicast tree to connect source and receivers in Figure 3.2

and that node E is node B's downstream node on the tree. Node B can explicitly inform the MAC layer that receivers are node E and D. Node C does not need to reliably reach node E but node E is always able to receive multicast packet from node C. In this way, we can reduce bandwidth consumption and obtain redundant transmissions at the same time.

3.6 Conclusion

In this chapter, we introduced a best effort multicast routing protocol, called the Multicast Routing protocol with Dynamic Core (MRDC) for mobile ad hoc networks. The aim of this work is to reach a compromise between forwarding overhead and routing overhead so we can minimize the total bandwidth consumption. Because traffic packets are generally much more numerous and bigger than control messages, our strategy chooses the techniques which are most efficient for data transmissions while usually creates heavier control overheads. We developed some techniques which significantly reducing the control overhead with the cost of loosing some data transmission efficiency to obtain an optimal bandwidth utilization. Finally the routing protocol uses the best effort approach to deliver packets.

According to this idea, MRDC is split into two plans: the control plan and the forwarding plan. The control plan chooses tree structure to achieve the best data transmission efficiency. As nodes move and the network configuration changes, the tree is periodically reconfigured to maintain efficiency. A multicast tree is rooted at the first source of a multicast session. Therefore, since the tree is source-based, MRDC achieves the best data transmission efficiency for in a single source multicast session. In multiple source applications the tree becomes group-shared to reduce control overhead. Another improvement to reduce the control overhead is that the construction and maintenance of any multicast tree are triggered in an on-traffic-demand basis at the cost of introducing transmission delays for the first packets. Faults are tolerated in the trees, which might affect packet delivery on the concerned sub-tree but can avoid heavy control overheads to maintain a strictly correct tree. MRDC only needs a small quantity of control overhead but can provide a delivery structure which generates less forwarding overhead, therefore reducing the total bandwidth consumption.

The forwarding plan is in charge of delivering multicast packets on a best effort basis. Considering the shortcoming of the current 802.11 standards in multicast packet transmission, we introduced an adaptive multicast forwarding mechanism in MRDC's forwarding plan for 802.11 wireless networks. This mechanism makes a transmission mode choice based on two metrics: Average Queue Length (AQL) and Medium Occupation for Reception (MOR). If both metrics are smaller than their respective thresholds, the forwarding mechanism treats a multicast packet as a set of unicast packets and delivers them with the RTS/CTS option of IEEE 802.11. If it is not the case, the forwarding mechanism passes a multicast packet directly to the MAC layer. Through this mechanism, MRDC provides some degree of reliability in one hop multicast delivery. Thus, we can offer a better service for upper layer applications. If they are sensitive to delay or benefit from some other mechanisms to recover delivery failures, MRDC can always use the broadcast mode to achieve short delays and low bandwidth consumption. For the other applications, MRDC is able to offer an optimal packet delivery ratio with respect to the network load. That is what we call **best-effort multicasting**.

We estimated the routing overhead and forwarding overhead of MRDC. This overhead is a function of the multicast tree size which is further decided by the distribution of group members in the network and their distance to the core. In the next chapter, we will study the performance of MRDC in a packet level simulator. After selecting the suitable key metrics for MRDC such as the period of tree refresh (PERIOD_REF) and the thresholds of mode selection procedure, we evaluate the size of MRDC multicast tree and discuss the efficiency and robustness of MRDC under different movement and traffic scenarios. We then compare its performance to other multicast routing protocols.

Chapter 4

Performance Analysis of Multicast Routing Protocol with Dynamic Core (MRDC)

In chapter 3, we introduced our proposition: Multicast Routing Protocol with Dynamic Core (MRDC), and briefly analyzed its performance such as routing overhead and forwarding overhead. In this chapter, we evaluate the performance of MRDC through detailed packet level simulation under a network simulator, ns-2 [9] to have a closer observation on this protocol. This performance analysis contains two goals: to select MRDC key parameters (e.g. period of multicast tree refresh and threshold for average queue length) and to analyze performance in different traffic loads and mobility pattern. The performance analysis is further divided into two parts: multicast tree analysis and protocol comparison.

The rest of this chapter is organized as following. Section 4.1 introduce the simulation environment followed by Section 4.2 in which we present the movement pattern and traffic pattern that will be used in the simulations. Section 4.3 describes the implementation decisions. Section 4.4 chooses optimal parameters of MRDC. With these parameters, we first analyze the correctness and robustness of multicast tree in Section 4.5. And then section 4.6 evaluates the performance of MRDC in comparison with some other multicast routing protocols. Finally, Section 4.7 concludes this chapter.

4.1 Simulation Environment

We conduct our simulations using the *ns-2* network simulator [9], with MONARCH project wireless and mobile extension ([43] and [51]). *Ns-2* is a publicly available discrete, event-driven simulator developed by the University of California at Berkeley and the VINT project. MONARCH project at Carnegie Mellon University extended ns-2 to provide support for simulating multi-hop wireless networks completed with signal strength, radio propagation, data link layer and IEEE802.11

MAC protocol[41]. A comparison of ns-2 with other popular simulators such as OPNET [52] and GloMoSim (QualNet) [53] can be found in [54] and [55].

Our simulation models a network of 50 mobile nodes placed randomly within a 1000mx1000m flat space. The physical radio characteristics of each mobile node's network interface, such as the antenna gain, transmit power and receiver sensitivity, are chosen to approximate the Lucent/Agere WaveLAN [56] direct sequence spread spectrum radio characteristics. The nominal bit-rate is 2 Mb/s and the nominal radio range is 250 meters, depending on capture effect and colliding packets. The link layer model is the Distributed Coordination Function (DCF) of the IEEE 802.11 wireless LAN standard. We have extended the existing simulation modules to enable multicast simulations with MRDC. Each node has a priority queue, called interface queue, for packets awaiting transmission of the network interface. This queue gives priority to routing messages. It holds up to 64 packets and is managed in a drop-tail fashion.

4.2 Simulation Scenarios

A number of movement scenarios and traffic scenarios are generated and used as inputs to the simulations. Each movement scenario file determines movements of 50 nodes. The movement model of nodes is the random waypoint model [43] without pause. Each node begins the simulation by selecting a random destination in the 1000mx1000m space and moves to that destination at a speed distributed uniformly between 0 and a maximum movement speed. Upon reaching the destination, the node selects another destination, and moves there as previously described. Nodes repeat this behavior for the duration of the simulation. Each simulation runs for 900 seconds of simulation time. Movement patterns are generated for different maximum speed. When maximum speed equals to 0, nodes do not move during a simulation which represents stable networks. A low maximum speed results to a low relative movement speed of nodes and corresponds to low mobility cases. On the contrary, a high maximum speed means high relative movement speed among nodes and corresponds high mobility. Because the performance of the protocols is very sensitive to node position and movement pattern, we generated 10 movement scenarios for each value of maximum speed. Thus, each collected data in figures and tables presents an average of these 10 movement scenarios with the same maximum speed. Network partition is tolerant in mobility scenarios while excluded in stable networks.

Traffic scenarios determine the number of groups, group members and multicast traffic. A number of nodes are chosen as multicast group member. To reduce side effects, membership control features are turned off. All group members join the multicast session at the beginning of the simulation and remain as members throughout the simulation. Multicast traffic is generated by constant bit rate (CBR) sources. Each source sends 4 packets per second. The size of data payload is 512 bytes. The transmissions start at times uniformly distributed between 30 and 60 simulation seconds and continue till the end. These sources are attached to nodes which were chosen among multicast members. The number of groups is mode two of the number of sources. For example a 5-source traffic scenario defines 3 multicast groups among which 2 groups have respectively 2 sources and the third one has one source. This configuration forms not only inter-group competition but also intra-group inter-sources competition.

4.3 Implementation Decisions

While implementing the MRDC in *ns-2*, we made following decisions. The Greatest-Range of JI message propagation is 4 hops. Upstreams wait for 0.5 seconds before broadcasting another JI message. Downstreams set the multicast routing entry to *tree-fault* state 1.5 seconds after detecting edge broken.

NEIGHBOR_HELLO period is set to 0.5 second and the timer of active neighboring entry is set to 1 second in the simulations. In order to improve bandwidth efficiency, MAC layer cooperation is used in updating active neighbor table. When a node successfully sends or receives a packet to/from a neighbor, it updates the corresponding entry in active neighbor table because MAC layer control message (RTS, CTS and ACK) is received from the neighbor.

4.4 Parameter Selection

The simulations in this step address to achieve a suitable period value for multicast tree refresh and optimal thresholds for transmission mode selection. These parameters will be used in the simulations of the performance analysis.

4.4.1 Period of multicast tree refresh

The period of multicast tree refresh is an important parameter of MRDC, which has direct impact on the performance of protocol. The longer the period is, the more slowly MRDC reacts to topology changes and the more fault might exist in multicast tree. That reduces the number of packets delivered to receivers. On the other hand, a shorter period means frequent network range broadcast which increases significantly routing overhead. Therefore, an ideal refresh period (PERIOD_REF) should permit this protocol to deliver as many as possible multicast packets without creating significant routing overhead. For this reason, the following two metrics are employed to select period of tree refresh.

- **Packet delivery ratio**: the ratio of the number of multicast data packets correctly delivered to the receivers versus the number of multicast data packets supposed to be received. The packets, which are sent when some receivers are unreachable for the sources because of network partition, are counted as supposed to be received by those receivers.

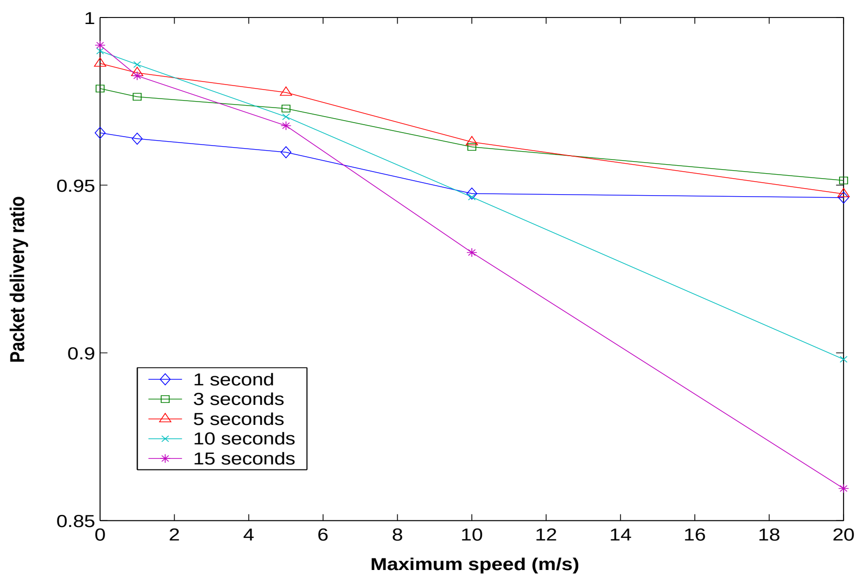
- **Number of control messages per second**: The rate of MRDC control messages transmitted for multicast tree construction and maintenance. This metric is used to investigate the resource consumed by multicast routing protocol.

Because periodical tree refresh mainly addresses topology changes, we use different movement scenarios without changing traffic scenario in this step. The maximum movement speed is varied from 0m/s (stable networks) to 20m/s (high mobility networks). A traffic scenario in which one multicast group contains 10 members and two traffic senders is chosen to simulate a group-shared case. One sender plays the role of core and the other one acts as normal group member. Mode selection is disable in the simulations. All routers broadcast multicast packets.

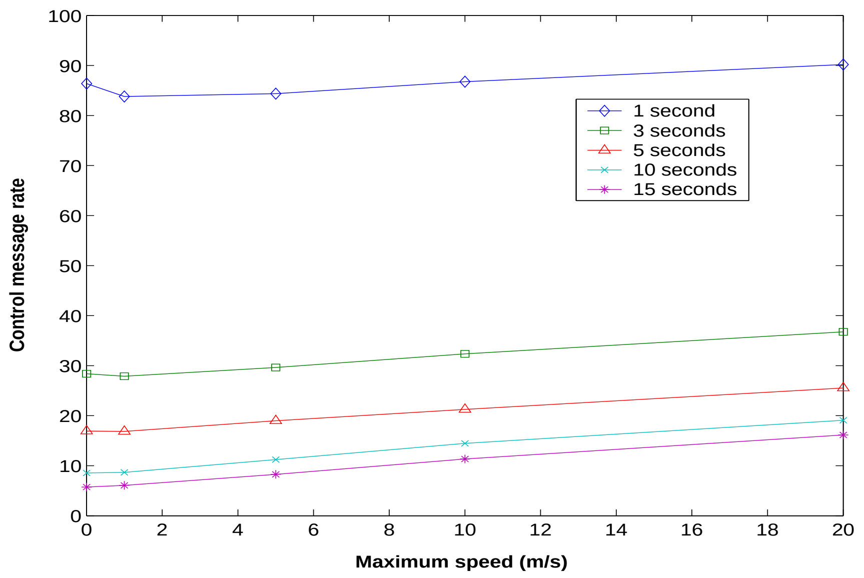
The simulation results are shown in Figure 4.1. Packet delivery ratio decreases with the increase of mobility speed but in shorter periods it resists better than in longer ones, as illustrated by Figure 4.1(a). Tree structure offers the unique route to distribute data packet from sources to receivers. Once topology changes touch multicast tree, packets transferred on the broken branch(es) will be dropped. High relative movement speed causes high degree of topology changes that in turn gives high tree break rate. A shorter tree refresh period produces more frequently reconfiguration and consequently can react more quickly to topology changes. That is why short PERIOD_REF is robust against topology. We will more deeply study the impact of mobility on a multicast tree in performance analysis section.

In terms of achieving a better packet delivery ratio, Figure 4.1(a) shows a contradiction that low mobility networks favorite long period while short period is preferred in high mobility networks. After analyzing the reasons of packet delivery failure, we find the answer of this contradiction. Besides low layer transmission failure and routing protocol, packet delivery failure is also caused by the collaboration of control plan with multicast forwarding mechanism. The bad collaboration of two parts is the main reason which makes the difference of packet delivery ratio in stable and low mobility networks. Recall that, in order to remove errors and form a tree more adapt to current topology, MRDC destroys old tree and constructs a new one. This results in that multicast packets cannot be correctly delivered to all receivers during that period. More frequent tree refresh causes more delivery failure relative to this fact. Believing that a smart forwarding mechanism can greatly reduce this type of delivery failure, short PERIOD_REF is preferred in all mobility cases.

As shown in Figure 4.1(b), bigger PERIOD_REF values generate smaller number of routing messages to construct and maintain multicast tree, while their control overhead increases more quickly than that of smaller ones with the increase of mobility. High degree of topology changes makes MRDC generate more control messages for local tree recovery. Frequent tree reconfiguration alleviates this requirement. Thus node mobility has less effect on control overhead of short PERIOD_REF than long ones. However, in all the cases, shorter PERIOD_REFs generate more overhead.



(a) Packet delivery ratio as a function of Maximum speed and PERIOD_REF



(b) Routing overhead as a function of Maximum speed and PERIOD_REF

Figure 4.1: MRDC's performance under different period of multicast tree refresh

Short PERIOD_REF makes protocol robust against topology changes. While, long PERIOD_REF makes protocol efficient with low control overhead. In the rest of simulations, we use 5 seconds as PERIOD_REF since in this case MRDC can deliver more than 94% data packet and create less than 5% routing overhead.

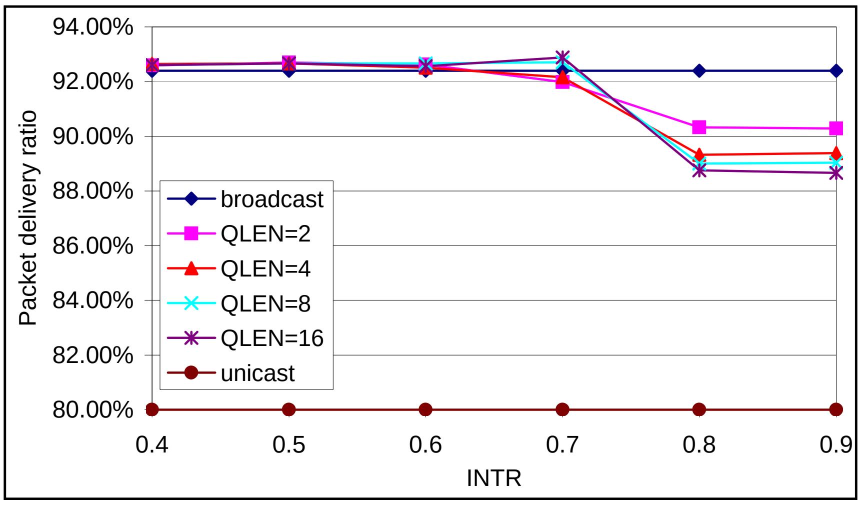
Thresholds of Mode Selection Procedure

In this section, we study the impact of QLEN and INTR on the performance of adaptive multicast forwarding mechanism to obtain an optimal pair. We set node's maximum movement speed to 5 m/s and choose 6-source traffic scenario, because this is the traffic scenario in which broadcast mode begins to outperform unicast mode (see Figure 4.4 in Section 4.6). This scenario defines three multicast groups and each group has 10 members and two CBR sources. We vary the QLEN from 2 to 16 and INTR from 0.5 to 1.0. MOR is always inferior to 1.0 because it does not consider medium occupied by a node itself for sending packets. Thus, by setting INTR to 0.9, which makes the metric MOR always smaller than its threshold, we simulate the case where MAC layer counters are unavailable. For comparison reason, we also test the performance of MRDC in the cases in which all nodes operate in broadcast mode (set INTR=0 for example) or in unicast mode (QLEN=65 and INTR=1.0). The former case is denoted as *broadcast* and later as *unicast*. Following two metrics are employed in the simulation of mode selection threshold:

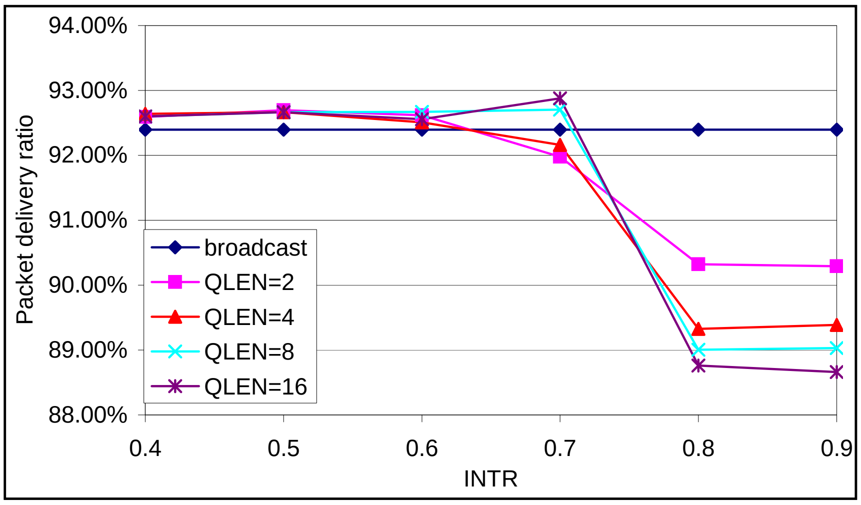
- **Packet delivery ratio**: Same as that in the Section 4.4.1
- **Average end-to-end delay**: the average time between that a packet is initiated by a source and that it is received by multicast receiver. This metric is important for some applications sensible to delay. This metric can also demonstrate how much transmission delay is introduced by unicast mode.

Figure 4.2 (a) shows that the packet delivery ratio of adaptive multicast forwarding mechanism as a function of INTR and QLEN. In order to show better the details of the performance curves, we enlarge the y-axis scale range from 88% to 94% and show the result in Figure 4.2 (b). The simulation results show that the adaptive multicast forwarding mechanism provides the best packet delivery ratio when INTR equals to 0.7 and QLEN is 16. If the MAC layer counter is not available, the QLEN should be set to 2. Small INTR and/or QLEN makes nodes easily switch from unicast mode to broadcast mode and stay in broadcast mode. Thus the corresponding results are similar to those of broadcast case, in which all nodes operate in broadcast mode. However, the nodes which do not locate in hot spot continue to use unicast that improves the packet delivery ratio compared to broadcast to reduce multicast transmission failure. As INTR and QLEN increase, more and more nodes tend to operate on unicast transmission mode, which gives a performance comparable to that of unicast case.

In general, when QLEN rests unchanged, packet delivery ratio increases first and then decreases as INTR increases. While, for a given INTR, the smaller the

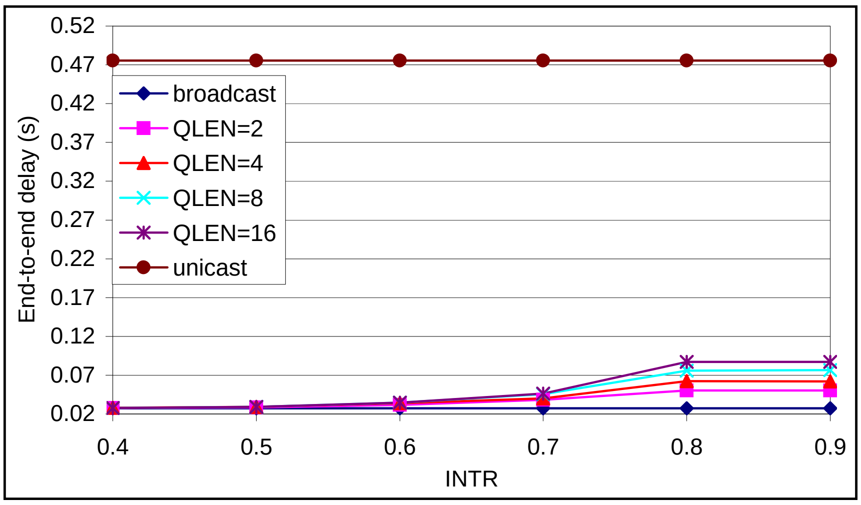


(a) Original (including unicast case)

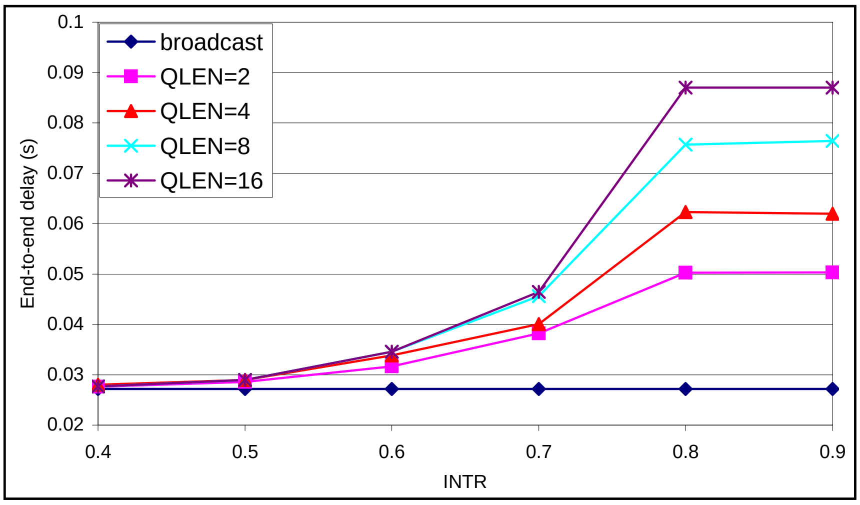


(b) Enlarged (without unicast case)

Figure 4.2: Packet delivery ratio v.s. QLEN and INTR



(a) Original (including unicast case)



(b) Enlarged (without unicast case)

Figure 4.3: End to end delay v.s. QLEN and INTR

QLEN is, the better the packet delivery ratio we get. But this rule is broken when INTR equals to 0.7, smaller QLEN gives worse packet delivery ratio than bigger one.This phenomenon is relative to node's position and their transmission mode. When INTR and QLEN are small, only the nodes which locate at the border of network operate on unicast mode. The increase of INTR and/or QLEN favorites nodes locate in central region or hot spot to stay on unicast mode instead of switching to broadcast mode. This tendency has two effects. On one hand, packet delivery ratio is improved by reducing transmission failure caused by hidden terminal problem. On the other hand, this change degrades the performance because unicast mode prevents delivery structure member getting packets from other links than those defined by the tree. When the members of delivery structure locate in the region far from the center, there is small possibility to have redundant transmission. Thus unicast mode is preferable. On the contrary, for the center nodes which have larger possibility to possess redundant transmission and normally at same time have more traffic to forward, broadcast mode is appreciated. For this reason, packet delivery ratio increases first and then decreases.

Figure 4.3 (a) illustrates end-to-end delay evolution as a function of INTR and QLEN. Figure 4.3 (b) enlarges a part of graph (a) by setting the y-axis scale range from 0 to 0.1 to show better the evolution. The results prove that unicast mode creates more delay than broadcast mode but has different consequence depending on which nodes practice this mode. When only border nodes employ unicast mode, the number of nodes in structure neighbor entry keeps small and the extra delay is tolerable. On the contrary, center nodes usually have a big structure neighbor list. Sending a multicast packet one by one to these neighbors creates important transmission delay. Thus, in terms of small transmission delay, INTR should be small. However, a small QLEN is appreciated in both packet delivery and delay.

INTR plays a more important role on the performance (Figure 4.2 and 4.3) when it is inferior to 0.7, and after that QLEN makes effects on both packet delivery and delay. A small INTR makes node easily think that it is in hot spot before packet accumulates in queue and as a result the influence of QLEN is reduced.

Although ($INTR = 0.7, QLEN = 16$) pair gives the best packet delivery ratio, it generates much greater delivery delay. To get a compromise between delivery ratio and delivery delay, we use ($INTR = 0.6, QLEN = 8$) pair as the thresholds of the mode selection in the performance comparison simulations.

4.5 Multicast Tree Analysis

We evaluated the performance of MRDC multicast tree in a variety of mobility and communication scenarios. Because we focused on control plan of MRDC, the performance in this chapter thus means the efficiency and robustness of multicast tree. Mode selection of forwarding plan is disabled and multicast routers broadcast multicast packets. Although the aim of this simulation is to evaluate the robustness and efficiency of multicast tree, we still introduce multicast traffic to test the performance in different network load.

4.5.1 Simulation Metrics

Performance analysis aims to demonstrate the robustness and efficiency of MRDC multicast tree. The robustness is to test whether multicast tree keeps connecting and covers all reachable group members when network topology changes or control message loss. On the other hand, the efficiency means whether the potential forwarding overhead and routing overhead of MRDC multicast tree scale well with different mobility and traffic scenarios. The following metrics are chosen:

- **Average number of multicast router**: This metric counts the average number of nodes on the multicast tree which transmit multicast packet during a simulation. It allows us to estimate the forwarding overhead in terms of the number of packet forwarded to deliver a multicast packet to receivers in broadcast mode and under an ideal condition (for example without transmission loss). Thus this metric provides the scalability and efficiency of multicast routing protocol.

- **Average number of non-member router**: It measures the means of the number of nodes which are on the multicast tree but at the same time not the group member. This metric can be used to compute routing overhead in periodical tree refresh but also the size of multicast tree. This metric gives the value of parameter x in Formula 3.1. Applying other two predefined parameters, number of mobile nodes and number of group members, we can calculate routing overhead of periodical tree refresh in a simulation. The number of non-member router plus the number of group members gives the total number of nodes on the tree if we do not consider network partition. That is the forwarding overhead in unicast transmission mode.

- **Number of tree repair times**: This metric counts the number of local tree repair times initiated by MRDC. MRDC's routing overhead comes from periodical tree refresh and local tree recovery. For a given simulation time, the routing overhead generated for periodical tree refresh can be calculated by the Formula 3.1. While the routing overhead of local tree recovery varies scenario to scenario. Therefore, this metric allows us to estimates the variation of routing overhead of MRDC in different scenarios.

- **Tree broken times**: It measures the number of multicast tree broken detected by simulator during a simulation. Supposing that all multicast routers operate on broadcast transmission mode to deliver multicast packets, simulator checks whether all group members within the same network (partition) as the core are reachable through the multicast tree. In other words, tree broken here means physical fragmentation of multicast tree, since simulator does not verify logical relationship among tree members. This metric reflects the robustness of a tree-based multicast routing protocol.

To calculate these metrics, after multicast sources begins their transmission, simulator reports every half second the number of total tree nodes and interior tree nodes and whether the tree covers all reachable group members. Reachable group members are the group members which are within the same network (partition) as the core or in other words core can reach these members directly or through some other nodes. There are totally 1696 such reports during a simulation. Interior tree nodes are tree members that have downstream nodes. The number of total tree nodes minus reachable group members gives the number of non-member routers. A multicast tree is called covers all reachable group members if core can reach all other group members within the same network partition through the tree. In the other case, the multicast tree is called broken.

4.5.2 Performance analysis

Mobility Speed

In this experiment, the maximum movement speed is varied from 0 m/s to 20 m/s to examine the robustness of the protocol against topology changes. One multicast group containing 20 members is simulated. The network load is set to very light (1 source) to exclude as much as possible the influence of traffic packets on control message transmission.

Maximum speed (m/s)	Average non-member routers	Average interior node	# of tree repair times	# of tree broken times
0	8.24	14.81	0	8
1	7.23	12.61	51	9
2	7.05	12.51	91	19
5	7.10	12.72	205	44
10	7.10	13.06	344	74
15	6.48	12.44	466	94
20	6.64	12.84	596	122

Table 4.1: Performance of MRDC multicast tree as a function of Maximum mobility speed

Table 4.1 illustrates the performance of MRDC multicast tree as a function of maximum movement speed. It shows that MRDC multicast tree is scalable in terms of forwarding overhead and remaining reasonable correct as topology change increases. The forwarding overhead in broadcast mode might remain stable since the number of interior node is nearly unchanged in dynamic networks. For the forwarding overhead of unicast mode, node mobility even decreases slightly the size of multicast tree. That can be obtained by adding the number of group members to the number of non-member routers. The result decreases from 28.23 (=20+8.23) to 26.64 (=20+6.64). One reason is that movement makes node uniformly distributed in network, and as a result, the distances, in terms of number of hops, from group members to core are reduced as shown in Table 4.2. In dynamic networks, we do not exclude network partitioning. Network partitioning makes some group member temporarily unreachable (see Table 4.2) which consequently also reduces the requirement of multicast router.

Maximum speed (m/s)	Distance (# of hops) from members to core	Times of members unreachable to core
0	63.3	0
1	49.6	0
2	49.1	0.0428
5	47.8	0.1694
10	47.0	0.1623
15	45.5	0.2077
20	45.8	0.1201

Table 4.2: Multicast group in mobility simulations: distance and unreachable time

Table 4.2 also demonstrates the advantage of multicast comparing with unicast and broadcast in delivering a packet to multiple receivers. The distance in terms of the number of hops from core to a multicast member is exactly the forwarding overhead of sending a packet to that member. The second column of Table 4.2 gives the forwarding overhead of unicast. This column divided by the third column of Table 4.1 gives the gain of multicast method. That of broadcast method is 50, the number of nodes in the network. Table 4.3 compares the forwarding overhead of unicast, multicast and broadcast methods and shows that multicast can at least reduce 3 times the forwarding overhead.

Maximum speed (m/s)	Forwarding overhead (unicast/multicast)	Forwarding overhead (broadcast/multicast)
0	4.3	3.4
1	3.9	4.0
2	3.9	4.0
5	3.8	3.9
10	3.6	3.8
15	3.7	4.0
20	3.6	3.9

Table 4.3: Multicast group in mobility simulations: distance and unreachable time

Both the number of local tree repair and the number of tree broken detected by simulator increase with the node mobility, which increases the control overhead for local tree recovery, but with the different speeds. The number of tree repair increases more quickly than tree broken times does. In fact, the number of tree repair times is decided by the frequency of link changes during simulations as shown in Table 4.4. The number of link changes is about 14-16 times of tree repair times. Therefore, the cost of maintaining a multicast tree in dynamic networks increases as the number of link changes in the network.

Maximum speed (m/s)	Tree repair times	Link changes	Link changes to tree repair ratio
0	0	0	-
1	51	711	13.94
2	91	1285	14.12
5	205	2810	13.71
10	344	4999	14.53
15	466	7339	15.75
20	596	9474	15.90

Table 4.4: Tree repair times v.s. the number of wireless link changes in mobility simulations

We observe that the simulator detects physical tree segmentation in stable network simulations. This phenomenon is due to control message especially CA messages loss during their transmission. Multicast tree is consequently not correctly constructed and does not cover all group receivers for that period. In dynamic networks, local recovery abandons branch repair after several attempts and leave periodical tree refresh to overcome such errors. This leads to an augmentation of tree fragmentation. However, when maximum movement speed is 20 m/s, simulator detects about 122 times tree broken over total 1696 reports during a simulation. This means that even in high mobility scenarios, MRDC multicast tree provides connectivity in 92% of time.

Multicast Group Size

We varied the number of multicast group size from 5 to 40 members to investigate the scalability of the protocol. The number of source is fixed at 1 and maximum speed at 1m/s to reduce as much as possible the affect of node's movement.

The performance of multicast tree as a function of group size is shown in Table 4.5. These results show that MRDC multicast tree scales well and is robust facing to group growth. The number of non-member routers increases firstly and then decreases. When the group size is small, group members may position in different direction from the core. To cover group member placed in different directions, MRDC should involves more nodes in multicast routing. But as the number of routers reaches a certain value, the multicast tree can cover most part of the network. Only a small number of extra nodes are needed to connect the new members which are in uncovered area. As a result, the number of average interior node

Group size	Average non-member routers	Average interior node	# of tree repair times	# of tree broken times
5	4.23	5.63	17	5
10	6.90	9.33	30	6
15	7.32	11.10	40	8
20	7.23	12.61	51	9
25	6.80	13.97	57	11
30	6.02	15.14	64	15
35	4.88	16.16	72	11
40	3.61	16.90	76	10

Table 4.5: Performance of MRDC multicast tree as a function of Multicast group size

logarithmically increases with the number of group members. To deliver multicast packet to 5 more receivers, only one more forwarding is sufficient when the group already contains 30 members. And from 35 members to 40 members, the forwarding overhead increases less than one.

As the group size increases, the size of multicast tree increases. As a result, multicast tree exposes more and more to wireless link changes. MRDC should run more local recovery to maintain these connectivities. On the other hand, if the trunk of multicast tree, which is formed by interior nodes, keeps physically connecting and covering most part of the network, the movement of node creates less tree segmentation. Some leave nodes may just move from the coverage range of one interior node to another interior node. Or when an interior node moves away, other interior nodes will cover its leave nodes. Therefore, the frequency of tree broken times decreases after 30-member scenarios since the coverage of multicast tree increases.

Number of Senders

In this experiment, we examine the performance of multicast tree constructed by MRDC with different number of senders which ranges in the set 1,2,3,4,6,8,12. The approach used in the simulations to transmit broadcast messages (CA messages, JI messages) and multicast packets on top of an IEEE 802.11 protocol is by flooding. Naively broadcasting by flooding may cause serious redundancy, contention, and collision in the network, which is called broadcast storm problem in [57]. We analyzed the collision problem in Section 3.4. The higher the network load is, the greater the possibility of control message loss exists. However, the contention problem is also important because it delays control message transmission. This experiment thus demonstrates the robustness of MRDC facing to control message lost or delayed. The multicast group size is set to 20. Node mobility speed is inferior to 1m/s so that the impact of topology change can be ignored.

Number of Senders	Average non-member routers	Average interior node	# of tree repair times	# of tree broken times
1	7.23	12.61	51	9
2	7.18	12.57	49	25
3	7.20	12.57	49	30
4	7.21	12.59	47	32
6	7.19	12.53	48	42
8	7.20	12.55	46	86
12	7.44	12.74	45	169

Table 4.6: Performance of MRDC multicast tree as a function of Number of senders

Table 4.6 demonstrates the performance of MRDC multicast tree as a function of the number of multicast senders. It shows that MRDC tree is sensible to control message loss. Tree broken times increases dramatically when the number of senders changes from 6 to 8 then 12. At the same time, average interior node number and tree repair times keep nearly unchanged. Control message and multicast packets contend on medium access because they share the same wireless channel. Heavy traffic load causes high loss rate due to collision for the broadcast control messages (CA message, JI message) and delays the transmission of the unicast control messages (RAR, RAA and Recovery messages) because of heavy contention. Losing or delaying a control message has direct effects on the performance of MRDC. For example, if a node fails in transmitting a CA message to its neighbors, it might make the constructed multicast tree incorrect and/or suboptimal. Multicast tree does not contain the necessary nodes to covers all group members, which reduces the number of router on the tree. On the other hand the shortest path is not discovered and tree is constructed by a longer path, which increases the number of router on the tree. As a result, we can observe that both the number of interior nodes and average non-member routers keep stable till 8-sender traffic scenarios and finally slightly increase in 12-source scenarios. CA message transmission failure and delayed RAR and RAA message transmission increases the time for multicast tree re-construction. The long duration of tree reconfiguration results in tree broken times increasing quickly in high load networks because simulator considers the tree is broken if it runs the tree test before MRDC finishes the multicast tree reconfiguration.

The above simulations results show that MRDC provides correct multicast tree in most of time. It can efficiently support multicast delivery in most cases. However, the trees become fragile when node's mobility increases. This protocol relies greatly on the correctness and prompt of routing message transmission. These points should be taken into account in the future works.

4.6 Protocol Comparison

After choosing the key parameters of MRDC and analyzing the efficiency and robustness of MRDC multicast tree, in this section, we study the performance of MRDC with some other multicast routing protocols to understand the behaviors of MRDC under different mobility and traffic scenarios. We set PERIOD_REF to 5 seconds, INTR to 0.65 and QLEN to 6 because they give an optimal performance.

4.6.1 Comparison Multicasting Protocols

For comparison, we make two versions of MRDC: MRDC-unicast, which forwards packets only in unicast mode, and MRDC-broadcast, which forwards packets only in broadcast mode. The MRDC integrated with adaptive forwarding mechanism is denoted as MRDC-adaptive in the simulations.

The Rice Monarch project [51] at rice university has made multicast extensions for ns2 [9]. The extensions include implementations of the Adaptive Demand-Driven Multicast Routing protocol (ADMR) [35] and the On-Demand Multicast Routing Protocol (ODMRP) [24] for routing in wireless multi-hop ad hoc networks [1]. We take these two protocols as references in protocol performance comparison. Here, we make a brief overview of these two protocols.

Adaptive Demand-Driven Multicast Routing protocol (ADMR)

ADMR constructs a source-oriented loose multicast tree on traffic demand. In ADMR, receivers must explicitly join a multicast group. Sources periodically send a network-wide flood, but only at a very low rate in order to recover from network partitions. In addition, forwarding nodes in the multicast tree may monitor the packet forwarding rate to determine when the tree has broken or the source has become silent. If a link has broken, a node can initiate a repair on its own; if the source has stopped sending any forwarding state is silently removed. Receivers likewise monitor the packet reception rate and can rejoin the multicast tree if intermediate nodes have been unable to reconnect the tree.

To join a multicast group, an ADMR receiver floods a MULTICAST SOLICITATION message throughout the network. When a source receives this message, it responds by sending a unicast KEEP-ALIVE message to that receiver, confirming that the receiver can join that source. The receiver responds to the KEEP-ALIVE by sending a RECEIVER JOIN along this same unicast path. In addition to the receiver s join mechanism, a source periodically sends a network-wide flood of a RECEIVER DISCOVERY message. Receivers that get this message respond to it with a RECEIVER JOIN if they are not already connected to the multicast tree.

Each node which acts as a receiver or forwarder maintains a counter of recently received packets, and if a certain number of consecutive packets are not received by a receiver, it concludes that it has become disconnected for the group and it

[1]The source code can be found from http://www.monarch.cs.rice.edu/multicast_extensions.html

starts a repair process. A node that is a pure receiver (and not a forwarder for that source/group) simply rejoins the group by sending a MULTICAST SOLICITATION message. A node that is only a forwarder sends a REPAIR NOTIFICATION message down its subtree to determine whether it is the closest node to where the packet loss is occurring. Any downstream nodes cancel their own disconnect timers when they get this notification. Once a node has determined that it is the most upstream node that has been disconnected, it transmits a hop-limited flood of a RECONNECT message. Any forwarder which receives this message forwards the RECONNECT up the multicast tree to the source. The source in return responds to the RECONNECT by sending a RECONNECT REPLY as a unicast message that follows the path of the RECONNECT back to the repairing node.

A receiver keeps track of how many times it has had to initiate a repair due to a disconnection timeout. If this number reaches a certain threshold then the receiver believes that it has encountered a situation of high mobility. In the next RECEIVER JOIN message sent to the source, the receiver sets a high mobility flag as a signal to the source indicating that the network is encountering high mobility. When the source receives a particular number of join messages with the high mobility flag on, then it switches to flooding for a limited period. During flooding, all the data packets are sent as network-wide flood and all repair messages are suppressed.

In multicast forwarding, it obtains redundant retransmission by flooding multicast datagram in the tree.

On-Demand Multicast Routing Protocol (ODMRP)

ODMRP creates on traffic demand a mesh which contains the selected (shortest) path of each source destination pair, thus provides path redundancy. Nodes on the mesh forms a "forwarding group" which forwards multicast packets via flooding (within the mesh), thus providing further redundancy. A soft state approach is taken in ODMRP to maintain multicast group members. Thus, 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 multicast sources have packet to send, but do not have routing or membership information, they broadcast a Join-Query control message to the entire network. When a node receives a non-duplicate Join-Query, it stores the source ID and the sequence number in its message cache to detect any potential duplicate. The routing table is updated with upstream node ID and the node rebroadcasts the message. When the Join-Query message reaches a multicast receiver, it creates and broadcast a Join-Reply to its neighbors. When a node receives a Join-Reply, it checks whether the next 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 it is part of the forwarding group and sets the forwarding group flag. It then broadcasts its own Join-Reply built on matched entries. The next hop node ID field contains the information extracted from its routing table. In this way, each forwarding group member propagates the Join-Reply until it reaches the multicast source via selected path. This process constructs (or updates) the routes from sources to receivers and builds a mesh of nodes. Multicast senders refresh the membership and updates routes by sending Join-Query control message periodically.

ADMR and ODMRP Simulation Parameters

In performance comparison simulations, the parameters of ADMR and ODMRP are identical to those used in [35]. This means in ADMR, the periodic data flood interval is 30 seconds and 1.2 for multiplicative factor of the average inter-packet time in the absence of data, and 2 missing packets to trigger disconnection detection. For ODMRP, the Join-Query flood interval is 3 seconds, and a forwarding state lifetime of 3 times this interval (a total 9 seconds).

4.6.2 Metrics

We have used the following metrics in comparing protocol performance. Some of these metrics are suggested by the IETF MANET working group for routing/multicasting protocol evaluation [58]. We use the following four metrics in performance comparison:

- **Packet delivery ratio**: Identical to that in section4.4.1
- **Average end-to-end delay**: Identical to that in section4.4.1
- **Number of data packets transmitted per data packet delivered**: "Data packets transmitted" is the count of every individual transmission of data by each node over the entire network. This count includes transmission of packets that are eventually dropped and retransmitted by the intermediate nodes. Note that in unicast protocols, this measure is always equal to or greater than one. In multicast, since a single transmission (broadcast) can deliver data to multiple destinations, the measure may be less than one.
- **Number of control bytes transmitted per data byte delivered**: Instead of using a measure of pure control overhead, we chose to use the ratio of control bytes transmitted to data bytes delivered to investigate how efficiently control messages are utilized in delivering data. Note that not only bytes of control messages (e.g. beacons, route updates, join requests, etc.), but also bytes of data packet headers are included in the number of control bytes transmitted.

The two later metrics concerns bandwidth utilization. The number of bytes transmitted per data byte delivered can be considered as uniformed forwarding overhead and the number of bytes transmitted per data byte delivered as uniformed control overhead. The sum of these two metrics is uniformed bandwidth consumption of each protocol.

4.6.3 Result analysis

This section introduces the simulation results of protocol comparison. We firstly test the performance of the multicast routing protocols in moderate dynamic networks with different traffic scenarios. And then we choose the 4-source traffic scenario to obtain the behavior of these protocols in different mobility networks. The results with confidence can be found in Annex A.1.1.

Number of sources and groups

In a first step, we set the maximum movement speed of mobile nodes to 5 m/s and each group contains 10 members. We vary the number of sources from 2 to 8 to see the performance of these five protocols. We choose 8 sources as the maximum traffic load because this traffic scenario reaches the maximum network capacity of MRDC-unicast. According to the calculation of [59], the maximum throughput of a node in a regular lattice network is $1/12$ of the channel capacity. For 512-byte packets transmitted with RTS/CTS option, this is $\frac{1}{12} * \frac{512}{512+40+39+47} * 2 = \frac{1}{12} * 1.6$ Mbps, or 0.13 Mbps. Supposing a node is located in a hot spot where all traffic travels through it with a bit rate of $4 * 512 * 8 = 16$kbps, in 8-source case, its maximum capacity is reached. As the number of sources increases, the medium access contention and bandwidth consumption also increases. This contention comes from different groups, the same group but different sources and also the competitions between multicast packets and control messages. Thus, this simulation tests the efficiency of these protocols in different network load cases and at the same time their robustness against control message error.

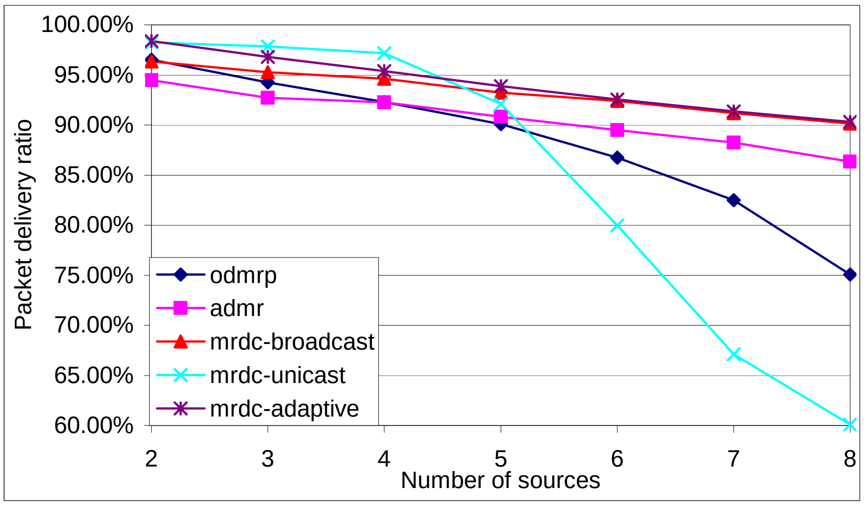


Figure 4.4: Packet delivery ratio v.s. Number of source

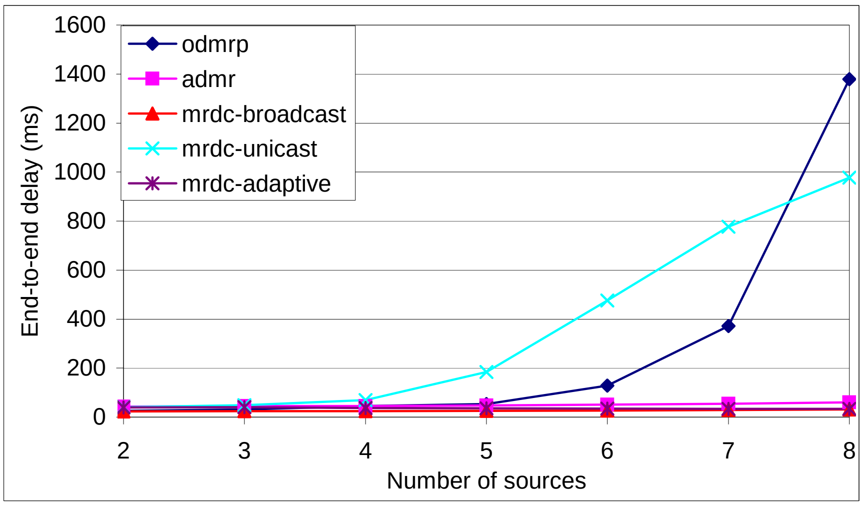


Figure 4.5: End to End delay v.s. Number of sources

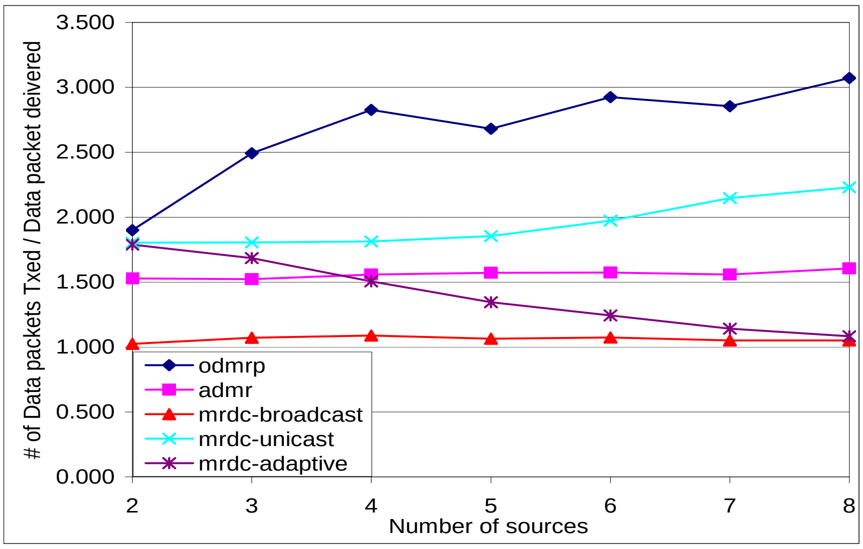


Figure 4.6: Number of data packets transmitted per data packet delivered v.s. Number of sources

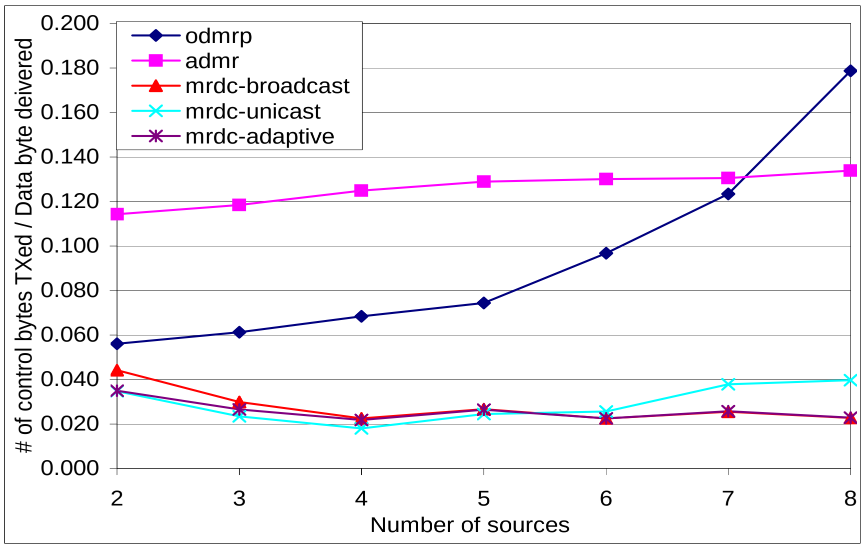


Figure 4.7: Number of control bytes transmitted per data byte delivered v.s. Number of sources

The packet delivery ratio of these protocols is illustrated in Figure 4.4. In low load networks in which the number of source is less than 5, MRDC-unicast delivers the most packets and ADMR gives the worst delivery ratio. MRDC-unicast and odmrp degrade more quickly than other protocols. MRDC-broadcast has the most stable delivery ratio. On the other hand, except two points (3-source and 4-source cases), MRDC-adaptive is the best protocol in terms of packet delivery ratio.

In terms of transmission delay, as shown in Figure 4.5, MRDC-broadcast performs the best. In fact this protocol maintains the end-to-end delay at about 30 ms. MRDC-adaptive and ADMR generates a little more transmission delay than MRDC-broadcast. MRDC-unicast creates the most transmission delay in the scenarios of less than 7 sources and then is overtaken by ODMRP.

Forwarding overhead which is represented by number of data packets transmitted over data packet delivered is demonstrated in Figure 4.6. MRDC-broadcast provide the best performance in terms of forwarding overhead.

The control information efficiency of five protocols are compared in Figure 4.7. All three approaches of MRDC generate about three times less control overhead than ODMRP and ADMR.

As the network load increases, the performance metrics of all protocols degrade with a different trend. MRDC-broadcast is the most stable protocol in these simulations. It creates the least transmission delay, forwarding overhead and control overhead to provide a reasonably good packet delivery. The performance variation of ADMR is also limited but that of MRDC-unicast and ODMRP is much greater than others. Let's study these protocols one by one to explain these phenomena.

ODMRP constructs mesh through including the shortest path of every source-destination pairs. Each source periodically refresh the path to destinations. Routing overhead therefore is a function of source number (Figure 4.7). ODMRP creates forwarding state within nodes in the network, that is expired after a fixed timeout. This timeout is set to a multiple of the periodic JOIN QUERY flood interval in order to ensure that loss of the flood packets will not cause disruptions in the delivery of multicast data. However, this mechanism leads to the creation of redundant state in the network, since new nodes may become forwarders for a group, while forwarders created during a previous periodic flood still have a set forwarding flag and may overhear packets for that group. While the redundancy that ODMRP creates increases its resilience to losses, it significantly increases the load on the network (see Figure 4.6). As a result of the high load, overall network performance degrades and transmission delay goes up (Figure 4.4 and 4.5).

ADMR generates the most routing overhead according to Figure 4.7. This is on one side because it adds on each traffic packet a packet header which contains 32 byte and sometimes 40 bytes. On the other side, ADMR constructs source-based tree. The fact that each source reconfigures its tree periodically results in routing overhead being a function of source number. ADMR also creates redundant state in the network when, as a result of tree breakage and repair, the forwarding tree no longer includes certain nodes that were part of the tree before the breakage of tree. However, nodes that forward for a source in ADMR expire their forwarding state when there are no downstream that are interested in receiving the multicast packets through them. It bears much less forwarding overhead than ODMRP (Figure 4.6) and delivery delay (Figure 4.5). However, the long period of source tree reconfiguration (30 seconds) compared with ODMRP and MRDC makes the tree structure not adapt to topology change in time and consequently delivers fewer packets than MRDC does in all cases and ODMRP does in low load cases (see Figure 4.4).

MRDC generates the least routing overhead because it uses group-shared multicast tree (see Figure 4.7). The routing overhead is mainly a function of group number. Adding new sources in a multicast group does not introduce important extra routing overhead. On the contrary, it improves the utilization of control messages since more traffic are delivered. That is why the ratio between control byte and data bytes delivered of MRDC-broadcast behaves as wave. Theoretically, group-shared tree generates more forwarding overhead than source-based tree in delivering traffics of non-core sources. However, MRDC does not create redundant state in the network this fact permits MRDC-broadcast become the protocol which generates the least forwarding overhead (see Figure 4.6). The forwarding overhead difference between MRDC-unicast and MRDC-broadcast is about 0.75 in the simulations where MRDC-unicast correctly maintains multicast trees (when the number of sources is smaller than 5). Using the result of 2 sources in Table 4.7, the forwarding overhead of unicast case is about $(10 + 6.66 - 1)/9 = 1.74$ and that of broadcast case is about $9.21/9 = 1.02$. This result confirms our previous discussion of forwarding overhead in these two modes. No redundant path in delivery structure makes the multicast trees of MRDC sensible to control message loss. MRDC-unicast aggravates this sensibility since this protocol delivers multicast packets respecting tree structure. High forwarding overhead generated by MRDC-unicast delays control message transmission and even makes some messages lose. As a result the observed tree broken times in MRDC-unicast increases dramatically as the network load increases, while the multicast trees of MRDC-broadcast are well maintained (see Table 4.7). Thus, we can see that when the number of sources is less than 5, multicast tree can be correctly established so that MRDC-unicast gives the best packet delivery ratio (see Figure 4.8) due to RTS/CTS option. Superior to 5, the performance of MRDC-unicast degrades quickly and MRDC-broadcast outperforms all the other protocols in both packet delivery ratio and end-to-end delay. The simulation results also show that MRDC-adaptive not only provides a compromise between MRDC-unicast and MRDC-broadcast but also improves the packet delivery ratio of MRDC-broadcast by 3% in low load cases.

Number of Sources	Average non-member routers	Average interior node	Tree broken times (broadcast)	Tree broken times (unicast)
2	6.66	9.21	40	30
3	6.74	9.26	39	31
4	6.75	9.27	35	39
5	6.81	9.29	42	90
6	6.77	9.29	38	160
7	6.75	9.30	41	218
8	6.78	9.30	41	247

Table 4.7: Multicast tree of MRDC-broadcast and MRDC-unicast as a function of Number of sources

MRDC-broadcast resists best face to network load increase and offers sometimes the best performance. MRDC-unicast is the most sensible to network load increase. Unicast mode requires more bandwidth than broadcast mode. In high load networks, this mode generates congestions and delays the transmission of not only multicast data packet delivery but also routing messages. Table 4.7 shows that the number of multicast tree breaks under MRDC-unicast dramatically increases as the network load does. Here, we only count physical tree fragmentation, while the number of logical fragmentation is much bigger than the number given in this table. Tree-based protocols depend more than mesh-based one on the correctness and punctuality of routing message transmission to maintain delivery structure. Therefore, the performance of MRDC-unicast degrades quickly when network load increases. MRDC-broadcast delivers the most packets to receivers in high load network. On one hand, tree structure contains fewer routers than mesh structure, hence, creates less collision and congestion than mesh when routers operate in broadcast mode. On the other hand, MRDC is more active than ADMR in terms of tree maintenance. MRDC-adaptive gives a compromised packet delivery ratio in this simulation.

Mobility Pattern

In the second step, we vary the maximum movement speed of mobile nodes from 0 m/s (stable network) to 20 m/s and choose a 4-source-traffic scenario in which two multicast group are defined and each has 10 members. The aim of this simulation is to evaluate the performance of these multicast routing protocols when topology changes and in certain degree of media access contention (e.g. among control messages and multicast packets).

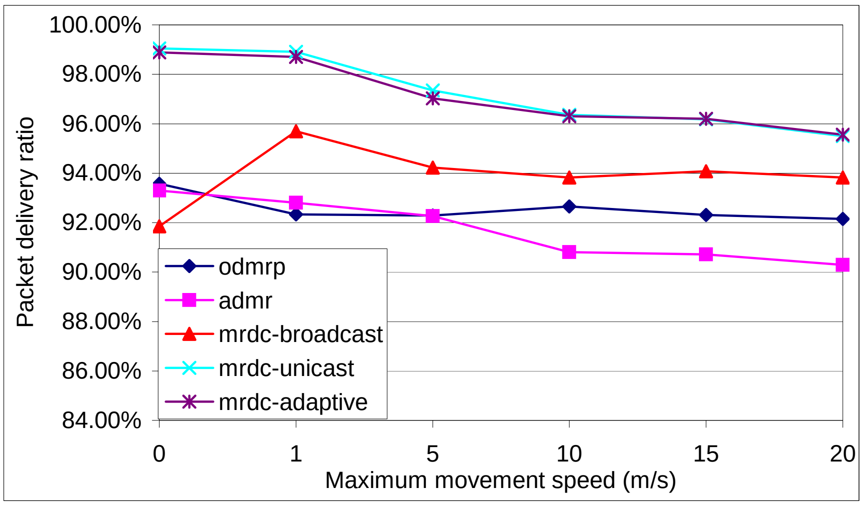


Figure 4.8: Packet delivery ratio v.s. Maximum movement speed

The results of packet delivery ratio (Figure 4.8) show that under low media contention environment, MRDC-unicast and MRDC-adaptive have almost the same results and both of them outperform other three protocols. MRDC-broadcast has a peak at 1 m/s scenarios and then slightly decreases to 94%. ADMR provides the worst packet delivery when maximum movement speed exceeds 5m/s.

The average end-to-end delay of five protocols is illustrated in Figure 4.9. MRDC-broadcast creates the least transmission delay (smaller than 30 ms), while MRDC-unicast creates the most sometimes reach 70 ms.

Figure 4.10 compares the number of data packet transmission to successfully deliver a data packet to a receiver under these five protocols. MRDC-broadcast requires the minimum number of packet transmission while ODMRP needs 3 times more.

The number of control bytes transmitted per data byte delivered as a function of maximum movement speed of five protocols is demonstrated in Figure 4.11.

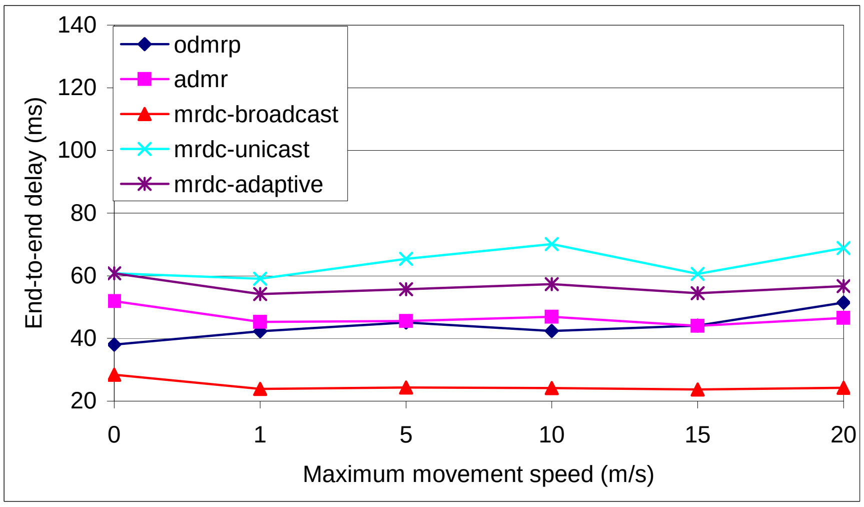


Figure 4.9: End to End delay v.s. Maximum movement speed

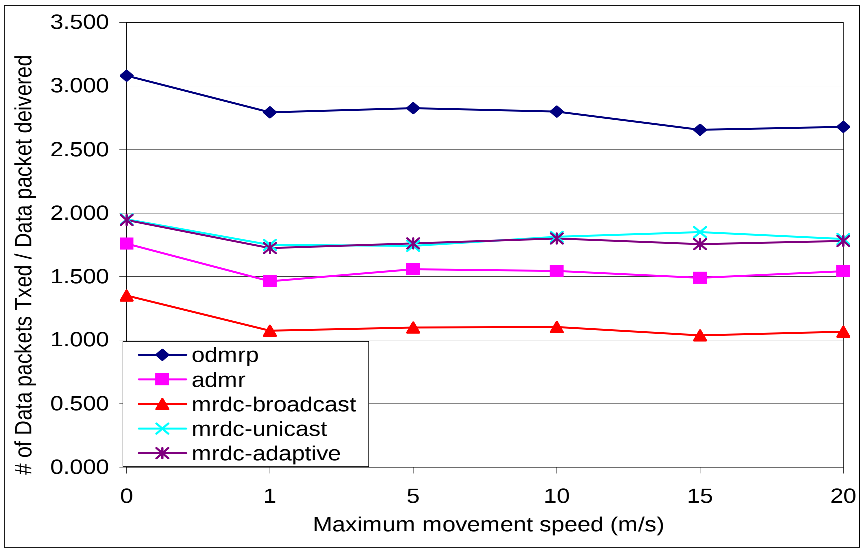


Figure 4.10: Number of data packets transmitted per data packet delivered v.s. Number of sources

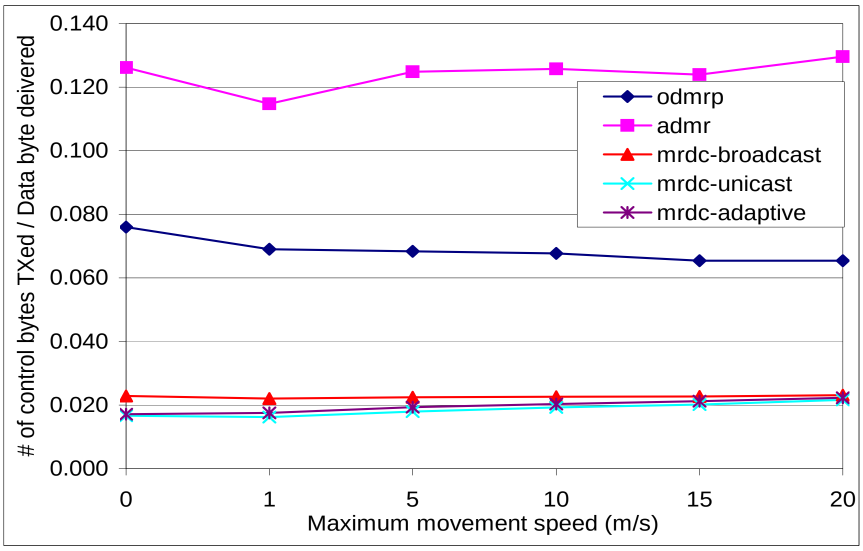


Figure 4.11: Number of control bytes transmitted per data byte delivered v.s. Number of sources

ADMR produces the most uniformed control bytes. It is MRDC-unicast that makes the least. The uniformed control bytes under MRDC-broadcast remains stable but under MRDC-unicast that increases with the movement speed. In 20 m/s these two protocols have almost the same results.

To understand the above behaviors, we should analyze the impact of node's mobility on the performance of these protocols.

ODMRP is source-based in control part but group shared in forwarding part. When a source reconfigures routes to receivers these routes are also available for delivery packet from other sources. More sources in a group implicitly increase the frequency of mesh reconfiguration, that makes mesh structure more robust against topology changes. This protocol depends on the periodical Join-Query and does not do local structure repair. Recall that the Join-Query flood interval used in the simulations is 3 seconds. Here two sources per group means a part of mesh is updated every 1.5 seconds. This frequent reconfiguration leads to that node's mobility has nearly no effect on its performance in dynamic networks. However, the transmission delay increases because it buffers multicast packets before completing mesh reconfiguration.

ADMR employs mainly local recovery against topology change since it has fewer redundant forwarding nodes. Figure 4.11 shows that ADMR gives the biggest the control bytes over data byte but this ratio keeps stable. However, we observed an increase of control messages as node's mobility increases in relation to tree maintenance. This increase is flushed by packet header. In ADMR, tree member should wait for a while before it affirms link break and runs local recovery after topology changes. This mechanism degrades the packet delivery ratio as network becomes more and more dynamic. Other metrics maintain stable.

MRDC offers better packet delivery ratio than ADMR and ODMRP in dynamic networks. The multicast tree maintenance mechanism used by MRDC is similar to ADMR. However, MRDC reconfigures multicast trees more frequently than ADMR does. That provides a tree adapting better to network topology. MRDC-broadcast keeps stable and offers the best performance in terms of the shortest transmission delay and the lowest forwarding overhead. Although the tree structure does not contain redundant path, broadcast mode benefits broadcast capacity of wireless interface to create redundant transmission so that it could improve the packet delivery ratio. On the contrary, unicast mode does not benefit this advantage, which provokes its packet delivery ratio into decreasing more quickly than in broadcast mode. MRDC-adaptive has almost the same behavior as MRDC-unicast since the forwarding mechanism takes only traffic related metrics into account and the traffic scenario used in simulations favors unicast mode. However, some minor differences demonstrate that certain nodes operate in broadcast mode, which degrades slightly the packet delivery ratio due to unreliable transmission but improves slightly the same metric in high dynamic networks thanks to redundant transmission.

In stable network, when all protocols forward multicast datagram on broadcast (ODMRP, ADMR and MRDC-broadcast), MRDC-broadcast provides the worst packet delivery ratio. This is caused by group-shared tree structure with significantly packet collision in mac layer. Using IEEE802.11, the longer distance a packet goes through, the greater possibility there is it to be lost on the route. The delivery structure of ODMRP and ADMR contains shortest path for each source receiver pairs. Sources can transmit multicast packets directly through these paths to destinations. Group-shared tree used by MRDC only contains the shortest path from receivers to core (the first source in MRDC) but not to non-core source(s). Non-core sources should send their packets to core in order to reach those group receivers which are not in their sub tree. This tree structure increases the path length from non-core source to receivers. For example, in figure 4.12, all receivers are sited no more than two hops far away from S1 in the tree. However, packets from S2 should travel five hops to reach R2 and four hops to R1 after getting through S1, nevertheless three hops are enough in ADMR. Therefore, in MRDC the traffic generated by non-core source suffers more transmission failure which degrades total packet delivery ratio. The position of nodes in stable network aggravates this problem because the distances from group members to core are longer and consequently the tree are lager than that in dynamic networks as shown in Table 4.8. Therefore MRDC-broadcast delivers the least packets to receivers. However, since the multicast trees are correctly established in these simulations, unicast mode can improve the packet delivery ratio significantly by reducing packet collision in MAC layer.

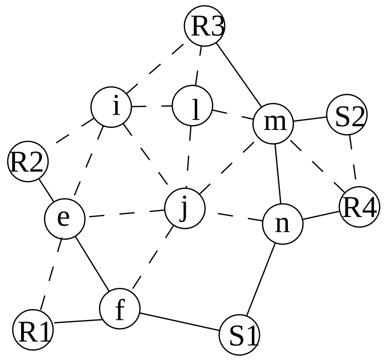


Figure 4.12: MRDC group-shared multicast tree

Maximum speed (m/s)	Distance from members to core	Average interior node
0	30.8	11.3
1	23.8	9.4
5	22.8	9.4
10	22.6	9.3
15	21.6	8.9
20	21.4	9.0

Table 4.8: Distance and Multicast tree size under MRDC-broadcast as a function of Maximum speed

4.6.4 General Discussion

In this section, we compare the performance of ODMRP, ADMR and MRDC operating in different transmission modes (broadcast, unicast and adaptive). These protocols use different types of delivery structure. ODMRP constructs a group shared mesh. ADMR uses source-based tree. MRDC provides group-shared tree. During packet delivery, the former two create redundant forwarding state within nodes in the network against route broken caused by topology change or control message loss. Whereas MRDC does not use this technique. The simulation results show that the greatest difficulty for those protocols to use redundant forwarding state is how to find the compromise between robustness and efficiency. Rich redundant forwarding state permits routing protocol maintain its performance in dynamic networks without the requirement of extra routing overhead. However, the forwarding overhead as a result of redundant forwarding state degrades protocol's performance in high load cases. As for MRDC, it provides the best performance since this protocol uses the least connectivity and in most cases control messages are correctly and duly transmitted. However, the group-shared trees do not provide the same performance for non-core sources. When group members are badly distributed in the network, MRDC provides a worse packet delivery compared to other two protocols under the condition that all of them broadcast multicast packets.

Unicast mode creates more forwarding overhead and depends more on the correctness of multicast tree than broadcast mode since multicast transmission strictly respects tree structure. Thus MRDC-unicast degrades more quickly than MRDC-broadcast as network load and mobility increase. MRDC-adaptive sometimes outperforms both MRDC-unicast and MRDC-broadcast because it might take advantage of unicast mode and broadcast mode at the same time. In fact, when MRDC broadcasts multicast packets, a tree member can receive multicast packets from neighbors which are not listed in its multicast routing entry. That provides a certain redundancy to improve packet delivery. Unicast multicast packets offers certain degree of reliability for transmission but deprive transmission redundancy. In ideal cases, MRDC-adaptive uses broadcast mode in hot spot to create transmission redundancy and avoid congestion and uses unicast mode to assure transmission in other region so that this protocol can provide the highest packet delivery.

4.7 Conclusion

In this chapter, we studied the performance of MRDC in ns2. At first, we selected key parameters of MRDC: tree refresh period PERIOD_REF and thresholds of mode selection. A longer period generates less routing overhead for multicast tree refresh but reduces the robustness and efficiency of MRDC since the packet delivery ratio decreases as node's mobility changes. On the other hand, shorter refresh interval makes MRDC robust against topology changes at the cost of high control overhead. The simulation results show a 5 seconds period gives a compromise between robustness and low routing overhead. As for mode selection thresholds, a small INTR and QLEN make nodes easily think that they are in hot spot and switch to broadcast mode. This is appreciated in high load network to avoid congestion created by unicast mode. However, it is not welcome in low load network because broadcast mode generates significant MAC layer packet collision. ($INTR = 0.6, QLEN = 8$) pair shows a good trade-off between delivery ratio and delivery delay, and is used in performance comparison simulations.

Then, we evaluated performance of MRDC under different movement and traffic scenarios. The evaluation contains two parts: the first, the characteristics of MRDC multicast tree and the second, the performance comparison. The simulation results of MRDC multicast tree such as the average number of interior nodes and the average number of non-group-member routers allow us to estimate the routing overhead and forwarding overhead of MRDC. The results also show that MRDC multicast tree scales well in terms of tree size as the group size increases, while the number of tree repair times is proportional to link changes during a simulation. On the other hand, the correctness of MRDC greatly depends on the transmission of control message. As the network load increases, tree fragmentation becomes more frequent owing to the loss of control messages. In performance comparison, MRDC operating in different transmission modes (broadcast, unicast and adaptive) are compared with other two multicast routing protocols: ODMRP and ADMR through four metrics: packet delivery ratio, transmission delay and routing and forwarding overhead. Due to tree structure, MRDC-broadcast generates the least forwarding overhead. MRDC-adaptive provides optimal results and sometimes the best multicast packet delivery regarding network load or node's mobility increasing.

During the simulations we find that the packet loss mainly comes from some reasons. The first is physical condition, for example network partition makes some group members unreachable by other members. Routing protocol can do nothing on this problem. The second is low layer transmission failures such as packet collision occurred in MAC layer. Routing protocol can alleviate this problem through redundant transmission. However too much redundant can aggravate packet collision and even created congestion. The third is routing protocol problem for example if delivery structures are not constructed in time or routing protocol does not repair the fragmentation in time, multicast packets cannot be delivered to the involved receivers during that period. MRDC does have this kind of problem because it does not preserve forwarding state during tree refresh. One solution is to introduce *forwarding* state into the states of multicast entry. Multicast routers firstly degrade from tree member to forwarding group member when receiving a CA message. Forwarding group members continue forwarding multicast packets during tree refresh. The forwarding group membership expires after a short while and the node either becomes multicast tree member or leaves multicast tree by setting entry state to *non-forwarder* after tree refresh. In this way, multicast delivery continues even during multicast tree reconfiguration.

Chapter 5

RELIABLE MULTICASTING FOR AD HOC NETWORKS

When MRDC provides best effort non guaranteed multicast delivery, some applications of MANETs have requirement beyond this. For example file distribution, whiteboard and Internet games are sensible to packet loss and require reliable data transfer to group receivers. To achieve reliability some error recovery mechanism for lost packets has to be implemented. Automatic repeat request (ARQ) is one widely used mechanism. This mechanism makes source or some other nodes to retransmit lost packet. In protocols designed for small (local area) multicast groups ARQ mechanism is usually realized at the sender, which is responsible for processing positive or negative acknowledgements (ACKs/NAKs) and for retransmitting packets. Examples for such sender-originated reliable multicast protocols are MTP [60] or AMTP [61]. However, with increasing size or geographic spread of the multicast group the performance of these protocols gets worse, and more scalable protocols are required. Some reliable multicast protocols have been developed for that purpose. Besides the sender these protocols allow either dedicated receivers (e.g. RMTP [62], SRM [63], TMTP [64]) or routers (e.g. AER [65], ARM [66], PGM [67]) to handle ACKs/ NAKs and to retransmit packets for members in their local environment (we will call these protocols receiver-assisted and router-assisted, respectively). Thus the cost for retransmissions can be decreased and the ACK processing load on the sender is relieved [68].

The properties of MANETs listed in 2 make the design of a reliable multicast protocol for MANET a challenging task. Such a protocol should consume little bandwidth and node's resources (e.g. processing, energy, memory space) but guarantee reliable delivery in a high packet loss rate environment. Some reliable multicasting protocols are studied APRM [69], Family ACK Tree (FAT) [70], [71], Anonymous Gossip [72], Reliable adaptive lightweight multicast protocol (RALM) [73], Reliable multicast alogrithm (RMA) [74] and ReMHoc [75]. Seeing the drawback of these protocols, we focus our interesting on retransmission method and design a reliable multicasting protocol, called active reliable multicast protocol with intermediate node support (ARMPIS), which extends both receiver-assisted and router-assisted scheme to MANET.

Our main contribution is that ARMPIS distributes packet storage and retransmission responsibility to all nodes which overhear multicast packets. These nodes are called intermediate nodes. According to this definition intermediate nodes include not only group members and nodes which forward multicast packets but also the neighbors of multicast packet forwarders. In reliable multicast, retransmission load of source is a function of link loss rate, size of network and group. In MANET, link loss rate is relatively high due to wireless interface and node's mobility. Thus, we think it is necessary to make intermediate nodes share retransmission tasks. Retransmission made by intermediate nodes lead to recovery packets travel a shorter route than original ones traveled and consequently achieves a higher recovery success and lower bandwidth consumption. Intermediate nodes need to store multicast packets for retransmission while limited memory prevents them to store all packets. Our strategy is that in ARMPIS, intermediate nodes randomly store overheard multicast packets to reduce duplicated cache among neighbors and a node queries its neighbors about the request packets before forwarding a retransmission request. Furthermore, this protocol needs no other control packet than negative acknowledge message (NACK) and independent of unicast routing protocols. The route to source is established by on-going traffic and retransmission paths are established during NACK forwarding. ARMPIS is initially designed for MRDC, which is a tree-based multicasting protocol, but it can also cooperate with mesh-based multicast routing protocols, such as [23], [20], [27], [24]. In the rest of this chapter, we do not specify which kind of underlying multicast routing is and use multicast delivery structure to indicate multicast tree and mesh in the rest paper.

The rest of this chapter is organized as following. In Section5.1, we first present in detail the mechanism of ARQ and analyze why it is necessary to make nodes other than source do retransmission in MANETs which are generally not large. Section5.2 briefly introduce current reliable protocols for MANET specially their shortcomings. Then Section 5.3 discuss ARMPIS in detail including system model, design principle and ARMPIS' procedures. Section 5.4 simulates the performance of ARMPIS on top of MRDC. Final Section 5.5 concludes this chapter by outlining remarks and pointing out future work.

5.1 ARQ Mechanism and Retransmission Load Analysis

ARQ is a mechanism used in reliable multicast protocols to provide error control. In this section, we briefly review this mechanism and then analyze the source retransmission load of different retransmission schemes on binary trees.

5.1.1 An Overview of ARQ mechanism

ARQ is a retransmission on demand mechanism, where the sender is alerted to packet losses through feedback from receivers and lost packets will be retransmitted by either the sender or other nodes[76]. An ARQ scheme can either be sender- or receiver-initiated. In a sender-initiated scheme, the sender maintains state information of receivers and detects packet losses. Receivers need to acknowledge every received packet by ACK to the sender. If the sender does not receive the ACK for a packet after time out, it will assume that the packet is lost and a retransmission or a congestion avoidance mechanism will be triggered. In a receiver-initiated scheme, receivers have the responsibility of detecting losses, e.g., by observing gaps in received packets. After a loss is detected, a NACK will be issued to report the loss and request retransmission. Usually, in multicast transmission, receiver-initiated schemes are more scalable than sender-initiated schemes [68], since the burden of maintaining reliability is distributed among receivers and NACKs are only issued when packet losses occur.

Since ARQ consists of feedback and retransmission, researchers discuss the efficiency of an ARQ reliable multicasting protocol from these two dimensions: the scalability of ACK or NACK message in both network view and individual router view and retransmission load. Feedback is generated by receivers and sent to the sender. It increases as the number of receivers and multicast packets. To reduce bandwidth consumption, feedback is aggregated to present the reception of a set of packets at receivers and downstream nodes at router. As for the retransmission load which is a function of the size and geographic spread of the multicast groupthe, receiver-assisted and router-assisted retransmission schemes are proposed. In receiver-assisted retransmission scheme, when a router receives a feedback it forwards this feedback to some other receiver in its sub-network instead of forward the feedback directly to the source. If fortunately the receiver has the required packets, it retransmit the packets. In order to reduce further bandwidth consumption nd reduce recovery latency, router-assisted retransmission scheme requires router store the multicast packets they forwarded for future retransmission.

5.1.2 Mathematic analysis of ARQ's Retransmission Load

We examine the source's retransmission load of three retransmission schemes: source-originated, receiver-assisted and router-assist protocols. We suppose that a multicast tree rooted at the source connects delivers packets to destinations which locate all at leaves. After one (re)transmission, a sub tree is constructured based on old one for next retransmission. This sub multicast tree contains only destinations which have not received the packet. All links have the same packet loss probability which is denoted as γ to facilit the analysis.

First, we consider one-level binary tree as shown in Figure 5.1 where one source deliver multicast packets to two destinations which locate in source's coverage range. The right part of this figure lists all possible retranmission tree. Tree A is

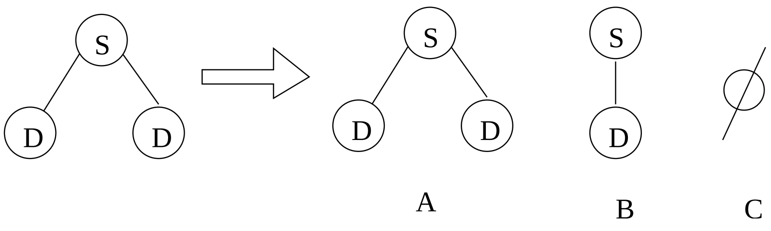


Figure 5.1: One-level binary tree and its sub trees

the result of transmissin failure to both destinations, while tree B is the case where one destination does not receive the packet and tree C is empty which represnt the successful delivery to two destinations. The probability distribution function of retransmission tree (number of retransmissions) can be determined based on a homogeneous discrete-time Markov chain (DTMC) shown in Figure 5.2.

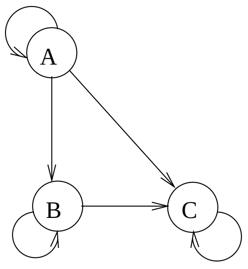


Figure 5.2: DTMC for one level binary trees

The corresponding transition probabilities can be written as matrix

$$\mathbf{P} = \begin{pmatrix} \gamma^2 & 2(1-\gamma)\gamma & (1-\gamma)^2 \\ 0 & \gamma & (1-\gamma) \\ 0 & 0 & 1 \end{pmatrix}$$

The initial state distribution of the DTMC is given by $\pi(0)$=($\pi_A(0)$, $\pi_B(0)$, $\pi_C(0)$)= ($\gamma^2, 2(1-\gamma)\gamma, (1-\gamma)^2$), since one transmission is already finished before retransmissions begin, i.e. we just have to consider the transition probabilities originating from state A. The probability to be in a certain state of the DTMC after n steps $\pi(n)$ can be determined simply by calculation of the *n*th power of P. We are only interested in being in state C, the case of empty tree, which indicates that all receivers have obtained the packet. Hence, we get

$$\pi_C(n) = \sum_{i=A,B} p_{i,C}^{(n)} \pi_C(0) = (1-\gamma^{n+1})^2$$

The probability f_n that exactly n retransmissions are required is given by $f_n = \pi_C(n) - \pi_C(n-1)$ for $n > 0$ and $f_n = \pi_C(0)$ for n = 0. The expected source's retransmission load can be written as

$$E[R](L = 1) = \sum_{n=1}^{\infty} n f_n \tag{5.1}$$

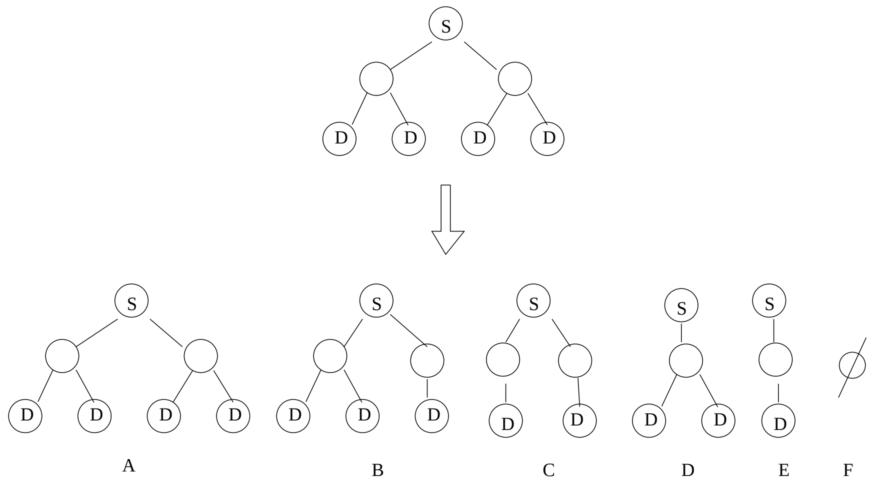


Figure 5.3: Two-level binary tree and its sub trees

Now, we analyze the source's retransmission load in a two-level binary tree as shown in Figure 5.3 under source-originated scheme. This multicast tree has six possible retransmission sub trees. The corresponding transition probabilities can be written as matrix

$$\mathbf{P} = \begin{pmatrix} P_{AA} & P_{AB} & P_{AC} & P_{AD} & P_{AE} & (1-\gamma)^6 \\ 0 & P_{BB} & P_{BC} & P_{BD} & P_{BE} & (1-\gamma)^5 \\ 0 & 0 & P_{CC} & 0 & P_{CE} & (1-\gamma)^4 \\ 0 & 0 & 0 & P_{DD} & P_{DE} & (1-\gamma)^3 \\ 0 & 0 & 0 & 0 & P_{AB} & (1-\gamma)^2 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

The transition probabilities are given by
$P_{AA} = \gamma^2 + 2(1-\gamma)\gamma^3 + (1-\gamma)^2\gamma^4$
$P_{AB} = 4((1-\gamma)^2\gamma^2 + (1-\gamma)^3\gamma^3)$
$P_{AC} = 4(1-\gamma)^4\gamma^2$
$P_{AD} = 2((1-\gamma)^3\gamma + (1-\gamma)^4\gamma^2)$
$P_{AE} = 4(1-\gamma)^5\gamma$
$P_{BB} = \gamma^2 + (1-\gamma^2)\gamma^2 + (1-\gamma)^2\gamma^3$
$P_{BC} = 2(2-\gamma)(1-\gamma)^2\gamma^2$
$P_{BD} = (1-\gamma)^2(\gamma + (-1\gamma)\gamma^2$
$P_{BE} = (1-\gamma)^3\gamma + 3(1-\gamma)^4\gamma$
$P_{CC} = \gamma^2 + 2(1-\gamma)\gamma^2 + (1-\gamma)^2\gamma^2$

$P_{CE} = 2(1-\gamma)^2(2-\gamma)\gamma$
$P_{DD} = \gamma + (1-\gamma)\gamma^2$
$P_{DE} = 2(1-\gamma)^2\gamma$
$P_{EE} = 1-(1-\gamma)^2$
The expected source's retransmission load can be derived from Formula 5.1.

For the receiver-assisted scheme we first have to divide the multicast group into subgroups. Let us assume that a receiver-assisted scheme would use three subgroups in our example (see Figure 5.4): $Group_1$ consisting of S, D_2 and D_3 (the source being responsible for retransmissions), $Group_2$ consisting of D_1 and D_2 (D_2 being the dedicated receiver) and $Group_3$ consisting of D_3 and D_4 (D_3 being the dedicated receiver). As a result, from point view of source, there are only

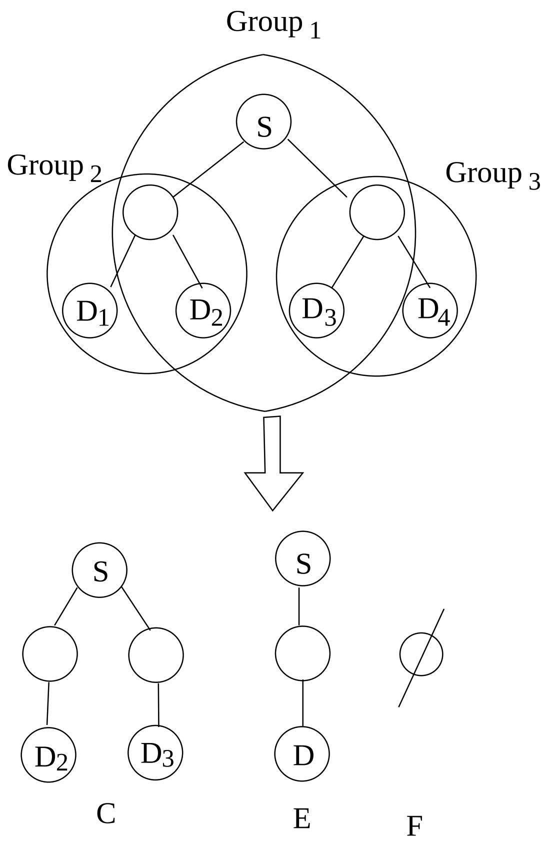


Figure 5.4: Subgroups for receiver-assisted retransmission

three forms of sub multicast tree which corresponding to sub-tree C, E and F in source-originated case. The transition probabilities matrix becomes

$$\mathbf{P} = \begin{pmatrix} P_{CC} & P_{CE} & (1-\gamma)^4 \\ 0 & P_{EE} & (1-\gamma)^2 \\ 0 & 0 & 1 \end{pmatrix}$$

The router-assisted scheme can be seen as source reliably transmit packets to its downstream nodes and then these downstream nodes guarantee packet delivery to their downstream nodes. Therefore, the soure's retransmission load is the same as the case of one-level binary tree.

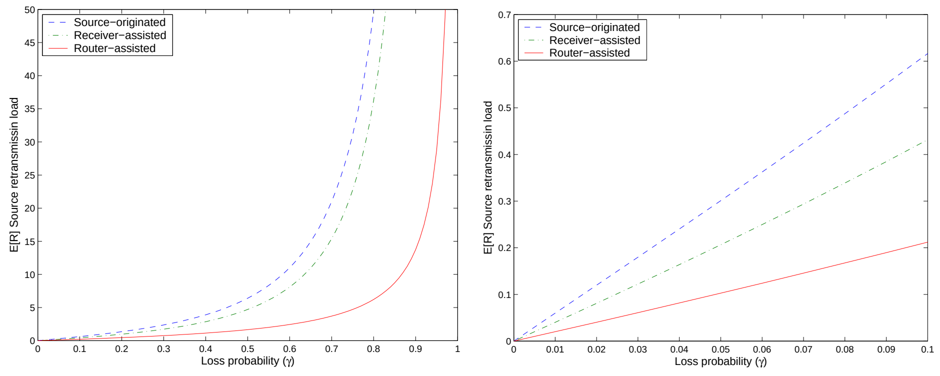


Figure 5.5: Retransmission load when varying γ

Figure 5.5 illustrates how the expected retransmission load for source-originated, receiver-assisted and router-assisted schemes varies with the loss probabilities γ. The right diagram is just an enlargement of the bottom left corner of the left one. Obviously the load of sender-originated scheme soon becomes unacceptable in two-level binary tree. Even for small loss probabilities the retransmission load is rather high. The improvement achieved by the receiver-assisted scheme is only marginal. In comparison to this both router-assisted scheme performs well, where all routers are responsible for retransmissions. This scheme results in acceptable load even for loss probabilities higher than 50%.

5.1.3 Discussion

When packet loss rate of wireless links is small, a two-level binary tree can be seen as a one level tree in which two leaves are sub one level tree. Thus, we can use the following formula to approximately calculate the retransmission load $E[R](L=2) = 3\frac{2\gamma+\gamma^2}{1-\gamma^2}$. Consequently, as network and group size increase, the retransmission load of source in a L level tree is $E[R](L) = (2^L-1)\frac{2\gamma+\gamma^2}{1-\gamma^2}$. This formula demonstrate that source retransmission load of ARQ is a function of multciast tree size and packet loss rate on links. In wired network, packet loss rate is relatively small and it is network size that plays a key role in retransmission overhead. That is why improvements address of maintaining the scalability of reliable multicasting to reduce retransmission in large scale networks [66]. Their common idea is to distribute retransmission task to nodes other than the sender through local recovery. According to the type of nodes which send recovery packets, the reliable protocols using local recovery technique can be further classified into receiver-assistance retransmission scheme and router-assistance retransmission scheme.

5.2 Current Reliable Multicasting protocol for MANET

The first reliable multicast protocols for MANET ([69]) try to make ARQ mechanism adapt to mobile multihop environment. They just use source to retransmit lost packets. While, in MANET, loss rate is relatively high due to wireless interface and node's mobility. Even in a small network, retransmission load might be important. For example, in a two-level binary tree where 6 links deliver packets to 4 receivers. As show in Figure5.5 if the packet loss rate on each link is 0.08, the retransmission rate of source (tree root) exceeds 0.5 in source-originated scheme. On the contrary, the receiver-assisted scheme can reduce the retransmission load of sources. If every node in the tree takes the retransmission responsibility to its direct downstream nodes, the retransmission load of source is always that in one level tree whatever the tree size is. Therefore, research efforts are made to extend receiver-assisted scheme ([72]) or router-assisted scheme ([70],[71]) to MANET to reduce retransmission load of source and total bandwidth consumption. The rest of this section gives a brief introduction some of these work.

5.2.1 Adaptive Protocol for Reliable Multicast in MANET (APRM)

In [69], the authors proposed an adaptive protocol for reliable multicast to a set of predefined group members against topology change, which we call APRM. A core based shared multicast tree is constructed to delivery messages reliably. In case of fragmentation due to node movement, a ”forward region” is introduced to glue together the fragmented tree and messages are flooded in this region. However, this protocol needs that each recipient sends feedback directly to the source and get recovery messages from the source. Neither receiver nor router assists in retransmission. Thus, this protocol uses sender-based retransmission scheme and is not efficient as the size of the multicast group or transmission failure increases.

5.2.2 Anonymous Gossip

Receiver-assistance retransmission schemes for Internet require router know the group receivers in its sub-network. As group membership or topology change, control message should be sent to update the information in routers. Anonymous Gossip [72] employs the receiver-assistance retransmission scheme for MANETs with efforts to reduce this kind of control overhead.

Each multicast receiver periodically generates a retransmission request, called gossip message, which lists lost packets. Then node forward this message to a node randomly chosen from its delivery structure neighbors. Upon receiving a gossip message, routers randomly selects another delivery structure neighbor as next hop and forwards the message. This procedure is repeated till gossip message arrives either at a group member. In this way a receiver establish connection with a randomly selected group member and tell this member about packets it has not received. The group member checks to see if it has the required packets and retransmits the found ones.

In their paper, the authors announce that anonymous gossip favors tree structure to prevent gossip message from reaching the same nodes twice. Consequently, the sub tree of a router can be seen as the sub-network in receiver-assistance retransmission scheme. If the selected next hop is a downstream node, it means that router executes local recovery. But the difference is that in case of locally recovery failure, router will not do another attempt to address other receivers in its sub tree or send the feedback to the source. If the selected group member has non or not all required packets, the initiator of the gossip request shall makes another attempt. Therefore, the performance greatly depends on the selected group member, which might generate significant overhead and recovery latency.

5.2.3 Family ACK Tree (FAT)

Router-assistance retransmission schemes are developed based on the assumption of stable network. If a router is no further the multicast relayor the stored packets become no use. While in MANETs, multicast delivery structure may change frequently due to node's mobility that results worse utilization of router's buffer and degrade the retransmission scheme to source-based case.

To overcome this shortcoming but also solve the scalability problem of source-based retransmission, Family ACK Tree (FAT) [70],[71] extends the router-assistance retransmission scheme to adapt MANET environment. In FAT, a tree, called family ACK tree, is constructed to assume the reliability multicasting. Each node on the tree know its parent, grand parent and children. Children confirms the reception of multicast packets to their parents. This protocol requires that nodes store temporary packet. In case of transmission failure, the parent looks for the packet in its cache and retransmits to the corresponding child. If a node decide to leave family ACK tree, it transfer its children to its parent and also the packets waiting for being acknowledged. On the other side, the children contact their grandparent for recovery packets if their parents disappear. However this protocol becomes inefficient in high mobility networks due to the difficulty of ACK tree maintenance.

5.3 Active Reliable Multicast Protocol with Intermediate node Support

5.3.1 System model

Other than the assumption mentioned in 3.1, we make the following further suppose for reliable multicast protocol designing. The reference assigned by source to multicast packets is consecutive so that receivers can detect losses primarily by reference gap in the data packets. During a multicast session, senders have all packets they sent and receivers have all packets they received. We consider a scenario where there are n sources and m receivers in the multicast group sharing the same multicast delivery structure.

5.3.2 ARMPIS Protocol Design Principle

The active reliable multicast protocol with intermediate node support (or ARMPIS in abbreviation) makes both receiver and router assist retransmission. Nodes caches multicast packet for retransmission. In case of the required packets do not in cache, routers look for them in their ”sub-network”, which is their neighborhood. In ARMPIS, intermediate nodes are group members, nodes which convey multicast traffic and the neighbors of these conveyors, in brief, all nodes that overhear multicast traffic. These nodes are active in the sense that they cache multicast packets and perform retransmission. When a multicast traffic conveyor forwards packets, the broadcast nature of the air interface permits its neighbors to overhear the packets. Thus, these neighbor nodes can help to cache data packets for future retransmission. For example, Figure 5.6 illustrates a simple MANET where source S sends packets to three receivers R_1, R_2, R_3. When $nodeA$ forwards multicast packets, its neighbor $nodeY$ can receive those packets at same time. Then $nodeY$ can store and participate retransmission if there is delivery failure to R_2 and R_3.

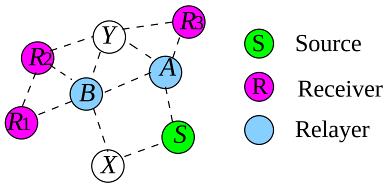


Figure 5.6: Multicast packet delivery

Intermediate nodes store packets with a certain probability (denoted by p) to realize distributed multicast data cache. There are some further reasons why we use such a probability.

- The memory capacity of mobile nodes is limited. If nodes store every data packet they receive, they can only keep the newest packets.

- It is unnecessary to store all packets. Simulation results ([23], [24] and 4.6) show that multicast routing protocol can deliver safely most of the traffic. Storing successfully delivered packet wastes memory capacity.

- Nodes mobility causes frequent changes in their roles. A node can be multicast traffic conveyor at a given moment and become a neighbor at the next moment, or is far away from the structure.

5.3.3 ARMPIS Protocol Description

ARMPIS is a receiver-initiated, NACK-based scheme in which receivers are responsible for detecting multicast packet losses and initiating retransmission request. This protocol contains two phases: data delivery phase and data repair phase. In the data delivery phase, when MRDC deliver data packets, intermediate nodes randomly cache these packets and fill in the duplication table of MRDC to avoid process duplications. In the data repair phase, nodes aggregate NACKs and try to get request packets locally. In case of local repair failure, nodes delete the information of request packets from MRDC's duplication table so that the node is ready for retransmission. Then NACKs is forwarded to the next hop along the reverse path to source. At last multicast routing protocol delivers the recovery packet. Each node in ARMPIS reserves a memory space as multicast packet caching buffer, which behaves in a FIFO fashion. ARMPIS defines two kinds of NACKs: local broadcast NACK which are sent to neighbors for local inquiry, and unicast NACK which are addressed to the request packet's source. A NACK message contains group identification, source identification and a reference list, each reference corresponds to a retransmission request. A NACK message can contains at most x requests for the same group and source pair. During data forwarding, a header is added in traffic packets in which there is a field to indicate that the packet is original or retransmitted. For each multicast flow, identified by $<$@group, @source$>$ pair, nodes aggregate NACKs using three sequence number arrays:

1. local repair array, which contains the sequence numbers that will be sent to neighbors;

2. request array, which contains the sequence numbers that have been sent to neighbors and will be sent to the next hop towards the source;

3. sent array, which contains the sequence numbers that have already been sent to the next hop towards the source.

These three arrays do not contain duplicate sequence numbers. Nodes delete the sequence number from these arrays when they receive the corresponding packet.

In the data delivery phase, the source assigns consecutive sequence numbers into data packets before sending them and MRDC use broadcast mode to deliver these packet to group receivers. Thus, no delivery guarantee is provided during transmission. When intermediate nodes receive a non-duplicate multicast packet, MRDC forward plan fills in duplication table of MRDC to avoid processing the same packet the second time. Then, MRDC forward plan passes the packet to ARMPIS. If the packet is original, nodes generate reverse path to the source by recording the node from which the packet comes. The path is stored in URTable of MRDC. Group receivers cache all multicast packet during the session. While, non-receiver nodes uses the following way to achieve cache received packets with probability p. Node asks a uniform distribution random value generator to generate a random number between 0 and 1. If the random number is smaller than p, node stores this packet.

In the data repair phase, receivers detect losses primarily by sequence gap in multicast packets. If finding such a gap, receiver waits for a short moment to make sure that the gap is not produced by disorderly delivery. If there are still some packets are not received after the timer out, the receiver considers these packets are lost and inserts their sequence numbers into local repair array. Each node periodically checks its local repair array. If the array is not empty, it initiates a local negative acknowledgment message (local broadcast NACK) for the first L sequence numbers and puts these L sequence numbers into request array. The local broadcast NACK message is sent to one hop away to see whether some neighbor has the lost packets. After waiting for a while, if a node still has some sequence numbers in its request array, it generates a unicast NACK message containing these sequence numbers and appends these sequence numbers to the sent array. The unicast NACK message is sent to the next hop on the reverse path to the source. When receiving such a NACK message, node checks whether it has some of request packets in its buffer because it is possible that this node did not receive the broadcast NACK one. Then, it deletes the sequence numbers which also appear in its three sequence number arrays and puts the rest sequence numbers into local repair array and erase the corresponding packet information from the MRDC's duplication table. In this way, this node is ready to forward the recovery packets. These steps are repeated till all requested packets are found or unicast NACK message reaches the source. Before transmission node marks in the packet header that this packet is a retransmitted one. Then, these requested packets are delivered by multicast routing protocol as a normal multicast packet and are forwarded only by the multicast routers which do not have the relevant packet information in their duplication table. In this way, retransmitted packets flow on the sub-tree where transmission failure occurred and will not go on the other part of the multicast tree. Intermediate nodes periodically delete the eldest L sequence numbers from their sent array. While receivers move these sequence numbers from their sent array to their local repair array. Thus the data repair phase continues in case of retransmission failure.

Let's take an example to see how ARMPIS works. Figure 5.7 shows a 30-node mobile ad hoc network. Nodes are differenced by their identification. If two nodes are in the coverage of each other, a dotted line connects them. In this network, there is a multicast group contains one sender, node 0, and four receivers, nodes 1, 2, 3, and 4. MRDC constructs a multicast tree to connect these group members. The tree is rooted at node 0 and contains 6 routers: nodes 8, 11, 15, 22, 25 and 28. Totally eight nodes are tree neighbor, they are node 5, 10, 12, 16, 18, 19, 23 and 27. If node 28 fails to transmit a multicast packet p_i to node 1, node 1 sends a broadcast NACK to its neighbors without knowing there is another group receiver node 4 in its neighborhood. If unfortunately, node 4 has neither p_i, node 1 can still expect to get the packet from node 28 or even node 12 if they chose to cache p_i. If none of these three nodes has packet p_i in their cache, node 1 then sends a unicast NACK to node 28. Node 28 in its turn inquires its neighbor for p_i. If none

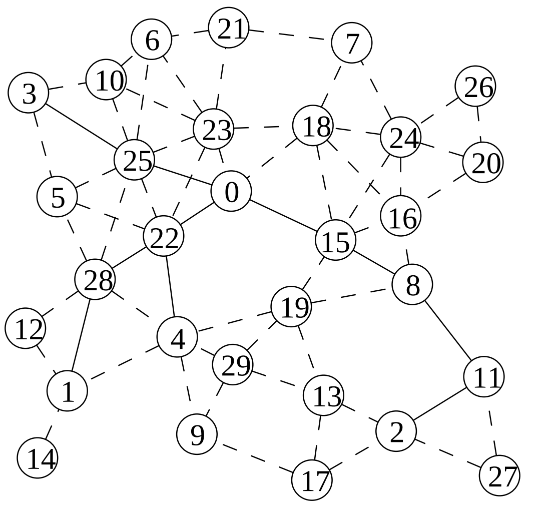


Figure 5.7: Multicast tree in a MANET

of its neighbors has p_i, node 28 removes packet information from duplication table and sends NACK to node 22. In this way, NACK is sent to source, node 0 if none intermediate nodes has p_i. Node 0 retransmits packet p_i. This packet flows on the branch of node 22 to node 1 and will not go to branches of node 15 and 25 since they consider it as duplication.

After several seconds, the network topology has been changed as a result of node's mobility as shown in Figure 5.8. The nodes' movement results in the path from node 0 to node 3 changes because the link between node 3 and 25 is broken. On the other side movement creates a shorter path from node 0 to node 2 owing to a new link established between node 0 and 19. MRDC reacts this topology changes by reconfigure the multicast tree which leads to nodes' role change. Old neighbors, nodes 19, 10 and 23, become router and old routers, node 15 and 25, become tree neighbor. However the multicast packets cached by these nodes are still available for local recovery.

5.4 Performance analysis

We evaluate the performance of our reliable multicasting protocol in terms of delivery guarantee and bandwidth consumption in ns2 .

5.4.1 Simulation Environment and Implementation Decision

The simulation environment is identical to that of Section 4.6. We use movement scenarios which contains more networks partitions. As for the traffic scenarios they

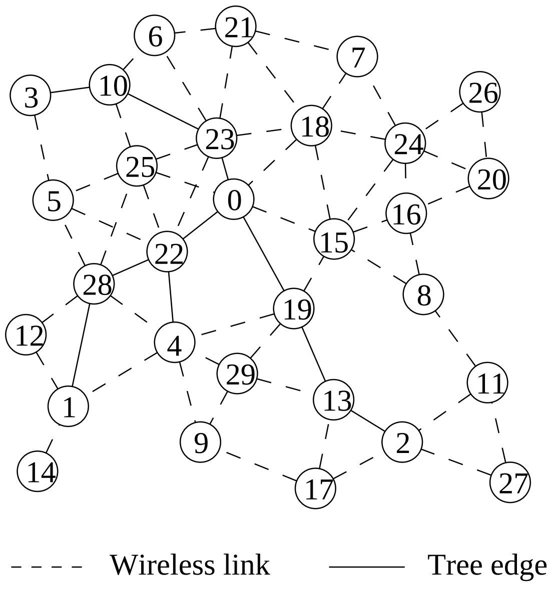


Figure 5.8: Multicast tree after topology change

are little different to those used in previous simulations. Instead of sending packets till the end of simulation, in this reliable experiments, each source transmitted 3200 packets during a simulation. With a speed of 4 packets per second, sources finish their transmission in 800 seconds. The rest simulation time (about 20 seconds) permits retransmissions finishes their work.

5.4.2 Protocols and parameters

The capacity of buffer is set to permit a node store 64 data packets. The time to check local repair array is every 1 second. Nodes wait for 1 second before sending a unicast NACK. And a sequence number rests in sent array for 3 seconds. So, the total time for NACK suppression is 5 seconds. The probability p of nodes storing a packet is 10%.

We studied the performance by varying three parameters: the probability p to choose a suitable cache probability, the maximum movement speed and the number of sources to test the performance. For comparison reason, we develop a reliable multicast protocol similar to [69] in which nodes do not cache packets and feedbacks are sent directly back to the source. This protocol is denoted as APRM in simulations.

Three metrics are used during the whole performance analysis:

- **Packet delivery ratio:** the percentage of data packets correctly delivered to receivers over the number of data packets that should have been received. The goal of our reliable multicast protocol is to provide 100% delivery ratio in most cases.

- **Total network load:** the total number of bytes sent during simulation, it includes both control messages and traffic packets.

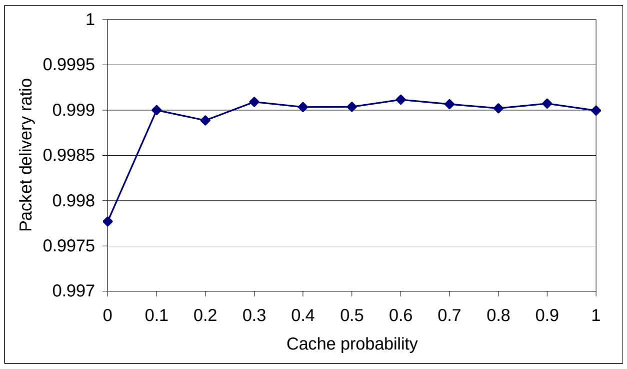
- **Source retransmission load:** the number of data packets retransmitted by sources. This metric counts the extra load of sources when providing transmission guarantee using ARQ technique.

The rest of this section presents the simulation results in detail. The simulation results with confidence can be found in Annex A.1.2

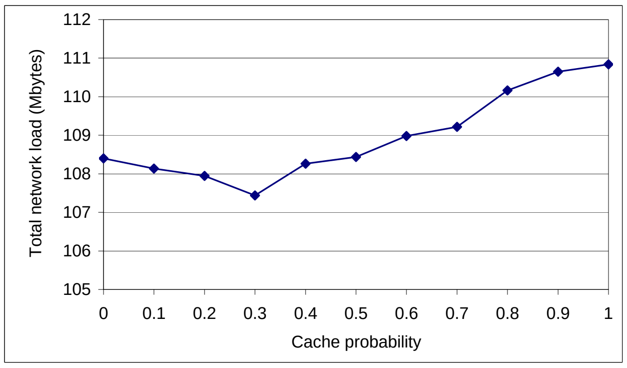
5.4.3 The choice of cache probability p

First, we set the number of source to 6 (three groups and two sources per group) and maximum movement speed to 5 m/s while vary the cache probability from 0 to 1 to see the behaviors of ARMPIS. When p equals to 1, nodes store all packets they overhear. This results to only the newest packets being stored in cache. On the contrary, when p is set to zero, nodes do not cache any packet.

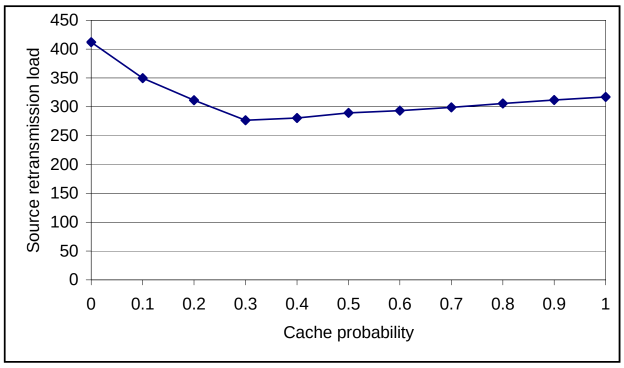
Figure5.9(a) shows that packet delivery ratio is improved when cache probability passes from 0 to 0.1 and then remains nearly stable. Thus, increase cache probability cannot enhance packet delivery ratio. ARMPIS cannot provide 100% delivery because there are some packets are not retransmitted before the simulation terminates. Therefore, the packet delivery ratio reflects how many packets are waiting for being retransmitted at the end of simulations. In our simulations, the lifetime of node's buffer, which means the possible eldest packet cached in nodes, covers $T = L/(p * r * s) = 10/(6 * 4 * p)$ last simulation seconds. If the original packet is generated during that time, the recovery packet might be found on the route to the source with a local recovery success probability of $p_s = 1 - (1 - p)^n$ if there are n neighbors. Otherwise, the request should be sent back to the source. A high probability leads to a high local repair success rate with the cost of shortening the lifetime of node's buffer. Consequently, a quick response may be given to the retransmission request of the latest packets, while a slow response to earliest packets. As we indicated in Chapter 4, three cases prevent MRDC to deliver multicast packet to the destinations. Routing protocol failure, lower layer protocol failure and network partition. Lower layers protocols failures are usually represented by MAC layer packet collision in simulations. Routing protocol failure means multicast protocol cannot react to topology changes in time or does not correctly constructed during tree refreshing due to control message loss. MAC layer packet collision and temporary tree fragmentation usually produce a short term packet loss and can be detected quickly, while network partition and tree fragmentations which persist during one or several periods may cause packet loss during a long period. When cache probability increases, ARMPIS becomes more and more efficient facing to transmission failure related to packet collision or temporary tree fragmentation by quickly recovering more packet loss , but less and less efficient facing to network



(a) Packet delivery ratio v.s. cache probability



(b) Total network load v.s. cache probability



(c) Source retransmission load v.s. cache probability

Figure 5.9: The performance behavior as a function of cache probability p

partition or long duration tree fragmentation by leaving more packets recovered by sources. As a result, the number of not being repaired packets keeps no change. 0 cache probability can be seen as an extreme cases of high probability where lifetime of buffers is zero and local recovery success rate is also zero. Thus, there is no recovery acceleration through router assistance. give the biggest non-repaired packet list and the worst packet delivery ratio.

Total network load as a function cache probability is illustrated in Figure 5.9(b)). For a give movement and traffic scenario, the difference of total network load is usually the result of retransmission overhead. The results show that total network load decreases firstly and then rises after reaches its minimum value along with the increase of cache probability. When nodes begins to cache packets, retransmission overhead is reduced through local recovery reduce. At the same time as cache probability increase, the distribution of multicast packets among neighbors becomes worse and the probability of neighbor nodes cache the same packets also increase. The probability of duplicate cache is $p_d = 1-(1-p)^n - np(1-p)^{(n-1)}$. When a node runs local recovery for a latest packet, it might get the same recovery packet from multiple neighbors. These duplications consequently increase retransmission overhead.

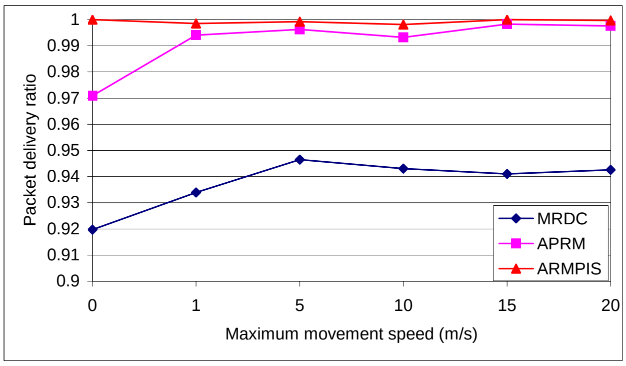
The same behaviors can be observed in source retransmission load as shown in Figure 5.9(b). The retransmission load of source fast decreases and then slightly increases after reaches the smallest value. The local recovery assisted by routers reduces the retransmission load of source, while the increase of cache probability makes local recovery concentrate more and to the latest packets and the sources as a result should deal with more and more earliest packets.

This simulation shows that ARMPIS gives the best compromise among packet delivery ratio, bandwidth consumption and source retransmission load when cache probability equals to 0.3. In the following simulations, we choose this value as cache probability.

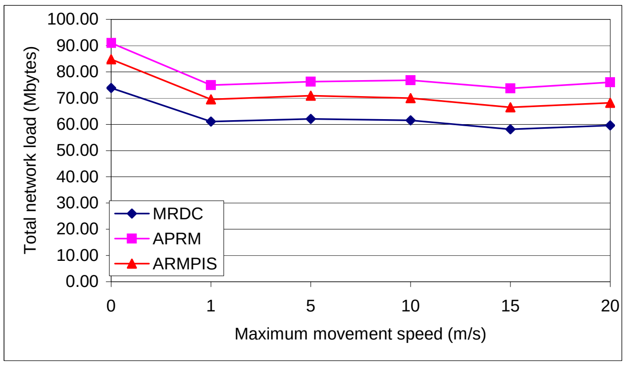
5.4.4 The impact of node mobility

In this aspect, the maximum movement speed of nodes range in the set {0, 1, 5, 10, 15, 20} m/s. A 4-source scenario is chosen as traffic pattern which defines two groups and two sources per group. Therefore, when reliable multicast protocol deals with topology changes, it should also face to slight inter group and intra group competition.

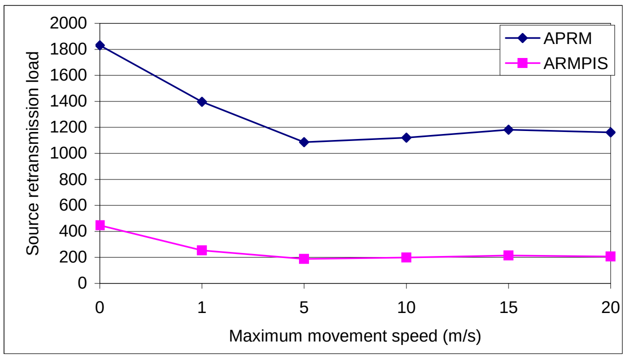
Figure5.10(a) shows the packet delivery ratio with different maximum speed of these three protocols. The results show that ARMPIS is reliable against frequent topology changes: mobility has nearly no impact on the performance of ARMPIS, which can provide almost 100% packet delivery ratio in all simulations, while the performance of underlying multicast routing protocol has significant changes. APRM gives a worse performance than ARMPIS. In APRM, only source can resend the lost packets, its worse than 0 probability of ARMPIS where there are still receivers assist to retransmit. Thereby, the recovery packets have the same loss



(a) Packet delivery ratio v.s. Maximum speed



(b) Total network load v.s. Maximum speed



(c) Source retransmission load v.s. Maximum speed

Figure 5.10: The impact of node's mobility

probability as the primary ones and provides a slow recovery speed. When MRDC provides a bad the packet delivery (for example in the stable networks), APRM leave a big list of packets not be recovered at the end of simulations. However, the local recovery mechanism helped by routers and receivers accelerates recovery procedure risk by proposing a shorter path for retransmission. Thus the packet delivery ratio is improved.

Figure5.10(b) demonstrates the total network load, which include control messages, original packets and retransmission packets, as a function of node's mobility. As topology changes becomes frequently, the routing overhead of MRDC increases to locally repair multicast trees. However, high node's mobility leads to a fair distribution of group members in the network. which reduces the size of MRDC multicast trees (see the analysis in Chapter 4). As a result, total network load of MRDC keeps stable. Depending on the performance of underlying multicasting protocol, ARMPIS generates less 20% extra bytes for retransmission while ARMPIS needs more than 30%.

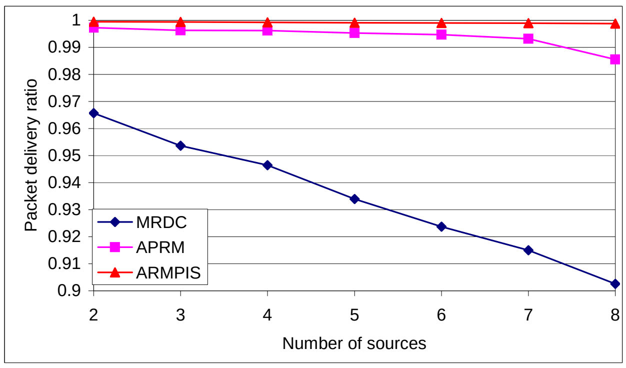
Figure5.10(c) illustrates the average number of packets retransmitted by sources. Sources in APRM should retransmit more packets when MRDC delivers less packets. ARMPIS makes sources retransmit five times less packets than APRM does. Compared with APRM, ARMPIS distributes retransmission responsibility and have less retransmission failures.

ARMPIS is reliable facing to topology changes and can deliver nearly 100% data packets in all mobility cases. This protocol is also scalable in the sense that it does not generates significant retransmission load as node's movement speed increases.

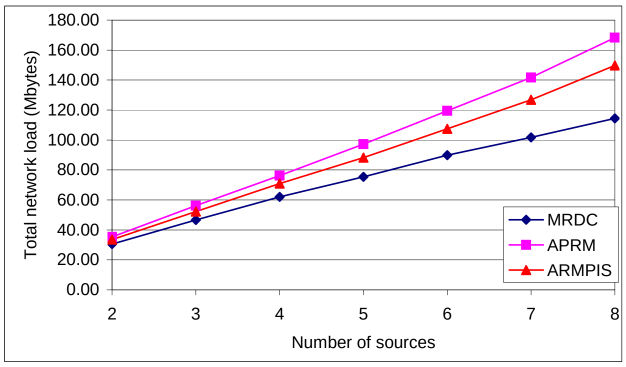
5.4.5 The impact of traffi c load

In traffic load experiment, node mobility speed is moderate with maximum speed at 5 m/s. The number of multicast sources increased from 2 to 8. The number of groups was consequently increased from 1 to 4. The total network load metric is not used in this simulation because low packet delivery ratio in high load network makes comparison unfair.

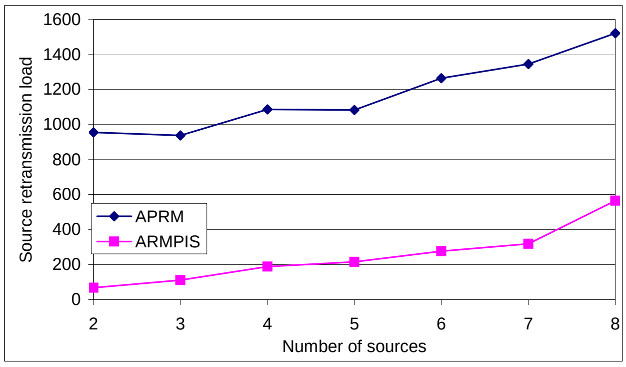
The packet delivery ratio as a function of the number of sources is presented in Figure5.11(a). ARMPIS maintains nearly 100% packet delivery ratio till seven sources and then appears a little degenerative. However, it can transfer more than 99% data packets to all receivers. This shows that this protocol is reliable when traffic augments. The performance of APRM exponentially degrades. MRDC has a linear degradation even when no congestion happens. This phenomenon is related to the data forwarding fashion employed by MRDC, which works on top of IEEE 802.11. This later does not offer delivery guarantee for broadcast and multicast packets. When MRDC forwards multicast packets, some packets are lost due to hidden terminal problem. And it becomes more and more serious when network load increases. In APRM, retransmission initiated by the original source adds considerable extra traffic to the network (see Figure5.11(b)), which raises



(a) Packet delivery ratio v.s. Number of sources



(b) Total network load v.s. Number of sources



(c) Source retransmission load v.s. Number of sources

Figure 5.11: The impact of traffic load

collision risk and introduces congestion. That's why the packet delivery ratio decreases more quickly after 7 sources. On the contrary, local recovery mechanism of ARMPIS tries to find the request packet as close as possible to the corresponding receivers. As a result, the retransmission load of ARMPIS is less important than that of APRM that makes ARMPIS outperform APRM. Since there is no retransmission congestion control, when traffic becomes heavy in the network, the performance of ARMPIS finely degrades.

As demonstrated in Figure5.11(c), the packets resent by sources in ARMPIS is much less than those in APRM. In the case of 8 sources, each source of APRM retransmits nearly the half number of primary packets while retransmission load of sources nearly no change. This phenomenon can be explained by the fact that wireless channel is saturated around sources which prevent them to receive further NACKs. These source do not consequently generate more retransmission load. It also explains why packet delivery ratio of APRM decreases so quickly from 7 sources to 8 sources while at the same time, the degeneration of MRDC is not so significant. On the contrary, in ARMPIS much more NACKs arrive at sources in the case of 8 sources than that of 7 sources. Then, the retransmission load of source is doubled.

Another reason of performance degradation of ARMPIS is due to the life time of node's buffer which is inverse proportional to the traffic load. Recall that the life time of node's buffer is $L/(p * r * s)$, where p is the cache probability, r is the transmission rate of source and s is the number of sources. The life time of buffer decreases as the number of sources increases. In a 8-sources scenarios, the buffer of nodes in hot spot can only covers about 1 second transmission (L = 10 packets, r = 4 packets/source/second and s=8 sources). With a so small life time (which is nearly equal to local repair array check period), router's buffer does nearly not help packet caching and retransmission and there are only receivers assist. As a result, both source retransmission load and network load increases, which consequently degrade slightly packet delivery ratio.

However, low recovery overhead generated by ARMPIS make this reliable protocol scale better than sender-based retransmission scheme as network load increases.

5.5 Conclusions and Future Work

In this chapter, we introduced our active reliable multicast routing protocol with intermediate node support (ARMPIS) to provide reliable multicasting in mobile ad hoc network. The retransmission load of ARQ is a function of both network size and transmission failure rate on each link. The properties of MANET such as frequent topology changes and wireless interface make multicast packets transmission easily fail. In order to reduce source's retransmission load and achieve scalability in high lossy wireless environment, ARMPIS extends the receiver and router assistance retransmission scheme to MANET by distributing retransmission burden to intermediate nodes. When receiving a retransmission request, nodes first check their buffer and also those of their neighbors in order to do a local repair before forwarding the request to the source. A cache probability is employed to decide store or not a message in each node to reduce packet cache duplication and stores as many as possible multicast packets among neighbors.

A high cache probability improves local recovery success rate of favors of the latest packets but reduces the lifetime of router's buffer and degrades multicast packet storage distribution among neighbors. The performance evaluations suggest 0.3 as cache probability to achieve an optimal compromise. The simulation results also show that ARMPIS is reliable in both stable and dynamic networks or in a relative high load situations by providing nearly 100% packet delivery to all receivers. And thanks to local recovery scheme, ARMPIS reduces significantly both network load and source's retransmission load compared to APRM, a source-based retransmission scheme.

Up to now, ARMPIS focus on retransmission scheme by extending both receiver assistant and router assistant retransmission scheme to MANET. However the congestion control is also a key issue in providing reliable multicasting. The future work intends to introduce flow control in reliable multicasting to avoid retransmission degrade the throughput of the networks.

Chapter 6

AD HOC NETWORK TESTBED

Software simulations are an excellent choice for the initial design and estimation of results. They are cheap and offer a developing background similar to real cases. When implementing MRDC in ns-2, we acquired the first experience in designing multicast routing protocol. In Section 6.3.2, we did a lot of trials to simulate the performance of MRDC in 50 node networks by using a software network simulator. Through those experiments, we chose the optimal parameters for MRDC, understood its behaviors in different mobility and traffic patterns. Furthermore, since the experiment of software simulations can be repeated, software simulations provide a standard way that allows us to directly compare MRDC's performance with some other multicasting protocols. As a results, we can propose the suitable applications and working environment of MRDC. But software simulator has many limitations. First, they do not realistically duplicate the physical layer. Second, a software simulation cannot catch subtle bugs seen in interactions between the operating system, the system hardware, and the real-life design environment. A software simulator also ignores interlayer communication, which is integral to the effectiveness of these protocols. The work of [77] demonstrate a number of significant discrepancies between simulated and hardware results. These limitations make hardware testing essential. In this chapter, we introduce our experience in implementation and validation both unicast routing protocol and multicast routing protocol in an ad hoc testbed. Our goal is not limited in routing protocol performance evaluation. We believe that an ad hoc testbed with suitable routing protocol is very useful for other layer protocol study but also for design new ad hoc applications.

Linux [78] is chosen as operating system in our testbed for its availability and familiarity. Both unicast routing protocol and multicast routing protocol are decided to run in user space to facility cross platform implementation and installation. However a slight difference exist between the architectures used in routing protocol implementation due to the kernel level forwarding support for unicast and multicast packets. Unicast routing protocol is developed as a routing daemon which runs in user space and maintains kernel level routing table via system call [79]. The following issues make multicast routing cannot use the same architecture as unicast routing. Firstly, not all Linux kernels support multicast packet forwarding. Secondly, those kernels, which support multicast packet forwarding, do not allow single device forwarding. That is to say kernel will not send packet to the same interface where it receives the packet to prevent forming transmission loop. However, in MANET, multicast forwarding happens on the same wireless interface. To overcome these limitation, a procedure of packet capture encapsulation, forwarding, decapsulation and delivery is used when we implement multicast routing protocol. This procedure permits multicast routing protocol to forward packets in user space.

We choose Distributed Dynamic Routing algorithm (DDR) [80] as unicast routing protocol. DDR is a distributed clustering algorithm with deterministic criteria, which deals with the problem of topology management in mobile ad hoc networks. The main idea of the algorithm is to select for each node a neighbor, called *preferred neighbor*, that has a maximum degree of connectivity in the neighborhood (i.e. criteria of election algorithm). This is done using only periodical beaconing process. It has been proved that whatever the network topology is, connecting each node to its preferred neighbor always yield to a forest [80]. In this algorithm, each tree of the forest forms a zone, and each zone is maintained pro-actively. Zones are connected to each other via the nodes that are not in the same zone but are in the direct transmission range of each other. Therefore, the network is partitioned into a set of non-overlapping zones. As a result, the algorithm combines two notions: forest and zone. Forest reduces the broadcasting overhead by selecting a subset of the set of neighboring nodes for forwarding a packet, and zones are used to reduce the delay due to routing process and to reach high scalability. DDR proactive part is validated under different network topology and movement scenario without traffic.

On the other side, MRDC is implemented as multicast routing protocol in the testbed. With additional function modules such as a simplified IGMP [6] module and a tree information collection module, we evaluated the bandwidth utilization of MRDC and tested the correctness of the implementation in both topology dynamic scenarios and membership dynamic scenarios using a popular multicast application.

The following sections describe these works in detail.

6.1 Implementation Structure

Before developing routing protocol program, we should firstly decide where to locate the program and how it cooperates with other system components. In this section, we discuss the architecture of Ad hoc testbed and our implementation structure choice in Linux system.

6.1.1 Architecture of Ad hoc testbed

An ad hoc testbed, should consists of at least following components as shown in Figure 6.1 according to TCP/IP network model [81]:

- Application layer, where end-user applications reside to send and receive messages.
- Transport layer, which handles communication among programs on a network. It also provides some further functionalities such as reliable transmission, quality of service supporting, flow control, and so on to support application's requirement. TCP and UDP falls within this layer.
- Network layer, which is used for basic commqunication, addressing and routing. It directs data packets from the sender to the receiver(s).
- Link layer, which defines the network hardware and device drivers. Usually MAC protocols are integrated in device drivers.

According to this structure, routing protocols should locate between transport layer protocols and link layer protocols to route packets among difference devices.

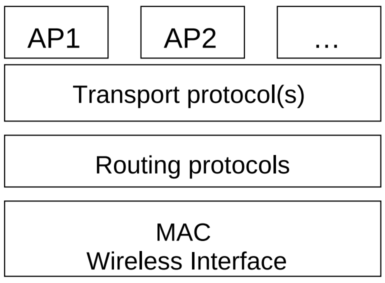


Figure 6.1: System components of an ad hoc testbed

In current Linux implementation, socket layer interface implemented in the Linux [82] provides standard API which allows user space programs to open communication endpoint to remote devices. The implementations of transport layer protocols (TCP and UDP) are hidden in the socket layer. Because of this reason, some researches developed their routing protocol in kernel to achieve efficiency. For example, Mornach project team of Carnegie Mellon University developed Dynamic Source Routing (DSR) [46] into network stack [83]. University of Maryland also developed an ad hoc network testbed on Linux by adding a Forwarding Engine (FE) into the kernel [84]. Figure 6.2 concludes this architecture. This strategy require the developing experience but also increase the difficulty of implementation and installation in different Linux since it should modify operating system kernel. This approach is a suitable approach for the final version.

Some researches prefer not to modify the existing Linux kernel codes. Modules stay in user space as much as possible. The testbed is implemented as a Linux

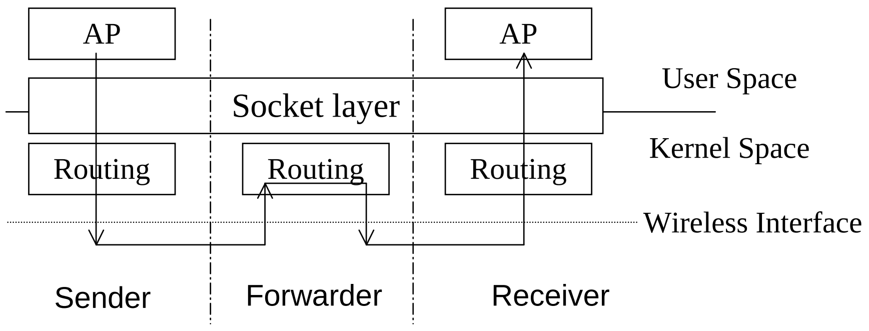


Figure 6.2: Kernel modified routing architecture

add-on rather than a Linux kernel patch. This approach offers the portability and flexibility needed for an experimental system. They usually benefit the kernel level IPv4 forwarding support built into the Linux operating system to implement routing protocol. IP forwarding modular provides the basic network layer packet routing. This modular analyzes the packet's header (destination address and TTL and sometimes source address), and then sends the packet to the corresponding device or drops the packet to the ground according to the routing table. The Linux kernel forwards packets by performing the following procedures. The network interfaces accept and send all packets to kernel. The kernel accepts all packets, checks the destination address with the kernel level routing table and decides whether to forwarding them. (A message with the destination not specified in the routing table is forwarded to the default gateway. If these is no viable forwarding location, the packet is dropped and ICMP [85] destination error message is sent.) Using this forwarding support, these researches implemented routing protocol as a user level routing daemon which updates and maintains the kernel level routing table. Figure 6.3 illustrates this idea. We chose this structure to implement unicast routing protocol since it on one side reduce developing charge and on the other side keep delivery efficiency by using kernel level forwarding. However, this structure meets difficulty when be used to implement multicast routing protocol since most Linux operating systems do not support kernel level multicast forwarding through the single device .

Unicast routing protocol uses the kernel level IPv4 forwarding support built into the Linux operating system. Unlike and [86] which modified kernel, our routing implementation is supposed to run in the user space. This strategy avoids modifying operate system and therefore reduces not only the requirement of developing experience but also the difficulty of installation.

However, multicast routing protocol implementation cannot use this architecture directly. Firstly, not all Linux kernels support multicast packet forwarding. Secondly, those kernels, which support multicast packet forwarding, do not allow single device forwarding. That is to say kernel will not sends packet to the same interface where it receives the packet to prevent forming transmission loop. How-

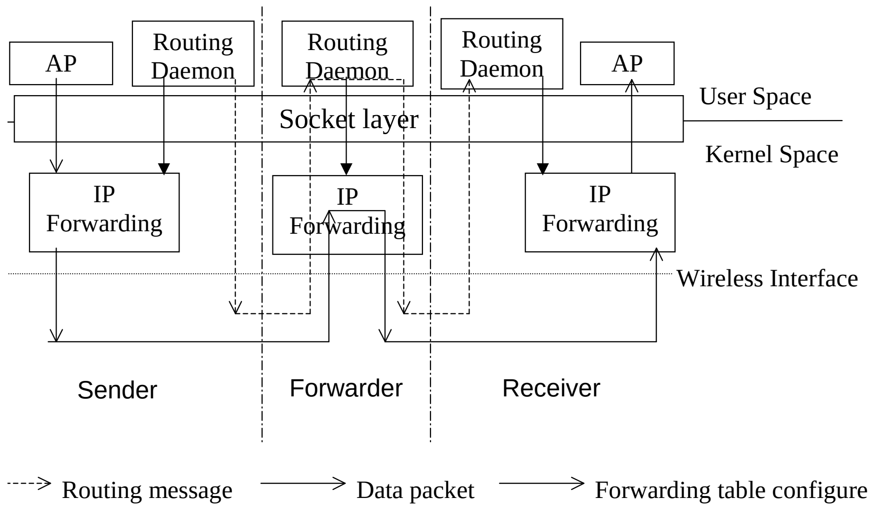


Figure 6.3: User space routing daemon architecture

ever, in MANET, multicast forwarding happens on the same wireless interface. To overcome these limitation, a structure illustrated in Figure 6.4 is used to implement multicast routing protocol. In this structure, when an application sends a multicast packet, the Linux kernel instead of sending it directly to wireless interface, give this packet to routing agent running in user space. Then routing agent encapsulates the packet and transmits it hop by hop in a suitable way (broadcast or unicast) till destination routing agent. Finally, the destination routing agent decapsulates the packet deliver this packet to local application. This procedure permits multicast routing protocol to forward packets in user space.

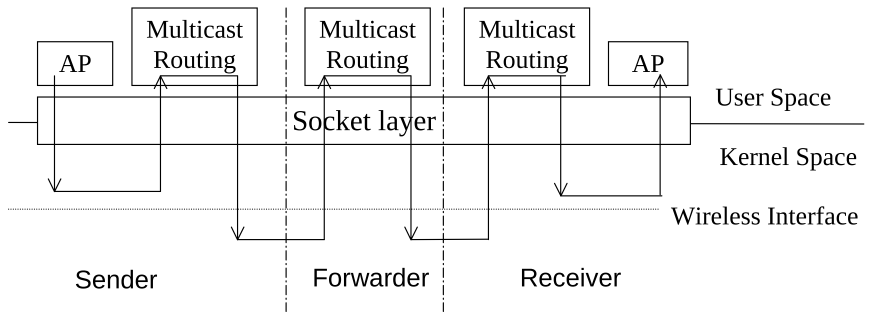


Figure 6.4: User space multicast routing architecture

6.2 DDR Implementation and Validation

In this section, we first give an overview of Distributed Dynamic Routing algorithm (DDR) and then introduce how we implemented this protocol and validated the implementation.

6.2.1 An Overview of DDR

DDR stands for distributed dynamic routing algorithm, which deals with the problem of topology management and contributes in local routing process through clustering in mobile ad hoc networks [80]. To do so, the algorithm selects for each node a neighbor, called *preferred neighbor*, that has a maximum degree of connectivity in the neighborhood based on the information provided by periodical beaconing exchanged *only* between a node and its neighboring nodes. During the beaconing process, each node gathers the information describing its neighborhood in its neighboring table. This table enables each node to elect their preferred neighbors. The link between a node and its preferred neighbor is then become a preferred link. The set of preferred links in the neighborhood forms a set of preferred path in the network, which will be used during the routing process. It has been proven that whatever the network topology connecting each node to its preferred neighbor yields to a forest (i.e. no cycle) [80]. Such a forest indeed reduces the broadcasting overhead by selecting a subset of neighboring nodes for forwarding a packet. Finally, each tree of the forest forms a zone, and is maintained proavtively. These zones were constructed in order to reduce the delay due to routing process and to reach high scalability. As a result, the algorithm extends the network topology from node level to to zone level (i.e. zone abstraction) by partitioning the network into a set of proactive zones.

In addition to the neighboring table, which maintains the node identifier (NID) and node degree (Deg) about the nodes within the transmission range, the algorithm builds two extra local tables, namely: intra-zone, and inter-zone. Intra-zone table is the table through which a node detects the structure and changes to the tree it belongs to. This table consists of two critical information: direct preferred neighbor (PN) and the neighboring preferred neighbor(s) learned by the direct preferred neighbor (learned_PNs). In order to construct this table, each node sends a beacon indicating its preferred neighbor, or if this latter remains unchanged a beacon is sent to indicate the learned_PN(s) of the preferred neighbor. Upon receiving such beacons, a node can gather information describing the tree members and the way to reach them. Such information is considered valid for a limited period of time, and must be refreshed periodically to remain valid. Expired information is purged from the table. This table is used to reduce the rebroadcasting overhead to minimal set of preferred neighbors (in graph theory, this set provides a heuristics to the problem of finding the minimum connected dominating set; MCDS), as well as to provide the local routing information to the routing protocol. Inter-zone table, on the other hand, keeps the information about the connectivity with the neighboring zones of the zone to which the node belongs. This table provides the gateways for the routing protocols to leave the zone.

6.2.2 DDR Software Architecture

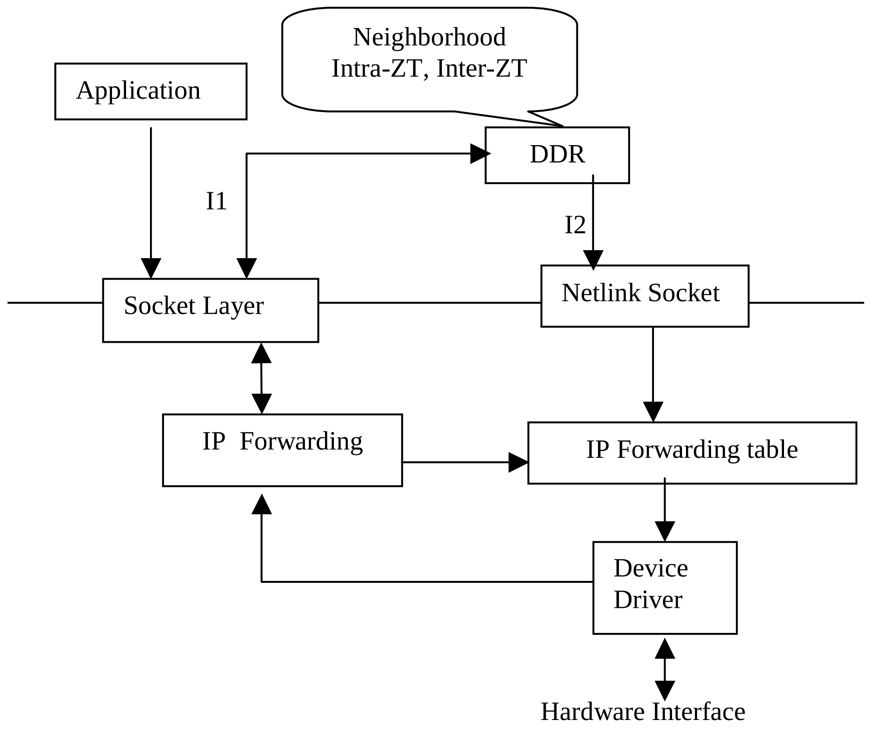


Figure 6.5: DDR implementation structure

This structure of DDR implementation is shown in Figure 6.5. DDR implementation opens two interfaces for communication. Interface I1 is a UDP socket for sending and receiving beacon and interface I2 is utilized to modify IP forwarding table through a netlink socket. Based on the information provided by beacons, DDR establishes neighbor table and selects preferred node. Then, DDR constructs intra-zone table (intra_ZT) and inter-zone table (inter_ZT) to define spanning tree. According to the intra-zone table, DDR sets its routing table and also modifies IP forwarding table. In this way, we implemented proactive unicast packet routing by using intra-zone table of DDR.

The flow chart in Annex A.2.1 gives a closer view of DDR's functionality implementation.

6.2.3 Validation of DDR Implementation

To validate the DDR implementation, we used four portable PCs. The operating system is Linux kernel version 2.4.18 provided by Red Hat 7.3. All portable PC equipped with IEEE802.11 wireless network card. These cards configured in ad-hoc mode, operated on 2.4 GHz bandwidth and communicated at the capacity of 2

Mb/s with transmission power of 1mW. The IP address configuration of these four PCs satisfies the relationship: $@Radio1 < @Radio2 < @Radio3 < @Radio4$.

Because we tested the functionalities of DDR such as prefer node selection, intra-zone clustering in this step, there is no need of traffic during the tests. We observed the routing tables constructed by DDR according to network topology.

We first placed these four portable PCs to construct an ad hoc network as shown in figure 6.6. Each node is in the transmission range of its direct neighbors. We observed the tables in four portable PCs are illustrated in Tables 6.1, 6.2 and 6.3. The inter-zone table of all four nodes are empty. In this scenario, the degree of Radio1 and Radio4 is 1 and that of Radio2 and Radio3 is 2. Radio1 chose Radio2 as prefer node, Radio2 chose Radio3, Radio3 chose Radio2 and Radio4 chose Radio3. These four nodes forms a DDR zone. As a result, their Inter-zone tables are empty.

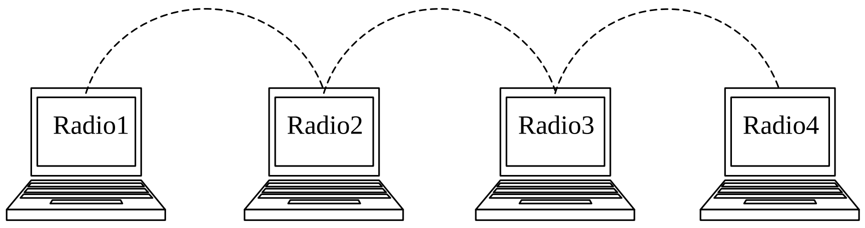


Figure 6.6: DDR validation: line scenario

Table 6.1: Neighbor tables

Radio2

(a) Radio1

Radio1
Radio3

(b) Radio2

Radio2
Radio4

(c) Radio3

Radio3

(d) Radio4

Then, we moved Radio4 to coverage range of all other nodes (Radio1, Radio2 and Radio3) as shown in Figure 6.7. We obtained routing tables of these four nodes as shown in Tables 6.4, 6.5 and 6.6 and inter-zone tables are not listed since they are blank. In network topology, the degree of Radio1 and Radio3 is 2 and that of Radio2 and Radio4 becomes 3. When the maximum degree corresponds to more than one neighbor, node selects the neighbor which has the biggest IP address as prefer node. According to this rule, Radio1 and Radio3 chose Radio4 as prefer node since $@Radio4 > @Radio2$, while Radio2 and Radio4 chose each other as prefer node. This scenario does not create the second zone either. Consequently the Inter-zone tables of nodes are empty.

These tests validate our DDR implementation and show that DDR can be directly used to route unicast packets in small networks. When the network size

Table 6.2: Intra-zone tables

Radio2	Radio3, Radio4

(a) Radio1

Radio1	
Radio3	Radio4

(b) Radio2

Radio2	Radio1
Radio4	

(c) Radio3

Radio3	Radio1, Radio2

(d) Radio4

Table 6.3: IP forwarding tables

Dest.	GW.
Radio2	Radio2
Radio3	Radio2
Radio4	Radio2

(a) Radio1

Dest.	GW.
Radio1	Radio1
Radio3	Radio3
Radio4	Radio3

(b) Radio2

Dest.	GW.
Radio1	Radio2
Radio2	Radio2
Radio4	Radio4

(c) Radio3

Dest.	GW.
Radio1	Radio3
Radio2	Radio3
Radio3	Radio3

(d) Radio4

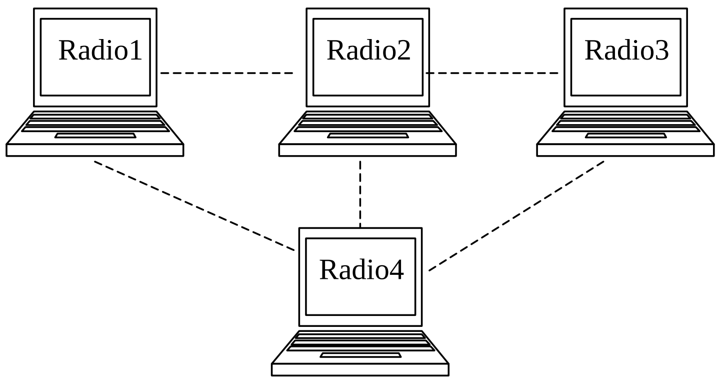


Figure 6.7: DDR validation: star scenario

Table 6.4: Neighbor tables

Radio2
Radio4

(a) Radio1

Radio1
Radio3
Radio4

(b) Radio2

Radio2
Radio4

(c) Radio3

Radio1
Radio2
Radio3

(d) Radio4

Table 6.5: Intra-zone tables

Radio4	Radio2, Radio3

(a) Radio1

Radio4	Radio1, Radio3

(b) Radio2

Radio4	Radio1, Radio2

(c) Radio3

Radio1	
Radio2	
Radio3	

(d) Radio4

Table 6.6: IP forwarding tables

Dest.	GW.
Radio2	Radio4
Radio3	Radio4
Radio4	Radio4

(a) Radio1

Dest.	GW.
Radio1	Radio4
Radio3	Radio4
Radio4	Radio4

(b) Radio2

Dest.	GW.
Radio1	Radio4
Radio2	Radio4
Radio4	Radio4

(c) Radio3

Dest.	GW.
Radio1	Radio1
Radio2	Radio2
Radio3	Radio3

(d) Radio4

increases, DDR will create more than one zone. In this case, a routing protocol (e.g. HARP [87]) is needed to explore route for nodes belong different zones.

6.3 MRDC Implementation and Validation

6.3.1 MRDC Software Architecture

According to the implementation structure (see Figure 6.4), MRDC is designed to route packets in user space to simplify installation and test in different systems. The implementation architecture of MRDC is shown in Figure 6.8. Besides tables defined in MRDC in chapter 3, this implementation contains a group table which records all multicast groups of which this node is a receiver and/or source. Because this implementation is aimed to demonstrate how to support multicast applications, other than MRDC core module, IGMP module and tree monitoring module are introduced. MRDC core module is further divided into two parts: Routing Part (RP) which, as describe in 3, constructs and maintains multicast tree on demand and updates multicast routing table, and Multicast Forwarding Part (MFP) which forwards multicast datagram according to the multicast routing table. IGMP module performances as simplified router to host part of Internet Group Management Protocol (IGMP) version 2 [6]. It detects membership changes on the local host and modifies group table. Group table can also be updated by MFP and provide group membership states to MRDC core module. Tree monitoring module is an additional functionality. It collects multicast routing information of a pre-defined group from multicast routing table in each multicast tree member. Then a tree monitor written in JAVA runs in a machine replays the multicast tree structure based on these information. MRDC opens three sockets for inside and outside communication. IGMP module owns an IGMP socket (named igmp_socket) to send and receive igmp packets. MRDC core module opens a UDP socket (called udp_socket) for inter-node message exchange and a raw socket (denoted as raw_socket) to deliver multicast datagram to local application. Tree monitoring module shares udp_socket for multicast routing information collection. This architecture forms a complete multicast detection, creation, delivery and tree monitoring flow for demonstration. The rest of this section explains these modules in detail.

Tables

The MRDC implementation possesses four tables: Group table, Duplication table, Unicast routing table (URTable) and Multicast routing table (MRTable). The three later tables are identical to those in chapter 3. Group table stores multicast group membership of local host.

As shown in table 6.3.1, a group table holds three fields: group ID number (GID) and two membership state fields SENDER and RECEIVER. GID represents the ID number of the group that this node is a sender and/or receiver. The ID number could be an address of class D in Internet. Group table is modified by IGMP

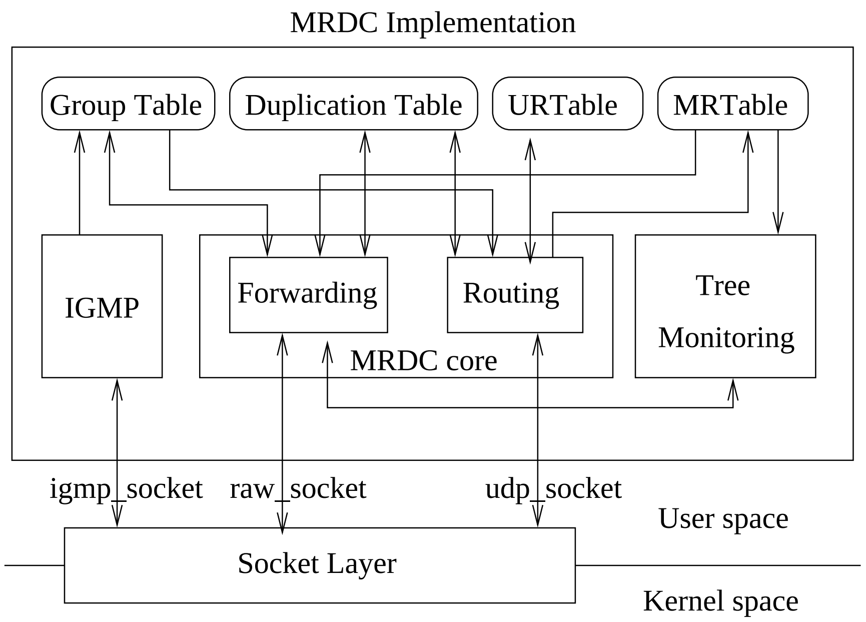


Figure 6.8: MRDC implementation structure

module and Forwarding module. When MFP detects that node is sending multicast packet to the group GID, it sets the SENDER field of that entry. On the other hand, if an application registers to receive packet of multicast group GID, IGMP module will receive group join message. Then this module sets the RECEIVER filed of the corresponding entry. If the membership is not confirmed before a time out or node recognizes a membership change (e.x. IGMP module receives a group leave message), the relevant field is unset. An entry is removed from group table if both membership state fields are unset.

Table 6.7: Group Table

GID	SENDER	RECEIVER

Routing Part

A multicast tree creation starts when multicast forwarding part sniffs a packet addressed to a multicast group which does not exist in multicast routing table.

In Routing Part (RP), MRDC creates and maintains multicast trees on demand. A multicast tree is created when the first sender sends the first multicast packet. (We elaborate on how to detect the first sender when we explain multicast forwarding part.) This node becomes the core, and routing part starts to create a multicast tree. The routing information is stored in multicast routing table. After tree construction, multicast forwarding part begins to deliver datagram. Nodes can join or leave the multicast group at any time during the session. The core maintains the multicast session by refreshing the tree periodically, but if the source status is timeout which means no multicast datagram is sent during a period of time, the core stops this maintenance and a multicast tree is automatically erased by deleting routing information from multicast routing table. In addition, this implementation supports core immigration. Each multicast source supposes itself is core and begins to broadcast its own CA message when multicast tree is erased. When receiving multiple CA messages addressing same multicast group, nodes compare core IP address and choose the biggest one. The source which has the biggest IP address thus becomes core comptition winner and continue to send its CA message periodically. While the other sources stop sending their CA message and join multicast tree as a normal group member.

The flow chart in Annex A.2.2 explains more detail how MRDC functions in this testbed.

Multicast Forwarding Part

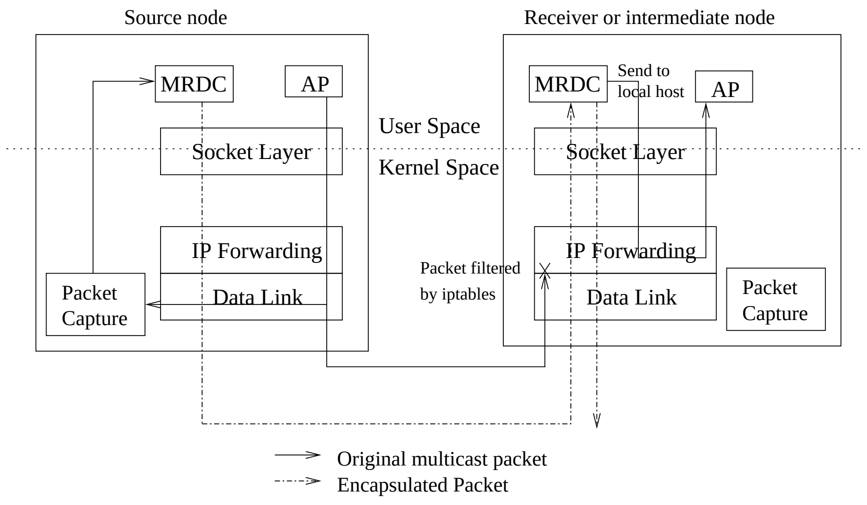


Figure 6.9: Multicast packet forwarding in MRDC implementation

We met several difficulties during MRDC implementation. i) How to detect multicast datagram. MRDC cannot get multicast datagram directly since it runs in user space. ii) How to forward multicast datagram. In table-driven unicast case, routing agent can run in user space and modifies routing table in kernel. IP forwarding module in kernel forwards unicast datagram according to the routing table. MRDC cannot use the same approach because multicasting in MANETs and multicasting in fixed Internet is different. The routing table's multicast tree states consist of local interfaces instead of neighbor identities and , the verification for incoming data is also done on incoming interface rather the sender. However, one MANET node can use the same interface talking to any neighbor on the same wireless channel. The incoming physical interface verification done by the IP kernel is no longer applicable. iii) How to detect duplications. Duplication cannot be avoided since nodes use the same wireless channel to communicate.

In Multicast Forwarding Part (MFP), we introduce the combination of packet capture, packet encapsulation/decapsulation and packet filtering, to solve the above problems. Figure 6.9 illustrates this procedure. A multicast datagram sent by an application is sniffed by packet capture at data link layer. Packet capture passes the packet to MRDC. MRDC encapsulates the captured packet into a MRDC data packet and then broadcasts it to neighbors through upd_socket. Nodes relay MRDC data packet till group receivers according to MRDC's multicast routing table. At last, MRDC in the side of receiver extracts the datagram from MRDC data packet and sends it to local application through raw_socket. To detect duplication, MRDC uses the informations stored in MRDC data packet header. On the other hand, if a group receiver is in coverage region of a group source, it receives multicast datagram directly from source. In order to avoid this phenomenon, the packet filtering (iptables installed in Linux) is applied to each node. We will explain in details how they work.

The basic technique used to capture multicast traffic packets in user space is the same as Unix tcpdump[88]. Since all machines use Linux as operating system, we use libpcap facility. It listens traffic at data link layer and sniff packets which are in accordance with a pre-defined rule set. For example, in this implementation, we want to capture all multicast datagram sent by a node itself. If the ip address of node A is 192.168.25.197, the following rule is set:
(tcp or udp) and ip multicast and src host 192.168.25.197
This rule captures all tcp and upd multicast packets sent by node A. Then libpcap facility passes the captured packets to MFP as a raw packet. MFP checks the destination address in the ip header of captured packet by consulting multicast routing table. If the destination address concerns a new multicast group, MFP sets up group source state in group table, caches the packet and informs routing part to create multicast tree. Otherwise, if the destination address concerns an existing group, MFP refreshes group source state and encapsulates the captured multicast datagram into a MRDC data packet and broadcasts it through udp_socket. A MRDC data packet header, as show in Figure 6.10, has two fields, Type and Reference. Type field distinguish a MRDC data packet from other mrdc control message and data packets are passed to MFP. Reference field stores the sequence number which is assigned by the source in order to detect packet duplication.

When receiving a MRDC data packet, MFP opens the IP header and UDP header of the payload packet to obtain source address and source port number. Then MFP passes these information combined with reference number in MRDC data header to duplication table, for duplication detection. If it concerns a packet received before, MFR drops it. Otherwise, duplication table stores packet informa-

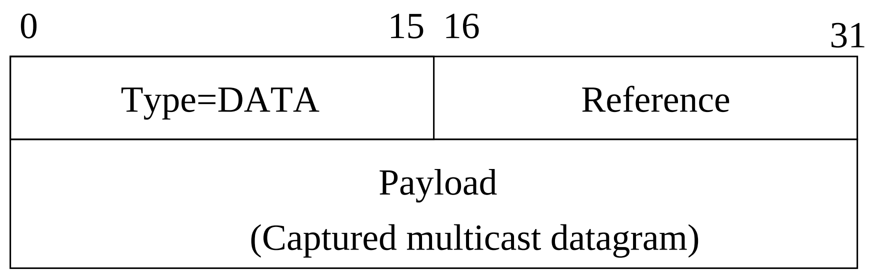


Figure 6.10: MRDC packet structure

tion and MFP broadcasts the MRDC data packet to its neighbors if it is a multicast tree member according to MRTable. At the same time if the group table indicates that local host is a receiver of the corresponding multicast group, MFP decapsulates the MRDC data packet and transfers the multicast datagram through raw_socket.

An important point that we should take into account is the broadcast characteristic of wireless link. An application may receive the same multicast datagram which has been already received if the node is in the coverage range of the source. Therefore it is necessary to prepare the packet duplication avoidance mechanism in the implementation. To filter the packet from source,we use the Netfilter/iptables facility [89]. It sits in-between the kernel IP stack and network device drivers and manipulates every packet in or out of this host according to pre-defined rules. Rules can be set or changed at any time through a command interface. For example, if the wireless interface is eth0, the following rule is set:
iptables -A INPUT -d 224.0.0.0/16 -p udp -i eth0
This rule drops all udp multicast packets coming from eth0.

IGMP Module

IGMP Module periodically sends igmp membership query message through igmp_socket and listens at igmp_socket. It sets up group receiver state in group table if receiving a membership report message. On the contrary, it unsets group receiver state if capturing a igmp leave group message.

On the other hand, group table periodically unsets the states which are not updated during last period.

Tree Monitoring Module

The node whose IP address is coincidence with the predefined monitor address is topology monitor. Monitor consulting the multicast routing table to check whether a multicast tree exists for the pre-defined group. If it is the case, monitor broadcasts a message to the rest of the network. This message is made up of monitored group, monitor address, sequence number and last hop fields, When this message propagates in the network, a reverse path to the monitor is constructed. All members of the corresponding multicast tree sends the information including the IP address of their upstream and downstreams to the monitor through this reverse path. This procedure is executed periodically.

Timers

We selected five seconds for multicast tree refresh interval. IGMP queries membership every eight seconds query and membership timeout was set to eighteen seconds. Tree monitoring collects tree information every second.

6.3.2 Validation of MRDC Implementation

Validate MRDC implementation in a real ad hoc testbed.

Testbed confi guration and (Implementation) Platform

Some issues, such as the expensive cost of hardware investment and the difficulty to organize mobile tests, discourage researcher from the construction of an ad hoc testbed. Our ad hoc testbed comprises portable personal computers (portable PC) and PDAs. On one hand, portable PC are stable and powerful. We use them to generate video stream and show multicast tree. On the other hand, PDAs are cheaper and lighter than portable PCs. This network configuration reduces hardware cost and at the same time facilitates mobility testing.

Ad hoc network nodes in our testbed are Intel Pentium III based Dell C600 laptops and Intel StrongARM based compaq iPAQ H3850s equipped with IEEE802.11 wireless network card.

MRDC was developed on Linux kernel version 2.4.18 provided by Red Hat 7.3. All tools and software packages that we used in our development originate from software bundle incorporated within the Red Hat Linux version 7.3 operating system package. The PDAs used Familiar Linux v0.7 package with kernel version 2.4.19-rmk6-pxal-hh13 as operating system. It is necessary to install packet capture and packet filtering modules on PDA since these packages are not available in the installation package bundle. These two modules were used in multicast forwarding and performance evaluation. We used arm-linux-gcc from tool-chain to make cross platform compilation on red hat 7.3 platform.

We created a six node testbed for our multicast experiments. We study the bandwidth utilization of MRDC in a stationary network scenario and verify the correctness of the MRDC in topology and membership dynamic scenarios. During the evaluation, all WaveLan devices operated on 2.4 GHz bandwidth and communicated at the capacity of 2 Mb/s with transmission power of 1mW. The WaveLan devices were operated in an ad hoc mode. An IP address is distributed to each node before tests. These IP addresses satisfy $@A < @B < @C < @D < @E < @F$.

Stationary Network Scenario and Results

The experimental network setting is shown in Figure 6.11. This topology is similar to [86]. Our network consisted of six nodes among which three are portable PCs (nodes A, B and C) and other three are PDAs (nodes D, E and F). All nodes can hear each other in the MAC layer. A virtual wall is constructed in the network layer via iptables. For example, a filter is set in node A to drop all packets coming from nodes D, E and F. A topology monitoring program developed by Hitachi ran in Monitor to display the multicast tree based on the informations collected by tree monitoring module. In this experiment, a file is multicast from node A to the receivers E and F. Figure 6.12 illustrates two multicast tree structures which were shown by topology monitor during the experiment. That is because MRDC chooses the first discovered path. If node F receives first new CA message from node E, a one branch multicast tree, tree_1, is constructed. Otherwise, a two branches tree, tree_2, is formed. Table 6.8 shows the measurement results. The total throughput is far below the full WaveLan data rate of 2 M/s. There are three reasons. The first, it is due to network layer multi-hop forwarding while nodes were physically placed together. Multicast source and forwarder shared the same wireless channel. The second reason is we did not prevent the original multicast traffic to be injected into the wireless channel. The third one is that two alternative multicast tree contain different number of interior nodes. Tree_1 has three interior nodes while Tree_2 contains four interior nodes. In the former tree, the bandwidth is divided by four (one original traffic and three forward traffic) and in the later tree, the bandwidth is divided by five. In this MRDC implementation, the MRDC overhead comes mostly from multicast packet encapsulation while routing messages such as CA, RAR and RAA messages can be ignored. IGMP control overhead can be ignored. However, tree monitoring overhead is high because we chose a small tree information collection interval. This small interval permit us to observe tree changes in next experiments.

In this implementation, MRDC operates in user space. If we succeed in moving MRDC to kernel, high costly kernel-to-user crossing for store-and-forward packets can be avoided, the overhead caused by encapsulation can be greatly reduced and original traffic will not be injected into network. We believe consequently the data throughput can be significantly improved.

Dynamic Network Scenario and Results

In this test, we removed the virtual wall constructed by iptables. All nodes were initially in the coverage region of the others as shown in Figure 6.13. Vic [90] addressed to a multicast address ran on node A, D, E and F to form a video conference group. A web-cam is connected to node A to serve as a multicast source and send video stream to the conference group. Because vic sends multicast packets at regular intervals to announce membership to other vics, although there is only one video source at application level, each vic is a multicast source at the point view of

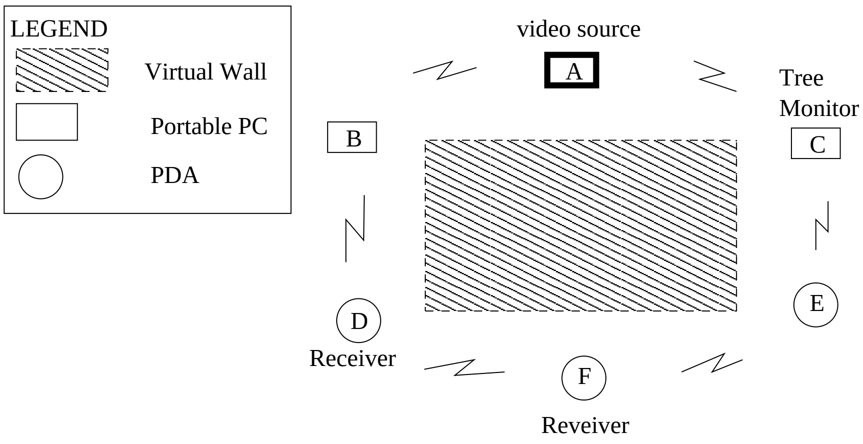
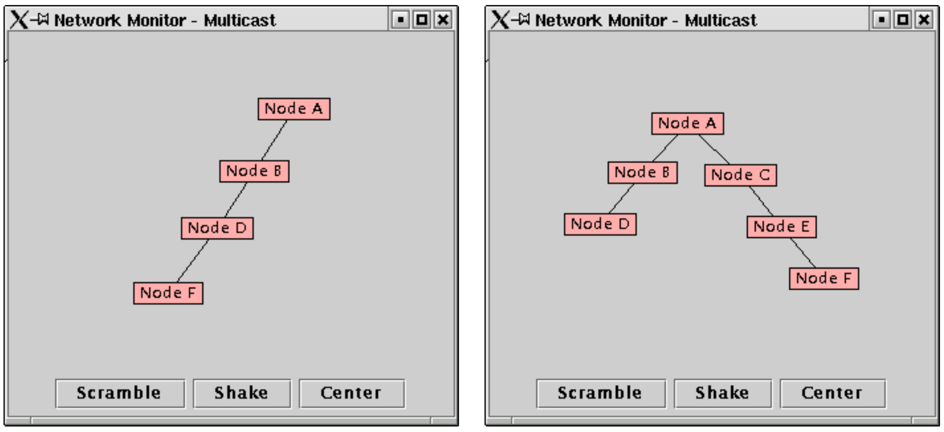


Figure 6.11: Stationary Network



(a) Tree 1 (b) Tree 2

Figure 6.12: Tree Structure in Stationary Network

Table 6.8: MRDC in stationary network with one multicast source

	Value	% of total
MRDC control packet O/H	0.26 kb/s	0.07%
MRDC data packet header	15.75 kb/s	4.11%
Total MRDC O/H	16.01 kb/s	4.18%
IGMP O/H	0.04 kb/s	0.01%
Tree Monitoring O/H	1.29 kb/s	0.34%
Avg. # of multicast tree branch	1.4	N/A
Effective data throughput	365.36 kb/s	95.47%
Total throughput	382.7 kb/s	100%

MRDC. Thus this conference group is a multiple source scenario for MRDC. We configured IP address of wireless nodes to satisfy $@A < @B < @C < @D < @E < @F$. We ran vic first on node D and then on nodes A, E and F. This running sequence resulted in that node D became core. Figure 6.14(a) demonstrates the tree structure.

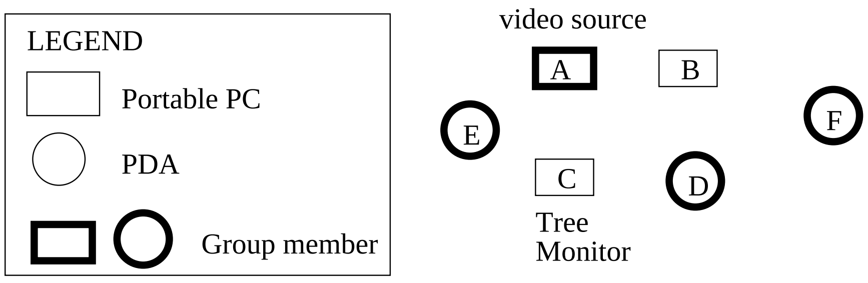
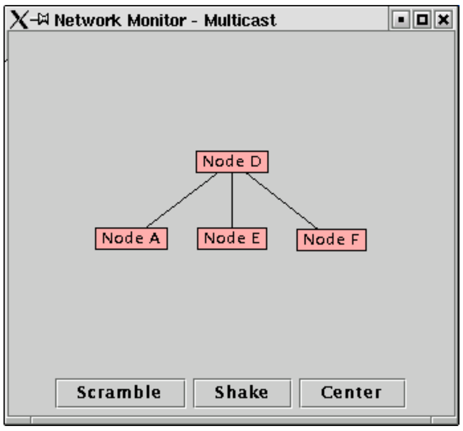


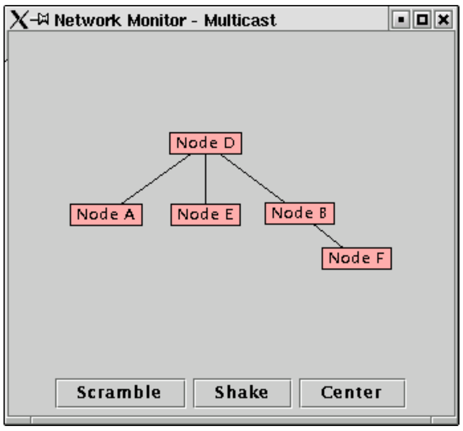
Figure 6.13: Dynamic Network

This test contains two parts: topology dynamic part and membership dynamic part to test correctness and efficiency of MRDC implementation. In topology dynamic part, we moved node F outside of the coverage range of node A and D but still in the coverage of node B. During the movement, node F firstly receives video stream directly from node A. Then multicast tree structure changes (see Figure 6.14(b)) and node F get video stream through the relay of node B. In membership dynamic part, we kept node F outside of the coverage range of A and D but in the coverage of node B and stopped vic on node D. Tree monitoring showed that tree structure changed, and after a short transient time, F became core and node D disappeared. Then, we re-ran vic on node D. This node joined the tree as a leaf node as shown in Figure 6.14(c).

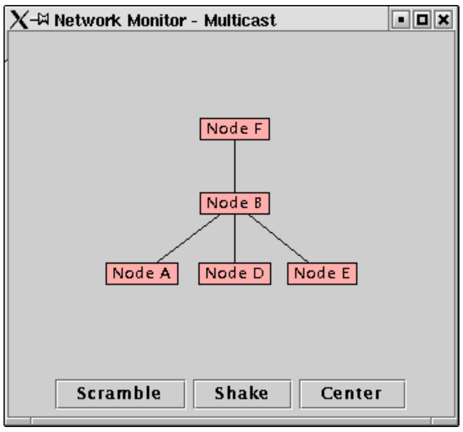
During this test, the video transmission rate was around 300kb/s. The replayed video was fluent in both portable PCs and PDAs and the error rate shown in vic in most of time was smaller than 10%.



(a) Initial Multicast Tree Structure



(b) Multicast Tree Structure after movement



(c) Final Multicast Tree Structure

Figure 6.14: Tree Structures in Dynamic Network

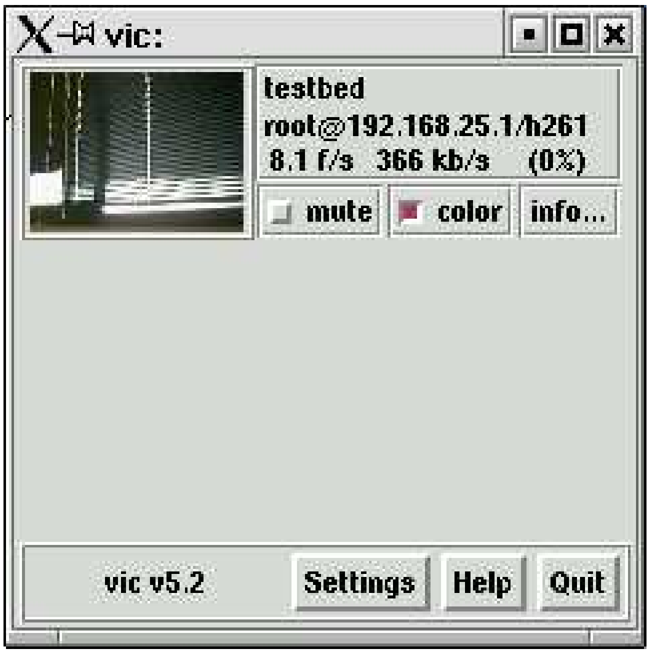


Figure 6.15: VIC and operating on ad hoc testbed

6.4 Conclusion

We presented our experience on implementation of DDR, a unicast routing protocol, and MRDC in an ad hoc testbed which consisted of portable PCs and PDAs. The hybrid network configuration reduce hardware cost of the testbed and facilitates mobility test since PDAs are smaller and cheaper than portable PCs.

Routing programs are decided to run in user space to reduce the difficulty of implementation, installation crossing different hardware and software environment. In order to realize packet forwarding in ad hoc networks, these programs either manipulate kernel ip forwarding table (the unicast routing case) or catch packet from data link layer and then transfer these packet in their way (the multicast routing case). Therefore, the applications can always use standard socket layer interface to communicate.

The main parts of MRDC's control plan, including tree construction and maintenance, have been successfully implemented in the user space of Linux operating system, while only the broadcast mode of MRDC's forwarding plan is employed in the forwarding module of the implementation. We also designed a mechanism in forwarding module to solve the problems of multicast packet forwarding and realize on-demand fashion when program runs in user space. Besides these, IGMP module and tree monitoring module have been designed and integrated in the MRDC implementation to form a complete solution for multicast application supporting and topology monitoring. We evaluated the bandwidth utilization of this implementation in a stationary network scenario and showed that if we do not consider encapsulation overhead, MRDC creates a little control overhead for multicast traffic delivery. Then, we used a MBone traffic - vic to test MRDC with node movement and membership changes. The results prove that MRDC correctly deal with topology dynamic and membership dynamic.

We implemented and tested the implementation of DDR and MRDC seperately. Current IP forwarding table does not store some particular information required in ad hoc network routing. For example the information of last update time, which is necessary for stale routing information detection, is not supported in kernel IP forwarding table. This issue compels DDR and MRDC to possess their own routing tables and obstructs them to share their unicast routing information. We should study an efficient way to facilitate their cooperation in order to reduce routing and storage overhead.

As the progress study of unicast routing protocol based on DDR, new functionalities will be introduced into the implementation to support packet routing cross zones and improve route selection. On the other side, the success of MRDC implementation in user space encourages us to bring these functionalities, or the forwarding module as the first step, into kernel and test the scalability of MRDC. When these works are finished, we will have an ad hoc testbed which can support both point-to-point and many-to-many communications. This testbed will allow us to study protocols of other layers and new ad hoc applications.

Chapter 7

Conclusion

Multicasting for mobile ad hoc networks is crucial studied during last years. It is a method of sending packets to more than one destination node at a time. Using this method, the sender only needs to send every datagram once and compared with broadcast, only relevant routers and hosts take part in the transmission and reception of multicast datagrams. Due to its ability of delivering point/multipoint to multipoint packets in an efficient and scalable way, multicasting is seen as a suitable method to support some potential applications of MANETs which are characterized as group-oriented. However the properties of MANETs such as wireless interface, frequent topology change, limited bandwidth, etc. make the design of multicasting protocol for such type of networks a challenger task. There are many open issues in multicasting for mobile ad hoc networks, for instance: group management, best-effort multicast routing, reliable multicasting, multicast transport and so on. The basic functionality is to deliver multicast packets to their destinations, or multicast routing.

Multicast routing protocols for MANET have twin design goals of high delivery success rate and low overhead. The overhead of routing a multicast packet to its receivers consists of control overhead and forwarding overhead. We have two methods to improve packet delivery success rate either update routing information more frequently to keep tracking topology changes or introduce a certain degree of redundancy to overcome transmission failures. The former method increases the control overhead and later adds extra forwarding overhead. Thus, it is difficult to maximize delivery success rate and minimize overhead simultaneously. Some degree of trade-off between them is always required. However, when looking for this trade-off, the application requirements should be also taken into account. Applications are either sensitive or insensitive to packet loss. For those loss sensitive applications, overhead should concede to delivery success rate, while for loss insensitive applications, reducing overhead becomes more important. In brief, multicast routing protocols for MANET should optimize delivery success rate and overhead with the respect of application's properties.

For this goal, we designed a multicast routing protocol for MANET, named

Multicast Routing protocol with Dynamic core (MRDC). This protocol contains two plans: control plan and forwarding plan. In the control plan, we focused on studying an optimal way of constructing and maintaining delivery structure which should consume less bandwidth for routing and future packet forwarding. In terms of providing transmission efficiency, MRDC employs a tree to connect group members. This tree is source-based for single source group and group-shared for multiple sources group. This multicast tree is periodically refreshed to adapt to current topology so that MRDC maintains its efficiency. In point view of reducing control overhead, multicast trees are constructed when traffic begins and destroyed once transmissions finish. That is what we call **on traffi c demand**. If node's mobility makes tree fragment, a local recovery procedure is executed to repair the tree. However, this procedure just attempts to maintain tree physically connected, logical faults such as logical fragmentation and containing longer path in tree are left to periodical tree refreshing. In this way, the transmission efficiency of tree structure is maintained in most cases and the cost for tree repairing is reduce. The simulation results show that when all protocols use the broadcast-like method to transmit multicast packets, MRDC outperforms ODMRP, a mesh-based multicast routing protocol, and ADMR source-based tree, in packet delivery success rate, end-to-end transmission delay and overhead (both forwarding overhead and control overhead). Furthermore, MRDC provides a stable performance in most cases as network load and nodes' mobility change.

The forwarding plan of MRDC addresses the problems of forwarding multicast packets with the respect of network situation and application requirement. Two transmission modes are defined in forwarding plan if the underlying MAC protocol is IEEE802.11-like. One transmission sends multicast packets with CSMA/CA mechanism without guarantee, just like IEEE802.11 sending broadcast packets. This mode called broadcast transmission mode. The other mode sends a multicast packet as a set of unicast packet with four-way RTS/CTS/DATA/ACK exchange. Thus this mode is called unicast transmission mode. The broadcast mode creates less forwarding overhead and transmission delay but transmission might fail due to collision or wireless interference. The unicast mode has the inverse effect. It gives certain degree of multicast transmission reliability with the cost of extra forwarding overhead and transmission delay. A mechanism, called adaptive forwarding mechanism, is studied to well choose transmission mode according to network situation. This mechanism selects unicast mode in low load networks and broadcast mode in high load networks. Transmission modes can also be smartly selected to support different type of applications. Broadcast mode is suitable for applications which can tolerate packet loss. Unicast mode can be used to support packet loss sensitive applications. If applications have no specific requirement, adaptive forwarding mechanism can be activated to optimize packet delivery success rate and forwarding overhead. The simulation results prove that unicast transmission mode can improve the packet delivery success rate of MRDC from 1% to 7% in low load networks compared with broadcast transmission mode. Well selected parameters allow adaptive forwarding mechanism choose a suitable transmission mode so that MRDC could provide better delivery success rate in both high and low load networks. MRDC can also employ these transmission modes to support different requirements of applications. If applications can tolerate packet loss, MRDC adopts broadcast mode to minimize overhead. On the other hand, for loss-sensitive applications, MRDC activates adaptive forwarding mechanism to provide the best delivery success rate.

Some applications do need a guarantee of multicast delivery (hundred percent deloivery success rate). In order to satisfy this kind of requirement and reducing retransmission overhead, ARMPIS is proposed. This protocol extends receiver-assistant and router-assistant retransmission to distribute retransmission responsibility. In receiver-assistant retransmission scheme, routers firstly query receivers in its sub network the request packets. ARMPIS defines "sub network " as neighborhood. In router-assistant retransmission scheme, routers store multicast packets for retransmission. Considering memory limitation and frequent topology changes, ARMPIS makes router randomly cache multicast packets. In this way retransmission responsibility is distribute to intermediate nodes, which reduce source's retransmission charge and also total retransmission overhead. The simulation results demonstrate that this protocol can provide a 100 % delivery success rate in most cases.

We developed an ad hoc testbed by implementation of DDR, a unicast routing protocol and MRDC so that the testbed can support both one-to-one communications and many-to-many communications. This testbed will allow us to analyze the performance of MRDC in real network. It can also be used to study protocols and new MANET applications.

Appendix A

A.1 Simultation results with confidence intervals

A.1.1 Simulation results of protocol comparison

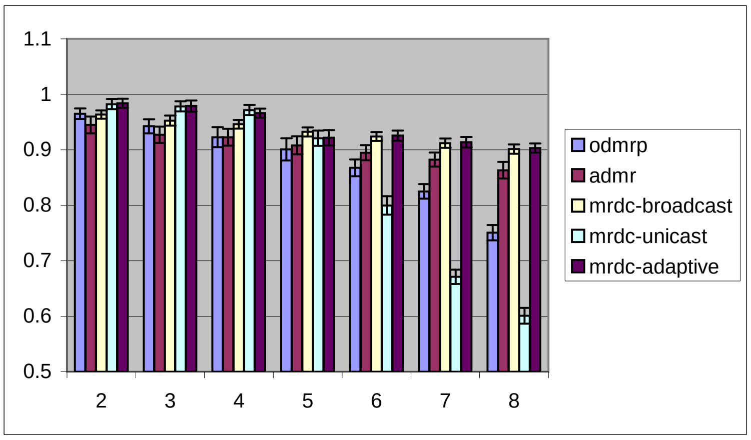


Figure A.1: Packet delivery ratio v.s. Number of source

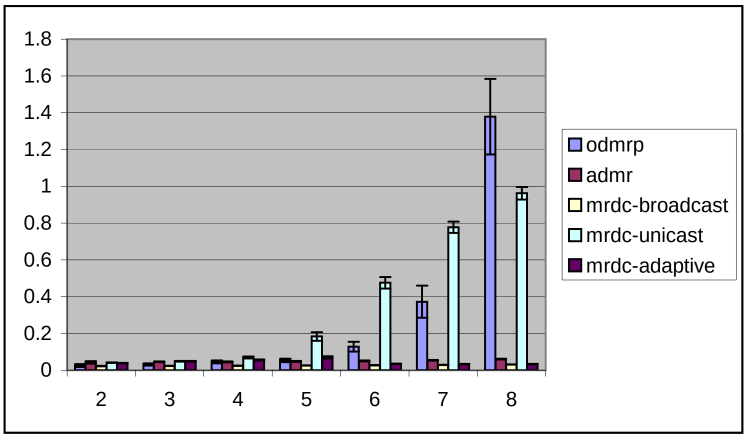


Figure A.2: End to End delay v.s. Number of sources

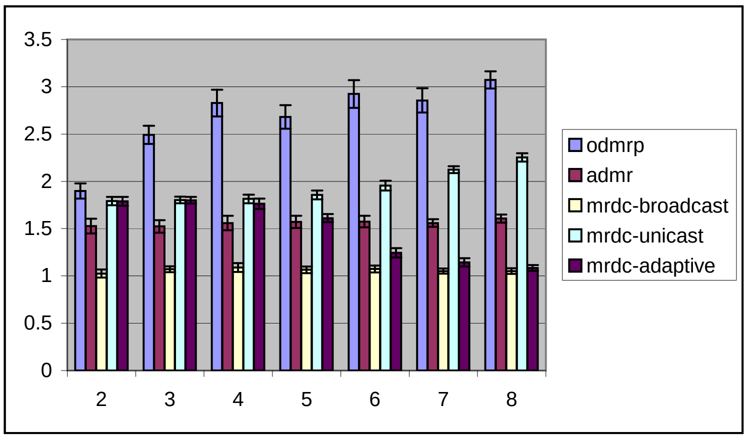


Figure A.3: Number of data packets transmitted per data packet delivered v.s. Number of sources

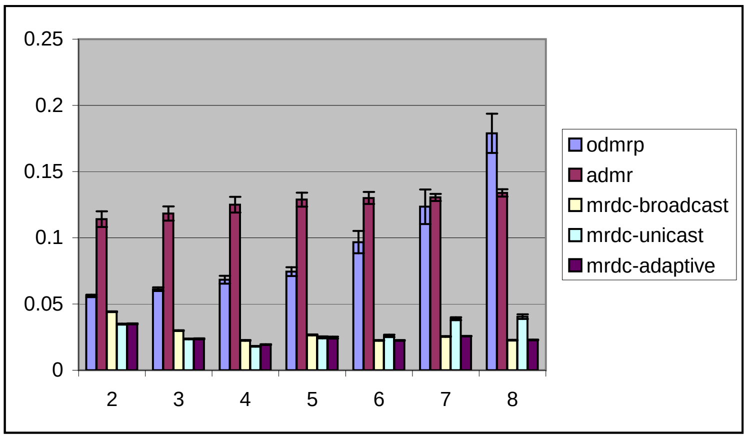


Figure A.4: Number of control bytes transmitted per data byte delivered v.s. Number of sources

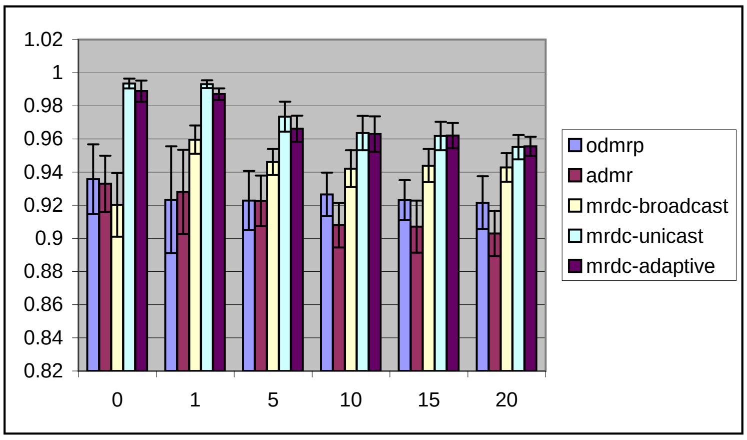


Figure A.5: Packet delivery ratio v.s. Maximum movement speed

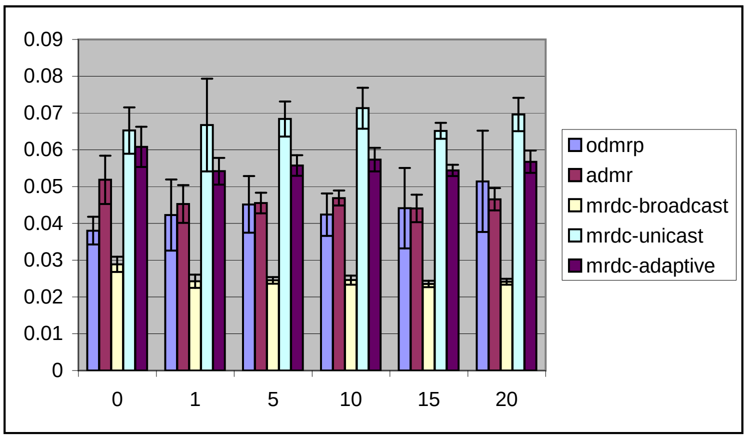


Figure A.6: End to End delay v.s. Maximum movement speed

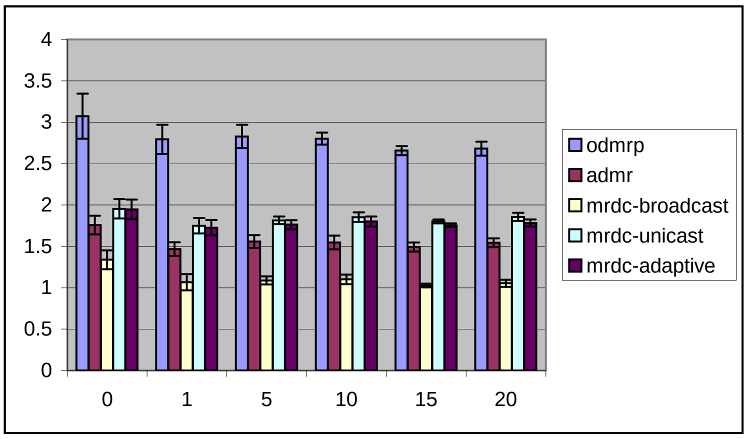


Figure A.7: Number of data packets transmitted per data packet delivered v.s. Number of sources

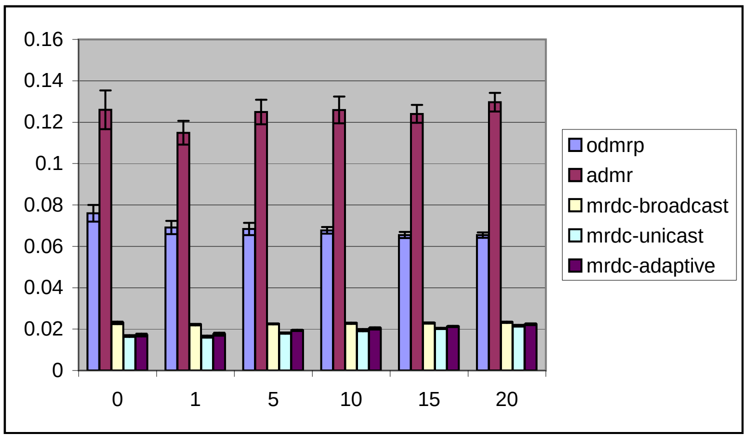


Figure A.8: Number of control bytes transmitted per data byte delivered v.s. Number of sources

A.1.2 Simultation results of reliable multicasting

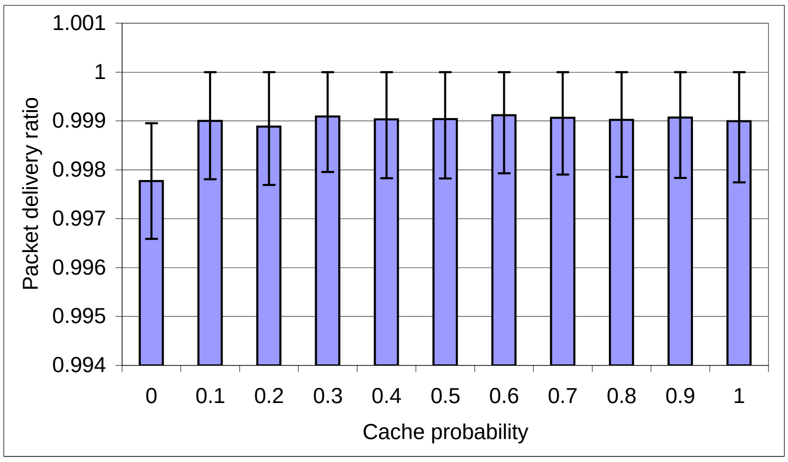


Figure A.9: Packet delivery ratio v.s. cache probability

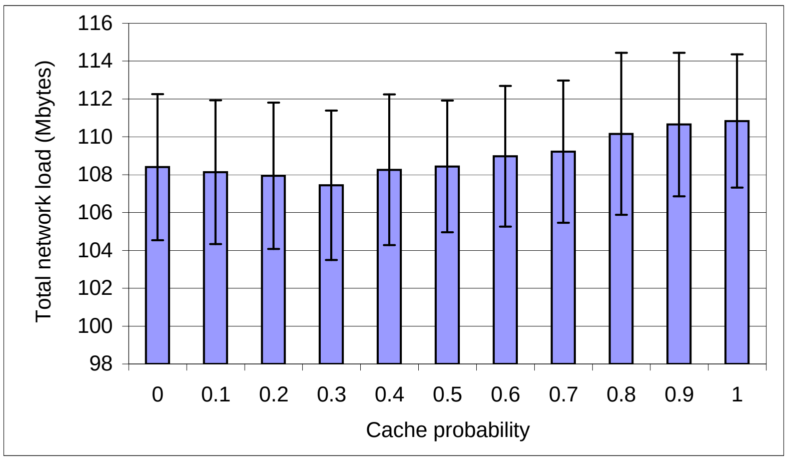


Figure A.10: Total network load v.s. cache probability

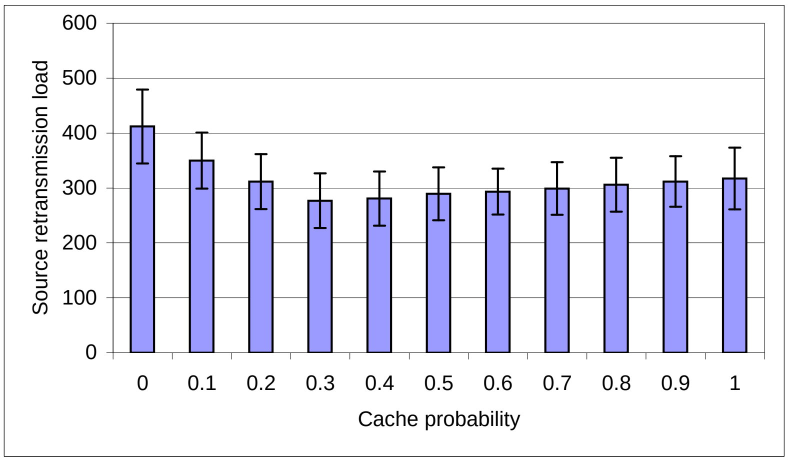


Figure A.11: Source retransmission load v.s. cache probability

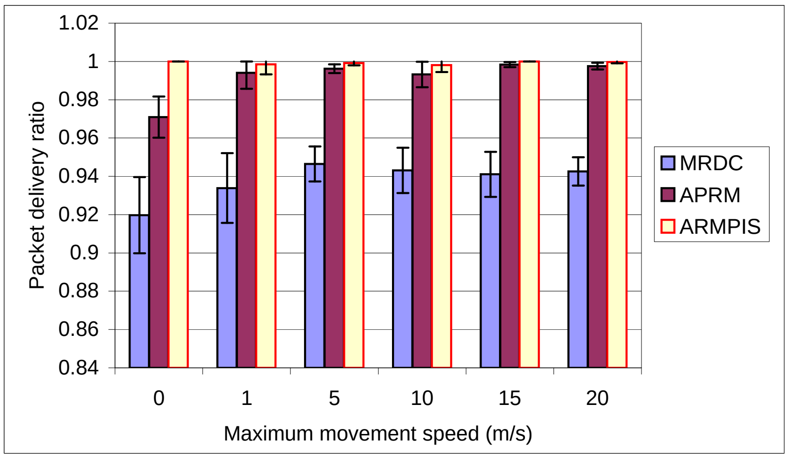


Figure A.12: Packet delivery ratio v.s. Maximum speed

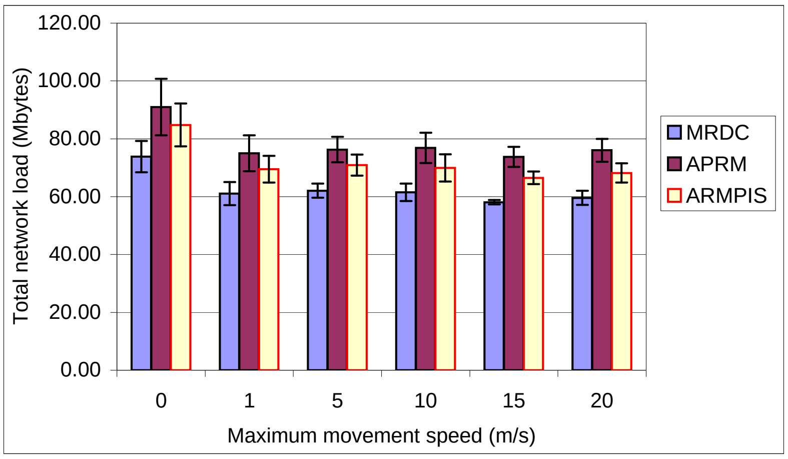


Figure A.13: Total network load v.s. Maximum speed

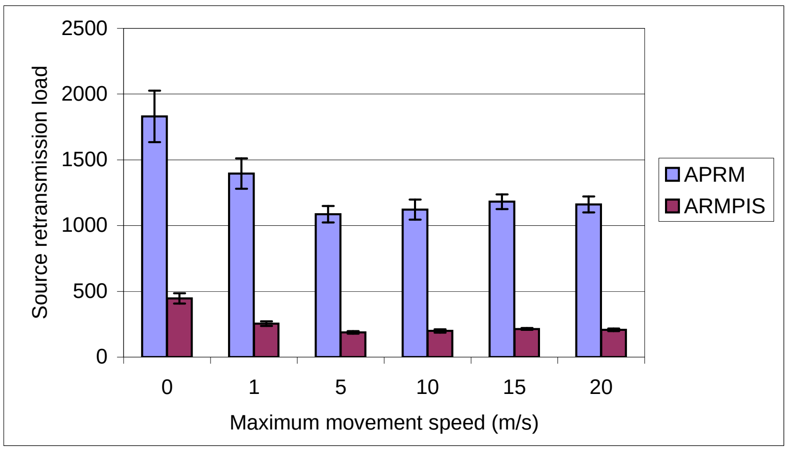


Figure A.14: Source retransmission load v.s. Maximum speed

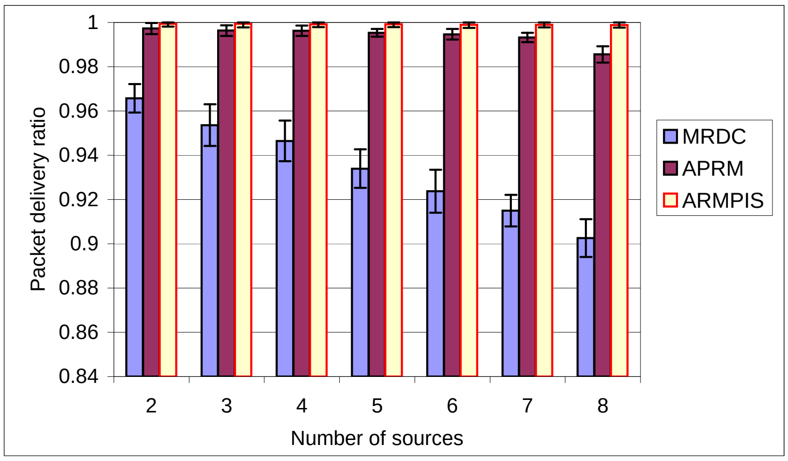


Figure A.15: Packet delivery ratio v.s. Number of sources

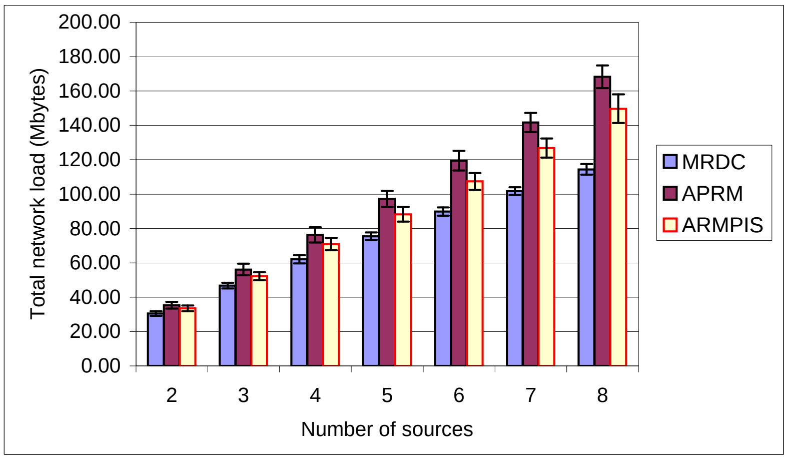


Figure A.16: Total network load v.s. Number of sources

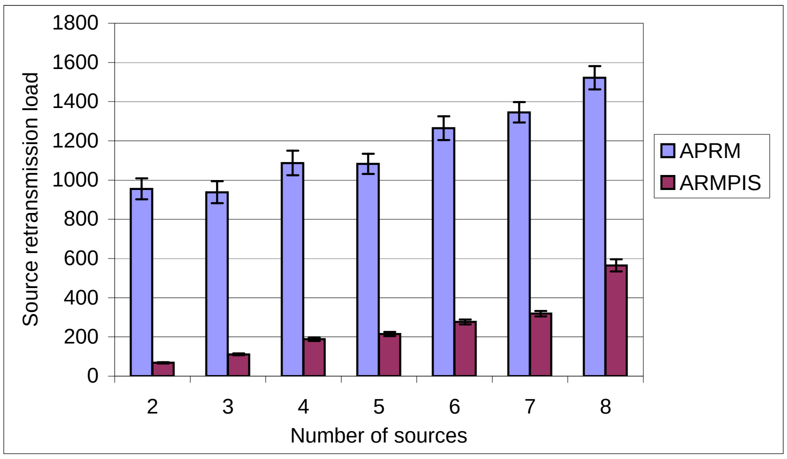


Figure A.17: Source retransmission load v.s. Number of sources

A.2 Flow Charts of Ad Hoc Testbed

A.2.1 Flow charts of DDR implementation

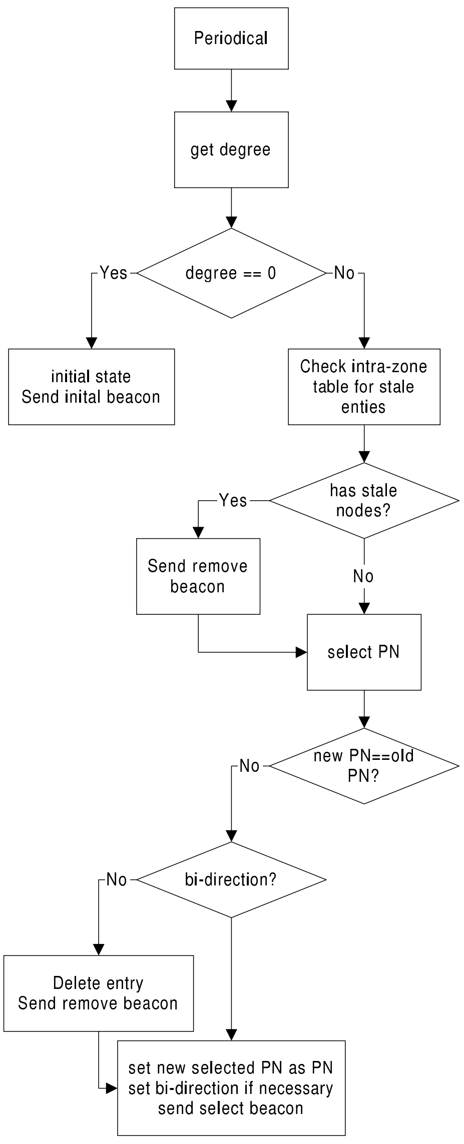


Figure A.18: Flow charts of DDR module

A.2.2 Flow charts of MRDC implementation

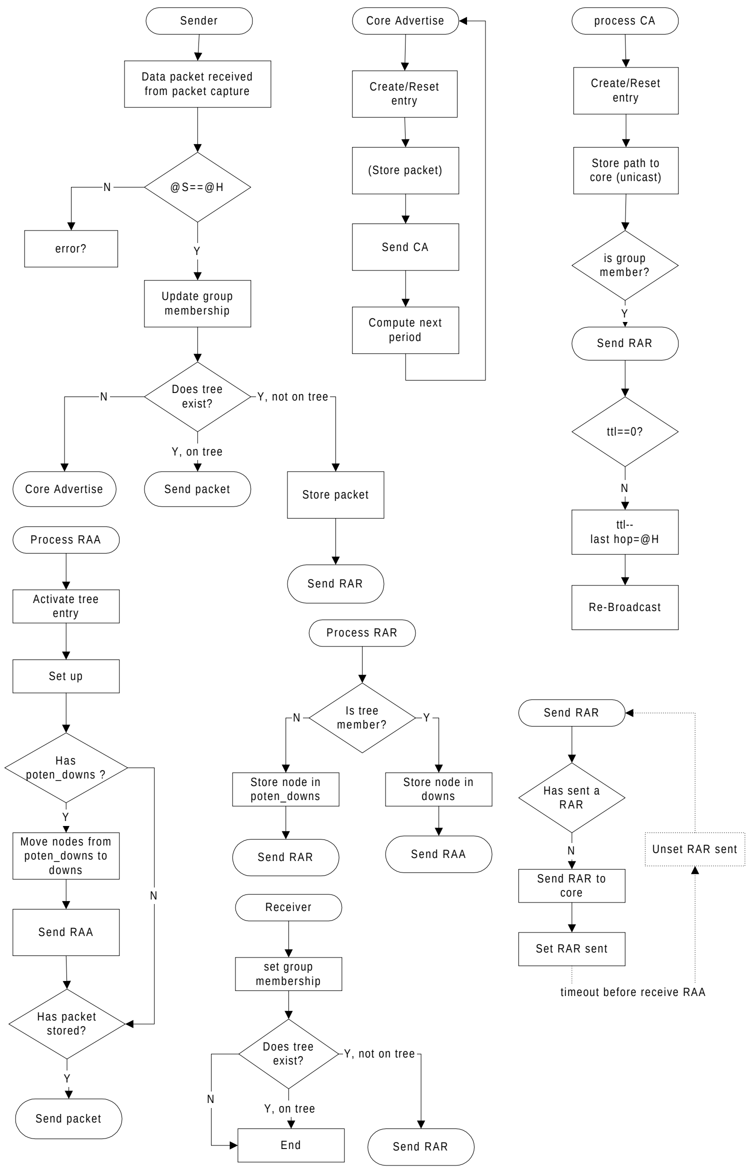


Figure A.19: Flow charts of MRDC module

Bibliography

[1] Mobile ad hoc network (manet). http://www.ietf.org/html.charters/manet-charter.html. Work in Progress.

[2] Elizabeth M. Royer and Chai-Keong Toh. A review of current routing protocols for ad hoc mobile wireless networks. *IEEE Personal Communications*, pages 46–55, April 1999.

[3] Masato Hayashi and Christian Bonnet. A study on characteristics of ad hoc network applications. In *CSN 2002*, Benalm´adena, M´alaga, Spain, September 2002.

[4] S. Deering. Host extensions for ip multicasting, networkworking group. Request for Comments 1112, August 1989.

[5] S. Deering. Host extensions for ip multicasting, networkworking group. Request for Comments 1112, Auguest 1989.

[6] W. Fenner. Internet group management protocol version 2. Request for Comments 2236, November 1997.

[7] B. Cain, S. Deering, I. Kouvelas, and A. Thyagarajan. Internet group management protocol version 3. Internet draft, draft-ietf-idmr-igmp-v3-09.txt, January 2002.

[8] Wanjiun Liao and De-Nian Yang. Receiver-initiated group membership protocol (rgmp): A new group management protocol for ip multicasting. In *ICNP '99*, pages 51 – 58, October 1999.

[9] Kevin Fall and Kannan Varadhan, editors. *ns notes and documentation*. The VINT project, UC Berkeley, LBL, USC/ISI, and Xerox PARC, February 2000. Avaible from http://www-mash.cs.berkely.edu/ns.

[10] Cordeiro de Morais Cordeiro, Hrishikesh Gossain, and Dharma P. Agrawal. Multicast over wireless mobile ad hoc networks: present and future directions. *IEEE Network*, 17:52–59, Jan.-Feb. 2003.

[11] S. E. Deering and D. R. Cheriton. Multicast routing in datagram internetworks and extended lans. *ACM Transactions on Computer Systems*, 8(2):85–110, May 1990.

[12] J. Moy. Multicast routing extensions for ospf. *Communications of the ACM*, 37(8):61–66, 114, Aug. 1994.

[13] S. Deering, D. L. Estrin, D. Farinacci, V. Jacobson, C.-G. Liu, and L. Wei. The pim architecture for wide-area multicast routing. *IEEE/ACM Transactions on Networking*, 4(2):153–162, April 1996.

[14] A. Adams, J. Nicholas, and W. Siadak. Protocol independent multicast dense mode (pim-dm): Protocol specification (revised). Internet draft, draft-ietf-pim-dm-new-v2-01.txt, February 2002.

[15] T. Ballardie, P. Francis, and J. Crowcroft. Core based trees (cbt) - an architecture for scalable inter-domain multicast routing. In *ACM SIGCOMM'93*, pages 85–95, Oct. 1993.

[16] Satish Kumar, Pavlin Radoslavov, David Thaler, Cengiz Alaettinoglu, Deborah Estrin, and Mark Handley. The MASC/BGMP architecture for inter-domain multicast routing. In *SIGCOMM*, pages 93–104, Vancouver, Canada, September 1998.

[17] B. Fenner, M. Handley, H. Holbrook, and J. Kouvelas. Protocol independent multicast sense mode (pim-sm): Protocol specification (revised). Internet draft, draft-ietf-pim-sm-new-v2-01.txt.

[18] Ching-Chuan Chiang, Mario Gerla, and Lixia Zhang. Forwarding group multicast protocol (fgmp) for multihop, mobile wireless networks. *Cluster Computing*, 1(2):187–196, 1998.

[19] Sung-Ju Lee, William Su, Julian Hsu, Mario Gerla, and Rajive Bagrodia. A performance comparison study of ad hoc wireless multicast protocols. In *INFOCOM 2000*, volume 2, pages 565–574, Tel Aviv, Israel, March 2000.

[20] J. J. Garcia-Luna-Aceves and E. L. Madruga. A multicast routing protocol for ad-hoc networks. In *IEEE INFOCOM'99*, pages 784–792, New York, NY, Mar. 1999.

[21] Lusheng Ji and M. Scott Corson. Differential destination multicast-a manet multicast routing protocol for small groups. In *INFOCOM 2001*, volume 2, pages 1192–1201, Anchorage, AK USA, Apr. 2001.

[22] C. W. Wu, Y. C. Tay, and C. K. Toh. Amris: A multicast protocol for ad hoc wireless networks. In *IEEE MILCOM'99*, volume 1, pages 25–29, Atlantic City, NJ, November 1999.

[23] E. M. Royer and C. E. Perkin. Multicast ad hoc on-demand distance vector (maodv) routing. Internet-Draft, Jul. 2000.

[24] S. J. Lee, M. Gerla, and C. C. Chiang. On-demand multicast routing protocol. In *IEEE WCNC'99*, pages 1298–1304, New Orleans, LA, Sep. 1999.

[25] J. Jubin and J. D. Tornow. The darpa packet radio network protocol. In *IEEE*, 1987.

[26] E. Bommaia, M. Liu, A. McAuley, and R. Talpade. Amroute: Ad hoc multicast routing protocol. Internet-Draft, Aug. 1998.

[27] J.J. Garcia-Luna-Aceves and E.L. Madruga. The core assisted mesh protocol. *IEEE Journal on Selected Areas in Communications, Special Issue on Ad-Hoc Networks*, 17(8):1380–1394, August 1999.

[28] L. Ji and M.S. Corson. A lightweight adaptive multicast algorithm. In *IEEE GLOBECOM'98*, volume 2, pages 1036–1042, Sydney, NSW Australia, Nov. 1998.

[29] C-K. Toh, Guillermo Guichal, and Santithorn Bunchua. Abam: on-demand associativity-based multicast routing for ad hoc mobile networks. In *VTC 2000*, volume 3, pages 987–993, Boston, MA, USA, September 2000.

[30] Chai keong Toh. Associativity-based routing for ad-hoc mobile networks. *Wireless Personal Communication Journal*, 1997.

[31] Charles E. Perkins and Elizabeth M. Royer. Ad-hoc on-demand distance vector routing. In *WMCSA '99*, pages 90–100, New Orleans, LA, USA, Febrary 1999.

[32] Prasun Sinha, Raghupathy Sivakumar, and Vaduvur Bharghavan. Mcedar: multicast core-extraction distributed ad hoc routing. In *IEEE WCNC'99*, volume 3, pages 1313–1317, New Orleans, LA, Sep. 1999.

[33] Raghupathy Sivakumar, Prasun Sinha, and Vaduvur Bharghavan. Cedar: a core-extraction distributed ad hoc routing algorithm. *IEEE Journal on Selected Areas in Communications*, 17(8):1454–1465, Auguest 1999.

[34] Prasun Sinha, Raghupathy Sivakumar, and Vaduvur Bharghavan. Cedar: a core-extraction distributed ad hoc routing algorithm. In *INFOCOM '99*, pages 202–209, New York, NY, March 1999.

[35] Jorjeta G. Jetcheva and David B. Johnson. Adaptive demand-driven multicast routing in multi-hop wireless ad hoc networks. In *the ACM Symposium on Mobile Ad Hoc Networking and Computing*, pages 33 – 44, Long Beach, CA, USA, October 2001.

[36] Sang Ho Bae, Sung-Ju Lee, W. Su, and M. Gerla. The design, implementation, and performance evaluation of the on-demand multicast routing protocol in multihop wireless networks. *IEEE Network*, 14(1):70–77, 2000.

[37] Seungjoon Lee and Chongkwon Kim. Neighbor supporting ad hoc multicast routing protocol. In *MobiHOC 2000*, pages 37–44, Boston, MA USA, Aug. 2000.

[38] Shiyi Wu and Christian Bonnet. Multicast rotuing protocol with dynamic core. In *IST 2001*, pages 274–280, September 2001.

[39] Shiyi Wu and Christian Bonnet. A reliable multicast protocol for ad hoc networks. In *WWC 2004*, pages 78–82, San Francisco, CA, USA, May 2004.

[40] Shiyi Wu and Christian Bonnet. Armpis: An active reliable multicasting protocol for ad hoc networks. In *SCI 2004*, Orlando, FL, USA, July 2004.

[41] Editors of IEEE 802.11,Wireless LAN Medium Access Control (MAC and Physical Layer (PHY) specification, Draft Standard IEEE 802.11, 1997.

[42] Philippe Jacquet, Paul Muhlethaler, Thomas Clausen, Amin Laouiti, Amir Qayyum, and Laurent Viennot. Optimized link state routing protocol for ad hoc networks. In *IEEE INMIC 2001*, pages 62–68, December 2001.

[43] Josh Broch, David A. Maltz, David B. Johnson, Yih-Chun Hu, and Jorjeta Jetcheva. A performance comparison of multi-hop wireless ad hoc network routing protocols. In *The 4th International Conference on Mobile Computing and Networking(ACM MOBICOM'98)*, pages 85–97, Dallas, Texas, USA, October 1998.

[44] Hasnaa Moustafa and Houda Labiod. A performance comparison of multicast routing protocols in ad hoc networks. In *Personal, Indoor and Mobile Radio Communications*, volume 1, pages 497–501, September 2003.

[45] Sang Ho Bae, Sung-Ju Lee, and Mario Gerla. Unicast performance analysis of the odmrp in a mobile ad hoc network testbed. In *Ninth International Conference on Computer Communications and Networks*, pages 148–153, Las Vegas, NV, USA, October 2000.

[46] David B. Johnson and David A. Maltz. *Dynamic Source Routing in Ad Hoc Wireless Networks*. Kluwer Academic Publishers, 1996.

[47] Giuseppe Anastasi, Marco Conti, and Enrico Gregori. *Ad Hoc Networking*, chapter IEEE 802.11 Ad Hoc Networks: Protocols, Performance and Open Issues. IEEE Press and John Wiley and Sons, Inc, 2004.

[48] K. Tang and M. Gerla. Random access mac for efficient broadcast support in ad hoc networks. In *IEEE WCNC 2000*, pages 454 – 459, September 2000.

[49] K. Tang and M. Gerla. Mac reliable broadcast in ad hoc networks. In *IEEE MILCOM 2001*, pages 1008 – 1013, October 2001.

[50] Min-Te Sun, Lifei Huang, Anish Arora, and Ten-Hwang Lai. Reliable mac layer multicast in ieee 802.11 wireless networks. In *ICPP'02*, pages 527 – 536, Vancouver, B.C., Canada, August 2002.

[51] http://www.monarch.cs.rice.edu/.

[52] Opnet modeler. http://www.opnet.com/products/modeler/home.html.

[53] L. Bajaj, M. Takai, R. Ahuja, K. Tang, R. Bagrodia, and M. Gerla. Glomosim: A scalable network simulation environment. Technical Report 990027, UCLA Computer Science Departmen, May 1999.

[54] David Cavin, Yoav Sasson, and Andr´e Schiper. On the accuracy of manet simulators. In *the Workshop on Principles of Mobile Computing (POMC)*, Toulouse, France, October 2002.

[55] Mineo Takai, Jay Martin, and Rajive Bagrodia. Effects of wireless physical layer modeling in mobile ad hoc networks. In *MobiHoc 2001*, pages 87–94, Long Beach, CA, USA, October 2001.

[56] Bruce Tuch. Development of wavelan, an ism band wireless lan. *AT&T Technical Journal*, 72(4):27–33, July/August 1993.

[57] Yu-Chee Tseng, Sze-Yao Ni, and En-Yu Shih. Adaptive approaches to relieving broadcast storms in a wireless multihop mobile ad hoc network. *IEEE Transactions on Computers*, 52(5):545–557, May 2003.

[58] M. S. Corson and J. Macker. Mobile ad hoc networking (manet): Routing protocol performance issues and evaluation considerations. Request For Comments 2501, January 1999.

[59] Jinyang Li, Charles Blake, Douglas S.J. De Couto, Hu Imm Lee, and Robert Morris. Capacity of ad hoc wireless networks. In *7th ACM International Conference on Mobile Computing and Networking*, pages 61–69, Rome, Italy, July 2001.

[60] Armstrong S., Freier A., and Marzullo K. Multicast transport protocol. Request for Comments 1301, February 1992.

[61] Heinrichs B. Amtp: Towards a high performance and configurable multipeer transfer service. In W. Effelsberg O. Spaniol, A. Danthine, editor, *Architecture and Protocols for High-Speed Networks*. Kluwer Academic, 1994.

[62] John C. Lin and Sanjoy Paul. Rmtp: a reliable multicast transport protocol. In *INFOCOM '96*, volume 3, pages 1414–1424, San Francisco, CA, USA, March 1996.

[63] Sally Floyd, Van Jacobson, Ching-Gung Liu, Steven McCanne, and Lixia Zhang. A reliable multicast framework for light-weigt sessions and application level framing. *IEEE/ACM Transactions on Networking*, 5(6):784–803, December 1997.

[64] Yavatkar R., Griffioen J., and Sudan M. A reliable dissemination protocol for interactive collaborative applications. In *ACM Multimedia '95*, pages 333–344, 1995.

[65] Sneha K. Kasera, Supratik Bhattacharyya, Mark Keaton, Diane Kiwior, Steve Zabele, Jim Kurose, and Don Towsley. Scalable fair reliable multicast using active services. *IEEE Network*, 14(1):48–57, January/February 2000.

[66] Li-Wei H. Lehman, Stephen J. Garland, and David L. Tennhouse. Active reliable multicast. In *INFOCOM '98*, pages 581–589, San Francisco CA, March 1998.

[67] T. Speakman, D. Farinacci, S. Lin, and A. Tweedly. Pretty good multicast. Internet-Draft.

[68] Don Towsley, James F. Kurose, and Sridhar Pingali. A comparison of sender-initiated and receiver-initiated reliable multicast protocols. *IEEE Journal on Selected Areas in Communications*, 15(3):398–406, April 1997.

[69] Sandeep K. S. Gupta and Pradip K. Srimani. An adaptive protocol for reliable multicast in mobile multi-hop radio networks. Technical report, the 2nd IEEE workshop on mobile computing systems and applications, 1999.

[70] Ming-Yu Jiang and Wanjiun Liao. Family ack tree (fat): a new reliable multicast protocol for mobile ad hoc networks. In *IEEE International Conference on Communications, ICC 2002*, volume 5, pages 3393–3397, New york, USA, April 2002.

[71] Wanjiun Liao and Ming-Yu Jiang. Family ack tree (fat): Supporting reliable multicast in mobile ad hoc networks. *IEEE TRANSACTIONS ON VEHICULAR TECHNOLOGY*, 52:1675–1685, November 2003.

[72] Ranveer Chandra, Venugopalan Ramasubramanian, and Kenneth P. Birman. Anonymous gossip: Improving multicast reliability in mobile ad-hoc networks. In *The 21st International Conference on Distributed Computing Systems, ICDCS 2001*, pages 275–283, Mesa, AZ, USA, April 2001.

[73] Ken Tang, Katia Obraczka, Sung-Ju Lee, and Mario Gerla. Reliable adaptive lightweight multicast protocol. In *IEEE International Conference on Communications*, volume 2, pages 1054–0158, Los Angeles, CA, USA, May 2003.

[74] Thiagaraja Gopalsamy, Mukesh Singhal, D. Panda, and P. Sadayappan. A reliable multicast algorithm for mobile ad hoc networks. In *International Conference on Distributed Computing Systems*, pages 563–570, Vienna, Austria, July 2002.

[75] Ahmed SOBEIH, Hoda BARAKA, and Aly FAHMY. Remhoc: a reliable multicast protocol for wireless mobile multihop ad hoc networks. In *Consumer Communications and Networking Conference,*, pages 146–151, 2004.

[76] Victor O.K. Li and Zaichen Zhang. Internet multicast routing and transport control protocols. *IEEE*, 90(3):360–391, March 2002.

[77] Sagar Sanghani, Timothy X Brown, Shweta Bhandare, and Sheetalkumar Doshi. Ewant: The emulated wireless ad hoc network testbed. In *IEEE WCNC2003*, volume 3, pages 1844–1849, March 2003.

[78] http://www.linux.org/.

[79] Rich Stevens. *Unix network programming*. Prentice-Hall, 1998.

[80] Navid Nikaein, Houda Labiod, and Christian Bonnet. Ddr-distributed dynamic routing algorithm for mobile ad hoc networks. In *MobiHoc 2000*, pages 19–27, Boston, MA, USA, August 2000.

[81] Guy Pujolle. *Les réseaux*. Eyrolles, August 2002.

[82] David A Rusling. *The Linux Kernel*. Online Book, 1999. URL: http://www.tldp.org/LDP/tlk/tlk-title.html.

[83] David A. Maltz, Josh Broch, and David B. Johnson. Experiences designing and building a multi-hop wireless ad hoc network testbed. Technical Report CMU-CS-99-116, CMU School of Computer Science, March 1999.

[84] Lusheng Ji, Mary Ishibashi, and M. Scott Corson. An approach to mobile ad hoc network protocol kernel design. In *IEEE WCNC'99*, pages 1303–1307, New Orleans, LA, USA, September 1999.

[85] J. Postel. Internet control message protocol. Request for Comments 792, September 1981.

[86] Sang Ho Bae, Sung-Ju Lee, and Mario Gerla. Multicast protocol implementation and validation in an ad hoc network testbed. In *ICC 2001*, volume 10, pages 3196–3200, Helsinki, Finland, June 2001.

[87] Navid Nikaein and Christian Bonnet. Harp - hybrid ad hoc routing protocol. In *IST- International Symposium on Telecommunications,*, 2001.

[88] tcpdump. URL: http://www.tcpdump.org/.

[89] R. Russell. Linux 2.4 packet filtering howto. URL: http://netfilter.samba.org/documentation/.

[90] A video conference application. URL: http://www-nrg.ee.lbl.gov/vic.

List of Publications

Conferences and Workshops

Shiyi WU and Christian BONNET, “Multicast Routing protocol with Dynamic Core (MRDC)”, in Proceeding of IST 2001: International Symposium on Telecommunications, Iran/Tehran 2001.

Shiyi WU and Christian BONNET, “An Alternative Packet Transmission Procedure For Mobile Network Simulation”, in Proceeding of SPECTS2002: 2002 International Symposium on Performance Evaluation of Computer and Telecommunication Systems, San Diego, CA, USA, 2002.

Shiyi WU and Christian BONNET, “A Reliable Multicast Protocol For Ad Hoc Networks”, in Proceeding of WWC2004: World Wireless Congress, San Francisco, CA, USA, 2004.

Shiyi WU, Yukihiro TAKATANI, Christian BONNET and Masato HAYASHI, “Implementation and Validation of a Multicast Routing Protocol in an Ad Hoc Network Testbed”, in Proceeding of Med-Hoc-Net 2004 : The Third Annual Mediterranean Ad Hoc Networking Workshop, Bodrum, Turkey, 2004.

Shiyi WU and Christian BONNET, “ARMPIS: An Active Reliable Multicasting Protocol for Ad Hoc Networks”, in Proceeding of SCI 2004 : The 8th World Multi-Conference on Systemics, Cybernetics and Informatics, Orlando, Florida, USA, 2004.

Poster Papers

Shiyi WU, Svetlana. RADOSAVAC and Christian BONNET, “Adaptive power-aware metric for Mobile Ad-hoc Network”, WTC2003: XVIII-World Telecommunications Congress, Paris, France 2002.

Shiyi WU and Christian BONNET “DDR-based Multicast Protocol with Dynamic Core (DMPDC)”, pfhsn2002 : 7th International Workshop on Protocols For High-Speed Networks, Berlin, Germany, 2002.

Demonstration

Jointly by Eurecom, HSAL, Hitachi Sophia Antipolis Laboratory ”Ad Hoc network”, in Symposium2003: 7th symposium of Eurecom-Hitachi, France/Sophia-Antipolis 2003.

Journal

Shiyi WU and Christian BONNET “Synthesis on Multicasting for Ad-Hoc Networks”, submitted to IEEE Communications, Ad Hoc and Sensor Networks Series