MIDDLEWARE ROUTING ALGORITHMS COMPONENTS FOR MOBILE AD-HOC WIRELESS NETWORKS

by

YOUSEF M. ABDELMALEK

A dissertation submitted to the Graduate Faculty in Electrical Engineering
in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

THE CITY UNIVERSITY OF NEW YORK

2010

This manuscript has been read and accepted for the Graduate Faculty in Engineering in satisfaction of the dissertation requirement for the degree of Doctor of Philosophy.

	Prof. Tarek Saadawi
Date	Chair of Examining Committee
	Dean. Mumtaz Kassir
Date	Executive Officer

Prof. Myung Lee (EE Dept. CCNY,CUNY)

Prof. Umit Uyar (EE Dept. CCNY,CUNY)

Prof. Yi Sun (EE Dept. CCNY,CUNY)

Prof. Kaliappa Ravindran (CS Dept. CCNY, CUNY)

Prof. Mehmet Ulema (CIS Dept. Manhattan College)

Supervisory Committee

THE CITY UNIVERSITY OF NEW YORK

Abstract

Middleware Routing Algorithms

Components for Mobile Ad-hoc Wireless Networks

by

Yousef M. Abdelmalek

Advisor: Professor Tarek Saadawi

In this research, we introduce middleware routing algorithms components for Mobile Ad-hoc Networks (MANETs). Unlike the conventional networks, MANET is a decentralized radio wireless network that can be established in situation where no infrastructure exists or where deployment of infrastructure is expensive or inconvenient. This inherent flexibility makes it attractive for applications such as military operations, vehicle to vehicle networks, sensor networks, etc. Hence, MANETs require special type of routing algorithms to operate efficiently in such dynamic environment (i.e., wireless channel, bandwidth constrains, nodes resources, etc …).

In this thesis, we propose an add-on generic solution to on-demand ad-hoc routing protocols to enhance the routing protocols performance with minimum control overhead. Our Solution, namely, Destination Assisted Routing Enhancement (DARE), is based on the new idea of transmitting frequent destination beacon packets. These beacon packets are able to refresh the routing cache tables and announce the destination node existence. This methodology results in dramatically minimizing the initialization (learning/optimization) connection set-time as well as the network overhead. Comparison

between the traditional Dynamic Source Routing protocol (DSR) and the DSR with DARE (DSR-DARE) are presented to show the potential of DARE middleware.

Second, we propose middleware protocol components in order to improve the real-time applications at the receivers' side; we propose an algorithm that gives the receiver dynamic ability to move from one multicast session to another based on the receiver capabilities and the path conditions leading to it. Our Multicast Multi-Stream (MMS) solution is added as an extension to the traditional Protocol Independent Multicast (PIM) protocol. Then, we present cooperative video caching technique in MANETs in order to reduce the average access latency as well as enhance the video accessibility. Efficient video caching placement and replacement strategies are developed at some of the distributed intermediate nodes across the network. The simulations results have shown that the system has better video perception (i.e. Quality of Service).

Acknowledgements

First, I would like to express my heartfelt gratitude to my supervisors Prof. Tarek Saadawi for his academic advice, constant encouragement, guidance, and support during my graduate school years. I am really grateful to him for contributing many suggestions and improvements since I join the City University of New York. My appreciation also goes to my colleagues at the Department of Electrical Engineering (CCNY) for the friendly academic atmosphere, and their encouragement.

Acknowledgement is also due to the members of my doctoral examination committee Professors Myung Lee, Mehmet Ulema, Kaliappa Ravindran, and Yi Sun for serving on my dissertation committee. I have benefitted greatly for their valuable comments for research and writing. I had a great time working on Collaborative Technology Alliance in Communications and Networks (CTA) program cooperating with Prof. Lee and learnt a lot of him.

I am also grateful to Dean Mumtaz Kassir and his office for continuously providing me with advice and guidance through my PhD program.

I want to thank Dr. Kenneth Young for his effort as program manager for the Army Research Laboratory Collaborative Technology Alliance in Communications (ARL/CTA). I also appreciate the cooperation between CCNY/CTA team and Telcordia /CTA team. I want to thank Dr. Mariusz Fecko, and Dr. Johm Sucec for their pioneering scientific discussion.

I am extremely grateful to my parents who have sacrificed themselves to give me the best education. From my early childhood, they raised me to love learning, and supported me to develop my interest in science and engineering. Their unreserved love and support for these many years is what makes this report possible.

Table of Contents

Chapter 1 1
1.1. Motivation 2
1.2. Preview: The Routing Problem in MANETs 4
1.3. Preview: Multimedia Transmission in Ad-hoc Environment 6
1.4. Structural of Thesis 8
Chapter 2 11
2.1. Introduction 12
2.1.1. Table Driven Proactive Protocols 13
2.1.2. On-demand Driven Reactive Protocols 13
2.1.3. Hybrid Protocols 13
2.1.4. Last Encounter Routing Protocols 14
2.1.5. Geographical Routing Protocol 14
2.2. Dynamic Source Routing Protocol 15
2.2.1. Route Discovery 16
2.2.2. Route Reply 16
2.2.3. Route Maintenance 17
2.2.4. Route Cache 17
2.2.5. Limitations 17
2.3. DARE Protocol 18
2.4. Numerical and Simulation Results 25
2.4.1. Analytical Results 26
2.4.2. Simulation Results 30
2.5. Conclusion 36
Chapter 3 38
3.1. Introduction 40
3.2. System architecture 43
3.3. Proposed rate control system 44
3.3.1. Objective 44
3.3.2. Switching up strategy 46
3.3.3. Switching down strategy 47
3.4. Simulation results 49
3.5. Conclusions 55
Chapter 4 56
4.1. Introduction 57
4.2. Related Work 60
4.3. System Architecture 61
4.3.1. Objective 61
4.3.2. Network Architecure 61
4.3.3. Extending Edge Router Functionality 62
4.3.4. Caching Replacement Algorithm 65
4.4. Simulation Results 67

4.5. Conclusions 72
Chapter 5 74
5.1. Introduction 75
5.2. Related Work 77
5.3. System Architecture 79
5.3.1. Selecting Virtual Backbone Nodes 79
5.3.2. System Model 84
5.3.3. Cache Placement Policy 87
5.3.4. Cache Replacement Policy 90
5.4. Simulation Results 92
5.4.1. Effect of Caching Size 94
5.4.2. Effect of Number of Mobile Virtual Backbone Nodes 95
5.4.3. Effect of Zone (Placement Distance) Size 96
5.5. Conclusions 97
Chapter 6 99
6.1. Summary 100
6.2. Future Directions 101

Bibliography 103

List of Figures

Figure 2.1 Schematic diagram for route discovery mechanism 18
Figure 2.2 Schematic diagram show how the destination beacon packets and RREQ operations 21
Figure 2.3 High level flow chart describing the DARE algorithm 22
Figure 2.4 Example of DARE technique using the destination beacon by the destination node X and route request by the source node A 23
Figure 2.5 Network topology 26
Figure 2.6 The probability the source finds the destination versus the life time of the beacon packet 27
Figure 2.7 The network overhead versus the beacon packet TTL 28
Figure 2.8 The probability the source finds the destination versus the life time of the RREQ packet and beacon packet 29
Figure 2.9 The probability the source finding the destination versus the simulation steps for different mobile nodes speed and different destination beacon packet TTL 30
Figure 2.10 Average route discovery time (sec) vs. time (sec) for different TTL beacon packet 33
Figure 2.11 Average route discovery time (sec) vs. speed (m/sec) for different TTL beacon packet 33
Figure 2.12 Average throughput (bit/sec) vs. time (sec) for different TTL beacon packet 34
Figure 2.13 Average route discovery time (sec) vs. number of network nodes for different beacon packet TTL 35
Figure 2.14 Average route discovery time (sec) vs. beacon packet inter-arrival time (packets/second) for different speed network nodes 36
Figure 3.1 Multicast Multi-Stream network topology 43
Figure 3.2 Multi-Streaming system architecture 44
Figure 3.3 Line utilization and Buffer management thresholds 49

Figure 3.4 Flow diagram for the clients dynamic moving 49
Figure 3.5 Simulation topology 51
Figure 3.6 Bit-rate vs. simulation time 52
Figure 3.7 Background traffic vs. simulation time 53
Figure 3.8 Receivers group vs. simulation time 53
Figure 3.9 Bit-rate for receiver vs. simulation time 54
Figure 4.1 Typical setting of DiffServ network 59
Figure 4.2 Structure of edge router layers in DiffServ networks. 62
Figure 4.3 Video retransmission algorithm at the edge router 64
Figure 4.4 Network topology 68
Figure 4.5 The average packet delay versus the time 69
Figure 4.6 Throughput versus the time 70
Figure 4.7 Average packet delay versus caching size 71
Figure 4.8 Number of lost packets versus caching size 72
Figure 5.1 Virtual backbone network architecture 78
Figure 5.2 Virtual backbone selection flowchart 83
Figure 5.3 Collaborative video caching flowchart 84
Figure 5.4 Average latency versus time 93
Figure 5.5 Average latency versus the memory cache size 94
Figure 5.6 Packet loss versus the number of virtual backbone nodes 96
Figure 5.7 Throughput versus the number of hops between two consecutive caching nodes. 97

List of Tables

Table 2.1 Typical value of simulation parameters 31

Table 5.1 System symbols and notations 87

Table 5.2 Average number of hops under different memory cache size (KB) 93

Acronyms

ACO	Ant Colony Optimization
AODV	Ad-hoc on Demand Distance Vector
ARAMA	Ant Routing Algorithm for Mobile Ad Hoc Networks
DM	Dense Mode
DARE	Destination assisted routing enhancement
DSR	Dynamic Source Routing
IGMP	Internet Group Management Protocol
MANETs	mobile Ad-Hoc Networks
OLSR	Optimized link state routing protocol
PIM	Protocol Independent Multicast
QOS	quality of service
ZRP	Zone routing protocol

Chapter 1

Introduction

1.1. Motivation

While applications drive the development for faster and more efficient wireless networks technology, on the other hand wireless networks technology opens up the opportunity for the development of new applications. Applications that were not feasible or even imaginable a few years ago are now widely used. The most current example is the development of multimedia real-time applications. These real-time multimedia applications are pushing the limits on current and future networks, since they require a great amount of bandwidth and, very sensitive amount of network latency, and specific quality of service (QOS) requirements. On the other hand, wireless networks technology has been fueled by advances in communication hardware and technologies that have enabled large scale wireless networks. Wireless multimedia networks and services have engendered a new paradigm that makes imperative use of some new wireless networks technology.

However, recent survey of the commercial world shows that wireless multimedia networks and services are floundering, with a lack of widespread proliferation of such system. We believe that the technological challenges to establish this paradigm are not trivial mission. We briefly state the basic problems as:

- Mobile ad-hoc networks characteristics: An Ad-hoc network is the cooperative engagement of a collection of wireless nodes without the required intervention of any centralized access point or existing infrastructure. These nodes function as wireless routers by discovering and maintaining routes in other nodes in the network. The need for infrastructure-less networks [known as Mobile Ad-hoc Networks (MANETs)] is

growing. This inherent flexibility makes it more reliable, durable, and more scalable. MANETs have the following key features; self-built, self-configured, self-healing and adaptive to dynamic changes.

- Best effort nature of the current internet: Fundamental problem of the internet traffic engineering (TE) is the best effort nature (i.e. no QoS). Multimedia transmission of high quality across such network is still a major challenge. Different approaches have been proposed to improve the network performance. However, these protocols should provide maximum possible throughout and minimum delay to the clients, while being friendly to the existing traffic on the network.
- Bad wireless channel performance: Wireless channels have a high channel bit error rate (BER) and limited bandwidth. The BER pose a challenge for the transport of compressed multimedia in mobile wireless networks. The high BER degrades the quality of multimedia transmission, and also increases the buffer size requirement at the client. Therefore, the degraded performance of wireless transport strongly discourages widespread use of wireless multimedia systems.

The scope of this thesis is confined to designing middleware components for data/video delivery of bursty high-bandwidth delay-sensitive across connections containing a band limited wireless link with a time-varying and at times severe BER.

In the first part of this thesis, Chapter 2, we identify and solve problems related to improving the connection set-up time, average route discovery time, and network performance of MANETs routing protocols. The end result is Destination-Assisted Routing Enhancement (DARE) algorithm that provides significantly improved route

discovery time, throughput, and average delay in MANETs under different mobility speeds, and network topologies, with less network overhead compared to MANET routing protocols.

In the second part of the thesis, from Chapter 3 to Chapter 5, we identify the challenges behind transmitted multimedia over wireless channel. Then we propose Multicast Multi-Stream (MMS) middleware component to enhance the video quality over protocol independent multicast (PIM) protocol in heterogeneous environment. The philosophy behind MMS middleware is to ensure high QoS of receivers within a multicast session. The receivers will have the capability to get the media rate and quality that is compatible with their needs and capabilities. In addition, we propose cooperative mobile caching placement and replacement mechanisms for real-time multimedia over MANETs. The philosophy behind this middleware component is to build cooperative caching mechanism between the mobile nodes. This leads to high video perceptual at the receivers' side and decrease the server load as well.

Section 1.2 introduced MANETs routing protocols limitations. In addition, the potential of using DARE algorithm in ad-hoc environment is presented. We discuss the fundamental challenges to efficient multimedia transport in heterogeneous network in Section 1.3. The organization of the rest of the thesis is described in Section 1.4.

1.2. Preview: The Routing Problem in MANETs

One of the key challenges for researchers, in the field of MANETs, is the routing protocol design ability to carry out a reliable route from the source node to the destination node with QoS requirements. They will want to test its features such as scalability, delay,

throughput, and network convergence, in the presence of rapidly changing link quality, mobility speeds, and route optimization. Although current ad-hoc routing protocols provide the network with routes from the source to the destination, the network resources will be extensively used; a number of these routes may not be needed, scalability issue will founder the connectivity, and collected of unused routing information. In addition, most of the ad-hoc routing protocols depend on flooding to find the paths from the sources to the destinations. This flooding covers the entire network even for regions that may not have source-destination traffic. The route discovery and the route maintenance will cost a large number of control packets. The number of control routing packets will increase dramatically as the network size increases, and nodes mobility increases. In order to tackle the limitations of ad-hoc routing protocols, specifically in critical connectivity network, we are proposing a middleware algorithm that can handle the challenge characteristics of MANETs and overcome the shortcoming of the existing ad-hoc routing protocols. We can summarize the objectives in designing an efficient routing protocol for ad-hoc networks with critical connectivity as follows:

- Low convergence time: to build routes quickly so that they can be used before the network topology changes.
- Robustness: to react quickly, re-established routing when topologies changes destroy existing routes.
- Minimum routing signaling overhead: to minimize the network overhead.

There are many scenarios characterized by critical connectivity, for instance, disaster recovery operation, military battlefield, Ad-hoc Space Networks (ASNs) [1]-[2]. In these critical frameworks, as well as other areas such Saami Network Connectivity (SNC), and

delay tolerant network (DTN) [2], the traditional MANETs routing protocol will not work out efficiently. In this thesis, we contribute on this research trend by proposing DARE module as a generic add-on middleware to MANET routing protocols. The goal of DARE is to assist the routing protocol to establish a path between the source node and the destination node quickly with high reachability probability by generating destination beacon packets, while considering the dynamic characteristic of MANET and the need for low routing control overhead. Simplified mathematical model as proof of concept is presented. We prove that the probability of finding the path between the source and destination using DARE outperforms the same probability using the traditional routing technique. The DARE approach is a generic approach and can be applied to most of the ad-hoc routing protocols. Specifically, dynamic source routing protocol (DSR) is used here as a platform to demonstrate the DARE concept. The simulation results show the potential of adding DARE middleware to DSR protocol to achieve higher network performance in terms of throughput, route discovery time, and connection set-up probability. A route discovery time was reported to be 20% less than the original DSR in mobile ad-hoc nodes with minimum overhead. A comprehensive description of the DARE middleware can be found in Chapter 2.

1.3. Preview: Multimedia Transmission in Ad-hoc Environment

Transmitting real-time video across a heterogeneous best effort network such as internet or infrastructure-less network such as MANETs, while achieving acceptable video perceptual quality at the receiver, is still a major challenge. Generally speaking, the

wireless channel is a source of speculation with any kind of data transfer, because of its characteristics. When delivering video over wireless channel, the problem is intensified because of the nature of the video data. Recent studies have shown that multicast multimedia is estimated by some to comprise 70% of tactical network traffic, [3]. Furthermore, content that is multicast will demand quality of service (QoS) in accordance with application delay and throughput needs. Multicast is bandwidth conserving technology that reduces the network traffic by simultaneously delivering a single stream of information to thousands of corporate recipients. A lot of research has been conducted on multimedia multicast from different angles and views. Such requirements include:

- Ability to serve heterogeneous receivers and support for highly mobile wireless environments.
- Operation without any central infrastructure.
- Minimize the end to end delay.
- Low wireless bandwidth.

Although several literature already studied the problem of transmission of real-time multimedia application over the wireless network in unicast and multicast environment, most of the research and work done in this field considered modify to the transport layer at the sender or the receiver side [4]-[5]. On the other hand by monitoring the application buffer at the sender side or the receiver side is more attention because it is independent on the network layer and transport layer of the network [6]. Moreover, attractive features of the video transmission can be inherited include the end to end delay, average available bandwidth, and flexibly on controlling the QoS.

We contribute on this research trend by introducing three middleware components that can be used positively to enhance the video perceptual quality. First, we propose a mechanism that gives the receiver dynamic ability to move from one multicast group to another multicast group based on the receiver capabilities and the multicast session status. We demonstrate through simulation that the proposed mechanism enforces fairness between the receivers and that the video quality adaptation is smooth so that the impact on the perceptual quality at the client is minimal. Second, we propose a media aware sub-layer middleware that can be added to the edge nodes. To reduce storage capacity requirement, video frames with high priority will be selected to cache at the edge nodes which increases the on-demand video files caching. We have evaluated our mechanism by adding the media aware sub-layer to edge node wherein Differentiated Services (DiffServ) network is enabled. Third, we proposed collaborative video caching techniques combine virtual backbone selection, cache placement and cache replacement algorithms. The cache placement algorithm ensured that the requested data will be cached at the highest reliable intermediate nodes and it prevents too many replications at the virtual backbone nodes. While the cache replacement technique helps in improving the accessibility and keeps the video segments at the caching nodes for its life time. The proposed technique resides above the routing layer, and it can be added as a module over ad-hoc routing protocols.

1.4. Structural of Thesis

Following the introduction in Chapter 1, which summarizes the motivation, and the objectives of this research, Chapter 2 presents an overview of MANETS routing

protocols focusing on the on-demand routing protocols. The DSR is outlined and extensive comparison between original DSR and DSR with DARE (DSR-DARE) is presented.

In Chapter 3 we highlight the recent work in Multimedia transmission in multicast environment. We describe multimedia multi-stream middleware strategies for real-time video streams over PIM. We present our simulation and verify the performance, in terms of received video quality, and network performance.

In Chapter 4, we are proposing a media middleware component that can be added to the edge routers in DiffServ networks in order to have the responsibility to enhance QoS at the receivers' side. Passing such video frames to DiffServ network increases the video quality and treats the video frames based on their priorities. We demonstrate through network simulation that the proposed technique decreases the average packet delay, and the packet loss, which yields a better video quality at the clients compared to existing techniques.

In Chapter 5, we discuss the qualitative and functional advantages of collaborative video caching in ad-hoc mobile networks. In order to reduce the average access latency as well as enhance the video accessibility, efficient video caching placement and replacement strategies are crucial at some of the distributed intermediate nodes across the network. Virtual backbone caching nodes will be elected by executing caching placement algorithm after running the routing protocol phase. We show in the simulation results that the proposed collaborative aggregate cache mechanism can significantly improve the video QoS in terms of packet loss and average packet delay.

Finally, Chapter 6 gives some conclusions on this work and sets the direction of further research.

Chapter 2

Destination-Assisted Routing Enhancement (DARE) Protocol for Ad-hoc Mobile Networks

The purpose of this chapter is to introduce the reader to MANETs routing protocols and describe the DARE middleware design, mathematical model, and simulation results.

We present in Section 2.1, MANETs routing problems and the classification of MANETS routing protocols. Brief description of the related DSR work and the DAR routing mechanism is illustrated in Section 2.2. In Section 2.3, we introduce the mathematical model and the basic description of the DARE mechanism and its flowchart. Section 2.4 is devoted for the simulation results in comparison with the original DSR protocol. Section 2.5 is maintained for the conclusion.

2.1. Introduction

With the increasing use of mobile devices and advances in wireless technologies, MANET has drawn great attention for being part of ubiquitous networks. MANET is an autonomous collection of mobile and/or fixed nodes that can communicate together over relatively bandwidth constrained wireless links, and the network topology may change rapidly and unpredictably over time. Unlike the conventional network, MANET is decentralized radio wireless network that can be established in situation where no infrastructure exists or where deployment of infrastructure is expensive or inconvenient. This inherent flexibility makes it attractive for applications such as military operations, vehicle to vehicle networks, sensor networks, etc. Because of the above challenging features of MANET, it is difficult to employ existing wired routing mechanism with MANET. Therefore, several routing protocols have been designed specifically for MANET environment [7]-[16]. Ad-hoc mobile routing protocols can be categorized into table driven proactive protocols, on-demand driven reactive protocols, hybrid protocols, last encounter routing protocols, and geographic routing protocols.

2.1.1. Table Driven Proactive Protocols

Destination sequence distance vector (DSDV), wireless routing protocol (WRP), optimized link state routing protocol (OLSR) are some of the most well known protocols under the table driven umbrella [9]. Generally speaking, in proactive routing protocols, nodes maintain tables that store the routes all the time to all other nodes in the network. Any change in the network topology will be propagated across the whole network topology which can become difficult when the network size becomes large or the nodes have high speed movement.

2.1.2. On-demand Driven Reactive Protocols

In reactive routing protocols such as ad-hoc on-demand distance vector (AODV) [10], signal stability routing (SSR),and dynamic source routing protocol (DSR) [11], creating paths between the nodes are invoked only on-demand by routing discovery process: either when a data needs to be sent to a new destination, or when a route which is in use fails. However this technique does not need constant broadcasts, but on the other hand causes latency and delays since the routes are not already available in the routing tables. Additionally, the flooding of the network may lead to supplementary routing control traffic which wastes the limited bandwidth.

2.1.3. Hybrid Protocols

In order to combine the merits of both proactive and reactive, the hybrid protocols such as zone routing protocol (ZRP) [12], and cluster based routing protocol (CBRP) are designed. The basic idea is to divide the network into zones and seek to employ proactive routing technique in some areas of the network at certain times and reactive routing for

the rest of the network based on the weakness and strength of these routing protocols. The proactive operations are restricted to a small domain in order to limit the control overheads and delays. The reactive routing protocols are used for locating nodes outside this domain, as this is more bandwidth-efficient in a constantly changing network. The disadvantage of hybrid protocol is that if the zone radius is too large the protocol can behave like proactive technique, while for small zone radius it behaves like reactive technique. In addition, the overlapping between the zones will increase the node redundancy; one node may behave as proactive and reactive technique in the same situation. However it exhibits better performance since hierarchal routing is used; the path to destination may be suboptimal. In addition, each node has higher topological information, and memory requirement is greater.

2.1.4. Last Encounter Routing Protocols

Last encounter based routing protocol (e.g., FResher Encounter SearcH, or FRESH) that utilizes encounter history to create time gradients for routing information in wireless networks. The age gradients are created according to node mobility patterns. This complicated interplay between mobility and last encounter protocols had not been investigated and still in its fancy stage. Fundamental design of FRESH protocol and temporal spatial correlation is studied in [13]- [14].

2.1.5. Geographical Routing Protocol

In contrast to the previous routing protocols, geographic routing protocols utilize physical location information to route the data packets [15]-[16]. There is no exchange of routing messages to establish routes. When a node has data packets to be transmitted to destination node, forwarding decision is made based on the position of the destination

and the position of the neighbors of the forwarding node. It is scalable and has low overhead. But to determine the physical position, each node must be equipped with global position system (GPS) or use some other type of positioning service.

Lately, the on-demand routing protocols have been gaining more attention. Still, for on-demand to be effective, route discovery overhead (which typically involves network wide flooding) and latency must be controlled and managed. We develop Destination-Assisted Routing Enhancement (DARE). DARE is middleware add-on component to improve the performance of ad-hoc routing protocols. Beacons packets are generated frequently from the destination nodes in order to enhance the route discovery phase. DARE is on-demand hop by hop mechanism that establish loop free destination trails after transmitting few of these beacon packets. Loop freedom is guaranteed by use of beacon sequence numbers and hop count information (time to live (TTL)). The probability of a route request packet to intersect with one node that is member of the destination trails increases. In addition, these trails are very efficient pointer to the new location of the destination as it moves. In this Chapter we examine the benefits of DARE deployment and using it as middleware component with DSR. DARE significantly decreases the time needed for connection setup, insures the delivery of data, and provides low overhead.

2.2. Dynamic Source Routing Protocol

We will focus our interest on reactive on-demand (DSR) protocols since these protocols have proven their efficiency for ad-hoc networks. DSR is one of the most well known and popular routing protocols, we will use DSR as a platform to demonstrate DARE mechanism. However, we would like to stress the fact that DARE can be applied

to many other types of routing protocols. The DSR can be illustrated in the following points:

2.2.1. Route Discovery

The route discovery mechanism is illustrated in Figure 2.1. This mechanism is invoked whenever a node wishes to send data packet to destination node which is not in its cache table and/or transmission range.

- The originator node (source node) initiates a route request packets (RREQ) placing its address and the address of the destination node; then it broadcasts this packet to all its neighbors.
- Each neighbor, upon receiving this RREQ consults its cache to find an eventual route to this destination. If this neighbor node is the destination or have the destination address in its cache table, it will return route reply (RREP) back to the sender; otherwise it will rebroadcast the same RREQ packet to all its neighbors after adding its address to RREQ header and learns from this request information to be added to its cache.
- Each RREQ packet carries a sequence number generated by the source node and the path it has traversed. The RREQ packet will be forwarded only if it is not duplicate RREQ packet.

2.2.2. Route Reply

This mechanism is executed by destination node or by intermediate node which has route to the destination node in its route cache. The following actions will be executed:

- Adds this new route to the cache table for future use.

- Adds the destination node or the intermediate node address at the end of the path contained in the header of DSR packets.
- Sends the RREP packet along the path contained in the DSR header.

2.2.3. Route Maintenance

Due to dynamic topology and constrains of wireless channel, forwarding packet node may not receive reception acknowledgement from its downstream link layer. Therefore this node will retransmit the same packet until it receives a positive acknowledgement or reaches the limit attempts number. If the retransmission limit number reached, the node would consider this link as broken link then it will remove each route containing this link in its cache table. In the mean time, the node will also pass route error (RERR) packet to inform the source node and all the intermediate nodes about this link failure. Each intermediate node follows the same procedure and deletes all routes containing this broken link. The source node will launch a new route request in order to find a new route to the destination node.

2.2.4. Route Cache

This procedure is used to avoid invoking the route discovery mechanism frequently. Each new route will be cached in the route cache table for future use. The intermediate nodes will use RREP and RERR packets to update its cache table.

2.2.5. Limitations

Regarding the specifications of DSR we can conclude the following limitations:

- Additional overhead imposed by source routing which hurts the scalability issue.

- The DSR is only efficient in MANETs with moderate number of nodes and moderate node mobility. Problems such as long connection set-up time between source and destination, and un-reachability appear in case of fast moving mobile nodes.
- The aspect of links stability is not taking into account, since unstable intermediate links causes topology changing which may lead to route errors and therefore new route requests.
- Only the first discovered route is used, and the others are cached for future use.

To overcome some of these limitations we are going to propose an improvement to DSR in which we are designing DARE mechanism as sub-layer to enhance the performance of DSR.

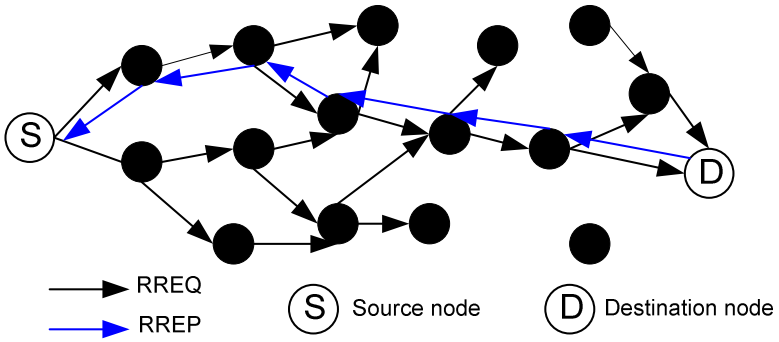


Figure 2.1 Schematic diagram for route discovery mechanism

2.3. DARE Protocol

The main approach in most of MANET routing protocol is based on the concept that the source has the responsibility of finding the destination. This may lead to large connection set-up time and/or difficulty in finding the path in case of high mobility or large size networks. To tackle these problems on ad-hoc on-demand routing protocols, all

the possible routes between the source and the destination should be found quickly before the topology changes while taking into consideration the special network characteristics (i.e., limited energy, limited bandwidth, limited processing power, mobility, and high bit-error rate). In addition, as every node may forward other nodes' data, the network resources usage (such as the limited energy, limited bandwidth, limited processing power, etc) should be fairly distributed across the networks nodes to avoid the high consumption of the resources in parts of the nodes and low consumption in other nodes. The routing algorithm should deal with the rapid changes in the network and it should have the ability to optimize more than one network parameters during the routing process.

We are proposing here a generic algorithm which can be added to many ad-hoc routing protocols to enhance its performance and to overcome the previous ad-hoc routing protocols limitations. To demonstrate the applicability of this approach, we will apply it to DSR protocol as a typical on-demand ad-hoc routing protocol. Figure 2.2 and Figure 2.3 depict schematic diagram showing how the DARE protocol operates as well as its flow chart. In a general view, the destination will participate in the routing process and assist the source in finding the destination. A new type of packet called beacon packets will be sent from the destination to declare its locations as shown in Figure 2.2.b and Figure 2.2.c. The destination node generates beacon packets independently and randomly with relatively low rate to insure low routing overhead. The destination beacon packet will move randomly in the network modifying and updating the caching routing tables.

Upon reception of the beacon packet, the intermediate node i collects the path information (i.e. previous visited nodes ID, destination node ID, beacon packet time to live) from the beacon packet and updates its routing cache table by creating a new routing

cache record leading to this destination node. This new record announces the destination node to node i. The beacon packet will be forwarded to randomly selected neighbor of intermediate node i until its life time expires (i.e. number of hops). The destination beacon packet or/and RREQ will be killed at the intermediate nodes if it exceeds the TTL (i.e. number of hops). The destination beacon packet does not search for the source; however its purpose is to announce to other close by nodes the destination location. This information could be used by any source node in the network. While the destination node sends the destination beacon packet, the source node sends out its regular route request packet. The path between the source and destination node will be established in two cases:

1) If the RREQ reaches the destination node as shown in Figure 2.3.
2) If the route request packet and the destination beacon packet intersect at some intermediate node i.

These two paths from the intermediate node to the source node and from the intermediate node to the destination will be selected to form a direct path from the source to the destination through the intermediate node i as given in Figure 2.2.d and Figure 2.3.

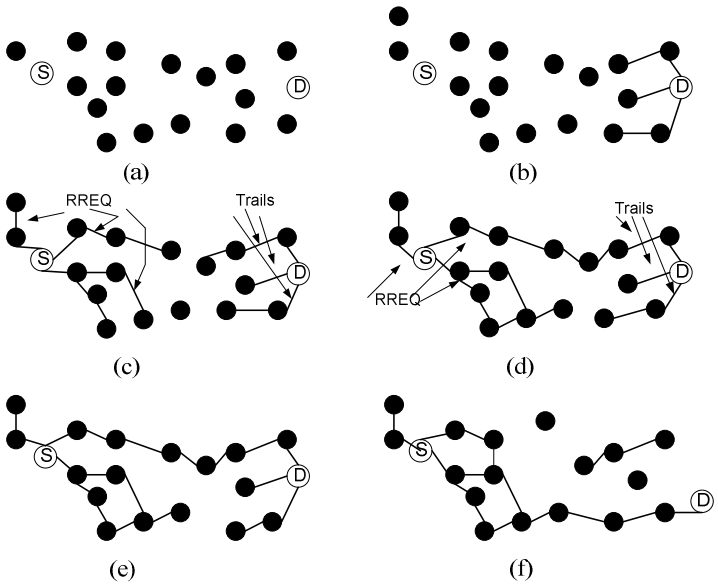
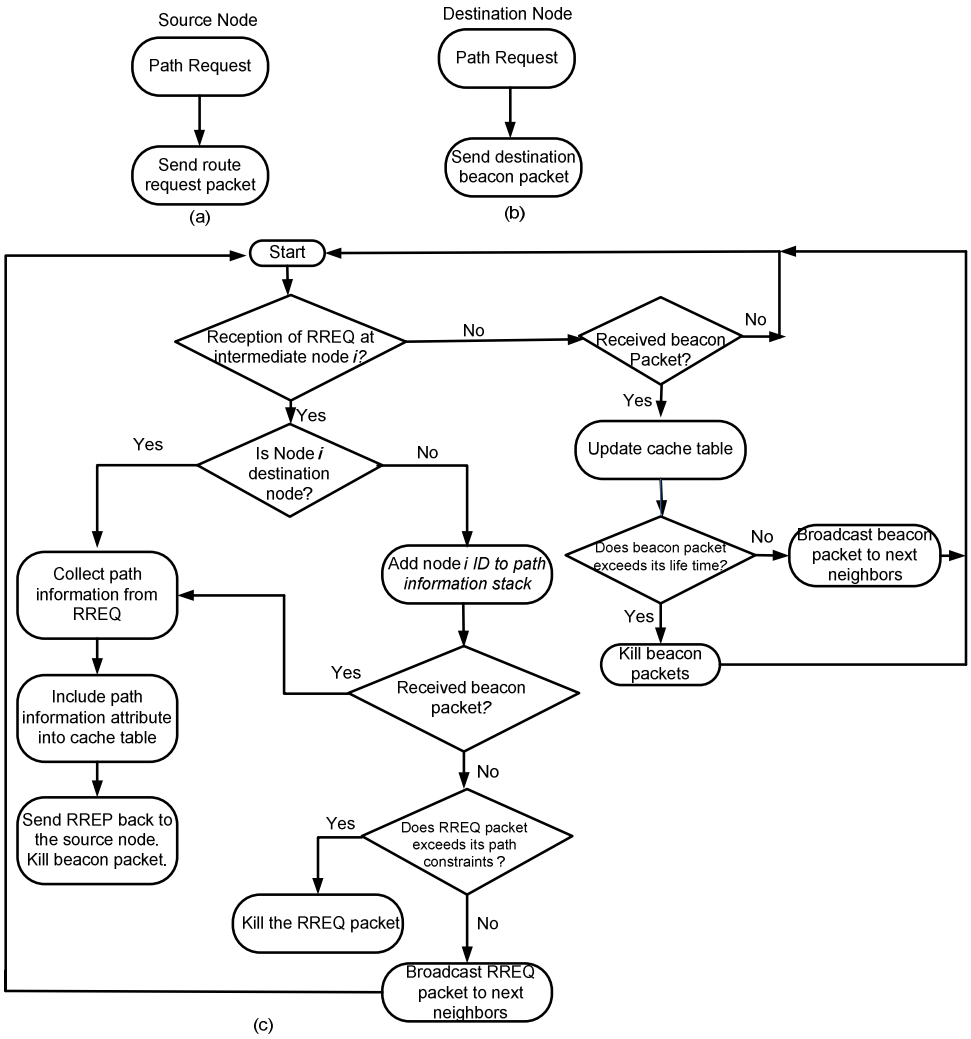


Figure 2.2 Schematic diagram show how the destination beacon packets and RREQ operations

Multiple paths will be available from the source to the destination through many intermediate nodes. In addition, these trails established by the beacon packets are very good pointer to the new location of the destination as it moves as shown in Figure 2.2.e and Figure 2.2.f.

This technique reduces the time needed for connection setup and insures the delivery of data. The main features of the proposed algorithm are:

1. Increase the probability of finding the destination.
2. Reduce routes establishment time.
3. Decrease the routing network overhead in case of large network size.
4. Robustness to react quickly to the topology changes when the nodes mobility increases.
5. The concept of DARE can be added to most of the existing routing protocols



a) Source node (b) Destination node (c) Intermediate node i

Figure 2.3 High level flow chart describing the DARE algorithm

In Figure 2.4 consider a wireless network made of N nodes. The links are distributed according to random wireless graph. In other words a link exists between two nodes if the distance between them is less than the wireless transmitting range. The DARE algorithm is built in two steps simultaneously: the first step is the generation of destination beacon from the destination to declare its location. For example, in Figure 2.4 the destination node X generates 3 destination beacon packets; the packets traverse in the

network randomly exploring the new network topology. The second step is the generation of the RREQ packet from the source. Some intermediate nodes in the network may work as intersection point between the destinations beacon packet and the RREQ packet; these nodes will have the availability to set-up a connection between the source and the destination (e.g. nodes K, M, P in Figure 2.4). The paths from the source to the specific intermediate node and from the destination to the same intermediate node are not overlapped.

Our mathematical model is based on probability analysis approach to evaluate the probability of finding the path between source and destination and compare it with the traditional routing protocol (i.e. without using destination beacon packets). The following assumptions are taken into account:

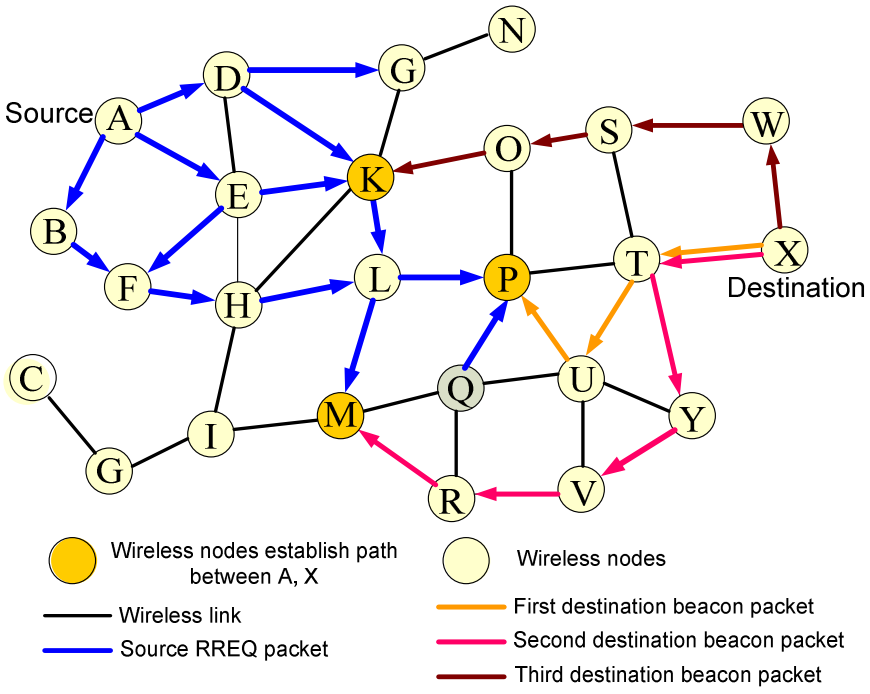


Figure 2.4 Example of DARE technique using the destination beacon by the destination node X and route request by the source node A

The analysis will calculate the probability of finding an intermediate node i with a valid path between the source node S and the destination node D.

1. All possible intermediate nodes will be selected for analysis, this will be affected by the life time of both route request packet and destination beacon packets.
2. For each intermediate node i the following will be calculated:
    a. The probability P_1 that RREQ is sent from the source and reached an intermediate node i

$$\text{Probability for route request packet to reach intermediate i} = p_1 = \sum_{j=1}^{n} \prod_{k_j=1}^{l_j} \text{prob}_j(k_j) \quad (2.1)$$

Where n represents the numbers of all possible routes from the source S to the intermediate node i, and $prob_j(k_j)$ is the probability of selecting the k_j link in route j between the source node S and the intermediate node i. The number of links in route j is l_j.

    b. The probability P_2 that a destination beacon packet is sent from the destination and reaches an intermediate node i.

$$\text{Probability for destination beacon to reach intermediate i} = p_2 = \sum_{j=1}^{m} \prod_{k_j=1}^{q_j} \text{prob}_j(k_j) \quad (2.2)$$

Where m represents the numbers of all possible routes from the destination node D to the intermediate node i and, $prob_j(k_j)$ is the probability of selecting the k_j link in route j

between the destination node D and the intermediate node i. The number of links in route j is q_j.

c. The probability P that intermediate node i will be reached by a single destination beacon and route request packet.

Probability for source destination nodes to reach i

$$P = p_1 x\, p_2 \tag{2.3}$$

The DARE algorithm can be run in two different modes; first as destination location announcement: in this mode the algorithm will use the destination beacons to update the routing tables. Second as initial route detection; in this mode RREQ packets from source and destination beacon packets from the destination will be used to quickly detect the initial route from the source to the destination, which can be used as starting point in the discovery component of the ad-hoc routing protocols. We will run the DARE algorithm in the second mode, using the DSR as platform ad-hoc routing protocol.

2.4. Numerical and Simulation Results

The comparison between the DARE algorithm and the traditional routing technique is illustrated in this section. As mentioned in Section 2.3, the main goal of our algorithm is to increase the probability of finding the destination and thus leading to reduction in the connection setup time between the source and the destination. We divided the results into two folds; in the first part, we provide the analytical proof of concept for DARE algorithm and comparing it with the traditional technique using C and matlab languages [17]. In the second part, we use Opnet network simulation [18] to compare DARE-DSR protocol with the original DSR.

2.4.1. Analytical Results

Figure 2.5 shows the network topology. We have used 50 network nodes. The nodes are randomly distributed over a terrain of size 1000 x 1000 m. Each node is equipped with transceiver which was capable of transmitting signals up to 250 m. We choose to run the worst case as following; node 28 and node 19 is chosen as source node and destination node, respectively. Nodes are mobile with maximum speed 20m/sec.

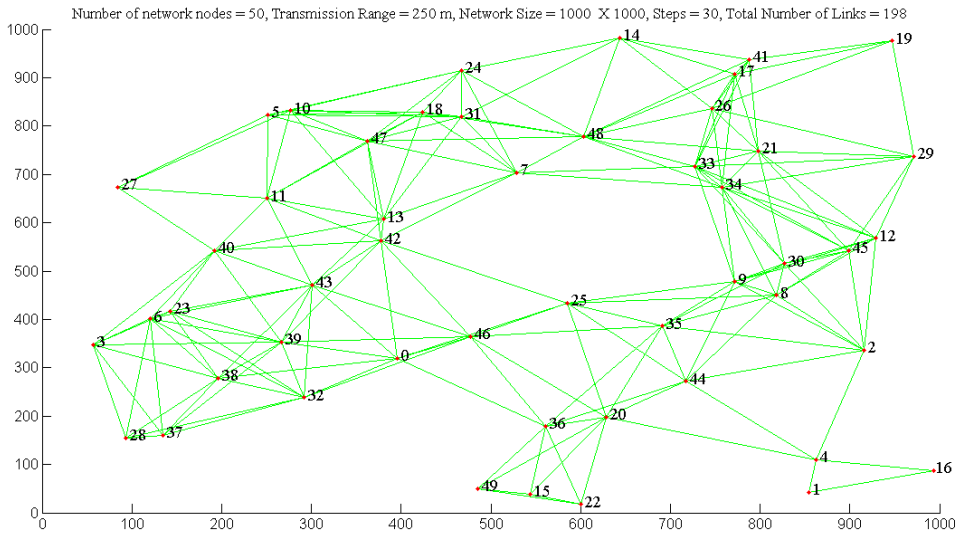


Figure 2.5 Network topology

In Figure 2.6, we have plotted the probability the source finds the destination versus the beacon packet time to live (TTL) for fixed RREQ packet TTL, $t^0 = 12$. As the beacon packet TTL increases, more beacon packets are available in the network for refreshing the routing tables of the intermediate nodes. Thus the probability of the source finding the destination increases. The impact of the beacon packet TTL, B is demonstrated in Figure

2.7 for RREQ time to live, $\ell = 12$. The computational overhead is defined as the number of visited hops due to the beacon packet. In this Figure, we provide the overhead for the same destination node used in the analysis (node 29). With an increase of the beacon packet TTL, the network overhead gradually rises and reaches a saturated value at B = 12.

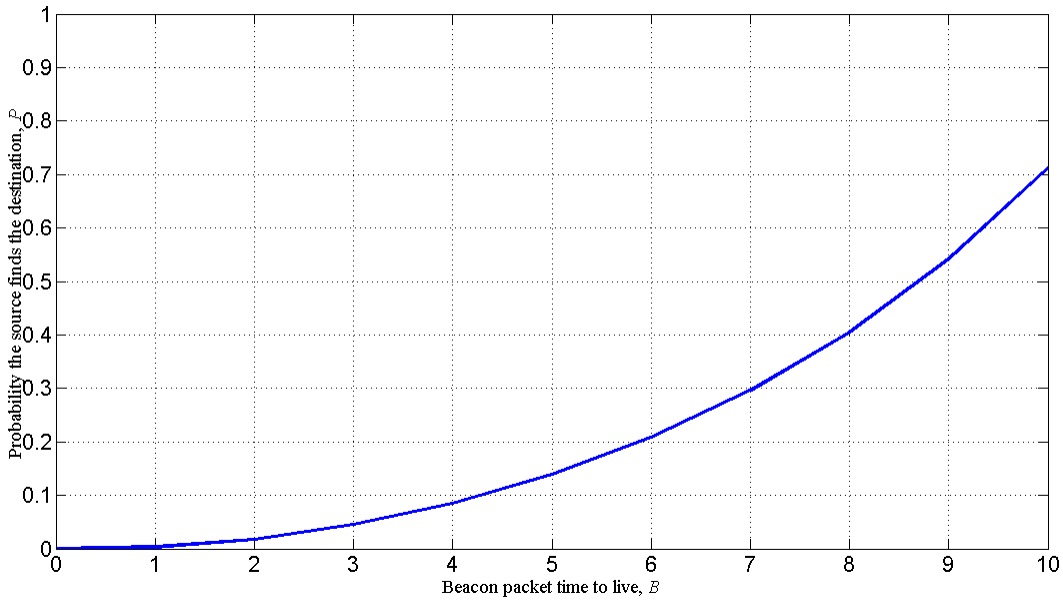


Figure 2.6 The probability the source finds the destination versus the life time of the beacon packet

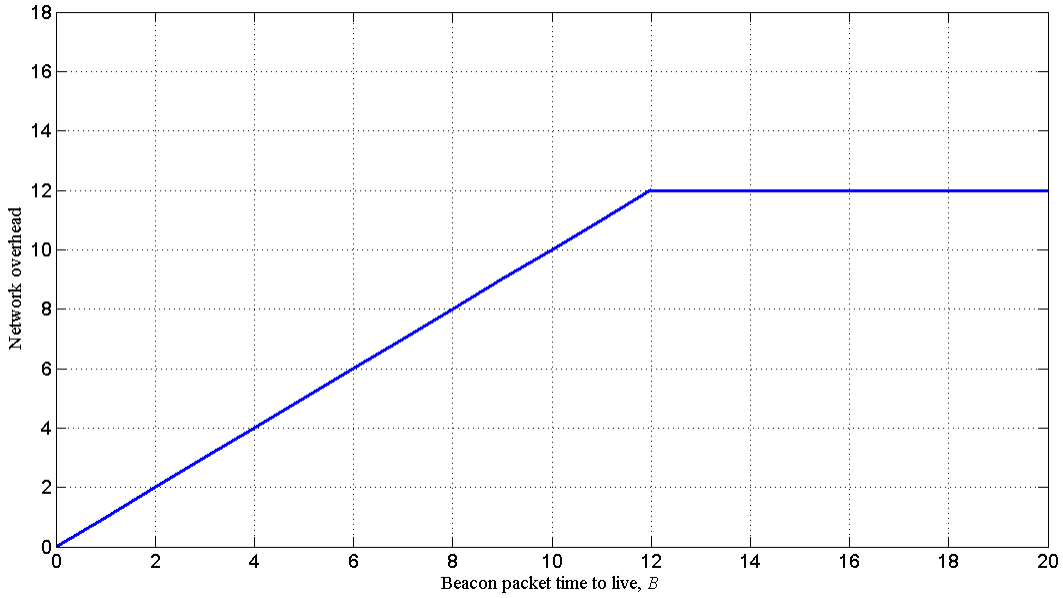


Figure 2.7 The network overhead versus the beacon packet TTL

Figure 2.8 depicts the relationship between the probabilities that the source finds destination versus the life time of RREQ generated by the source for different destination beacon packets life time. The probability of the source finding the destination increases as the life time of the destination beacon packets increases and the life time of the RREQ increases. In addition, it can be deduced that DARE technique gives much more improvement than the traditional routing. To make the comparison between the DARE technique and traditional routing somewhat fair, we proceed as follows. For example if the route request life time is 11 hops and destination beacon life time is 9 hops, the probability to find the destination using the DARE is 0.52. When comparing to traditional routing based algorithm with life time 20 hops, the probability to find the destination using the traditional routing technique is 0.23.

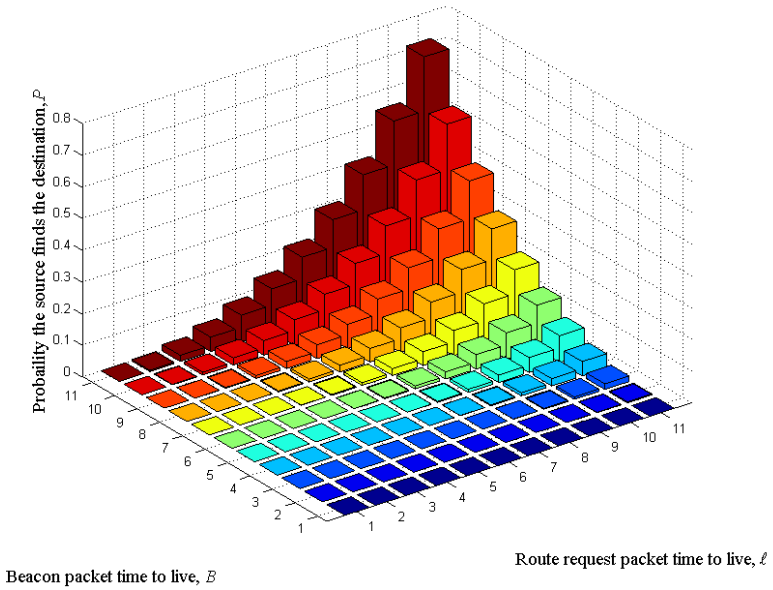


Figure 2.8 The probability the source finds the destination versus the life time of the RREQ packet and beacon packet

Finally in Figure 2.9, we consider a scenario in which all the network nodes are mobile with random mobility speed and direction. The Figure shows how the probability source finding the destination changes as we vary the nodes speed, $V \in \{10,40\}$, the beacon packet TTL, $B \in \{5,6,7,8\}$, and RREQ time to live $\ell = 15$ .The results show that for slower nodes speed, the probability of the source finding the destination increases as expected. However, at larger speeds in the traditional routing protocols the route breakage will require the initiation of the route discovery phase which will take a fairly high route establishment time. With DARE, the beacon packet provides the intermediate nodes with the routing information regarding the destination position, resulting in increasing the probability of the source finding the designation.

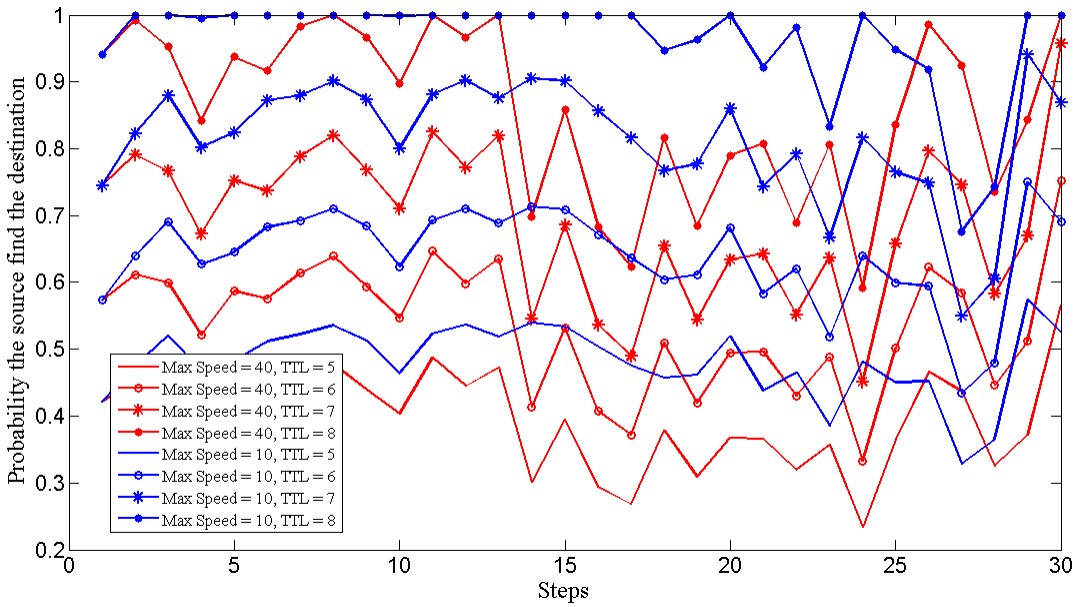


Figure 2.9 The probability the source finding the destination versus the simulation steps for different mobile nodes speed and different destination beacon packet TTL

2.4.2. Simulation Results

We implement DSR-DARE in OPNET [18] and evaluate the performance with DSR routing protocol, which is widely used as on-demand routing protocols. IEEE 802.11 is set as the medium access control (MAC) layer transmission protocol. Data rate at the 802.11 MAC layer is set to 2Mbps. The transmission range is 300m. The wireless propagation model is the two ray ground model. Other MAC layers parameters are set as default. The packet size is 1024 bytes. The time to live (TTL) of the destination packet is set to 1, 4, 8, 10. Table 2.1 details the typical values of our simulation parameters.

We simulated traffic scenario with file transfer protocol (FTP). Inter file message are sent over 1 sec. Network size of 250 mobile nodes is used in our simulation covering an area of 2500 X 2500 m^2. However, in Figure 2.13 we show the effect of network nodes. All the nodes are mobile with 5 m/s in all the Figures. However, Figure 2.11 and Figure 2.14 we

vary the nodes speed to show the effect of mobility on various parameters. Each scenario has been repeated many times with different simulator seeds to test the protocol reliability. Simulation scenarios have been done to compare between the performance of DSR and DSR-DARE when finding the path between the source and destination.

Table 2.1 Typical value of simulation parameters

Parameters	**Value**
Modulation scheme	DSSS,BPSK
Traffic rate	11 Mbps
Radio Tx power	0.005 w
Terrain dimension	2500X2500 m^2
Simulation model	300 sec
Transmission range	200 m
Mobility model	Random way point
Mobility speed	0 m/s to 25 m/s
Pause time	10 sec
Propagation path loss	Two way
Propagation fading model	Rayleigh
MAC protocol	802.11
Default routing protocol	DSR
Number of nodes	250
Traffic	FTP
Packet size	1024 bytes

Figure 2.10 shows the average route discovery time (sec) versus the simulation time (sec) for different destination beacon packet TTL. The proposed idea makes DSR-DARE reduce by more than 20% of the original DSR route discovery time. Moreover, by

increasing the beacon packet TTL, the route discovery time needed to establish the connection between the source and destination is reducing. The advantages of having less route discovery time include: 1) it is less likely to incur route saturation due to many repeated RREQ and packet dropping; 2) less chance of packet dropping due to MAC layer collision. These advantages are especially observed by long beacon packets TTL.

Average route discovery time has been plotted versus the nodes speed in Figure 2.11. All the network nodes are mobile, and the path between source and destination changes over time especially at higher speed scenarios. It is clear that, at all speeds, DSR-DARE outperforms the DSR with respect to the average route discovery time. The longer the beacon packets TTL, the higher the probability that RREQ message has better chance to reach the destination node through intermediate nodes which have refreshed its cache tables with the information provided by the beacon packet. DSR-DARE uses the intermediate nodes as guide to find the destination.

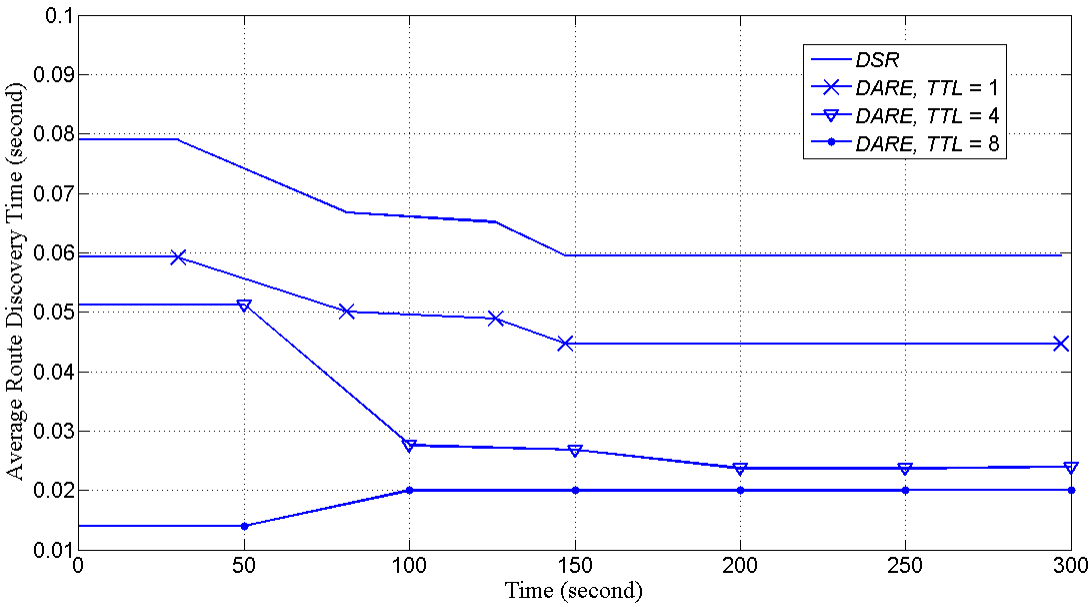


Figure 2.10 Average route discovery time (sec) vs. time (sec) for different TTL beacon packet

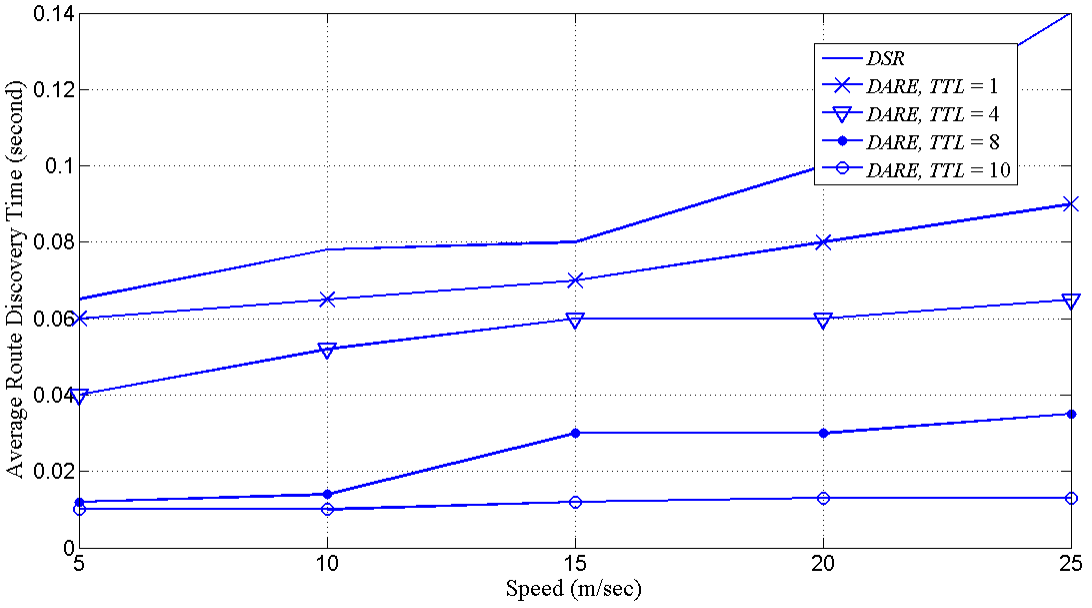


Figure 2.11 Average route discovery time (sec) vs. speed (m/sec) for different TTL beacon packet

DSR-DARE can achieve significantly higher throughout without sacrificing latency performance as shown in Figure 2.12, it is shown that DSR-DARE increases the average network throughput by sensibly sending the beacon packets. In contrast, DSR effectively decreases the average network throughput, especially if the network size is relatively large as shown in Figure 2.13. The proposed scheme is an efficient way to increase the DSR resiliency against the scalability issue with minimum network overhead. In fact, DSR-DARE outperforms DSR in all of our network simulation under different network scenarios.

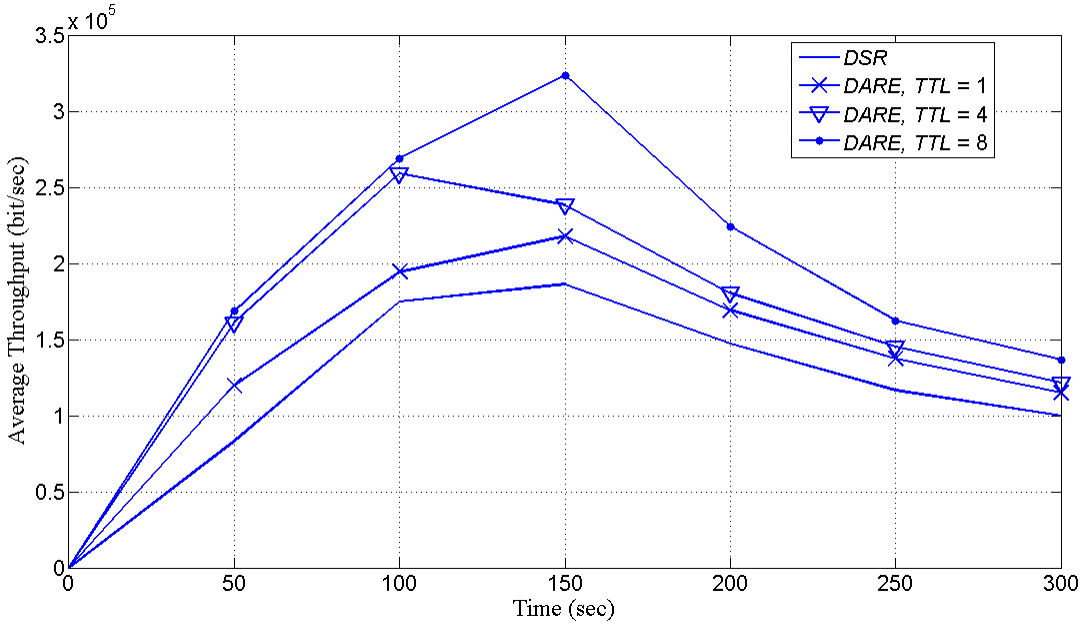


Figure 2.12 Average throughput (bit/sec) vs. time (sec) for different TTL beacon packet

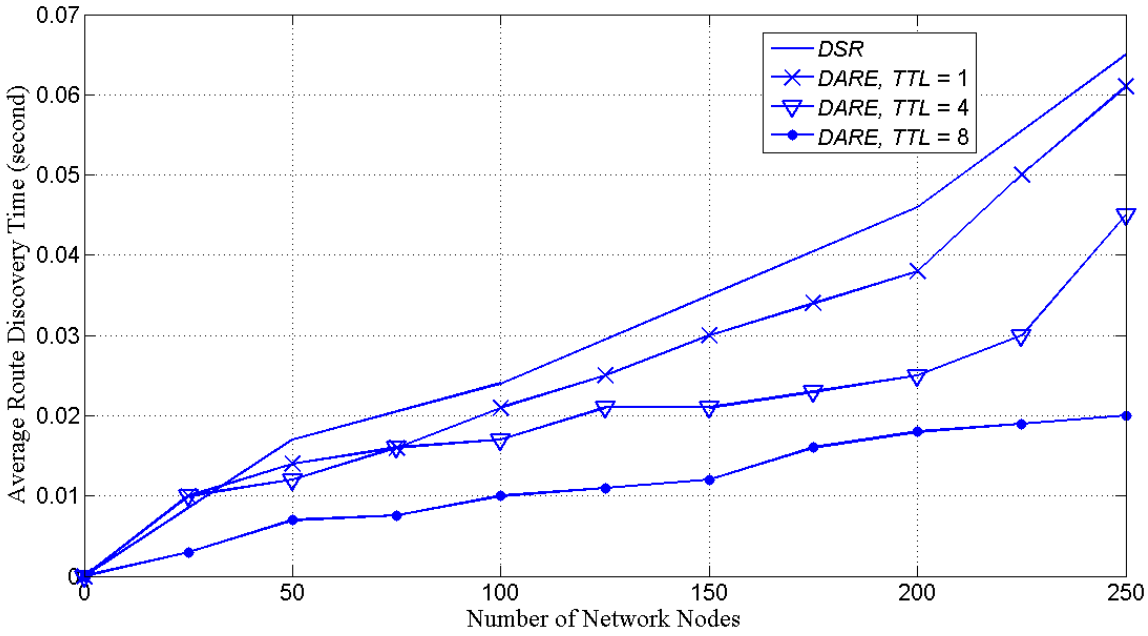


Figure 2.13 Average route discovery time (sec) vs. number of network nodes for different beacon packet TTL

In order to show the improvement that can be achieved using different beacon packets rate, we have plotted Figure 2.14, the average route discovery time against the beacon packet rates for different nodes speeds. The resilience of the DSR-DARE to high mobile nodes speed is increased as much we increase the beacon packet rate. Although the effects of the broken links are severe especially at high nodes speed, the beacon packet generation will tackle this problem by refreshing the intermediate caching tables. As shown in Figure 2.14 increasing the rate of the beacon packets can contribute to decrease the time needed to establish the connection between the sender and receiver.

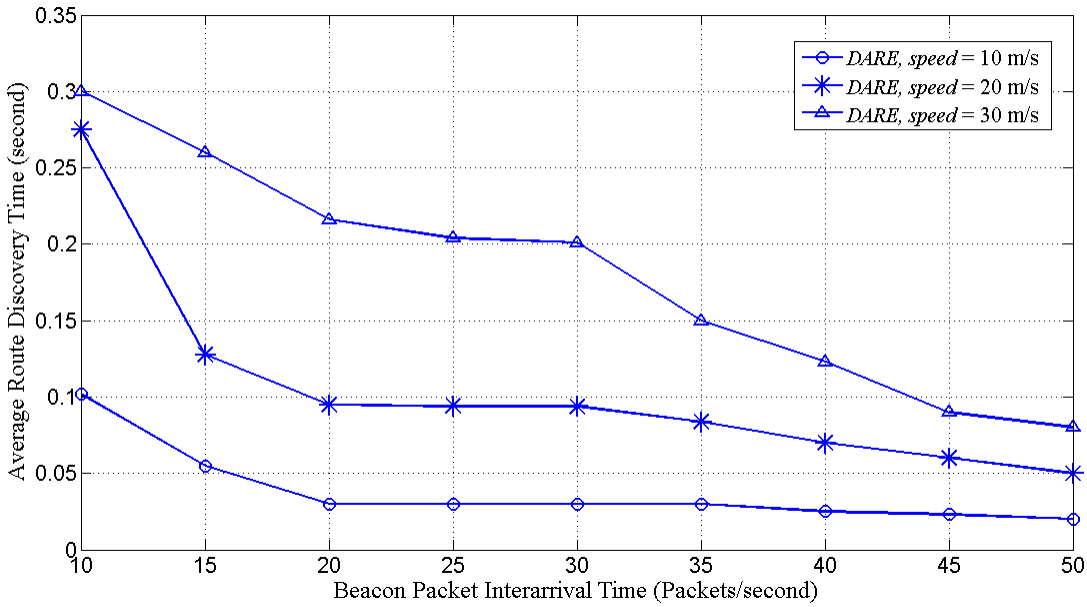


Figure 2.14 Average route discovery time (sec) vs. beacon packet inter-arrival time (packets/second) for different speed network nodes

2.5. Conclusion

The proposed algorithm is a generic middleware component that can reside above many ad-hoc routing protocols. Our performance evaluation demonstrates the efficiency of the DSR using the proposed DARE idea as compared with original DSR. This Chapter presents DSR-DARE protocol that selects and routes data in an ad-hoc mobile wireless network. The method includes sending a destination beacon control packet from a destination via at least one intermediate node at relatively short time intervals to ensure low extra overhead. Simulations have shown that the DARE has higher probability to find the destination with relatively low routing overhead, higher network performance in

terms of network throughput, and route discovery time. Moreover, the DARE protocol does perform well under mobility and achieves lower connection time.

Chapter 3

Multicast Multi-Streaming Video Technique

A key benefit of multicast routing is the reduction of communication costs for applications that send data to multiple recipients. The efficiency of multicast makes it an attractive method for one-to-many content distribution in Army tactical networks where the capacity of wireless links is a scarce resource. By exploiting the efficiency of multicast, wireless link capacity can be conserved (e.g., as compared with what would be consumed by disseminating common content to n users via n separate unicast sessions) thereby making network capacity available for either higher fidelity content distribution or for additional types of application traffic. Examples of possible Army applications that could benefit from multicast distribution include situation awareness data and real-time video collected from reconnaissance aircraft and surveillance cameras. It is the optimization of multicast distribution for real-time video that is the focus of this Chapter.

In recent years, dissemination of multicast real time video over the Internet has emerged as an important area of network research. However, the video distribution task is still challenging for two reasons: 1) the potential for a large number of receivers with different heterogeneous capabilities such as client's buffers, reception rate, processing speed rate, etc and 2) the sensitivity of real time media, in terms of packet loss and average packet delay. Thus streaming applications must deal with persistent rate changes to prevent network congestion and avoid overwhelming the client buffer. Therefore, receivers have to be conscious of their own network bandwidths and choose the stream rate appropriately.

In this Chapter, we address a potentially key that enabling capability for the efficient distribution transmission of the streaming video flow via multiple video sub-streams (e.g.,

SemCast [19]). The objective of sub-streaming is to employ rate/quality adaptation control by judiciously regulating the volume application traffic injected into or propagated through the network. This can be realized (as implemented here, for example) by receivers dynamically joining and leaving the multicast groups associated with specific video flow sub-streams.

The Chapter is organized as follows; Section 3.1 gives a short overview of topics tightly linked to our algorithm and an instance of a typical single stream in multicast environment. In Section 3.2, we describe the proposed multicast multi-stream system architecture. In Section 3.3, we describe the proposed mechanism in terms of receivers' provision joining/leaving the multicast trees based on the tree status and receiver buffer. In Section 3.4, the simulation results for our proposed mechanism are presented. We conclude this chapter with a summary in Section 3.5.

3.1. Introduction

A lot of research has been conducted to address the multimedia adaptation techniques [20]-[28], which can be classified into three categories: receiver-driven approach, sender-driven approach, and hybrid: sender-driven and receiver-driven approach. Receiver-driven layered multicast schemes allow receivers individually to tune the received transmission according to their needs and capabilities. McCanne introduced the receiver-driven approach [20], using the layered video codec to generate multiple video streams from a single video source then transmits each stream as a separate multicast group. The receiver can join/leave the multicast group based on predefined policy. Due to the lack of bandwidth information, the receiver will move from a group to a group in order to get

high video quality. However, the receiver movement will lead to many fluctuations at the video streams. Later, Packet per receiver driven Layered Multicast protocol (PLM) was developed in [21]. Packetpair receiver is used per receiver as a hint for bandwidth estimation. However in all these receiver-driven approach the sender's strategy is predefined and the changing in the network conditions may lead to joining/leaving oscillations at the receivers' side. Most of the sender driven algorithms may be grouped under the umbrella of the quality adaptation techniques. There are three popular techniques for adapting stored video to time-varying available bandwidth: on the fly encoding, adding dropping layers, and switching among multiple encoded versions. On the fly encoding [22] is CPU intensive and thus generally regarded as unsuitable for streaming real time video. In [23]-[24] the adding and dropping techniques are presented. The video stream is partitioned into several layers. It is composed of the basic layer which contains the most essential information for the video playback and the enhancement layers. Thus the server can add or drop layers to regulate the transmission rate based on the network available bandwidth. Quality adaptation based on the multiple encoded versions has been shown to provide better performance and less overhead than adding/dropping layers as in [25]. Sender driven approach for unicast routing protocols based on monitoring the application buffer at the sender side is presented [26]. By both receiver-driven and sender-driven approaches, Leannec et al proposed a hybrid technique for layered multicast in [27]. A mechanism for a rate allocation in each video layer is proposed to provide efficient bandwidth usage for all receivers. However, feedback mergers need to be deployed through the network which increases the system complexity.

We contribute on this trend by proposing a novel technique for controlling the rates at which multicast video content is distributed to users. The approach is independent of the multicast routing protocol and is applicable to both wired and wireless networks. We introduce a fair scheme where each receiver should obtain video stream which is commensurate with the receiver buffer capabilities and the path leading to it, regardless of the other receivers' capabilities. In particular, the video is compressed at different rates with different quality levels, and each version is stored and available for transmission at the server side as shown in Figure 3.1.

The multicast routing protocol constructs different groups (trees) based on the receivers heterogeneity and the geographical spreading of the receivers. Each multicast tree requests for the desired video stream from the video server. Based on the tree conditions and the receiver buffer capability, the clients will be able to join/leave the multicast groups by using IGMP (Internet Group Management Protocol), thus varying the stream rate for the contents that they receive. For example in Figure 3.1, in a typical setting, a single multicasting tree would have been used to serve all the intended heterogeneous receivers, Since the users have different capabilities (e.g., reception bit rate, buffer capacity, etc.), the single multicast tree would be running at the lowest common denominator among all the receivers. In our approach, and in reference to Figure 3.1, we have grouped the users into four groups according to their capabilities, say according to their reception bit rate capabilities. Now, four video streams are generated by the source (one for each group rate) and sent over its multicasting tree. Each user joins the multicast tree according to its capabilities. Users may have move to another multicasting group depending on its receiver buffer conditions and the multicasting tree congestion status.

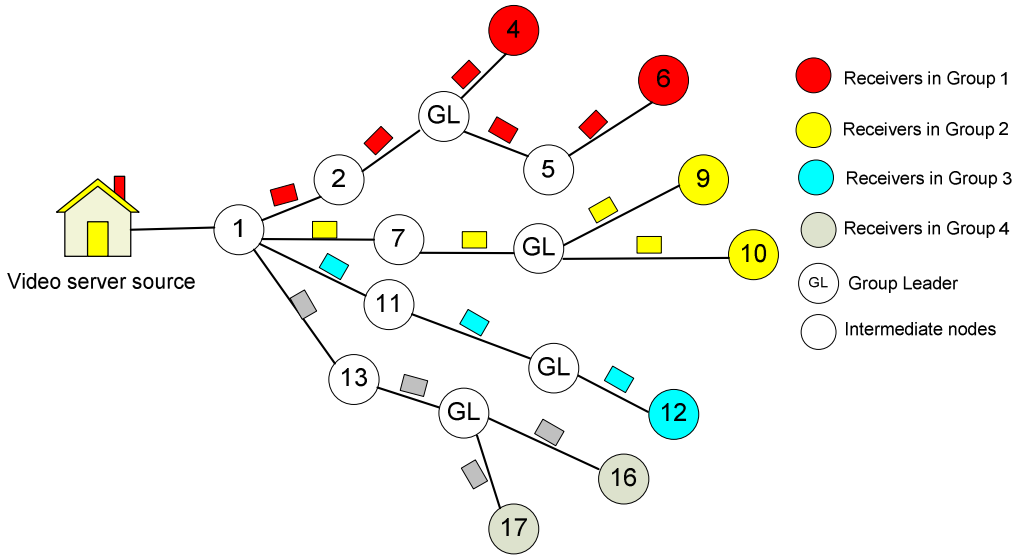


Figure 3.1 Multicast Multi-Stream network topology

3.2. System architecture

Figure 3.1 and Figure 3.2 depicts our architecture, portraying video multicast server, multicast groups, group leaders and a number of receivers that dynamically join and leave the multicast groups based on the network congestion status and the receiver resources status. The sender's role is to encode multiple data streams with different video rates $\{R_1, R_2, ..., R_i, \mathrm{K}, R_N\}$, where N is the number of video streams that can be encoded from the same original video source. All the streams carry the same video but are compressed with different parameters resulting in different video quality. The multicast routing protocol role is to construct multiple multicast groups $\{G_1, G_2, ... G_i ... \mathrm{K}, G_N\}$ from the video source node; each multicast group can carry a video stream with certain rate. Each multicast group, G_i, serves a set of number of receivers $\{U_{ji}\}$, where j is the

number of receivers participating in a certain multicast group and i is the index of the multicast groups$1 \le i \le N$.

In this technique, we aim to enhance the video quality at the receiver side by dynamically moving the receiver from one multicast group to another based on the receivers' capabilities and the availability of the bandwidth.

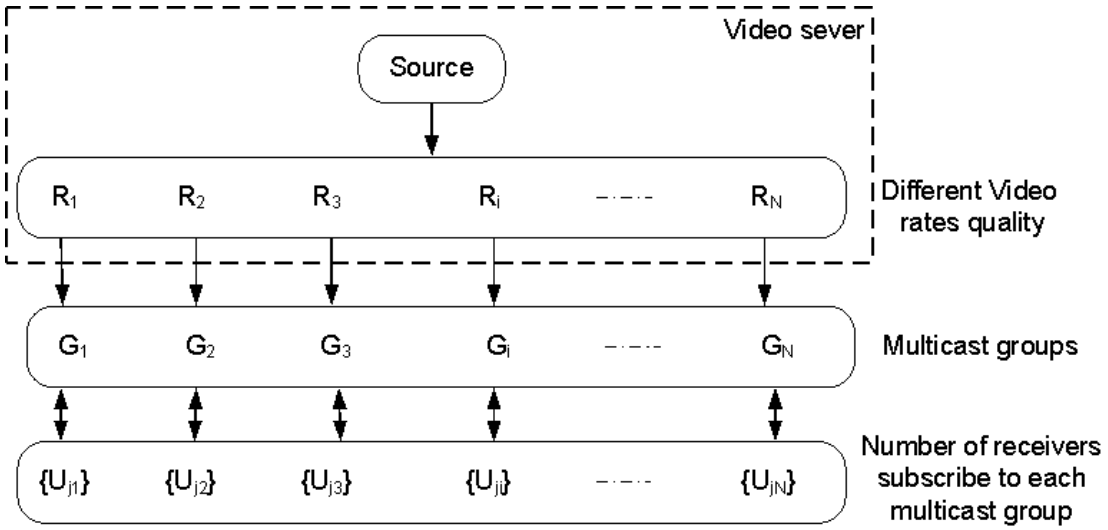


Figure 3.2 Multi-Streaming system architecture

3.3. Proposed rate control system

3.3.1. Objective

The main idea in the proposed system is to serve heterogeneous receivers without losing scalability or introducing network inefficiency. In a general view, the heterogeneity of the receivers on the internet is mainly controlled by two parameters namely, the

available bandwidth and the receiver capability while other constrains comes usually as second considerations.

To gain deeper understanding to our mechanism, we identify a certain policy for the receivers to join/leave the multicast groups. The proposed algorithm monitors the application receiver buffer and the line utilization of the group leaders (assuming that each multicast group contains a router who serves as group leader of the group). The line utilization of the group leader gives us an indication about the multicast group status whether congested or not. The receiver buffer measurements dictate whether the receiver capability will get high or low video quality stream (i.e., higher or lower video bit rate). In addition, it tries to minimize the client buffer starvation. Based on the above two parameters the receiver can dynamically move from one group to another group. The mechanism attempts to minimize the receiver movement (switching to a multicasting tree that can be with higher bit rate or lower bit rate) such that the number of video quality changes decreases the effect on the user perceived quality. We introduce below the criteria used for moving to (i.e, switching) higher or lower multicast tree stream. The algorithm is in the context of a real time stream which the server receives from a live source and forward to the multicast trees.

In Multicast Multi-stream module, we aim to enhance the video quality at the receiver side by dynamically moving the receiver from one multicast group to another based on the following messages:

- Feedback Request (Req_ID, Grp_ID, Time_stamp); this message has been sent periodically from the video source/group leader to multicast receiver to collect the state of the video reception.

- Feedback Response (Res_ID, Rcv_ID, Grp_ID, Congestion level, Strm_Resol, Time_stamp); this message has been sent from the multicast receiver to the multicast source/group leader in order to show the receiver application congestion level.

- Join Multicast Group (Rcv_ID, Grp_ID, Strm_Resol, Time_stamp); this message has been sent from the receiver multicast to the multicast source/group leader in order to send dynamic join movement request.

- Leave Multicast Group (Rcv_ID, Grp_ID, Strm_Resol, Time_stamp); this message has been sent from the receiver multicast to the multicast source/group leader in order to send dynamic leave movement request.

3.3.2. Switching up strategy

Figure 3.3 shows an outline of the operation of our mechanism. We used the line utilization as an estimation of the tree status conditions. The video source measures whether a multicast session is congested or not by measuring the line utilization for each multicast session. The video source monitors the congestion status of the shared line for each multicast session. The line utilization gives indication about multicast session status. We define the line utilization for each multicast session as the ratio between the actual session multicast data transmitted (bit/sec) to the line capacity (bit/sec). When the line utilization is less than a certain threshold Th_1, the video source/group leader will send a Feedback Request message to all the receivers in the multicast session. The receivers react to this Feedback Request message by measuring their buffers occupancy and sending Feedback Response message back to the video source/group leader. If the

receiver buffer size exceeds a certain threshold Th_buff2 or between Th_buff1 and Th_buff2, the receiver maybe move to a higher bit rate tree since its instantaneous available resources can support higher video stream. If the receiver buffer size is less than certain threshold Th_buff1, the receiver maybe move to a lower bit rate tree since its instantaneous available resources cannot afford higher video stream. The receiver requests a new video stream from the video source by sending Join Multicast Group message. The video source checks the new group capability if it will be able to support this new client. In turn it sends positive or negative acknowledgement (Admission Response) to the receiver to either remove it or keep it in the multicast group. Explanation to the dynamic receiver movement is given below.

```
If (line utilization < Th_1)
    If (the receiver buffer fullness > Th_buff2) or
    (Th_buff1>the receiver buffer fullness > Th_buff1)
            return no congestion; /* switch to higher group rate
    If (the receiver buffer fullness < = Th_buff1)
             return congestion;/* switch to lower group rate
End
```

3.3.3. Switching down strategy

If the line utilization is higher than a certain threshold Th_1, Feedback Request message will be sent to all the receivers in the multicast session. The receivers react to this Feedback Request by measuring their buffers occupancy and sending Feedback Response to the video source/group leader. If the receiver buffer occupancy is less than a certain threshold Th_buff1, the receiver will send a Join message asking to join another

multicast group of a lower video rate. Succession joining to this lower multicast group will lead to its removal from the previous multicast group by sending Leave message. The same action will be taken if the receiver buffer fullness is between Th_buff1 and Th_buff2 due to the multicast session congestion. However, if the receiver buffer capacity is greater than Th_buff2, the receiver is in a stable status and will remain in the same multicast group. A clear explanation of the rules that cover the receiver movement is given below:

```
If (line utilization > Th_1)
    If (the receiver buffer fullness < Th_buff1)
    or (Th_buff1 < the receiver buffer < Th_buff2)
        return congestion;/* switch to lower group rate
     If (the receiver buffer > Th_buff2)
       return loaded; /*stable status
     End
```

Some streams may be left without any receivers. Those streams are deactivated by the video source and the corresponding video stream is not transmitted. In order to summarize the description of our overall mechanism strategy, we represent it as a flow diagram as shown in Figure 3.4. In summary, our mechanism can adjust the traffic to match the line utilization and the receiver buffer capacity by controlling the dynamic receiver join/leave multicast groups, causing the network to converge in a non congestion status.

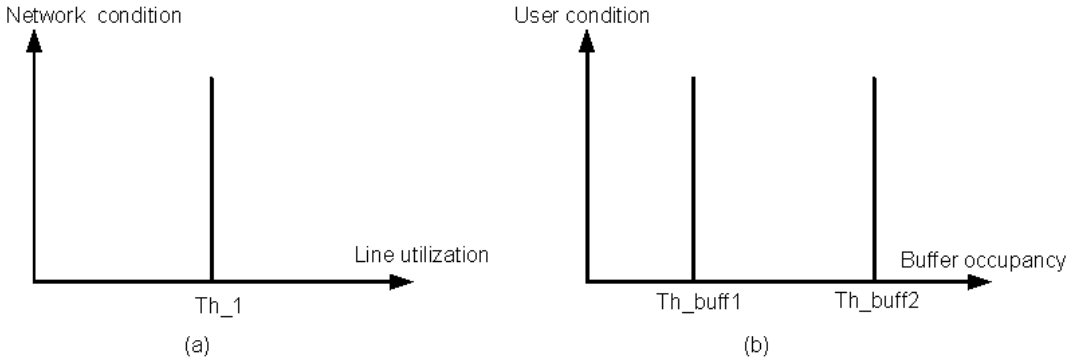


Figure 3.3 Line utilization and Buffer management thresholds

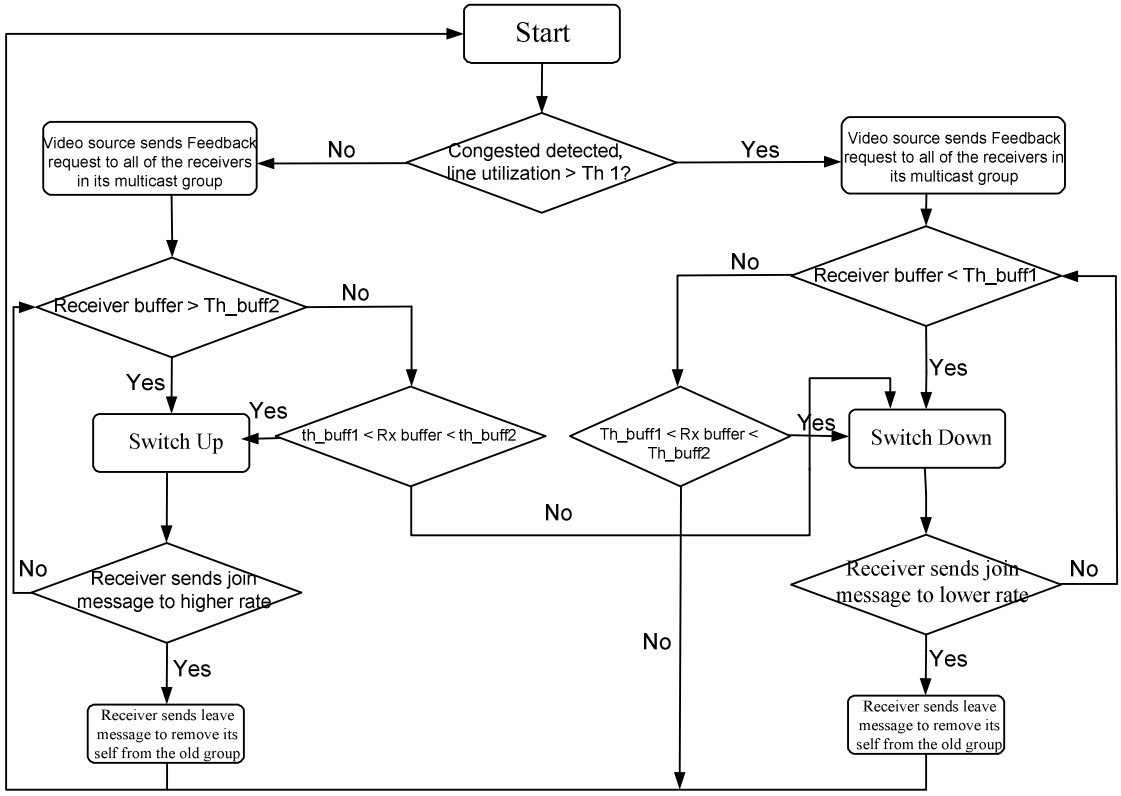


Figure 3.4 Flow diagram for the clients dynamic moving

3.4. Simulation results

In this section, we implement our proposed architecture. We use the network simulator NS-2, [29] for our simulation. The purpose of the simulation is to measure the

performance of the algorithm scheme with the multi versions video technique. Figure 3.5 displays the topology of the simulated network. Cisco's Protocol Independent Multicast (PIM) multicast routing protocol is used, [30]. All the routers (intermediate nodes) are fairly queued and each router has a queue size of 50 packets. We simulate a dynamic network conditions by generating several CBR flows as background traffic. We first discuss some important parameters and aspects in our implementation and then we introduce the results obtained from the simulation.

The source S1 is representing the video server. We used Xvid video codec [31] to encode a $320 \times 240 - 15$ frames per second (fps) video sequence at different video streams quality rates, $R_i \ (100, 255, 420\, Kbps)$. Each one of these video streams will feed into a multicast tree. Therefore we assume that we have 3 multicast groups. Each multicast tree has initially 3 receivers. We have used a different receiver's buffer capability to highlight the receivers heterogeneous as shown in Figure 3.5. Also we have a wide mix of the distances between the receivers and the group leaders by varying the round trip delay of the available paths. We measure the video quality in terms of the achieved average bit rate, variation in the quality at the clients, and the availability to prefetch buffer starvation which can result in receiver dynamic movement to higher or lower multicast group.

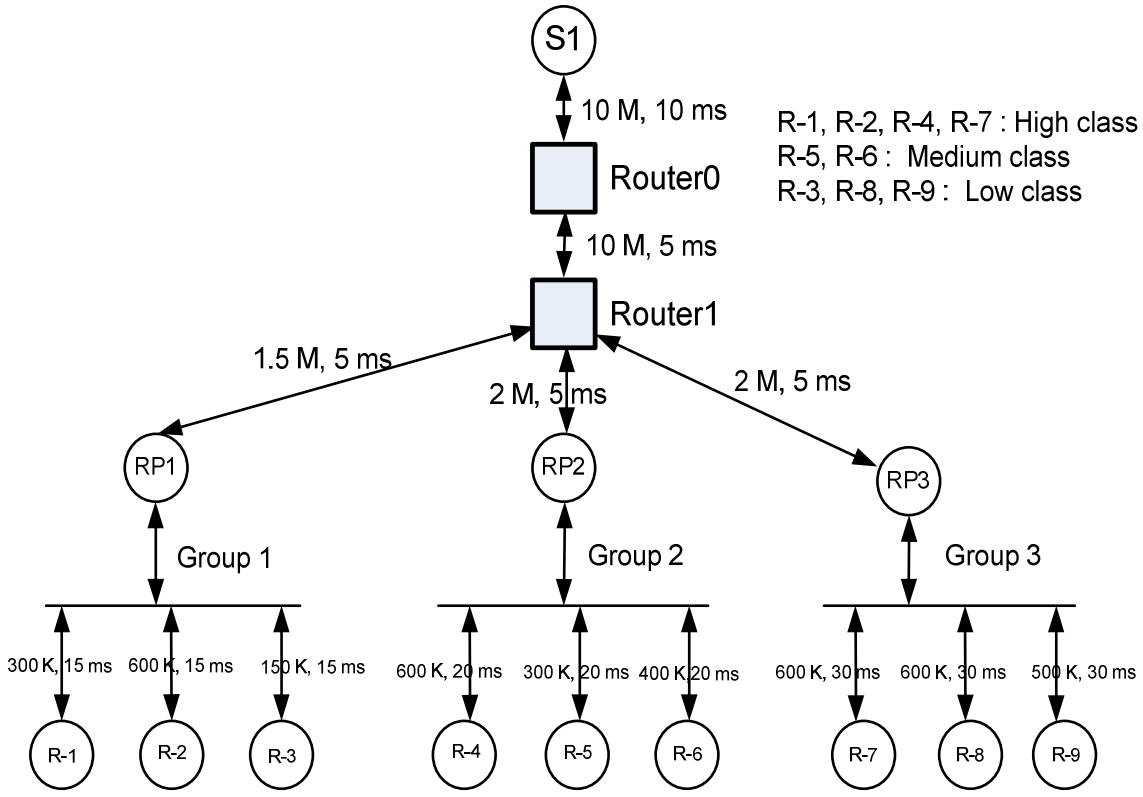


Figure 3.5 Simulation topology

Figure 3.6 shows the average received bit rate for three different receivers' capability (R-1 high receiver starting from group 1, R-5 medium receiver capacity starting from group 2, and R-9 low receiver starting from group 3). We average the received bit rate value at the receiver side between any two consecutive intervals. The receivers dynamic movement from multicast group to another can be noticed through the variations in the average received bit rate per receiver. We noticed that the higher capacity receiver tries to achieve better video quality by joining higher rate multicast stream rate. For example, approximately at time 100 sec. the high capacity receiver left group 1 and joined group 2. Then after 50 sec. the high capacity receiver left group 2 and joined group 3 which support better video quality. The receiver movement continues till it reaches its steady

state (the best video quality can be achieved). The high receiver capacity may go from group 3 to group 2 if the background traffic affected badly the link between the router and the group leader of group 3 as shown at 300 sec and Figure 3.7. The medium capacity receiver moves between group 2 and group 1 based on its capability and the network status. The low capacity receiver received low video quality (low bit rate) but generally acceptable.

In Figure 3.7 we have plotted the background traffic versus the simulation time for the selected receivers R-1, R-5, and R-9. Background traffic is generated from the source to the selected receivers. We have used CBR as background traffic.

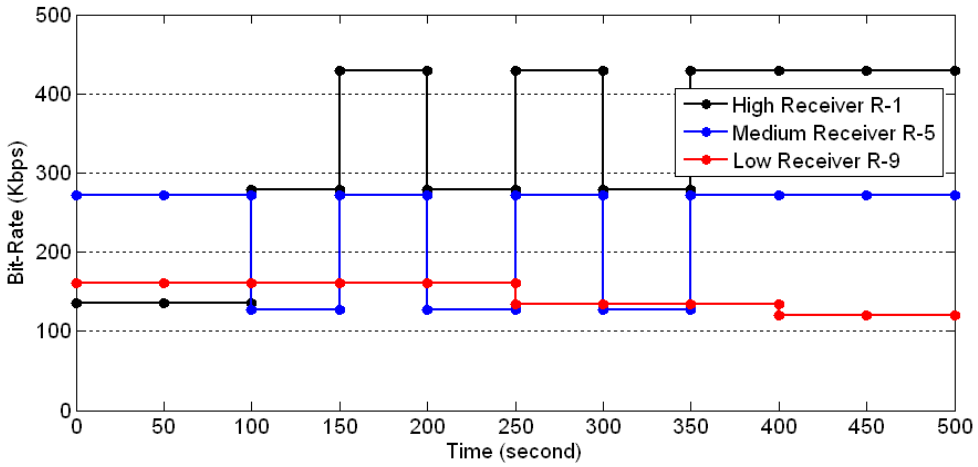


Figure 3.6 Bit-rate vs. simulation time

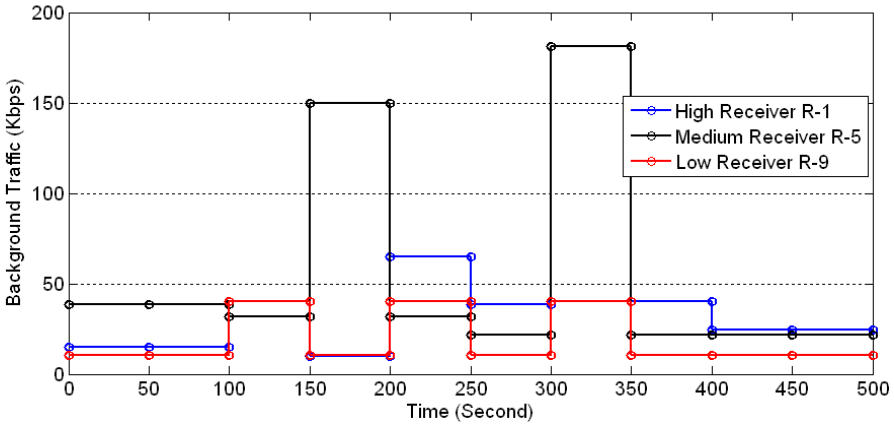


Figure 3.7 Background traffic vs. simulation time

Figure 3.8 depicts the relationship between the numbers of receivers per each multicast group vs. the simulation time. It is obvious that the receiver switching over the multicast group happens at the beginning of the time simulation then the number of receivers per multicast group became approximately fixed.

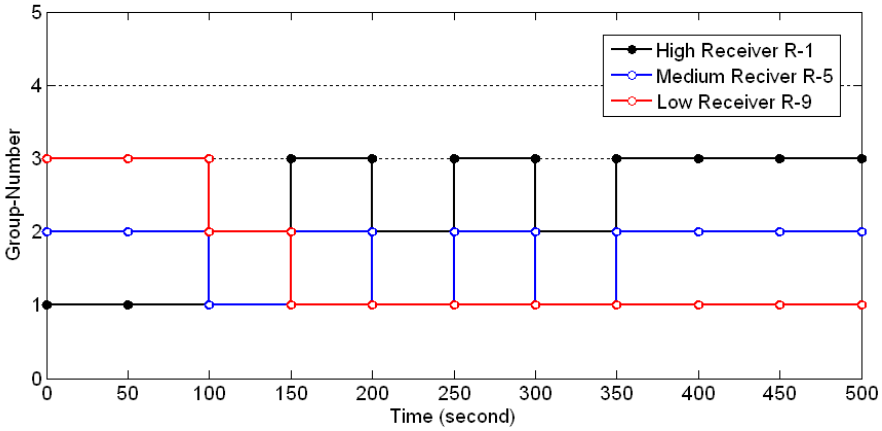


Figure 3.8 Receivers group vs. simulation time

In an attempt to prove the proposed algorithm in wireless environment, we simulate the algorithm in ad-hoc environment using multicast ad-hoc on-demand distance vector

(MAODV) as the underlying ad-hoc multicast routing protocol [32]. The basic idea is to generate the video with multiple streams each with different bit-rate and send each stream over a multicast tree. These trees are ideally disjoint with each other to increase robustness to errors and other transmission degradations. We show that for a given level of tree connectivity, all the members in the same multicast tree can receive the video with acceptable video quality based on the tree bit-rate. Thus building multiple trees do not increase the cost of network overhead significantly. We present simulation results for a 3 group leaders and 9 receivers node with low mobility to show the affect of the algorithm, and with network size of 1300 X 500 m. Three multicast trees have been constructed. Each multicast tree is provided with video bit-rate R_i $(64, 150, 210\,Kbps)$. Figure 3.9 shows the average bit-rate received for each receiver versus the simulation time. The Figure clearly shows that the receivers can join/leave a multicasting group based on its buffer capability and the wireless link bandwidth leading to it.

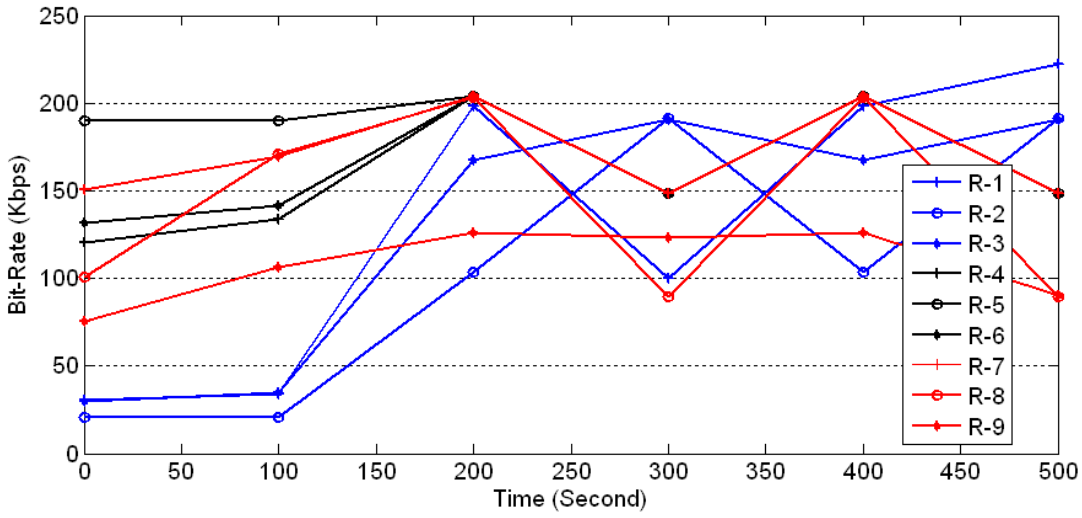


Figure 3.9 Bit-rate for receiver vs. simulation time

3.5. Conclusions

In this Chapter, we proposed a novel mechanism for real-time video transmission in multicast environment. The receivers make autonomous joining/leaving to the multicast groups based on its buffer status conditions and the network status conditions. The architecture is “closed loop” for multicast system with the joint control from both the receiver side and the group leader. By implementing timers at the receiver side for the switching up and down strategies, the techniques provide robustness to the effect of oscillations at the video quality. Furthermore, while the proposed algorithm is cast here in wired multicast environment, the algorithm is acceptable at the wireless networks.

The proposed algorithm resides above the routing layer, and it can be added as a module over any multicast routing protocol. Simulations have shown that the system has better adaptive video quality using switching techniques.

Chapter 4

Media-Aware Caching Mechanism in DiffServ Networks

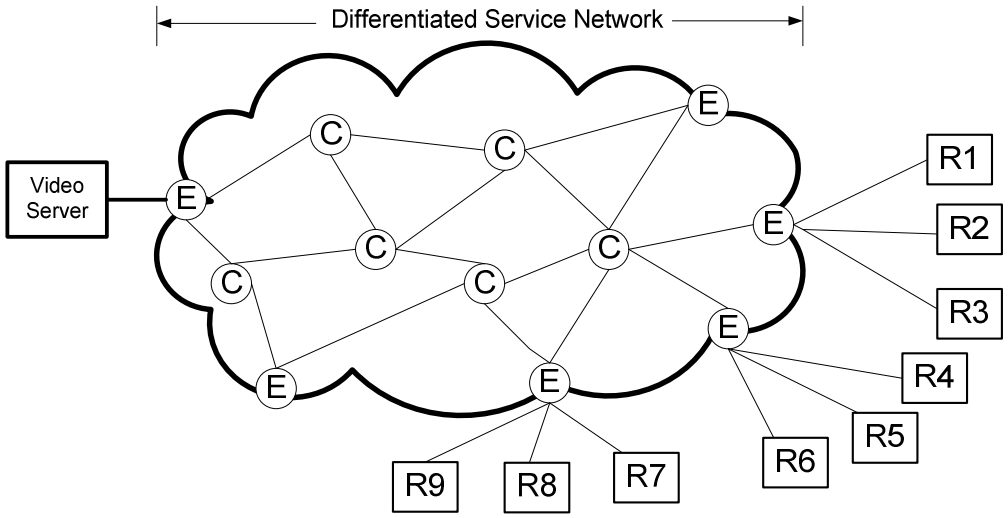
Video server typically can serve only very limited number of concurrent video streams content. This problem has known as server or network input/output bottleneck, limits the scalability and quality of services for video transmission. In this Chapter, we tackle this problem in differentiated services (DiffServ) networks in a multicast environment, [33]-[36]. Real-time applications desire high quality of Service (QoS) support from the underlying network. The DiffServ approach provides many benefits for these real time applications. The DiffServ scheme provides a less complicated and scalable solution when compared to the Integration Services (IntServ) approach.

The rest of this Chapter is organized as follows. In Section 4.1, we describe the high level architecture design of the proposed technique. In Section 4.2, we briefly provide an overview of the previous related work. We introduce the system architecture and a basic description of the edge routers extended functionality in Section 4.3. Section 4.4 is devoted for the network simulation, and study the performance of the proposed solution by investigating the average packet delay and the throughput. Finally, we present the conclusions in Section 4.5.

4.1. Introduction

Common video encoding standards like MPEG or H.263 are quite similar concerning their means of data compression to reduce the temporal and statistical redundancy between the video frames. Although motion compensated coding can achieve high compression efficiency, it is not designed for transmission over lossy channel. In this scheme the video sequence is composed of two main types of frames; intra frames (I-frames) and inter frames (P or B frames). I-frame is a single frame of digital content that

independent of the frames that precede and follow it and stores all of the data needed to display that frame. The more I-frames that are contained, the better quality the video will be; however, I-frames contain the most amount of bits and therefore take up more space on the storage medium. P-frame is encoded through motion estimation using preceding I- or P-frame as a reference frame. B-frames contain only the data that have changed from the preceding frame or are different from the data in the very next frame. The pixel value differences between the original inter-frame and its motion predicted are encoded along with the motion vector. This poses a severe problem, namely error propagation where errors in reference frames due to packet lost propagate to all the dependent frames leading to visual artifacts. Much of recent work in video streaming focused on the error propagation problem in [38]-[41]. With the multicast in DiffServ network, shown in Figure 4.1, delay/loss sensitive media applications can be built much easier and their scalable realization will become less painful. The integration between the DiffServ and multimedia applications is investigated in [38]-[41]. The unequal error protection (UEP) has been proposed by presenting relative priority score based on the video frames importance in [23].



C: core router, E: edge router, and R_i: Client i

Figure 4.1 Typical setting of DiffServ network

We extend the edge router functionality to support high QoS for real-time applications. Figure 4.1 shows the deployment of the media aware sub-layer above the IP network layer of the edge router. The edge router works as relaying node. Basically it checks the high priority video frames for each requested stream and caches them in the cache memory if there are available memory spaces, otherwise runs the cache replacement algorithm. The expired video frames will be removed from the cache memory when there are new frames that need to be cached and no available memory spaces. This process will be executed by the cache replacement algorithm. We will evaluate the performance of the extended edge routers versus traditional edge routers in differentiated services networks by measuring the average packet delay, and throughput.

The strength of our mechanism has come from the following three contributions:

1. Extending the functionality of the edge routers to support video caching which reduces the delay required to retransmit lost packets.

2. Content aware caching where we cache high priority video frames, i.e., I-frames,
3. We treat video differently in the edge and core routers based on their importance, which improves the video quality at the receiver side.

4.2. Related Work

The MPEG video international standard specifies the coded representation of the video data. To achieve high video quality and meet the coding requirement, three different frames have been defined (I-B-P frames).The organization of the three types in a sequence is done by the group of picture (GOP) concept [43]. Resilient to high priority (I-frames) packet loss is a crucial requirement to improve the video quality since the error propagation problem decreases the video quality and causes severe problems in reconstructing the video stream at the end user.

In [38]-[42], feedback control channel is enabled to receive the negative acknowledgement (NACK) packet loss sequence number and packet retransmission is applied. However the error propagation problem is partially solved, the packet retransmission has to traverse the whole round trip between the source and destination in a best effort networks which may lead to packet loss and as a result the average packet delay increases dramatically.

In [44], an adaptive key-frame reference picture selection (KAPS) algorithm for video transmission has been proposed. The KAPS video frame selection is fixed in advance. All the non keys inter coding frames select their references in the conventional way. The KAPS tries to switch between itself and the other coding techniques which increase the coding complexity.

In [33]-[35] and [45], DiffServ model with centralized management entity (tree manager) has been proposed. This entity must have full details information about all the nodes member and resources available in order to maintain admission control. Its complexity becomes severe when it treats the receiver's join/leave messages for all the group members. In [46], the authors used encapsulation based approach. This mechanism does not require all the state information as in [44], however edge routers have to keep detailed information about the multicast tree. In [47], probe based technique is implemented to update the edge router about the multicast status and whether they are congested or not.

In [48] explicit congestion notification is used to dynamically regulate real time sessions to meet the network bandwidth availability. An important advantage of this work is that the intermediate nodes do not keep any previous information. A rate control mechanism uses per hop MAC layer measurement from packet the transmissions as a feedback; while source based admission control is used for the aggregate real-time traffic.

4.3. System Architecture

4.3.1. Objective

In this Chapter, we enhance the caching ability over DiffServ networks by storing the high priority video frames instead of the whole stream, [37]. In addition, caching at the edge routers decreases the average packet delay without excessive reservation of the storage space. Caching only the high priority video frames in the edge routers increases the number of caches demand files and reduces the retransmission from the video server.

4.3.2. Network Architecure

In a typical setting, Figure 4.1 and Figure 4.2 depict our architecture, portraying video multicast server and multicast groups. In traditional DiffServ, if a video packet has been lost at the client, it sends NACK to the multicast video server. This increases the load on the video server, and dramatically increases the average packet delay. To improve the QoS support in such environment, the sender application assigns priority levels, which are based on the frame's importance to the reconstructed video quality. We consider the GOP structure with one I-frame followed by several P-frames and B-frame in this Chapter, I frames can assigned higher reliability level than B and P frames. The P-frames that are closer to the preceding I-frames are more valuable and gives higher probability than the other P-frames and/or the B-frames. The video frames will be treated differently at the core routers and the edge routers based on its importance for reconstructed the video frames.

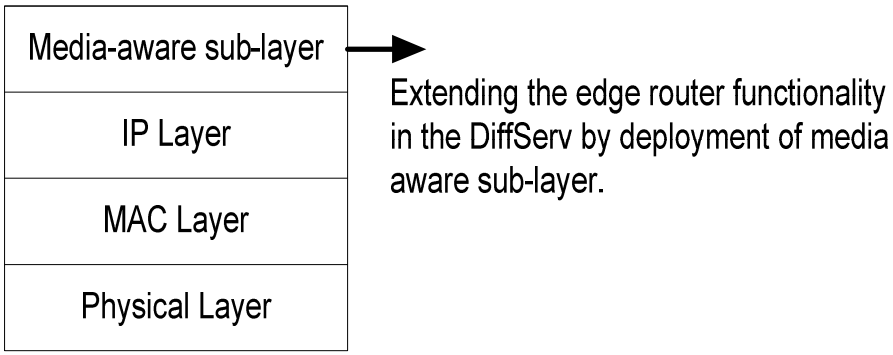


Figure 4.2 Structure of edge router layers in DiffServ networks.

4.3.3. Extending Edge Router Functionality

When a new packet arrives from the source, high priority frames are cached in the cache memory along with its lifetime. Once the lifetime is expired the packet is erased from the cache memory. The new packet is then forwarded to the receiver. If a NAK

arrives from the receiver, the edge router processes this NAK packet and performs a lookup in its cache memory for a correct copy of the corrupted packet. The retransmission methodology flowchart is explained in Figure 4.3. The high priority video frames are cached at the edge routers for its life time. A copy of each high priority packet is cached in the retransmission buffer, along with its life time. All the packets in the same frame have the same priority and caching life time. The clients play the initial part of a video clip when the quality degrades under the acceptable threshold. The client will check the packet loss if packet sequence number is missing, it will send retransmission request NACK to the edge router. The media aware sub-layer receives the retransmission request packet, and precedes it by looking for the lost packet at retransmission cache buffer. The sub-layer may retransmit the packet if it is not expired from the edge router cache memory. Once the packet life time is expired, there is no need to retransmit the lost packet again or cache it in the cache memory. All the packets inside the same frame will have the same life time. The frames life time is calculated as follows:

$$T_L(N) = T_R(N) + D_S$$

Where $T_R(N)$ an estimate for the rendering time of frame N at the receiver, and D_S is a slack term to compensate the in accuracies in estimating the one way delay (OWD) from the edge router to the client. The receiver processing delay is neglected. The rendering time is calculated as follows;

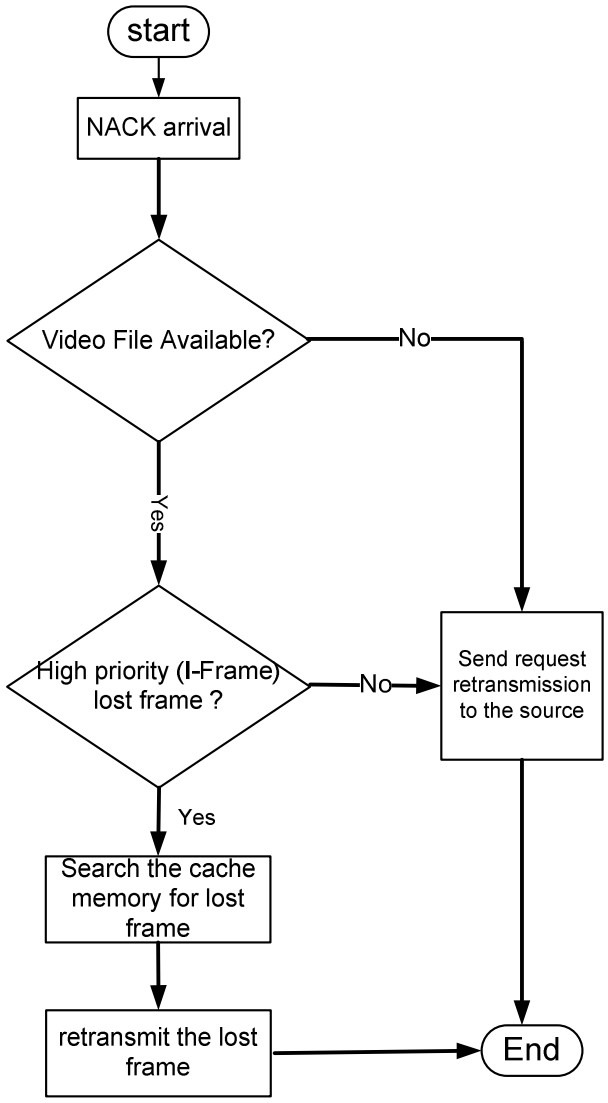


Figure 4.3 Video retransmission algorithm at the edge router

$$T_R(N) = T_o + T_D + N / R \tag{4.1}$$

Where T_o is the video session initiation time, T_D is the receiver play out delay, and R is the frame number.

Due to caching placement at the edge routers, the round trip time is decreased by:

$$rtt = \sum_{i=1}^{n} (d_i^b + d_i^f) \tag{4.2}$$

Where d_i^b, d_i^f the backward and forward delay between hop i and hop i-1, respectively. *n* is the number of hops from caching node to the video server. As traversing fewer hops the packet gains higher packet successful probability which yield higher throughout and less packet loss. Let's denote the length of video packet and length of NACK by L_v , L_{nack}, respectively. Assuming forward bit error rate and backward bit error rate from hop i to hop i-1 are p_i^f , p_i^r . The probability of unsuccessful retransmission can be calculated as follows;

$$p = 1 - \prod_{i=1}^{h} (1 - p_i^f)^{L_v} \times (1 - p_i^b)^{L_{nack}} \tag{4.3}$$

Where h is the number of hops between the client and the caching node (edge router). Finally the probability that retransmission packet is successfully delivered between the client and the edge node after q attempts can be calculated as follows.

$$k(q) = p^{q-1} \times (1 - p) \tag{4.4}$$

The retransmission success probability is used in the simulation model in defining the link packet loss probability.

4.3.4. Caching Replacement Algorithm

When the cache memory is full, the unused data need to be replaced with the new arriving data. The replacement algorithms of the traditional video caching scheme focus on caching the whole video streams, however they are not desirable for long continuous media stream. If we cache the entire video stream, the cache space is not enough for

variety video over the internet. Although the media aware sub-layer technique is profitable caching method that only cache the high priority video frames, the limited cache size of the edge nodes must be adopted. Cache replacement algorithm will have the responsibility to evict data from cache memory when a new data arrived and the cache memory is full. The main cache replacement is least recently used (LRU) [49] that removes the least recently used items first. This algorithm needs to keep monitoring of all data. However, some researchers [50] enhanced the LRU by considering some other parameters such as data invalidation rate, data size, transfer time, etc. However getting such input parameters from the system that is changing and not easy to estimate is difficult task. In this Chapter, we focus on usage of the frame life time, as given by the previous equations in Subsection 4.3.3. The sender application shall calculate the frame life time based on the rendering time of a frame at the receiver, the one way delay (OWD) from the edge router to the client, video session initiation time, and the receiver play out delay. A logical list is maintained for each video stream, storing the high priority video frames. As explained in Figure 4.3 the new high priority video frames are cached intro warehouse and wait for being used in case of packet loss. If a frame arrives and there is enough memory space, no cache replacement is needed. Otherwise, the logical list shall start to remove the oldest high priority video frames and check the required cache space. If the required cache space is enough to cache the new arrived video frames, new cache process will be done. Otherwise, cache replacement algorithm will be re-executed until it succeeds to free enough memory cache slots for the new high video frames. Remaining buffer size can be calculated as follows;

$$\text{Re}\,maing\ buffer\ size = B - \sum_{j=1}^{m}\sum_{i=1}^{n} f_{ij} \quad (4.5)$$

Where B is the edge node buffer size, m is the number of simultaneous video streams transmissions, and n is the number of the current cache video frames.

The pseudo code for the cache replacement algorithm is given as below:

```
1. Notations:
2. B: buffer size.
3. Sq: cached set frames size at time t, where q is the number of cached frames.
4. Frame (k): arrives at time t.
5. Procedure cache replacement policy ()
6. Free Buffer_Space = B – Sq
7. If (Free_Buffer_Space > = Frame (k))
8.      Accept frame (k);
9. Else
10. {
11. Calculate the life time for cached frames set Sq;
12. Create a list of the cached frames Sq;
13. Free_Buffer_Space = 0;
14. Drop_Set_List=0;
15. Buffer_occupance = Frame (k) + cached frames Sq;
16. While (Buffer_occupance > B)
17. {
18.   Drop_set = Drop_Set_List ∪ pick (oldest frame life time)
19.   If (frame (K) ∈  Drop_Set_List)
20.      Break;
21. }
22. If (k ∈ Drop Set)
23. {
24. Drop_set = 0;
25. Drop frame k;
26. }
27. Accept frame (k);
28. Drop all frames in Drop_Set;
29. }
```

4.4. Simulation Results

In this section, we implement our proposed architecture. The overall simulation setup for the proposed video aware sub-layer and DiffServ interaction is implemented by opnet simulation network [18]. We evaluate the performance of our caching mechanism and retransmission of the high priority video frames in comparison to the existing technique. A simple differentiated services network is shown in Figure 4.4. We simulate a dynamic network conditions by different packets drops and generating several background traffic flows on the links between the edge router and clients. We first discuss some important parameters and aspects in our implementation and then analyze the results obtained from the simulation.

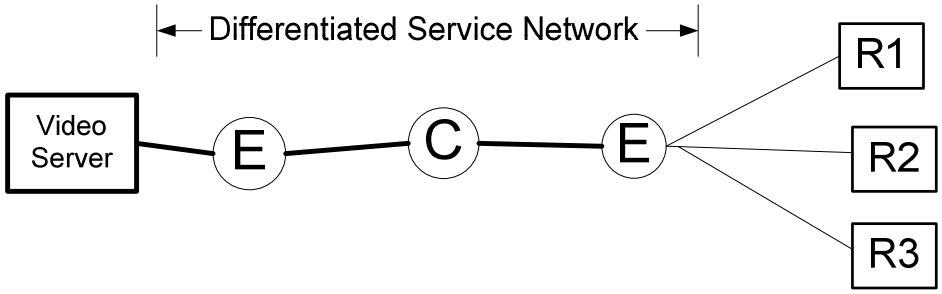


Figure 4.4 Network topology

The link lies between the core router and the edge router, with a 2 Mbps bandwidth. All the other links are with a 1 Mbps bandwidth. The playback deadline is 200 ms. Packets will be dropped randomly at the network links, and the averaged packet loss rate ranging from 0 to 15%. We set the packet size to 500 bytes for all the video frames. The video sequences used in our simulation followed MPEG-2. Sixty seconds of high motion video sequence are encoded at 15 frames per second (fps), which results in a sequence of 900 frames. There are 12 frames in a GOP, with the structure that one I frame followed by eleven P frames. The frame resolution is quarter common intermediate format (QCIF,

176 X 144 pixels/frame), which is the most common format at low bit rates. The coding rate is 200 Kb/s.

Figure 4.5 depicts the relationship between the average packet delay (sec) against the time (sec). The Figure shows that caching the video frames decreases the round trip delay between the clients and the video source. As caching the high priority video frames at the last edge router, the probability of the packet loss is decreased, and the number of hops between the clients and the caching node (edge routers) is decreased as well.

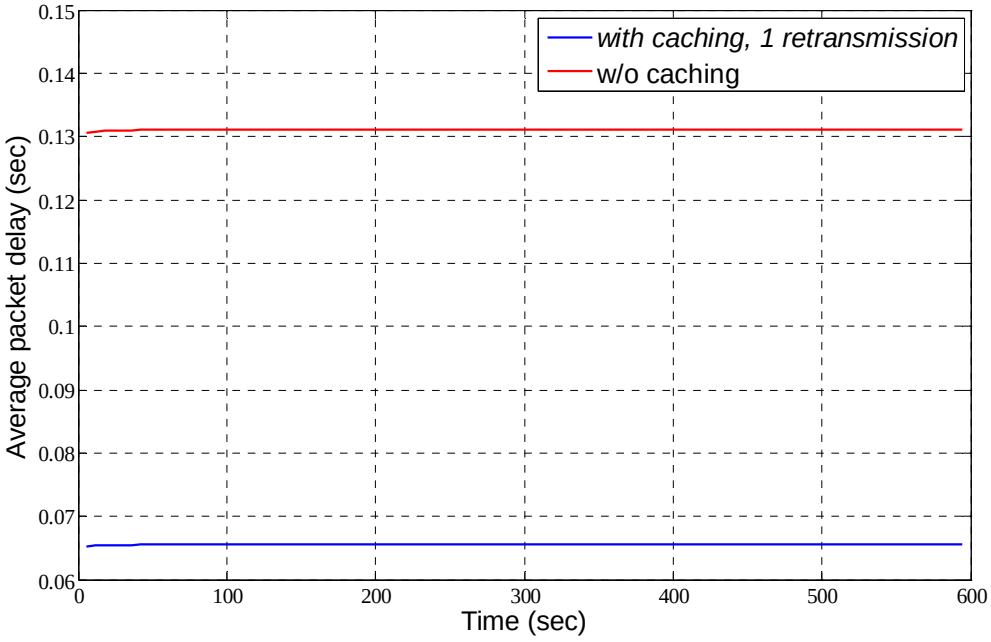


Figure 4.5 The average packet delay versus the time

Figure 4.6 shows the average throughput (Packets/sec) versus the time (sec) as we change two parameters namely, the number of retransmission requests from the clients and the packet loss at the last hop before reaching the clients. The results show that in the proposed mechanism the throughput is higher than the existing technique. Caching the

high priority video frames at the edge router will contribute in decreasing the packet loss at the client and also decreasing the number of trips between the video server and the clients.

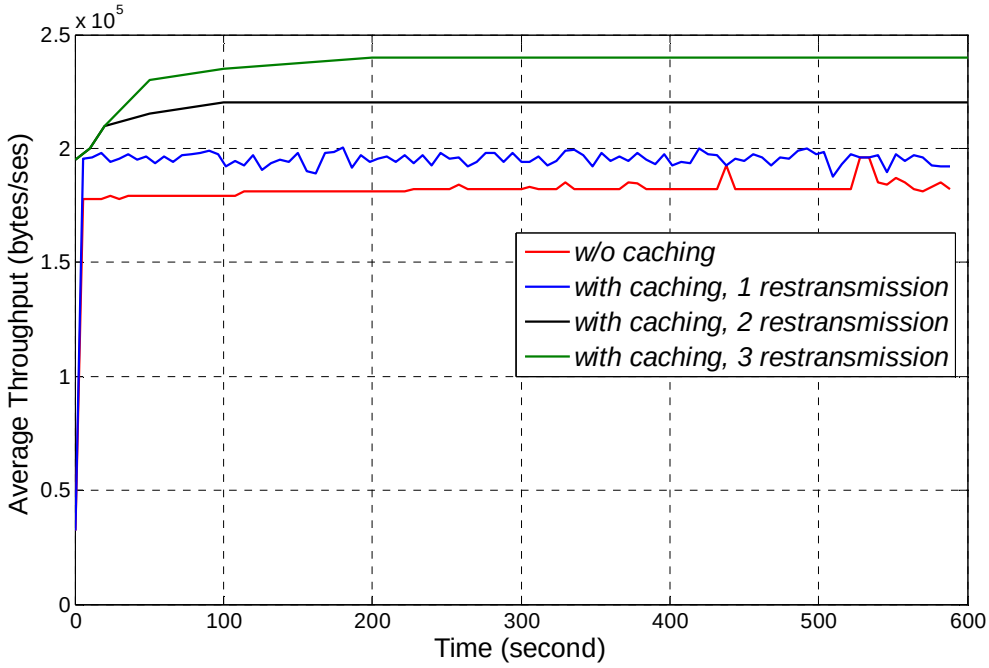


Figure 4.6 Throughput versus the time

We tested the proposed technique with different number of retransmission attempts from the receiver. As we can from Figure 4.7 that with a single retransmission attempts, the average packet delay will still high, due to failure of getting the lost packet from the edge node which leads to request the packet from the original source node. With increasing the number of requests from the clients in case of redundant packet loss, the probability that the retransmitted packets will be received before their deadline increases. Also it can be seen that there is slight enhancement of the average packet delay as we increase the

number of retransmission attempts. This suggests that the receiver may limit the number of retransmission to three, which limits the overhead of the mechanism.

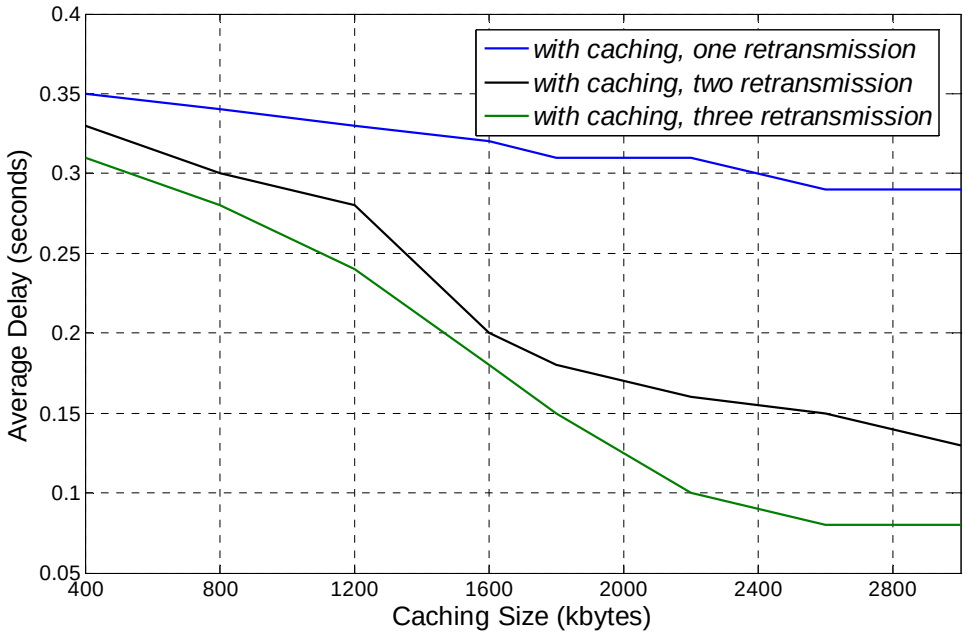


Figure 4.7 Average packet delay versus caching size

Figure 4.8 shows the packet lost versus the caching size for different retransmission attempts. As was shown before, increasing the caching memory size will enhance the mechanism performance by decreasing the packet lost. It can be seen from the Figure that packet will decrease as we increase the number of the retransmission attempts. The reason behind this that increasing the retransmission attempts is adaptive, in the sense that it only adds the retransmission overhead when there is loss in the video stream. Although that there is control packets between the edge node and the client, they have less contribution to the overhead as they are small in size compared to the video stream packets.

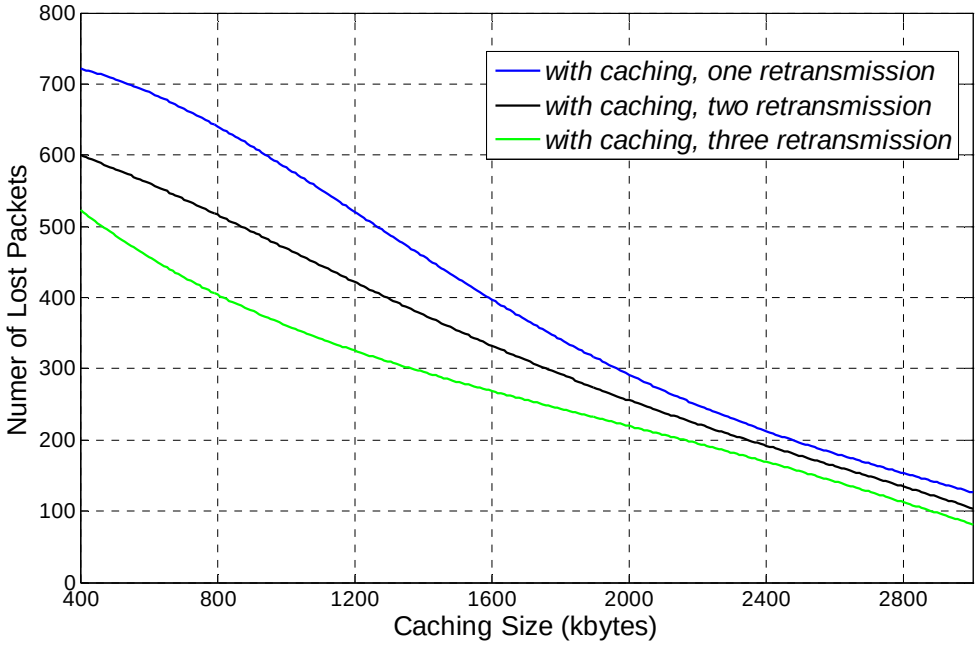


Figure 4.8 Number of lost packets versus caching size

4.5. Conclusions

Video transmission on the Internet generally has very high bandwidth requirements and yet is often unresponsive to packet loss and network congestion. In order to avoid the round trip delay and improve video quality at the clients, we propose a media aware sub-layer that can be added to the edge routers. To reduce storage capacity requirement, video frames with high priority will be selected to cache at the edge router which increases the on-demand video files caching. We have evaluated our mechanism by adding the media aware sub-layer to edge router wherein DiffServ network is enabled. The proposed mechanism decreases the average packet delay and increases the throughput which yields to better video quality at the clients. Furthermore, while the proposed algorithm is cast here in wired DiffServ multicast environment, the algorithm is acceptable at the wireless networks.

As a future work, we plan to implement and examine the proposed mechanism in ad-hoc testbed networks. Also we plan to investigate the cache replacement for optimizing the cache memory size at the edge routers.

Chapter 5

Collaborative Multimedia Content Caching Algorithms for Mobile Ad-hoc Networks

5.1. Introduction

In this study, we focus on video caching as an attractive technique to efficiently tackle MANET challenges. Video caching has been well accepted as viable method to ease the ever growing bandwidth needs, improve the speed of video delivery, reduce the server load, and decrease the client access latency. Three main direct approaches have been proposed in the literatures, namely; simple video server caching, video servers' replication onto all network nodes, and distributed video cooperative nodes. The design of a cooperative environment where a number of cooperative caching nodes are working together to serve a set of clients is considered in our work. The cooperative scheme helps to cache more video streams without overloading the video server. The architecture

builds a better system in terms of fault tolerance as caching nodes can share the burden when certain caching intermediate node is unavailable. Our work is different from the previous work in many aspects; first we are carefully selecting the nodes to cache the video based on the links reliability, the nodes resources, and the availability to serve other nodes in the network. Second, we develop caching placement mechanism in an attempt to achieve high network performance by distributing the video streams among the virtual backbone nodes. The virtual backbone nodes have the function to manage the clients request in such a way to decrease the server overload, to minimize the protocol control overhead, and not to consume the nodes' resources as well. Our main idea is to distribute the video streams among the virtual backbone nodes based on the nodes location. The virtual backbone nodes create Network Information Table (*NIT*) that carries records of the caching video segments at each node and the life time of the video segments as well. Each virtual backbone node maintains the *NIT* updated by adding a new tuple if placement or replacement is performed. *NIT* contains the following fields {cached video stream segment id, cached node id, segment life time}. We design a caching control messages to exchange the *NIT* periodically.

We investigate the data retrieval challenge MANETs and propose a scheme, called Virtual Backbone Cooperative (VBC) for video caching. The goal of VBC is to reduce the average access latency, enhance video accessibility, reduce cache discovery overhead and provide better cooperative caching performance in MANETs environment. In VBC, we use a color convention c_i in determining the roles of each node i in the network as shown in Figure 5.1 To enhance the system performance, initially we select the nodes that will cache the video streams (i.e. black virtual backbone nodes). This selection

algorithm is developed in the entire network nodes, and the decision is taken locally to maintain low control packets overhead. Virtual overlay links are constructed between the virtual backbone nodes. To further improve the efficiency of VBC caching mechanism, a cooperation management scheme including cache placement, a replacement policy called Least Video Segments Life Time (LVS-LT) has also been developed. Our LVS-LT scheme determines cache replacement not only favorable to a client but also important to other newly incoming clients. We also perform optimization study for the VBC caching system. Simulation experiments are performed to evaluate the proposed VBC caching scheme and compare it with existing strategies such as Simple Caching and Data Caching Schemes for mobile ad- hoc environment.

This Chapter is organized as follows. Section 5.2 gives a short overview of caching strategies. In Section 5.3, we describe the proposed system architecture and formulate the caching placement and replacement algorithms. Section 5.4 shows the simulation results for our proposed mechanism. Section 5.5, concludes the Chapter and presents some of the future direction for our work.

5.2. Related Work

The problem of data placement and replacement for wired networks has been studied previously [49]-[62]. In wireless cellular networks, a cache management scheme for wireless cellular phones has been proposed in [55], detailing the caching placement performance at the base station for the case of streaming services, [56]. Consistency maintenance functions cost are derived to determine the cost of maintaining data objects locally and globally. Promising cache placement approach is developed and studied in

terms of power efficiency and power consumption. In ad-hoc networks [58], the focus is on a distributed semantic caching scheme to handle location dependent data in mobile environment. When a node requests a query, it processes the query and finds out the results from the appropriate cache. The drawback of such scheme is that the frequency of cache accessing is not taken into consideration. Hara et al proposed more general replication techniques to overcome the previous shortcoming, [59]. Access frequency, neighbors' access frequency, and overall network topology are the main parameters that are used in such scheme for replication data items technique have shown a performance increase. However the limited size of data (few kilo bytes) is restricting the implementation in real-time applications. Another general approach has been proposed in [60]. Sailhan et al [49], proposed cooperative caching technique [62] to increase the data accessibility by peer to peer communication between the mobile terminals when clients are out of the bound of a fixed infrastructure. Cooperative technique that focuses on data dissemination and replacement strategy has not been well explored.

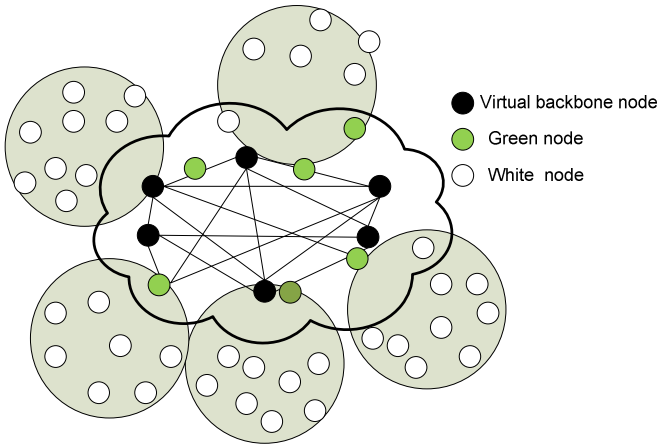


Figure 5.1 Virtual backbone network architecture

5.3. System Architecture

In light of the work done by Kozat [51], we implement our algorithm version to identify the nodes status. We first present Initial selection of backbone nodes, followed by Cache placement policy, and then describe the Cache replacement policy. Detailed explanations for each step are given in the following subsections.

5.3.1. Selecting Virtual Backbone Nodes

In VBC, we use a color convention c_i in determining the roles of each node i in the network. We have three types of nodes (black, green, white). All the nodes initially have white color in their status. All the nodes shall broadcast Hello packets with time to live equal one hop. The hello packet size is 50 bytes to ensure low overhead. In order to build the network information table (*NIT*), the nodes will send and receive the hello packets for certain waiting time T_w. At the end of the waiting period, the white nodes shall check its normalized link failure frequency (*NLFF*) that gives high indication about the links stability of this node as follows;

$$NLFF_j \leq nlff_{th} \tag{5.1}$$

Where $nlff_{th}$ is the normalized link failure frequency threshold, [51]. Note that $NLFF_j$ is simply the proportion of the link losses at node j in a fixed time window. Figure 3 shows simplified version of the virtual backbone formation algorithm [51]. As depicted in Figure 3, a white node j checks whether it has any black neighbors. If there are one or multiple black neighbors, the best black node will selected as virtual access point to node j and the node j will convert its status to green node. Best black node will be calculated based on nodes connectivity, nodes stability, and nodes memory space. Nodes

connectivity is required to have small backbone size which yields to minimize the network overhead. Nodes stability is measured based on the term NLFF. In this simplified algorithm, the best node will be selected among the candidate nodes (i.e. nodes are achieving equation 5.2) based on the highest connectivity. If two nodes have the same connectivity, then the priority for being the best node is given to node that has highest memory, then to the highest node ID. If there is no black neighbor, then the white node will check for white neighbors and choose the best node as a candidate black node. The white node will compare between its node connectivity and the candidate node, if the white node is the best node among its white neighbors, it will converts its status to black node. Otherwise it will convert its status to green node, and chose the candidate node as its virtual access point. The same procedure will be repeated if there are green neighbor nodes.

Assume that there is a set of black nodes $B = \{b_1, b_2, b_3, \mathrm{K}\ b_B\}$ where B is the total number of the black nodes. The nodes that are not joining the black set will check their neighbors' status periodically. The problem of selection *VAP*, is formalized to minimize the cost c_{ij} from node *i* to the selected virtual access point node *j*:

$$\sum_{i \in \{\{N\}-\{B\}\},\ j \in \{b_1, b_2,\ \Lambda\ b_B\}} c_{ij} x_{ij} \tag{5.2}$$

Subject to;

$$\sum x_{ij} \le 1, \quad \forall i, j \tag{5.3}$$

$$x_{ij} = \begin{cases} 1, \textit{if node } i \textit{ selects node } j \textit{ as VAP} \\ 0, \textit{otherwise} \end{cases}$$

We will define the set of green nodes as follows; $G = \{g_1, g_2, g_3, \mathrm{K}\ g_G\}$ The remaining nodes $W = N - (\{B\}\, U\, \{G\})$ will remain white until one of the above two cases is met. If a node is still white and does not have the highest node degree among its neighbors, extension of T_w will be granted to this node. If no node *k* has a link losses $NLFF_j \le nlff_{th}$ then node *j* decides as its $nlff_{th}$ is set to infinity. The algorithm will force the other subnets to continue the process. We refer the reader to [51] for lemmas that show that all the nodes in the network will decide in a finite amount of time its status, and each VAP node has other VAP nodes within 3 hops distance if the network size is large enough. After the virtual backbone phase is carried out, the algorithm will construct virtual links among all the backbone nodes. The goal of selecting the virtual backbone nodes is to obtain relatively stable and reliable virtual backbone networks. The selection algorithm is highly distributed and based on local decisions which make it more reliable to network topology changes. Some conflicts may arise between a node's decision and its neighbors' decision. In these cases, the maintenance phase will be used to resolve such conflicts between the neighbors. Additionally, the maintenance phase will be more recognizable when network topology changes due to nodes mobility. Basically, checking the green and black nodes neighbors is the main criteria to determine the node color status in the maintenance phase. For example if a black node deserted by all its green neighbors, it becomes a while node and entered the virtual backbone selection phase again. Another scenario if a black node has another black node neighbor, the best node will stay black and the other one will change its status to green node. Similar rule has to be applied to the green nodes as well. We refer the reader for more details about the maintenance to reference [51].

Only the black and green nodes are responsible to create overlaying virtual links. Virtual access point (VAP) is a black or green node that serves one or more of its clients' neighbors. The white nodes are the remaining nodes (i.e. non black or green nodes). Video cooperative flowchart is illustrated in Figure 5.3 As client *i* requests video stream, it will check first its cached memory (local caching) if it still cached video segment from the previous video reception. If the client cache memory is empty or does not contain the whole requested video stream, the client has to send media request packet to its VAP_i node. The VAP_i will check its cache memory and send media response packet to the requested client *i*. The media response packet should contain the local cached video segment number. The VAP_i also will check its *NIT* which contains information about the other virtual backbone nodes who are caching the requested video stream. This VAP_i will forward the media request packet to the virtual backbone node(s) that cached part (one segment or more) of the requested video stream. As a result this caching virtual backbone node(s) will transmit the cached video segments to the requested client *i* in sequence based on the client request. If part of the video stream is only stored at main video server, the main video server has the responsibility to send this part. That video segment stream may be cache at one of the virtual backbone nodes during its transmission; in turn the cache replacement policy may run if there is no space to cache the new segment.

The second and third phases are to use cache placement and cache replacement algorithms on the candidates' mobile nodes (i.e. virtual back bone nodes) to ease the client video delivery and increase the video quality.

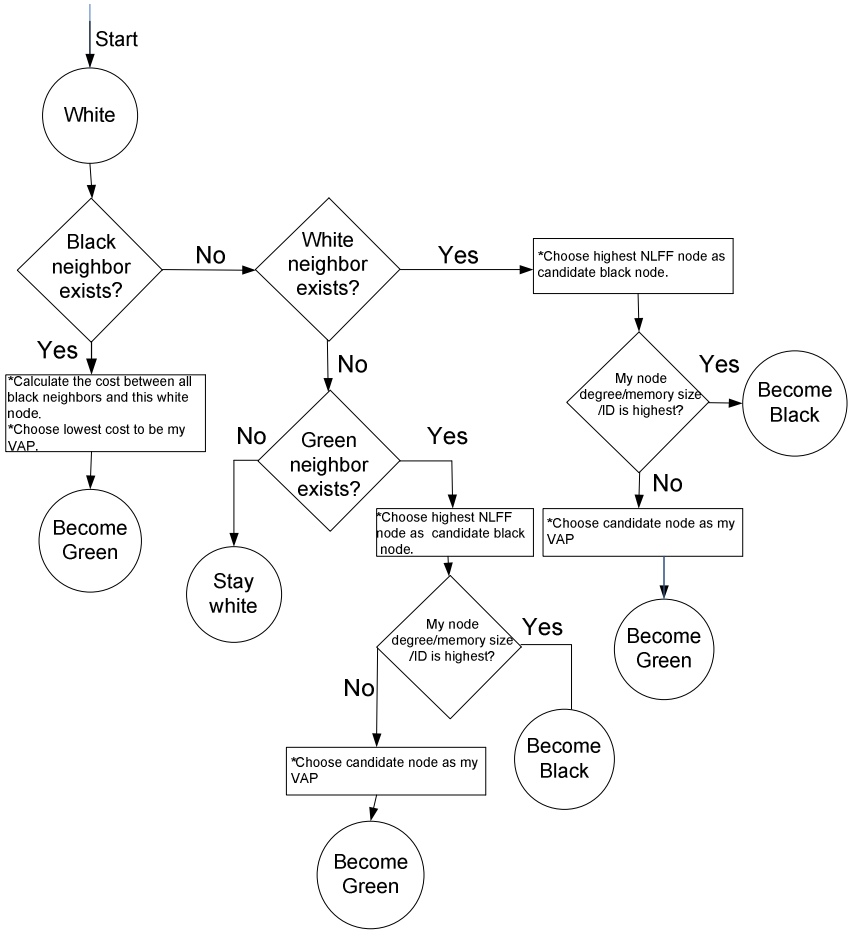


Figure 5.2 Virtual backbone selection flowchart

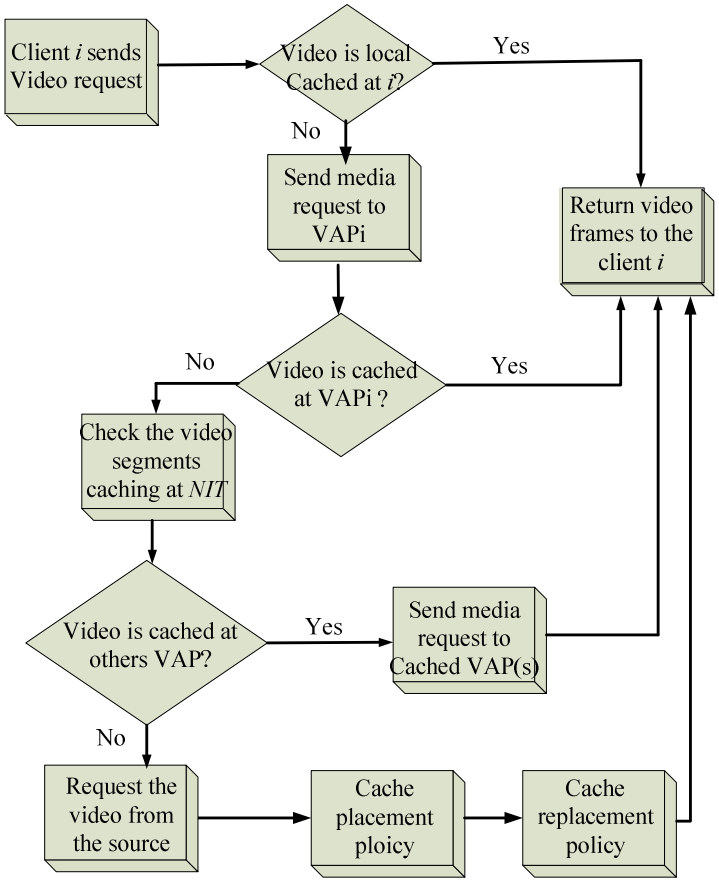


Figure 5.3 Collaborative video caching flowchart

5.3.2. System Model

The system environment is assumed to be an ad-hoc network where ad-hoc clients' nodes try to access video stream. The virtual backbone nodes that hold a copy of the video segments is called data source/server/center. The video stream request initiated by a client is forwarded hop-by-hop along the routing path until it reaches the data source and then the data source sends back all/part of the requested video stream. Each data center maintains local cache in its cache space memory. To reduce the bandwidth consumption and average access latency, the number of hops between the data source/cache and the requester should be as small as possible. Most virtual backbone

nodes do not have sufficient cache storage and hence the caching strategy is to be devised efficiently. Table 5.1 summarizes the different notations and parameters used in this Chapter. Literatures show that the caching placement problem is NP-complete [52], we present a baseline heuristics for VBC technique. We will show how the baseline heuristics can be enhanced to realize dynamic cache reconfiguration with minimum network overhead in the simulation section.

The system has total of *B* virtual backbone nodes, B_j $(1 \leq j \leq B)$.

We combine the cache of all virtual backbone nodes to form the aggregate cache space M^B where;

$$M^B = \sum_{j=1}^{B} M_j^B \tag{5.4}$$

Where M_j^B is the memory cache size of virtual backbone node *j*. We combine the cache of all clients nodes to form the aggregate cache space M^C where;

$$M^C = \sum_{j=1}^{B} \sum_{k=1}^{k_j} M_{j,k}^C \tag{5.5}$$

Where $M_{j,k}^C$ Memory cache size at client *k*, which is served by node *j* as its virtual access point backbone.

In practice, the aggregate cache space combining the virtual backbone nodes and the clients is less than the total volume of the cached video streams. The following equation shows the first constrain in the proposed system, since we don't consider replication on this work. That is the video segment is stored once in the caching nodes:

$$M^C + M^B \leq \sum_{i=1}^{V} \upsilon^i \tag{5.6}$$

Where υ^{j} ($1 \leq i \leq V$) is the video stream *i* volume. The second and third constraints follow the cache space limit of virtual backbone node *j* and client *k* of virtual backbone node *j*, respectively.

$$M_{j}^{B} \geq \sum_{i=1}^{v} \sum_{\lambda=1}^{\lambda_s} P_{j}^{i,\lambda} \cdot s_{\lambda} \tag{5.7}$$

$$M_{j,k}^{C} \geq \sum_{i=1}^{v} \sum_{\lambda=1}^{\lambda_s} q_{j,k}^{i,\lambda} \cdot s_{\lambda} \tag{5.8}$$

Where $P_{j}^{i,\lambda}$ probability of cached video segment $\lambda$of video stream i at virtual backbone node *j* and $q_{j,k}^{i,\lambda}$ *is* Probability of cached video segment $\lambda_{v}^{i}$of video stream *i* at client node *k.*

To compute the number of hops between the client and the data source for specific video segment $\lambda$of video stream i , there are four different cases of data hit that can be combined as follows;

$$\{q_{j,k}^{i,\lambda} \times 0 \; + \; \sum_{j=1}^{B} P_{j}^{i,\lambda} \times \; d_{j,k} + \sum_{\substack{r=1, \\ r \neq j}}^{B} P_{r}^{i,\lambda} \times \; d_{k,r} + P_{S}^{i,\lambda} \times \; d_{k,S}\} \tag{5.9}$$

Where $d_{j,k}$, $d_{k,r}$, and $d_{k,s}$ denote the average number of hops between the client *k* and virtual backbone node *j*, the average number of hops between the client k and virtual backbone node *r*, and the average number of hops between the client *k* and source node *s*. Each video segment is periodically updated at data source. After video segment update, its cached copy (maintained on one or more clients) may become invalid. This is limited to the life time of each video segment.

Table 5.1 System symbols and notations

Symbol	Definition
B	Number of cooperative virtual backbone nodes
K	Total number of clients
k_j	Number of local client attached to virtual backbone node j
υ^i	Video stream $i \in \{1,2,\mathrm{K}\ ,V\}$ where V is the total number of video streams simultaneously across the network
L^i	Length of the video stream I (sec)
R^i	Video stream i bit rate (bit/sec)
M_j^B	Memory cache size of virtual backbone j
$M_{j,k}^C$	Memory cache size at client k, who is using node j as its virtual backbone
M^B	Total memory cache size at all virtual backbone nodes
M^C	Total memory cache size at all client nodes
λ_v^i	Video segment λ of the video stream i, $\lambda \in \{\lambda_1, \lambda_2, \mathrm{K}\ , \lambda_s\}$
S_λ	Video segment λ size (bytes)
δ	Number of hops between two consecutive caching nodes
$P_j^{i,\lambda}$	Probability of cached video segment λ of video stream i at virtual backbone node j
$q_{j,k}^{i,\lambda}$	Probability of cached video segment λ of video stream i at client node k who is using node j as its virtual backbone
$d_{j,S}$	Average number of hops between Source node and virtual backbone node j
$d_{j,k}$	Average number of hops between client node k and virtual backbone node j
d_{j_1,j_2}	Average number of hops between two virtual backbone d_{j_1,j_2}

5.3.3. Cache Placement Policy

In this subsection, we investigate the cache placement problem for virtual backbone nodes for a given video stream. After the virtual backbone nodes have been selected in the network. The virtual backbone nodes will have the responsibility to cache the video segments. In order to increase the data accessibility and decrease the latency as well, the proposed mechanism will cache video segments as much as possible at the backbone nodes, while trying to eliminate video segments duplications.

Initially, the system has one copy of the video streams at the video server. To request a video stream υ^i , a client k_j can call lookup(υ^i) procedure in its virtual access backbone

node. The call lookup(υ^i) procedure may return several caching places belonging to the virtual backbone networks only if this video stream υ^i has been requested before by any other client. Lookup(υ^i) procedure is explained below.

Lookup(υ^i)

{

1. Retrieve *NIT* of all the virtual backbone nodes for the video stream υ^i from the client's virtual access point VAP_k.
2. Check *NIT* for each video segment λ_v^i for the requested video stream υ^i.
3. If any video segments for video stream υ^i are cached at the VAP_k , transmit it to client *k*.
4. The VAP_k will start to collect the remaining video segments from other virtual backbone nodes, and send it back to the client.
5. If some video segments do not exist in the virtual backbone nodes, the VAP_k will seek it from the main video server, and send it back to the client *k*.

}

Since we eliminate the caching replication, we will assume that the client *k* retrieves the video stream υ^i from the video server in case of first video stream request. While the video stream is downloading from the video server, a backbone node *j* can become a member of this video stream virtual backbone node by caching one or more video segments of υ^i depending on its cache memory size. To cache video segment λ_v^i, virtual backbone node *j* calls the following cache placement procedure *(j,* λ_v^i*)*

Cache (j, λ_v^i*)*

{

1. Retrieve *NIT* of all the virtual backbone nodes from the backbone node *j* for others backbone node.
2. If the video segment λ_v^i for the requested video stream υ^j does not exist in *NIT* and there is enough memory space, the video segment is cached.
3. Otherwise triggering call cache replacement procedure at virtual backbone node *j*.
4. Update the *NIT* with a new record, say (*j*, λ_v^i)
5. Update virtual backbone nodes with the new record entered at *NIT*.
6. Repeat the steps from 2 to 6 on all the virtual backbone nodes that are located in the route between the client and main video source.
7. If the virtual backbone nodes that are located in the route source and client do not have enough space to cache the video stream υ^j, the node *j* will recall other virtual backbone nodes by executing *Select procedure*.

}

Select Procedure (k)

{

1. Find out the virtual backbone nodes from *NIT* that satisfies the following two conditions:
   - δ hops away from the virtual backbone node *j*.
   - Enough memory cache space to cache the new video segments.
2. If such virtual backbone node is found, cache the subsequent video segments by moving the new incoming segments.
3. Otherwise, if there is no space available, call *cache replacement procedure.*

4. Update *NIT* with the new record.

}

If the virtual backbone node is located within δ hops from the source node that contains the original video stream, then it does not cache the segments. The primary idea is that in order to increase the video steam accessibility, the mechanism will try to cache as much as possible while trying to minimize the inefficient caching places and the network overhead. There is tradeoff between access latency and data accessibility. If the popular video streams are duplicated a lot, then the average latency to average number of accessibility is reduced because there is high popularity of finding those data items in other virtual backbone nodes. With high duplication, however, the number of distinct data items in the aggregate cache is less. Thus the probability of finding less popular data from other virtual backbone nodes comes low. Although caching popular data aggressively in closer virtual backbone nodes helps in reducing the latency, in this work, we use delta values to enable more distinct data item to be distributed over the entire backbone nodes. We aim in increasing the overall data accessibility.

5.3.4. Cache Replacement Policy

LVS-LT has been developed in the virtual backbone nodes. LVS-LT is taking place when virtual backbone node wants to cache video segment frames, but there is no available slot in the cache memory, and thus it needs to victimize some old video segments. The criteria used for cache replacement in this Chapter, deletes the oldest cache segment which has an expired life time. Cache replacement procedure is shown below:

Cache replacement procedure (k)

{

1. M_j^B Buffer size (bytes) of virtual backbone node *k*
2. *Sq*: cached set segment size (bytes) at time (-*t*), where *q* is the number of cached segments
3. Video Segment λ_v^i with size *B* (bytes) arrives at time *t*
4. Free buffer space = $M_j^B - Sq$
5. If (Free buffer space >= *B*)
6. Accept cache segment (λ)
7. Else
8. {
9. Calculate the life time for cached segments set *Sq*
10. Create list of cached segment *Sq*
11. Free buffer space = 0;
12. Delete set list=0;
13. buffer occupancy = *B* +cached segment set *Sq*
14. While (buffer occupancy > M_j^B)
15. {
16. Delete set = Delete set list (choose oldest expired segment life time)
17. If (segment λ_v^i == Delete set list)
18. {
19. Delete set = 0;
20. Delete λ_v^i; break;
21. }
22. If (buffer occupancy <= M_j^B)
23. Cache λ_v^i cu
24. Don't Cache λ_v^i pa
25. Call *Select Procedure*

}

5.4. Simulation Results

In this section, we will examine the effect of the proposed caching VBC algorithm in ad-hoc networks. To evaluate our proposed framework and compare it with simple caching and Data caching techniques, we have developed a testbed by using OPNET simulation model [18]. For the link layer, the IEEE 802.11 MAC protocol is used. The mobile nodes communicate among themselves over wireless channel using 2 Mbps. The physical layer characteristic is extended rate PHY. The transmission range has been set to 250 meters. The transmission power is 0.005 watt. The simulation network area is assumed of size 1500 X 1500 meter with 250 nodes. The packet is size is 1024 bytes. The video source *S* is representing by video server. We used Xvid video codec [31] to encode MPEG-2 compliant video codec. The frame resolution is quarter common intermediate format (QCIF) with 176×144 pixels/frame, 15 frames per second (fps) video sequence. The coding rate is 200 kb/s. All network nodes are generated background traffic to send to randomly chosen destinations. The background traffic has constant length 512 bits and exponential background inter-arrival time with man equal 0.2 second. The video playback delay is 200 ms to eliminate the jitter in receiving the video packet.

Figure 5.4 depicts the relationship between the average latency (seconds) versus time (seconds) for different schemes, namely; simple caching, Data caching, and co-operative caching. In co-operative caching, a client request can be acknowledged by one or more of the virtual backbone nodes located before the main video server, unlike the simple cache mechanism has to receive the video stream through the video server. The video stream

can be accessed much quicker. It is obvious that the collaborative caching mechanism significantly outperforms the single caching mechanism. This improvement will positively affect the video quality of service at the receiver sides and decrease the number of trips between the client and the video center as well.

Table 5.2 shows the number of hops for the different video caching schemes. The results clearly demonstrate the effectiveness of deployment cooperative video caching technique in ad-hoc environment.

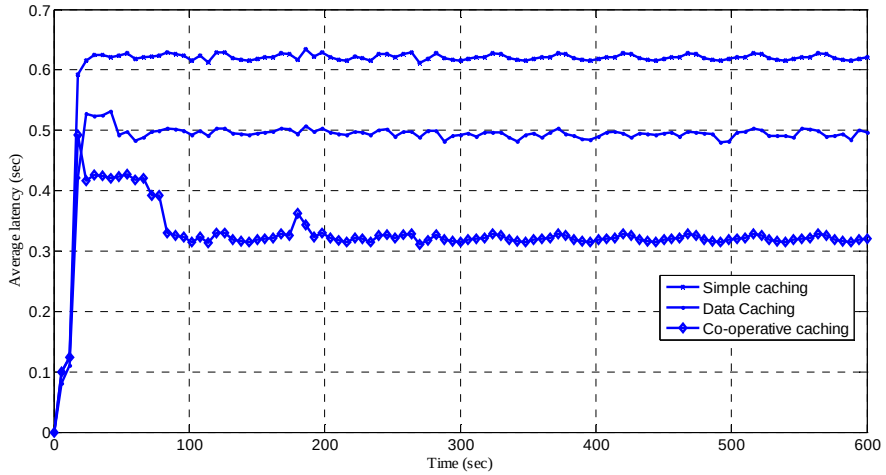


Figure 5.4 Average latency versus time

Table 5.2 Average number of hops under different memory cache size (KB)

Cache size	200KB	400KB	600KB	800KB	1000KB	1200KB
Simple Cache	4.75	4.74	4.7	4.72	4.69	4.67
Data cache	4.22	4.19	4.00	3.92	3.94	3.77
Cooperative cache	3.54	3.43	3.38	3.41	3.3	2.89

In our performance study we will show the effect of caching memory size, number of virtual backbone nodes, effect of inter-distance between two caching nodes, and virtual backbone arrival/departure. As will be shown in the next subsections, comparison

between the co-operative proposed caching technique, simple caching, and Data caching is illustrated.

5.4.1. Effect of Caching Size

Our simulated experiment studies the effect of cache memory size on system performance by varying the virtual backbone cache size from 200KB to 1400KB. Figure 5.5 shows that the average access latency improves, as the cache memory size gets larger. Cache data scheme considers only cache placement policy between the source and destination node. The intermediate node will cache the data when it finds that the data item is popular and it has enough memory space. Only one node will cache the data item. No cooperative protocol among mobile hosts is developed. As a result, the average access latency of the cooperative protocol is less than the others schemes. The virtual backbone nodes can cache more video segments with increasing cache memory size, so it decreases the average access latency.

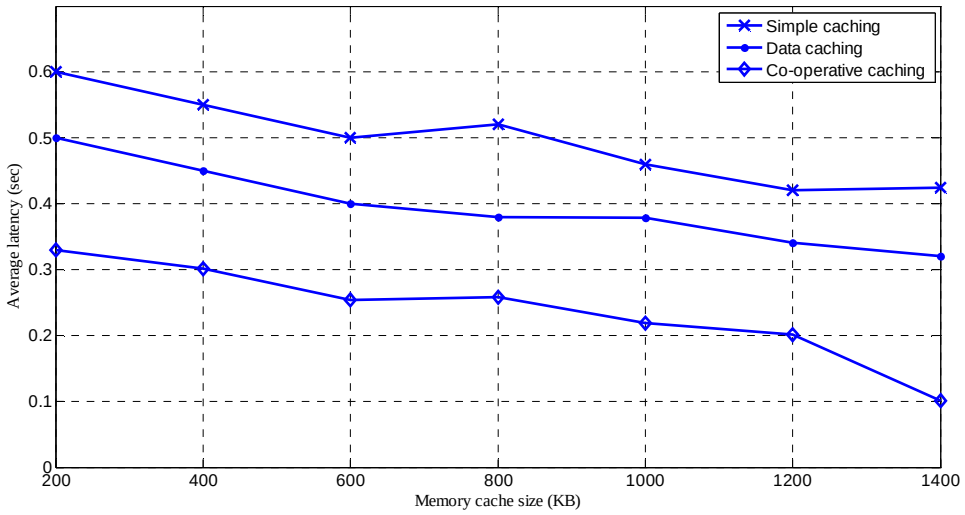


Figure 5.5 Average latency versus the memory cache size

5.4.2. Effect of Number of Mobile Virtual Backbone Nodes

The system scalability is evaluated by increasing the number of virtual backbone nodes from 2 to 12 nodes. Figure 5.6 illustrates that the average packet loss of the proposed mechanism decreases sharply, when there are more than 6 virtual backbone nodes; it also reveals that the system performance can effectively improve by utilizing the virtual backbone nodes in the system. Since the absence of cooperation protocol on Cache Data scheme and cooperative caching scheme, the increasing number of virtual backbone nodes does not lead in very impressive improvement in the packet losses as expected. When two virtual backbone nodes with no or little common access transmission range approach closer together, they may not be able to take advantage of the cooperative cache mechanism from one another; they actually degrade the system performance in terms of network overhead and power consumption. This is because they have to consume more power not only to send/receive control packets from others virtual backbone nodes, but also to discard unintended messages. It can be seen that the proposed technique achieves better performance than the Data and simple caching techniques. Even video packet get lost at the cooperative technique, the cooperative node can retransmits the packet without need to go back the whole way to the video server. In addition to decrease the number of trips between the requested client and the faraway server node will lead to better performance in terms of packet loss and successful receiving packet at the client side. In summary, the collaborative proposed caching technique for multimedia contents improved the video reception significantly and the proposed cache placement and cache replacement algorithms is a viable solution to increase the video quality of service.

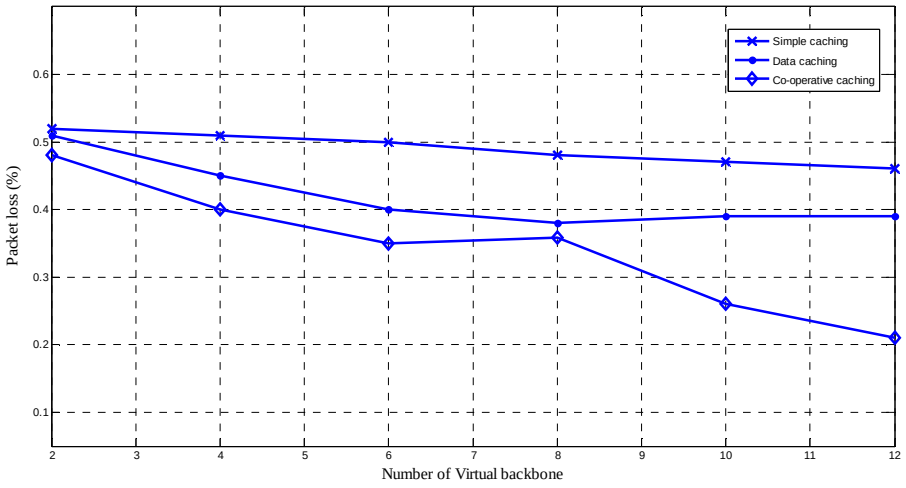


Figure 5.6 Packet loss versus the number of virtual backbone nodes

5.4.3. Effect of Zone (Placement Distance) Size

To evaluate the effect of inter number of hops δ between two consecutive caching virtual backbone nodes on the system performance as depicts in Figure 5.7; we increase the placement distance δ from 1 to 6 hops. When the placement distance is equal to one, the probability of caching the video streams in very dense area will sharply increase. As a result, the number of clients who may be served by those caching virtual backbone nodes will decrease, that may lead to lower system performance. With increasing the caching placement distance between the virtual backbone nodes, the video segments will be distributed across the network. As a result, there is a higher chance for the new clients to obtain their desired video stream from the caching virtual backbone nodes. The caching nodes in Cache Data scheme and simple scheme has a minor effect with the cache placement distance since both of the schemes does not support collaboration protocols between the intermediate caching nodes. It can be seen that there is slight improvement in the system performance as we increase the caching placement distance to greater than 6.

This suggests that the optimum number of hops between the virtual backbone nodes is 4 hops to ensure that the video segments will be distributed across the network. The increasing placement distance causes two effects on system performance. First, the larger inter-distance between virtual backbone nodes leads to higher bandwidth consumption because the virtual backbone nodes have to handle more global cache and discard more unintended messages. Second, it also induces higher network throughput for the new incoming clients, which in turn decrease the average access latency, as shown in Figure 5.7.

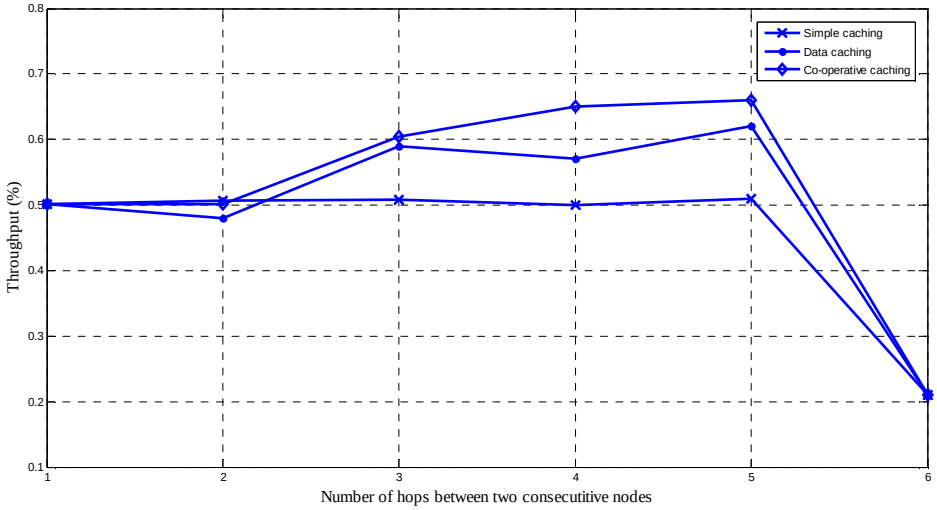


Figure 5.7 Throughput versus the number of hops between two consecutive caching nodes.

5.5. Conclusions

In this Chapter, we proposed collaborative video caching techniques combine virtual backbone selection, cache placement and cache replacement algorithms. The cache placement algorithm ensured that the requested data will be cache at the highest reliable intermediate nodes and it prevents too many replications at the virtual backbone nodes. While the cache replacement technique helps in improving the accessibility and keeps the

video segments at the caching nodes for its life time. The proposed technique resides above the routing layer, and it can be added as a module over any ad-hoc routing protocol. Simulations have shown that the system has better adaptive video quality using the proposed solution.

Chapter 6

Conclusions and Future Works

We conclude this dissertation with a summary of our contribution and directions of the future work.

6.1. Summary

Our goal in this dissertation is to design and implement middleware generic routing components for mobile wireless in unicast and multicast environment, heterogeneous receivers environment, wired and wireless medium. We analyzed the problems posed by the existing routing protocols and solved them using a combination of application layers techniques and modifications and enhancements to the routing protocols. During this dissertation, we build form of a good understanding of the different ad-hoc routing protocols, to be able to compare fairly between them with and without our enhancement middleware generic modules.

Destination-Assisted Routing Enhancement Protocol: We designed and evaluated an extension to DSR protocol to enhance the protocol performance, [62]. We refer to the extension as DSR-DARE protocol. DSR-DARE dynamically generate beacon packets from the destination nodes to announce for its existence. Low beacon packets generation is targeted to ensure low overhead. The new DSR-DARE is effective in providing high throughput, reliability in mobile ad-hoc wireless channel, connection set-up, and routing discovery time. The details of the protocol extension are discussed in details in Chapter 2.

Multicast Multi-Stream Protocol: Multicast Multi-Stream (MMS) address a potentially key that enabling capability for the efficient distribution transmission of the streaming video flow via multiple video sub-streams. The objective of sub-streaming is to employ

rate/quality adaptation control by judiciously regulating the volume application traffic injected into or propagated through the network multicast sessions. MMS module aims to enhance the video quality at the receiver side by dynamically moving the receiver from one multicast group to another based on the receivers' capability and multicast session status. The new module has been integrated with PIM. The new module is effective in providing high video quality of service with minimum overhead. The details of MMS are given in Chapter 3.

Co-operative Video Caching Contents Protocol: We proposed collaborative video caching techniques combine virtual backbone selection [63], cache placement and cache replacement algorithms. The cache placement algorithm ensured that the requested data will be cached at the highest reliable intermediate nodes and it prevents too many replications at the virtual backbone nodes. While the cache replacement technique helps in improving the accessibility and keeps the video segments at the caching nodes for its life time. The proposed technique resides above the routing layer, and it can be added as a module over any ad-hoc routing protocol. The simulation results indicate that the proposed collaborative aggregate cache mechanism can significantly improve the video QoS in terms of packet loss and average packet delay. These details are discussed Chapter 4 and Chapter 5.

6.2. Future Directions

There are several interesting directions for future work based on the work described in this dissertation. Some of these extensions of our work, while some others are

motivated by the more general problem of multimedia over mobile wireless networks are given below.

The algorithms parameters and functions should be fine-tuned and made more adaptive to the network dynamics: We intend to study complex scenarios with more focusing on the performance for large networks with high nodes mobility. Heuristics analysis can be used as input functions to increase the system performance. More parameters such as delay, bandwidth, and packet loss will be used in optimization. In addition, the simulation can be improved by granting different priorities for different nodes.

Cross layer approaches for adaptive error protection for video in mobile ad-hoc networks: Inspired by our research in error protection for ad-hoc wireless, we are planning to study the knowledge of the channel condition retrieved from lower layers. We believe that through sharing the information collected by the medium access control (MAC) and link layers such as signal to noise ratio, link stability, channel quality, and energy level. Then adaptive scheduler can decide the video service so as to meet the channel status. In corporation with the cross layer technique, we believe that our middle ware components will be more efficient.

Cooperative caching on MANETs has the nature of P2P systems: The main problem of P2P systems is that the users may have different incentives and they cannot be trusted to work toward the common benefit of the whole system. Some users may only want to stream video files from others but do not want share their own files. If this situation continues, there will be no user providing shared files in the system. Similarly in cooperative caching, if some caching node refuses to send the request data for other

nodes while keeps asking others for the data it needs, eventually no node can obtain desired data from its peer nodes. To prevent this scenario from happening, we are planning to study fairness between the users and adapt the user upload speed (i.e. bandwidth) based on its contribution in the network. Currently, VBS enables cooperation policy on the virtual back bone nodes, which requires that a node shall answer a peer's request if it has the requested data in the local cache. This policy ensures the minimum level of fairness that no node can deny service to others. Thus, another future work for MANET cooperative caching is to enforce fairness at a higher level between users and all the network nodes, adapt the channel bandwidth based on the user participation in the system.

Bibliography

[1] A. Doria, M. Uden, and D. P Panday, ”Providing connectivity to the saami nomadic community,” in 2nd International Conference on Open Collaborative Design for Sustainable Innovation, Bangalore, India, 2002.

[2] S. Burleigh, A. Hooke, L. Torgerson, K. Fall, V. Cerf, B. Durst, K. Scott, and H. Weiss, ”Delay-Tolerant Networking: An Approach to Interplanetary Internet,” IEEE Communications Magazine, vol. 41, no. 6, pp. 128-137, 2003.

[3] A. Adams, J. Nicholas and W. Siadak, “Protocol independent multicast - dense mode (PIM-DM): Protocol specification (revised),” IETF RFC, No. 3973, January 2005.

[4] P. Mehra and A. Zakhor, ”TCP-Based Video Streaming Using Receiver-Driven Bandwidth Sharing,” Proceedings of the 13th International Packet Video Workshop, France, April 2003.

[5] P. Hsiao, H. Kung, K. Tan, ”Streaming Video over TCP Receiver-based Delay Control”, Proceedings of ACM NOSSDAV, 2001.

[6] D. Saparilla, K. Ross, ”Streaming Stored Continuous Media over Fair-Share Bandwidth,” International Workshop on Network and Operating Systems Support for Digital Audio and Video (NOSSDAV) 2000, Chapel Hill, 2000.

[7] O. H. Hussein, T. N. Saadawi, and M. J. Lee,” Probability routing algorithm for mobile ad-hoc networks,” JSAC. vol. 23, pp. 2248-2259, Dec. 2005.

[8] Y. M. Abdelmalek, O. H. Hussein ,T. Saadawi, ”Simplified algorithm for receiver assisted routing enhancement,” in the Proc. of bio-inspired models of network, Information and Computing Systems, pp. 106-110, Hungary, 2007.

[9] P. l. Yang, C. Tian, Y. Yu, ”Analysis on optimizing model for proactive ad-hoc routing protocol,” IEEE military communication conference, vol. 5, pp. 2960 – 2966, 17-20 Oct. 2005.

[10] C. E. Perkins et al., ” Ad-hoc On-Demand Distance Vector (AODV) Routing,” RFC 3561, http://www.ietf.org/rfc/rfc3561.txt, July 2003.

[11] D. B. Johnson and D. A. Maltz, ”The Dynamic Source routing Protocol for Mobile Ad-hoc Networks (DSR),” IETF Internet draft, http://www.ietf.org/internet-drafts/draft-ietf-manetdsr- 10.txt, 19 July 2004.

[12] P. Samar, M. R. Pearlman, Z. J. Haas,” Independent zone outing: an adaptive hybrid routing framework for ad-hoc wireless networks,” IEEE/ACM Trans. vol. 12, no. 4, pp. 595 – 608, Aug. 2004.

[13] M. Grossglauser and M. Vetterli, ”Locating nodes with EASE: Mobility diffusion of last encounters in ad-hoc networks,” In Proc. of IEEE INFOCOM, April 2003.

[14] H. D. Ferriere, M. Grossglauser, and M. Vetterli, ”Age matters: Efficient route discovery in mobile ad-hoc networks using encounter ages,” Proc. of ACM MobiHoc, June 2003.

[15] S. Lee, B. Bhattacharjee, and S. Banerjee,”Efficient geographic routing in multihop wireless networks,” proc. of MobiHoc, pp. 230-241, ACM press, New York, 2005.

[16] K. Seada, A. Helmy, and R. Govindan, ” On the effect of localization errors on geographic face routing in sensor networks,” in proc. of IPSN, pp. 71-80, ACM press, New York, 2004.

[17] Mathwork Inc, http://www/mathwork.com

[18] Opnet technologies Inc. http://www.opnet.com/

[19] O. Papaemmanouil and U. Cetintemel, “Semantic multicast for content-based stream dissemination,” Proc. WebDB 2004, Paris, France, 17-18 June 2004, pp. 37-42.

[20] S. McCanne, V.Jacobson, and M. Vetterli, “Receiver-driven layered multicast,” in Proc. ACM SIGCOMM, 1996, pp. 117–130.

[21] Legout, “PLM: Fast convergence for cumulative layered multicast tansmission scheme,” in Proc. ACM SIGCOMM, 2000, pp. 13–22.

[22] H. Kanakia, P. Mishra, A. Reibman, “An adaptative congestion control scheme for real-time packet video transport,” SIGCOMM Symposium on communications architechures and protocols, California, Sep. 1993.

[23] X. et al. Li, “Layered video multicast with retranmission (LVMR): Evaluation of error recovery schemes,” in Nossdav, Springer, Berlin, 1997.

[24] Hiroyuki Oouchi, Ken Takahashi, Hiromichi Nagata, and Kouichi Kamasawa, “Multi-rate control method using layered content,” in Proc. Symposioum on Applications and the Internet, 2005.

[25] P. Cuetos, D. Saparilla, K. Ross, “A adaptative streaming of stored video in a TCP- friendly context: multiple versions or multiple layers?,” International packet video workshop, Korea, April, 2001.

[26] Abd El Al, T. Saadawi, M. Lee, “Supporting real-time video in SCTP networks,” in Proc. MILCOM, 2006.

[27] F. L. Leannec, J. Vieron, X. Henocq, and C. Guillemont, “Hyprid sender and receiver driven rate control in multicast layered video transmission,” in Proc. IEEE ICIP, pp. 532-535, Sep. 2000.

[28] Scot Seidel, Tim Krout, and Larry Stotts, “An adaptive broadband mobile Ad-Hoc radio backbone system,” DARBA Netcentric Demonstration – FT Benning, GA, January 2006.

[29] The network simulator (NS). [Online]. Available http://www-mash.cs.berkely.edu/ns/ns.html.

[30] Adams, J. Nicholas, “Protocol Independent Multicast – Dense Mode (PIM-DM),” RFC 3973, January 2006.

[31] Xvid video codec, www.xvid.org

[32] Sun Baolin; Li Layuan, “On the reliability of MAODV in ad-hoc networks,” MAPE, IEEE International Symposium 2005. pp. 1514-1517 Vol. 2

[33] Striegel and G. Maniaran, “A scalable Protocol for Member Join/Leave in DiffServ Multicast,” in proc. IEEE Local Computer Networks (LCN), Florida, USA, Nov. 2001, pp.395-404

[34] A. Striegel, A. Bouabdallah, H. Battahar, and G. Manimaran, “EBM: A New Approach for Scalable DiffServ Multicast,” in proc. NGC, Sep. 2003, pp. 131-142.

[35] R. Bless and K. Wehrle, “Group Communication in Differentiated Services Networks,” in proc IQ workshop at CCGRID 2001, May 2001, pp. 618-625.

[36] Y. Abdelmalek, A. Abd El Al, T. Saadawi, M. Lee, J. Sucec, M. Fecko, “The Join/Leave Policy for Video Multicast Group Members”, in proc. of IEEE Military Communications Conference (MILCOM), Nov. 2007.

[37] Y. M. Abdelmalek, A. Abd El Al, T. Sadaawi, “Media aware caching mechanism over differentiated services networks,” in the Proceeding of the 6th Consumer Communication and Networking Conferences 2009 (CCNC 09), Las Vegas, Nevada, 2009.

[38] A. Abd El Al, T. Saadawi, “A Multi-Path Error Control Mechanism for Interactive Video in Ad-Hoc Networks”, Journal of Ad-Hoc and Sensor Wireless Networks, Aug. 2007, vol. 4, no. 1-2, pp. 125-148.

[39] O. Papaemmanouil and U. Cetintemel, “Semantic Multicast for Content-Based Stream Dissemination,” in proc. WebDB 2004, Paris, France, 17-18 June 2004, pp. 37-42.

[40] S. McCanne, V.Jacobson, and M. Vetterli, “Receiver-Driven Layered Multicast,” in proc. ACM SIGCOMM, 1996, pp. 117–130.

[41] A. Legout, “PLM: Fast Convergence for Cumulative Layered Multicast Transmission Scheme,” in proc. ACM SIGCOMM, 2000, pp. 13–22.

[42] Junu Kim, Jinsung Cho, Heonshik Shin, “Layered resource allocation for video broadcasts over wireless networks,” IEEE Transactions in Consumer Electronics, vol. 54, issue 4, Nov. 2008.

[43] A. Mufit Ferman, S. Krishnamachari, A. Murat Tekalp, M. Abdel-Mottaleb and R. Mehrotra, “Group-of-Frame/Picture Color Histogram Descriptors for Multimedia Applications”, in proc. ICIP 2000.

[44] T. Yang, W. Huang, Q. Peng, and C. Zhu, ”An Adaptive Key-Frame Reference Picture Selection Algorithm for Video Transmission via Error Prone Networks,” in proc. Int. Symposium Autonomous Decentralized Systems (ISADS2005), Chengdu, China, April. 2005.

[45] J-H. Cui, J. Kim, A. Fie, M. Faloutsos, and M. Gerl, “Scalable QoS Multicast Provisioning in Diff-Serv Supported MPLS Networks,” in proc. IEEE Globecom 2002, vol 21, no. 1, Taiwan.

[46] A. Striegel, , and G. Manimaran, “DSMCast: a Scalable Approach for DiffServ Multicasting,” in proc. Computer Networks, vol. 44, issue 6, 22 April 2004, pp. 713-735.

[47] G. Bianchi, N. Belfari-Mellazi, G. Bonafede, E. Tintinello, “QUASIMODO: Quality of Service Aware Multicast over Diffserv and Overlay Networks,” in proc. IEEE Networks, vol. 17, no. 1, Jan/Feb. 2002, pp.38-45.

[48] O. B. Kari, M. Fathy, S. Yousefi, “Application Level Wireless Multi-level ECN for Video and Real-time Data,” in Proc. of the International Conference on

Networking, International Conference on Systems and International Conference on Mobile Communications and Learning Technologies ,pp. 137, 2006.

[49] J. Shim, P. Scheuermann, and R. Vingralek, “Proxy Cache Algorithms: Design, Implementation, and Performance,” in proc. IEEE Trans. Knowledge and data Eng., vol. 11, no. 4, July-Aug., 1999.

[50] L. Yin, G. Cao, and Y. Cai, “A Generalized Target Driven Cache Replacement Policy for Mobile Environment,” in proc. 2003, Int. Symp. Application and the internet (SAINT), Jan. 2003.

[51] U. C. Kozat and L. Tassiulas. “Network layer support for service discovery in mobile ad-hoc networks” in Proc. IEEE INFOCOM, San Francisco, CA, 2003.

[52] B. Li, M. J. Golin, G. F. Ialiano, and X. Deng, “On the optimal placement of web proxies in the internet,” in Proc. IEEE INFOCOM, New York, Mar. 1999, pp. 1282–1290.

[53] I. Cidon, S. Kutten, and R. Soffer, “optimal allocation of electronic content,” in Proc. IEEE INFOCOM, Anchorage, AK, Apr. 2001, pp. 1773–1780.

[54] L. Qiu, V. N. Padmanabhan, and G. M. Voelker, “On the placement of web servers replicas,” in Proc. IEEE INFOCOM, Anchorage, AK, Apr. 2001, pp. 1587–1596.

[55] Z. Xiang, Z. Zhong, and Y. Zhong, “A cache cooperation management for wireless multimedia streaming,” in Proc. IEEE Int. Conf. Info-Tech and Info-net (ICII’01), Beijing, China, Oct. 2001, pp. 328-333.

[56] Z. Wang, M. Kummar, S. K. Das, and H. shen, “Dynamic cache consistency schemes for wireless cellular networks,” in Proc. IEEE trans. On wireless commun., vol. 5, no. 2, Feb. 2006.

[57] Q. Ren, and M. H. Dunham, “Using semantic caching to manage location dependent data in mobile computing,” in proc. ACM MOBICOM, Boston, Massachusetts, 2000, pp. 200-209.

[58] T. Hara, “effective replica allocation in ad-hoc networks for improving data accessibility,” in Proc. IEEE INFOCOM, 2001, pp.1568-1576.

[59] T. Hara, “Replica allocation in ad-hoc networks with period data update,” in Proc. 3rd int. conf. on mobile data management (MDM), 2002, pp.79.

[60] F. Sailhan, and V. Issarny, “Cooperative caching in ad-hoc networks,” in Proc. 4th int. conf. on mobile data management (MDM), 2003, pp.13-28.

[61] Zhaolei Duan, Zhimin Gu, Xiaoguang Ding, “WPCC: A novel web proxy cache cluster,” in proc. 11th International Conference Advanced Communication Technology (ICACT), 2009. vol. 03, 15-18 Feb. 2009 pp. 2205 – 2208.

[62] J. Jeong, M. Dubois, “Cache replacement algorithms with non-uniform miss costs,” IEEE Transactions on Computers vol. 55, issue 4, April 2006 pp.353 - 365.

[63] Y. M. Abdelmalek, and T. Saadawi “Destination-Assisted Routing Enhancement (DARE) for Ad-Hoc Networks,” in the Proceeding of the Int. Conf. Sarnoff April 1-3, 2009 –Princeton, New Jersey, USA.

[64] Y. M. Abdelmalek, A. Abd El Al, T. Sadaawi, “collaborative video caching technique in mobile ad-hoc networks,” in the Proceeding of 11th Internet Management 2000 (IM09), IFIB/IEEE Hofstra university, New York, June 2009.