STREAMING VIDEO USING

COOPERATIVE NETWORKING

A Dissertation

presented to

the Faculty of the Graduate School

at the University of Missouri-Columbia

In Partial Fulfillment

of the Requirements for the Degree

Doctor of Philosophy

---

by

YINGNAN ZHU

Dr. Wenjun Zeng, Dissertation Supervisor

MAY 2009

The undersigned, appointed by the dean of the Graduate School, have examined the dissertation entitled

STREAMING VIDEO USING
COOPERATIVE NETWORKING

presented by Yingnan Zhu,

a candidate for the degree of doctor of philosophy

and hereby certify that, in their opinion, it is worthy of acceptance.

Professor Wenjun Zeng

Professor Yi Shang

Professor Michael Jurczyk

Professor Yunxin Zhao

Professor Zhihai He

ACKNOWLEDGEMENTS

My deepest gratitude is to my graduate advisor Dr. Wenjun Zeng for leading me to the world of independent research and for teaching me how to write good papers and how to be a good presenter. I enjoyed every meeting with Dr. Zeng, and his insightful and sharp thoughts can always help me to find the best way. Without Dr. Zeng's help, I could not have come to this far in the exciting fielding of multimedia communication.

I am enormously grateful to all my committee members, Dr. Yi Shang, Dr. Michael Jurczyk, Dr. Yunxin Zhao and Dr. Zhihai He. I would like to thank each of them for the precious time they took from their busy schedules and their constructive suggestions on my research work.

I would like to thank Dr. Hang Liu and Dr. Yang Guo who are from Thomson Corporate Research in New Jersey. During my internship at Thomson, they gave me numerous suggestions and help. Discussion and conversation with them always help me to find the right direction of research.

I would also like to express sincere thanks to the fellow graduate students in our lab, which includes Hongbing Jiang, Wei Liu, Lina Dong, Qia Wang, Qiuming Yao and Aleksandre Lobzhanidze. Thanks to my friends Mian Pan and Li Liu who helped me throughout my 5 years' study. It has been a great experience working with them.

I would like to express my gratitude to the faculties and staffs in Computer Science department, University of Missouri – Columbia.

Lastly, I would like to express my greatest grateful to my parents, my sister and my wife Shuang Wang for their endless love, support and encouragement to me. Without their support and care, my life and work would not have been so meaningful and fruitful.

Table of Contents

List of Figures

List of Tables

ABSTRACT

With the increase of bandwidth in computer networks, video streaming is expected to become a common practice in various networking environments. The Internet was originally designed to provide best-effort service to transport data without any guarantee on the end-to-end performance. It does not make any promise about the variation of the packet delay jitter, bandwidth, and packet loss, either. It is also more challenging for supporting video streaming services in wireless networks that constantly suffer from quality degradation caused by interference, channel fading, mobility, etc.

The main objective of this dissertation is to improve the overall video streaming performance in various networking environments, such as IP-multicast in wired network and wireless mesh networks (WMNs), using cooperation among participants including clients and routers. We investigate a number of key challenging issues associated with video streaming, i.e. the reliability issue in IP-multicast, the high throughput routing metric design issue, the server/peer selection issue, the admission control issue and peer cooperation issue in WMNs. We explore solutions to the above issues using a cooperative networking approach, which includes constructing overlay Peer-to-Peer (P2P) retransmission networks and exploring hybrid architecture of content distribution networks (CDN) and P2P networks.

The first part of this dissertation addresses the reliability-related issues in IP-multicast. We propose a novel overlay P2P retransmission architecture to exploit path diversity. An approach that leverages both disjoint path finding and periodic selective probing to take into

account peer's recent packet loss probability, retransmission delay and recent retransmission performance is proposed to effectively construct an efficient and dynamic overlay peer retransmission network.

The rest of this dissertation focuses on addressing some important challenges in video streaming over WMN. We first investigate high throughput routing metric for video streaming over WMNs. We propose a radio and bandwidth aware routing metric for video streaming for WMNs. Our experimental results using a testbed show that the enhanced routing protocol has better performance than the traditional minimal hop count based routing protocol in terms of packet loss and network jitter.

Next we explore *a Unified Peer-to-Peer And Cache framework (UPAC*) for high quality video on demand services over infrastructure multi-hop WMNs. UPAC is a hybrid approach combining the CDN and P2P approach.

The research is divided into the following steps:

- We first study the general cooperative mesh content server selection problems in WMNs. A main feature of the UPAC architecture is to deploy mesh content servers in the WMNs so as to balance the load of the network and improve the content availability. We propose a routing metric based cross-layer server selection scheme. Simulation results demonstrate that the proposed scheme performs significantly better than other common server selection schemes used in CDN.

- We then propose a novel admission control algorithm with per-flow routing in WMNs to select the mesh content server and transmission path. We formulate the admission control problem with the interference constraint and investigate the optimal solution using a centralized algorithm. Because the optimal solution is NP-hard, we propose a heuristics approach with admission control and per-flow routing. We demonstrate that the proposed heuristics solution can achieve better overall video streaming quality than the approaches without admission control.

- Finally, a BitTorrent-like P2P approach for UPAC is proposed to further explore the peer/server cooperation in WMNs. In order to meet the stringent requirements on delay and packet priority for video streaming in WMNs, we propose to use routing layer information to select peers and manages peer/data list using an urgent data first strategy combined with the rarest first strategy. Simulation results show that, with the proposed P2P cooperation support, the network load is further balanced, and the network capacity can be utilized much more efficiently.

Chapter 1. Introduction

1.1 Motivation

Recent advances in digital video technology and the computer networks have made streaming video in various network environments practical. Meanwhile, the requirements on Quality of Service (QoS) and Quality of Experience (QoE) become more stringent. In earlier years, most of the data transmitted over the Internet is text or image because of the bandwidth limitation. Today one can easily get access to the Internet with bandwidth as much as 1Mbps (e.g. ADSL). The rapid development of video compression techniques also helps to provide higher definition video with lower data rate. All these technologies have made video streaming a mainstream of communication.

According to a report released (based on one million broadband users in North America) by Sandvine Networks [1], the Internet traffic has changed significantly each year. According to the report, the streaming traffic generated by websites such as Youtube.com increased 5% from 2007 and now share about 14.8% of the total broadband traffic.

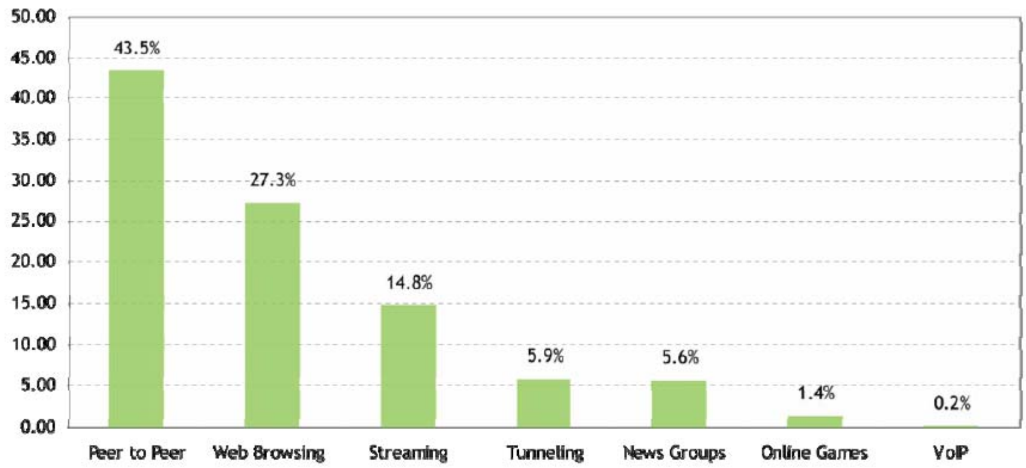


Figure 1.1 Broadband traffic in North America, 2008

Figure 1.1 shows that 43.5% of the total Internet traffic is generated by P2P applications. Obviously, the video/audio application and P2P application account for about 60% of the total broadband traffic. These statistics show that there is a high demand for applications combining P2P techniques and video streaming.

Even with today's Internet bandwidth, it is still important to find a bandwidth efficient way to transmit the multimedia data, which consumes much more bandwidth than text and image data. Different architectures and frameworks have been explored to further improve the overall performance of video streaming systems, such as using IP-multicast in video conferencing and remote learning to reduce the traffic of the network. Beyond the traditional wired network, Wireless Mesh Networks (WMNs) is attracting more and more attention because of its ease of deployment and good scalability. In addition, cooperative architectures such as Content Distribution Networks (CDN) and Peer-to-Peer (P2P) networks provide an efficient way to utilize the network bandwidth. In the CDN architecture, more servers are involved to cooperatively work with each other to help the clients. In P2P networks, the bottleneck problems, e.g. the flash crowd problem in traditional client/server model, are alleviated. Peers use their spare uplink bandwidth more efficiently and the total available resources on the network increase dramatically.

To improve the video streaming quality, transporting data efficiently is one of the key issues. Among the existing architectures, IP-multicast (or native-multicast) is one of the bandwidth efficient transmission mechanisms for applications where there are multiple participants.

The reliability issue is one of the significant limitations of currently deployed IP-multicast networks for video streaming applications. Reliability in IP-multicast scenarios is typically more challenging than a unicast scenario since it suffers from several unique problems, including heterogeneous client bandwidth and capabilities, bandwidth inefficiency and NACK implosion in IP-multicast retransmission [7], and potential sender overloading.

We propose to address the reliability problem using cooperative networking. In the IP-multicast scenario, all the participants in one multicast group are receiving the same set of data. The peer who is suffering packet loss problem can ask for retransmission from either the sender or other peers. To retransmit data from other peers efficiently, peers can construct a cooperative retransmission overlay. The strategy we propose to construct such an overlay is to locate some good peers which are in good receiving status and can provide the needed data in time. In IP-multicast scenarios, the peers with disjoint path from the sender are good retransmission peers for each other. To find good retransmission peers, “tracert”, a simple Internet tool, can be used to estimate the path disjoint peers. In the mean time, probing strategies can also be used to find good peers dynamically. The detail of the proposed work will be discussed in Chapter 3.

Although IP-multicast is a bandwidth efficient mechanism to transmit real time video data over the Internet, there are high demands of accessing the Internet and receiving multimedia streaming more conveniently, especially in cities. Wireless mesh networks meet these requirements because they are low cost, fast and flexible in deployment and easy to extend to areas where wireline deployment is economically infeasible.

Significant progress has been made in deploying 802.11 based WMNs to provide the last-mile accessibility for the Internet users. More than 200 wireless municipal mesh networks (muni-mesh) have been deployed or are being set up worldwide, e.g. Google's WiFi muni-mesh network in Mountain View, CA, and MetroFi' mesh network in Portland, OR. Figure 1.2 shows a typical WMNs topology in living area.

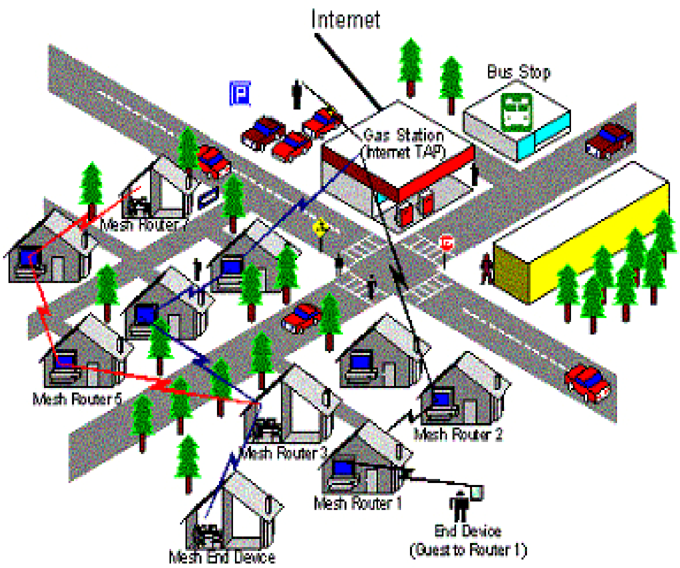


Figure 1.2 A Typical WMNs

Why WMNs are getting so popular? The reason lies in several properties of WMNs. The nodes in WMNs can automatically organize into a network and maintain their mesh connectivity. WMNs are resilient and easily expandable. Individual nodes can be quickly set up and become involved in the network. Those properties make WMNs easy to deploy and cost effective for various application scenarios, including residential, office, campus/community/public access network, and military/public safety.

There are two types of WMNs: client-mesh networks and infrastructure-mesh networks (infrastructure WMNs). Client-mesh networks or ad hoc networks are formed by client devices with no infrastructure required, wherein each node plays the same role and participates in packet forwarding. In contrast, infrastructure WMNs consists of mesh access points (MAPs)/routers and client devices. The MAPs are interconnected via wireless links to form a multi-hop backhaul infrastructure. One or more MAPs connect to the wired network and are referred to as the gateways. Generally a MAP has two or more radio interfaces. One radio interface is an access interface, which is for network access of clients. A second radio interface is a relay interface, which is for routing and data forwarding. Client devices, e.g., laptops, dual-mode smart phones and personal digital assistants (PDAs), associate themselves with a nearby MAP to access the WMNs. They do not participate in packet relay or routing process. A client device sends (or receives) packets to (or from) it's associated MAP. Packet delivery in the WMNs is handled by the MAP through a backhaul routing protocol. In this dissertation, we focus primarily on the infrastructure WMNs.

In wireless networks, high throughput transmission path is essential for real time high data rate applications, e.g. video streaming. The path established between two nodes may go through several relay nodes. Due to interference in wireless medium, the path capacity decreases rapidly as the hop count increases [11]. Furthermore, the large hop counts elevate the chance of wireless signal interferences thus negatively affect other established connections (cross interference) and reduce the overall system capacity. However the hop count is not the sole factor that determines the path quality. The quality of a radio link depends on the received radio signal strength, the packet loss rate, the contention among nearby nodes, link data rate, and traffic load on the link. 802.11 radios support multi-rate adaptation based on link quality. A multi-hop high rate path may be capable of achieving better throughput and shorter delay than a single hop low rate path. How to provide scalable, high quality media streaming services over WMNs is a challenging problem. In addition, the routing protocol and routing metric are critical design factors for performance optimization in WMNs. Many routing protocols e.g. AODV [12], DSR [13], and OLSR [14] have been developed. However, because those routing protocol is not designed for video streaming over WMNs, it is necessary to design a routing algorithm that takes the link load, link quality into account to support video streaming over WMNs. Furthermore, because of the low throughput problem of multi-hop path in 802.11 networks, the intuitive way to reduce the effect is to distribute the heavy traffic load throughout the network and reduce the chance of multiple relay and multi-hop for the end clients. Some strategies include caching at MAPs and using cooperative P2P overlay. To address these problems, we design UPAC, which is a hybrid solution using the CDN and P2P networks. In UPAC, there are mesh content servers and clients. Mesh content servers construct the CDN and all the servers and peers construct the P2P overlay. Each participant works cooperatively to improve the overall video streaming quality of the whole WMNs.

In summary, the challenges in designing the framework for video streaming over WMNs include: to find a good routing metric for multimedia delivery in WMN and maintain a good path from time to time; to find a good server/peer selection strategy that can help to reduce the impact of interference and improve overall video streaming quality in terms of low end-to-end delay, low packet loss ratio and high throughput; to design a P2P mechanism that is efficient for video streaming over WMNs. One of the main objectives of this dissertation is to address the above problems.

In addition, when we design the video streaming framework in various network environments, we usually consider the following possible types of network related faults:

- *Network Congestion:* Depending on traffic within the network and the topology, network could be congested within the network, causing longer delays or packet loss.

- *Packet Loss:* Since in the real time video streaming, the main transport protocol we use is RTP which runs on top of UDP and is connectionless. It implies the packets could be discarded during transmission due to congestion and buffer overflow.

- *Long End-to-End Delay:* End-to-End delay includes transmission delay, propagation delay, queuing delay in routers and buffering delay at end client. For video streaming application, the end-to-end delay is crucial because video playback has critical requirement for delay. Longer end-to-end delay normally results in larger packet loss rate.

- *Transmission Error:* It occurs more frequently in the wireless network environment. Packets could be corrupted or dropped due to the unreliable wireless medium.

By considering the above issues, the main task is to develop a scalable and reliable architecture to support video streaming.

1.2 Contribution

We explore a number of key challenges associated with video streaming over various networks such as the IP-multicast network and WMNs.

We first present an error recovery enhancement in IP-multicast by using a P2P cooperative retransmission overlay approach. The approach leverages both disjoint-path-finding and periodic selective probing to take into account peer's recent packet loss probability, retransmission delay and recent retransmission success rate.

Specifically, the main contributions of this part include:

1) A novel overlay P2P cooperative retransmission architecture to exploit path diversity to address the reliability and scalability of both live and on-demand IP-multicast applications. The proposed framework is practical, scalable (in terms of multicast session size), and easy to deploy, requiring no change to the existing network infrastructure.

2) A novel exploitation of both path diversity and P2P network is proposed to construct a highly distributed, robust, lightweight, cooperative overlay retransmission network to significantly improve the retransmission performance for IP-multicast-based media distribution.

3) Cross-layer interaction is exploited that leverages existing functions supported by the current IP network to help construct an efficient, underlying network topology aware P2P overlay retransmission network.

4) The proposed cooperative architecture can be further extended to bridge IP-multicast networks and non- IP-multicast networks and provide a unified framework to address both multicast connectivity and QoS performance.

We evaluate its performance in comparison to the traditional approach and demonstrate the significant advantages it provides in support of the QoS performance (reliability, delay, scalability) of multimedia applications.

Next, we focus on video streaming in WMNs. By considering the properties and constraints of 802.11 wireless networks, we discuss and design a framework, called a Unified Peer-to-Peer (P2P) And Cache framework (UPAC) for high quality video on demand services over infrastructure multi-hop WMNs.

The main contributions of UPAC include:

1) A new high throughput metric and an enhanced routing algorithm are proposed to support video streaming in WMNs. Routing in video streaming in WMNs has more demanding requirements than routing in other environments and a high throughput routing metric is essential to the performance.

2) A unified cooperative framework combining content distribution network and P2P network to address the capacity utilization issue in WMNs. The framework takes the

advantages of both the CDN and P2P network. It encourages cooperation among participants, is practical, scalable and easy to deploy.

3) A routing metric based cross-layer mesh content server and path selection approach is proposed. This approach takes both routing layer and application layer metric into account. Simulation results show that this approach significantly outperforms other common server selection schemes in CDN.

4) A general centralized admission control and server/path selection problem is formulated and analyzed. The problem provides a solution to minimize the network utilization given a fixed number of requests in WMNs. It considers necessary constraints such as flow conservation, inter-flow and intra-flow interference, delay constraint and etc. This problem formulation can be used in other data transmission/scheduling problems in WMNs.

5) A heuristic solution with new per-flow routing algorithm to solve the admission control problem in WMNs. The solution encourages participants to cooperate more efficiently in UPAC, and each new video streaming request will be admitted by en-route MAPs. Only part of the requests may be admitted to improve the capacity utilization of the WMNs.

6) A new cooperative P2P framework designed for video streaming in WMNs. In this framework, a device forms the P2P relationship with both mesh content servers and other client devices. Meanwhile, the client-server relationship is established between the device and the mesh content servers. This framework improves the scalability of

WMNs. In the framework, new chunk and peer selection methods that are more suitable for video streaming applications are proposed.

Extensive simulations are performed to show the advantage of the UPAC framework.

1.3 Overview of the Dissertation

The rest of this dissertation is organized as follow:

Chapter 2 reviews the background and some important state-of-the-art works related to this dissertation.

Chapter 3 introduces the proposed novel architecture for path-diversity P2P overlay retransmission for reliable IP-multicast. In this chapter, an architecture overview is given first and then we present the detailed design of different key components in the proposed architecture. We examine the architecture performance through extensive NS2 simulations.

We address video streaming over WMNs in the next two chapters. We start with the routing metric design in Chapter 4. A new radio and bandwidth aware routing metric which is suitable for video streaming applications in WMNs is proposed. We present the enhancement of AODV routing algorithm with the proposed routing metric.

In Chapter 5, we first describe the overall architecture of UPAC and propose a cache server selection mechanism. Then we formulate a problem to optimally schedule/admit the new streaming request with minimal average network usage. We propose a heuristic solution with routing layer admission control and a novel per-flow routing algorithm. Lastly, we propose P2P support in UPAC. Simulation results illustrate that using UPAC, the network load could be further balanced and the capacity of WMNs could be significantly improved.

Finally, Chapter 6 summarizes the major contributions of this dissertation and gives several suggestions for future work.

Chapter 2. Background and Related Works

This chapter presents the related works in the literature. First, the research works in IP-multicast are discussed. Then the cooperative architectures such as CDN networks and P2P networks are presented. Lastly the existing P2P file sharing and streaming techniques are introduced.

2.1 IP-multicast

Traditional IP communications allow a host to send packets to another host (unicast transmissions) or to all hosts (broadcast transmissions). IP-Multicast provides a third communication alternative: allowing a host to send packets to a group that is made up of a subset of the hosts on the network. IP-Multicast is a bandwidth-conserving technology specifically designed to reduce traffic by simultaneously delivering a single stream of information to potentially thousands of corporate recipients or homes. By replacing copies for all recipients with the delivery of a single stream of information, IP-Multicast is able to minimize the burden on both sending and receiving hosts and reduce overall network traffic. Within a multicast network, routers are responsible for replicating and distributing multicast content to all hosts that are listening to a particular multicast group [2].

There have been significant deployments of IP-multicast in exchanges and securities trading companies, enterprises and college campuses, edge networks, and military networks

[3]. For example, the Abilene Network in Internet-2 [4], a private network that inter-connects most major universities in the US for the education and research purpose, fully supports IP-multicast, and application platforms such as Microsoft Research's ConferenceXP platform [5] have been developed for such networks. Despite all these progresses, we are yet to see "the ubiquitous deployment of a revenue generating native multicast infrastructure capable of securely and robustly supporting both reliable, TCP-friendly file transfer, all manner of streaming media, and any style of audio/video conferencing (with minimal jitter and end-to-end delay) - all with only minimal additional router complexity, deployment effort, management needs, or cost" [2]. The main challenges for traditional IP-multicast include wide-area-network deployment, reliability and congestion control, state-scalability, security, inter-domain and source discovery, etc. In the meantime, application layer multicast (ALM) [6] that addresses the deployment and complexity issues at the cost of some performance loss (in terms of link stress, relative delay penalty, and instability due to the nodal join and leave activities) has been gaining popularity. Research efforts on IP-multicast continue [7] and the concept has been revisited in both academia and industry [8][9][10]. Examples are the on-going Automatic Multicast Tunneling (AMT) work in IETF [9], and the recent initiative in IRTF on scalable adaptive multicast (SAM) [10] that aims to devise multicast framework that addresses limitations of native multicast and leverage both application layer and native techniques.

2.2 Content Distribution Networks

Content Distribution Networks (CDN) consists of a collection of (non-origin) servers that attempt to offload work from the origin server by delivering content on its behalf. CDN improves network performance and offers fast and reliable applications and services by distributing content to cache servers located close to users. For each request, CDN attempts to locate a surrogate server close to the client, where "close" does not only mean the close in physical distance, it is an abstract metric that considers a variety of criteria, i.e. physical distance, speed, reliability, and data transmission costs. Not only based on this metric, CDN will probably select a server based on a balance of multiple metrics, i.e., proximity, server load (the load of servers or network paths to the servers), and an aggregate of them. For example, a proximity-load-threshold algorithm [44] uses proximity metric to select the nearest surrogate server, and uses a load metric to ensure that the server is not overloaded.

The servers belonging to CDN may be located at the same site as the origin server, or at different locations around the network, with some or all of the origin server's content cached or replicated amongst the CDN servers. CDN acts as a trusted overlay network that offers high-performance delivery of common Web objects, static data, and rich multimedia content by distributing content load among servers that are close to the clients.

There are several issues to address in CDN, such as how to redirect user's request to the appropriate surrogate server to use the network more efficiently, and where to place the surrogate servers in order to help to provide the best performance. To select the most appropriate surrogate server for content routing, most current CDN systems use Domain Name System (DNS) redirection [45] and some also use URL rewriting [46][47].

Furthermore, the server placement problem can be thought of as the CDN topology creation problem. A well designed system should maximize the client-perceived performance and minimize the infrastructure's cost. In general, the server placement problem is to place M surrogate servers among N different sites ($N > M$) to yield the lowest cost (widely known as the minimum K-median problem). To help to reduce the number of surrogate servers needed and the size of content (replicated on them), several placement algorithms have been proposed (such as Greedy algorithm, which incrementally places replicas, Hot Spot, which places replicas near the clients generating the greatest load, and the Tree based replicas). These algorithms specify the locations of the surrogate servers in order to improve the performance with low infrastructure cost.

2.3 Peer-to-Peer Networks

According to [48], a peer-to-peer (or P2P) computer network uses diverse connectivity between participants in a network and the cumulative bandwidth of network participants rather than conventional centralized resources where a relatively low number of servers provide the core value to a service or application. P2P networks are typically used for connecting nodes via largely ad hoc connections. Sharing content files containing audio, video, data or anything in digital format is very common. Real time data, such as video streaming, telephony traffic, is also being transmitted using P2P technology.

2.3.1 Peer-to-Peer File Sharing

P2P system first appeared as a file sharing system. Among different P2P file sharing systems, BitTorrent [55] and KaZaa are the popular ones.

- BitTorrent

  BitTorrent has emerged as a very popular and scalable peer-to-peer file distribution system. It has been successful at distributing large files quickly and efficiently without overwhelming the capacity of the origin server.

The BitTorrent architecture normally consists of the following entities:

- A static meta-info file (a “torrent file”);
- A ‘tracker’;
- An original downloader (“seed”);
- The end user downloader (“leecher”).

In the basic BitTorrent system, the meta-info file is called the “torrent” file. The torrent file contains the filename, size, hashing information and the URL of the “tracker”. The “torrent” is needed by anyone who wants to download the file that the torrent is created from, and can be distributed by e-mail, IRC, http etc. The tracker keeps a log of peers that are currently downloading a file, and helps them find each other, and it is not directly involved in the transfer of data and does not have a copy of the file. The tracker and the downloading users exchange information using a simple protocol on top of HTTP. First, the user gives information to the tracker about which file it is downloading and the ports it is listening on. The response from the tracker includes a list of other users which are downloading the same file and information on how to contact them. The group of peers that all share the same torrent represents a ‘swarm’. An original downloader known as a “seed” has to be started. A “seed” is a user that has the entire file. A downloading user that has nothing or only parts of a file is called a “leecher”. When creating the torrent file from the original file, the original file is cut into smaller pieces, usually 512 KB or 256Kb in size. Whenever a piece is downloaded and verified, the downloading peer reports to the other peers in the swarm about its new piece. This piece is now available for other peers. BitTorrent uses random first piece for users who do not have any part of the file to start download, and it uses rarest first strategy to help to maintain the availability of the wholeness of the content. It uses the tit-for-tat policy to avoid free riders.

However, with one centralized tracker, the BitTorrent network is not very fault tolerant. If the tracker goes down the file will no longer be available, since there are no way the peers can know about each other. One centralized tracker also makes the network vulnerable to denial of service attacks.

In 2005, a distributed version of BitTorrent was released. The tracker is distributed in the sense that every client or node in the network acts as a lightweight tracker. The solution is based on distributed hash tables (DHTs). This makes it possible to share files with minimal resources, but no guarantees can be made with respect to reliability.

- KaZaa

KaZaa is also one of the most popular and widely used P2P systems. It has over 85 million downloads worldwide and an average of 2 million users online at any given time. Because KaZaa has little documentations, most of the research works are based on inferring how KaZaa works [56].

According to [56], KaZaa does not use a dedicated server for tracking and locating content, and not all peers are equal. KaZaa has two classes of peers, Ordinary Nodes (ONs) and Super Nodes (SNs). SNs are the more powerful peers and have greater responsibilities. Each ON is assigned to a SN and uploads information about the files it is sharing to its SN. This allows the SN to maintain a database which includes the identifiers of all the files its children are sharing, metadata about the files, and the corresponding IP addresses of the ONs holding the files. Each SN then becomes a (mini) Napster-like hub. In contrast with Napster, a SN is not a dedicated server (or server farm); instead, it is typically a peer belonging to an individual user. Each database in SNs only keeps records for files located in its direct children and it does not track the files that are in ONs under other SNs. For each file that is shared, the metadata includes: the file name, the file size, the ContentHash, and the file descriptors. As part of the signaling traffic, KaZaA nodes frequently exchange the lists of super nodes with each other. By frequently exchanging the lists of the super nodes, nodes maintain up-to-date lists of active SNs. Peers in KaZaA mainly use two criteria for ON-to-SN and SN-to-SN neighbor selection. One of these criteria is the workload of the super node, and the other is based on locality, that is, nodes (both ONs and SNs) appear to choose overlay neighbors that are close.

2.3.2 Peer-to-Peer Streaming

Both Bittorrent and KaZaa are P2P file sharing systems, in which the users only care about the completeness of the file instead of when to receive which part of the data before a deadline. To apply P2P systems to transmit time sensitive data, some works considered how the BitTorrent can assist in VOD system [57][58][59]. The basic idea of [58] is to keep track of the streaming order of the piece, and of the preferred pieces that will be played soon. The video data is divided into three sets: a downloaded set, a download remain set (may contain rare data) and a high priority set. Because video data has delay constraints, the most urgent data has the highest priority. BASS [59] applies BitTorrent to the traditional VOD system and uses the BitTorrent method to balance the load of the network. BASS introduces one dedicated external streaming server to help to guarantee the dada availability. If the peer cannot get data from other peers in time, it will get data from the server. The modification to BitTorrent is that it does not download any data prior to the current playback point. Similar to BASS, PONDER [57] also includes a video server to assist the VOD session. The main task of the video server in PONDER is to provide the first segment of the video file to ensure data can be fetched in time.

Chapter 3. Path-diversity P2P Overlay Retransmission for Reliable IP-multicast

To address the reliability and scalability issues in IP-multicast, in this chapter, a novel, highly distributed, and light-weight overlay peer-to-peer retransmission architecture is proposed which exploits path-diversity by taking advantages of both IP-multicast and an overlay network. An approach that leverages both disjoint-path-finding and periodic selective probing to take into account peer's recent packet loss probability, retransmission delay and recent retransmission success rate is proposed to construct an efficient and dynamic overlay retransmission network.

3.1 Related Works

To help address data loss in challenging network environments, forward error correction (FEC) [15][16] and selective retransmission have been proposed to recover lost packets. FEC introduces bit overhead in the transmission, thus should be used judiciously. There is no efficient "one-fit-all" FEC solution in IP-multicast due to heterogeneous client channel conditions. In addition, delay will be introduced when FEC is applied across packets to address packet-loss. On the other hand, selective retransmission is generally considered more bandwidth efficient, at the cost of some delay penalty. There is on-going work in IETF to support RTP retransmission (with the support of early feedback and more frequent RTCP feedback [17]) which is considered to be an effective packet loss recovery technique for reliable real-time applications with relaxed delay bounds [18][19]. Selective retransmission can also be combined with FEC to achieve better performance, especially for reliable multicast [7][18].

Traditionally, to address reliability, the receiver sends the retransmission requests to the original sender who then may choose to retransmit the lost packets if deemed important. A common problem in this approach is that it is very likely that the retransmitted packets will go through roughly the same routes as the original packets, which unfortunately are probably congested as suggested by the loss of the original packets. As a result, the retransmitted packets are also likely to experience congestion and loss, especially in a bursty loss scenario. If not done wisely, the retransmitted packets may even deteriorate the congestion condition. In an IP-multicast scenario, there is also a scalability problem, more specifically, the request and reply message implosion problem in supporting retransmission. Typically the retransmitted packets are also IP-multicast to all participants who have subscribed to the retransmission session, even though only one or few participants might have experienced the loss of that particular packet. This would result in a waste of the bandwidth. The original sender may choose to only retransmit (multicast) packets that are requested by many participants at the cost of additional delay introduced in aggregating the feedbacks from multiple participants, and of some lost packets not being retransmitted. Separate unicast session can also be established by the original sender to convey retransmissions to each of the requesting receivers. This, however, will significantly increase the load of the original sender.

Switching between unicast and IP-multicast retransmission based on the number of requests has also been considered [20][21].

Error recovery for reliable IP-multicast transport with controlled bandwidth overhead and reduced latency through *local* IP-multicast retransmission has been studied in the past [20]-[32]. Most of these previous works, however, require various degrees of static configuration, centralized coordination [23], or router/proxy support [24] [29]-[32], which make them difficult/costly to deploy in practice. In [25], a hop-based scope control for IP-multicast retransmission and use of separately addressed local recovery multicast groups are proposed to address local recovery. As discussed in [25], the hop-based scope control approach has performance limitation when compared to global error recovery if the topology of a session is star-shaped, and the group-scoped error recovery has more overhead on the members as well as on the underlying IP-multicast routing. Complexity and practical deployment is again a major obstacle. More significantly, the focus of all the above *local* IP-multicast retransmission approaches is mainly to address the bandwidth overhead issue inherent in IP-multicast retransmission. Path diversity is not exploited in any of the above approaches. As a result, retransmission still suffers from the congestion bottleneck on the IP-multicast tree.

Overlay networks have been considered as effective cooperative solutions to address some fundamental and challenging networking problems. The concept has been used to build peer-to-peer systems for file sharing, content distribution, streaming, and distributed storage, e.g., in [33]-[37], to address resilient routing [38] and IP security [39], to develop application layer multicast [40] to overcome the deployment issue of IP-multicast, etc. Path-diversity has also been exploited in the construction of some of the overlay networks, e.g., in [33][38][41] and has shown to provide robustness in video communication applications. Given multiple paths, video streams can be divided into sub-streams, which can be transmitted over different paths simultaneously. If multiple paths are disjoint, each sub-stream experiences relatively independent packet loss and that help to improve the reliability of data transmission. Error recovery for video streaming in application-level multicast is addressed in [42] by adding some extra paths or "links" into the ALM tree and replicating packets along these links with some probability, and in [43] by constructing multiple independent ALM trees to reduce the packet loss correlation between nodes on different ALM trees, at the cost of some additional delay and bandwidth inefficiency. In [43], only delay is used as the metric in choosing the recovery nodes and designing the retransmission schedule which are performed before data delivery or during the ALM tree reconfiguration. This multi-tree delay-centric approach is mainly targeted for recovering packet loss due to node join/leave/failure (a significant issue in ALM), but has limited performance for recovering packet loss due to network congestion, as close recovery neighbors, even though on different ALM trees, tend to suffer from the same network congestion problem. Note that the issues that need to be addressed to achieve reliability in IP-multicast and ALM are quite different. For example, packet loss in ALM is a result of network congestion, node join/leave, or node failure, while packet loss in IP-multicast is mainly due to network congestion, which is the main issue we address in this dissertation.

Typically, peer selection and maintenance plays an important role in a P2P system and often is the most costly part of the system, as peers are expected to serve the bulk data. In the first generation of P2P network, such as Napster [49], peer selection is done by a central server, which is not scalable. Peer selection can also be facilitated by leveraging different capacities of the nodes. For example, KaZaa [50], selects super nodes from peers which store some peers' information. A popular technique is to find peers using the distributed hash table (DHT). It assumes that all peers have the same 'hash' function and each content has a unique key. Examples are Chord [51] and Pastry [52], which concern about how to find a close peer with the requested content. They do not consider load-balancing and path quality fluctuation. In [53], the authors present a machine learning approach such as decision tree learning for peer selection and Markov decision for peer switching, but the online update of the rating system is a concern and the current system does not use multi-source transmission which is important for a P2P system. In [54], the authors present a topology-aware peer selection scheme and dynamic switching scheme for active sender selection in a multi-sender to one receiver streaming scenario, in which each peer will infer the topology of the network several times throughout the session by using logical topology, available bandwidth and loss rate (measured on a segment basis). After finding the logical topology, each receiver tries to cluster peers with similar path to the receiver and select a subset of senders who have path diversity to the receiver (so that they have a good aggregated streaming rate and smaller loss rate at the receiver). Because measuring available bandwidth in such application is important but costly, the authors also explore alternative ways to measure the bandwidth with less extra effort by trading off accuracy. Their approach relies on some underlying P2P substrates such as Pastry to provide, with additional lookup complexity, initial peer lists based on which the proposed peer selection is performed.

In this chapter, an important objective is to build a light-weight cooperative retransmission overlay, which is not designed for bulk data delivery. We propose a two-step process in building the overlay. The first step is performed once by each receiver when it joins the multicast session and it concerns only about inferring the logical topology. In this step, the receiver find the path information from the original sender to the receiver and by exchanging the information with other peers, the receiver can preliminarily identify a subset of candidate retransmission peers with disjoint path to the sender. The second step is to dynamically select the “best” retransmission node from the candidate subset on the fly by leveraging measured end-end delay and packet loss ratio through selective probing and taking into account successful retransmission rate.

3.2 Overview of the Proposed Framework

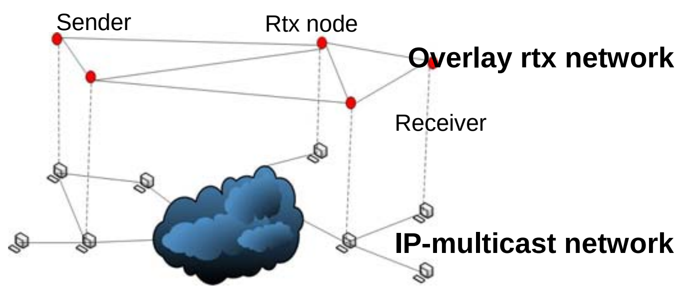


Figure 3.1 An overlay retransmission network architecture for IP-multicast

The basic idea of the proposed path diversity overlay retransmission is to build a very simple overlay network among the participants of the IP-multicast session, in which each receiving node identifies a few peer receiving nodes who, upon requests, will be responsible for retransmitting a lost packet to the requesting receiving node through unicast. These identified peer nodes are called retransmission nodes. The original data packets will be delivered using IP-multicast to all receiving nodes. It is expected that in most cases the retransmission nodes would have received the original packets and thus could perform retransmission upon requests. This hybrid network architecture for reliable IP-multicast applications is illustrated in Figure 3.1.

Intuitively, the retransmission nodes for a receiver should be those end hosts that have been recently having little problem receiving packets from the original sender (source), have a good network connection to the receiver, and are typically closer to the receiver than the original sender. Having little problem receiving data from the original sender implies that this candidate node is not sharing the congested link in the IP-multicast tree with the retransmission requesting node. Identifying peer nodes that have a more disjoint path to the sender will help to address this issue, as will be discussed later. The quality of the network connection between the candidate retransmission nodes and the receiver can be estimated by the receiver, for example, based on past retransmission experience, or by periodic probing in a streaming scenario. The quality of the network connection between the original sender and the candidate retransmission nodes is estimated by the candidate retransmission nodes themselves and then conveyed to the receiver upon periodic probing. When there are multiple sending sources (e.g., in video conferencing), each receiver identifies a small set of "good" retransmission nodes for each potential sending source.

With the identification of a small set of good retransmission nodes, a receiver can send the retransmission request (with the original sequence number of the lost packets and some other necessary information) to one (presumably the best one) of its retransmission nodes who typically would have received or will receive that packet. The chosen retransmission node then forwards the requested packet, if available, using a separate unicast RTP session to the requesting receiver. If the retransmission nodes are chosen intelligently, the retransmitted packet would go through a unicast route that is different from the original congested path along the IP-multicast tree, and thus have a much better chance of getting to the requesting receiver reliably and timely, improving the user experience of the requesting receiver. Furthermore, this overlay retransmission architecture provides load balancing that redistributes the retransmission load of the original sender to other peers, making the system much more scalable. The retransmission nodes are highly distributed without the limitation of a regular topology, which makes the system very robust to network failure. It also addresses the potential bandwidth inefficiency problem of the traditional approaches that use IP-multicast retransmission.

Since the overlay network is mainly used for the retransmission purpose, the construction and maintenance of the overlay network should be very lightweight. The traffic generated by the retransmission is only a small portion of the overall video/audio streams. Thus it may be an overkill to try to optimize the overlay structure as it may incur large overhead to maintain such optimum (i.e., by dynamically changing the overlay based on the current *global* network condition). Therefore, we choose to use a highly distributed lightweight overlay network. There is no central controller. Each receiver simply picks a small set of candidate retransmission nodes. Each receiver periodically probes its retransmission nodes. Based on its own load, the retransmission node only accepts a limited number of overlay requests. Therefore, the only extra traffic is generated by periodic probing. To reduce the probing overhead, the probing interval can be chosen to be relatively large, e.g., every tens of seconds. We show in Section 3.4 that with a small cost of periodic probing, the overlay retransmission network can significantly improve the retransmission performance in the case of challenging network environments with dynamic background traffic.

3.3 Algorithm Design

In an IP-multicast scenario, data loss is mainly due to network congestion. Typically, multiple nodes will simultaneously face the same congestion problem as they share the same congested link. Other nodes that have a more disjoint path to the sender may not have the identical congestion problem, therefore will have a good chance to help the nodes in trouble to recover the lost data, potentially using some paths different from the congested one. For example, in [41], the AS-level information is used to find alternative path. In the design, we consider how to explore both path diversity information and periodic probing to help dynamically identify the best retransmission candidates to recover the lost data.

The construction of the overlay network topology is the most important component in the design. In dynamic environments, periodic probing can help construct a good overlay network. Nodes can monitor the status of a small set of other nodes and the path quality through periodic probing to decide how to choose a neighbor in the overlay network. A basic probing approach is to send probing packets between each pair of nodes, as is done in RON [38], resulting in $O(n^2)$ overall probing traffic for an overlay network with a size of n. Although this method can achieve accurate information, the probing overhead may not be acceptable in our application scenario. In this chapter, a periodic *selective probing* mechanism is used in which only a small subset of a constant size of the overlay nodes are probed, resulting in $O(n)$ complexity for the entire multicast group.

As discussed above, another important consideration in the design of the overlay network is the underlying network path diversity for the nodes in the overlay. A readily available network utility function "Tracert" is used to gather the path information to the sender for each overlay node and identify nodes with disjoint path to the sender. Note that "Tracert" is only used to help *initially* identify a subset of candidate peers with path diversity property. Identification of more accurate candidate retransmission nodes is done dynamically later through periodic probing.

We have the following basic assumptions in the design:

- All the routers support the "Tracert" function
- The exchange of "Tracert" information is effectively lossless.

3.3.1 Identify Path-disjoint Retransmission Candidates

In IP-multicast, packets flow along an IP layer tree rooted at the source. The leaves of the tree are the receivers in the multicast session, the IP-multicast routers are the internal nodes on the tree, and the links form the edges of the tree. A very simple example is illustrated in Figure 3.2, where node *A* is the root, node *E*, *F*, *C*, *G* and *H* are the receivers, and *B*, *D* are IP-multicast routers.

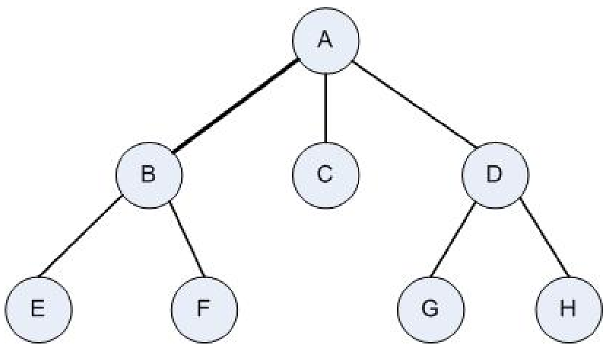


Figure 3.2 A simple IP-multicast tree

In this structure, any packet loss along any "internal" link will result in all the downstream nodes/receivers not receiving the packet. For example, if congestion occurs along the link between node A and node B, then both node E and node F will have the same packet loss problem. This congestion, however, will not affect the receiving status of node C, G and H. Therefore choosing some appropriate receiver node different from the original sender as the retransmission node may help. A good strategy is to choose the receiver nodes with the most disjoint path to the sender. Here two sets of nodes are defined, routers and end nodes (including both senders and receivers). We denote the routers with capital letter "R" and denote the end nodes with lower case "r" for receivers and "s" for senders. Assume the root is node s, and the receiver $r \in \{r_i \mid i = 1, 2......n\}$, where n is the total number of receivers in the multicast group, and let $P_{l_{ab}}$ be the loss probability of the link l_{ab} from node a to node b along the IP-multicast tree.

In Figure 3.3, each dash line represents potentially more than one links in the tree. For a pair of nodes including a receiver node r_1 and one of its potential retransmission candidates, node r_3, k_{13} denotes the number of links from node s to node R_1, n_{13} is the number of links from node R_1 to node r_1, and m_{13} is the number of links from R_1 to r_3. When r_1 experiences packet loss, r_1 would try to identify a candidate retransmission node that has the highest probability of receiving that packet. The problem can be formulated as finding a "good" node g with the maximum probability $P(g=1|r_1=0)$, where $r_1=0$ means r_1 loses the packet and $g=1$ means g receives the packet.

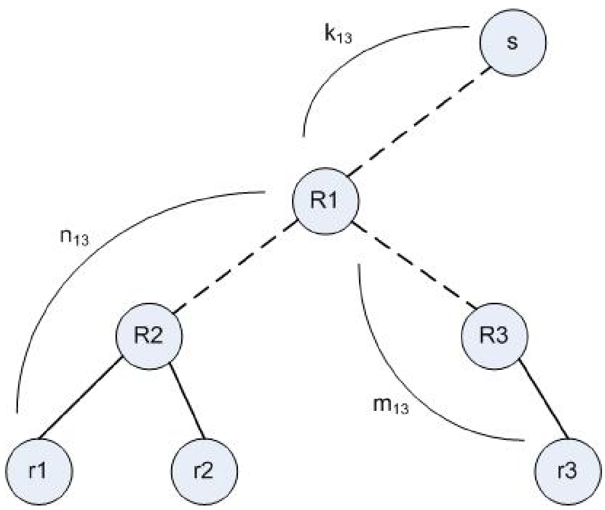


Figure 3.3 Finding a retransmission node with the most disjoint path to the sender

Let $\tilde{P}_{AB} = \prod_{l_{ab} \in \{\text{links from } A \rightarrow B\}} (1 - P_{l_{ab}})$ be the probability of successful transmission along all the links from a node *A* to another node *B*. We know that for r_3,

$$P(r_3=1|r_1=0)=\tilde{P}_{sR_1}\tilde{P}_{R_1r_3}[1-\tilde{P}_{R_1r_1}]/[1-\tilde{P}_{sr_1}] \qquad (3.1)$$

If for each particular receiver *i* and each of its potential retransmission candidate *j*, the values of k_{ij}, m_{ij} and n_{ij} can be determined, and then the retransmission candidates with the best disjoint-path can be identified. For this purpose, we use a common tool "Tracert" that is readily available in most, if not all, network systems. At the beginning of the multicast session, the active/joining receivers will send a "Tracert" request to the original sender to collect the information about the *unicast* route to the sender and then send it to others through the *IP-multicast channel*. We assume there is no packet loss during this route information exchange phase, so each node will have the information about the path and hop count to the original sender of all or sufficient number of active participants, and k_{ij}, m_{ij} and n_{ij} can be determined. Denoted $c_{ij} = k_{ij}+n_{ij}$. Let's assume that $P_{l_{ab}}$ is the same for each link and can be denoted as P. Then we have

$$P(r_j=1|r_i=0)=[(1\text{-}P)^{k_{ij}+m_{ij}}\text{-}(1\text{-}P)^{m_{ij}+c_{ij}}]/[1-(1\text{-}P)^{c_{ij}}] \qquad (3.2)$$

so for r_i, it is necessary to identify $j=\operatorname{argmax}(P(r_j=1|r_i=0))$. In the simulation, to simplify the problem, the number of unshared links normalized by the total number of links on the path from the sender to the receiver is used to estimate this probability.

Note that "Tracert" collects the information about the *unicast* route to the sender, which may be different from the route on the IP-multicast tree. Nevertheless, this is sufficient for our application as only approximate route information is necessary to determine an initial set of path-disjoint nodes. In addition, periodic probing is also employed to dynamically determine the retransmission node, making the accuracy of route information less critical. Another important consideration of using unicast "Tracert" is that it is supported by all routers, making the proposed system more practical and deployable.

3.3.2 Integrated "Tracert" and Periodic Probing

"Tracert" allows each end node to know the approximate path to the sender, and by exchanging this information with other peers, a receiver can find good retransmission candidates with a more disjoint path. This will increase the probability of successfully retransmitting the lost packet to the receiver. However, as will be shown in our simulation results, "Tracert" alone is not sufficient to effectively solve the reliability problem. When the network load is more or less static, using retransmission node with the most disjoint path can perform well where a peer that does not share the same congestion problem as the receiver will typically be chosen. However, since the network load will typically change over time, there is also a high probability for a node with a disjoint path to have different congestion problems at the same time. In this case, it is difficult to tell whether the chosen retransmission peer, although with a good disjoint path, has the packet the receiver requests for. The only way for the receiver to judge whether the chosen retransmission node is performing well is to calculate its retransmission success rate, which is the ratio of the number of successful retransmissions to the total number of retransmission requests. If the chosen retransmission peer is not performing well, this ratio will decrease quickly and based on this the receiver could make a decision to switch to a different retransmission candidate. This adaptive method can help to identify a good retransmission candidate eventually, but it is a passive approach with a cost of delay. For delay-sensitive multimedia applications, this may not provide satisfactory performance.

If a receiver is periodically updated about the status of other peers through unicast probing, it can identify a good retransmission candidate more quickly and accurately, because from periodic probing a receiver can know the delay and packet loss information of other peers, which are important for choosing a good retransmission candidate. But periodic probing alone also has some limitations. First, periodic probing only provides the peer's recent past information. If a receiver's close neighbor has no problem receiving data before congestion occurs, the receiver will most likely choose this close neighbor as the retransmission candidate as the retransmission path from this close neighbor is short. However, this close neighbor may share a number of common links on the IP-multicast tree with the receiver, therefore experiencing the same congestion problem as the receiver. Second, probing overhead is a concern that may limit the system's scalability.

From the above discussion, we know that "Tracert" can help to identify a set of retransmission candidates with good disjoint path, but has limited performance when the network load is more dynamic. Periodic probing can perform well when the network is more dynamic, but cannot exclude close neighbors on the IP-multicast tree from being selected as the retransmission node in a timely manner. Therefore, a hybrid approach that integrates both "Tracert" and probing will get the benefits of both. It will help to identify a good set of retransmission candidate nodes by excluding close neighbors on the IP-multicast tree, and adaptively and timely choose a good retransmission node in a dynamic environment.

3.3.3 Reducing the Probing Overhead - Selective Probing

In an overlay network with n nodes, a full scale unicast probing will gather more accurate information. However, the probing overhead will be $O(n^2)$. There are ways to reduce the probing overhead in overlay networks. For example, in structured peer-to-peer systems [82], a node maintains connections to $O(\log n)$ neighbors. Each node periodically probes its neighbors and replaces the neighbors that are in bad receiving states. Thus the overall probing overhead is reduced to $O(n \log n)$. In the proposed overlay network, the main objective is to help retransmit the lost data for the requesting node. So to avoid incurring significant probing overhead, the construction/maintenance process of the overlay should be very lightweight. In fact, for each node, it is not necessary to maintain a long list of retransmission candidates. Each node could periodically probe only a small subset of the nodes in the overlay network. For example, it can randomly select C nodes as the probing targets, and as a result, the probing overhead is reduced to $O(n)$. In the proposed solution, instead of randomly selecting the probing targets, the node selects the targets by considering the frequency that the target node appeared in its candidate retransmission nodes list. More specifically, a node always chooses *C* best (based on the previous performance) candidate retransmission nodes to probe, because typically these nodes have a better chance to be a good retransmission node. Other candidates will be probed much less frequently. As a result, the best candidates are updated with a unicast probing overhead of O(n) for the entire overlay network.

3.3.4 Design of the Hybrid Scheme for Retransmission Node Selection

In this section, we describe in more detail the distributed algorithm to build the light-weight overlay retransmission network. The process of building the overlay network mainly concerns the selection of the best retransmission nodes for each receiver. Once the retransmission nodes are determined, RTP unicast channels can be established directly between each receiver and its retransmission nodes. The algorithm is run at each receiver so that each receiver can dynamically determine its own retransmission nodes.

Based on previous discussion, the best retransmission candidate should satisfy the following conditions:

1. Use less shared IP-multicast path with the retransmission requesting node.
2. Not too "far away" from the retransmission requesting node, which means the retransmission path in the overlay should satisfy the delay constraint of the multimedia application.
3. Has better receiving status than the retransmission requesting node, so that it has a high probability of having the packets that need to be retransmitted.

The information obtained through "Tracert" can be used to determine condition 1 and use the Round Trip Time (RTT) between the receiver and the original sender and the RTT between the receiver and the retransmission candidate to determine condition 2. Finally, condition 3 could be enforced by considering the packet loss ratio (PLR).

In each RTP IP-multicast session, each receiver in the multicast group can easily measure the RTT between itself and a particular original sender, as well as the packet loss ratio of the data it receives from that sender. Only the very recent RTT and packet loss ratio that reflect the current network condition are of relevance. Let us define RTT_i^j as the RTT between sender *j* and receiver *i* as measured by receiver *i*, and PLR_i^j as the packet loss ratio observed by receiver *i* for the data sent by sender *j*.

As discussed previously, at the beginning of each multicast session, peers will exchange their "Tracert" information. After gathering the "Tracert" information, each receiver *k* could calculate, for a particular sender *j*, the probability $P(i=1|k=0)$ denoted as $\mathrm{PT}_{ki}{}^{j}$ according to Eq.(3.2), where *i* is the potential retransmission candidate. Based on this information, a subset of peer nodes can be identified to serve as the initial candidate retransmission nodes.

In addition, each receiver will periodically probe through unicast a subset of other receivers about their receiving statistics with respect to the original sender. Once a node receives a probing packet, it will send its own measured RTT and packet loss ratio with respect to the original sender back to the probing node. For example, if node 1 receives a probing packet from node 2 for sender 3, node 1 will send RTT_1^3 and PLR_1^3 to node 2. Node 2 will also measure the unicast delay between node 1 and 2 in the probing process. For the retransmission purpose, the receiver should be more concerned about the delay between the retransmission nodes and the receiver as opposed to the delay from the original sender to the retransmission node. With all the periodically updated probing feedback, each receiver can determine a good retransmission node.

The best retransmission node is chosen using the following strategy.

For node *k*, the retransmission node for sender *j* is

$$RTX_k^j = \underset{i=1,2,3...\quad i\neq k}{\arg\min}\{A*\left(\alpha\frac{RTT_k^i}{RTT_k^j}+\beta\frac{PLR_i^j}{PLR_k^j}\right)+B*F(PT_{ki}{}^j)\}T(1/G_k^i) \tag{3.3}$$

and $A+B=1$, $\alpha+\beta=1$, $\frac{RTT_k^i}{RTT_k^j}\leq 1$ and $\frac{PLR_i^j}{PLR_k^j}\leq 1$.

where function *F* is to calculate the normalized $PT_{ki}{}^j$, and is defined as $F(x)=(1-x)^\gamma$, where γ is set to 1 in our experiments, $\frac{RTT_k^i}{RTT_k^j}$ and $\frac{PLR_i^j}{PLR_k^j}$ are the normalized *RTT* and *PLR* respectively for node *i* with respect to sender *j*, which are not greater than 1 as nodes with larger *RTT* or *PLR* are not considered as good candidates, *A*, *B*, *α*, and *β* are the weighting factors, and G_k^i is the success rate of retransmitting packets from node *i* to node *k*, which is initially set to 1. After node *i* is chosen by node *k* as the retransmission node, G_k^i will reflect the quality of the retransmission path between node *i* and node *k* and also reflect how well node *i* receives data from the original sender. The choice of the function *T* will be determined by the system design and it could be a polynomial function. We use the linear function as the *T* function in our simulations. If G_k^i is low, it means that either node *i* often does not have the data that node *k* requests for, i.e., it probably also has a packet loss problem, or the path between node *i* and *k* is bad such that the retransmitted data from node *i* cannot reach node *k*. In fact, G_k^i is more important in the decision process than the normalized *RTT* and *PLR*, as in some cases, two nodes that are physically very close to each other and are in the low level of the IP-multicast tree are likely to face the same congestion problem. We need to be able to avoid these nodes being retransmission node of each other. This situation can be avoided by considering both the degree of path disjoint and the retransmission success rate. Because those physically close nodes will have similar packet loss pattern, when congestion occurs, although the normalized *RTT* and *PLR* collected in the last probing could be good enough, the retransmission success rate will decrease dramatically because that node most likely does not have the packets requested to be retransmitted. Using Eq.(3.3), this problem can be successfully addressed very quickly. Because the decision will be frequently adjusted based on the performance of the selected retransmission node at a finer scale and on the periodic probing feedback from others at a coarser scale, it can address any potential faulty decision and switch to better retransmission nodes quickly.

3.3.5 Additional Design Considerations

Although the focus of this proposed work is not on system implementation, it is worthwhile to discuss a few additional design considerations here.

Retransmission session and synchronization: A separate unicast session is used to send the retransmission packets for each receiver, a good strategy that has been discussed sufficiently in [18]. As a matter of fact, we can exactly follow the recommendations made in [18] on the retransmission payload format, association/synchronization of a retransmission stream to its original stream, use of the retransmission payload format with the extended RTP profile for RTCP-based feedback (including retransmission requests) [17], congestion control, and other considerations, keeping in mind that the retransmission session is between the receiver and the retransmission node, as opposed to the original sender.

Dynamic join and leave: When a new node joins the multicast session, it will send a "Tracert" request to the original sender. As soon as it receives the reply, it will send the results through the IP-multicast session. Other nodes who receive this message will reply with their own "Tracert" results through unicast. In a multicast session with size of n, this is an overhead of O(n) unicast operations plus one IP-multicast operation per joining node, and is done only once when the node joins. If overhead is still a concern for large n, then two strategies can be employed to reduce the unicast operations per joining node. One is random reply, with a probability depending on n resulting in a constant number of total replies, from existing nodes to the joining node. The other is to only let those existing nodes who deem, based on the information they have (e.g., path diversity or delay), that they might be a good node to reply to the joining node. When a node leaves the session, those who have selected this node as a retransmission candidate will detect it in the periodic probing process and will replace it with another good node in their candidate set.

A sudden increase of the number of users may cause the flash crowd phenomenon which often occurs in a traditional client-server network, and may still present a problem in a pure P2P broadcast network, although less severe. In the discussed scenario, a new user only needs to send "Tracert" once to the sender. The bulk data is delivered via more stable and efficient IP-multicast. So potential bandwidth bottleneck caused by the "Tracert" packets is not a significant issue here. In a normal P2P situation, according to coolstreaming [79], the peak joining rate is about 150/minute (or 2.5 nodes per second) in a system with 15000 users. So the "Tracert" packet processing complexity and communication overhead is not an issue.

Peer finding: Typically, NAT/firewall is a significant problem for a P2P streaming system. However, unlike pure P2P system, the availability of IP-multicast channel makes peer finding an easy job, as each new joining node will declare its existence by sending a message, e.g., its “Tracert” information, to the IP-multicast group. It will also learn the existences of other peers through their feedback message (e.g., in “Tracert” exchange). The limited uplink bandwidth in an asymmetric network does not pose a significant issue here either as the bandwidth requirement for retransmission is significantly smaller than regular data packet delivery.

3.4 Simulation Results

In this section, we evaluate the performance of the proposed path-diversity overlay retransmission mechanism. First, we will compare the performance of the proposed architecture based on “Tracert” only, periodic probing only, and the hybrid of them, against the traditional approach. Then in a large dynamic topology with random burst background traffic, we study the relationship between the probing interval and the performance and overhead introduced by periodic probing so that a good probing interval can be determined. We then show that selective probing will save significant probing overhead without degrading the system performance noticeably.

In all simulations, the delay constraint for the *retransmission* (referred to as the retransmission deadline, which starts from when a packet loss is detected) is set to 400 ms to reflect a moderate real-time requirement, and retransmitted packets that do not arrive within the retransmission deadline will be treated as lost packets. We only consider the case of performing only *one* retransmission per lost packet. This is to focus our experiments on the effectiveness of the proposed retransmission mechanisms. Based on the tests using random transit-stub (TS) graphs of size up to a few hundred nodes, we empirically identified a good set of parameters in Eq.(3.3), i.e. A=0.75, B=0.25, α =0.3 and β = 0.7, which performs better than any other choices we tested. When the deviation of α and β is 0.1 and the deviation of A and B is around 0.25, the deviation of the effective packet loss ratio is observed to be within 5% of the lowest effective packet loss ratio observed in the tests, which suggests that the performance is not very sensitive to the choice of the parameters. Unless otherwise specified, the periodic probing interval is 10 seconds.

3.4.1 Performance of "Tracert" and Periodic Probing

We use the network simulator2 (NS2) [80] with some add-on codes to implement our protocol. The test topology is a transit-stub (TS) graph created by Georgia Tech GT-ITM network topology generator [81]. Figure 3.4 shows one of the test topologies which we use here for ease of demonstration purpose. Results for a larger network with more dynamic background traffic will be presented in Section 3.4.2. There are 40 nodes including the routers. Most of the link capacity is 1 Mbps and some of them are 2 Mbps. The propagation delays of the links range from 5 ms to 50 ms. The queue in the router can buffer 50 packets by default. The packet drop policy used in the routers is Droptail. In our tests, there are eight nodes joining the multicast group and two of them are data senders. In particular, node 11 and node 7 both send data to the multicast group, and therefore act as both a sender and a receiver. Other active nodes, i.e., nodes 3, 6, 8, 10, 12, 13, serve only as receivers. Each of the two senders, node 11 and node 7, sends a CBR (constant bit rate) video to the multicast group. The bit rate is 400 kbps. From the 9th second, there is a CBR (about 1.8 Mbps) background traffic on the link between node 1 and node 0 that has a bandwidth of 2 Mbps, which is a main path in both of the two sender-based IP-multicast trees. For node 8, the path from sender 11 in the IP-multicast tree is 11→26→1→0→9→8, which includes the congested link. For node 10, the path is the same except the final link. But for node 3, the IP-multicast path from node 11 is 11→26→1→17→3, which does not include the congested link.

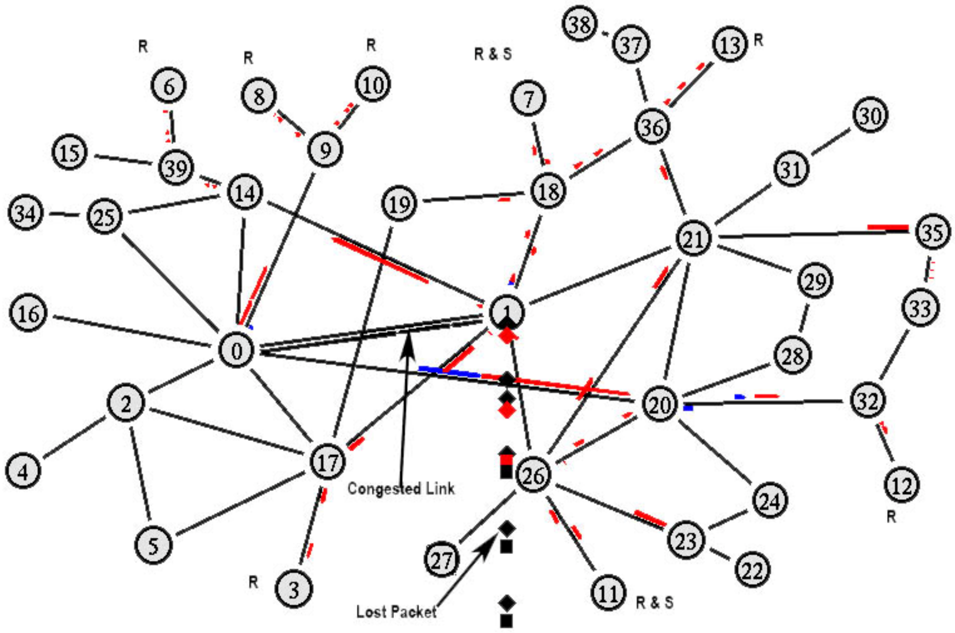


Figure 3.4 One of the test topologies with 40 nodes. "R" and "S" represent receiver and sender, respectively. "Falling dots" below node 1 represent packet dropping at node 1 (for both RTP packets and background traffic packets). Other small dots on some of the links are multicast RTP data packets.

3.4.1.1 Performance of Periodic-Probing-only based Overlay Retransmission

We first compare the proposed periodic probing based overlay retransmission approach to the conventional approach that performs unicast retransmission from the original sender.

As shown in Figure 3.5, the congestion occurs at about the 9th second when the background traffic starts to take effect. With the conventional retransmission approach, the effective packet loss ratio decreases to some extent, but is still as high as 10%. We can see that the periodic probing based overlay retransmission performs much better than retransmission from the original sender. In addition, if a physically close neighbor is selected to be a retransmission candidate because of its previous good receiving state, the measured retransmission success rate can help to address this problem by allowing another good candidate to be chosen. In Fig. 3.5, we observe a small peak for the periodic probing based overlay retransmission approach at the beginning of the congestion. This is because initially node 8 took node 10 as a good retransmission candidate because prior to congestion, node 10 was in an excellent receiving condition and is also close to node 8. Then within a short time, node 8 was able to detect that node 10 had problem too, thus changing the retransmission node to node 3 who was really in a good receiving condition. This demonstrates the adaptability of our algorithm in choosing the best retransmission node.

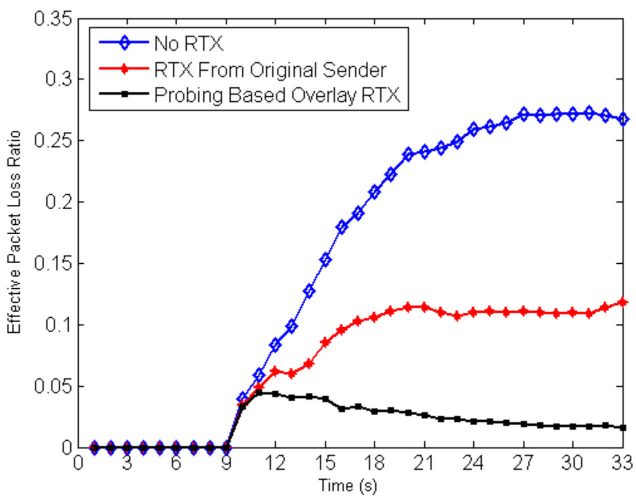


Figure 3.5 Effective packet loss ratio of node 8 for the conventional retransmission and the periodic probing based overlay retransmission.

3.4.1.2 Performance of "Tracert" Only based Overlay Retransmission

It is observed that in most cases, physically close neighbors have little chance to be a good retransmission node of each other, because they share most of the IP-multicast path and typically face the same congestion problem. Probing only based approach can avoid choosing such neighbors as retransmission candidates as discussed above, but it may take a couple of seconds for the receiver to detect the problem and correct it. We evaluate the second strategy, i.e., to choose the nodes with the most disjoint path to the sender to be the retransmission node. By taking "Tracert" results into account, the receiver can initially identify a least correlated node in terms of packet loss, so as to avoid the close neighbor selection problem in the first place.

Fig. 3.6 shows that, with "Tracert", node 8 will choose the path-disjoint node 12 as the retransmission node. From the figure, we can see that the initial peak diminishes.

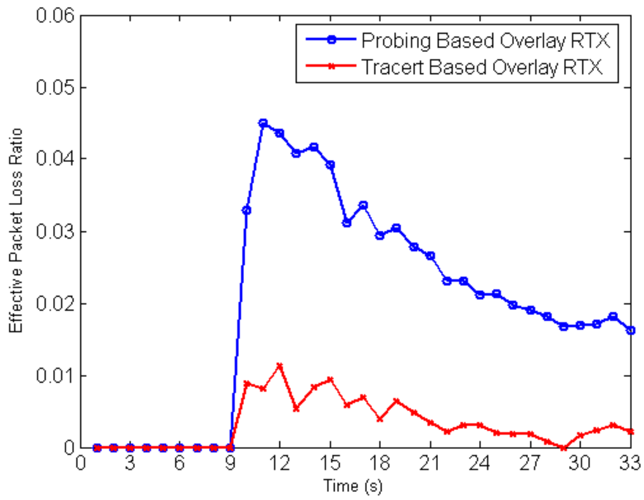


Figure 3.6 Effective packet loss ratio of node 8 for the periodic probing based and the "Tracert" based overlay retransmission.

3.4.1.3 Performance of Hybrid Overlay Retransmission

We evaluate the impact of dynamic traffic on the performance of different approaches. To simulate dynamic traffic, another traffic is injected at the 15th second on link 20→32, which will greatly affect the receiving state of node 12. Apparently, node 12 is not a good candidate any more. If only “Tracert” is used, node 8 will still treat node 12 as a good candidate until it detects the problem based on the retransmission success rate. If only periodic probing is used, it cannot overcome the close neighbor selection problem. If we integrate probing information with “Tracert”, it can dynamically determine an overall best node as the candidate retransmission node, since it will avoid the close neighbor selection problem based on the “Tracert” information and select a node with disjoint path that has good receiving state. For this particular example, node 8 will choose node 3 as the best candidate. Fig. 3.7 shows the effective packet loss rate of node 8 for different approaches. It can be seen that the hybrid approach is very effective in finding a good retransmission node. The effective packet loss rate is reduced to almost zero. The “Tracert” only based approach, on the other hand, is not able to adapt to the dynamics of the network sufficiently. When the second background traffic takes effect at the 15th second, the performance suffers. Although the receiver did detect the problem based on the retransmission success rate and switch to other candidates, it has difficulty in finding a good retransmission node in a timely manner.

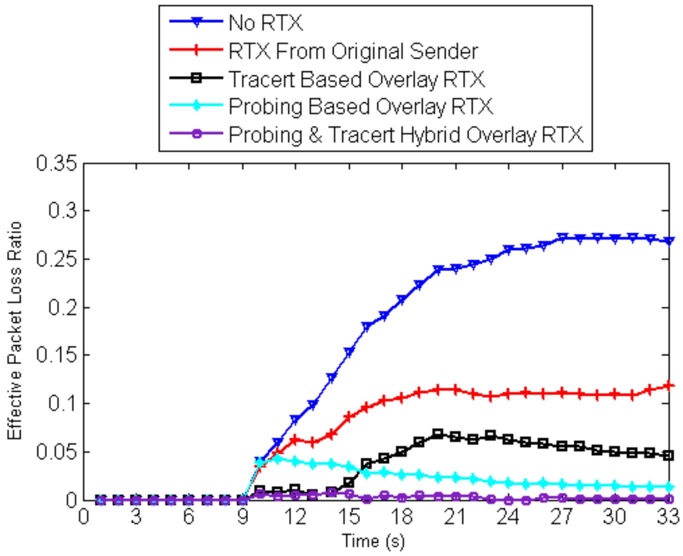


Figure 3.7 Effective packet loss ratio of node 8 for different approaches when background traffic is more dynamic.

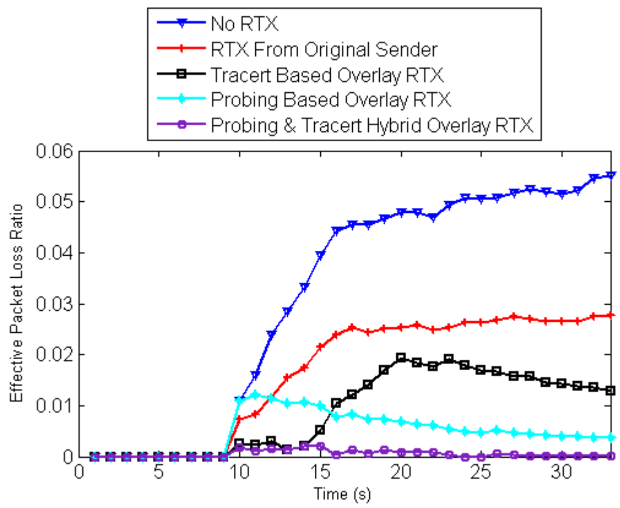


Figure 3.8 Average effective packet loss ratio for all nodes for different approaches when background traffic is more dynamic.

Fig. 3.8 shows the average performance for all nodes in the multicast group. We can see again that "Tracert" only based approach cannot adapt to the network dynamics well. The periodic-probing-only based approach suffers from the close neighbor selection problem discussed before. In fact, it may even perform worse than the conventional approach occasionally as shown around the 10^{th} second. On the average, the hybrid approach performs the best as it takes both path diversity and dynamic receiving status into account.

We also observe that when the retransmission is from the original sender, the average delay introduced by retransmission is 332 ms. With the hybrid overlay retransmission, the average delay introduced by retransmission is reduced to 145 ms. In summary, the simulation results show that our proposed retransmission architecture can significantly decrease the effective packet loss ratio, at the same time significantly reduce the retransmission delay.

3.4.2 Scalability and Probing Overhead

In order to better understand the performance of our proposed architecture, we do several tests to show (1) the proposed architecture works well for large scale topologies with more dynamic traffic; (2) appropriate probing interval can be determined to achieve a good performance without excess probing overhead; (3) selective probing can be used to save the periodic probing overhead without sacrificing the performance.

In the following tests, GT-ITM is used to generate a 500-node TS topology. The average degree of each node is 1.9. In these tests, there are 40 nodes joining the multicast group and two of them are sending data to the group. The simulation runs for 100 seconds. The links' delay is distributed uniformly from 1 ms to 20 ms, the backbone link bandwidth is 10 Mbps, and the bandwidth of edge/last links is distributed from 1 Mbps to 2 Mbps. We use distance vector for IP-multicast routing and shortest path for unicast routing. To simulate the dynamics of the network, we randomly generate 400 exponential burst background traffics by NS-2, each traverses two randomly chosen nodes and has an average bit rate distributed from

500 kbps to 5 Mbps. The random background traffic is injected starting from the 5th second, and its duration is Gaussian distributed from 1 to 100 seconds.

3.4.2.1 Performance in a Large Topology with more Dynamic Traffic

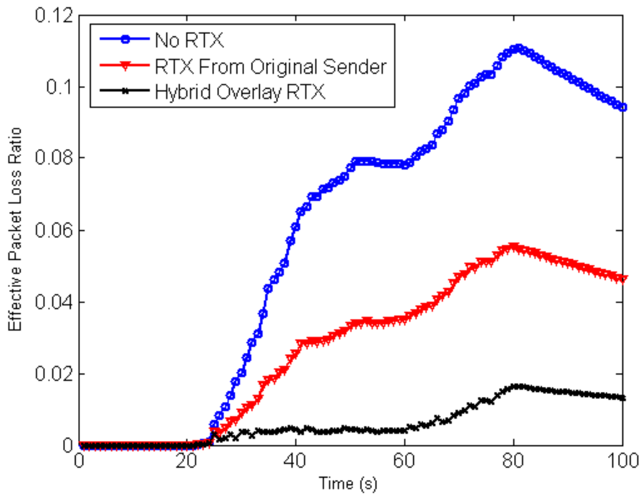


Figure 3.9 Average effective packet loss ratio for all nodes for different approaches in a large topology.

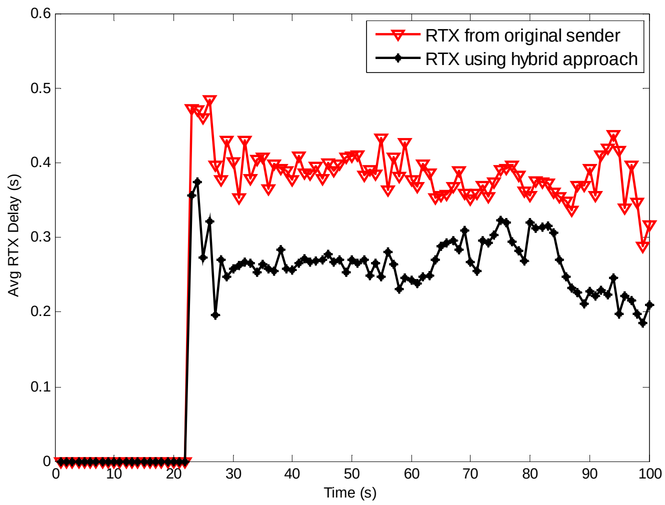


Figure 3.10 Average retransmission (only received retransmitted packets are considered here) delay for all nodes for different approaches in a large topology.

Figure 3.9 and Figure 3.10 shows the average performance of the proposed hybrid approach for the large topology compared to the traditional approach that performs retransmission from the original sender. We can see that the hybrid approach again significantly outperforms the traditional approach, and can recover about 85%-90% lost packets on average with only one retransmission attempt. This result demonstrates that our proposed architecture can scale well in large networks. We also observe that the average delay introduced by retransmission is reduced from 361 ms for the conventional approach to 238 ms for the hybrid overlay retransmission.

3.4.2.2 Determine the Appropriate Probing Interval

In general, more frequent probing will gather more accurate information about the network dynamics, but at the same time, it will incur more overhead which will impose more stress to the links and potentially may worsen the system performance. Based on the above simulation setting, we set the probing interval to be 2 seconds and 10 seconds respectively to study the retransmission performance.

Figure3.11 shows that the smaller probing interval has similar performance as the 10 seconds probing interval. However, the probing overhead is about 5 times as large. Our experiments show that a probing interval of 10 seconds is a reasonably good choice to achieve good retransmission performance. The overhead introduced is negligible compared to the multimedia data traffic.

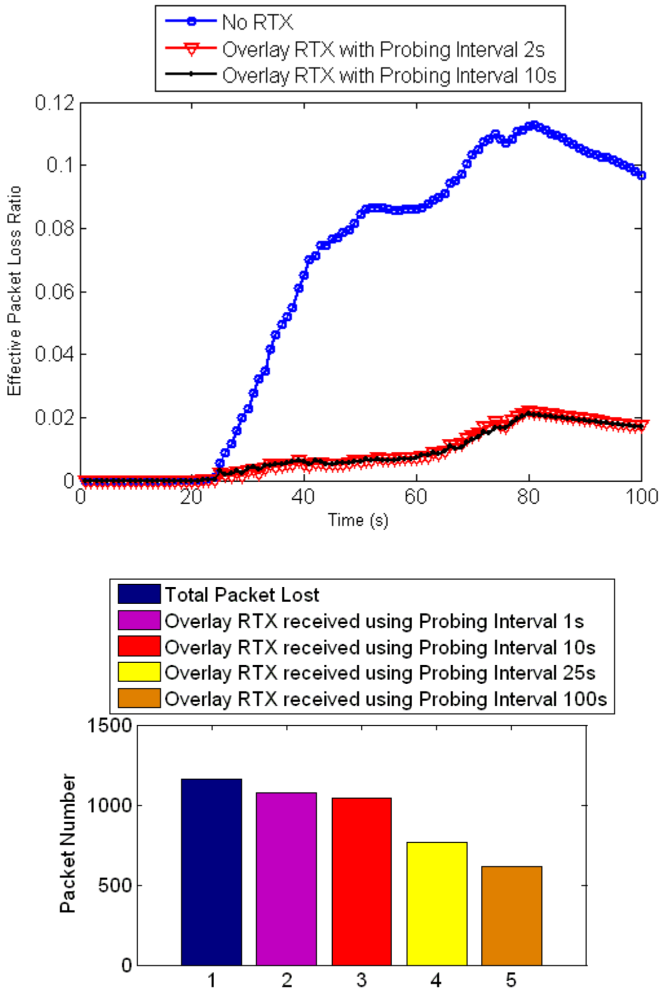


Figure 3.11 Performance comparison between different probing intervals in hybrid overlay retransmission

3.4.3 Selective Probing

As discussed in Section 4.3, with a constant number of *C* probing targets, the receiver can randomly select *C* nodes in the multicast group to probe. This random probing may not be efficient as it may probe the bad candidates therefore missing the chance to probe more good candidates. The basic idea in selective probing is to give the current best candidates more weights when selecting the probing targets. Each receiver will maintain a sorted retransmission candidates list and exclude those nodes whose delay does not satisfy the delay constraint (see Eq.(3.3)). In each probing period, the receiver selects several (6 in the experiments) best candidates to probe. For other candidates, to get their updated information, the probing interval is 3 times as long as the probing interval of the best candidates.

Figure 3.12 shows that selective probing, with much less probing overhead (i.e., only 6/40 of that of full scale probing), can achieve similar performance as the full scale probing. In addition, in scenarios with more dynamic random background traffic such as in this example, selective probing outperforms random probing significantly with the same probing overhead. In all three cases, the proposed hybrid scheme is used, with the only difference being the probing method.

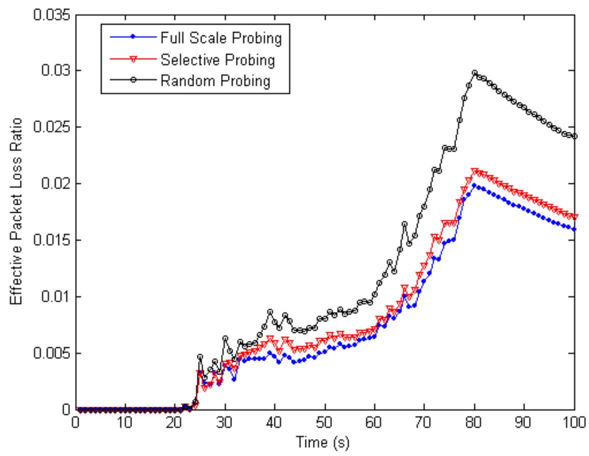


Figure 3.12 Performance comparison of selective probing, random probing, and full scale probing.

3.5 Summary

In this chapter, a novel path-diversity overlay retransmission architecture for IP-multicast based multimedia applications is proposed. In this architecture, peers are helping each other to retransmit the requested data that is difficult to get from the original server. A readily available network utility function "Tracert" is used to help identify the path disjoint retransmission nodes, and periodic probing is employed to adaptively and more accurately identify an overall good retransmission node. A hybrid approach that exploits both has been shown to achieve the best retransmission performance. Furthermore, to save the probing overhead, the receiver can use selective probing to probe only a subset of the candidate retransmission nodes who have the highest probability to be a good candidate. Simulation results show that it can significantly improve the reliability, delay, and scalability of IP-multicast based multimedia applications.

Chapter 4. Efficient Routing for Video Streaming Over Wireless Mesh Networks

High throughput path is essential for video streaming over multi-hop wireless networks. In WMNs, the performance highly depends on the quality of the wireless links. To explore the high quality video streaming service in a cooperative manner, it is necessary to first look into the routing layer in WMNs, which has its own unique characteristics and is different from the wireline networks. This chapter proposes a radio and load aware routing metric that can help to select high throughput path for video streaming in WMNs.

4.1 Overview

In WMNs, a single-hop or multi-hop path needs to be selected to forward data packets from a source node to a destination node. The routing protocol and routing metric are critical design factors for performance optimization in WMNs. They need to be carefully chosen especially when transmitting high rate streaming data, such as video. Many routing protocols such as AODV [83], DSR [84], and OLSR [85] have been developed. The implementation in this dissertation is based on the AODV because of its efficiency, maturity and also because AODV will be the basis of the default routing protocol in the emerging IEEE mesh network standard IEEE 802.11s [90].

Several public domain AODV implementations such as AODV-UU [86] and AODV-UCSB [87] exist. They are similar in that they both use a user space daemon in cooperation with kernel space functions. In addition, AODV-UU implemented several other features such as multiple interface support and a basic gateway. But the problem is that all of these functionalities are based on the traditional AODV protocol, which presents the following issues:

First, the minimal hop count is used as the routing metric. The path with the minimum number of hops is selected to forward the data packets. However, minimal hop count paths do not take into account the link transmission rate, link quality and available bandwidth along the path. It has been shown that this metric is not efficient in many situations because it tends to include lossy and unstable radio links with a long physical span, incurring a number of retransmissions and a low physical layer data rate. Many radio transmission systems, for example IEEE 802.11 radios, adapt the physical layer data rate depending on the link quality so the transmission rate will be reduced greatly when the distance is large. Selecting a path with minimum hops may actually result in poor throughput and delay as well as reducing the efficiency of the network utilization compared with selecting a path with more hops but better link quality. In this chapter, we propose and describe an implementation of a radio and bandwidth aware metric to improve the routing performance.

Second, there is route discovery delay in AODV. AODV is an on-demand routing protocol using a Route Request (RREQ) and Route Reply (RREP) mechanism to establish the route between the source and destination. The route discovery is activated by the source only when the source wants to transmit data. Although the on-demand routing protocol may reduce the routing overhead, it incurs route discovery delay and the data have to be buffered at the source to wait for an established path. The route discovery delay is even more detrimental when an active path breaks or a station is handed over from one mesh access point to another. In our implementation, we propose a method to reduce the route discovery delay while still able to discover the best route.

Third, in the traditional AODV, once a route is selected to be the best path, it will probably be kept in use until it fails/breaks. But in a real environment, this route may no longer be the best route, because of changes in the network topology, routing metric, etc., and because other better paths may become available. So we need a mechanism to re-discover the up-to-date optimal route.

Finally, there is no mesh proxy function in AODV. To use mesh access point to work as a gate for the clients associated with it, a more flexible mesh proxy function is needed at mesh routers/access points. Only the mesh access points need to run the routing protocol. They should support all kinds of clients/stations that are not equipped with the routing functions and do not have knowledge of the mesh networks. The mesh access point with the proxy function should maintain the route for each station that is associated with it.

Consider the above factors, we design a radio and bandwidth aware metric to improve the routing performance. The metric captures the impact of various aspects of wireless links in mesh networks. The value of the link metric may change frequently due to dynamics of mesh networks, which may lead to route instability. A quantization method is also proposed to maintain the route stability while achieving quick response to the changes of link state and network topology. Furthermore, in order to improve the performance, several enhancements for the AODV routing protocol are implemented, including (1) a new method for processing RREQ messages and generating RREP messages so that the best path can be discovered without incurring significant route discovery latency. (2) Periodic maintenance to discover the most up-to-date optimal path. In addition, a proxy function in mesh access point is implemented so that the stations can access the mesh through the access point without any modification to themselves. Using the prototype, it will be shown that the video quality can be significantly improved over the traditional minimal hop based routing protocol.

4.2 Related Works

To improve the traditional hop count metric, in [88], a metric called "expected transmission count" (ETX) has been proposed. The ETX estimates the expected number of MAC layer transmissions needed to successfully deliver a packet through a wireless link. The path with the lowest sum of the ETX's of all links along the path (the path with the lowest ETX cost) is selected. The ETX captures the effects of packet loss rates on a link but does not take different transmission rates and the available bandwidth of the links into account. A metric called "expected transmission time" (ETT) has been proposed to improve the ETX by considering the differences of link transmission rates in [89]. The ETT of a link is defined as the expected MAC layer duration for a successful transmission of a packet on the link. The cost of a path is the sum of the ETT's of all links along the path. The ETT takes the impact of different link transmission rates into account. However, it does not fully capture the impact of the traffic load and the available bandwidth of the link as well as the interference in the networks due to the shared medium.

4.3 Metric Design and Routing Algorithm Enhancement

4.3.1 Radio and Bandwidth Aware Routing Metric

The unique characteristics of mesh networks and streaming video support, such as the shared nature of the wireless medium and flow-type traffic pattern impose unique requirements on designing routing protocols and metrics. A new radio and bandwidth aware routing metric is designed to fit these requirements. The metric captures the impact of various aspects of a wireless link in mesh networks, including the radio transmission rate, loss rate, traffic load and available bandwidth of the link as well as the interference due to the shared medium in the networks. The value of the link metric may change frequently due to dynamics of link quality and traffic load, especially when the traffic load on a link is considered in the metric, which may lead to route instability. It is important to ensure the route stability with good routing performance even in the face of rapidly changing link quality and load variations. Therefore a method is also proposed to achieve quick responses to link state and network topology changes while maintaining the route stability by quantizing the routing metric.

Let T_{oh} denote the protocol overhead at the medium access control and physical layers, and L denote the test packet size. Given a radio transmission system, T_{oh} is a constant, for example, 185μs for 802.11a radio and 699μs for 802.11b radio [90]. The size of test packet L can be a pre-configured constant, for example, 8224 bits. Furthermore, let R denote the link bit rate at which the node transmits a packet of the standard size L under current channel conditions. The link bit rate depends on the local implementation of the link rate adaptation. Let E_r denote the packet error rate if the node transmits packets of the standard size L at the transmission bit rate R. E_r can be measured and estimated by a node in the mesh network locally. Let ρ denote the load/utilization of the channel that is related to the available bandwidth on the channel. The weighted radio and bandwidth aware (WRABA) routing metric for a radio link can be calculated as

$$WRABA = (T_{oh} + \frac{L}{R}) \times \frac{W_1(\rho)}{1-\rho} \times \frac{1}{1-E_r} \qquad (4.1)$$

where $W_1(\rho)$ is a weight function, depending on ρ. Some possible forms of $W_1(\rho)$ are

(1) $W_1(\rho) = 1$

In this case, all links are weighted equally.

(2) $$W_1(\rho) = \begin{cases} 1 & \rho \le \rho_0 \\ K_1 & \rho_0 < \rho \le \rho_{\max} \\ \infty & \rho > \rho_{\max} \end{cases}$$

In this case, links with channel load less than ρ_0 are weighted equally. Links with channel load between ρ_0 and $\rho_{\max}$ are weighted with an integer K_1. Links with channel load greater than $\rho_{\max}$ are not considered in the path selection because their cost is infinite.

The WRABA link cost function represents a composite routing metric, which takes the radio resources consumed by sending a packet over a particular link and the load/available bandwidth on the link into account. Both the link quality and load varies so the value of WRABA changes frequently. If WRABA is used directly as the routing metric, the path may change frequently, leading to route instability. In order to achieve quick response to the link state and network topology changes but also to maintain the route stability, a quantized version of WRABA is used as the link cost function. The quantized WRABA (QWRABA) can be in the form of

$$QWRABA = \begin{cases} Ceiling(M \times WRABA / Q) & Ceiling(M \times WRABA / Q) \leq M \\ M+1 & Ceiling(M \times WRABA / Q) > M \end{cases} \tag{4.2}$$

where M is the number of quantization levels and Q is the quantization factor. Generally, the system designer can choose a suitable Q depending on some targeted tradeoffs of route stability and network response time to link state and network topology changes. In order to use a limited number of bits (a fixed size field) to represent the value of QWRABA, the value of QWRABA can be truncated to M+1 if it is larger than M+1.

A node can estimate the load of the channel to each of its neighbors. One possible method to estimate the channel load is to use the channel busy time. Due to the shared nature of wireless channels, the channel busy time is defined as any time when any node within the interference range performs transmission on the channel. When a node uses the channel to transmit a packet, this channel is busy and other nodes within the interference range cannot simultaneously transmit using the same frequency. If another node attempts to transmit during this busy time, a collision occurs and the transmitted packets of both nodes suffer errors. If the estimated channel busy time is T_{busy} during a measurement period T_p, the channel load is $\rho = T_{busy} / T_p$. Channel busy time is the time during which either the physical carrier sense or network allocation vector (NAV) indicates channel busy. When the QWRABA link cost metric is used, the path with the lowest total sum of the QWRABA of all links along the path is selected. In the implementation, we add a new field called *Pcost* in the routing table. The RREQ and RREP messages in the AODV are also extended to include a metric field to carry the metric information.

4.3.2 AODV Enhancement

AODV enhancements are implemented in this prototype. A new method for processing RREQ messages and generating RREP messages is proposed so that the best path can be discovered without incurring significant route discovery latency. Periodic route maintenance is also implemented to adapt to the up-to-date optimal path.

The route discovery latency is especially detrimental when an active path breaks or a station is handed over from one mesh access point to another. To reduce the route discovery latency, in one mode of the traditional AODV operation, an intermediate node with a valid route to the destination node may respond to the RREQ with an RREP message. The RREP message is sent back to the source node in unicast and establishes a forward route to the destination node. If a flag in the RREQ is set, this intermediate node also unicasts a gratuitous RREP to the destination node so that the destination node learns the route to the source node. However, in the traditional AODV, if an intermediate node generates a RREP (because it has a valid route to the destination node), it discards the RREQ. With this approach, although the source node can find a route to the destination node more quickly, the best end-to-end route may not be found because the route cached in the intermediate node may not be the up-to-date best route to the destination. The metrics of the path may have changed due to the dynamics of wireless networks, making the cached route less desirable.

In order to discover the best route without incurring significant route discovery latency, we enhance the AODV in our implementation. When a source node S wants to discover the route to a destination node D, it broadcasts a RREQ with the destination node D specified in the destination list and the metric field initialized to 0. The RREQ message contains a new flag 'F' for each destination node. The source node sets the flag 'F' corresponding to the destination node in the RREQ when it initiates the RREQ flooding to discover a route to the destination node. Each node may receive multiple copies of the same RREQ that originated in S and traversed a unique path from S to the node. When a node receives a RREQ it creates a route to S or updates its current route if the RREQ is fresh enough and is traversed through a route that offered a better metric than the current route to S. If a route is created or modified, the RREQ is also forwarded (re-broadcasted). Whenever a node forwards a RREQ, the metric field in the RREQ are updated to reflect the cumulative metric of the route to the RREQ's source from that the forwarding node. During the RREQ flooding, the first intermediate node with a valid route to the destination node responds to the RREQ with an RREP message if the flag 'F' is set. The RREP message is sent in unicast towards the source node and thereby quickly establishes a temporary forward route to the destination. Thus, the source node can use this temporary forward route to send data packets with low route discovery latency. Furthermore, instead of sending a unicast gratuitous RREP to the destination node, the first intermediate node changes the flag 'F' in the RREQ message and forwards (re-broadcasts) the updated RREQ message downstream towards the destination node. The reason to change the flag 'F' (after the RREP message is sent) is to suppress any RREP messages from subsequent intermediate nodes for the destination. Along the route traversed by the RREQ message, only the first intermediate node with a valid route to the destination node replies with a RREP for this destination node. Since the flag in the RREQ has been changed, the downstream intermediate nodes would not respond to this RREQ and only propagate it even if the downstream intermediate nodes have a valid route to the destination node. The RREQs eventually reach the destination node. The destination node can select the best route based on the end-to-end metrics and send a new RREP back to the source node to establish the best route between the source node and the destination node. If the best path is different from the temporary forward path that was established via the RREP from the intermediate node, the source node will switch to the best path once the best path is established. Note the difference between our method and the traditional AODV is that the RREQ continue to be forwarded after the intermediate node sends the RREP. Therefore the best path can be discovered in our proposed way.

Due to the dynamics of a wireless environment, it is possible that the initial best metric route established may become worse or other routes may become available that provide a better end-to-end metric. However the traditional AODV protocol continue to use a discovered route until it fails/breaks, which results in performance degradation, especially with a long-term video streaming session. Each active source node sends a periodic RREQ message (maintenance RREQ) for the destination(s) that it is communicating with. Maintenance RREQs enable the nodes to adapt to the changes in the network conditions and help maintain the best metric routes between nodes. The time between two consecutive maintenance RREQs is referred as a refresh-round. Route-maintenance is driven by the source node and the rest of the nodes do not need to do any special processing of control packets during the route maintenance phase. Specifically, processing of maintenance RREQs is the same as processing of the RREQs generated during the route discovery phase. Note that the flag 'F' is not set at the source in the maintenance RREQ so the intermediate node will not respond the maintenance RREQ with the RREP and only propagate it even if the intermediate node has a valid route to the destination node. The reason is that a valid route from the source to the destination has existed at the route maintenance.

In some cases, when a RREQ propagates through the network, a node that already has the best metric route to the RREQ originator may learn lower quality metric routes to the originator before receiving information through the current best metric route. Hence, a cache may be used to improve route stability by not immediately switching to a worse metric route than the best-known route. It allows a good route to continue to be used for a period of time when a RREQ message that would normally propagate along the best path is lost or when a new worse metric RREQ is seen before the best metric RREQ is seen during a particular refresh round. Note that the source node can generate one RREQ per destination, and to reduce control traffic in the network, it may generate one RREQ and include multiple destinations.

4.4 Experimental Results

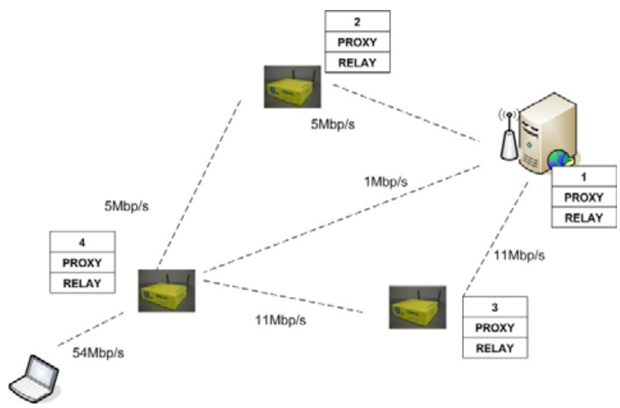


Figure 4.1 Experiment setup

The experimental results are gathered from the following setup as shown in Figure 4.1. The testbed is composed of 5 nodes to form a basic mesh network. Each of nodes is within the transmission range of other nodes. There are four nodes running the enhanced AODV routing algorithm at the IP layer and node 4 also works as a proxy to provide the access to the mesh for the laptop (node 5). The relay nodes are equipped with two mini-PCI 802.11a/b/g wireless interfaces. All of the nodes use Linux OS with the kernel version 2.6.9. Among the mesh nodes, we create an 802.11b network in channel 1. For simplicity, we manually configured the link rate of each link used by the mesh nodes. The link rate between the video server (node 1) and the proxy node (node 4) is set to be 1Mbp/s, the rates of the link between node 1 and node 2 and the link between node 2 and node 3 are both set to 5.5Mbp/s, and the rates of the link between node 1 and node 3 and the link between node 3 and node 4 are both set to 11Mbp/s. In our initial implementation and experiments, we only take the impact of the link rate into account and do not consider the traffic load and link loss rate. These factors will be investigated in the future. The station associates with the proxy in an 802.11a network using channel 40. The media file we use is an MP4 file compressed by H.264 codec with a frame rate of 24 frames/s and the average data rate is about 950kbp/s.

The laptop (node 5) starts a streaming request to the server from 0 second and we will turn on node 2 at around 50 seconds and turn on node 3 at around 70 seconds. The whole streaming session will last 120 seconds. The route refresh interval is set to be 10 seconds.

Figures 4.2 and Figure 4.3 show the total packet loss number and jitter experienced at the receiver within the test period, respectively. The red line shows the status when we just use the traditional AODV algorithm, and once the video streaming session is started, it would keep using the one hop link 1<->4. We can see that during the streaming session, because the average video data rate is close to the available link rate and the required link rate for the video streaming will actually be higher than the available link rate 1 Mbps at most of the test time due to the protocol overhead and variable video bitrate. There is a continuous data loss in the session and the jitter is very large so the video quality is poor.

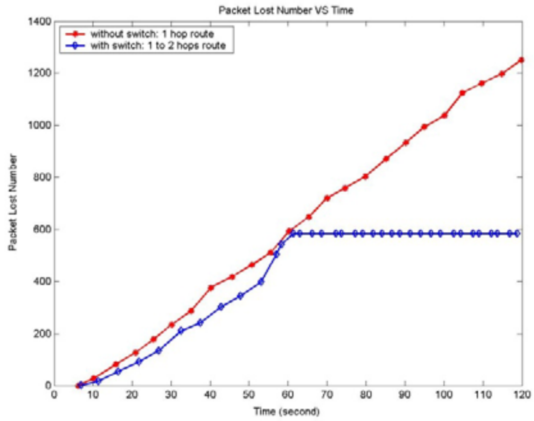


Figure 4.2 Accumulated packet loss number vs. time in two cases

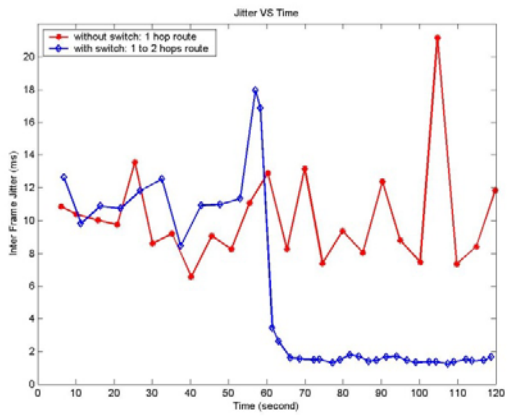


Figure 4.3 Inter frame jitter vs. time in two cases

The blue line shows the status that we turned on the relay nodes in the middle of the video streaming session. When we turned on node 2, in the next routing refresh period of the server, the server sends out the refresh RREQ and at this time, the cost of the two hops path

1<->2<->4 will be better than the one hop path 1<->4, then the server changes the route to node 4 by selecting node 2 as the next hop. In this path the effective link rate could be around 2.75Mbp/s that is higher than the data rate required by video streaming. We can see from Figures 4.2 and Figure 4.3, after node 2 was turned on, the accumulated packet loss number did not increase and the jitter in the receiver side also dropped significantly. It showed the low-cost two-hop path is much better than the high-cost one-hop path. In addition, after we turned on node 3, the route will be switched to 1<->3<->4 because it has the lowest cost. Because the previous route is already good enough to transmit the video, we do not expect to see much improvement in packet loss number or jitter. This test showed that the minimum hop count route will not be a good metric in route selection. Using radio and bandwidth aware metric is more reasonable and with periodic route maintenance, applications can always try to maintain the best route available to improve the quality of services and the whole network throughput. Besides these, the proxy node can provide a flexible and easy way to access to the mesh network for those stations which do not need to run the routing protocol.

4.5 Summary

In this chapter, we describe the design and implementation in a prototype for video streaming over wireless mesh networks. A radio and bandwidth aware metric is proposed and compared with traditional minimal hop count based routing metric. The metric captures the impact of various aspects of a wireless link in WMNs, including the radio transmission rate, the loss rate, the traffic load and the available bandwidth of the link as well as the interference due to the shared medium in the networks. The value of the link metric may change frequently due to dynamics of the link quality and the traffic load, especially when the traffic load on a link is considered in the metric, which may lead to route instability. It is important to ensure the route stability with good routing performance even in the face of rapidly changing link quality and load variations. A quantization method is proposed to maintain the route stability while achieving quick response to the network dynamics. Furthermore, in order to improve the performance, several enhancements for AODV routing protocol are implemented. In addition, a proxy function in mesh access point is implemented so that the stations can access the mesh through the access point without any modification to themselves. The initial Experimental results show that the traditional minimal hop count based routing protocol is not suitable for video streaming in the terms of packet loss and network jitter. The routing metric and enhanced routing algorithm we proposed performs much better than the tradition AODV routing algorithm.

Chapter 5. Video Streaming in Infrastructure Wireless Mesh Networks

Multimedia streaming over multi-hop Wireless Mesh Networks (WMNs) is a challenging research topic. WMNs have a number of advantages including ease of deployment, no cable cost, automatic construction among mesh access points (MAPs) and self-maintained reliability. Since WMNs differ from Mobile Ad Hoc NETworks (MANET) in that the MAPs are stationary and not power-constrained, and they can provide mesh cache services, they are more suitable for high data rate services such as video streaming. It is more difficult to do streaming over WMNs than that over the wireline Internet. There are many reasons for that, and one of them is that the wireless link quality fluctuates from time to time. The wireless link quality depends on the received radio signal strength, the packet loss ratio, the link data rate, the traffic load on the link, and the contention among nearby nodes which reduces the throughput of multi-hop path greatly as the hop count increases.

In this chapter, we propose UPAC, a unified P2P and cache framework for video on demand over the infrastructure WMNs. One of the main goals of UPAC is to explore the cooperation among all the participants including mesh content server and streaming peers. In general, the MAPs in UPAC have the capability to cache selective content, and each MAP and client in the system is willing to cache content and cooperates with others for content exchange.

5.1 Related Works

To date, most of the progresses in WMNs are focusing on providing last-mile accessibility for Internet access [60][61], and the capability for streaming services over WMNs has not been well explored. Research works to improve the performance of multi-hop wireless networks have also been done at different layers. High throughput routing metrics are well researched [62][63], and the performance of existing transport protocols and new transport protocols have been studied [64][65] to improve the throughput of the multi-hop wireless network.

Multimedia streaming is one of the applications with the most demanding requirements for network capacity and path quality because of its delay-sensitive, loss-tolerant and high quality requirements. Research on video streaming over multi-hop wireless network includes the following categories: application layer processing, transport layer enhancement, cross-layer design, cache placement/replacement and overlay cooperative approach.

Some of the previous works focus on cross-layer adaptation design or centralized multi-user resource allocation [66][67]. In [68][69], for the case of single receiver, the authors propose schemes based on path diversity and MDC to improve the robustness of the video reception, but few of them show the scalability, capability and video streaming performance with a large number of clients within a given size of WMNs. Few consider the cooperation among clients and servers either. In WMNs, One of the key tasks of mesh routers is to relay packets for the end clients. The relay cost could be very large, hurting the scalability and capability of the whole WMNs [70]. In multimedia streaming applications, the problem could be more severe because of the high data rate and the long duration of the application. In [71], the authors describe a bandwidth estimation algorithm to perform admission control over multi-rate wireless mesh networks. However, the authors did not describe how the route change of each flow will impact the entire routing process.

In [73], the authors describe a TCP-friendly transport protocol for ad hoc networks, and the idea is to use the end-to-end network measurements to detect the wireless network behaviors. However, in 802.11 wireless network scenarios, it is not sufficient to use only end-to-end metric without considering the lower layer information.

In [72], the authors propose a media-aware rate allocation algorithm that adjusts the video rate based on both video content and network congestion. According to the link state information, the video sender can change the video rate. This scheme considers the cross-layer information of the application layer and the routing layer. In [74], the proposed cross-layer design considers different control parameters across different layers, e.g., applications, networks, medium access control and physical layers. The algorithm tries to maximize the decoded video quality in multi-hop wireless network. However, most of these cross-layer approaches do not consider the cooperative potentials of MAPs and clients, thus cannot fully explore the capability of WMNs.

To use the cache capability of the MAPs, some of the research works tried to address the cache placement/replacement problems in WMNs [75][76]. Although the cache placement/replacement problems have been well studied in the wireline networks, the strategies used in wireless scenarios are different from those in the Internet. In [75], the authors proposed CacheData, CachePath scheme and a hybrid approach which explores the strategy of caching data on the transmission path only. In [76], the authors present the sub-optimal solution for caching in multi-hop wireless networks by caching data at a limited number of nodes across the network, considering the trade-off between caching overhead cost and access latency. With these caching strategies, the system performance is improved in terms of throughput and access latency. But these approaches do not fully consider the mesh topology in today's WMNs and are lack of peer cooperation.

The works about P2P streaming over WMNs include [77][78]. In [77], the collaboration among peers is studied. The authors use the scalable video coding to design cross-layer optimization strategies which allow efficient adaptation to varying channel conditions and available resources. However, this approach requires extra control overhead and the dynamic path update is not considered.

In [78], the authors studied peer-to-peer streaming over WMNs, in which a central server is used to find the best route from each client to other peers that minimizes the received video distortion. This approach requires the central server has the knowledge of the complete network connection status, which is difficult in dynamic wireless environments.

In summary, although significant progress has been made for video streaming over multi-hop wireless networks, there still lack works on cooperative caching and peer collaboration by considering the lower layer information, and the capability of the WMNs for video streaming applications are still not clearly understood. One of our contributions in this chapter is to propose a framework in which cache servers and clients are collaborating to improve the overall video streaming performance in WMNs.

5.2 Design of the UPAC Framework

In this section, UPAC, a unified P2P and cache framework for video on demand over the infrastructure WMNs is described. The framework employs both CDN and P2P techniques. UPAC has the following characteristics: (1) In UPAC, a client device can concurrently form a P2P relationship with MAP content cache servers and other peer devices as well as a client-server relationship with MAP content cache servers. (2) The MAP content cache server in UPAC supports both content streaming and P2P data downloading/fetching. It is important to note that the scheduling schemes for content streaming and P2P content fetching are different. The content streaming requires in-order delivery of streamed content. The P2P content fetching may use a different policy of dissemination among the peers. (3) Server and peer selection schemes with the consideration of cross-layer metric are performed. (4) Distributed admission control algorithm with per-flow routing is proposed. (5) Further improvement for streaming is implemented e.g. periodical path maintenance and complementary streaming from cache servers. One of the basic goals of UPAC is to use the cooperation among cache servers and peers in an interference aware manner to reduce the impact of the long and unreliable path so as to improve the overall streaming quality of WMNs.

We assume that the MAPs use 802.11 radios to construct the wireless mesh backhaul infrastructure. Selected MAPs in the mesh network have the capability to cache. Alternatively, content servers are co-located with selected MAPs in the WMNs. In the rest of this chapter, we simply call MAP content cache server, or the content server co-located with a MAP as mesh content server. The mesh content servers in the framework play two roles: VOD streaming server and peer. As a VOD server, a mesh content server can stream video to the clients as requested. As a peer, it is a downloading peer in P2P networks. The mesh content server supports two scheduling schemes, streaming and downloading. The video streaming requires in-order delivery of streamed video data. P2P downloading may use a different dissemination policy, e.g. to maximize the data availability among the peers. Client devices, if available in the mesh, also serve as best-effort peers to further reduce the consumption of network resource on the path from the source to the sink, and balance the network traffic load. A device can form a P2P relationship with mesh content servers and other peer devices. Meanwhile, it establishes a client-server relationship with mesh content servers.

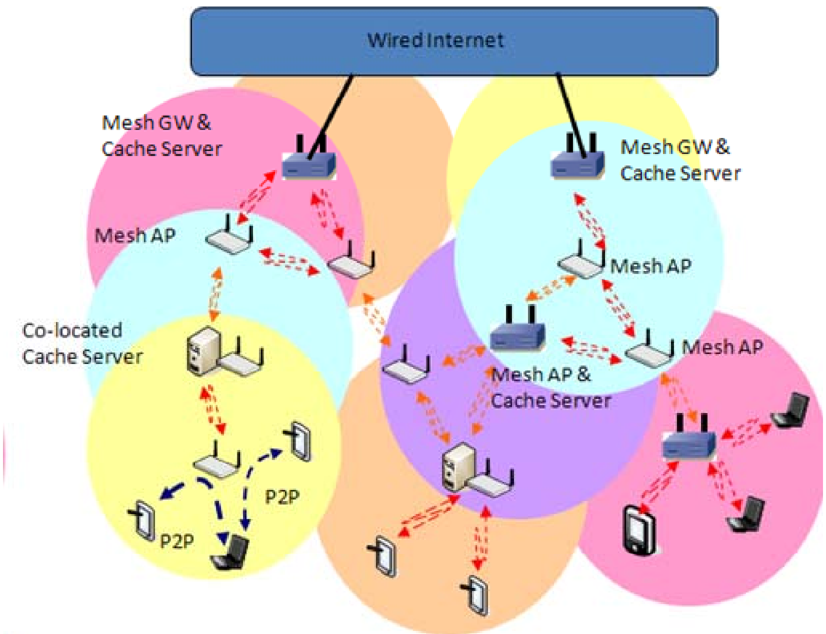


Figure 5.1 A unified P2P and cache server system for video streaming over WMNs.

Figure 5.1 illustrates an infrastructure WMNs. It consists of MAPs/routers and client devices. The MAPs are interconnected via wireless links to form a multi-hop backhaul infrastructure. One or more MAPs connect to the wired network and work as gateways. MAPs participate in routing and data forwarding too. Due to the two functions of access and relay, a MAP supports two types of wireless network interfaces. The access interface provides network access for client devices, while the relay interface is used to construct the wireless multi-hop backhaul and relay client's traffics to the destinations. In general, the two interfaces work on non-overlapped channels to avoid interference with each other. A mesh client device (e.g., laptop or dual-mode smart phone) associates with one of its nearby MAPs to access the mesh network. It does not participate in the packet relay and routing process. The device simply sends (or receives) packets to (or from) the associated MAP in the same way as what it does in a single-hop WLAN. The rest of the packet delivery is handled by the MAPs through the backhaul routing protocol.

It is assumed that there is a main content server as the original content source. The main content server may reside outside or inside the WMNs. Many schemes have been proposed to download the content to the cache servers. It is assumed that content is delivered to the mesh content servers located in the WMNs through mechanisms such as off-peak hour's delivery. It is necessary to pay attention that the mesh content servers are the MAPs with the capability of cache or co-locating with a content server.

The mesh content servers are placed according to the policy that each mesh client device can access at least one mesh content server within a few hops, because each mesh content server will serve some parts of the content to the nearby client devices and the hop count should be as small as possible. This is especially true in a single radio WMNs since the hop count affects the available bandwidth and the delay significantly. WMNs use the shared medium. In a shared medium, a flow may interfere with itself during data forwarding from one hop to the next and also interfere with other neighboring flows. In UPAC, we assume that the mesh content servers are placed appropriately.

In this framework, to incorporate the P2P downloading, the clients try to retrieve the video content through P2P overlay as much as they can while the servers devote their resource to stream the urgent data and provide better QoS.

The content file is divided into multiple equal-size segments which are denoted as clips. The playback time of the start of a clip minus a time delay D is defined as the deadline of this clip, i.e. the deadline of a clip is the time D before the playback time of the start of the clip. D is a parameter related to the network transmission and processing delay. For a different clip, a client device may have different mesh content servers and peers. The client looks each clip as an independent file and obtains the clips in their original order before their deadlines. By dividing the large file into clips, the client device can adapt to dynamic network conditions and topologies better. Different mesh content servers may cache different content or different clips of the same content. For each clip, a client device discovers the mesh content servers either in a centralized scheme via the main content server or in a distributed way, and then a primary server and a secondary server are selected. Different mesh server discovery and selection schemes will be described and compared in Section 5.5.

For P2P data fetching, similar to BitTorrent, a tracker module is used. A P2P tracker module can be hosted on a MAP or a mesh content server, or it could be an entirely separate device. It provides the P2P network directory service for the client devices. The P2P tracker for the content or content clip is known in advance by the client devices through configuration or other means. Each peer updates their status periodically with the P2P tracker so that the P2P tracker maintains the up-to-date information of the peers in the P2P network for the content clips. It should be noted that each mesh content server can run the P2P protocol and serve as a peer as well. Once its P2P fetching for a content clip is activated, a client device first issues a query to the P2P tracker. The P2P tracker sends reply messages to the client device. The reply messages contain a set of the peers that can provide the content requested by the client device. Then the client device establishes peer relationships with the selected peers to fetch/provide content to itself and other peers.

Because of the limited number of peers, scarce network and processing resource, and the dynamic that each peer may experience in WMNs, there is no guarantee that the client device can get the data in time from other peers. The client device can request the first N content clips ($N \geq 1$) streamed from one or more mesh content servers to ensure that the data requested by the client device is available and the startup delay is minimized. The client device requests the first clip (clip $i=1$) from the primary mesh content server selected by its first clip. If the primary mesh content server becomes unavailable, the client device will immediately request the first clip from its selected secondary mesh content server. Then the client requests the second clip (clip $i=2$) from the primary/secondary mesh content server selected by its second clip. This process continues until clip i ($i=N$) is received from the primary/secondary mesh content server of clip i.

Meanwhile, the client device requests and fetches other clips of content ($i>N$) from its peers and tries to use peer resources as much as possible. For the P2P data fetching of each clip in UPAC, a mechanism similar to the BitTorrent system is used. The clip is further divided into smaller chunks or sub-clips. These small chunks are exchanged (fetched or provided) among the peers. Within one clip, an example policy is that the rarest data chunks are first fetched from the peers. In our framework, an urgent data first policy combing with rarest data first policy is used for P2P data fetching. We discuss this in more detail in Section 5.7.

If the content in a clip cannot be fetched from peers before its playback deadline, the client device requests the missing data from its primary mesh content server directly. Then the primary mesh content server streams the missing data to the client in order. Furthermore, if the primary mesh content server becomes unavailable, the client will immediately request the missing data streamed from its secondary mesh content server. The secondary mesh content server streams the missing content data to the client device.

In general, a mesh content server has three main tasks. First, a mesh content server is responsible for streaming the first N clips of the requested video content to the requesting client device. Second, a mesh content server provides complementary streaming for missing data before the playback deadline of a clip. Third, the mesh content server serves as a P2P seed for content data. When a client device requests content using P2P technique, the client device takes some time to discover the peers, establish routes to the peers, locate and download the desired content. In real time applications, a long startup delay is not desirable. In addition, there is no guarantee that other peers have the requested content data. Therefore the mesh content servers are used to stream the first N clips of the content data to the client so that the startup delay is reduced. Each clip of content should be fetched before its playback deadline with the above P2P technique. Once the playback deadline of a clip is reached, no P2P fetching of the playback clip is allowed since the newly downloaded data can be outdated. Instead, the client device requests complementary streaming of miss data in the clip from the mesh content server because complementary streaming provides the content data in its original order with less latency. Complementary streaming helps the client device get the data which cannot be fetched in time from other peers.

Note that a client device only associates with one of the MAPs and does not participate in routing within the infrastructure WMN. The client device sends the peer-request packet to its peers via the MAP that the client device is associated with. When the MAP receives a peer-request packet, or any packet destined for a peer, the MAP discovers, establishes, and maintains the best route to the peer on behalf of the client device using an on-demand or proactive routing protocol based on the destination address in the peer-request packet.

To facilitate cross-layer design for improving P2P data fetching performance, we implement a proxy at each MAP. The MAP informs the associated client device of the path cost to the peers of client device and whether the peer is associated with the same MAP as the requesting client device. Therefore, the client device has the path cost information to each of the peers with which the client device can establish communications for the purpose of exchanging content. When a client device fetches data from its associated peers, the client device gives higher priority to the peers associated with the same MAP or with better path cost.

5.3 Problem Formulation for Server and Path Selection

One of the important issues that affect the video streaming quality in WMNs is the interference within the networks. There are inter-flow and intra-flow contentions around each MAP. To support high quality streaming services, it is very important to select proper servers/peers and path that can reduce the impact of interference. In this section, we formulate the server and path selection problem in WMNs and a heuristic solution will be proposed in Section 5.5.

5.3.1 Overview

A wireless mesh network can be represented by an undirected connected graph $G=(V,E)$, where V is the set of wireless nodes in the network, which includes gateway (GW), MAPs and mesh clients, where gateway is the connection between the mesh backhaul and the Internet. In graph G, all the nodes belong to $V=\{v_i \mid i=1,2,...N\}$, and $E=\{e_{ij} \mid v_i, v_j \in V\}$ is the set of wireless links. The set of MAPs is denoted as $V_M=\{v_M(i) \mid i \in \mid V_M \mid\}$, and the set of clients is denoted as $V_C=\{v_C(i,j) \mid i \in \mid V_M \mid, j \in \mid V_C \mid\}$, where $V_M \cup V_C = V$. We assume there are a fixed number of video servers that locate at the MAPs all over the WMNs, and the set of servers is Vs ($V_S \in V_M$). The set of clients associated with MAP i is denoted as $V_C(i)=\{v_C(i,j) \mid v_C(i,j) \textit{ associated with } v_M(i)\}$. For each link e_{ij}, we denote the capacity of the link as C_{ij} and the link error rate as err_{ij}.

The path from mesh content server i to client j can be defined as:

$$Pa(i,j)=\sum_{e_{uv}\in E} e_{uv}\cdot K_{uv}(i,j), \text{ where } K_{uv}(i,j)=\begin{cases}1 & e_{uv} \textit{ is on the path } i \to j\\ 0 & e_{uv} \textit{ is not on the path } i \to j\end{cases} \quad (5.1)$$

In WMNs, each MAP has two interfaces, one is the relay interface for the backhaul connection and the other is the access interface for the clients to associate with this MAP.

To avoid inter-cell interference, we assume the neighboring MAPs use orthogonal channels for access interface. Then there are two kinds of interferences in the network. They are interference between MAPs and the interference among clients associated with the same MAP. For the relay interface of each MAP, if there is a constant transmission range T_R and a constant interference range I_R, where $I_R > \alpha \cdot T_R$ ($\alpha > 1, usually \geq 2$), then an interference free transmission between node i and node j must satisfy the following constraints:

1) node i and node j are communicating through the same channel, 2) their Euclidian distance DIS_{ij} is less than the transmission range T_R , and 3) any other node k with $DIS_{ik} < I_R$ or $DIS_{jk} < I_R$ must not transmit any data when node i, j are communicating with each other.

These constraints are strict in order to achieve a collision-free scenario. According to these definitions and constraints, the interference model for the backhaul of WMNs can be modeled as a conflict graph $G'(V', E')$, in which the vertices are wireless links and are denoted as $V' = E = \{e_{ij} \mid v_i, v_j \in V\}$. To generate the edge set E', we will draw an edge between two vertices if these two wireless links cannot be used simultaneously.

For example, a connected graph and the corresponding conflict graph are shown in Figure 5.2 and Figure 5.3 respectively.

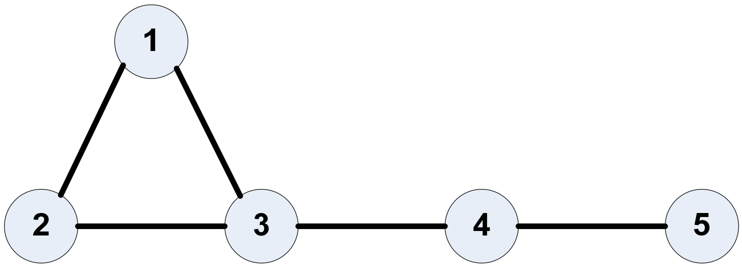


Figure 5.2 Example: original topology

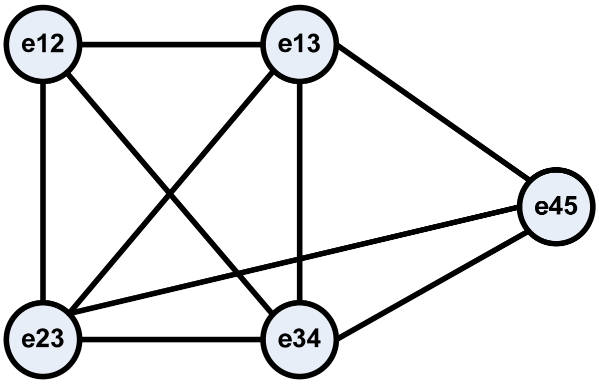


Figure 5.3 Example: conflict graph for the original topology

In this graph, there is no edge between e_{12} and e_{45}, which implies they are not interference with each other and they can transmit data simultaneously.

Because the interference happens only between neighbor edges in G', the neighborhood nodes in G' of node e_{ij} is defined as $N_{e_{ij}} = \{e_{i'j'} \mid e_{i'j'} \text{ is neighbor of } e_{ij}\}$

For video streaming applications, there is a minimal requirement for the link resources to guarantee the QoS. Because of the existence of the interference from the intra-flow and inter-flow contention, for a given network, the path needs to be carefully selected to minimize the effect of contention.

Next, we will discuss the link utilization and path assignment constraints in 802.11 wireless networks. In this discussion, we consider CBR traffic with a constant data rate $R = I * S$, where S is the size of the packet and I is the number of packets sent per second. We cannot use $\frac{R}{C}$ as the link consumption ratio, where C is the link capacity, because the protocol overhead in the 802.11 MAC layer plays an important role in bandwidth consumption. For any single hop transmission without RTS/CTS, the transmission time for a single packet with size S can be written as:

$$T_{PKT} = 2T_{plcp} + T_{difs} + T_{backoff} + T_{ack} + T_{sifs} + T_{hmac} + \frac{S}{C} = T_{oh} + \frac{S}{C} \qquad (5.2)$$

where S is the size of the packet, and $T_{backoff}$, T_{plcp} , T_{difs} , T_{sifs} T_{ack} and T_{hmac} are the time for retransmission back-off, transmission of physical layer header, DIFS, SIFS, ACK and MAC layer header respectively.

The total air time/second used by this flow will be $I * T_{PKT}$, which is also the link utilization ratio of this flow.

We define a flow from server i to client j as f_{ij} and $f_{ij}(e_{uv})$ as the traffic of flow f_{ij} on link e_{uv} . We define $f(e_{uv})$ as the aggregate traffic of all flows on link e_{uv} , and then we have $f(e_{uv}) = \sum_{i \in V_S, j \in V_C} f_{ij}(e_{uv})$. We also denote the total outgoing flow from node u as $f_{u*} = \sum_{e_{uv} \in S(u)} f_{ij}(e_{uv}) \cdot K_{uv}(i, j)$ and the total incoming flow to node v as $f_{*v} = \sum_{e_{uv} \in D(v)} f_{ij}(e_{uv}) \cdot K_{uv}(i,j)$, where $S(u)$ is the set of the outgoing links from node u and $D(v)$ is the set of the incoming links to node v.

If we assume that the packet size for all flows is the same, denoted as S, we can also define the air time of f_{ij} on link e_{uv} to be

$T_{uv}(f_{ij}) = T_{PKT}(u,v,i,j) = T_{oh} + \frac{S(f_{ij})}{C_{uv}} = T_{oh} + \frac{S}{C_{uv}}$, where T_{oh} is the radio protocol overhead and C_{uv} is the capacity of the link e_{uv}.

We assume each packet has a delay constraint τ, which is related to the network transmission delay, queuing delay and the playback buffer time. If the average queuing delay at mesh router is defined as Q, then the delay constraint can be written as

$$\sum_{e_{uv} \in E} T_{uv}(f_{ij}) \cdot K_{uv}(i,j) + (\sum_{e_{uv} \in E} | K_{uv}(i,j) | - 1) * Q < \tau \qquad (5.3)$$

Within the scope of our discussion, it is assumed that there is no multi-path routing exists. That means there is no flow splitting at each MAP. Once the decision of the server and path selection is made, the flow always originates from the server to the client follow the same path. This constraint can be written as $\sum_{e_{uv} \in S(u)} K_{uv}(i,j) \le 1$. For flow f_{ij} at each MAP, only one edge is selected for the next hop.

5.3.2 Problem Formulation

In UPAC, there are $|V_S|$ streaming servers and $|V_C|$ streaming clients. In order to achieve the goal of admitting the maximum number of concurrent streaming sessions under certain QoS requirements, each server will serve the clients through optimal selected path and each client will use an optimal selected server. The problem then becomes to finding an optimal server selection and path selection algorithm that, for a certain number of clients, the overall link usage for all the flows can be minimized. For each streaming request, not only the path, but also the best server for the request is selected. Because the future client joining is unknown, the residue resources of the network needs to be maximized or the overall network resources usage needs to be minimized at any time if more clients need to be admitted in the system. In other words, for a given number of clients, we need to minimize the overall usage of the network resources subject to certain constraints. For any mesh router node m, we define $A(m)$ as the set of clients associated with mesh router m. For any node c that is not on the WMN backhaul, $M(c)$ is the mesh router that node c is associated with. We denote $f_{souce}(m)$ as the aggregate traffic generated by servers associated with mesh router m and $f_{sink}(m)$ as the aggregate traffic that will end at client associated with mesh router m. Given a static WMNs, the network resources are fixed. All the network resource consumption is generated by traffic and interference. For a fixed number of flow requests from clients, the overall objective will be:

$$\min \sum_{e_{uv} \in E} \frac{f(e_{uv})}{C_{uv}} \qquad (5.4)$$

Subject to:

$$\sum_{i \in V_S} K_{iM(i)}(i,j) = 1 \quad \forall f_{ij} \tag{5.5}$$

$$K_{M(j)j}(i,j) = 1 \ \ \forall f_{ij} \tag{5.6}$$

$$f_{u*}(i,j) = f_{*u}(i,j) \ \ \forall u \in V / \{i,j\} \tag{5.7}$$

$$f_{source}(u) = \sum_{i \in A(u) \bigcap V_S} f_{ij} \tag{5.8}$$

$$f_{sink}(u) = \sum_{j \in A(u) \bigcap V_C} f_{ij} \tag{5.9}$$

$$f_{u*} - f_{source}(u) = f_{*u} - f_{sink}(u) \ \ \forall u \in V / \{V_S, V_C\} \tag{5.10}$$

$$\sum_{s \in V_S} f_{*s} = 0 \tag{5.11}$$

$$\sum_{c \in V_C} f_{c*} = 0 \tag{5.12}$$

$$f(e_{uv}) \geq 0 \quad \forall e_{uv} \in E \tag{5.13}$$

$$\frac{f(e_{uv})}{C_{uv}} + \sum_{e_{u'v'} \in V'} \frac{f(e_{u'v'})}{C_{u'v'}} \leq 1 \quad \forall u, v \in V / \{V_S, V_C\} \tag{5.14}$$

$$\sum_{v} \frac{f(e_{uv})}{C_{uv}} \leq 1 \quad \forall v \in A(u) \bigcap V_S \ or \ v \in A(u) \bigcap V_C \tag{5.15}$$

$$\sum_{e_{uv} \in S(u)} K_{uv}(i,j) \leq 1 \ \ \forall f_{ij} \tag{5.16}$$

$$\sum_{e_{uv} \in E} T_{uv}(f_{ij}) \cdot K_{uv}(i,j) + (\sum_{e_{uv} \in E} |K_{uv}(i,j)| - 1) * Q < \tau \quad \forall f_{ij} \tag{5.17}$$

Constraint Eq.(5.5) states that for any flow, it has to use and only use one server. Constraint Eq.(5.6) states that any flow will be end at the requesting client. Constraint Eq.(5.7) states that for each flow, at each intermediate node except source and sink, the flow is balanced. In, constraint Eq.(5.8) and Eq.(5.9), for any mesh router, the aggregate traffic generated from the server co-located with the mesh router is defined as $f_{source}(u)$ and the aggregate traffic going to the clients associated with the mesh router is defined as $f_{sink}(u)$ respectively. Constraint Eq.(5.10) is the flow conservation constraint at each MAP. Constraint Eq.(5.11) and Eq.(5.12) state that there is no traffic going to a server and there is no traffic generated from a client. Constraint Eq.(5.13) guarantees that there is no negative flow on any link. Constraint Eq.(5.14) guarantees that in the backhaul, within the interference range for any link, the accumulated link usage cannot exceed 1. Constraint Eq.(5.15) is the constraint of access link usage, which means that within the interference range of any access link, the total link usage ratio is less than 1. Constraint Eq.(5.16) is the single path constraint, which means that the traffic does not split at any MAP. Constraint Eq.(5.17) is the delay constraint for each flow.

By solving this problem, under all the constraint, we can achieve an optimal solution for a given number of streaming requests. The solution is the optimal for all the streams to select the optimal server and path. When the number of streaming requests is increased, the system will recalculate the optimal solution. Once an optimal solution is not feasible, we can say the capacity of the network is reached.

5.4 Solving the Mixed-Integer Programming Problem

In the last subsection, we formulated the problem of server and path selection with a centralized control. By solving the problem, the system could obtain the optimal solution under interference constraint, flow conservation constraint, delay constraint and etc. This problem is a mix-integer programming (MIP) problem which is known to be NP-hard.

Generally, a mixed-integer programming is the minimization or maximization of a linear function subject to linear constraints. More explicitly, a mixed-integer programming with n variables and m constraints has the form:

$$s* = \text{minimize } C^{T}x$$

$$\text{subject to} \quad l^{c} \leq Ax \leq u^{c} \qquad l^{x} \leq Ax \leq u^{x}, \; l^{x} \leq Ax \leq u^{x}$$

where $x_j \in \mathbb{Z}, \; \forall j \; is \; index$.

There are several common approaches to solve the problem:

- Branch and Bound

  Branch and bound is the most widely used method for solving integer programs. Sub-problems are created by restricting the range of the integer variables. For binary variables, there are only two possible restrictions: setting the variable to 0, or setting the variable to 1. More generally, a variable with lower bound l and upper bound u will be divided into two problems with ranges l to q and q+1 to u respectively. Lower bounds are provided by the linear-programming relaxation of the problem: keep the objective function and all constraints, but relax the integrality restrictions to derive a linear program. If the optimal solution to a relaxed problem is integral, it is an optimal solution to the sub-problem and the value can be used to terminate searches of sub-problems whose lower bound is higher.

- Branch and Cut

  For branch and cut, the lower bound is provided by the linear-programming (LP) relaxation of the integer program. The optimal solution of this linear program is at a corner of the polytope which represents the feasible region. If the optimal solution of the LP is not integral, this algorithm searches for a constraint which is violated by this solution but not by any optimal integer solutions. This constraint is called a cutting plane. When this constraint is added to the LP, the previous optimal solution is no longer valid and a new optimal solution will be generated. The new solution is different and might provide a better lower bound. The iteration of cutting planes is break until an integral solution is found, or it becomes impossible or too expensive to find another cutting plane. In the latter case, a traditional branch operation is performed and cutting planes are searched on the sub-problems.

It is important to understand that in a worst-case scenario, the time required to solve integer optimization problems grows exponentially with the size of the problem. For instance, we assume that a problem contains n binary variables, the time required to solve the problem in the worst case may be proportional to 2^n. It is obvious that 2^n is huge even if n is a moderate value. In practice, this implies that we should focus on computing a roughly optimal solution quickly but not locating an optimal solution.

The branch and bound method is used to solve the problem formulated in the last section with a certain relaxation of the termination criterion. The result of some certain test cases will be presented in the Section 5.6.3.

5.5 Heuristic Solution for Admission Control

The optimal solution requires a centralized control to select the server and the path for each streaming request at any given time. The system will recalculate the solution whenever there is a new streaming request. Due to the computational complexity of the MIP problem, it is impossible to get an optimal solution in real time. To solve this problem in the real time scenario, we propose a heuristic solution based on a distributed admission control algorithm with per-flow routing.

In the following, we first discuss the general server discovery and selection schemes to determine the general server selection strategy. With this basic strategy, we implement the distributed admission control algorithm with per-flow routing to improve the overall system performance.

5.5.1 General Server Discovery and Selection Schemes

Before the admission control algorithm is further discussed, we first describe several server discovery and selection schemes for video streaming in WMNs. We compare the following schemes. The first two schemes are used in the wireline CDN networks and the rests are schemes we propose.

- Centralized scheme with server load as the selection metric (Centralized-Load Scheme).

  In this scheme, a client device sends a request to the main server. The main server selects a primary mesh content server and a secondary mesh content server to serve this client device. It informs the client device of the selected mesh content servers. The two mesh content servers with the least load or the least number of client devices being served are selected for the client device as the primary mesh content server and the secondary mesh content server, respectively. This mechanism does not require the client device to have information about the server load and the path quality between the client and the server. However, it requires the mesh content servers to report their loads to the main server periodically.

- Overlay scheme with end-to-end delay as the selection metric (Overlay-Delay Scheme).

  In this scheme, the main server sends a list of candidate mesh content servers to the client device after the main server receives the request from the client device. The client device measures the end-to-end delay to each candidate mesh content server using probing packets. The client device selects the mesh content server with the minimum end-to-end delay as the primary mesh content server, and the one with the second least delay as the secondary mesh content server.

- Distributed scheme with hop count as the selection metric (Distributed-HopCount Scheme).

In this scheme, the client device floods the WMNs with a mesh content server request message for a content clip. Each mesh content server with the requested content clip sends a server reply to the requesting client device. Note that the client device is associated with a MAP and does not participate in routing. However, with the underlying routing protocol, the mesh content servers have information about the hop counts between themselves and the MAP with which the requesting client device is associated. There may be multiple paths available between the mesh content server and the MAP. Only the path with the minimum hop count is selected and used by the routing mechanism. Each mesh content server uses its routing layer information and informs the client device of its minimum hop count to the client's associated MAP in the server reply. The client device selects the mesh content server with the least value of the minimum hop count as the primary mesh content server and the one with the second least hop count as the secondary mesh content server.

- Distributed scheme with a routing metric as the selection metric (Distributed-Routing Metric Scheme).

  The WMNs may use a routing protocol with a specific routing metric. For example, the load-aware routing metric we proposed in Chapter 4 and the expected transmission time (ETT) are this kind of mesh routing metrics. In the comparison, we use ETT due to the public recognizing of this routing metric. We mentioned before that the ETT for a link L is defined as the expected MAC layer duration for successfully delivering a packet over the link. $ETT_L = (1/(1-e_L))*s/r_L$, where e_L is the packet loss rate, r_L is the transmission rate of link L, s is the standard test packet size (e.g. 1000 bytes). The cost of a path p is simply the summation of all the links along the path of ETT. The ETT metric captures the impact of packet loss and the link data rate on the performance of the path. The path with minimum ETT cost is used by the routing protocol. In the Distributed-ETT mesh server selection scheme, a cross-layer approach is employed. Similar to the Distributed-HopCount scheme, the client device floods a mesh content server request message over the WMNs. Through the underlying routing protocol, the mesh content server obtains the ETT cost of the best path from it to the MAP with which the client device is associated. The best path is the path with the minimum ETT cost. Each mesh content server uses its routing layer information and informs the client device of the ETT cost of its best path to the MAP with which the client device is associated in the mesh content server reply. Then the client device selects the mesh content server with the least value of the ETT cost as the primary mesh content server and the one with the second least path ETT cost as the secondary mesh content server.

In addition, a client device treats each clip of content as a separate file to adapt to the dynamic network conditions. The client device discovers and selects the primary and secondary mesh content servers for each clip independently. During the serving time of each content clip, if the primary mesh content server becomes unavailable, the client device will switch to the secondary mesh content server to obtain the content data. Meanwhile the client device will re-initiate the server discovery and selection process using one of the above schemes to identify a new secondary mesh content server. Simulation results (Section 5.6) show that the Distributed-Routing Metric Scheme outperforms all other schemes and will be used as the basic sever selection method in UPAC.

5.5.2 Admission Control and Per-flow Routing

In the last subsection, several server selection schemes are proposed and the performance comparison will be presented in the simulation Section 5.6.3. It is observed that the routing metric based server and route selection algorithm performs better than traditional centralized scheme and the schemes used in CDN. It is also shown that the distributed solution with ETT as the routing metric performs the best among all the schemes. In this subsection, admission control is introduced for UPAC.

In WMNs, it is possible that several streaming session flows go through the same path and the capacity of the wireless link is reached very quickly. Any incoming streaming request will become a burden for the network and severe packet loss may happen. All the new streaming flows will impact the quality of the existing video streaming sessions as well. Therefore it is necessary to let the MAPs perform the admission control. The MAPs will monitor the current local link load and determine whether to admit new streaming requests. With the admission control, in a heavy-loaded local network which reaches the capacity already, the new streaming requests will be denied and the quality of the existing streaming sessions is protected. By using the admission control, the network load is distributed further so that more streaming requests can be supported through the whole WMNs.

In the original AODV routing algorithm, there is no flow information stored by each MAP. In the proposed enhanced AODV with admission control, each MAP stores the information of the video flows passing through the node. As we discuss in Section 5.3, the total traffic cannot exceed the network resource within the interference range. More formally, if the up/down traffic usage around a MAP is considered, the routing algorithm can use the interference Eq. (5.14) in Section 5.3 as the admission control criteria. In the enhanced

AODV, each MAP should maintain a flow table to record the information of all the flows passing through the MAP. The information includes the unique ID of the flow, the source destination, amount of reserved bandwidth and the flow status. Each flow entry has a life-timer and all the entries are soft state. They can be automatically deleted if the timer expires. The status can be one of the three values: requesting, reserved, and activated. The extra information fields added to the routing protocol help distribute the topology and flow requirement information to each MAP, so that MAPs can make an admission decision. Upon a receiving ARP request, the source node broadcasts a route request (RREQ) to its neighbors. Four fields are added to the AODV messages and they are "Flow ID", "Rate", "Packet Size", and "$B_{upstream}$" which specifies the link rate used by the previous hop. $B_{upstream}$ is 0 at the source node since it is the first hop. $B_{upstream}$ is used by the next-hop node to calculate the aggregate link utilization required by the requesting flow. Based on the received RREQ, each intermediate node calculates the aggregate link utilization introduced by the transmissions over the last two hops and the next hop. This aggregate link utilization of the flow U_{upaggr} (aggregate link utilization) is checked against the interference constraint denoted in Eq. (5.14). If the requirement can be met, then the node forwards (i.e., re-broadcasts) the RREQ message. It also inserts an entry into its Flow Table, recording source, destination and Flow ID, It fills in the ''reserved bandwidth" field with U_{upaggr} and sets the status to ''Requesting". If the requirement cannot be satisfied, then the node silently discards the RREQ. Although in this way, it under-estimates the aggregate channel utilization introduced by the flow, it is used as the first pass to filter certain non-qualified routes. Upon receiving the route request, the intended destination sends back a route reply (RREP) if the available bandwidth is large enough. Note that multiple copies of RREQ might arrive along different paths to the destination. To increase the possibility of discovering a qualified path, the destination sends back a RREP for each copy of the RREQ. The same four fields added to RREQ are added to RREP. At each forwarding MAP node, the admission control is performed. Each forwarding MAP node calculates the aggregate utilization. Consequently, by summing up the aggregate link utilizations calculated during the RREQ and RREP phases, a MAP node can get updated total link utilization U_{upaggr} of the requesting flow and make admission decision using Eq.(5.14). If satisfied, RREP is forwarded to the next hop and this soft-state reservation is reflected in the Flow Table as ‘‘Reserved” for the status field. Only after the occurrence of the first data packet of the flow, nodes update the status of the corresponding entry to ‘‘Activated”, which means the indicated bandwidth that is actually in use. Although nodes make reservations during the RREP phase, the route may not be established successfully due to a variety of reasons, e.g. link failures and a decision of using a different route by the source. Therefore, reservations must be maintained as soft-state and are deleted when the associated timer expires. Specifically, each entry is associated with three timeout values: route-reply timeout, data start timeout, and activity timeout. The route reply timeout is used to remove an entry from the Flow Table if a node only receives a route request, but does not receive the corresponding route reply in time. The data start timeout expires if a node received a route reply for a reservation, but did not receive any data frame from the source within this period yet. The activity timeout is used to keep track of flows which have not transmitted any data frame for a long period of time. Besides the Flow Table, the routing table in each router node will also be updated. For each Flow ID, there will be an entry. In the original AODV, each MAP node manages the routing table for each (source, destination) pair. With admission control enabled, it is possible that there exist different paths for the same (source, destination) pair. If there is no flow information in the routing table, the route will become unstable whenever a new flow joins. It is necessary to differentiate different flow IDs with the same (source, destination) pair. With per-flow routing, on each MAP, each flow has its own specific route entry and the route for each flow should be maintained specifically. One concern of per-flow routing is the overhead introduced. But the routing process is done at the initial stage of each request or at the periodical route maintenance stage. Compared with the amount of video data transmitted, the additional overhead is negligible.

5.6 Simulation Results for Server Selection and Admission Control

In this section, we present simulation results to demonstrate the efficacy of the proposed server discovery and selection scheme discussed in Section 5.5.1 and the distributed admission control algorithm with per-flow routing presented in Section 5.5.2.

5.6.1 Server Discovery and Selection

In this subsection, we compare different server selection schemes.

In the simulations, the impact of multi-server vs. single-server is investigated. The performance of the server selection schemes described in Section 5.5.1 is compared. The impact of the periodical route maintenance that can keep the up-to-date route is also studied. In this simulation, we add two more schemes, one is the single server scheme and the other is the distributed routing metric scheme with periodically update.

Network simulator 2 (NS2) is used for the simulation. The routing algorithm is AODV-UU with periodical route maintenance. A 10 x 10 grid topology with 100 mesh router nodes is used. Each mesh router has one streaming client. The distance between neighboring nodes in the horizontal and vertical direction is 80 meters. For each node, the transmission range is set to be 130 meters so that each node can talk directly to all the direct neighbors. Carrier sensing range is set to be 260 meters. The transmission rate between nodes is determined by the distance. The transmission rate between the vertical and horizontal neighbors is 54Mbps, and that between the diagonal neighbors is 36Mbps. There are five servers in the simulation. The main server is in the center of the grid and other servers are located in the corners. Each video stream is sent as a CBR traffic flow with a bit rate of 400 kbps and a packet size of 1460 bytes. In this simulation, each MAP has one streaming client.

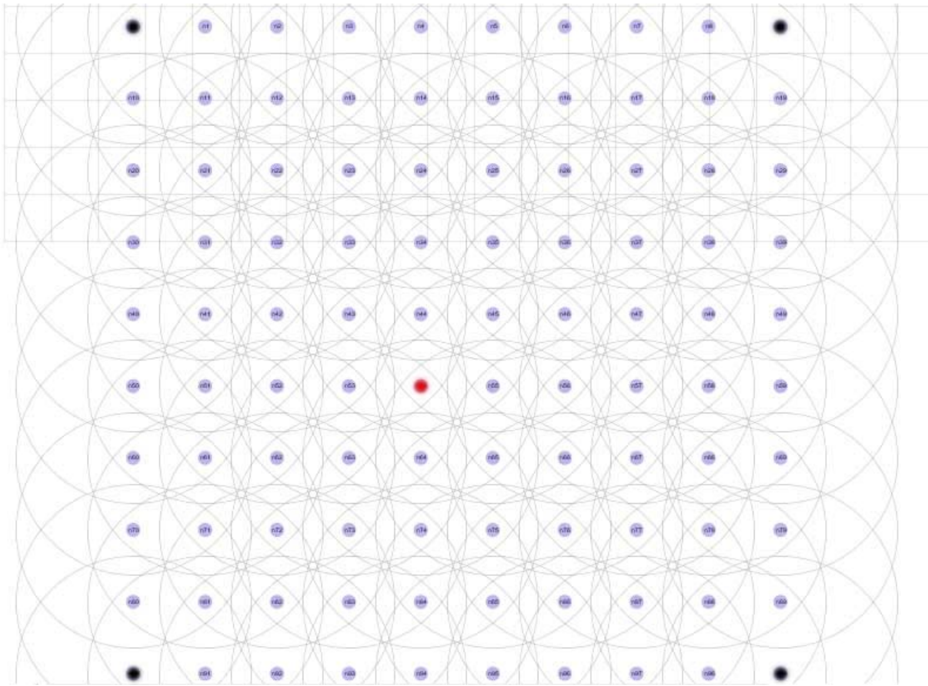


Figure 5.4. Simulation topology for server discovery and selection

Figure 5.5 shows the results of the packet loss ratio for six schemes, (1) single content server, e.g. only the main server in the center of the grid; (2) five mesh content servers with centralized-load scheme for primary and secondary server selection; (3) five mesh content servers with overlay-delay scheme for primary and secondary server selection; (4) five mesh content servers with distributed-HopCount scheme for primary and secondary server selection; (5) five mesh content servers with distributed-ETT routing scheme for primary and secondary server selection; (6) five mesh content servers with distributed-ETT routing scheme for primary and secondary server selection and periodical route updates.

Figure 5.5 shows the average packet loss ratio for all the streaming clients. There are only about 20 sessions can be supported without severe packet loss if there is only one server. By adding 4 mesh content servers, there are more streaming clients can be supported. In the common CDN schemes, e.g. scheme (2) and scheme (3), the number of clients can be supported is only about 30 to 40. Figure 5.5 also shows that the cross-layer approaches (distributed-HopCount and distributed-ETT) have significantly better performance. The schemes using routing layer metric perform better than the traditional CDN approaches. The main reason is that the routing layer metric reflects the real status of the wireless network better than the end-to-end approaches in the traditional CDN approach. The scheme using ETT as routing metric outperforms the scheme using hop count as routing metric. This is because ETT takes the link load, error rate into account which can reflect the wireless link better for the video streaming applications. In addition, because the wireless link is varying from time to time, with periodical route maintenance, the performance could be further improved and there are up to about 70 sessions can be supported.

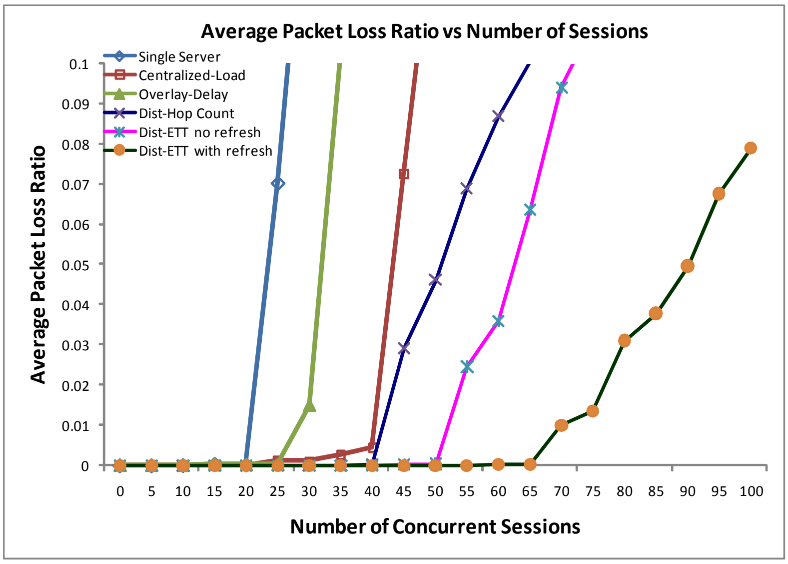


Figure 5.5 The packet loss ratio vs. the number of concurrent video sessions in the WMNs for different sever selection schemes

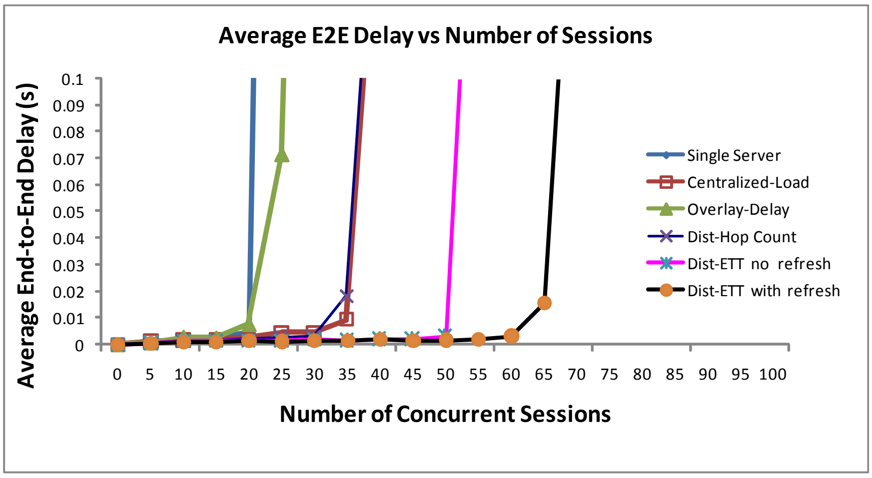


Figure 5.6 The end-to-end delay vs. the number of concurrent video sessions in the WMNs for different server selection schemes.

Figure 5.6 shows the average end-to-end delay for the above six schemes. The scheme in which the client selects the server using ETT with periodical route update also performs the best. In these simulations, the route refresh interval is set to be 10 seconds and the overhead caused by the route refresh is not noticeable.

The simulation results show that, for the high bandwidth-consuming and delay-sensitive video streaming applications, the distributed routing metric scheme performs best in selecting the best server and path. The total number of client can be supported is increased and the capacity of the WMNs is further improved for video streaming applications.

5.6.2 AODV Multi-Channel Extension

In the previous simulation, each MAP has one client, and the multi-channel property has not been stated. In this subsection, the multi-channel extension of AODV-UU is presented. In WMNs, it is necessary to make the MAP work at different orthogonal channels at the same time, e.g., one channel for backhaul relay and one channel for local client access. To reduce the interference between the access channel and the relay channel. Because current implementation of NS2 does not support multi-channel assignment, we extended the NS2 using Ramon's approach [91] to support multiple channels. However, the solution in Ramon's approach is only for DSR routing algorithm and is not suitable for AODV algorithm which is the default routing algorithm in WMNs. In order to enable AODV-UU to support the multi-channel model of NS2, we made some necessary changes to AODV-UU. With the multi-channel support, the relay channel and the access channel of mesh routers can be assigned to be orthogonal to reduce the interference. By reducing the interference, the wireless link resources can be further utilized.

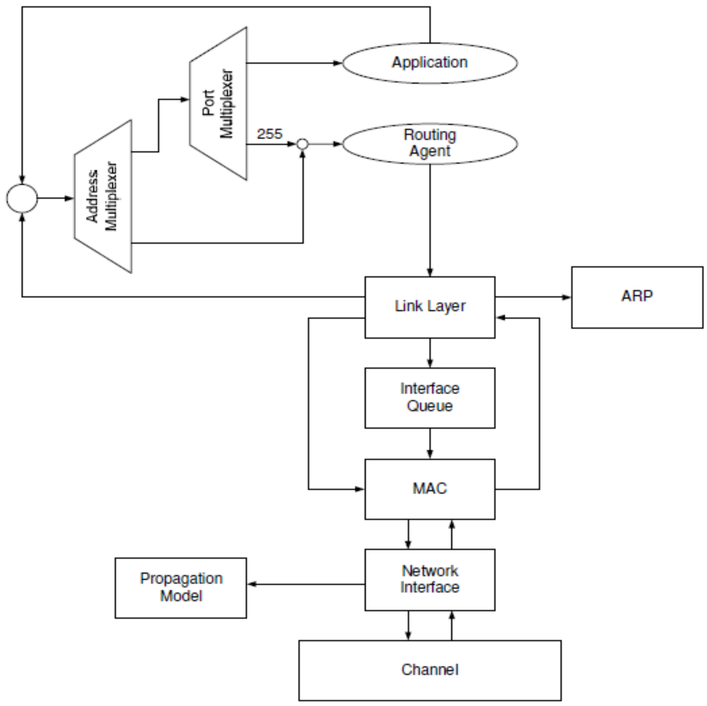


Figure 5.7. A typical mobile node structure in NS2

Figure 5.7 shows the original architecture of a MobileNode in NS2. Each node in the model has a chain of modules emulating the different protocol stack entities that any host would have in the real life: “Link Layer", “MAC Protocol", “ARP", “Interface Queue" and “Network Interface". In this model, each node has only ONE chain of module, which means at any time, each node can be assigned to only one shared wireless channel. In this case, in any single simulation session, all the wireless nodes can only use one single channel and each link will interfere with each other.

To address this problem, in Ramon’s approach, there are multiple “chains” created, shown in Figure 5.8. Each node would have as many copies of the original chain of entities (the one shown before) as many interfaces it has. In addition, the single module which is not repeated is the “Propagation Model". Because the assumption is to work exclusively with IEEE 802.11 networks, in which nodes could use more than one channel at the same time. In these circumstances, it is reasonable to use a single propagation model.

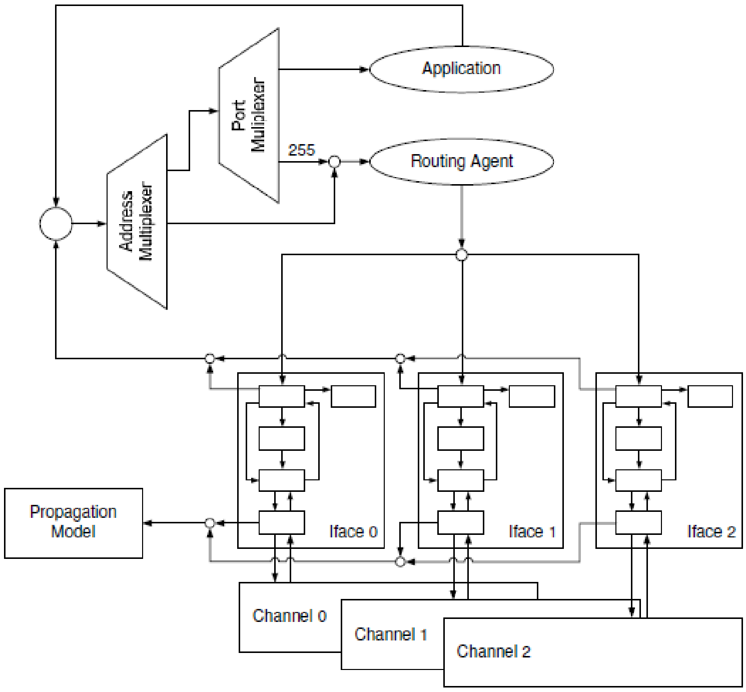


Figure 5.8 The multi-channel extension for mobile node in NS2

To support this multi-channel model in AODV-UU, for each outgoing packet, the routing agent should decide on which channel the packet should go. We modified the AODV-UU by adding a dispatcher module and some necessary interfaces for low layer components.

In all the rest simulations in Chapter 5, we use this multi-channel feature in all the nodes.

5.6.3 Admission Control With Per-flow Routing

In this subsection, we first show the numerical results for the centralized admission control by solving the problem formulated in Section 5.3. We solve the MIP problem with optimized problem solver for a topology of 49 mesh routers. To get the optimal solution, when each client joins and sends a request to the main server, the centralized controller will recalculate the server and path selection solution for all the active flows to guarantee that there is no interference among the flows and all the flows satisfy all the constraints in Section 5.3. The results show the maximum number of concurrent streams that can be supported by the WMNs.

Then, the results of the heuristics solution are shown and compared with the optimal solution in terms of maximum number of supported concurrent streams, the average end-to-end delays, the average packet loss ratio and the average throughput of each streaming session.

In the experiments, a 7 x 7 grid topology with 49 mesh routers is used. Each mesh router has 10 mesh clients associated with it. In total, there are 490 wireless nodes through the whole simulation. Each mesh router works as a gateway for each of its clients and has two 802.11 channels on it. 802.11a/g is used to simulate the wireless radio. The channels for mesh backhaul links and for the access radio of each router are non-overlapping channels, and there is no interference between them. The distance between neighboring nodes in both the horizontal and vertical direction is 80 meters. For each node, the transmission range is set to be 130 meters so that each node can talk directly to all the direct neighbors. Carrier sensing range is set to be 260 meters. The transmission rate between nodes is determined by the distance. The transmission rate between the vertical and horizontal neighbors is 54Mbps, and the transmission rate between the diagonal neighbors is 36Mbps. There are five servers and the main server is in the center of the grid and other servers are located close to the corners. The delay constraint is set to be 300ms for each flow and there is no queuing delay in the numerical analysis. The common setting of parameters for 802.11 a/ g is in table 5.1.

Table 5.1 Common parameters for 802.11 a/g

Parameters	802.11a	802.11g
T_{slot}	9us	20us
T_{plcp}	20us	20us
T_{difs}	34us	28us
T_{sifs}	16us	10us
CW_{min}	15	15

In Figure 5.9, the feasible optimal solution for the maximum number of concurrent video streams in different settings is shown. All client requests are random and the solutions satisfy all the constraints so that interference free is guaranteed. In Figure 5.9, when there is only the main server, the maximum number of concurrent streams is 29, and with 3 servers, the system can support 73 concurrent video sessions, and with 5 servers, the optimal solution can support 105 concurrent video sessions.

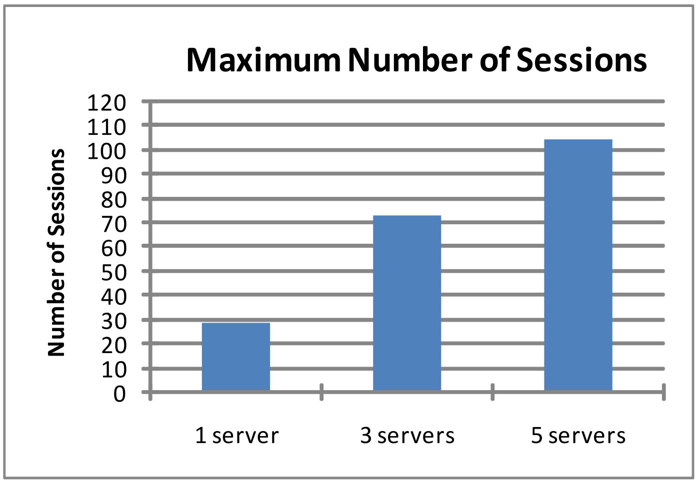
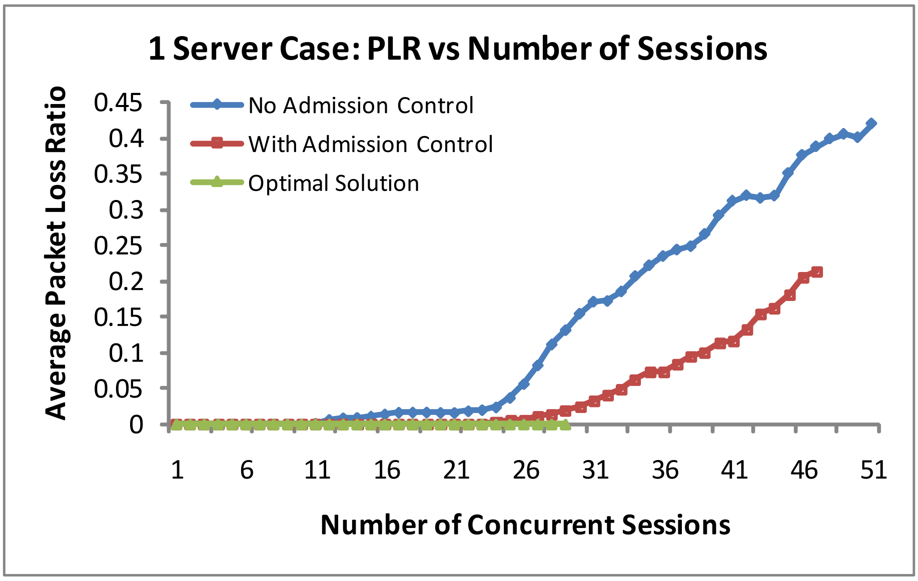


Figure 5.9 Maximum number of supported concurrent sessions

From Figure 5.10 to Figure 5.18, the simulation results in three cases with different number of servers are shown. In each case, three different scenarios are compared, which are the optimal solution, the distributed ETT server and path selection scheme without admission control/per-flow routing, and the distributed ETT server and path selection scheme with admission/per-flow routing. In the distributed ETT solution, the client selects the server according to the routing layer metric (ETT). In the previous subsection, this solution outperforms other overlay or distributed solutions. Recall that the distributed ETT solution without admission control does not explicitly consider interference around routers.

Figure 5.10, Figure 5.11 and Figure 5.12 show the average packet loss ratio, the average end-to-end delay and the average throughput for all concurrent video session respectively. The x axis is the total number of concurrent video sessions. These figures show that using distributed ETT selection scheme, without admission control, the system can establish about 53 video sessions. After about 15 sessions have started, the average performance decreases dramatically in terms of packet loss ratio, end-to-end delay and throughput. Without admission control, the route establishment is in a best-effort way and the system cannot support most of the requests. In the scenario with admission control, the system can support about 25 video sessions and the total number of admitted video session is about 47. With admission control, some of the streaming requests will be denied and only the request that will not cause severe burden to the network will be admitted. Even with admission control, there is still packet loss and long end-to-end delay. This is because the admission control mechanism estimates the interference around the MAP itself only and there is a gap between the accurate interference and the estimated one. More accurate estimation will cause more traffic overhead within the network.



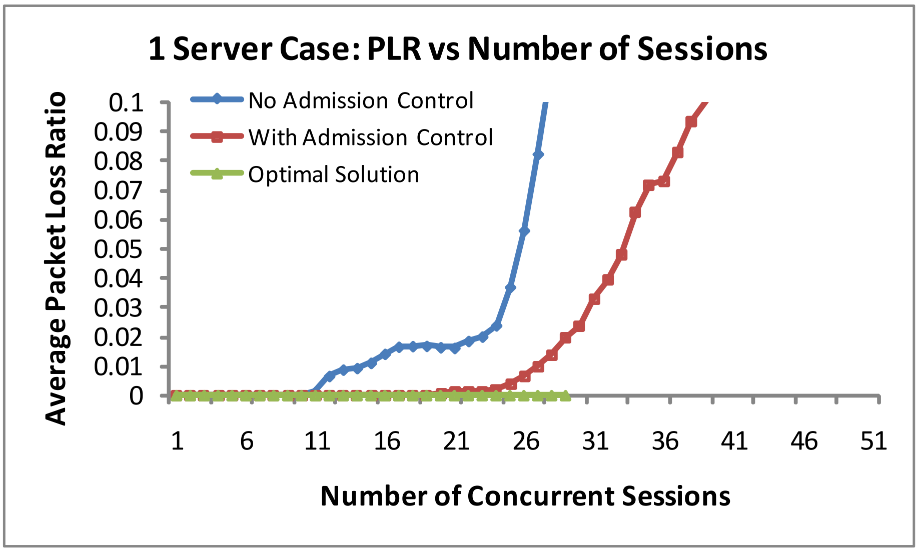


Figure 5.10 One server case: the packet loss ratio vs. the number of concurrent video sessions with/without admission control

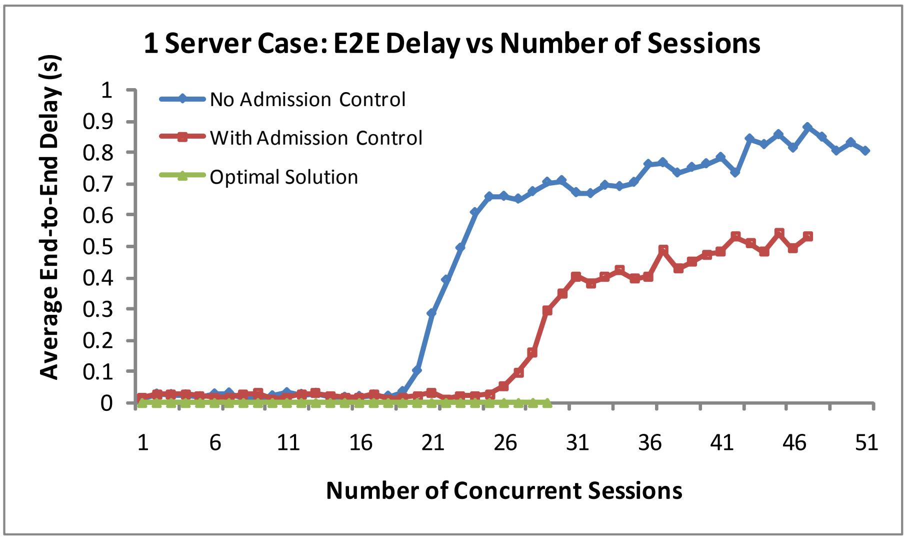


Figure 5.11 One server case: the end-to-end delay vs. the number of concurrent video sessions with/without admission control

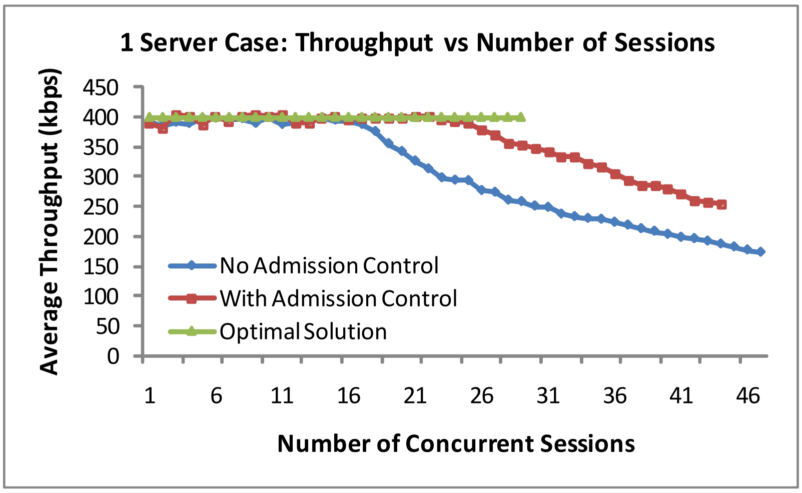
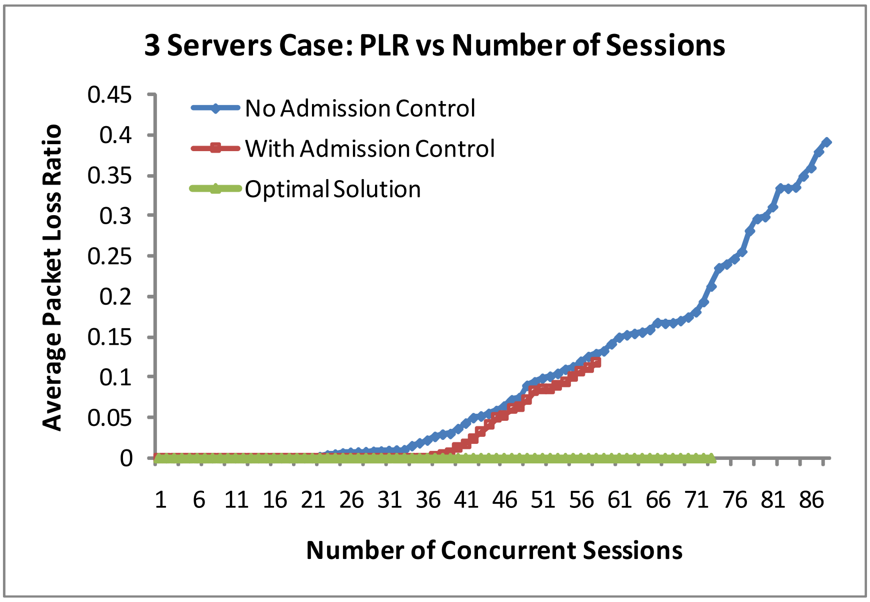


Figure 5.12 One server case: the throughput vs. the number of concurrent video sessions with/without admission control

Figure 5.13 to Figure 5.15 demonstrate comparison of the optimal solution with the distributed ETT solution with/without admission control in the case of 3 servers. In this case, the optimal solution can achieve 73 video session established without any performance degradation. The distributed ETT solution without admission can admit up to 90 sessions but only support about 30 sessions. With admission control, the system can admit about 58 sessions and support about 38 sessions which is also outperforms the scenario without admission control.



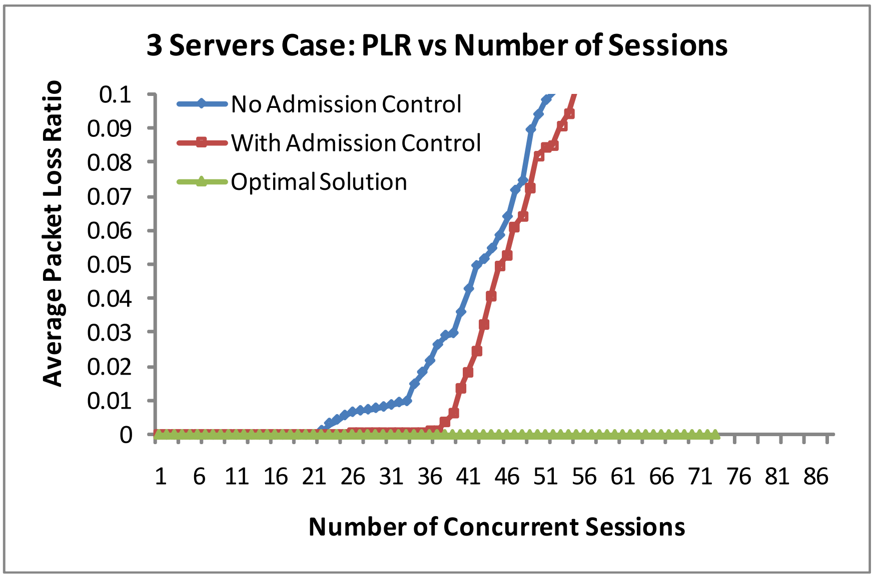


Figure 5.13 Three server case: the packet loss ratio vs. the number of concurrent video sessions with/without admission control

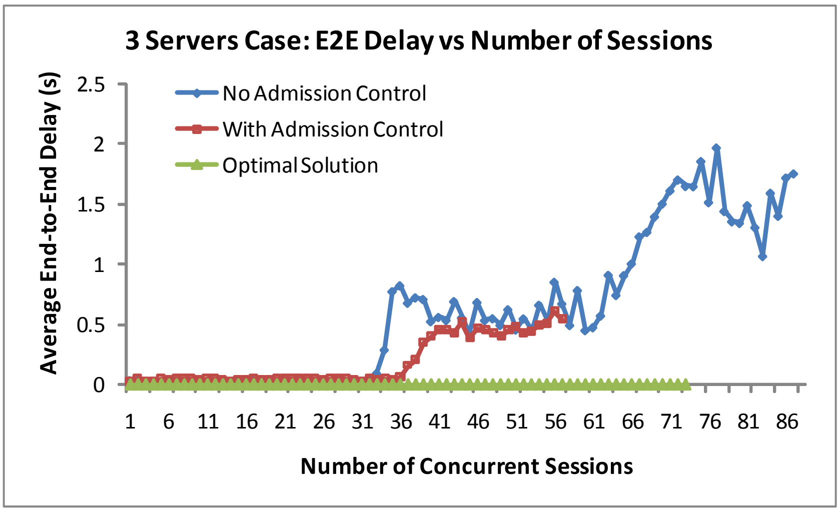


Figure 5.14 Three server case: the end-to-end delay vs. the number of concurrent video sessions with/without admission control

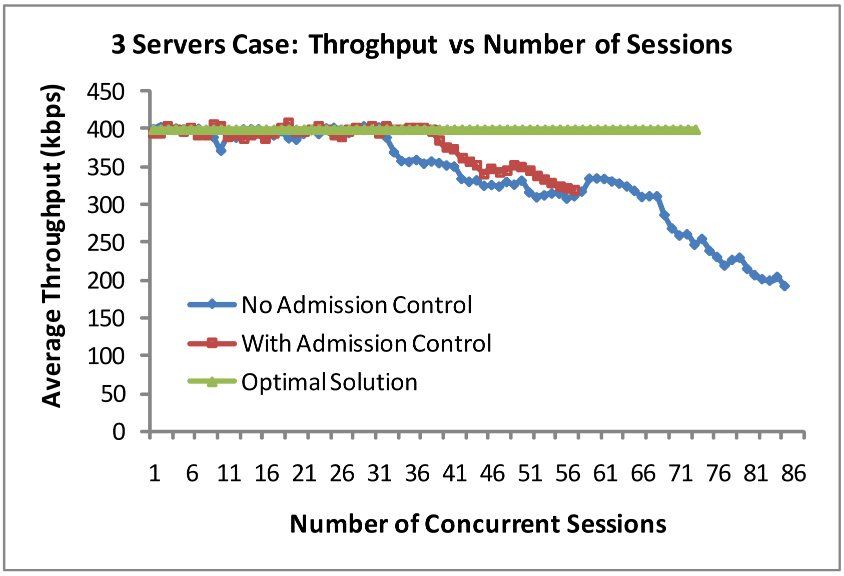


Figure 5.15 Three server case: the throughput vs. the number of concurrent video sessions with/without admission control

Figure 5.16 to Figure 5.18 demonstrate the results of the case of 5 servers. Similar to the above two cases, in this case, the optimal solution can achieve 105 video session established without any performance degradation. The distributed ETT solution without admission control can admit up to 102 sessions but only support about 50 sessions. With admission control, the system can admit about 80 sessions and support about 65 sessions.

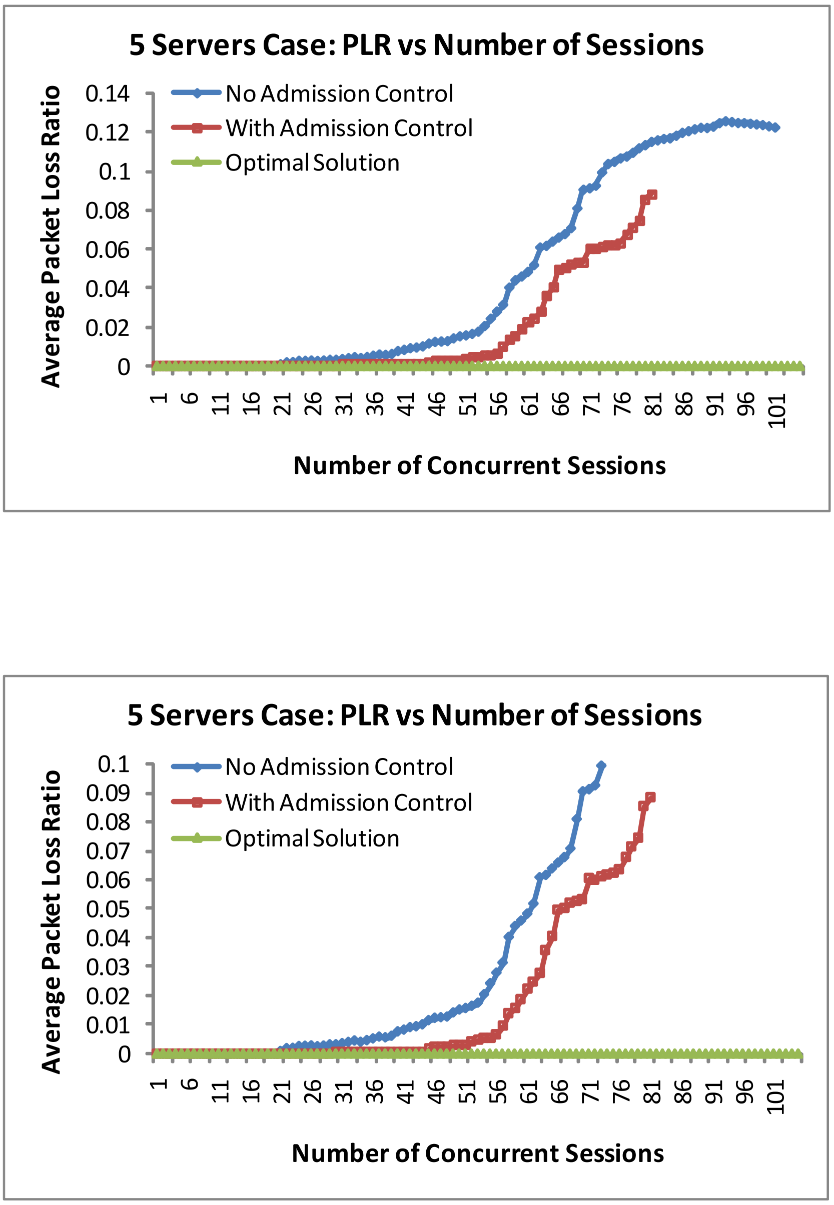


Figure 5.16 Five server case: the packet loss ratio vs. the number of concurrent video sessions with/without admission control

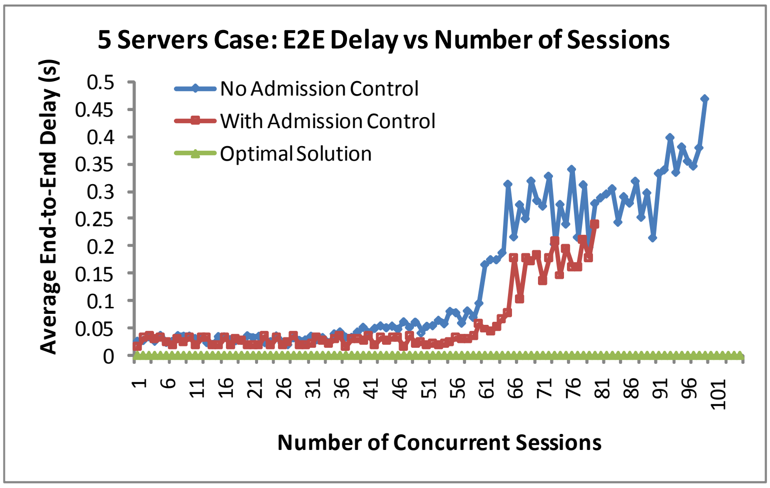


Figure 5.17 Five server case: the end-to-end delay vs. the number of concurrent video sessions with/without admission control

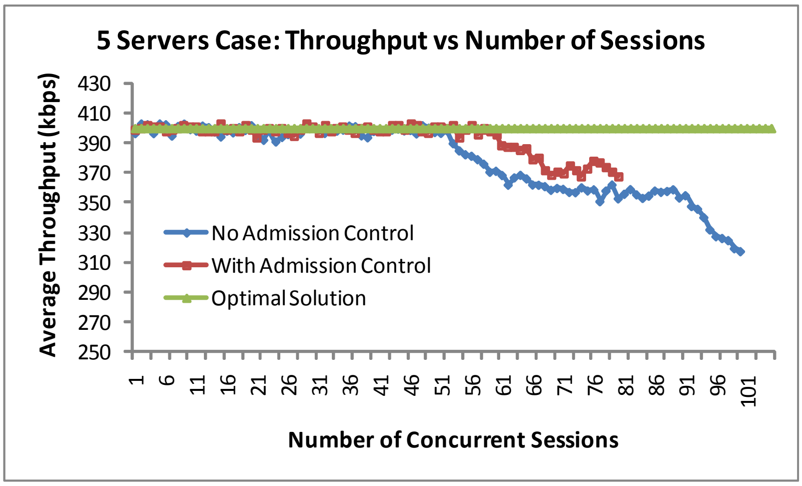


Figure 5.18 Five server case: the throughput vs. the number of concurrent video sessions with/without admission control

The overall numbers of admitted session and the number of supportable session are shown in table 5.2.

Table 5.2 Comparison of results with admission control, without admission control and the optimal solution

Supportable/admitted	Optimal Solution	Without Admission Control	With Admission control
1 server	29/29	15/53	25/47
3 servers	73/73	30/90	38/58
5 servers	105/105	50/102	65/80

These results show that the interference in WMNs is crucial to the performance of the network and will affect the real capacity of the network. Providing video streaming services in a best-effort way is not suitable for WMNs. To further utilize the network capacity, it is necessary to schedule each video streaming request. Optimally select server and path will cause expensive computation complexity. To solve the problem heuristically, we can incorporate an admission control algorithm with per-flow routing at each MAP. The admission control algorithm takes the interference around each MAP into account. To estimate the accurate interference around a MAP will cause a significant number of communications among neighboring nodes. The potential overhead and delay is not negligible. In our proposed heuristic solution, although it under-estimates the complete interference, it considers the intra-flow interference and the interference generated by other flows from the same MAP. In this way, the overhead and delay is reduced dramatically. In addition, to achieve admission control at each MAP, it is necessary to differentiate different streaming request with the same source and destination. In this way, the route for each video flow is stable and is not affected by route change of other flows. Simulation results show that the enhanced routing algorithm with admission control achieves significant better results than the solution without admission control.

5.7 Peer-to-Peer Support in UPAC

As discussed previously, UPAC is composed of CDN and P2P overlay networks. The original data is provided by the main content server which may locate around the gateway or outside the mesh network or from some cache content servers within the WMNs. Actually, these servers function as “permanent seed” in a P2P network. All the streaming data should be fetched from the servers initially. After the stage of fetching-from-server, peers form a P2P overlay network and dynamically maintain such an overlay and exchange data efficiently.

In UPAC, a similar framework to BitTorrent is used and the new core features that are suitable for P2P streaming applications are implemented. In the general architecture, there is a known server, the tracker, for each different video session. By using the tracker, the P2P search problem is leveraged, and the tracker is not involved in content distribution, so there is no the bottleneck problem. One assumption in UPAC is that each participant is willing to share the content with others and there are no free riders.

As we know, chunk and peer selections are the key strategies that affect the P2P performance. We discuss in detail about the chunk and peer selections in our system. Figure 5.19 shows the general chunk and peer selection diagram.

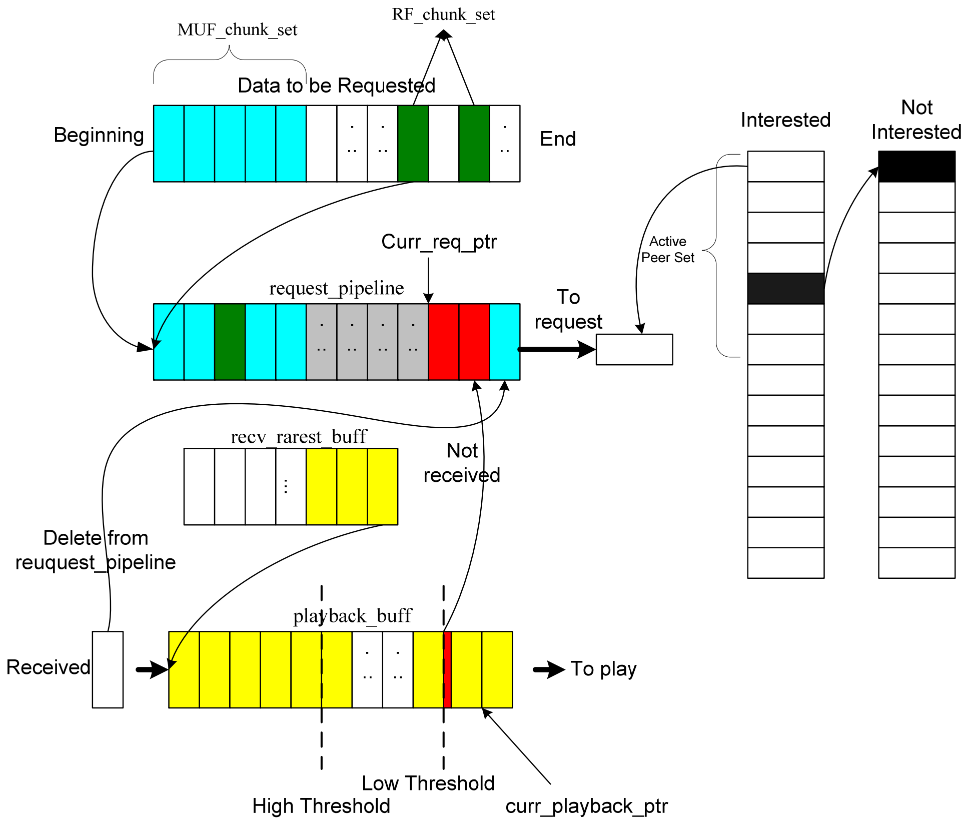


Figure 5.19 Chunk and peer selection diagram

5.7.1 Chunk Selection

In the BitTorrent algorithm, the chunk selection uses the rarest first strategy. The chunk whose availability is the lowest throughout the network is selected first. This algorithm can help to improve the availability of the content (saving the content that is dying). However, BitTorrent is a P2P file sharing system and it does not consider anything about the delay-sensitive data and have no time constraint for data downloading. In this case, the data could be downloaded without caring about the order, and the rarest chunk always has the highest priority. But for P2P streaming application, a streaming-centric framework, the video data has to be played according to the timing order. The rarest first strategy is not suitable for UPAC. An efficient chunk selection algorithm is to make sure the selected chunk satisfy the video playback requirements.

UPAC uses the “most urgent first” (MUF) strategy with help of the rarest first strategy. We assume each peer has a limited size of playback buffer due to the consideration that most P2P streaming applications today do not use hard disk but only use memory.

There are two sets of data, MUF chunk set and RF chunk set (Rarest First). Figure 5.19 shows that when the effective data buffered in the playback buffer exceed a high_threshold, which indicates that the buffer has enough data to play at this time, the chunk select strategy switch to the rarest first. The playback buffer also has a low_threshold. When there are chunks which are not received and the playback of the chunk is close, these chunks become the most urgent chunks and will be retransmitted again with the complementary streaming that will be described later. By using the help of rarest first strategy, the late joined client can help the client who starts the streaming earlier since the rarest first strategy does not require the data to be delivered in order.

To illustrate how the chunk selection scheduling works, the following terms are defined:

MUF_chunk_set: is the most urgent chunk queue in which each unit is ordered by sequence number.

RF_chunk_set: is the rarest chunk queue.

request_pipeline: requests have been sent but not received in which each unit is ordered by sequence number, once received, the unit will be delete from this queue, otherwise, remains in the queue until timeout.

recv_rarest_buff: receiving buffer for the rarest chunk queue.

playback_buff: chunk playback queue in which each unit is ordered by sequence number.

curr_req_ptr: the chunk whose request is sent out, the chunks between this and the head of the pipeline are already requested but not received chunks; the chunks after this are those who have not been requested yet.

curr_playback_ptr: the chunk is being playback right at the time.

The pseudo-code for chunk selection scheduling is in table 5.3.

Table 5.3 Chunk selection scheduling

Algorithm 1 Chunk selection scheduling

```
initialize MUF_chunk_set, RF_chunk_set, request_pipeline,
            recv_rarest_buff, playback_buff
thread1:  while tmp←head of request_pipeline < curr_playback_ptr+low_threshold
          if tmp < curr_playback_ptr
                request_pipeline ->dequeue
          else
                resend request for tmp
          endif
thread2:  while request_pipeline is not full
          if curr_recv_ptr - curr_playback_ptr > high_threshold
          && recv_rarest_buff is not full
                select chunk from RF_chunk_set to request_pipeline
                 else
                select chunk from MUF_chunk_set to request_pipeline
          endif
```

5.7.2 Peer Selection

Once a peer joins a video streaming session, it first gets a peer list from the tracker and tries to find good peers according to the peer selection algorithm. In each update interval, the peer will update the peer list from the tracker and add the new peers to its own peer set. There are two kinds of peer selection schemes in UPAC, one is to select the cache server and the other is to select the client peer. Each peer maintains an "interested" peer set and others are in an "uninterested" set. Peer only sends request to the other peers in the interested peer set. Peer A is interested in peer B when peer B has the chunk which peer A does not have and will request eventually. In the BitTorrent algorithm, there is a choke algorithm which is used in UPAC because it is assumed that each peer will be cooperative in UPAC and no free riders exist. In the interested peer set, there is an active peer set, and a peer only sends streaming request to the peer in the active peer set. Selected cache servers are in the interested peer set. Cache servers in good condition are always in the active peer set. Peer selection metric could be end-to-end delay or routing metrics, e.g., hop count and ETT.

At the beginning, if the peer cannot identify the active peer set in time, the peer sends the first several chunk requests to the best cache servers. For each peer, it will update the active peer set periodically. The pseudo-code is shown in table 5.4

Table 5.4 Peer set update

Algorithm 2 Peer set update

```
get peer set from tracker
probe new peer
if remote peer is probed && remote peer does not have chunk needed
        put remote peer into not interest peer set
else
        put remote peer into interested peer set
endif
if active peer set has been decide
        foreach peer in active peer set
if selection metric > metric_threshold || effective chunk lost rate > lost_threshold
                        delete from active peer set
                        put into not interested peer set
                        put the next peer in peer set into active peer set
                else
                        sort active peer set
                endif
else
        sort peer set
        set active peer set according to selection metric
endif
```

In summary, unlike the P2P file sharing approaches in the Internet, e.g. BitTorrent, the peer selection and management for video streaming in WMNs has more stringent requirements on delay and packet priority. The rarest first strategy is not suitable for video streaming application any more, especially in WMNs. A new hybrid strategy of the most urgent first and the rarest first is proposed. In this strategy, the data is divided into different priority sets. The data whose playback timestamp approaches the current playback time has the highest priority to be requested. The rarest first strategy is activated when most data is not urgent so that the later-joined node can help to contribute in content distribution. In addition, in UPAC, there are two kinds of peers. The mesh content servers and the client peers. Peers are managed in different peer sets. The mesh content servers work as a seed and are always in peers' "interest peer" list. By dynamically update the peer list, the P2P mechanism can help to further balance the load of the network and improve the network capacity utilization. The P2P support in UPAC is demonstrated to be suitable for video streaming over WMNs.

5.7.3 Simulation Results for P2P Support

In this section, the simulation results show the improvement with P2P support in UPAC. The results demonstrate that with P2P architecture in UPAC, the overall performance of video streaming improved significantly in terms of packet loss ratio, end-to-end chunk delay and throughput.

A 7 x 7 grid topology with 49 mesh routers is used. Each mesh router has 10 mesh clients associated with it. In total, there are 490 wireless nodes through the whole simulation. Each mesh router works as a gateway for each of its client and has two 802.11 channel on it. 802.11a/g is used to simulate the wireless radio. The channels for mesh backhaul links and for the access radio of each router are non-overlapping channels and there is no interference between them. The distance between neighboring nodes in the horizontal and vertical direction is 80 meters. For each node, the transmission range is set to be 130 meters so that each node can talk directly to all the direct neighbors. Carrier sensing range is set to be 260 meters. The transmission rate between nodes is determined by the distance. The transmission rate between the vertical and horizontal neighbors is 54Mbps, and that between the diagonal neighbors is 36Mbps. There are five servers and the main server is at the center of the grid and other servers are located close to the corners. According to [65], turning off RTS/CTS handshake results in a statistically significant improvement in throughput, delay and delay jitter especially for connection with few hops.

We use AODV-UU with multiple channels support in NS2. In the popular P2P streaming applications in the Internet, the video rate is about 300 kbps to 600 kbps. The memory used to buffer data is normally about 30 MB. The data rate is set to be 400 kbps and a packet size of 1400 bytes. The video file size is set to be 100 MB and the chunk size is set to be 8 KB. In each peer, the active peer set size is set to be 5. The playback buffer size is set to be 10 MB which can buffer about 200 seconds of 400 kbps video. The recv_rarest_buff is set to be 5 MB. The low_thresh is set to be 2 second and the high_thresh is set to be 20 seconds.

In the simulations, several streaming schemes are compared and we use the following short names for them in the figures.

M_O: Main server only mode. In this mode, there is no server selection. The path selection is based on hop count in traditional AODV. There is no P2P support.

S_D: Servers only mode using end-to-end delay. In this mode, there are 5 servers and the server selection metric is end-to-end delay. The path selection metric is hop count. There is no P2P support.

S _N_E: Servers only mode using ETT. In this mode, there are 5 servers and the server selection metric is ETT. The path selection metric is ETT. There is no P2P support;

P _N_E: P2P mode using ETT. In this mode, there are 5 servers and the server selection metric is ETT. The path selection metric is ETT. There is P2P support.

Figure 5.20 to Figure 5.22 show that adding P2P support into the system significantly improves the system performance. In Figure 5.20, the average chunk loss ratio is much smaller with P2P support than other scenarios since most of the data is transmitted from close peers. In Figure 5.21 and Figure 5.22, the chunk end-to-end delay and throughput are presented. A chunk is an aggregation of data packets and is the basic transmission unit in P2P protocol. The performance of the scheme with P2P support, in terms of both chunk delay and throughput, has the best performance among all the schemes. With P2P support, there are more clients can be supported over the whole WMNs.

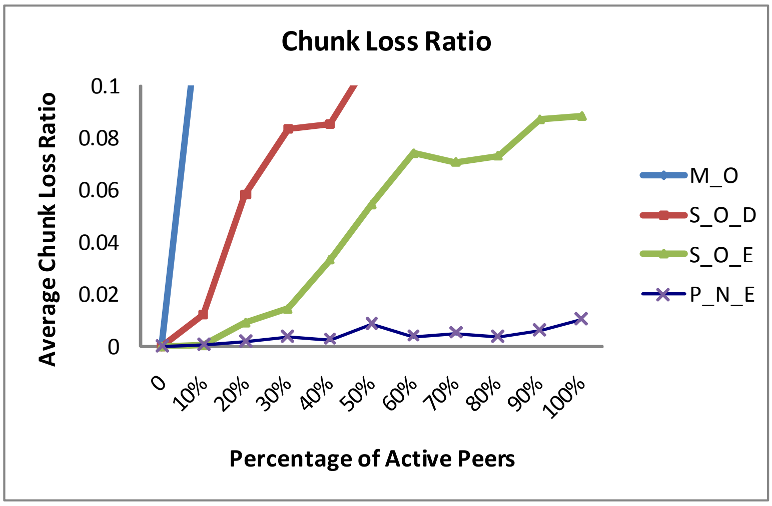


Figure 5.20 Average chunk loss ratio vs the percentage of active peers

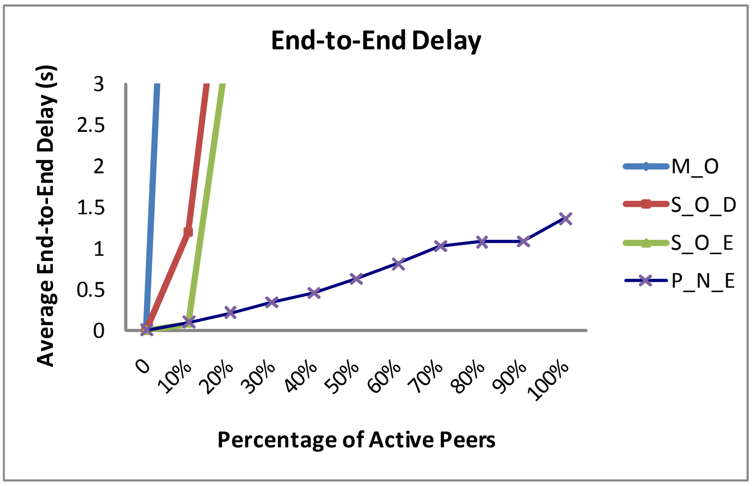


Figure 5.21 Average end-to-end chunk delay vs the percentage of active peers

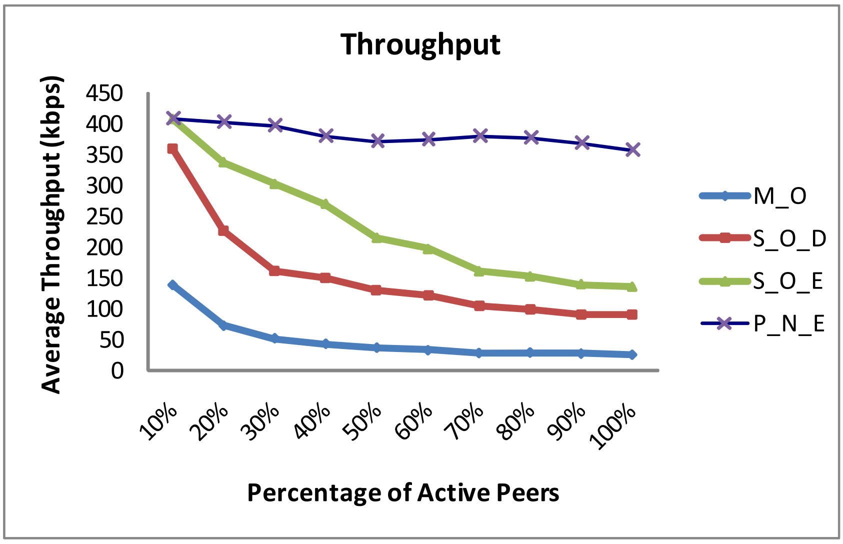


Figure 5.22 Average throughput vs the percentage of active peers

In summary, WMNs is a shared medium network, interference will impact the performance of the network significantly and make the capacity of the network limited. To further improve the capacity utilization of the network, one of the keys is to reduce the impact of the interference and to improve the availability of content for the high data rate applications e.g. video streaming. With the P2P support in UPAC, it extensively increases the content availability, therefore, the overall better paths are selected which significantly reduce the impact of the interference on the backhaul of the WMNs.

5.8 Summary

This chapter proposes a unified peer-to-peer and cache framework for high quality video on demand services over infrastructure multi-hop WMNs. First, the architecture of the framework is proposed. One of the main ideas of UPAC is to use cache servers to cooperate with streaming clients, so that the content can be further distributed over the whole network.

Second, the server and path selection metric from routing layer is proposed and compared with other traditional server selection schemes. In WMNs, the inter-flow and intra-flow interference affect the quality of wireless link and the quality of data transmission. To reduce the impact of the interference among video flows and increase the number of video flows that can be supported by the WMNs, in UPAC, an admission control mechanism is then proposed which takes the interference and other video streaming constraints into account. The admission control mechanism let the MAP to make admission decision for each new streaming request. In the admission control mechanism, to manage each video streaming request individually, it is necessary to use per-flow routing to differentiate different flow requests with the same source and destination. Simulation results show the improvement of system performance using admission control. More concurrent streaming requests are supported with lower packet loss ratio, lower end-to-end delay and higher throughput. To further study the cooperation within the WMNs, the P2P support is proposed. The P2P system in UPAC groups the data into different priority sets and uses the most urgent data first strategy combined with the rarest first strategy. Simulation results show that with P2P support, video data is shared and transmitted within close peers and the stress of the backhaul of WMNs is reduced. In this cooperative way, the performance of UPAC increased greatly in terms of scalability and reliability.

Chapter 6. Summary and Future Works

In this chapter, we summarize the major contributions of this dissertation and propose possible future works for video streaming over WMNs.

6.1 Summary

In this dissertation, we propose frameworks and architectures in both IP-multicast and WMNs to improve video streaming performance using cooperative networking. Best-effort network is effective enough to support data applications due to their tolerance to large delay and delay jitter. However, its lack of special effort to deliver packets in a timely manner makes it extremely challenging to support delay-sensitive video streaming applications. With the cooperation among all participants, the load of the network could be further balanced, the errors could be recovered more efficiently, and the delay could be reduced significantly.

First, we propose a novel path-diversity overlay retransmission architecture for IP-multicast based multimedia applications. The key idea is to find peers with disjoint path from the original sender at low cost and construct a cooperative retransmission overlay. A common network tool "tracert" is used to help to identify the path disjoint retransmission nodes and periodic probing is used to maintain more accurate sets of retransmission nodes. Furthermore, to save the probing overhead, the receiver can use selective probing to probe only a subset of the candidate retransmission nodes who have the highest probability to be a good candidate. By building such a cooperative overlay network, the overall reliability of IP-multicast is improved.

The idea of using help from peers to improve video streaming quality could be used in other network environments as well, such as the WMNs. The research work of the capacity utilization of WMNs has not been well addressed before. The fluctuation of wireless links, channel fading and low throughput of multi-hop path will all affect the capacity of multi-hop wireless network especially for high data rate and delay-sensitive applications. When we design the video streaming services in WMNs, it is necessary to consider an efficient routing algorithm to maintain the good paths and use the hybrid CDN and P2P networks efficiently.

In Chapter 4 and Chapter 5, we present the key issues in improving video streaming in WMNs. UPAC, a unified P2P and cache server framework, which use both the advantages of the CDN and P2P network is proposed.

First, a new routing metric and an enhanced routing algorithm are proposed. Traditional ad hoc networks use the minimum hop count as the routing metric when the data transmitted is not delay-sensitive and high bandwidth demanding. However, in video streaming over WMNs, a longer path with high end-to-end throughput may be a better choice than a shorter path with low throughput. The routing metric we proposed takes the wireless link load, error rate into account and can help to choose better path for video streaming in wireless networks. The enhanced routing algorithm helps to maintain the best route periodically. Simulation results show the proposed routing metric perform much better than traditional hop count based routing metric, and the periodically route maintenance can help maintain a better route from time to time.

Second, the mesh content server discovery and selection problem is further explored by comparing several server selection schemes. We propose and compare several approaches to select the mesh content servers to establish the end-to-end routes between the client and the selected servers. Simulations results show that the routing metric based server selection scheme significantly outperforms other common traditional schemes used in CDN. The proposed scheme increases the number of clients that can be concurrently served in the infrastructure WMNs and improves the received video quality.

The other major contribution is the admission control algorithm with per-flow routing. We formulated a centralized admission control and streaming scheduling problem with necessary constraints in WMNs, e.g., the interference constraint, the flow conservation constraint, the delay constraint etc. The formulated problem is an MIP problem which is known to be NP-hard. Therefore, we propose a heuristics solution to solve the problem in a distributed way. In the proposed solution, each MAP on the potential streaming route makes admission decision based on the accumulated local link utilization. Meanwhile, the per-flow routing algorithm is used for each streaming request to get its own specific route between the client and the server. Simulation results demonstrate the improvement of the overall video streaming performance in terms of less packet loss, shorter end-to-end delay, higher throughput and higher number of concurrent supported video sessions in WMNs.

Cooperating P2P techniques in streaming over WMNs is a challenge work too. There are several difficult problems, e.g. multi-hop throughput, peer interference, and channel allocation etc. On the other hand, the benefits of using P2P are also obvious. Peers can help to reduce the path length and to increase the content availability over the whole WMNs, so that the end-to-end delay is reduced and the throughput of streaming applications is increased. Therefore we proposed a BitTorrent-like P2P framework for video streaming over WMNs. It is necessary to notice that the video streaming through P2P is different from the file sharing through P2P. The strategy we proposed is the most urgent data first strategy combining with the rarest first strategy. In addition, the proposed chunk selection and peer selection mechanism help to achieve the effective P2P cooperation for video streaming. Furthermore, the cooperative P2P networks help to balance the load and increase the availability of content in WMNs. Simulation results show that with the P2P support, the capacity of WMNs for video streaming applications has been significantly improved in terms of throughput, end-to-end delay, packet loss ratio and number of concurrent sessions.

6.2 Future Works

As described in Chapter 5, a BitTorrent-like mechanism is used to enable the P2P ability of the framework. However we should notice the important changes from BitTorrent system. First of all, unlike the BitTorrent system, the streamed data is not fetched randomly or using rarest first strategy. In the streaming application, the data have different priorities, and the most urgent has the highest priority. An optimal data request scheduling algorithm can help to use the network resource more wisely and improve the video playback experience. Second, because of the broadcast nature of the routing algorithm, we can improve the routing algorithm so that the route establishment can be achieved with the content discovery at the same time. To improve the routing algorithm, one of the options is adding messages of content discovery into the route request and the route reply packets. Then, the content discovery can be achieved distributed and the discovery overhead can be reduced. Third, the new generation of WMNs will enable multi-radio and multi-channel communications. For mesh routers, the basic way is to use one radio for the client access and the other radio for the backhaul relay. Furthermore, because 802.11b supports 3 channels and 802.11a supports up to 12 channels, we can use more than one non-overlapping channel in the backhaul. To use multiple non-overlapping channels simultaneously, the challenges are how to allocate the proper channel to maximize the overall throughput for each peer, how to reduce the end-to-end delay and minimize the overall interference. It is worth investigating the channel assignment and scheduling algorithm. Achieving this feature will reduce the interference among links of the neighbors dramatically.

Bibliography

[1] "2008 Analysis of Traffic Demographics in North-American Broadband Networks", by sandvine intelligent broadband networks.

[2] "IP Multicast Technical Overview", Cisco.

[3] K. Almeroth, "Multicast Help Wanted: From Where and How Much?" Keynote speech, *Workshop on Peer-to-Peer Multicasting*, 2007 *Consumer Communications and Networking Conference,* January 2007. http://www.ieee-ccnc.org/pdfs/P2PM-07.pdf

[4] Internet-2 homepage: http://www.internet2.edu/.

[5] J. Beavers, et al, "The learning experience project: enabling collaborative learning with ConferenceXP," Technical Report MSR-TR-2004-42, Microsoft Research, April 2004.

[6] Y. Chu S. Rao, and H. Zhang. (2000). "A case for end system multicast," *Proc. ACM SIGMETRICS Conference*, 2000.

[7] B. Adamson, C. Bormann, M. Handley, and J. Macker, "NACK-oriented reliable multicast (NORM) protocol," IETF Internet-Draft, work in progress, March 2007.

[8] S. Ratnasamy, A. Ermolinskiy, and S. Shenker, "Revisiting IP-multicast," *ACM Sigcomm*, Sept. 2006, Pisa, Italy.

[9] D. Thaler, M. Talwar, A. Aggarwal, L. Vicisano, and T. Pusateri, "Automatic IP-multicast without explicit Tunnels (AMT)," IETF Internet-Draft, work in progress, Nov. 2006. http://tools.ietf.org/html/draft-ietf-mboned-auto-multicast-07

[10] Internet Research Task Force (IRTF), Scalable Adaptive Multicast (SAM) Research Group, www.samrg.org.

[11] Shugong Xu and Tarek Saadawi, "Does the IEEE 802.11 MAC Protocol. Work Well in Multihop. Wireless. Ad Hoc. Networks?" *IEEE Communications Magazine*, Vol. 39, No. 6, pp. 130 - 137, June 2001

[12] C. Perkins, E. Belding-Royer, S. Das, Ad hoc on-demand distance vector (AODV) routing, IETF RFC 3561, July 2003.

[13] David B. Johnson, David A. Maltz, Yih-Chun Hu, "The Dynamic Source Routing Protocol for Mobile Ad Hoc Networks (DSR)," IETF draft-ietf-manet-dsr-10.txt, July 2004.

[14] T. Clausen, P. Jacquet, "Optimized Link State Routing Protocol (OLSR)," RFC 3626, Oct. 2003.

[15] W.-T. Tan and A. Zakhor, “Video multicast using layered FEC and scalable compression,” *IEEE Trans. Circuits Syst. Video Technol.*, vol. 11, no. 3, pp. 373–386, Mar. 2001.

[16] T. Lestayo, M. Fernández, and C. López, “Adaptive approach for FEC reliable multicast,” *Electron. Lett.*, vol. 37, no. 22, pp. 1333–1335, Oct. 2001.

[17] J. Ott, et al, “Extended RTP profile for RTCP-based feedback,” IETF RFC 4585, July 2006. http://www.ietf.org/rfc/rfc4585.txt

[18] J. Rey, et al. “RTP retransmission payload format,” IETF RFC 4588, July 2006. http://www.ietf.org/rfc/rfc4588.txt

[19] J.-C. Wu, K.-J. Peng, M.-T. Lu, C.-K. Lin, Y.-H. Cheng, P. Huang, J. Yao, and H. H. Chen, "HotStreaming: Enabling scalable and quality IPTV services," *Proc. of IPTV Workshop in conjunction with the 15th International World Wide Web Conference*, Edinburgh, Scotland, Uk, May 2006.

[20] H. W. Holbrook, S. K. Singhal, and D. R. Cheriton, “Log-based receiver reliable multicast for distributed interactive simulation,” in *Proc. ACM SIGCOMM*, Aug. 1995.

[21] J. C. Lin and S. Paul. “RMTP: A reliable multicast transport protocol,” *Proc. IEEE INFOCOM*, Apr. 1996, pp. 1414–1424.

[22] M. S. Lacher, J¨org Nonnenmacher, and E.W. Biersack, “Performance comparison of centralized versus distributed error recovery for reliable multicast,” *IEEE/ACM Trans. Networking*, vol. 8, no. 2, pp. 224–238, Apr. 2000.

[23] R. Yavatkar, J. Griffioen, and M. Sudan, “A reliable dissemination protocol for interactive collaborative applications,” in *Proc. ACM Multimedia*, 1995.

[24] C. Papadopoulos, G. Parulkar, and G. Varghese, “An error scheme for large-scale multicast application,” in *Proc. IEEE INFOCOM*, Mar. 1998.

[25] C.-G. Liu, D. Estrin, S. Shenker, and L. Zhang, “Local Error Recovery in SRM: Comparison of Two Approaches*,” IEEE/ACM Transactions on Networking,* vol. 6, no. 6, pp. 686—692, Dec. 1998.

[26] B. N. Levine and J. J. Garcia-Luna-Aceves, “A comparison of known classes of reliable multicast protocols,” in *Proc. Int. Conf. Network Protocols*, Oct. 1996.

[27] A. Mankin, A. Romanow, S. Bradner, and V. Paxson, “IETF criteria for evaluating reliable multicast transport and application protocols,” *RFC2357*, 1998.

[28] K. Obraczka, “Multicast transport mechanisms: A survey and taxonomy,” *IEEE Commun. Mag.*, , Vol. 36, No. 1, 1998, pp. 94-102.

[29] J. Gemmell, T. Montgomery, T. Speakman, and J. Crowcroft, “The PGM reliable multicast protocol,” *IEEE Network*, vol. 17, no. 1, pp. 16–22, Jan./Feb. 2003.

[30] M. Calderon, M. Sedano, A. Azcorra, and C. Alonso, “Active network support for multicast applications,” *IEEE Network*, pp. 46–52, May/Jun. 1998.

[31] S. K. Kasera, S. Bhattacharyya, M. Keaton, K. Kiwior, S. Zabele, J. Kurose, and D. Towsley, "Scalable fair reliable multicast using active services," *IEEE Network*, pp. 48–57, Jan./Feb. 2000.

[32] E. Kim, S. G. Kang, and J. Choe, "A router-assisted session tree configuration mechanism for reliable multicast," *IEEE Commun. Lett.*, vol. 6, no. 9, pp. 464–466, Sep. 2002.

[33] V. N. Padmanabhan, H. J. Wang, P. A. Chou, and K. Sripanidkulchai, "Distributing streaming media content using cooperative networking," *ACM NOSSDAV*, Miami Beach, FL, May 2002.

[34] J. Li, "PeerStreaming: A Practical Receiver-Driven Peer-to-Peer Media Streaming System," Technical Report MSR-TR-2004-101, Microsoft Research, 2004.

[35] Z. Xiang, Q. Zhang, W. Zhu, Z. Zhang, and Y-Q. Zhang, "Peer-to-peer based multimedia distribution service," *IEEE Trans. Multimedia*, vol. 6, no. 2, April 2004.

[36] X. Zhang, J. Liu, B. Li, and T.-S. P. Yum, "DONet/CoolStreaming: A data-driven overlay network for live media streaming," *IEEE INFOCOM*, vol. 3, pp. 2102-2111, Miami, FL, USA, Mar. 2005.

[37] J.-C. Wu, K.-J. Peng, M.-T. Lu, C.-K. Lin, Y.-H. Cheng, P. Huang, J. Yao, and H. H. Chen, "HotStreaming: Enabling scalable and quality IPTV services," *Proc. of IPTV Workshop in conjunction with the 15th International World Wide Web Conference*, Edinburgh, Scotland, Uk, May 2006.

[38] David G. Andersen, Hari Balakrishnan, M. Frans Kaashoek, Robert Morris, "Resilient overlay networks," *ACM Symposium on Operating Systems Principles* (SOSP), 2001.

[39] A. D. Keromytis, V. Misra, and D. Rubenstein, "SOS: Secure Overlay Services," In *Proceedings of ACM SIGCOMM*, pp. 61-72, Aug. 2002.

[40] D. A. Tran, K. A. Hua, and T. T. Do, "A peer-to-peer architecture for media streaming (Zigzag)," *IEEE Journal of Selected Areas of Communications*, pp. 121-133, Jan. 2004.

[41] T. Fei, S. Tao L. Gao, and R. Gu´erin "How to select a good alternate path in large peer-to-peer system?", *IEEE Infocom* 2006.

[42] S. Banerjee, S. Lee, B. Bhattacharjee, and A. Srinivasan, "Resilient multicast using overlays," in *ACM SIGMETRICS*, San Diego, CA, Jun. 2003.

[43] W.-P. Ken Yiu, et al, "Lateral error recovery for media streaming in application-level multicast," *IEEE Trans. Multimedia*, Vol. 8, No. 2, pp. 219-232, April 2006.

[44] Balachander Krishnamurthy, Craig E. Wills, Yin Zhang, "On the use and performance of content distribution networks", Internet Measurement Workshop 2001, 169-182

[45] Bo Li, M. Golin, G. Italiano, X. Deng, and A. K. Sohraby `On The Optimal Placement of Web Proxies in the Internet," IEEE Infocom'99.

[46] Athena Vakali, George Pallis, "Content Delivery Networks: Status and Trends," IEEE Internet Computing, vol. 7, no. 6, pp. 68-74, November/December, 2003

[47] Douglis, F.; Kaashoek, M.F.," Scalable internet services", Internet Computing, IEEE, Volume 5, Issue 4, Jul/Aug 2001 Page(s):36 - 37

[48] http://en.wikipedia.org/wiki/Peer-to-peer

[49] www.napster.com.

[50] N. Leibowitz, M. Ripeanu, and A. Wierzbicki, "Deconstructing the kazaa network" *Proceedings of the 3rd IEEE Workshop on Internet Applications*. Santa Clara, CA.

[51] I. Stoica, R. Morris, D. Karger, M.F. Kaashoek, and H. Balakrishnan, "Chord: a scalable peerto-peer lookup service for Internet applications". *Proceedings of the ACM SIGCOMM*, 2001.

[52] A. Rowstron and P. Druschel, "Pastry: Scalable, decentralized object location and routing for large-scale peer-to-peer systems", *IFIP/ACM Middleware* 2001, Heidelberg, Germany, November 2001.

[53] D. S. Bernstein, et. al, "Adaptive Peer Selection", 2nd *International Workshop on Peer-to-Peer Systems*, 2003.

[54] M. Hefeeda, A. Habib, B. Botev, D. Xu, B. Bhargava, "PROMISE: Peer-to-Peer Media Streaming Using CollectCast", *Proc. of ACM Multimedia*, 2003.

[55] BitTorrent, *http://www.bittorrent.com*

[56] J. Liang, R. Kumar, and K.W. Ross, The Kazaa Overlay: A Measurement Study, Computer Networks (Special Issue on Overlays), 2005

[57] Yang Guo, Saurabh Mathur, Kumar Ramaswamy, Shengchao Yu, and Bankim Patel, "PONDER: Providing Commercial-Quality Video-on-Demand Service Using Peer-to-Peer Network", GLOBECOM, 2007

[58] A. Vlavianos, M. Iliofotou, and M. Faloutsos, "BiToS: Enhancing BitTorrent for Supporting Streaming Applications", in Global Internet Workshop in conjunction with IEEE INFOCOM 2006, April 2006

[59] C. Dana, D. Li, D. Harrison, and C.-N. Chuah. "BASS: BitTorrent assisted streaming system for video-on-demand". IEEE International Workshop on Multimedia Signal Processing (MMSP), October 2005.

[60] http://wifi.google.com/

[61] http://www.metrofi.com/

[62] D. De Couto, D. Aguayo, J. C. Bicket, and R. Morris. "A high-throughput path metric for multi-hop wireless routing", Proceedings of ACM MOBICOM 2003

[63] R. Draves, J. Padhye and B. Zill, "Routing in Multi-radio, Multi-hop Wireless Mesh Network", Proceedings of ACM MOBICOM 2004.

[64] Dimitrios Koutsonikolas, Jagadeesh Dyaberi and etc., “On TCP throughput and window size in a multihop wireless network testbed”, In Proceedings of ACM Mobicom International Workshop on Wireless Network Testbeds, Experimental evaluation and Characterization (WiNTECH 2007), Montreal, QC, Canada, September 10, 2007v

[65] Vikas Kawadia and P. R. Kumar, ``Experimental Investigations into TCP Performance over Wireless Multihop Networks ," in SIGCOMM 2005 - Workshop on Experimental approaches to wireless network design and analysis

[66] L. Haratcherev, J. Taal, K. Langendoen, R. Lagendijk, and H. Sips, “Optimized video streaming over 802.11 by cross-layer signaling,” IEEE Communications Magazine, vol. 44, no. 1, pp. 115-121, Jan. 2006

[67] E. Setton, T. Yoo, X. Zhu, A. Goldsmith, and B. Girod, “Cross-layer design of Ad hoc Networks for real-time video streaming,” IEEE Wireless Communications Mag. , pp59-65, Aug 2005

[68] W. Wei and A. Zakhor, “Multipath unicast and multicast video communication over wireless ad hoc networks,” in Proc. Int. Conf. Broadband Networks, Broadnets, 2002, pp 496-505

[69] S. Mao, S. Lin, S.S. Panwar, Y. Wang and E. Celebi, "Video Transport over Ad Hoc Networks: Multistream Coding with Multipath Transport", IEEE J. on Selected Areas in Communication, Vol. 21, 2003

[70] Peng Xue and Surendar Chandra, "Revisiting multimedia streaming in mobile ad hoc networks,” ACM Network and Operating Systems Support for Digital Audio and Video (NOSSDAV '06), May 22–23, 2006, Newport, Rhode Island

[71] M.A. Ergin et al., ”Available bandwidth estimation and admission contorl for QoS routing in wireless mesh networks”, Comput. Commun. (2008)

[72] X. Zhu and B. Girod, "Media-Aware Multi-User Rate Allocation over Wireless Mesh Networks", Proc. IEEE First Workshop on Operator-Assisted (Wireless Mesh) Communicty Networks, (OpComm-06), Berlin, Germany, September 2006

[73] Zhenghua Fu, Xiaoqiao Meng and Songwu Lu, "A Transport Protocol for Supporting Multimedia Streaming in Mobile Ad Hoc Networks", IEEE Journal on Selected Areas in Communications (JSAC), 21(10), December 2003.

[74] Yiannis Andreopoulos, Nicholas Mastronarde, Mihaela van der Schaar, “Cross-Layer Optimized Video Streaming Over Wireless Multihop Mesh Networks,” IEEE Journal on Selected Areas in Communications 24(11): 2104-2115 (2006)

[75] L. Yin and G. Cao, “Supporting cooperative caching in ad hoc networks,” IEEE Transactions on Mobile Computing, Volume 5 , Issue 1, pp 77-89, Jan 2006

[76] Pavan Nuggehalli, Vikram Srinivasan, Carla-Fabiana Chiasserini, Ramesh R. Rao, “Efficient cache placement in multi-hop wireless networks,” IEEE/ACM Transaction on Networking 14(5): 1045-1055 (2006)

[77] N. Mastronarde, D. Turaga, and M. van der Schaar, "Collaborative resource exchanges for peer-to-peer video streaming over wireless mesh networks," IEEE J. Sel. Areas Commun., vol. 25, no. 1, pp. 108-118, Jan. 2007

[78] Danjue Li, Qian Zhang, Chen-Nee Chuah, and S. J. Ben Yoo, "Error Resilient Multi-source Multi-path Video Streaming over Wireless Mesh Networks", Proceedings of 15th International Packet Video Workshop, Hangzhou, ZheJiang, P. R. China, 2006

[79] X. Zhang, J. Liu, and B. Li, "On Large-Scale Peer-to-Peer Live Video Distribution: CoolStreaming and Its Preliminary Experimental Results", *IEEE Multimedia Signal Processing Workshop*, Shanghai, China, October 30, 2005.

[80] The network simulator ns-2, http://www.isi.edu/nsnam/ns/.

[81] E. W. Zegura, "*GT-ITM: Georgia tech internetwork topology models*" http://www.cc.gatech.edu/fac/Ellen.Ze- gura/gtitm/gtitm.tar.gz, 1996.

[82] A. Rowstron and P. Druschel, "Pastry: Scalable, distributed object location and routing for large-scale peer-to-peer systems". *IFIP/ACM International Conference on Distributed Systems Platform*s (Middleware), Heidelberg, Germany, pp 329–350, 2001.

[83] C. Perkins, E. Belding-Royer, S. Das, "Ad hoc on-demand distance vector (AODV) routing," IETF RFC 3561, July 2003.

[84] David B. Johnson, David A. Maltz, Yih-Chun Hu, "The dynamic source routing protocol for mobile ad hoc networks (DSR)," IETF draft-ietf-manet-dsr-10.txt, July 2004.

[85] T. Clausen, P. Jacquet, "Optimized link state routing protocol (OLSR)," RFC 3626, Oct. 2003.

[86] Uppsala University AODV Implementation. http://core.it.uu.se/AdHoc.

[87] UCSB MeshNet Project. http://moment.cs.ucsb.edu/meshnet.

[88] D.S.J. De Couto, D. Aguayo, J. Bicket, R. Morris, "A high-throughput path metric for multi-hop wireless routing," ACM Mobicom, 2003.

[89] Richard Draves, Jitendra Padhye, and Brian Zill, "Routing in multi-radio, multi-hop wireless mesh networks," ACM Mobicom, 2004.

[90] IEEE 802.11s, ESS Mesh networking, Draft D 0.03, Aug. 2006.

[91] http://personales.unican.es/aguerocr/files/ucMultiIfacesSupport.pdf,"Adding Multiple Interface Support in NS-2."

Vita

Yingnan Zhu was born in Beijing, China in 1978. He received his bachelor degree of B.E. in Electronic engineering from Tsinghua University, China, in 2001. He received his M.S. in Electrical Engineering from Chinese Academy of Sciences, China, in 2004. Since then, he is working toward his Ph.D. degree in department of Computer Science in University of Missouri-Columbia (MU), USA. He was a research intern at the Thomson Corporate Research, Princeton, NJ, from 2006 to 2008. Now he is a senior engineer in Samsung's Digital Media Solution Lab, Irvine, CA. His research interests include multimedia communication, P2P networks, WMNs, IPTV and digital TV.