Multipath Unicast and Multicast Video Communication over Wireless Ad Hoc Networks

by

Wei Wei

B.Eng. (Tsinghua University) 1999
M.Eng. (Tsinghua University) 2001

A dissertation submitted in partial satisfaction
of the requirements for the degree of

Doctor of Philosophy

in

Engineering-Electrical Engineering and Computer Sciences

in the

GRADUATE DIVISION

of the

UNIVERSITY OF CALIFORNIA, BERKELEY

Committee in charge:

Professor Avideh Zakhor, Chair
Professor Jean Walrand
Professor Peter Bickel

Fall 2006

The dissertation of Wei Wei is approved.

Chair	Date
	Date
	Date

University of California, Berkeley

Fall 2006

Multipath Unicast and Multicast Video Communication
over Wireless Ad Hoc Networks

by

Wei Wei

Abstract

Multipath Unicast and Multicast Video Communication
over Wireless Ad Hoc Networks

by

Wei Wei

Doctor of Philosophy in Engineering-Electrical Engineering and Computer Sciences

University of California, Berkeley

Professor Avideh Zakhor, Chair

With the increase in the bandwidth of wireless channels and in the computational power of mobile devices, video applications are expected to become available on wireless ad hoc networks in a near future. However there are many challenges for supporting video communication over wireless ad hoc networks, e.g. the mobility of wireless nodes, random packet loss due to channel error or collision and small bandwidth, which make video communication over wireless ad hoc networks a challenging proposition.

In this dissertation, we introduce new path diversity schemes in order to provide robustness for both unicast and multicast video communication applications over wireless ad hoc networks. We first propose a general architecture for multipath video streaming over wireless ad hoc networks in order to increase the robustness of video applications. This architecture includes a video encoder with error control, a traffic allocator, which decides how to distribute video packets into multiple paths, a multipath unicast/multicast routing protocol, and a rate control scheme.

For the unicast case, we propose multipath streaming with two node-disjoint paths, which have minimum concurrent Packet Drop Probability (PDP) among all path pairs. This approach minimizes the probability of concurrent loss of all the descriptions, thus optimizing the worst case Multiple Description Coding (MDC) video quality over all times. We model the effects of interference between different wireless links, and estimate the concurrent PDP of two node-disjoint paths. We show that the above optimization is an NP-hard problem. Then we propose a heuristic PDP aware multipath routing protocol based on our path selection model, whose performance is shown to be close to that of the "optimal routing", and significantly better than that of the node-disjoint multipath routing and the shortest-widest routing, through extensive NS simulations and actual experiments.

For the multicast case, we propose multiple tree video multicast with MDC to provide robustness for video multicast applications. Specifically, we first propose a simple distributed protocol, Serial Multiple Disjoint Tree Multicast Routing (Serial MDTMR). This scheme results in reasonable tree connectivity while maintaining disjointness of two trees. In order to reduce the routing overhead and construction delay of Serial MDTMR, we further propose Parallel Multiple Nearly-disjoint Trees Multicast Routing (Parallel MNTMR), which constructs two nearly disjoint trees simultaneously in a distributed way. Simulation shows that video quality of multiple tree multicast video communication is significantly higher than that of single tree multicast video communication, with similar routing overhead and forwarding efficiency.

Professor Avideh Zakhor
Dissertation Committee Chair

To my wife, Yewei,

To my son, Andrew,

To my mom and dad.

Contents

List of Figures

List of Tables

Acknowledgements

This thesis was made possible by the help, support and guidance of many people. First I would like to thank my advisor Avideh Zakhor for giving me a chance to work with her during the past five years. Avideh has been a constant source of encouragement and guidance at every step in my graduate career. She is an excellent researcher and maintains the highest standards for herself and others surrounding her.

I thank Prof. Jean Walrand and Prof. Peter Bickel for their advice and support as members of my thesis committee. Prof. Randy Katz served on my qualifying examination committee and I learned a lot through my conversations with him.

Many colleagues have helped me over the years. I thank Minghua Chen for his invaluable help, support and frequent discussions on my research problems. Especially, I would like to thank Cindy Hsin-I Liu, Allan Gu, Minghua Chen, Gap N Thirathon, Steven Jian, and John Lo for spending hours helping me run the experiments on the testbed. Without their kind help, it is not possible to establish the testbed and finish the experiments. I would like to thank Thinh Nguyen, Samson Cheung, Yufeng Shan and Vito Dai for the helpful discussions and guidance during my early years in Berkeley. The years I spent in Berkeley were most enriched by my day-to-day interactions with my colleagues: Pierre Garrigues, Matulya Bansal, Christian Fruh, David Harrison, Siddarth Jain, Sang H. Kang, Ali Abbas Lakhia, Puneet Mehra, John Secord, Christophe De Vleeschouwer. I would also like to thank Daniel Tan at HP Labs for the discussions, help and encouragements.

I would like to acknowledge the financial support for my research from AFOSR contract F49620-00-1-0327.

Most of all, I would like to thank my family for their love, affection and support at all times. Thanks to my father for being a great role model and sparkling my early interest in science and engineering. Thanks to my mother for her love and countless

sacrifice to support me unconditionally. Thanks to my sister and brother-in-law, who are always behind me. Without their help and support, it is not possible to finish my thesis.

I am deeply indebted to my dear wife Yewei Zhang for her love, support and understanding during my graduate years. She is always behind me and gave her unconditional support. She helped me setting up the testbed even during her pregnancy period. Without her love and support, the achievements of my thesis are impossible. Finally, I would like to thank my son Andrew for the happiness he brings to me.

Chapter 1

Introduction

Wireless ad hoc networks are becoming increasingly popular as they provide users access to information at anytime and from anywhere. Conventional wireless networks are usually supported by a wired fixed infrastructure. A mobile device would connect to a base station through a single-hop wireless connection. In contrast, a wireless ad hoc network is a collection of wireless mobile nodes that dynamically form a temporary network without an infrastructure. The nodes in a wireless ad hoc network connect with each other via multi-hop paths in a peer-to-peer fashion. Intermediate nodes between a communication pair act as information forwarder. Thus nodes in the network operate both as end hosts and as routers. The nodes could be potentially mobile, so the topology of the network may change randomly and unexpectedly.

With the increase in the bandwidth of wireless channels, and in the computational power of mobile devices, video applications are expected to become available on wireless ad hoc networks in a near future. To understand the challenges of supporting video applications over wireless ad hoc networks, in Section 1.1, we introduce design issues, and review current routing protocols briefly. In Section 1.2, we analyze main challenges of video streaming over wireless ad hoc networks. We introduce re-

lated works on multipath video streaming in Section 1.3, and summarize the thesis contributions to video streaming over wireless ad hoc networks in Section 1.4.

1.1 Wireless Ad Hoc Networking

A wireless ad hoc network is a collection of wireless mobile nodes that dynamically form a temporary network without an infrastructure. Figure 1.1 demonstrates a typical wireless ad hoc network. There are nine nodes in the network. Two nodes, which can not talk to each other directly, form a path consisting of several forwarding nodes.

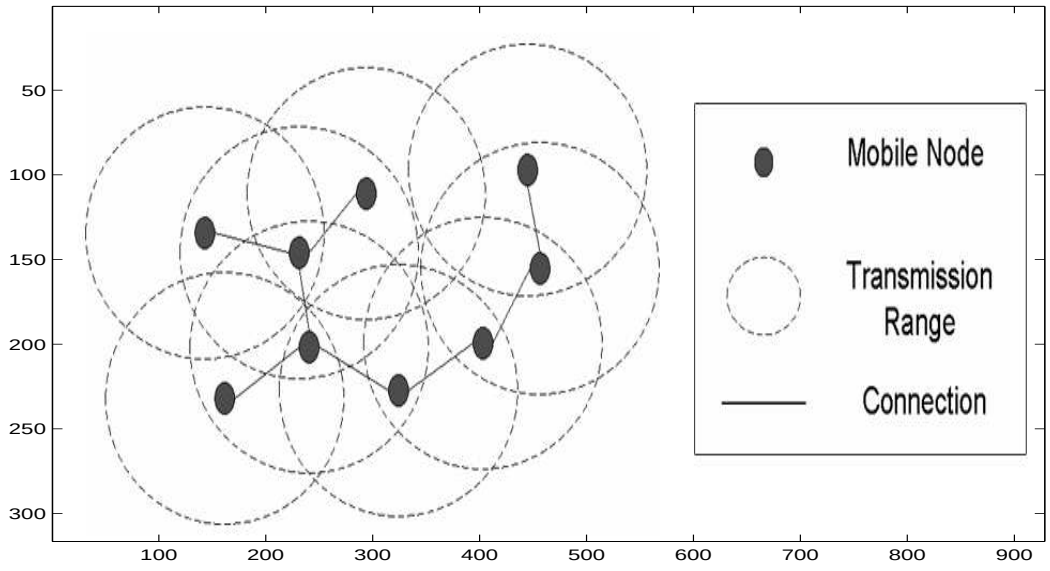


Figure 1.1. Demonstration of a wireless ad hoc network.

In designing a wireless ad hoc network, we need to consider the following issues [1].

- **Dynamic topology:** The topology in a wireless ad hoc network may change

randomly due to nodes' mobility. As nodes move in and out of the range of each other, some links break and new links are created.

- **Multi-hop paths:** Nodes inside an ad hoc network are often not within direct communication range. Thus the support of multi-hop paths is essential to the design of an ad hoc network. Also because of multi-hop paths, the end-to-end packet drop due to channel error is increased and the end-to-end throughput is greatly decreased, compared to the single-hop infrastructure based wireless networks.

- **Unreliable wireless medium:** The wireless communication medium has variable and unpredictable characteristics. Due to varying environmental conditions, such as different levels of electro-magnetic interference (EMI), the signal strength and propagation delay fluctuate with respect to time and environment.

- **Self-organizing:** The ad hoc network must autonomously determine its own configuration parameters including: addressing, routing, clustering, identification, power control, and etc.

- **Energy conservation:** Most ad hoc nodes, e.g. laptops, PDAs and sensors, have limited power supply, and can not generate power themselves. Thus energy efficient protocols are critical for the longevity of the operation of the network.

- **Scalability:** Because of the extensive mobility and the lack of fixed infrastructure, pure ad hoc networks do not tolerate mobile IP or a fixed hierarchy structure. Thus, mobility, jointly with large scale is one of the most critical challenges in the design of ad hoc networks.

- **Security:** Because of the ability of the intruders to eavesdrop and jam/spoof the channel, the security problem of ad hoc networks is severe.

- **Connection to the Internet:** Extending the infrastructure based wireless networks opportunistically with ad hoc networks is also an important issue. For instance, the reach of a wireless LAN can be extended to previously unreachable area through ad hoc networking techniques.

As a result of the above issues, a wireless ad hoc network is prone to numerous types of faults including:

- **Transmission errors:** Packets can be corrupted and dropped due to the unreliability of the wireless medium.
- **Node failures:** Nodes may fail at any time in the network. Nodes can also drop out of the network either voluntarily or when their energy is depleted.
- **Link failures:** Node failures and varying environmental conditions, e.g. increasing level of EMI, can cause links between two nodes broken. Both node and link failures can break a route, causing packet drop.
- **Congestion:** Depending on the topology of the network and the traffic flows, certain areas of the network can be congested, causing longer delays or packet loss.

Routing is one of the most significant challenges in designing wireless ad hoc networks, and is essential for the basic network operations. There is a great deal of research on designing routing protocols in wireless ad hoc networks. Here, we briefly review several popular unicast and multicast routing protocols.

Based on the way of building and maintaining routes, unicast routing protocols can be classified as proactive routing, on-demand routing, and hybrid routing. Proactive routing protocols maintain routes between all pairs of nodes regardless of whether

these routes are actually used. Therefore, when the need arises, the traffic source already has a route available, thur reducing the delay for route discovery. Destination-Sequenced Distance-Vector (DSDV) [2] is one of the earliest protocols designed for wireless ad hoc networks. The main idea in DSDV is the use of destination sequence numbers to avoid loops in the route. Each node maintains a monotonically increasing sequence number for itself, and the highest known sequence number for each destination. These sequence numbers are used to determine the relative freshness of distance information generated by two nodes for the same destination. Routing loops are prevented by requiring destination sequence numbers along any valid route to monotonically increase toward the destination. Optimized Link State Routing (OLSR) [3] is an optimized version of traditional link state protocols such as Open Shortest Path First (OSPF) [4]. It uses the concept of MultiPoint Relay (MPR) to efficiently disseminate link state updates across the network. The main idea of MPR is that each node selects a small subset of its neighbors as MPRs, which are sufficient to cover its two hop neighborhood. When a node broadcasts the link state update packet, only the MPRs of the node rebroadcast the packet, and their MPRs rebroadcast and so on. Moreover, link state updates contain only the links between MPR nodes and their selectors in order to reduce the update size. Thus OLSR greatly reduces the control overhead compared to its wired peer OSPF.

Another category of routing protocols is on-demand (reactive) routing. On-demand routing protocols only build and maintain routes required by sources. The main advantage of on-demand routing is lower routing cost, since it does not need to maintain all the routes. However, data packets experience extra delay at the source during the initialization stage. Dynamic Source Routing (DSR) [5][6] is characterized by the use of source routing. The sender knows the complete end-to-end route to the destination. Data packets carry the source route in the packet header. When a node wants to send a data packet to a destination, if the sender does not know the route, it

starts a route discovery process by broadcasting a Route Query (RREQ) packet to the network. Each node rebroadcasts a non-duplicate received RREQ packet, unless it is the destination or it has a route in its route cache. Such a node replies to the request through unicasting a Route Reply (RREP) packet back to the sender. If any link in the route is broken, a Route Error packet is generated, and is sent back to the sender. The source will initiate a new route discovery process after receiving the Route Error packet. Ad hoc On-demand Distance Vector (AODV) [7] uses route discovery process similar to DSR to gather routing information and build routes. Unlike DSR, AODV uses traditional routing tables, one entry per destination to store routes. Without using source routing, AODV uses routing table entries to route data packets to the destination. AODV also uses destination sequence numbers as in DSDV [2] in order to prevent routing loops and determine the freshness of routing information.

Zone Routing Protocol (ZRP) [8] is a hybrid approach between proactive and reactive protocols. ZRP defines a zone for each node X, which includes all the nodes that are within a certain distance, zone radius, around the node X. Nodes that are exactly zone radius distance away from node X are called border nodes of X's zone. A proactive link state protocol is used to keep every node aware of the complete topology within its zone. When a node X needs to obtain a route to another node Y that is not in the zone, it initiates a route discovery process, which works similar to flooding except that it involves only X's border nodes and their border nodes and so on. Route query messages accumulate the traversed route on its way outward from X. When the query reaches a border node which is in the destination Y's zone, the border node sends back a reply packet to the sender using the obtained route from the query.

Location-based routing [9][10] is a different routing approach that utilizes geographic location of nodes. Location-based routing protocols assume that each node knows its own location by using the global positioning system (GPS) or some indi-

rect, localization techniques. Every node learns positions of its immediate neighbors by exchanging periodical Hello message. Location-Aided Routing (LAR) [9] uses an estimate of destination's location to restrict the flood to a small region in order to reduce control overhead. The source floods a route request packet by including its estimate of destination's location and its estimated distance to the destination in the request. Neighboring nodes calculate their distance to the destination after receiving the route request packet. Those nodes with a smaller estimated distance to the destination node forward the request packet further until the request packet reaches the destination node.

The routing protocols discussed above use minimum hop counts as a metric to build routes. However shortest path is generally not optimal in terms of throughput because the links along a shortest path might not necessarily have high quality. D. De Couto, et al. [11] have proposed a new routing metric called Expected Transmission Count (ETX). The ETX metric measures the expected number of transmissions, including retransmissions, needed to send a unicast packet across a link. The path metric is the sum of the ETX values for each link along the path. Based on ETX, Draves, et al. [12][13] propose Weighted Cumulative Expected Transmission Time (WCETT) as a metric to select a path in a multiple channel multihop wireless network. The WCETT considers link bandwidth, channel loss rate, and channel diversity.

Most proposed wireless ad hoc routing protocols are unipath protocols, which only use a single path to send packets to the destination. In contrast, multi-path routing protocols build multiple routes at the same time between the source and the destination nodes. There are several advantages to multi-path routing protocols.

- Fault tolerance: When a link is broken, alternative routes can be used to route the packets.

- Load balancing: We can distribute traffic over multiple routes, balancing traffic load among the network.

- Higher aggregate bandwidth: It is possible to use multiple paths simultaneously to transmit packets, resulting in a larger aggregate bandwidth than a single path.

Split Multipath Routing (SMR) is one of the best known multipath extensions to DSR [14][15]. SMR is similar to DSR, and is used to construct maximally disjoint paths. It uses a modified RREQ packets flooding scheme in the process of route query. Duplicate RREQs are not necessarily discarded. Instead, intermediate nodes forward RREQs that are received through a different incoming link, and whose hop counts are not larger than the previously received RREQs. By doing this, SMR increases the probability of two disjoint paths to the destination. Unlike DSR, intermediate nodes do not keep a route cache, and do not reply to RREQs. This is to allow the destination to receive all the routes so that it can select the maximally disjoint paths. Maximally disjoint paths have as few links or nodes in common as possible. The destination node returns the shortest path and another path that is maximally disjoint with the shortest path to the source node.

Two Multipath extensions of DSR (MDSR) are proposed in [16]. The main objective of MDSR is to increase the interval between route queries using multiple paths, thus reducing routing overhead. However it continues to use route query method of DSR, which only obtains highly correlated paths. Several path selection criteria for building multiple paths are proposed in [17]. The criteria include node-disjointness, small length differences between the primary path and the alternative paths, and small correlation factors between any two of the multiple paths.

AOMDV [18] is an extension of AODV for building multiple loop-free and link-disjoint paths. To keep track of multiple routes, the routing entries for each desti-

nation contain a list of the next-hops along with the corresponding hop counts. For each destination, a node maintains the advertised hop count, which is defined as the maximum hop count for all the paths. To ensure loop free paths, a node only accepts an alternate path to the destination if it has a lower hop count than the advertised hop count. In [20], the authors discuss more issues regarding to multipath routing over wireless ad hoc networks.

Multicast is an essential technology for many applications, such as group video conferencing and video distribution, and results in bandwidth savings as compared to multiple unicast sessions. Due to the inherent broadcast nature of wireless networks, multicast over wireless ad hoc networks can be potentially more efficient than over wired networks [19]. However since nodes could move around freely and rapidly, the network topology changes frequently, making wireless ad hoc multicasting more challenging than its wireline Internet counterpart. Thus the primary goal of ad hoc multicast protocols should be able to construct and maintain a robust and efficient multicast tree even during high network dynamics. In the following, we will introduce a few proposed multicast routing protocols [20-29].

The On-Demand Multicast Routing Protocol(ODMRP) [22][23] builds multicast mesh through periodically flooding the network with control packets to create and maintain the forwarding state of each node, when the source has packets to send. It takes advantage of the broadcast nature of the wireless network by forwarding group flooding, which provides a certain amount of diversity. By flooding a Join Query (JQ) message, a source node starts building a forwarding group for the multicast group. When a node receives a non-duplicate JQ message, it stores the upstream node ID and rebroadcasts the packet. when the JQ message reaches a multicast receiver, the receiver sends back a Join Reply (JR) message back to the sender along the reverse path traversed by the JQ message. Multicast sources refresh the membership information and update the routes by sending JQ messages periodically. A mesh

structure is equivalent to a tree structure with *tree flood* enabled[24]. In the remainder of this thesis, we refer to ODMRP as a single tree multicast protocol.

The Adaptive Demand-Driven Multicast Routing (ADMR) [24] attempts to reduce non-on-demand components within the protocol as much as possible. ADMR uses no periodic network-wide floods of control packets, periodic neighbor sensing, nor periodic routing table exchanges. In ADMR, forwarding state is specific to each sender rather than being shared by the entire multicast group. This approach reduces unnecessary forwarding data redundancy. There is also a local subtree repair scheme to detect broken link by downstream node in ADMR.

The Adaptive Core Multicast Routing Protocol (ACMRP) [25] is an on-demand core-based multicast routing protocol that is based on a multicast mesh. A multicast mesh is created and maintained by the periodic flooding of the adaptive core. A core emerges on demand and changes adaptively according to the current network toplogy. This scheme outperforms ODMRP in multi-source scenarios.

Multicast Core-Extraction Distributed Ad Hoc Routing (MCEDAR) [26] classifies nodes in the network into two groups: a core node or an immediate neighbor of a core. MCEDAR uses a mesh structure called the *mgraph* as the multicast routing structure. Only the core nodes can become members of an *mgraph*. When a node wants to join a group, it requires its dominating core to join the *mgraph*. The dominating core then performs the join operation by broadcasting a JQ message. The JQ message is relayed by the non-member nodes. When a group member receives a JQ message, it sends back a JR message. The forwarding protocol of MCEDAR follows the core broadcast procedure. When a data packet arrives at a *mgraph* member, the member only forwards the packet to those nearby core nodes on the *mgraph*. This procedure is more efficient than the hop-by-hop flooding.

Differential Destination Multicast (DDM) [27] is intended for small group multi-

cast. DDM includes the forwarding states in the packet header rather than intermediate nodes. From the information carried in the packet headers, any intermediate node knows how to forward or duplicate the packet. Although packing routing information together with data traffic enlarges data packet size, it reduces the total number of control packets generated by the protocol. Besides, when the group is idle, there is no control overhead. DDM is more suitable when the multicast source only sends out sporadic packets, and is not suitable for video multicast source, which transmits video packets all the time.

Ad-hoc Multicast Routing Protocol(AMRoute) [28] creates a per group multicast distribution tree using unicast tunnels connecting group members. Each group in the network has at least one logical core that is responsible for discovering new group members and creating/maintaining the multicast tree. There are two main phases in the protocol operations: mesh creation and tree creation. During the mesh creation stage, all the members for the same multicast group discover the existence of each other and select one core for the group. The core is responsible for initiating the tree creation process by sending out periodic TREE-CREATE messages. Group members receiving non-duplicate TREE-CREATE messages forward them on all mesh links except the incoming one, and mark the incoming and outgoing links as tree links. AMRoute operates independent of the underlying unicast protocols.

The Independent-Tree Ad Hoc Multicast Routing (ITAMAR) [29] creates multiple multicast trees based on different metrics in a centralized way. ITAMAR constructs multiple edge disjoint or nearly disjoint trees. The main objective of this protocol is to improve the average time between multicast trees failures. The algorithms are basically based on Dijkstra SPF algorithm, which is a centralized approach, and requires knowledge of network topology. One possible problem of ITAMAR is that routing overhead might be very large in order to get enough information about the

network to build multiple trees, and the authors only show how ITAMAR works based on perfect network information.

1.2 Challenges of Video Streaming over Wireless Ad Hoc Networks

Video communication applications have been important for wired networks, such as the Internet, for quite some time. However, with the development of broadband wireless networks, attention has only recently turned to transmitting video over wireless networks, especially over wireless ad hoc networks. With the increase in the bandwidth of wireless channels, and in the computational power of mobile devices, video applications are expected to become available on wireless ad hoc networks in a near future. There are many possible video communication applications over wireless ad hoc networks, such as spontaneous video conferencing in a place without wireless infrastructure, transmitting video on the battlefield, and search and rescue operations after a disaster.

Video communication is fundamentally different from data communication, since video applications are delay and loss sensitive. Even though some packet loss is tolerable, the quality of reconstructed video or audio will be impaired and errors will propagate to consecutive frames because of the dependency introduced among frames belonging to one group of pictures at the encoder [32].Unlike data packets, late arriving video packets are useless to the video decoder. Thus, the retransmission techniques, which guarantee the successful transmission of data packets, are not applicable to video communication applications.

There are additional challenges for supporting video communication over wireless ad hoc networks. First, nodes in a wireless ad hoc network are allowed to move in an

uncontrolled manner. Such node mobility results in a highly dynamic network with rapid topological changes. Thus the established connection routes between senders and receivers are likely to be broken during video transmission, causing interruptions, freezes, or jerkiness in the received video signal. Second, the underlying wireless channel provides much lower and more variable bandwidth than wired networks. An end-to-end connection route in wireless ad hoc networks generally consists of multiple wireless links, which makes the available bandwidth per node even lower. Third, a wireless link usually has high transmission error rate because of shadowing, fading, path loss, and interference from other transmitting users. An end-to-end route in wireless ad hoc networks has higher error rate compared to single hop wireless connections in a wireless network with an infrastructure, since it is the concatenation of multiple wireless links. Fourth, mobile devices running on batteries have limited energy supply, and video encoding and transmission typically consume a great deal of power. These constraints and challenges, in combination with the delay and loss sensitive nature of video applications, make video communication over wireless ad hoc networks a challenging problem.

1.3 Related Work on Video Streaming with Path Diversity

Path diversity has been shown to provide robustness in video communication applications [33][34][35]. The existence of multiple paths between each communication pair is likely due to the mesh topology of ad hoc networks. Given multiple paths, video streams can be divided into different substreams, which can be transmitted over different paths simultaneously. If multiple paths are disjoint, each substream experiences relatively independent packet loss. Thus received video quality can be

improved with appropriate video coding and packet dispersion techniques. Connectivity of multiple paths is much less likely to be broken than that of a single path in wireless ad hoc networks. Path diversity can also be exploited to reduce the congestion level of wireless ad hoc networks [36]. In this section, we will review existing schemes which transmit video with path diversity.

1.3.1 Multipath Video Streaming over Wired Networks

Nguyen and Zakhor [37] propose a framework for simultaneous streaming of video from multiple mirror sites to a single receiver, in order to increase available bandwidth, and to reduce packet loss and delay. The receiver coordinates simultaneous transmissions from multiple senders through the control packets sent to all the senders. The proposed protocol employs a rate allocation scheme, which determines the sending rate on each route, and a packet partition algorithm, which guarantees that each packet is sent only once. In [38], Nguyen and Zakhor extend the rate allocation algorithm to incorporate Forward Error Correction (FEC) to combat bursty packet loss. They show that the combination of FEC and multiple paths is more effective than using FEC with a single path over wireline Internet. They also propose the optimal strategy for using FEC under various network conditions. In [39], Nguyen and Zakhor propose a path diversity system that allows a single sender to send packets simultaneously on both a default and a redundant path to the receiver using the overlay network technique.

In [33], Apostolopoulos proposes a multiple state video coding technique to encode the video into multiple independently decodable streams, and to explicitly transmit different streams through different network paths in wireline networks. The author also proposes two ways of constructing multiple paths in the wireline Internet. The first one is via IP source routing, and the other one is via relay nodes. In [40], the

authors design a Multiple Description Coding (MDC) video communication system that is effective in both balanced and unbalanced cases. Unbalanced MDC video streams are created by adjusting the frame rate of each description. In [41], the authors develop models to predict and compare the distortion of MDC video and path diversity against Single Description Coding (SDC) over a single path. In Content Delivery Networks (CDN), the authors in [42] propose to distribute different streams of MDC video from different edge servers, which is equivalent to transmitting MDC video through different network paths.

In [61], Chakareski, et al. compare performance of transmission of MDC video and layered coding over both a single path and multiple paths in the wireline Internet. They conclude that when rate-distortion optimized packet scheduling is applied, layered coding outperforms MDC video. On the other hand, when there is no such packet scheduling, MDC video outperforms layered coding.

Begen, et al. [43] study how to select multiple paths that maximize the *average* quality of video at clients on Internet overlay networks. The optimization problem of [43] is NP-hard, which is intractable in large topologies. Thus the authors in [44] provide a fast heuristic, which reduces the computation by exploiting the infrastructure features of the Internet. Mao et al. [45] further propose a meta-heuristic approach based on Genetic Algorithms to solve the above path selection problem. However these approaches are too complex to be performed in real-time. Also the model considers neither the interference of flows on neighboring links, nor the influence of the incoming video flow on the characteristics of links; this is not appropriate in current wireless ad hoc networks, because a new video flow generally consumes a large percentage of wireless resource, thus changing characteristics of wireless links significantly.

In [46], the authors select two paths with minimal correlation for MDC streaming

over Internet overlay networks. In [47], Chen, et al. present a multipath heuristic, which achieves the end-to-end bandwidth requirement of a video application. The authors also propose a data scheduling algorithm, implemented at the server, which achieves theoretical minimum end-to-end delay. The authors in [48] have proposed a rate allocation algorithm for MDC combined with path diversity to minimize the overall distortion based on the estimated available bandwidth and packet loss rate. In [57], Liang, et al. propose Internet video transmission using path diversity and rate-distortion optimized reference picture selection.

Multicasting MDC video through multiple trees was first discussed in CoopNet [49] in the context of video multicast over peer-to-peer networks to prevent web servers from being overwhelmed by large number of requests. CoopNet uses a centralized tree management scheme, whereby each tree link is only a logical link, consisting of several physical links, and as such, is very inefficient in wireless ad hoc networks. Thus the approaches used in CoopNet are not suitable for video multicast over wireless ad hoc networks. Bansal and Zakhor [50][51] propose to stream multicast video over the Internet through multicast k-DAGs, which are Directed Acyclic Graphs in which each receiver has k parents. Similar to the unicast case, by streaming video from multiple parents, losses among various paths are not correlated, reducing the burstiness of lost packets, therefore improving the effectiveness of FEC. The authors also present a rate allocation and packet partitioning algorithm to adjust sending rates from each parent based on the loss characteristics of each path, in order to minimize the overall loss at the receiver.

1.3.2 Multipath Video Streaming over Wireless Networks

In [52], the authors propose to transmit layered video coding over multiple paths in wireless ad hoc networks. In their scheme, base layer and enhancement layer packets

are transmitted over different paths. Base layer packets are protected by automatic repeat request (ARQ). A lost base layer packet is retransmitted through the path where enhancement layer packets are transmitted. The authors in [53][54] propose to transmit different MDC substreams over different paths in wireless ad hoc networks. In [55], the authors compare the performance of transmitting MDC video and layered coding video over multiple paths. They conclude that when retransmission is not allowed, MDC outperforms layered coding. When retransmission is allowed, if the loss rate in two paths are similar, performance of MDC is similar to that of layered coding with ARQ, and when the loss rate of two paths are different, performance of MDC is worse than that of layered coding with ARQ.

In [36][59], Setton, et al. develop a congestion-optimized multipath routing algorithm, which finds multiple routes and performs optimal traffic partitioning to minimize a global congestion measure. In [60], Zhu and Girod propose a distributed algorithm to achieve congestion-minimized multipath routing. In [58], Zhu, et al. propose a rate allocation scheme to optimize the expected received video quality based on models of encoder's rate-distortion performance and network's rate-congestion tradeoffs. However the traffic model used in the above papers assumes that all nodes transmit simultaneously, and the capacity of each link is decided by the Signal Interference Noise Ratio (SINR). This model is different from the flow model decided by 802.11 MAC layer protocol, which is commonly used in wireless ad hoc networks.

In [62], Man and Li propose to distribute layered coding over multiple paths in wireless ad hoc networks. The authors give base layer packets higher priority over enhancement layer packets to improve the overall system performance. In [63], Miu, et al. try to enhance low latency video communication over wireless LANs by sending video through multiple Access Points (AP). They exploit the schemes which use multiple paths simultaneously or switch between multiple paths based on channel condition. Further in [64], Miu, et al. propose a fine-grained client-specific path

selection scheme among a set of neighboring APs. The scheme attempts to choose an AP based on short-term frame delivery statistics, with the goal of adapting to short-term variations using path diversity.

In [56], the authors propose a genetic algorithm based solution for multiple tree multicast streaming over wireless ad hoc networks, assuming that (a) they obtain each link's characteristics, and (b) consecutive links' packet loss rates are independent. The proposed scheme in [56] is too complicated to implement in practice.

1.4 Thesis Contributions and Outline

In this dissertation, we design and develop a framework for multipath unicast and multicast video streaming over wireless ad hoc networks. Transmitting video streams through multiple paths combats unpredictable packet loss in wireless ad hoc networks. Given multiple paths, a video stream can be divided into different substreams, which can be transmitted over different paths simultaneously. With careful design, each substream can experience relatively independent packet loss this way. The receiver can still achieve acceptable video quality with a portion of video packets received. Our proposed framework combines approaches from network routing protocols to video error control schemes in order to improve the quality of received video.

In Chapter 2, we present the general architecture for multipath video streaming over wireless ad hoc networks. There are four main components in the architecture: a video encoder with error control schemes, a traffic allocator, a multipath unicast/multicast routing, and a rate control scheme. Within this framework, we propose Robust MultiPath Source Routing (RMPSR) to support multipath streaming. RMPSR builds and maintains multiple nearly disjoint route sets for the video communication. RMPSR uses alternative sub-routes of each route set to salvage pack-

ets dropped in the middle of the network. We examine the performance of RMPSR through extensive NS simulations.

One main disadvantage of RMRSR is that it does not consider the interference between node disjoint paths. In Chapter 3, we propose a multipath routing protocol, which selects two node-disjoint paths with minimum concurrent Packet Drop Probability (PDP) of all path pairs. The proposed protocol both optimizes the worst case video quality of MDC streaming and improves performance of video streaming with FEC. We propose a model to estimate the concurrent PDP of two node-disjoint paths, given an estimate of cross traffic flows' rates, and bit rate of the video flow. We show that the above optimization is an NP-hard problem. We then propose a heuristic PDP aware multipath routing protocol based on our model. The proposed protocol obtains a path with approximately minimum PDP as the first path. After updating all the link metrics of the network graph, such as flow rate, the protocol finds the second path with approximately minimum PDP based on the new graph. The performance of the proposed scheme is shown to be close to that of the "optimal routing", and significantly better than that of the node-disjoint multipath routing and the shortest-widest routing through extensive NS simulations and actual experiments.

In Chapter 4, we study the problem of multicast streaming. We first show that for a given tree connectivity level, the difference between node densities required for single and double tree schemes is small. Thus building multiple trees does not significantly increase the cost of network deployment. We then propose a simple serial Multiple Disjoint Tree Multicast Routing (Serial MDTMR) protocol, which constructs two disjoint multicast trees sequentially in a distributed way, to facilitate multiple tree video multicast. This scheme results in reasonable tree connectivity while maintaining disjointness of two trees. However Serial MDTMR has a larger routing overhead and construction delay than conventional single tree multicast routing protocols, as it constructs the trees in a sequential manner. To alleviate these drawbacks, we further

propose parallel multiple nearly-disjoint trees multicast routing (Parallel MNTMR) in which nearly disjoint trees are constructed in parallel, and in a distributed way. Using the Parallel MNTMR, each receiver is able to always connect to two trees, regardless of the node density. Simulations show that multiple tree video multicast with both Serial MDTMR and Parallel MNTMR improve video quality significantly compared to single tree video multicast; at the same time routing overhead and construction delay of Parallel MNTMR is significantly lower than Serial MDTMR, and is approximately the same as that of a single tree multicast protocol.

Chapter 2

The Framework for Multipath Video Streaming over Wireless Ad Hoc Networks

2.1 Motivation

From the discussion in the first chapter, it can be concluded that supporting video applications in wireless ad hoc networks is a challenging task. A wireless ad hoc network lacks infrastructure, resulting in frequent route breakage as the nodes move. A path in a wireless ad hoc network generally consists of multiple wireless links, which has higher packet losses and smaller throughput, compared to single hop paths. To make it more challenging, video applications generally have stringent bandwidth, loss, delay, and delay jitter requirements. Thus it is hard to maintain an end-to-end route in wireless ad hoc networks, which is stable, error-free, and has sufficient bandwidth to support video applications.

In order to increase the robustness of video applications, we propose to transmit

video over wireless ad hoc networks through multiple paths instead. Given multiple paths, a video stream can be divided into different substreams, which can be transmitted over different paths simultaneously. With careful design, each substream can experience relatively independent packet loss. The receiver can achieve acceptable video quality when only receiving a portion of video packets. Using multiple paths can also potentially increase the aggregate end-to-end throughput for video applications.

The rest of this chapter is organized as follows. In Section 2.2, we introduce a general architecture for multipath video streaming over wireless ad hoc networks. In Section 2.3, we introduce RMPSR, which uses node-disjointness as a metric to build multiple paths. We show simulation results of RMPSR in Section 2.4, and conclude this chapter in Section 2.5.

2.2 The General Architecture

The general architecture for multipath video streaming over wireless ad hoc networks is shown in Figure 2.1. There are four main components in the architecture: a video encoder with error control, a traffic allocator, a multipath unicast/multicast routing protocol, and a rate control scheme.

The video encoder with error control generates video stream with protection to be transmitted over the wireless ad hoc network. Multiple Description Coding (MDC) and Forward Error Correction (FEC) are two common error control schemes for video applications. The basic idea behind MDC is to generate multiple compressed descriptions of the media in such a way that a reasonable reconstruction is achieved if any one of the multiple description is available for decoding, and the reconstruction quality is improved if more descriptions are available [65][67]. The main advantage of MDC

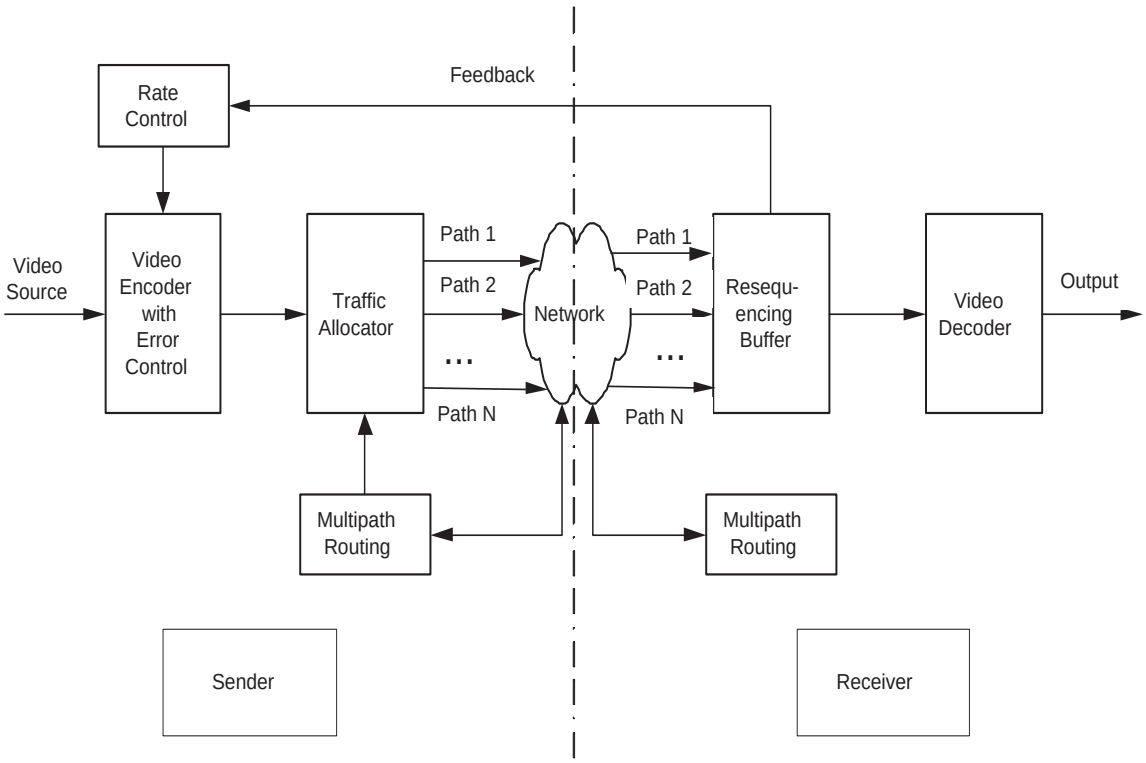


Figure 2.1. The General Architecture of Multipath Video Streaming over Wireless Ad Hoc Networks.

over layered coding is that no one specific description is needed in order to render the remaining descriptions useful. However, there is a penalty in coding efficiency in using MDC as compared to Single Description Coding (SDC). Specifically, for a given visual quality, the bit rate needed for MDC exceeds that of SDC depending on the number of descriptions. An excellent overview of the theoretical bounds and proposed MDC algorithms can be found in [68]. Because all the descriptions of MDC are equally important, it is not necessary to protect one stream over another. Also, because each description alone can provide a low but acceptable quality, no retransmission is required, making MDC more suitable for applications with stringent delay requirements, e.g. interactive video applications.

The basic idea behind FEC is to use redundant information to recover lost or corrupted information. A common class of FEC techniques is erasure codes [69].

Erasure codes assume that the receiver knows the exact location of lost packets. The receiver can detect the location of the lost packets using packet sequence number. In a typical erasure code technique, the sender encodes redundant packets before sending both the original and redundant packets. The receiver can reconstruct the original packets if it receives a certain fraction of total packets. Reed-Solomon (N, K) code, one well-known erasure code, takes K original packets, and generates $(N - K)$ redundant packets, resulting in N total packets. As long as the receiver receives K or more packets out of N packets, all of the K original packets can be recovered. One disadvantage of FEC is that the resulting video bitrate is larger than the original bitrate, because of redundant packets. Also, compared to MDC, FEC requires a longer delay, since the receiver has to receive at least K packets to recover the lost packet.

Traffic allocator decides how to distribute video packets into multiple paths in order to maximize the received video quality. For MDC video, since different descriptions are equally important, we simply distribute packets representing different descriptions into different paths. For FEC video, given a set of paths, it is possible to extend techniques proposed in [38] to wireless ad hoc networks.

The problem addressed by multipath unicast/multicast routing protocols in the architecture is how to build multiple paths/trees, in order to maximize the received video quality at the receive side. The key issue for the success of multipath streaming is to make packet drop over multiple paths as uncorrelated as possible. One natural metric for selecting multiple paths is to require them to be node disjoint, which means that there are no shared middle nodes among multiple paths. Packet drop due to link failure or path breakage caused by nodes' movement are independent among node disjoint paths. A more complicated metric is to minimize the concurrent PDP among multiple paths, in order to maximize the worst case video quality.

Due to the varying channel conditions and varying cross traffic flows, the available bandwidth for video applications varies from time to time. Thus a rate control scheme, which adjusts video sending rate according to channel condition and cross traffic flows, is a necessary component in the architecture. We propose a simple rate control scheme in Chapter 3, which follows the design philosophy of Additive Increase Multiplicative Decrease (AIMD) [83].

2.3 Robust Multipath Source Routing (RMPSR)

In this section, we propose RMPSR [74], which uses node-disjointness as a metric to build multiple paths in a wireless ad hoc networks. Packet drop due to channel error or node movement over two node-disjoint paths is assumed to be uncorrelated. RMPSR is a multipath extension to DSR [73]. RMPSR utilizes desirable features of other multipath routing approaches, and applies several new rules to address requirements of video applications. To describe RMPSR, we need to introduce two definitions first.

Definition 2.1: Two routes are nearly disjoint, if the ratio of the number of shared nodes to the number of the nodes of the shorter route is smaller than a prespecified threshold value.

Definition 2.2: As illustrated in Figure 2.2, a route set consists of one primary route, shown in solid arrows, and several alternative routes, shown in dotted arrows. The primary route connects the source node and the destination node; alternative routes connect intermediate nodes and the destination node. An alternative route and the corresponding subroute of the primary route, which connects the same starting node of the alternative route to the receiver, are required to be nearly disjoint. Two route sets are nearly disjoint, if corresponding primary routes are nearly disjoint.

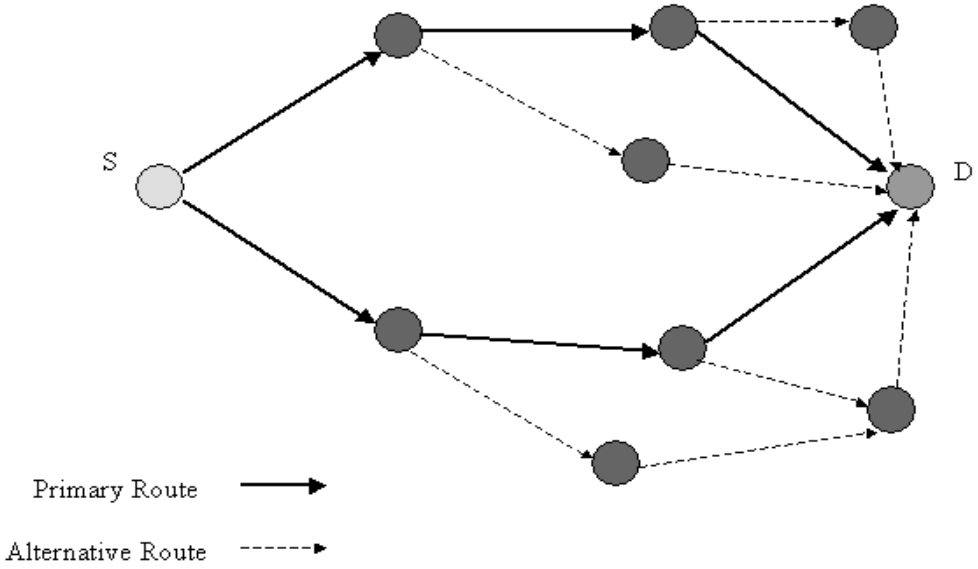


Figure 2.2. An illustration of the design goal of RMPSR.

To provide robustness for video applications, RMPSR builds and maintains multiple nearly disjoint route sets for the video communication almost all the time.

Similar to DSR[73] and SMR[14], we also use an on-demand source routing approach. The reasons for choosing source routing are that (a) it has been shown to outperform table based approaches in many scenarios[72], and (b) it is convenient to build multiple disjoint route sets using source routing, since the destination node knows the entire path of all the available routes.

When a source needs to establish routes to a destination, it originates a route discovery process. Route discovery process typically involves a network-wide flood of Route Query (RREQ) packets targeting the destination, and the return of Route REPly (RREP) packets from the destination. In DSR protocol, duplicate copies of the RREQ packet at the intermediate nodes are discarded. Although this approach can

minimize the number of RREQ packets for one route discovery process, it also reduces the probability of discovering multiple disjoint routes[14][17]. In order to increase the probability of discovering multiple disjoint routes, while keeping the number of RREQ packets small, we use a modified form of the RREQ packet forwarding scheme in [17]. When an intermediate node receives a Route Query (RREQ) packet, it forwards the packet, if the packet is not duplicate or the path included in this packet is disjoint with paths included in previous RREQ packets and the hop count of the path is not larger than hop counts of previous paths. This approach increases the probability of discovering multiple disjoint routes at the expense of an increase in control overhead.

We choose to construct route sets at the destination node. There are two route caches in the destination node. After receiving the first RREQ packet, the destination records the route carried in the packet in both cache 1 and cache 2, and sends back the route as a primary route to the source node. Thus the route discovery latency required by the nature of on-demand routing protocols is minimized, which is important for video applications. For a short period of time after this, when the destination receives a new RREQ packet, it first puts the route into cache 2, then it returns the route as a primary route to the source node and records it into cache 1, if the route is nearly disjoint to all the primary routes in cache 1. After the destination has waited for the certain amount of time, it stops receiving newly arriving RREQ packets in this round, and computes alternative routes for subroutes of every primary route in cache 1 based on route information in cache 2. The scheme does not require the alternative route to be disjoint with subroutes of other primary routes though, since these alternative routes are only used to salvage packets. Thus, although not all the subroutes are guaranteed to have an alternative route, this scheme maximizes the probability of finding an alternative route for every subroute. Then the destination node returns all the alternative routes to corresponding intermediate nodes. In our implementation, we set the waiting time to be 100 ms.

Our proposed RMPSR uses a per-packet allocation scheme to distribute video packets over two primary routes of two route sets. When one transmitting primary route is broken, the intermediate node that corresponds to the broken link will send a Route ERRor (RERR) packet to the source node. Upon receiving the RERR packet, the source node removes the broken primary route from its route cache, and switches the transmission to another primary route.

To support video applications better, we further introduce three new schemes.

1. When the transmission route is broken, alternative routes in the same route set are used to salvage mid-way packets. Unlike traditional salvaging schemes, rather than transmitting new packets, alternative routes are only used for salvaging ongoing packets. The reason is that routes in the same route set are correlated, so if the primary route is broken, it is likely that alternative routes have been broken or will be broken shortly. Thus in order to avoid further loss of future packets, the transmission is switched to another primary route as soon as the transmitting primary route is broken. When salvaging a packet, the number of times that it has been salvaged is maintained in the packet, to prevent a packet from being salvaged indefinitely . Since video packets have strict time deadlines to arrive at the destination, the maximum salvage number should be set to a small number, which is two in our implementation.

2. RMPSR triggers new route request process before the connectivity is entirely lost in order to reduce the number of temporary network outages during the transmission. In our implementation, the protocol triggers new route request process when there is only one primary route left in the route cache of the sender. Since each temporary network outage may cause a "freeze" in video playback, this scheme enhances the performance of video at the expense of additional control overhead.

3. Similar to other multipath extensions, RMPSR increases the probability of discovering multiple disjoint routes at the expense of an increase in control overhead. To alleviate the impact of routing overhead on the network, both RMPSR and DSR are deployed at each node with different classes of traffic being handled by different routing protocols. Video traffic is given higher priority using RMRSR, while other traffic is given lower priority using DSR. This scheme helps to lower the overall routing overhead, and to maintain high quality of video applications when the amount of other data traffic in the network increases.

RMPSR has several advantages. First, it builds nearly node disjoint route sets for video applications, which makes packet drop due to channel error, link failure and route breakage uncorrelated. This property enhances the effectiveness of FEC [38], since the average length of burstiness is decreased. It also enhances the worst case video quality of MDC, since the probability that both descriptions are dropped is reduced. Second, the salvaging performed by sub-routes further reduces the number of dropped packets because of link failure and route breakage. Third, RMPSR can act similar to the Differentiated Services (DiffServ) [75] by assigning higher and lower priority video applications with different routing protocols, thus maintaining high video quality with a low overall routing overhead. Last but not least, RMPSR is very simple. It is easy to implement RMPSR in a real world wireless ad hoc network testbed. One main disadvantage of RMRSR is that it does not consider the interference between node disjoint paths. As such, packet drop due to congestion is still correlated. Networking scenarios in which packet drop is primarily due to congestion, RMPSR does not significantly outperform other protocols. We will address this issue shortly in Chapter 3.

2.4 Simulation Results

In this section, we show the effectiveness of RMPSR through extensive NS simulations. We show that RMPSR improves both the performance of interactive video applications with MDC and video on demand applications with FEC.

2.4.1 Performance evaluation for interactive video applications

In this section, we test the performance of interactive video applications with MDC using RMPSR. Each packet of an interactive video application has a strict delay constraint. If a video packet arrives later than its deadline, it is useless to the decoder. As such, it is desirable to drop late packets at the sender or in the middle nodes rather than attempt to transmit them after the deadline has passed. So techniques based on retransmission are unsuitable for interactive video applications, as they both increase the delay.

We use a simulation model based on NS-2 [76]. The Monarch research group in CMU has extended the NS-2 network simulator to include physical layer, link layer and MAC layer models to support multi-hop wireless network simulations [72]. The distributed coordination function (DCF) of IEEE 802.11 for wireless LANs is used as the MAC layer. The radio model is based on the Lucent/Agere WaveLAN/OriNOCO IEEE 802.11 product, which is a shared-media radio with a transmission rate of 2 Mbps, and a radio range of 250 meters. A detailed description of the simulation environment and the models is available in [72].

The random waypoint model [72] is used to model mobility. Each node starts its journey from a random location to a random destination with a randomly chosen speed, which is uniformly distributed between 0 and maximum speed. Once the

destination is reached, another random destination is targeted after a pause. We only consider the continuous mobility case. To change the mobility level of the network, we vary the maximum speed from 0 m/s to 15 m/s, i.e. from a static network scenario to a highly dynamic network scenario.

We simulate a 60 node network in a 1200 meters by 800 meters rectangular region. Simulations are run for five hours. There are five random 12 kbps cross sessions in the network.

MP-MDVC [65], an MDC technique based on matching pursuits (MP) signal decomposition, is used to code MDC sequence. We encode each frame into two descriptions. Intra-frame encoding is identical for both descriptions. For each description, an I-frame is packetized into two packets, and a P-frame is packetized into one packet. As such, every packet is smaller than the Maximum Transmission Unit of 802.11 wireless networks. Thus the routing protocol does not need to consider fragmentation and reassembly in IP layer. We use standard MPEG QCIF sequence Foreman coded at 146 kbps at 18 fps with Group Of Pictures (GOP) size of 15 for our experiments. The playback deadline of each frame is 150 milliseconds (ms) after it is generated.

To describe the metrics we use, we need to introduce the definition of bad frame first.

Definition 2.3: A description of an I-frame is decodable at the playback deadline, if all packets corresponding to the description is received. A description of a P-frame is decodable, if at the playback deadline, both the packet corresponding to the description is received and the same description of the previous frame is decodable. A frame of a MDC stream is called a bad frame, if neither one of its two descriptions is decodable; a frame of a SDC stream is called a bad frame, if it is not decodable.

We evaluate the performance of interactive video applications with RMPSR using the following metrics:

a. **The ratio of bad frames:** The ratio of the number of bad frames to the total number of frames. We use this metric rather than packet delivery ratio because it is more indicative of the quality of received video due to the following two reasons. First it considers the dependency between different frames. For example, for SDC, if an I-frame is a bad frame, all the subsequent P-frames in the same GOP are considered to be bad frames, regardless of whether or not packets representing those P-frames are received. Second this metric reflects the fact that MDC can to some extent conceal the undesirable effects caused by missing packets, by decoding and displaying one of the two descriptions.

b. **The number of bad periods:** A bad period consists of a number of contiguous bad frames. This metric reflects the number of times received video is interrupted by the bad frames.

c. **Normalized packet overhead:** The total number of control packets transmitted by any node in the network, divided by the total number of video packets received by all the receivers. This metric represents the control packet overhead of the routing protocol normalized by the successful video packets received.

We compare the following three schemes:

a. DSR [73] with single path video transmission;

b. SMR [14] with multipath video transmission;

c. RMPSR with multipath video transmission.

We transmit MDC video with all three schemes, in order to compare the effectiveness of routing protocols fairly.

The ratio of the number of bad frames over the number of all frames is shown in Figure 2.3, and the number of bad periods, consisting of contiguous bad frames, is

shown in Figure 2.4. The smaller the two metrics are, the better the video experience. As shown in Figures 2.3 and 2.4, the performance of interactive video is enhanced using RMPSR as compared to SMR and DSR, in the sense that both the ratio of bad frames and the number of bad periods are reduced greatly. MDC can potentially be a suitable match for multipath communication in a sense that even when one path is broken, packets corresponding to the other description on the other path can still arrive at the receiver on time. With MDC, quality of these frames is still acceptable. To fully utilize this property of MDC, it is important for multipath routing protocols to maintain multiple routes as long as possible. RMPSR builds more than two nearly disjoint routes in the route request process and triggers new route request process before all the routes are broken. These steps help RMPSR maintain multiple routes longer than SMR and DSR. Another reason for the enhanced performance of RMPSR is its effective packet salvaging scheme. For example, as shown in Table 2.1, when the maximum speed is 12.5 m/s, the ratio of the number of salvaged packets to the total number of transmitted packets over a ten hour simulation period is much larger for RMPSR than that for SMR and DSR.

Figure 2.5 shows comparison of normalized control packets among the three routing protocols. DSR has the least amount of overhead. Next is RMPSR, followed by SMR. The number of control packets increases with the mobility level of the network. In a typical scenario, there are small number of video sessions and a large number of other data sessions. Our approach is to apply RMPSR to video sessions and DSR to other data sessions. Thus, extra routing overhead for RMPSR does not greatly affect the performance of video applications. RMPSR increases the average time between route query processes by constructing multiple routes in one route query round, thus reducing control packets as compared to SMR.

Our simulations with ten 12 kbps cross traffic flows also confirm that RMPSR

outperforms both SMR [14] and DSR [73] for interactive video applications over wireless ad hoc networks.

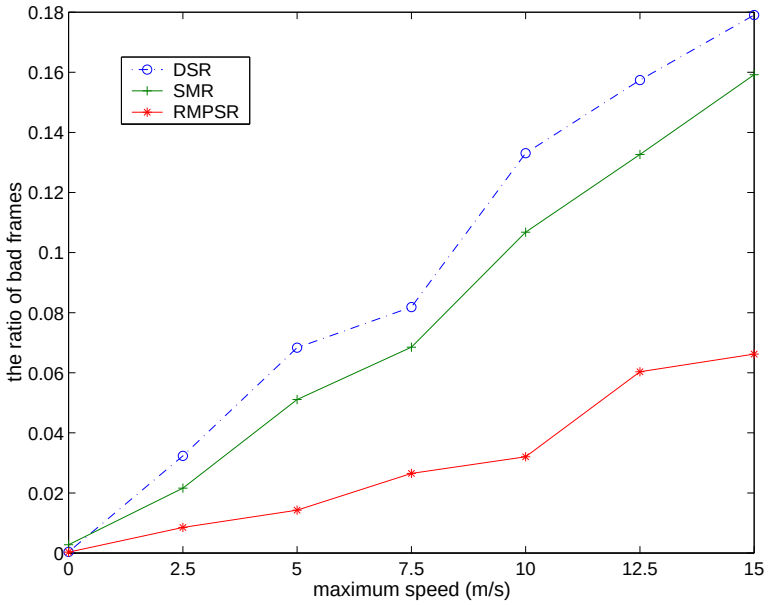


Figure 2.3. Performance evaluation for interactive video applications using MDC: The Ratio of Bad Frames as a function of maximum speed for DSR, SMR and RMPSR

Table 2.1. Comparison of salvaged packets.

Protocol	RMPSR	SMR	DSR
Ratio of salvaged packets	0.0078	0	0.0023

2.4.2 Performance evaluation of video on-demand applications

We now compare RMPSR, DSR, and SMR for video on-demand applications with FEC. Each simulation is run for 600 seconds, and results are averaged over 50 runs. The bit rate of video stream is 144 kbps. We use (100,75) Reed-Solomon erasure code, which takes 75 data packets and produces 25 redundant packets in one block. Thus the total sending rate of video stream is 192 kbps. We use a simple per-packet allocation scheme, which distributes odd numbered packets into the first route and

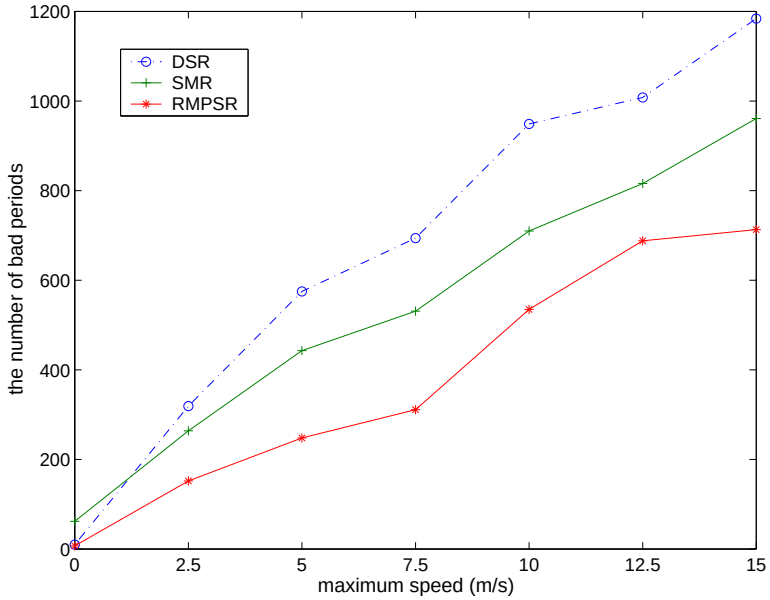


Figure 2.4. Performance evaluation for interactive video applications using MDC: The Number of Bad Periods as a function of maximum speed for DSR, SMR and RMPSR

even ones into the second route. The receiver first pre-buffers 5 seconds' worth of video packets before starting to play back. If a packet arrives after its playback deadline, it is discarded. When the playback buffer in the receiver is empty, the receiver stops playing temporarily, and re-buffers 5 seconds' worth of video packets before re-starting the playback. There are five random 12 kbps cross traffic sessions in the network. Other simulation settings are the same as that of simulations in the previous section.

We evaluate the performance of video on demand applications with RMPSR using the following metrics:

a. **Goodput Ratio:** Goodput ratio is ratio of the number of video packets played at the receiver to those transmitted from the video source. Not every received video packet is useful for decoding, because each video packet has a strict deadline, and

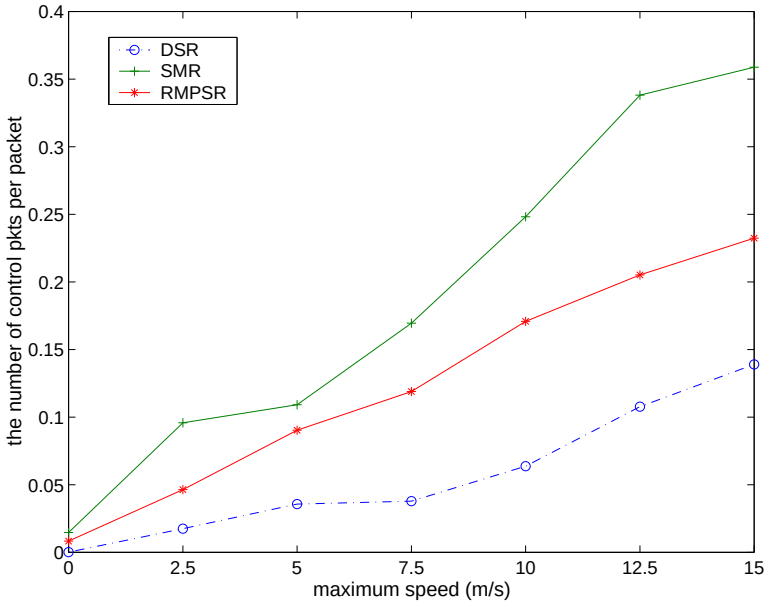


Figure 2.5. Normalized Control Packets of DSR, SMR and RMPSR as a function of maximum speed

is discarded if it arrives after its deadline. Therefore, we use this metric rather than the successful packet delivery ratio to represent the received video quality.

b. **The Number of Rebufferings:** Number of rebufferings denotes how many times a receiver freezes during one experiment. For video on demand applications, before playback, the receiver pre-buffers a certain length of video. When the network gets congested or the path is broken, the received bitrate is smaller than the transmitting bitrate. In this case, the receiver has to playback from the buffered video. When the video buffer becomes empty, the receiver stops the playback, rebuffers a certain amount of video, and then continues to playback. This kind of freeze is quite annoying to end users, and thus is an important measure to video quality.

Figures 2.6 and 2.7 show goodput ratio and the number of rebufferings as a function of maximum speed for the three routing protocols. In each figure, the lower plot shows the number of simulations in which a particular scheme performs as good or

better than the competing schemes. There are a total of 50 simulations. As seen in Figures 2.6 and 2.7, performance of all three protocols is similar in the static network, while their performance gap widens in more dynamic scenarios. Specifically, in over 90% of all scenarios, the RMPSR performs the best or equal to the other two protocols. As expected, as the maximum moving speed becomes larger, goodput ratio of all three protocols goes down, and "number of rebufferings" goes up. However, RMPSR loses much fewer packets than either SMR or DSR protocol, and suffers fewer freezes at the receiver in dynamic scenarios. Since RMPSR builds multiple disjoint route sets, the connection is less likely to be broken than DSR or SMR. RMPSR also salvages packets at intermediate nodes with alternative routes, and triggers new route request process ahead of time. These steps further help RMPSR perform better in delay sensitive video applications.

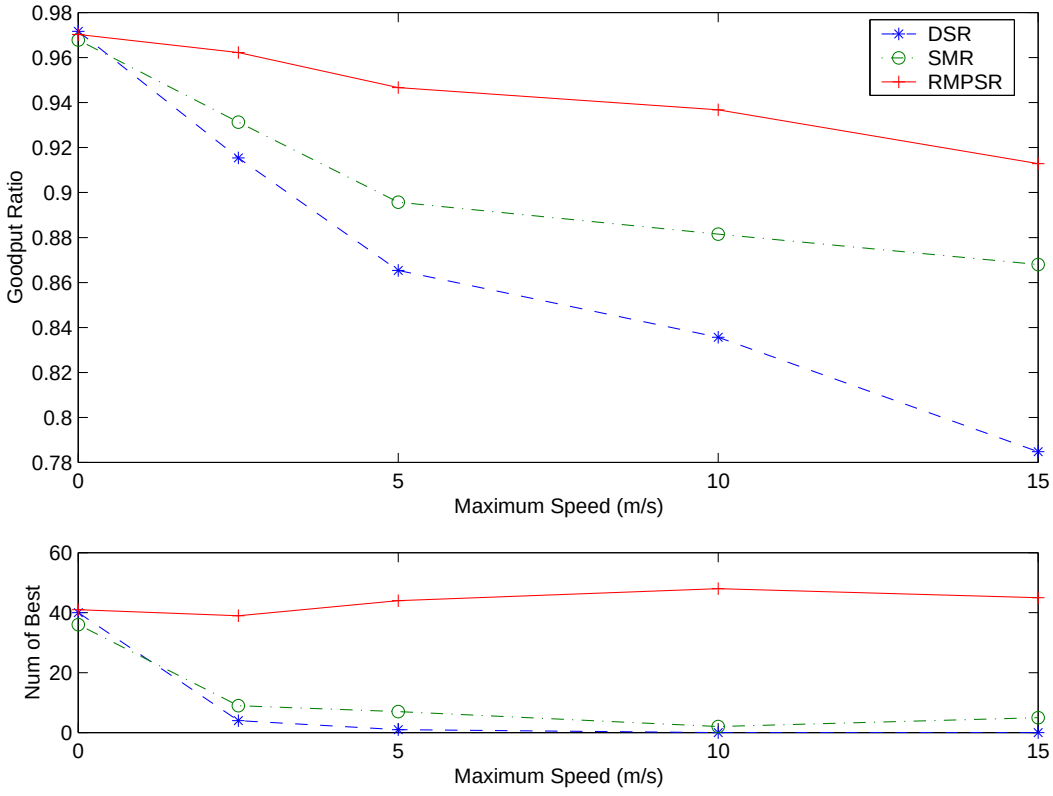


Figure 2.6. Performance evaluation for video on-demand applications using FEC: Goodput Ratio as a function of maximum speed for DSR, SMR and RMPSR

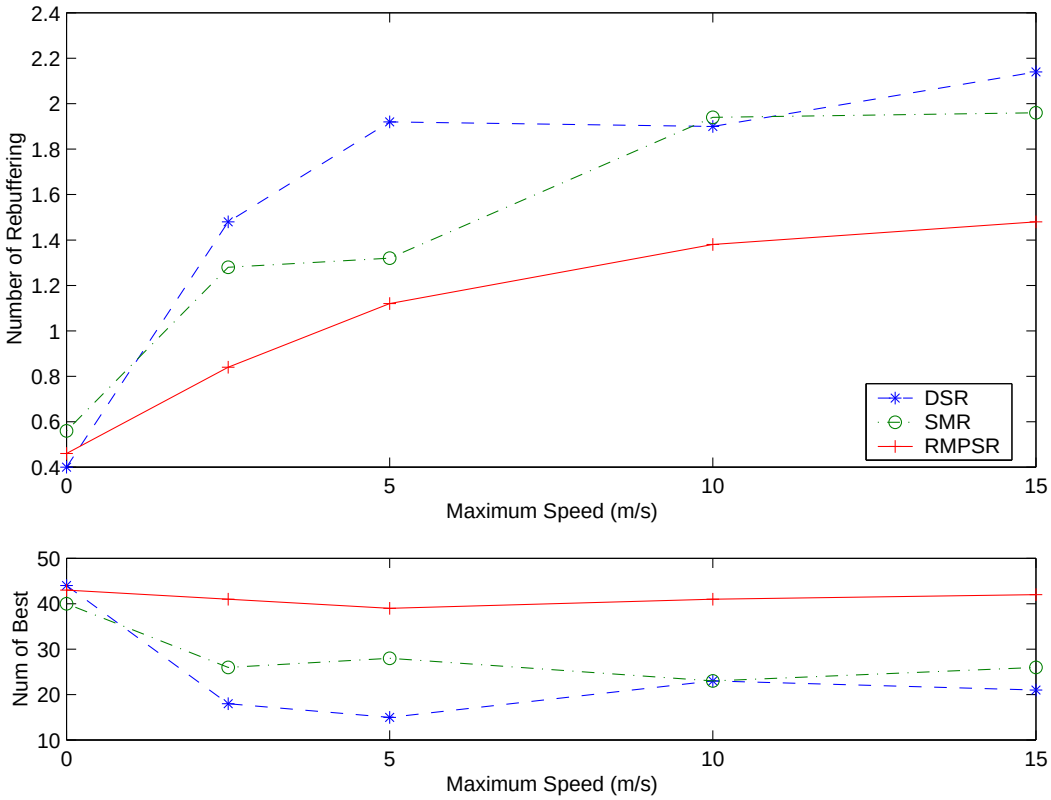


Figure 2.7. Performance evaluation for video on-demand applications using FEC: Number of Rebufferings as a function of maximum speed for DSR, SMR and RMPSR

2.4.3 Comparison of MDC with SDC

In this subsection, we compare performance of multipath transmission of MDC and SDC content. For the same quality of video, bit rate of MDC has been shown to be approximately 30% - 60% larger than that of SDC [65][67]. Coding efficiency of MDC depends on encoding schemes and GOP size. Since an I-frame is much larger than a P-frame and MDC encodes an I-frame twice, i.e. one for each description, if we increase the GOP size, the bit rate of MDC is lowered for the same video quality. However we can not set GOP size too large for wireless scenarios, because packet drop is common and we need to refresh with I-frame from time to time. This is a tradeoff between coding efficiency and error resilience. We set the frame rate as 18 fps, and GOP size as 15. Standard MPEG QCIF sequence Foreman is coded with MP-MDVC

[65] at 146.0 kbps for MDC sequence, and is coded with Matching Pursuit coding [66] at 92.7 kbps for SDC sequence. The PSNR of two description MDC and SDC are approximately same. RMPSR is used with both SDC and MDC. Other simulation settings are the same as the simulation for interactive video applications.

We compare the following two schemes:

a. MDC with multipath transmission: each description of MDC is sent through a different path;

b. SDC with multipath transmission: packets of SDC stream alternate between two paths. When one path is broken, the source will select a new path from the route cache.

Figure 2.8 compares performance of MDC and SDC as a function of maximum speed. Video traces of MDC and SDC described above are used in the simulation. As shown in Figure 2.8(a), SDC achieves fewer number of bad frames than MDC in this scenario. When only one path is broken, MDC can conceal the effect by decoding only one description. Thus MDC could potentially reduce the number of bad frames as compared to SDC schemes if they could both result in the same overall bitrate. However SDC results in a lower level of network congestion compared to MDC due to its lower bit rate and higher coding efficiency, and in the mean time, RMPSR reduces the number of packet drop significantly, so SDC results in lower ratio of bad frames than MDC in this scenario. From Figure 2.8(b), we see that MDC has fewer bad periods than SDC. However taking into account Figure 2.8(a), MDC has longer average bad periods as compared to SDC. This indicates that MDC conceals scattered packet losses, thus reducing the number of short bad periods, while the lower bit rate of SDC helps to reduce the number of long consecutive packet losses.

Figure 2.9 compares performance of MDC and SDC as a function of bit rate for a

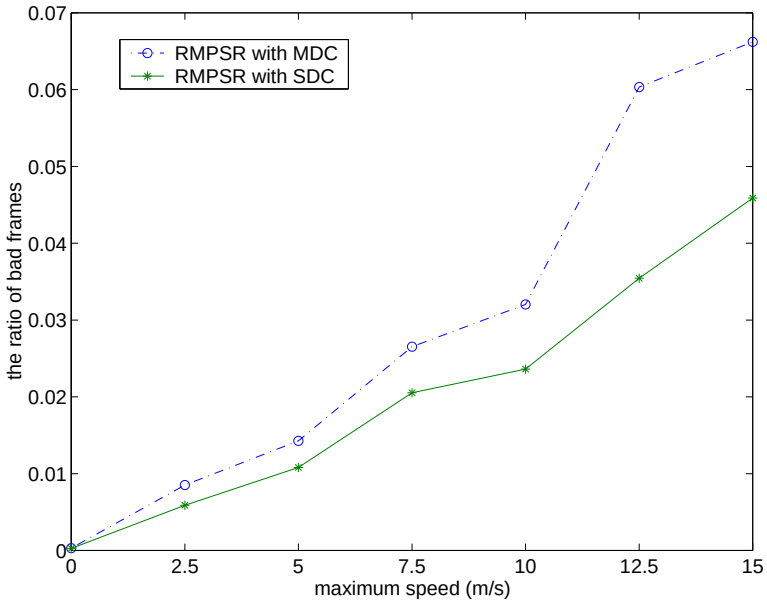
scenario with maximum speed of 5 m/s. The x-axis corresponds to bit rate for SDC, while corresponding bit rates of MDC are assumed to be 33% higher. This assumption is reasonable with a large GOP size. The ratio of bad frames increases for both schemes with bit rate. When bit rates are low enough not to cause congestion, MDC outperforms SDC. When the bit rate is around a crossover threshold value, which is about 160 kbps for SDC and 210 kbps for MDC in this scenario, the performance of two schemes is almost the same. For bit rates above the threshold value, SDC outperforms MDC. We also carried out simulations with different level of cross traffic and mobility of the wireless network, and have reached similar conclusions. We have empirically found the crossover threshold value to depend on available bandwidth, which is a dynamic value in wireless ad hoc networks, on cross traffic, link capacity and mobility level of the network.

From simulation results and the analysis, we see that as compared to SDC with multipath transmission, MDC scheme does not necessarily improve quality of interactive video applications over wireless ad hoc networks. The lower bit rate and fewer number of packets of SDC scheme in some situations make it suffer less from possible network congestion, offsetting the inherent robustness of MDC.

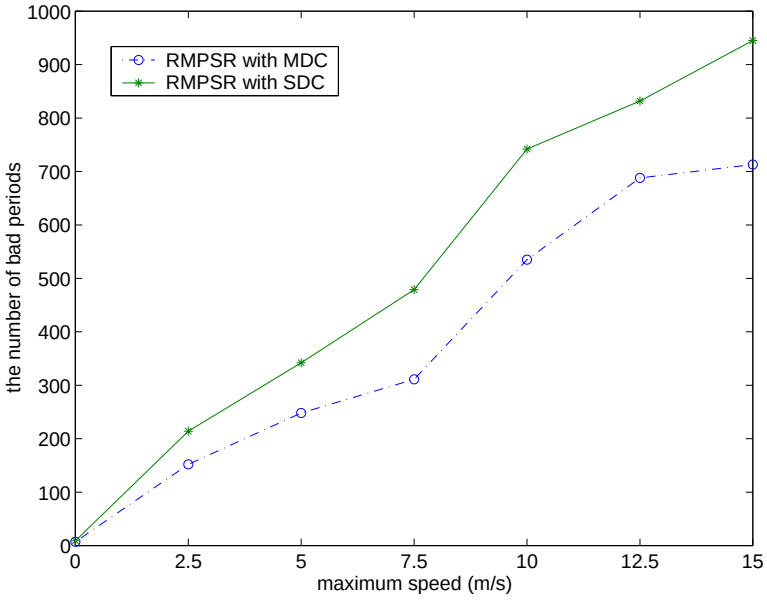
2.5 Conclusions

In this chapter, we propose a general architecture for multipath video streaming over wireless ad hoc networks in order to enhance the robustness of video applications. The general architecture includes a video encoder with error control, a traffic allocator, a multipath unicast/multicast routing protocol, and a rate control scheme. We also introduce RMPSR, which uses node-disjointness as a metric to build multiple paths, and applies a few other new techniques, e.g. salvaging sub-routes, preemptive routing, and DiffServ through applying different routing protocols to different classes

of applications. We show that RMPSR improves both the performance of interactive video applications with MDC and video on demand applications with FEC through extensive NS simulations. We also compare performance of multipath transmission of MDC and SDC content, and conclude that there exists one crossover threshold value, above which SDC outperforms MDC; otherwise, MDC outperforms SDC.

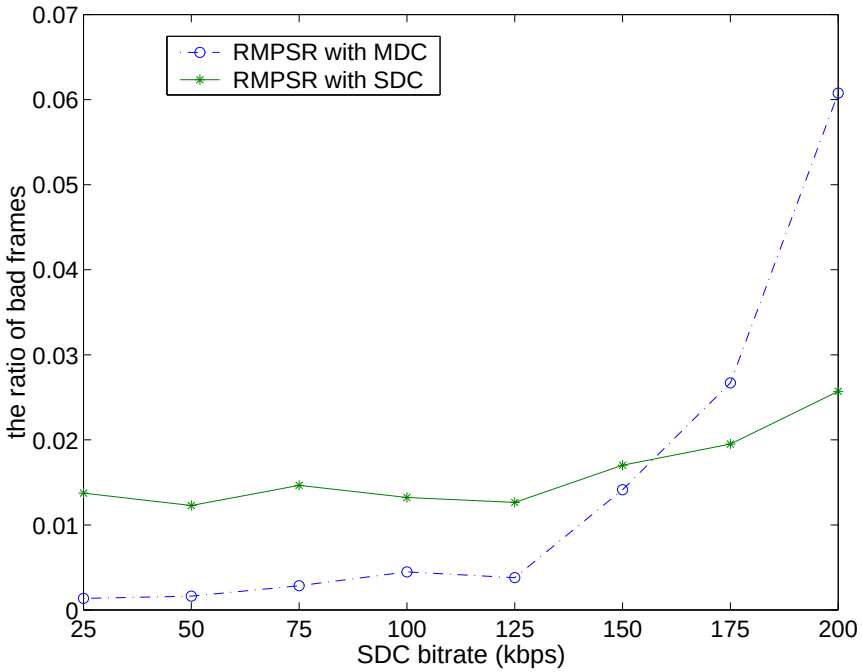


(a)



(b)

Figure 2.8. Comparing MDC with SDC for interactive video applications: (a) Ratio of Bad Frames; (b) Number of Bad Periods as a function of maximum speed.



(a)

Figure 2.9. Comparing the ratio of bad frames for MDC with SDC as a function of bit rate for maximum speed of 5 m/s.

Chapter 3

Path Selection for Optimal Multipath Video Streaming

As discussed earlier, there are many challenges to support video communication over wireless ad hoc networks. An end-to-end connection route in wireless ad hoc networks generally consists of multiple wireless links. As such, it has much smaller throughput and higher random packet loss than single hop wireless connections in a wireless network with an infrastructure. Due to the mobility of wireless nodes, the established connection routes between senders and receivers are likely to be broken during video transmission, causing interruptions, freezes, or jerkiness in the received video signal. These constraints and challenges, in combination with the delay and loss sensitive nature of video applications, make video communication over wireless ad hoc networks a challenging proposition.

Recent efforts on multipath routing of MDC video have successfully demonstrated improved robustness in video communication applications[33][34][61][74]; this is done either by assuming that the set of paths is given, or by simply selecting two node/link disjoint paths. Few recent works have addressed the difficult problem of selecting

optimal paths for MDC video streaming [43][45][46]. Begen et al. have studied how to select multiple paths that maximize the *average* video quality at clients on Internet overlay networks[43]. Mao et al. have further proposed a meta-heuristic approach based on Genetic Algorithms to solve the path selection problem[45]. In [46], the authors propose to select two paths with minimal correlation for MDC streaming over Internet overlay networks. These approaches however are too complex to be performed in real-time. Also these models consider neither the interference of flows on neighboring links, nor the influence of the incoming video flow on the characteristics of links; this is needed in current wireless ad hoc networks, because a new video flow generally consumes a large percentage of wireless resource, thus changing characteristics of wireless links significantly.

In this chapter, we propose a technique for choosing two node-disjoint paths, which achieve minimum concurrent PDP of all path pairs. Our motivation is to increase robustness of video applications over wireless ad hoc networks. While most of our simulation results refer to MDC, our basic results and conclusions can be easily extended to FEC video as well. For MDC streaming, different descriptions are transmitted on different paths in order to fully utilize path diversity. Streaming over the Path Pair with Minimum concurrent Packet Drop Probability, denoted by PP_MDP, minimizes the probability of concurrent loss of all the descriptions, thus optimizing the worst case MDC video quality over all times. For FEC streaming, concurrent packet drop over the selected PP_MDP can be shown to be less likely than that of simple node disjoint paths, resulting in lower unrecoverable probability.

In this chapter, we use a *conflict graph* [77][78][79] to model effects of interference between different wireless links. The conflict graph indicates which groups of links interfere with each other, and hence can not be active simultaneously. We propose a model to estimate the concurrent PDP of two node-disjoint paths, given an estimate of cross traffic flows' rates, and bit rate of the video flow. We show that the above

optimization is an NP-hard problem. We then propose a heuristic PDP aware multipath routing protocol based on our model, whose performance is shown to be close to that of the "optimal routing", and significantly better than that of the node-disjoint multipath routing, and the shortest-widest routing.

In Chapter 2, we propose RMPSR, which builds and maintains multiple nearly disjoint route sets for the video communication. However RMRSR does not consider the interference between node disjoint paths. As such, packet drop due to congestion is still correlated. To overcome the disadvantage of RMPSR, in this chapter, we analyze packet drop probability due to congestion and propose a framework for selecting two node-disjoint paths with minimum concurrent PDP of all path pairs. The new approach outperforms RMPSR when packet drop is primarily due to congestion.

The rest of this chapter is organized as follows. In Section 3.1, we introduce the optimal multipath selection problem. We propose the Interference aWare Multipath Routing (IWM) in Section 3.2. We represent the simulation and actual experiment results in Section 3.3 and 3.4 respectively. In Section 3.5, we conclude this chapter.

3.1 Optimal Multipath Selection Problem

Our goal is to minimize concurrent PDP of two node-disjoint paths in a wireless ad hoc network which is equivalent to optimizing the worst case video quality at clients. The node-disjoint constraint is useful for mobile wireless ad hoc networks, because it can significantly decorrelate packet drop between different paths.

3.1.1 Envisioned Network Model

We consider a wireless ad-hoc network with N nodes arbitrarily distributed in a plane. Let n_i, $1 \leq i \leq N$ denote the nodes, and d_{ij} denote the distance between nodes

n_i and n_j. Each node is equipped with a radio with communication range r, and a potentially larger interference range ω. Our interference model is similar to that of the protocol model introduced in [78][79], with changes to reflect the implementation of 802.11 MAC protocol in NS-2 [76].

Protocol Interference Model: Suppose node n_i wishes to transmit to node n_j. We use SS_{ij} to denote the signal strength of n_i's transmission as received at node n_j. The transmission between nodes n_i and n_j is successful if all of the following conditions are satisfied:

- $d_{ij} \leq r$; intuitively this is equivalent to nodes n_i and n_j being within each other's communication range.
- Any node n_k, such that $d_{ki} \leq \omega$ is not transmitting. This is motivated by the CSMA/CA scheme in the 802.11 MAC protocol, which states that node n_i can not transmit if any node in its interference range is transmitting.
- Any node n_k, such that $\frac{SS_{ij}}{SS_{kj}} \leq CPThresh$, is not transmitting, where $CPThresh$ denotes the capture threshold, with default value of 10 in NS-2. This implies that no node with sufficiently large signal strength interfering with link n_i to n_j is transmitting.

We can easily make minor changes to the protocol interference model, if the underlying MAC layer protocol is changed.

A wireless ad hoc network can be modelled as a directed graph $G(V, E)$, whose vertices V correspond to wireless stations, and the edges E correspond to wireless links. There is a link from vertex n_i to vertex n_j if and only if $d_{ij} < r$. As in [77][78][79], we make use of a "conflict graph" to model the interference relationship between different links of a network. Every directed link in the graph $G(V, E)$ is represented by a node in the directed conflict graph $CG(V^C, E^C)$. If the transmission

over link l_{ij} makes the transmission over link l_{kl} unsuccessful, link l_{ij} interferes with link l_{kl}, resulting in a directed link from node l_{ij} to node l_{kl} in the conflict graph. To avoid confusion, we use the terms "node" and "link" in reference to the connectivity graph $G(V, E)$, while using "CG-node" and "CG-link" to refer the conflict graph $CG(V^C, E^C)$. Figure 3.1(a) shows an example of a conflict graph. CG-nodes 1 through 5 correspond to five wireless links in the original wireless network. The wireless link represented by CG-node 1 interferes with wireless links represented by CG-nodes 2, 3, 4 and 5, while none of the other CG-nodes interfere with each other.

We have summarized notations used in this chapter in Table 4.1.

Table 3.1. Notations used in Chapter 3

Symbols	Definition
$G(V, E)$	graph representation of the network
$CG(V^C, E^C)$	corresponding conflict graph of the network
N	the number of nodes in the network
C	wireless channel capacity
d_{ij}	the distance between nodes n_i and n_j
r	communication range
ω	interference range
SS_{ij}	the signal strength of n_i's transmission as received at node n_j
$CPThresh$	capture threshold
l_{ij} or (i, j)	wireless link connecting node n_i and n_j
F	flow rates vector
F_i	combined rate of all flows over link l_i
$I(l_{ij})$	interfering links set of link l_{ij}
CF_k	the combined flow rate over all links in the independent set IS_k
$P_{S,D}$	a path connecting nodes N_S and N_D
$N_{S,D}$	the set of the nodes on path $P_{S,D}$
$L_{S,D}$	the set of the links on path $P_{S,D}$
$\mathbf{x}$ and $\mathbf{y}$	indication vectors for path $P^1_{S,D}$ and $P^2_{S,D}$ respectively

3.1.2 The Optimal Multipath Selection Problem

The main objective of the optimal multipath selection problem for MDC video streaming over wireless ad hoc networks is to select two node-disjoint paths with minimum concurrent PDP.

Definition 3.1: A path $P_{S,D}$ connecting nodes N_S and N_D in a graph $G(V, E)$, is a sequence of nodes $v_1, \ldots v_n$, which satisfy the following two conditions. (a) $\forall i, 1 \leq i < n$, we have $(v_i, v_{i+1}) \in E$; (b) no node appears more than once. The set of the nodes on this path is represented by $N_{S,D} \subseteq V$, and the set of the links on this path is denoted by $L_{S,D} \subseteq E$.

Let $P^1_{S,D}$ and $P^2_{S,D}$ be any two paths connecting nodes N_S and N_D, $L^1_{S,D}$ and $L^2_{S,D}$ denote the set of links on each path respectively, and $N^1_{S,D}$ and $N^2_{S,D}$ denote the set of the nodes on each path respectively. We define two indication vectors $\mathbf{x} = (\ldots, x_{ij}, \ldots)^T$ and $\mathbf{y} = (\ldots, y_{ij}, \ldots)^T$ to represent $P^1_{S,D}$ and $P^2_{S,D}$ respectively, where x_{ij} is set to 1 if link $(i, j) \in L^1_{S,D}$ and is set to 0 otherwise. Similarly y_{ij} is set to 1 if link $(i, j) \in L^2_{S,D}$ and is set to 0 otherwise. The dimension of vectors $\mathbf{x}$ and $\mathbf{y}$ is the number of links in the graph.

The optimal multipath selection of two node-disjoint paths with minimum concurrent PDP can be formulated as follows:

$$\text{Minimize } P_{\text{drop}}(P^1_{S,D}; P^2_{S,D})$$

$$\text{with respect to } x_{ij}, y_{mn} \in \{0, 1\},\ \forall (i, j), (m, n) \in E$$

Subject to

$$\sum_{j:(i,j)\in E} x_{ij} - \sum_{j:(j,i)\in E} x_{ji} = \begin{cases} 1 & i = N_S \\ -1 & i = N_D \\ 0 & otherwise \end{cases} \tag{3.1}$$

and

$$\sum_{n:(m,n)\in E} y_{mn} - \sum_{n:(n,m)\in E} y_{nm} = \begin{cases} 1 & m = N_S \\ -1 & m = N_D \\ 0 & otherwise \end{cases} \tag{3.2}$$

$$N^1_{S,D} \cap N^2_{S,D} = \{N_S, N_D\} \tag{3.3}$$

Equation (3.1) is the flow constraint in order to guarantee the first path connects the source S and the destination D. It represents that (a) for each node in the first path, except the source and the destination, the number of incoming links is equal to the number of outgoing links; (b) for the source node, the number of outgoing links is 1; (c) for the destination node, the number of incoming links is 1. Similarly, Equation (3.2) is the flow constraint for the second path. Equation (3.3) is the node-disjoint constraint to ensure that the two selected paths do not share common middle nodes.

We can show the following claim for the optimal multipath selection problem.

*Claim 3.1:*The optimal multipath selection over wireless ad hoc networks as defined above is NP-hard.

The proof is shown in the appendix.

Since the optimal multipath selection problem is NP-hard, the required computation needed by the optimal solution will be very high. One approach is to enumerate all possible pairs of node-disjoint paths from a source N_S to a destination N_D, estimate the concurrent PDP for each path pair using the scheme to be discussed in Section 3.1.3, and choose the best one. We refer to this solution as the Optimal Multipath Routing (OMR). Unfortunately as computation complexity of the OMR grows exponentially with the size of the network, it can not be run in real time. For instance, it takes Matlab implementation of OMR approximately 8.2 seconds to select the best path pair in a network of 49 nodes, and 237.6 seconds in a network of 100 nodes. However, as will be seen shortly, OMR can be used to provide an upper bound

on the performance of other low complexity heuristic schemes that can be run in real time. In Section 3.2, we propose one such heuristic solution, and compare its performance with OMR. Before doing so, we will first develop a technique for estimating concurrent PDP in the next section.

3.1.3 Concurrent PDP of two node-disjoint paths

In this section, we show how to compute the concurrent PDP of any given two node-disjoint paths connecting the same source and destination nodes.

We assume that we have already estimated the flow rates F_i over each link l_i. Before computing the PDP, we hypothetically include the new arriving video flow into the network by increasing the flow rate over each link in $L^1_{S,D} \cup L^2_{S,D}$ by the amount of video flow rate that will be transmitted over that link.

We define random variable

$$X_{ij} = \begin{cases} 1 & \text{packet drop in link } l_{ij} \\ 0 & \text{otherwise} \end{cases}$$

We refer to the correlation of two random variables X_{ij} and Y_{mn} as follows.

$$\rho_{xy} = \frac{Cov(X_{ij}, Y_{mn})}{\sqrt{Var(X_{ij})}\sqrt{Var(Y_{mn})}} \tag{3.4}$$

We now argue that the correlation between PDP of two node-disjoint links is small. Recall that in a wireless ad hoc network, congestion, contention, time-varying wireless channel, and mobility of nodes are four main causes of packet loss. Packet drop due to mobility of two node-disjoint links is clearly independent of each other. PDP due to contention or wireless channel error is generally small, because of 802.11 MAC layer retransmission scheme. Thus we only need to consider PDP due to congestion of two node-disjoint links. Clearly, if two node-disjoint links interfere with each other,

their PDP due to congestion are correlated. For instance, when PDP of the first link is lowered, the probability that packets sent over the second link do not access the channel is higher, increasing PDP of the second link. However in practice we expect the correlation between packet drop due to congestion for two node disjoint links to be small due to the random backoff scheme in the 802.11 MAC layer protocol. This is because the random backoff results in small correlation.

We have applied NS simulations to verify the above conjecture. We deploy 12 nodes in a 100 by 100 square meters area, with all links interfering with each other. We transmit two UDP flows of 500 kbps each over two node-disjoint links, and vary the number of cross traffic flows, whose bitrates are uniformly distributed in the range of [200, 300] kbps over other links. The cross traffic flows do not share nodes with each other. We have used trace files in NS to verify that using these parameters, the main cause of packet drop is congestion. The correlation between packet drop of two UDP flows as computed by Equation (3.4) are shown in Table 3.2. The results show that if packet drop rate over each link is small, the correlation between packet drop over two node-disjoint links is also small. In practice, one can argue that the random backoff retransmission schemes in 802.11 typically result in small packet drop rate over each link. We carried out 30 groups of simulations, and arrived at similar conclusions.

Table 3.2. Correlation of packet drop over two node-disjoint links

Num. of Cross Flows	2	3	4
Pkts Drop Rate of Flow 1	0.0421	0.1610	0.3211
Pkts Drop Rate of Flow 2	0.0364	0.1515	0.3131
Concurrent Pkts Drop Rate of Flow 1 & 2	0.0002	0.0073	0.0647
Correlation	-0.0374	-0.1336	-0.1715

Since correlation between packet drop over two node-disjoint links is small, so is the correlation between packet drop over two node-disjoint paths. This because two node disjoint paths only share two nodes, i.e. source and destination. Thus we can

compute the concurrent PDP over two node-disjoint paths $P^1_{S,D}$ and $P^2_{S,D}$ as follows:

$$\begin{aligned} P_{\text{drop}}(P^1_{S,D}; P^2_{S,D}) &\approx P_{\text{drop}}(P^1_{S,D}) \cdot P_{\text{drop}}(P^2_{S,D}) \\ &= [1 - \prod_{l_{ij} \in L^1_{S,D}} (1 - P_{\text{drop}}(l_{ij}))] \cdot [1 - \prod_{l_{mn} \in L^2_{S,D}} (1 - P_{\text{drop}}(l_{mn}))] \end{aligned} \tag{3.5}$$

In the next section, we will show how to estimate PDP over one link in order to compute the concurrent PDP of two node-disjoint paths.

3.1.4 Computation of PDP over a link

As discussed earlier, in a wireless ad hoc network, congestion, contention, time-varying wireless channel, and mobility of nodes are four main reasons for packet loss. Thus PDP over link l_{ij} can be represented as

$$P_{\text{drop}}(l_{ij}) = P_{\text{drop-cong}}(l_{ij}) + P_{\text{drop-cont}}(l_{ij}) + P_{\text{drop-chan}}(l_{ij}) + P_{\text{drop-mob}}(l_{ij}) \tag{3.6}$$

where $P_{\text{drop-cong}}(l_{ij})$, $P_{\text{drop-cont}}(l_{ij})$, $P_{\text{drop-chan}}(l_{ij})$, and $P_{\text{drop-mob}}(l_{ij})$ are PDP over link l_{ij} due to congestion, contention, wireless channel error, and mobility respectively. Our basic approach is to use existing results in the literature to estimate the last three quantities, and to develop a new approach to estimate $P_{\text{drop-cong}}(l_{ij})$.

The packet loss probability $p_{\text{loss}}(l_{ij})$ without retransmission over a link l_{ij} due to channel error or contention can be measured using the broadcast packet technique described by De Couto et al. [11]. In summary, each node periodically sends out a broadcast probe packet. Broadcast packets are not retransmitted by the 802.11 MAC layer protocol. Nodes track the number of probes received from each neighbor during a sliding time window, and include this information in their own probes to their neighbors. Each node uses this information about itself to estimate its own $p_{\text{loss}}(l_{ij})$. The combination of $P_{\text{drop-cont}}(l_{ij})$ and $P_{\text{drop-chan}}(l_{ij})$ can be represented as:

$$P_{\text{drop-cont}}(l_{ij}) + P_{\text{drop-chan}}(l_{ij}) = p_{\text{loss}}(l_{ij})^{N_{rt}} \tag{3.7}$$

where N_{rt} is the number of retransmissions before the MAC layer drops the packet, which is 4 for large packets in the 802.11 standard. Regardless, by estimating P_{loss}, it is possible to estimate $P_{\text{drop-cont}} + P_{\text{drop-chan}}$.

We estimate the PDP over a link due to mobility $P_{\text{drop-mob}}(l_{ij})$ using link availability results in [80][81]. According to Jiang et al. [81] and McDonald et al. [80], link availability $A(l_{ij}, T)$ of link l_{ij} is defined as the probability that a link is available until time $t_0 + T$, given that it is an active link at time t_0, where T is the update period. Link availability can be computed as $\frac{T_a}{T}$, where T_a is the sum of all non-continuous time periods that the link is available between t_0 and $t_0 + T$. Assuming packet rate for video communication is approximately constant, the ratio of packet drop due to mobility is approximately the ratio between the duration of the time period that the link is unavailable in T, and the total time period T. Thus

$$P_{\text{drop-mob}}(l_{ij}) \approx 1 - A(l_{ij}, T) \approx 1 - \frac{T_a}{T} \tag{3.8}$$

In the remainder of this section, we describe how to compute PDP over link l_{ij} due to congestion $P_{\text{drop-cong}}(l_{ij})$. For brevity, we refer to PDP due to congestion as PDP-congestion in the rest of this section. One possibility is to measure packet drop due to interface queue (ifq) overflow at each node, and use the ifq packet drop rate at node n_i to approximate $P_{\text{drop-cong}}(l_{ij})$. The disadvantage of this method is that it does not consider the influence of the incoming video flow on the packet drop rate of the link. In current wireless ad hoc networks, a new video flow generally consumes a large percentage of wireless resource, which can change PDP-congestion significantly. Thus we propose a new scheme to estimate a link's PDP-congestion based on the estimation of equivalent bandwidth used in the link's neighborhood.

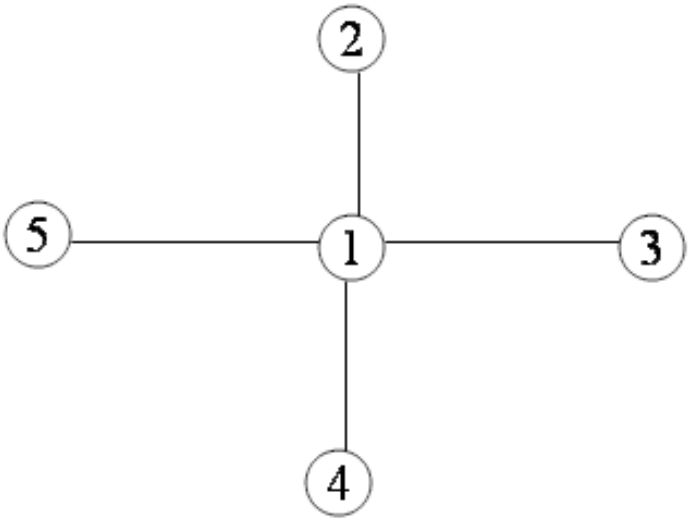
We define the *interfering link set*, consisting of all the links that interfere with link l_{ij} as follows:

$$I(l_{ij}) = \{l \in E, \text{and } l \text{ interferes with } l_{ij}\} \bigcup \{l_{ij}\}$$

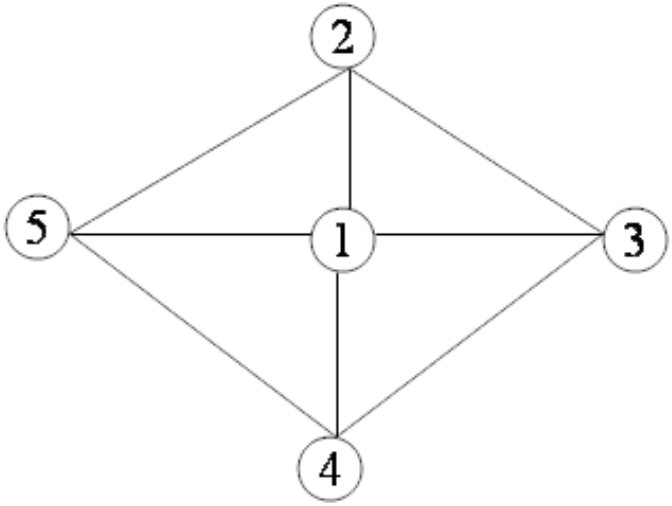
where E is the link set consisting of all the links in the graph. A naive way to compute the PDP-congestion of link l_{ij} is as follows:

$$P_{\text{drop-cong}}(l_{ij}) \approx \max(1 - \frac{C}{\sum_{l_k \in I(l_{ij})} F_k}, 0) \tag{3.9}$$

where F_k corresponds to the aggregate incoming flow rate into the k^{th} link l_k of the set $I(l_{ij})$, and C is the channel capacity. However Equation (3.9) unnecessarily overestimates the PDP-congestion [79], and as such can not effectively differentiate between congested and uncongested links. Figure 3.1 shows two conflict graphs that illustrate an example of the ineffectiveness of the naive estimation. Recall that for a conflict graph, each CG-node is a link in the original connectivity graph, and if link l_i interferes with link l_j, there is a directed CG-link connecting CG-node i and CG-node j. To simplify our explanation, we assume bi-directional CG-links in the example. Also assume that flow rates $F_j, j = 1 \ldots 5$ are equal. Intuitively, the PDP-congestion of link l_1 in Fig. 3.1(a) is smaller than that of Fig. 3.1(b), since there are extra interferences between other four links in conflict graph 3.1(b). However, Equation (3.9) would incorrectly imply PDP-congestion of link l_1 in both conflict graphs to be the same.



(a)



(b)

Figure 3.1. An example to show the ineffectiveness of the naive estimation; (a) conflict graph a; (b) conflict graph b.

We now propose a method to estimate the PDP-congestion more accurately. Note that set $I(l_{ij})$ represents both an interfering link set in the connectivity graph, which consists of all the links that interfere with link l_{ij}, and a CG-node set in the corresponding conflict graph. We partition the interfering link set $I(l_{ij})$ into several disjoint subsets, such that each subset is an independent set. In a conflict graph, an *independent set* denoted by IS, is defined to be a set of CG-nodes that have no edges between them. Intuitively, this corresponds to any set of links whose transmissions do not interfere with each other. Note that for each link l_{ij} there are multiple possible partitions. The set of independent sets resulting from a particular partition of $I(l_{ij})$ is denoted by $PT(l_{ij})$, and can be represented as follows:

$$PT(l_{ij}) = \{IS_1, IS_2, \dots IS_{q_i}\}$$

where

$$\bigcup_{k=1,\dots q_i} IS_k = I(l_{ij})$$

and

$$IS_k \bigcap IS_m = \phi,\ 1 \leq k, m \leq q_i$$

IS_k is the k^{th} independent set, and q_i is the number of subsets in this particular partition. Physically, each independent set is selected by the MAC layer protocol with some probability at each time.

Assume that we can compute all the possible partitions of set $I(l_{ij})$, calling them $PT(l_{ij})_1, PT(l_{ij})_2, \dots PT(l_{ij})_{N_i}$, where N_i is the number of partitions. We model the selection of an independent set at time t as a two level process. First, the MAC layer selects partition $PT(l_{ij})_k$ with probability p_k; second, it selects one independent set in the partition $PT(l_{ij})_k$ using the corresponding schedule for that partition, to be described shortly. The estimation of PDP-congestion of link l_{ij} can be written as

$$P_{\text{drop-cong}}(l_{ij}) \approx \sum_{k=1}^{N_i} p_k \times P_{\text{drop-cong}}(l_{ij}|PT(l_{ij})_k) \tag{3.10}$$

We define a link is an *active link*, if either of its two nodes have available packets to transmit through it. At a given time t, we define an independent set to be an *active independent set*, if at least one of links in the independent set is active. From the definition of the independent set and partition, we know that all links in the same independent set can transmit simultaneously, but not those links in different independent sets. For a given partition of the set $I(l_{ij})$ with q_i independent sets, we define the corresponding schedule as follows. Let $m_i(t)$ denote the number of active independent sets in the partition at time t. The corresponding schedule allows the r^{th} active independent set at time t to access the shared wireless medium with probability $p(r, m_i(t))$. In essence the corresponding schedule for a partition allows every active independent set to access the shared wireless medium according to some predefined schedule. All *active links* in one independent set can transmit at time t, if the independent set is assigned by the MAC layer to access the wireless medium at that time.

For example, for the conflict graph shown in Fig. 3.1(b), one possible partition of the interfering link set $I(l_1)$ is $PT(l_1) = \{\{l_1\}, \{l_2, l_4\}, \{l_3, l_5\}\}$. At time t, assuming that independent set $\{l_2, l_4\}$ accesses the wireless medium, l_2 and l_4 can transmit simultaneously, if both links have available packets to send, i.e. being active. A possible corresponding schedule for $I(l_1)$ would be to let $\{l_2, l_4\}$ access the wireless medium, followed by $\{l_3, l_5\}$, followed by $\{l_1\}$ each with probability $\frac{1}{3}$. For our purposes, the precise order in which independent sets are give access to medium is immaterial.

We can define equivalent rate of flows over all links in the independent set IS_k as follows:

$$CF_k = \max_{l_m \in IS_k} F_m \tag{3.11}$$

where F_m is the aggregate incoming flow rate over the m^{th} link l_m in the independent set IS_k. Since links of the same independent set transmit simultaneously, the equiv-

alent rate of an independent set, which is the maximum rate among all the links, is link l_{ij}'s channel resource needed by all the links in the independent set per unit of time.

Given a partition of the set $I(l_{ij})$, assuming that the corresponding schedule is applied by the MAC layer, we can estimate the PDP-congestion of link l_{ij} as follows:

$$P_{\text{drop-cong}}(l_{ij}|PT(l_{ij})) \approx \max(1 - \frac{C}{\sum_{IS_k \in PT(l_{ij})} CF_k}, 0) \tag{3.12}$$

Note that even though one independent set is chosen to transmit at any point in time, we can add the equivalent rate for different independent sets because CF_k is the equivalent rate for k^{th} independent set at any *time*, and not what k^{th} independent set transmits in the time slots in which it is granted access by the MAC layer.

Combining Equations (3.10) and (3.12), it is possible to obtain PDP-congestion of link l_{ij}. Unfortunately, computing all of the independent sets in a graph grows exponentially in the number of nodes [78]; also it is difficult to estimate the probability p_k that the 802.11 MAC layer selects partition $PT(l_{ij})_k$, $k = 1, 2, \ldots N_i$; as such, the computational overhead of the above method is too high and the implementation is impractical. To circumvent this, we consider the partition $PT(l_{ij})^*$ that minimizes $P_{\text{drop-cong}}(l_{ij}|PT(l_{ij}))$ and refer to it as *the most efficient partition*. Since

$$\begin{aligned} P_{\text{drop-cong}}(l_{ij}) &\approx \sum_{k=1}^{N_i} p_k \times P_{\text{drop-cong}}(l_{ij}|PT(l_{ij})_k) \\ &\geq \sum_{k=1}^{N_i} p_k \times P_{\text{drop-cong}}(l_{ij}|PT(l_{ij})^*) \\ &= [\sum_{k=1}^{N_i} p_k] \times P_{\text{drop-cong}}(l_{ij}|PT(l_{ij})^*) \\ &= P_{\text{drop-cong}}(l_{ij}|PT(l_{ij})^*) \end{aligned} \tag{3.13}$$

Therefore $P_{\text{drop-cong}}(l_{ij}|PT(l_{ij})^*)$ is the lower bound of link l_{ij}'s PDP-congestion. Since computing the actual PDP-congestion is prohibitively compute intensive, our

approach is to use its lower bound instead, i.e. the PDP-congestion of the *most efficient partition*, as a metric in comparing PDP of two links, and subsequently two paths. Specifically, combining Equations (3.12) and (3.13), we get

$$P_{\text{drop-cong}}(l_{ij}) \geq \max(1 - \frac{C}{\sum_{IS_k \in PT(l_{ij}^*)} CF_k}, 0) \tag{3.14}$$

where $PT(l_{ij}^*)$ denotes the most efficient partition.

As such, we will shortly propose a greedy algorithm to approximately find the most efficient partition. We note that using the most efficient partition results in underestimating the interference around link l_{ij}, and the PDP. Nevertheless, we have verified through simulations that it is sufficient to use the lower bound of each link's PDP as an approximation to our metric in order to compare and select paths. Also, it can be argued that with the development of more efficient MAC layer protocol in the future, our estimation approaches the optimal results.

3.1.5 Estimating the most efficient partition

We now propose a greedy partition algorithm to estimate the most efficient partition. The basic idea behind the greedy partitioning algorithm is to combine links with large flow rates together in order to reduce the sum of equivalent flow rates of independent sets, thus minimizing the probability $P_{\text{drop-cong}}(l_{ij}|PT(l_{ij}))$. The algorithm first selects the link with the largest flow rate into the first independent set, then selects other qualified links into the same independent set in the order of flow rate. After obtaining one independent set, the algorithm repeats the above process to obtain other independent sets, until every link in the interfering link set $I(l_{ij})$ is in one independent set.

Figure 3.2 provides an example to demonstrate the idea. There are four CG-nodes in the network. Without confusion, we use the number on each CG-node to denote

both the flow rate of the node and the node itself. The algorithm obtains the following partition: {{6,5}, {4,3}}. Another possible partition is {{6,4},{5,3}}. The combined flow rate of the first partition is $\max\{6,5\}+\max\{4,3\}=10$, while that of the second one is $\max\{6,4\}+\max\{5,3\}=11$. Obviously, the PDP-congestion is lower using the first partition than using the second one. Algorithm 1 shows the pseudocode for the proposed greedy algorithm.

Algorithm 1 Partitioning set $I(l_{ij})$

Sort CG-nodes in the set $I(l_{ij})$ based on the the flow rate, in the order from the largest to the smallest
Set $k = 0$
while ($I(l_{ij})$ not empty) **do**
  Start with an empty independent set IS_k
  Add the first CG-node I_0 into IS_k
  Update $I(l_{ij}) = I(l_{ij}) \backslash I_0$
  for (Haven't finished searching $I(l_{ij})$) **do**
    if (CG-node I_m does not have an edge to any of the CG-nodes in IS_k) **then**
      Add I_m into IS_k
      Update $I(l_{ij}) = I(l_{ij}) \backslash I_m$
    end if
  end for
  Increase k by one
end while

We use the example shown in Fig. 3.1 to explain how the proposed estimation scheme works. Assume that flow rates of $F_i, i = 1, 2, \ldots 5$ are equal. For the conflict graph shown in Fig. 3.1(a), the interfering link set $I(l_1)$ is partitioned into $\{\{l_1\}, \{l_2, l_3, l_4, l_5\}\}$, while for Fig. 3.1(b), $I(l_1)$ is partitioned into $\{\{l_1\}, \{l_2, l_4\}, \{l_3, l_5\}\}$. Both partitions are *most efficient partitions* for conflict graph

1(a) and (b) respectively, which can be verified through manually numerating all possible partitions. Using Equation (3.12) the PDP-congestion for link l_1 in Fig. 3.1(a) is given by

$$\begin{aligned} P_a &\approx \max(1 - \frac{C}{max(F_1) + max(F_2, F_3, F_4, F_5)}, 0) \\ &= \max(1 - \frac{C}{2F_1}, 0) \end{aligned} \tag{3.15}$$

where F_i is the flow rate over link l_i. Similarly, the PDP-congestion for link l_1 in Fig. 3.1(b) is given by

$$\begin{aligned} P_b &\approx \max(1 - \frac{C}{max(F_1) + max(F_2, F_3) + max(F_4, F_5)}, 0) \\ &= \max(1 - \frac{C}{3F_1}, 0) \end{aligned} \tag{3.16}$$

As expected intuitively $P_a \leq P_b$.

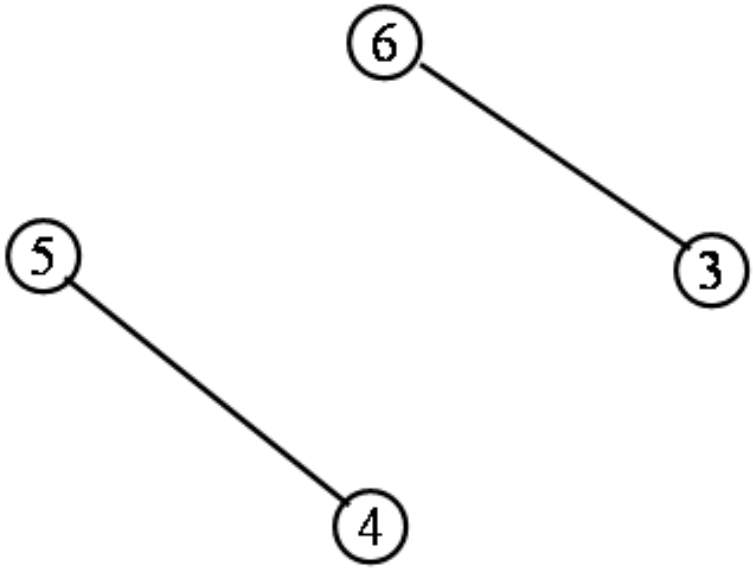


Figure 3.2. An example to show the idea of the greedy partitioning algorithm.

3.1.6 Summary and Implementation Issues

In our current implementation, we apply a centralized approach in order to estimate PDP. Each node measures flow rate and packet loss probability of broadcast packets of links to which it is connected, and broadcasts this information to the network periodically. In the end, each node receives flow rate and packet loss probability of broadcast packets of all other links.

In order to estimate flow rate of each link, the routing agent of each node parses all the outgoing packets, obtaining next hop and the packet's size. The number of bytes transmitted over a link in a time window *bytes_sent_win* can be computed through summing up the size of all packets transmitted through this link during the time window. The flow rate can be computed using a moving window average scheme, and updated as follows:

$$curr\_flow\_rate = \alpha \times prev\_flow\_rate + (1 - \alpha) \times \frac{bytes\_sent\_win}{win\_size} \tag{3.17}$$

where $curr\_flow\_rate$, $prev\_flow\_rate$, win_size represent current flow rate, previously estimated flow rate, and length of time window respectively. α is a parameter that can be used to trade off importance of past measurements versus new ones.

The procedure of estimating concurrent PDP of two paths can be summarized as follows:

- Given packet loss probability of broadcast packets, the combination of PDP over link l_{ij} due to channel error and contention is estimated using Equation (3.7).
- Given flow rate of links interfering with link l_{ij}, PDP over link l_{ij} due to congestion is estimated by first estimating the most efficient partition and then applying Equation (3.14). Our current implementation in this chapter does not take into account PDP over a link due to mobility.

- Using Equation (3.6), we compute PDP over link l_{ij} from PDP over link l_{ij} due to channel error, contention and congestion.
- Using Equation (3.5), we compute concurrent PDP of two paths from PDP over a link.

To summarize, in Section 3.1, we have developed a technique to estimate PDP of two node disjoint paths. We can use this in the next section to arrive at a practical path selection algorithm.

3.2 Interference aWare Multipath Routing (IWM)

Since the optimal multipath selection problem is NP-hard and the OMR algorithm described in Section 3.1 is prohibitively compute intensive, in this section we propose a heuristic solution, called IWM, for choosing two paths with minimum concurrent PDP. IWM applies the technique summarized in Section 3.1.6 to estimate PDP of each path. The basic idea behind IWM is to obtain a path with approximately minimum PDP as the first path. After updating all the link metrics of the network graph, such as flow rate, IWM finds the second path with approximately minimum PDP based on the new graph.

3.2.1 Centralized Implementation

We first propose a centralized protocol. We assume that flow rate and packet loss probability of broadcast packets of each link, are distributed over the whole network periodically. Thus the sender knows both the topology of the network and characteristics of each link. In this case, the sender is able to compute the PDP given any two paths in the network.

By assuming that the PDP of each link is small, we approximate $P_{\text{drop}}(P^1_{S,D}; P^2_{S,D})$ represented in Equation (3.5) as follows:

$$P_{\text{drop}}(P^1_{S,D}; P^2_{S,D}) = \sum_{l_{ij} \in L^1_{S,D}} P_{\text{drop}}(l_{ij}) \cdot \sum_{l_{mn} \in L^2_{S,D}} P_{\text{drop}}(l_{mn}) \tag{3.18}$$

Therefore, we relax the optimal multipath selection problem by allowing the first path to minimize PDP and the second path to minimize PDP among all node disjoint paths with the first one.

The optimization problem of finding the first path can be formulated as follows.

$$\underset{x_{ij}}{\text{Minimize}} \sum_{l_{ij} \in E} x_{ij} P_{\text{drop}}(l_{ij})$$

such that the constraint in Equation (3.1) is satisfied. $P_{\text{drop}}(l_{ij})$ as defined by Equation (3.6) denotes the PDP over link l_{ij}, which can be viewed as the cost assigned to link l_{ij}. $P_{\text{drop}}(l_{ij})$ is estimated through the procedure described in Section 3.1.6. We obtain the first path by solving the above OSPF-like Weighted Path Cost routing problem using the Dijkstra's algorithm.

After obtaining the first path, we first update flow rate over each link, by taking into account the incoming video flow rate into corresponding links. Given the first path, for computing the second path, we define a link cost as follows:

$$C_{mn} = P_{\text{drop}}(l_{mn}) + \text{nd\_cost} \tag{3.19}$$

where

$$\text{nd\_cost} = \begin{cases} b_1 \gg 1 & \text{destination node of link } l_{mn} \text{ in } P^1_{S,D} \\ 0 & \text{otherwise} \end{cases}$$

is a penalty factor to maintain the node-disjointness between the two paths. b_1 is chosen to be an arbitrarily large constant to trade off between disjointness and minimizing PDP.

The optimization problem to find the second path is formulated as follows:

$$\underset{y_{mn}}{\text{Minimize}} \sum_{l_{mn} \in E} y_{mn} C_{mn}$$

such that the constraints in Equation (3.2) are satisfied. We also apply the Dijkstra's algorithm to solve this optimization problem.

Both optimization problems for the first and the second paths are basically shortest path problems, which can be solved in polynomial time. Thus the complexity of IWM is comparable to other Link State routing algorithms [4][3].

The advantage of the proposed centralized approach is that it is very easy to implement. Also when a node needs to transmit video applications, it can compute two paths from the link state cache immediately, i.e. there is no start delay with this approach. However there are several disadvantages of the centralized approach.

- The node has to collect link state information of all the links in the network and store them to the link state cache. In order to build and maintain the link state cache, each node needs to periodically broadcast characteristics of its links to the whole network. Collecting link state information needs a large amount of control overhead. Two techniques can be applied to reduce the amount of control overhead. The first one is Multipoint Relay (MPR) used by Optimized Link State Routing (OLSR) [3]. The second one is partial topology information report technique applied by Topology Broadcast based on Reverse-Path Forwarding (TBRPF) [82]. In TBRPF, each node reports only a part of its source tree to all neighbors in order to reduce the size of topology updates.

- The period for updating the link state information can not be too short, otherwise the amount of control overhead becomes prohibitively large. So this approach is only suitable for static networks or networks with slow moving nodes. For the kind of network, whose condition changes very fast, the link

state information in the cache is not accurate, which will affect the selection of the best path pair.

3.2.2 Distributed Implementation

In order to reduce the amount of control overhead, we further propose a distributed protocol for IWM. The basic idea behind the distributed implementation of the IWM is that the protocol builds two paths in two steps. In the first step, the sender initiates a route discovery process by sending out a Route Query (RREQ) message. The RREQ message carries a value representing path cost of the path traversed by the message. When a node receives a non-duplicate RREQ message, before forwarding it, it updates the path cost as follows:

$$\text{new\_path\_cost} = \text{old\_path\_cost} + \text{link\_cost} \tag{3.20}$$

where the link_cost is PDP of the link connecting the previous hop and the current node, which is computed based on the link's two hop neighbors' information and Equation (3.6). The receiver collects paths carried in arrived RREQ messages within a short time period $[t_0, t_0+d]$, where t_0 is the time that the first RREQ message arrives. Then the receiver selects the path with the smallest path cost and sends a Route Reply (RRER) message carrying the path back to the sender. After receiving the RRER message carrying the first path, the sender sends out another RREQ message with a different sequence number. We use odd sequence numbers representing RREQ messages for the first path, and even ones for the second path. This time, the RREQ message carries both the path cost and the nodes' IDs of the first path. The middle nodes update a path cost carried by a RREQ message as shown in Equation (3.20), except the link_cost is represented by Equation (3.19). The sender will select the second path in a similar way.

In order to increase the probability of selecting the best path pair without increasing too much routing overhead, we further propose two enhancement techniques.

- When receiving a duplicate RREQ message, instead of simply discarding it, the middle node first compares the *path_cost* value carried in the message with the minimum path cost value stored in the node for the same route discovery process. If the *path_cost* value carried in the current message is smaller, the middle node updates the minimum stored path cost value, and forwards the newly received RREQ message.

- When a node forwards a RREQ message, the node applies an extra forwarding delay, which is proportional to the path cost carried in the message. Thus the path with smaller cost has a larger chance to arrive at the receiver. This way, the receiver has a better chance of selecting the ideal best path.

In order to learn two hop neighbors' information, each node sends beacon messages to its neighbors periodically. The beacon message carries both characteristics of links connected to the current node, and information of links connected to the current node's neighbors. Thus each node can learn its two hop neighbors' links' characteristics through exchanging beacon messages.

3.3 Simulation Results

In this section, we present simulation results to demonstrate the efficacy of the proposed multi-path selection scheme for a streaming application.

3.3.1 Simulation Setup

We use a simulation model based on NS-2 [76]. The simulation model was briefly introduced in Chapter 2. Note that the wireless channel capacity can not be fully utilized due to the inefficiencies in the distributed nature of the 802.11 MAC protocol [84]. In [84], it is shown that the throughput of the 802.11 MAC protocol depends on the number of transmitting stations in the network, the size of the backoff window, and the packet size. We perform a simple simulation in NS, varying the number of transmitting stations from 1 to 25. All the links interfere with each other, and the packet size is set to be 512 bytes. We have observed throughput of approximatly 1.24 Mbps even as the number of transmitting stations varies. In order to avoid serious congestion, we scale the throughput by a factor of 0.8, i.e. to 1.0 Mbps, to use as the channel capacity, C, in our PDP estimation model. We assume that each node knows the flow rate of other links. In our simulations, we mostly study the case of static wireless ad hoc networks with stationary nodes, and assume PDP due to contention and channel error is very small after retransmissions. Thus in this section, the only contribution to PDP is assumed to be congestion.

We randomly choose one video sender and one video receiver. Standard MPEG QCIF sequence Foreman is MDC coded with MP-MDVC [65]. We encode each frame into two descriptions, and the group-of-pictures (GOP) size is chosen to be 15. Intra-frame encoding is identical for both descriptions. For each description, an I-frame is packetized into two packets, and two consecutive P-frames are packetized into one packet, in order to make each packet smaller than Maximum Transmission Unit (MTU) of ethernet, and to reduce the number of total packets. For the same visual quality, as measured by Peak Signal to Noise Ratio (PSNR), the bit rate needed for MDC is around 30% - 60% larger than that for Single Description Coding (SDC). This is due to inevitable compression inefficiency of MDC as compared to SDC [65][67].

We evaluate the performance of video streaming applications using the following metrics:

a. **The ratio of bad frames:** The ratio of bad frames is the ratio of the number of bad frames to the total number of video frames that should have been decoded in the receiver. Note that the ratio of bad frames is different from packet delivery ratio or the number of packet loss bursts.

b. **The number of bad periods:** A bad period consists of contiguous bad frames. This metric reflects the number of times that received video is interrupted by the bad frames.

3.3.2 Verification of the Proposed Multipath Selection Model

In our proposed multipath selection model, given two paths $P^1_{S,D}$ and $P^2_{S,D}$ connecting the sender N_S and the receiver N_D, applying the procedure summarized in Section 3.1.6, we can compute the concurrent PDP of these two paths as the metric to select the best path pair among different path pairs in the proposed multipath routing protocol. In this section, we verify that concurrent PDP could be a reasonable indicator for streaming applications' performance. We do this by comparing the results of NS simulations for ratio of bad frames and that of the estimation model based on concurrent PDP. Note that the concurrent PDP and the ratio of bad frames are highly correlated, but not the same. Specifically the term bad frame takes into account the error propagation property of current video coding schemes and dependency across the frames within a GOP. Intuitively, we would expect these two terms to be related, and that the lower concurrent PDP of two paths, the lower ratio of bad frames observed at the receiver side.

We consider a grid network consisting of 49 nodes, placed in a 7×7 grid. The distance between neighboring nodes is 200 meters, slightly shorter than the communication range. We randomly choose a video sender and receiver. The shortest path between the sender and the receiver is five hops. The bitrate of the MDC video flow is 121.7 kbps. We insert 20 one-hop cross traffic flows, whose bit rates are uniformly selected in the range of [0,200.0] kbps, and packet size is 512 bytes.

We manually select six paths connecting the sender and the receiver, and consider 21 transmission scenarios as follows: 15 path pairs with all possible combinations of every two paths, plus 6 single paths. We transmit a different description of the video flow over each path in a path pair case, and both descriptions over one path in a single path case.

We obtain ratio of bad frames for different transmission scenarios through packet level NS simulations. Each simulation lasts 3000 seconds in order to obtain statistically reliable results. We also compute concurrent PDP for each transmission scenario through the estimation model summarized in Section 3.1.6. We then order each transmission scenario based on bad frame ratio for NS simulation results and concurrent PDP for the estimation results, and show the rank of each transmission scenario in Figure 3.3. As seen, the results of the estimation model match those obtained by the NS simulation quite well. Based on the estimation model, the 3 best transmission scenarios are $5, 16$, and 17, which also happen to be best performing transmission scenarios according to NS simulations. This means that if we select the optimal transmission scenario based on the concurrent PDP estimation model of Section 3.1.6, we are likely to have chosen the best performing transmission scenario in terms of the ratio of bad frames. We have also tested our PDP estimation model with other networks, whose nodes are placed randomly, and have reached similar conclusions.

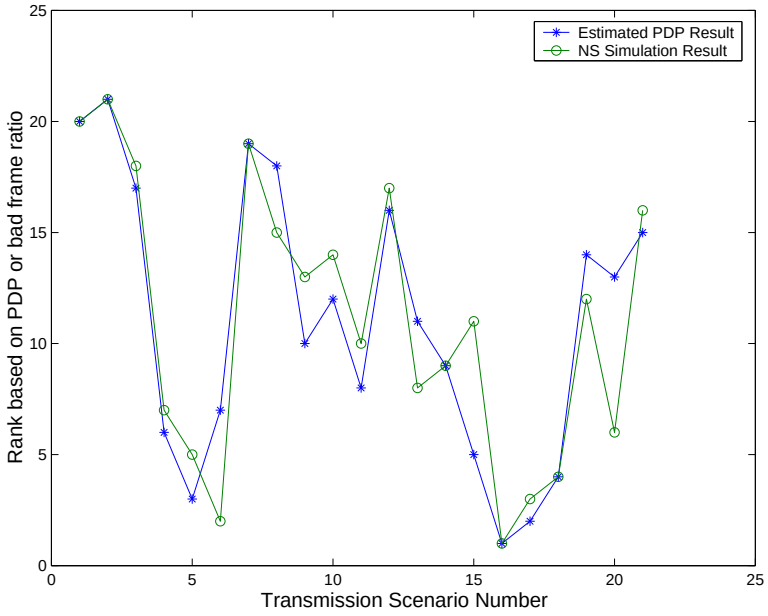


Figure 3.3. Verification of the PDP model

3.3.3 Performance of the Centralized IWM

In this section, we use NS simulations to compare OMR, Centralized IWM, the node-disjoint multipath routing (NDM), and the shortest widest path routing (SWP) [85]. For OMR and IWM we use the PDP estimation method summarized in Section 3.1.6, where flow rates are assumed to be perfectly known at every node. For the NDM routing, we first find the shortest path p_1, and then update the links' cost as follows:

$$D_{mn} = \begin{cases} \infty & m \in p_1 \text{ or } n \in p_1 \\ 1 & otherwise \end{cases} \tag{3.21}$$

We then use the updated links' cost to calculate the new shortest path p_2.

We have tested these four protocols in the 7×7 grid network described in the previous section. The bit rates of cross flows are changed every 30 seconds. All other settings are identical to those of simulations in Section 3.3.2.

We run 30 simulations for different network topologies and select different senders

and receivers in each scenario. Each simulation lasts 900 seconds. Figures 3.4(a) and 3.4(b) show the ratio of bad frames and the number of bad periods of all 30 runs of the four schemes respectively. As seen, the average performance of IWM is very close to that of OMR, and is significantly better than that of NDM and SWP, even though its computational complexity is as low as NDM and SWP. Specifically, as shown in Table 3.3, IWM has the lowest ratio of bad frames among all protocols in 26 out of 30 runs. Figure 3.5 shows the length of the achievable shortest path between the sender and the receiver in all 30 scenarios. IWM is particularly effective when the distance between the sender and the receiver is large, e.g. run #1, #3, #8 . In this case, IWM distributes the video traffic between two paths which are far from each other. This has two advantages. First, packet drop over two paths far from each other are independent. Second, the aggregate bandwidth of two paths far from each other is larger. Thus IWM outperforms NDM and SWP in this case. On the other hand, when the sender and the receiver are close to each other, e.g. run #15, #18, #19, the gain brought by longer detoured paths are offset by the extra resource consumed. In this case, IWM will select two paths close to each other or even a single path, resulting in similar performance to NDM and SWP. Both the simulation results and the analysis show that the relaxation of the optimal multipath selection problem used by IWM is very efficient.

We have also run simulations over a random wireless network consisting of 100 nodes, distributed in a 1250 by 1250 meters square area. The bitrate of the MDC video flow is 121.7 kbps. There are 25 one-hop cross traffic flows, whose bit rates are uniformly selected in the range of [0,100.0] kbps. The results are shown in Tables 3.5 and 3.6, Figures 3.6(a) and (b), and similar conclusions are reached.

Table 3.3. Summary for the Grid Network: the ratio of bad frames

	OMR	IWM	NDM	SWP
Average	0.0655	0.0685	0.1864	0.1755
Num. of Best	29	26	7	8

Table 3.4. Summary for the Grid Network: the number of bad periods

	OMR	IWM	NDM	SWP
Average	74.7	79.1	186.0	153.3
Num. of Best	20	17	2	7

3.3.4 Performance of the Distributed IWM

In this section, we compare Centralized IWM, Distributed IWM, NDM and SWP. We test these four protocols in the 7×7 grid network described in Section 3.3.3. The bit rates of cross flows are changed every 100 seconds. All other settings are identical to those of simulations in Section 3.3.3. We run 30 simulations for different network topologies and select different senders and receivers in each scenario. Each simulation lasts 1500 seconds. There are 20 cross traffic flows in the network, whose bitrates are selected uniformly between 0 and 180 kbps.

Figures 3.7(a) and 3.7(b) show the ratio of bad frames and the number of bad periods of all 30 runs of the four schemes respectively. Simulation results show that the average performance of Distributed IWM is better that of NDM and SWP, but is worse than Centralized IWM. As shown in Table 3.7, the average ratio of bad frames of Centralized IWM, Distributed IWM, NDM and SWP are 0.0334, 0.0684, 0.1041, 0.1009 respectively. Distributed IWM has the lowest ratio of bad frames among all protocols in 18 out of 30 runs, and has the lowest number of bad periods in 13 out of 30 runs. In summary, the performance of Distributed IWM lies between Centralized

Table 3.5. Summary for the random network: the ratio of bad frames

	OMR	IWM	NDM	SWP
Average	0.0223	0.0241	0.0445	0.0562
Num. of Best	29	27	19	18

Table 3.6. Summary for the random network: the number of bad periods

	OMR	IWM	NDM	SWP
Average	28.2333	29.9667	52.4000	57.1667
Num. of Best	21	17	13	13

IWM and NDM or SWP. There are mainly two reasons for the performance gap between Distributed IWM and Centralized IWM. First, since Distributed IWM only uses local knowledge of each node, the path cost computed by the protocol is not as accurate as that by Centralized IWM. A node does not learn the accurate topology in its two hop neighborhood, and thus does not model the interference around it as accurately as Centralized IWM does. Second, during the process of flooding RREQ messages to the network, and sending RRER messages back to the sender, some useful RREQ messages might be dropped or lost in the middle of the network, thus the sender does not obtain the best path all the time.

Table 3.7. Performance of the Distributed IWM in a Grid Network: the ratio of bad frames

	Centralized IWM	Distributed IWM	NDM	SWP
Average	0.0334	0.0684	0.1041	0.1009
Num. of Best	28	18	13	10

Table 3.8. Performance of the Distributed IWM in a Grid Network: the number of bad periods

	Centralized IWM	Distributed IWM	NDM	SWP
Average	74.8000	109.5333	189.9333	157.0333
Num. of Best	23	13	10	9

3.4 Testbed Implementation and Evaluation

To demonstrate the feasibility of the proposed multipath selection framework and the IWM, we have built a small wireless ad hoc network testbed, consisting of desktops

and laptops. In this section, we summarize the key components of the testbed, and report the results obtained from the performance study conducted on it.

3.4.1 Software Architecture

Figure 3.8 shows the entire software architecture of a node. We have implemented the proposed IWM protocol in the Mesh Connectivity Layer (MCL), which is an ad hoc routing framework provided by Microsoft Research [13]. MCL implements a virtual network adapter, i.e. an interposition layer between layer 2 (the link layer) and layer 3 (the network layer). The original MCL maintains a link cache in each node to store loss rate and bandwidth information of each link. Also the original MCL implements a routing protocol named Link Quality Source Routing (LQSR) to route packets. The LQSR supports different link-quality metrics, e.g. Weighted Cumulative Expected Transmission Time (WCETT) and Expected Transmission Count (ETX) [13].

We have made two major modifications to MCL. First we implement IWM inside the MCL framework such that it coexists with the LQSR in MCL. When forwarding a packet, the MCL uses one bit of information transmitted from the upper layer to decide which routing protocol to use. If the packet is high priority video packet, MCL uses IWM to route it, otherwise, it still uses LQSR. This way, we can run IWM and LQSR simultaneously in the network, and compare them under same network conditions. In our experiments, IWM is used to route MDC packets and LQSR is used to route SDC packets[1]. The second modification we have made is to enable the estimation of flow rate of each link in order to compute the PDP using the scheme described in Section 3.1.6. We set α in Equation (3.17) to be 0.1.

[1]Recall that SDC rate is about 30% - 60% lower than that of MDC video due to compression inefficiency of MDC.

We have also implemented both MDC and SDC streaming protocol in the application layer. In the streaming protocol, we have implemented timestamping, sequence numbering, feedback functions and the rate control scheme to be described in the next section. UDP sockets are used at the transport layer. The deadline of each frame is 2 seconds after the transmission time. If a packet is received after its deadline, it is discarded.

3.4.2 A Simple Rate Control Scheme

In our multipath selection framework, we assume that there exists a rate control scheme to determine the video application's sending rate. This way the sending rate can be adjusted according to the amount of congestion in the network.

The basic idea behind our rate control scheme is to employ an Additive Increase Multiplicative Decrease (AIMD) algorithm, which is the default congestion control mechanism used in TCP today. The receiver transmits a feedback packet to the sender periodically, in order to inform the sender whether the network is congested. Since PDP due to contention and wireless channel error is generally small, due to 802.11 MAC layer retransmissions, the scheme uses lost packets as a signal to detect congestion. The receiver detects lost packets using sequence numbers carried by each packet. In order to alleviate the out-of-order problem caused by multipath transmission, the receiver counts packets received from each path separately.

At the sender side, after receiving the feedback packet, if the network is not congested, the sender increases the video transmission rate by 1 fps in each time period. If the network is congested, the sender decreases the video frame rate immediately by half. If the sender has not received one feedback packet in a time interval twice of the feedback period, this triggers a timeout event, and the sender reduces the video transmission rate to the minimum transmission rate.

For simplicity, we change the transmission bit rate through changing the number of transmitted video frames per unit time without even dropping a frame. This has the effect of changing the playback duration of a given chip at the receiver. Our motivation for doing so is purely ease of implementation. This way, we do not have to implement fine grain or temporal scalability in order to compute our metrics, such as ratio of bad frames or bad periods. For a fixed GOP, this method results in the same metrics as modifying the encoding and decoding rate on-the-fly, i.e. applying temporal scalability. For example, assuming GOP of 15, if frame #4 is non-decodable, the number of bad frames for both methods is 12.

3.4.3 Testbed Setup

We deploy an 11-node wireless ad hoc network testbed on the third floor of Cory Hall, the office building of EECS, University of California at Berkeley. The nodes are placed in offices and in the aisles, which are separated from each other with floor-to-ceiling walls and solid wood doors. Figure 3.9 shows the deployment of our testbed.

Each node in the testbed is either a standard desktop or laptop running Windows XP. Each desktop is equipped with either a Linksys 802.11 a/b/g PCI card or a Netgear 802.11 a/b/g PCI card. Similarly, each laptop is equipped with either a Linksys 802.11 a/b/g PCMCIA card or a Netgear 802.11 a/b/g PCMCIA card. All cards operate in the ad hoc mode.

All of our experiments were conducted over IPv4 using statically assigned addresses. Except for configuring ad hoc mode and fixing the frequency band and channel number, we use the default configuration for the radios. The cards all perform autorate selection.

3.4.4 802.11g wireless ad hoc network result: static nodes

We first performed a series of tests to show the performance of our proposed Centralized IWM in 802.11g wireless ad hoc network. We carried out eight 300 second long experiments. Only nodes 1 to 7, 9 and 11 are activated in this scenario. Nodes 1 and 2 are MDC and SDC video senders respectively, and nodes 5 and 6 are MDC and SDC video receivers separately. In ad hoc mode, both Netgear and Linksys cards' maximum throughput is only 11 Mbps.

We compare our proposed IWM and MDC with LQSR using metric WCETT and SDC. Metric WCETT has been shown to be more effective than other path selection metrics, e.g. ETX and shortest path, for single path routing[13].

Figures 3.10(a) and 3.10(b) show the result of the ratio of bad frames and the number of bad periods for all eight runs. As seen, performance of IWM/MDC is significantly better than that of LQSR/SDC in all eight runs. During the experiment, we observed that the throughput of each link change drastically due to the change of channel quality resulting in packet drops. Since our proposed scheme transmits MDC through two node-disjoint paths, packets of different descriptions are usually not dropped simultaneously. Thus both the ratio of bad frames and the number of bad periods are reduced.

Figure 3.11 shows the result of PSNR of the received video for all eight runs. In seven out of eight runs, IWM outperforms LQSR by several dBs, and on average, IWM outperforms LQSR by 2.8 dB. We plot PSNR and loss traces of run1 using IWM/MDC and LQSR/SDC in Figures 3.12 and 3.13 respectively. For IWM/MDC, it can be seen in Figure 3.12(a) that PSNR drops gracefully, when there is packet loss only in one substream. As seen in Figure 3.12(b), most of the time, packet losses of two substreams do not overlap, thus reducing both the number and the amount of PSNR drops. The PSNR curve of LQSR/SDC shown in Figure 3.13(a) has more

frequent and severe drops than that of IWM/MDC; this is because PSNR drops for every packet drop in SDC video, and would drop severely when there is a burst of packet loss.

3.4.5 802.11a wireless ad hoc network result: static nodes

We have performed a series of tests in 802.11a wireless ad hoc networks. The maximum throughput of each link is 54 Mbps, which is much larger than that of a 802.11g wireless ad hoc network. We do not expect the results of a 802.11a network to be different from a 802.11g network, except we need to create large cross traffic flows to make the network congested. We have carried out ten 360 second long experiments with varying cross traffic level. The senders and receivers are the same as those of the previous experiments. In runs 1 through 8, there are two one hop UDP cross traffic, whose bit rate is changed every 30 seconds based on uniform distribution. In runs 9 and 10, the cross traffic is one two-hop TCP connection.

Figures 3.14(a) and 3.14(b) show the result of the ratio of bad frames and the number of bad periods of all ten runs. The horizontal axis shows the average bit rate of combined cross traffic. As seen, IWM/MDC significantly outperforms LQSR/SDC in nine out of ten runs, and the performance gap with IWM/MDC and LQSR/SDC increases as cross traffic increases. Once again, this shows the advantage of path diversity with MDC over single path transmission of SDC video.

Figure 3.15 compares PSNR of two schemes for all ten runs. On average, IWM/MDC outperforms LQSR/SDC by 1.1 dB, and in eight out of ten runs. The reason for slightly worse performance in runs 2 and 3 is low packet loss rate for both schemes in these runs. As a result, the PSNR of received video in these runs are close to the PSNRs of original MDC and SDC videos respectively. The PSNR of encoded MDC is slightly lower than that of encoded SDC, because in practice it is very hard

to make two video flows achieve the exact same PSNRs. In general, we would expect performance gain of IWM/MDC over LQSR/SDC to become wider as packet drop probability increases, which is also in agreement with the results in Figure 3.14.

We plot PSNR, loss traces and frame rate traces of run 7, i.e. the first run with cross traffic 8000 kbps, using IWM/MDC and LQSR/SDC in Figures 3.16 and 3.17 respectively. IWM/MDC outperforms LQSR/SDC by 1.1 dB in run 7. As seen in Figure 3.16(a), for IWM/MDC, PSNR drops gracefully, when there is packet loss in only one substream. As seen in Figure 3.16(b), most of the time, packet losses of two substreams do not overlap, thus reducing both the number and the amount of PSNR drops. The PSNR curve of LQSR/SDC shown in Figure 3.17(a) has more frequent and severe drops than that of IWM/MDC; this is because PSNR drops for every packet drop in SDC video, and would drop severely when there is a burst of packet loss. As seen in Figure 3.16(e), our simple rate/frame control scheme adjusts the video rate promptly, whenever there is packet drop in any path, and keeps the maximum sending rate, whenever there is no packet drop.

3.4.6 802.11a wireless ad hoc network result: moving nodes

We also carried out experiments with one moving node in 802.11a wireless ad hoc networks. In these experiments, we do not take into account PDP due to mobility even though the nodes are slowly moving. During the experiment, we randomly select one laptop, move it to a random position, and repeat the process. The senders and receivers are the same as those of previous experiments. At any time, there is always one laptop moving. Figures 3.18 and 3.19 show the results of three 600 seconds experimental run.

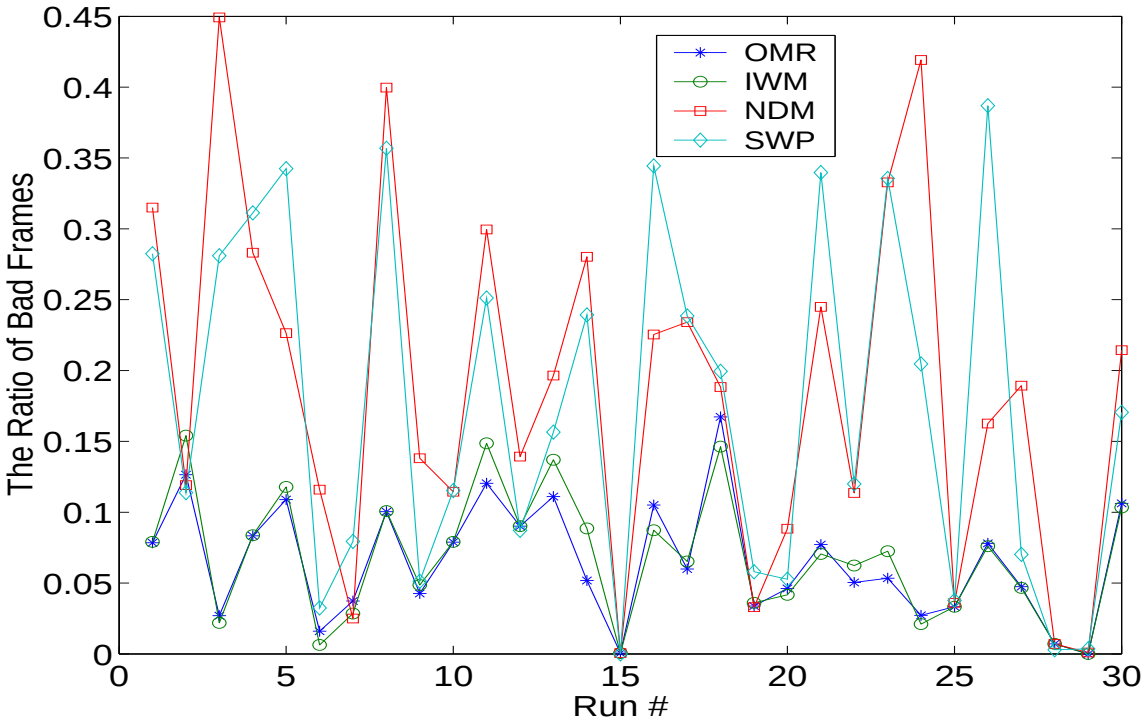
As seen in Figure 3.18, the ratio of bad frames and the number of bad periods are both greatly reduced for IWM/MDC in all three runs. With the continuous movement

of one node, one path is broken from time to time. If the path selected by LQSR is broken during the video transmission, the SDC receiver suffers from packet loss and interruption of video playback. In contrast, even if one path selected by IWM is broken, the received video quality is still acceptable. Figure 3.19 compares PSNR of two schemes. Averaged over three runs, IWM outperforms LQSR by 2.1 dBs.

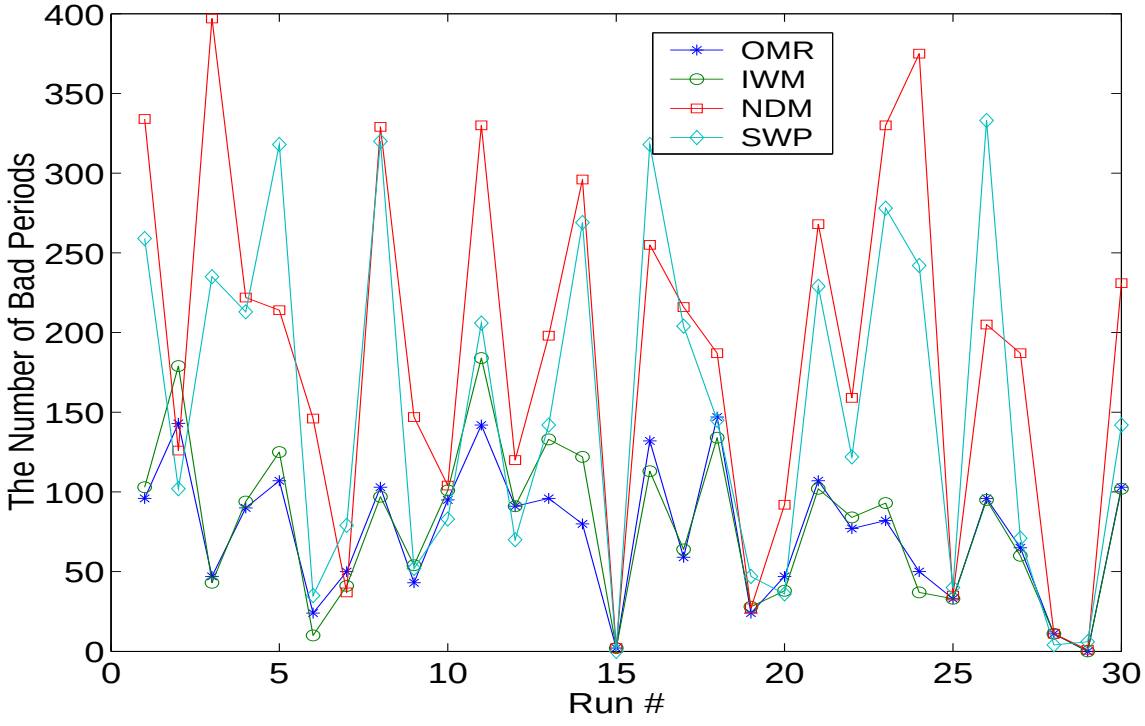
We plot PSNR and loss traces of run2 using IWM/MDC and LQSR/SDC in Figures 3.20 and 3.21 respectively. For IWM/MDC, it can be seen in Figure 3.20(a) that PSNR drops gracefully, when there is packet loss only in one substream. Shown in Figure 3.20(b), packet losses of two substreams do not overlap most of the time, because two paths do not fail simultaneously in general. The PSNR curve of LQSR/SDC shown in Figure 3.21(a) has much more frequent and severe drops than that of IWM/MDC.

3.5 Conclusions

In this chapter, we propose a novel multipath streaming framework in order to provide robustness in video communication applications over wireless ad hoc networks. We have proposed a model to estimate the concurrent PDP of two paths by taking into account the interference between different links, and formulate an optimization problem in order to select two paths with minimum concurrent PDP, which optimizes the worst case MDC video quality over all times. Then we propose a heuristic IWM routing protocol based on our path selection model, whose performance is shown to be close to that of the "optimal routing", and significantly better than that of existing schemes, through both NS simulations and actual experiments in a testbed.



(a)



(b)

Figure 3.4. Simulation Results comparing OMR, IWM, NDM, SWP on the 7×7 Grid Network for 30 runs: (a) The ratio of bad frames; (b)The number of bad periods.

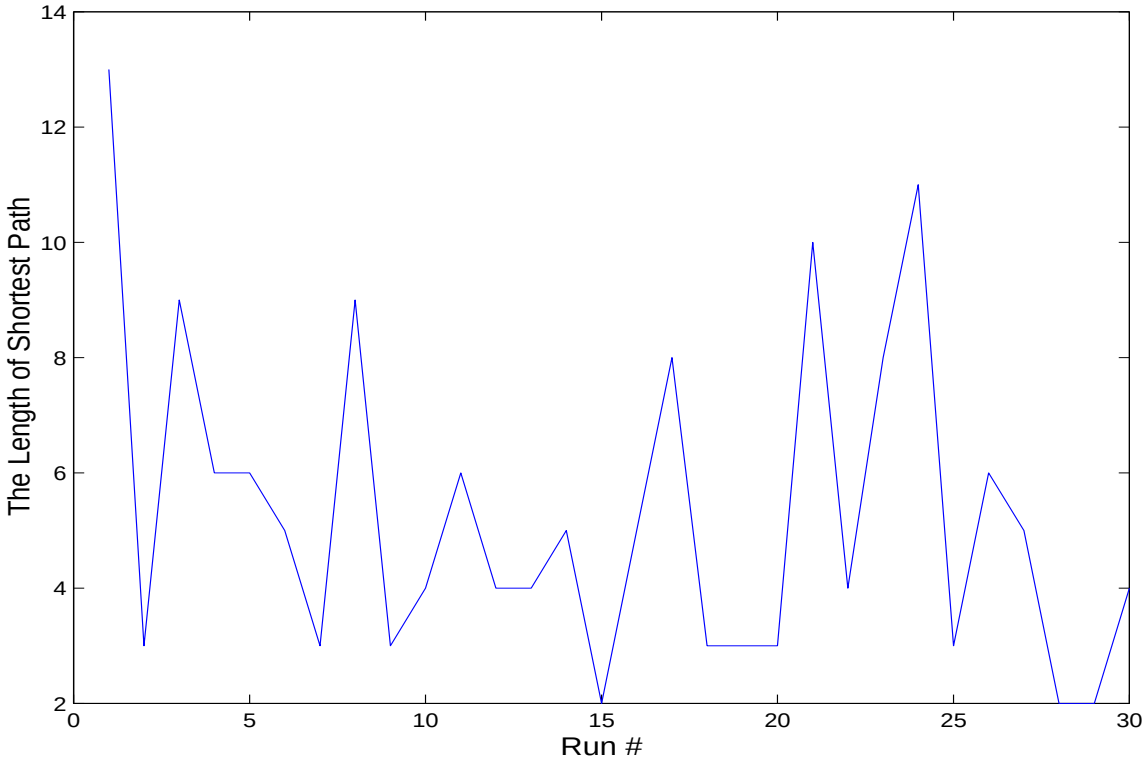
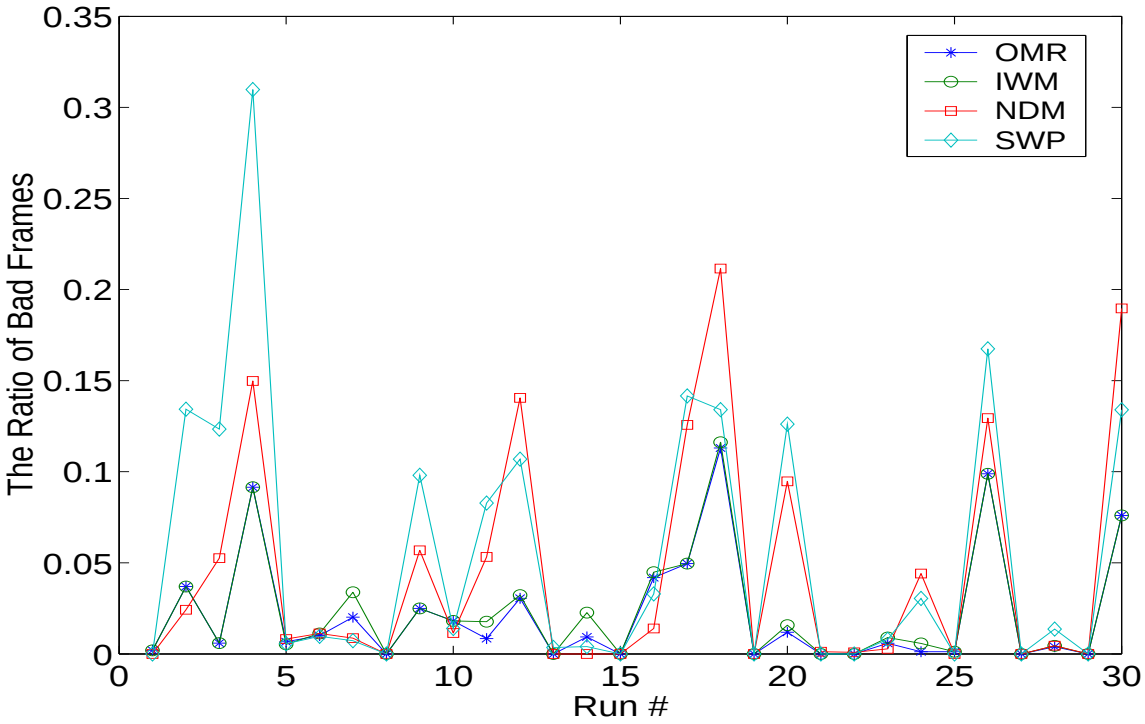
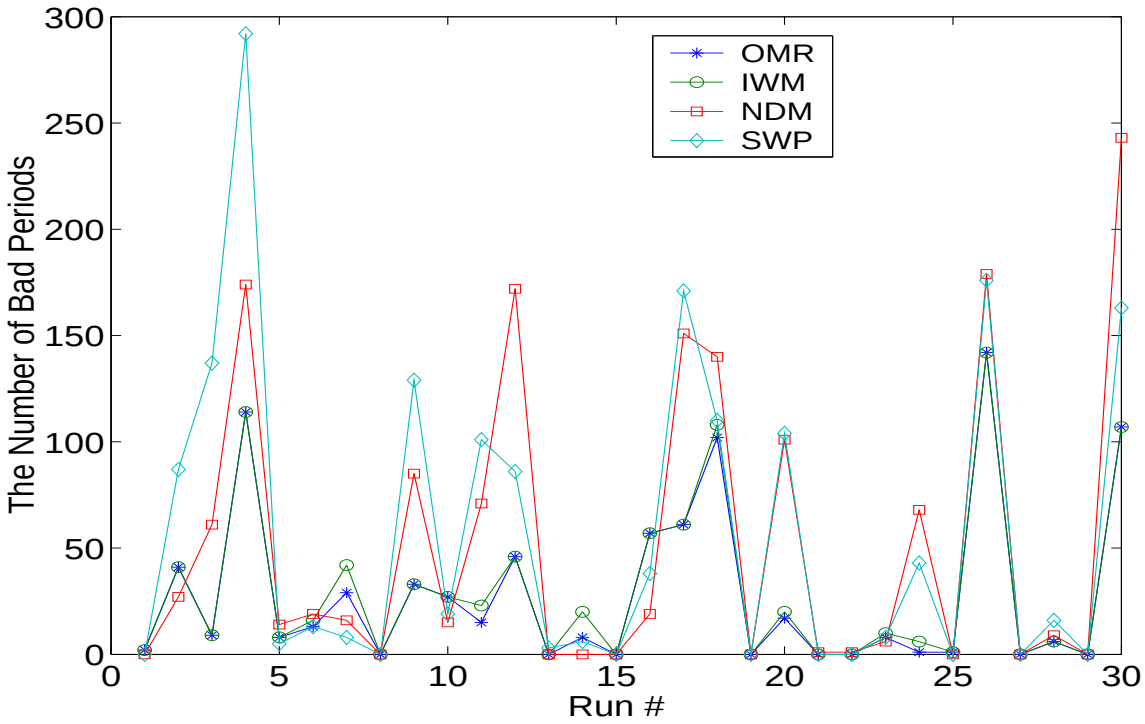


Figure 3.5. Length of the achievable Shortest Path for the 7×7 Grid Network

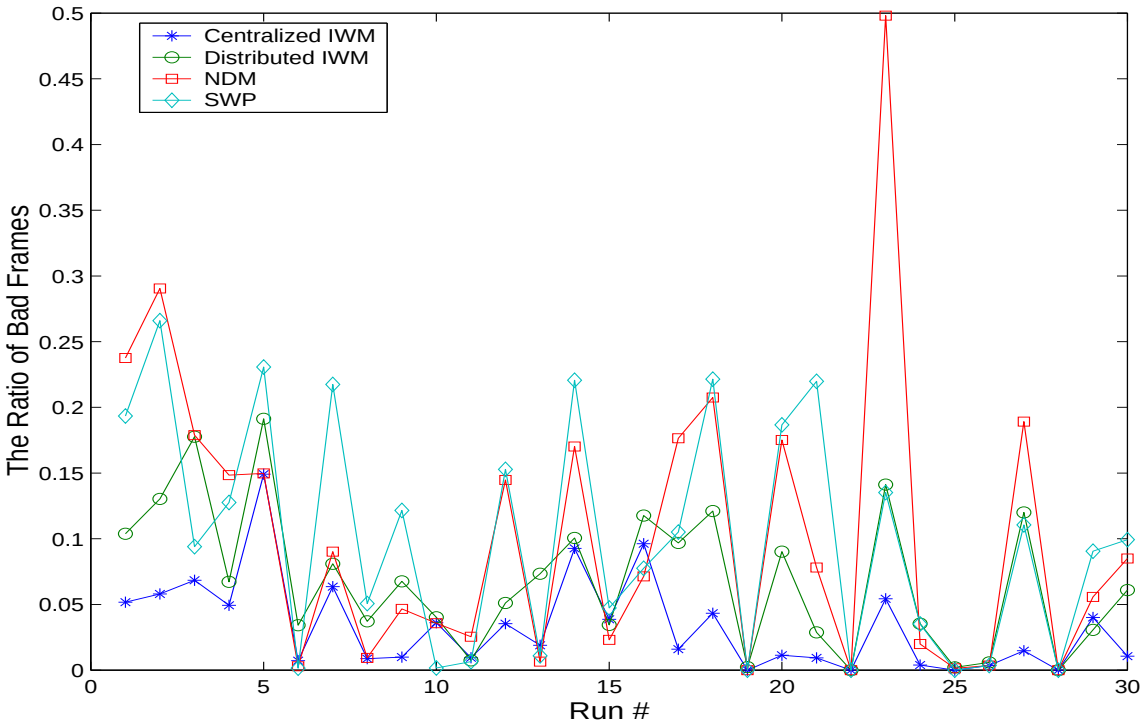


(a)

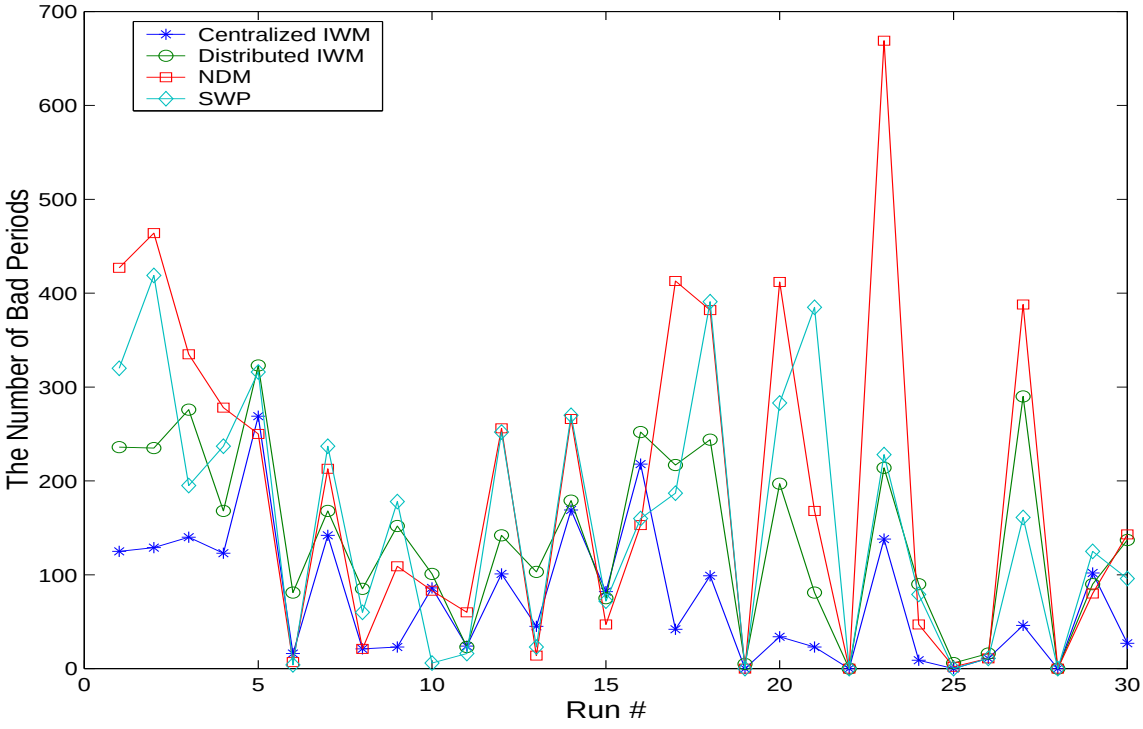


(b)

Figure 3.6. Simulation Results for the Random Network comparing OMR, IWM, NDM, SWP for 30 runs: (a) The ratio of bad frames; (b)The number of bad periods.



(a)



(b)

Figure 3.7. Performance of the Centralized IWM, Distributed IWM, NDM, SWP in a Grid Network for 30 runs: (a) The ratio of bad frames; (b)The number of bad periods.

Streaming Protocol with Rate Control (Application Layer)
IP Layer
Mesh Connectivity Layer (with LQSR and IWM)
Ethernet Layer (802.11, 802.16 ...)

Figure 3.8. Software Architecture

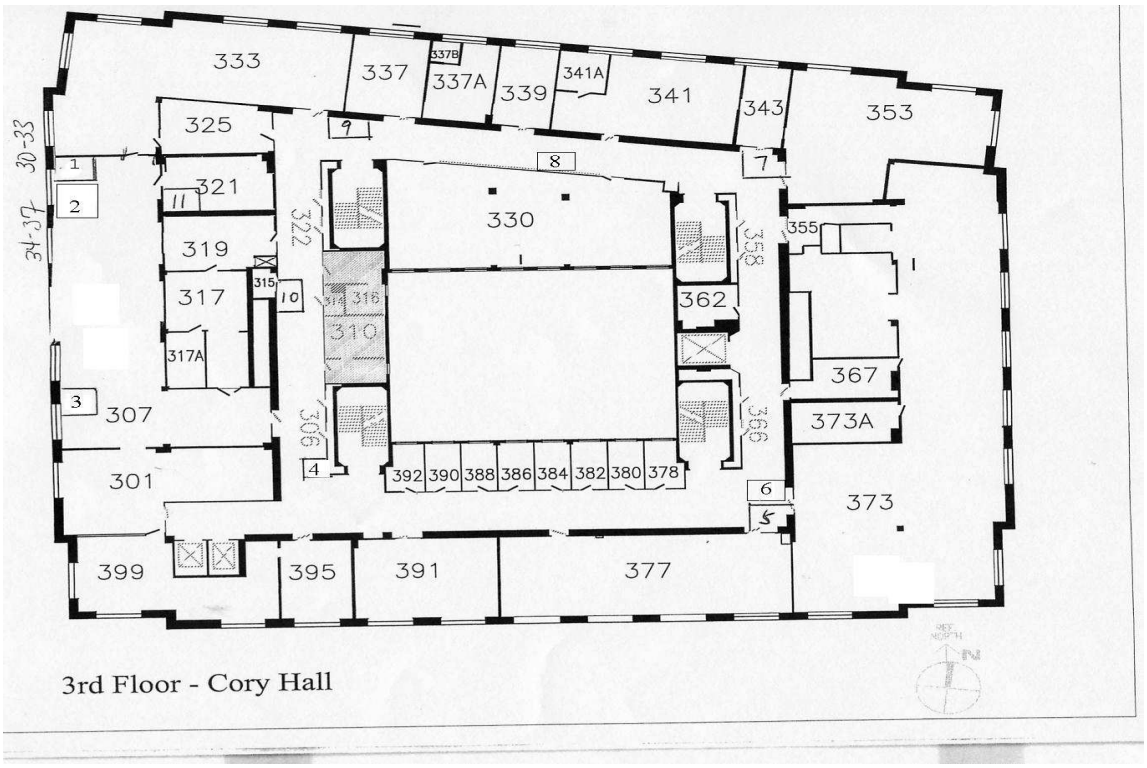
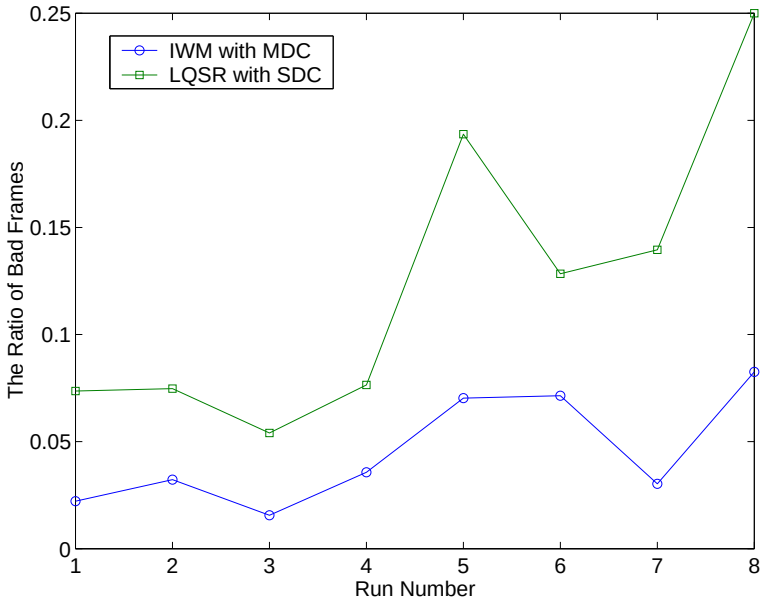
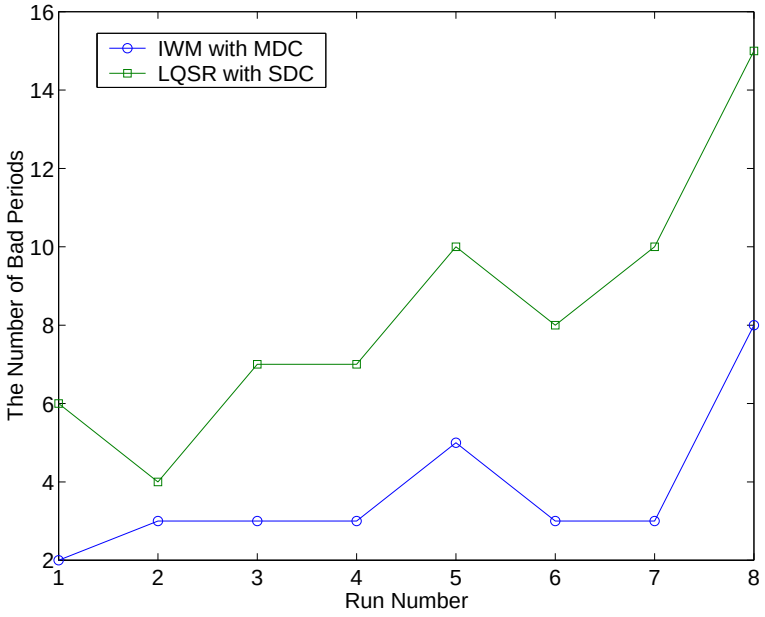


Figure 3.9. Nodes Deployment of the Testbed



(a)



(b)

Figure 3.10. Performance evaluation for 802.11g with static nodes (a) The ratio of bad frames; (b)The number of bad periods.

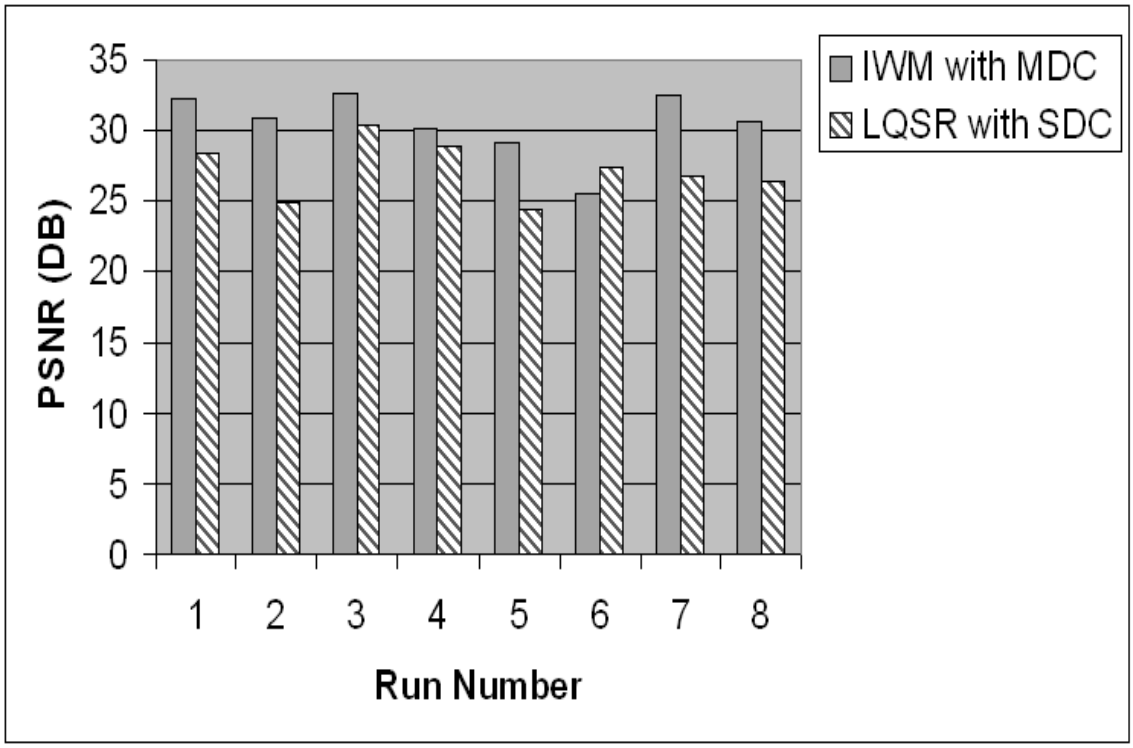
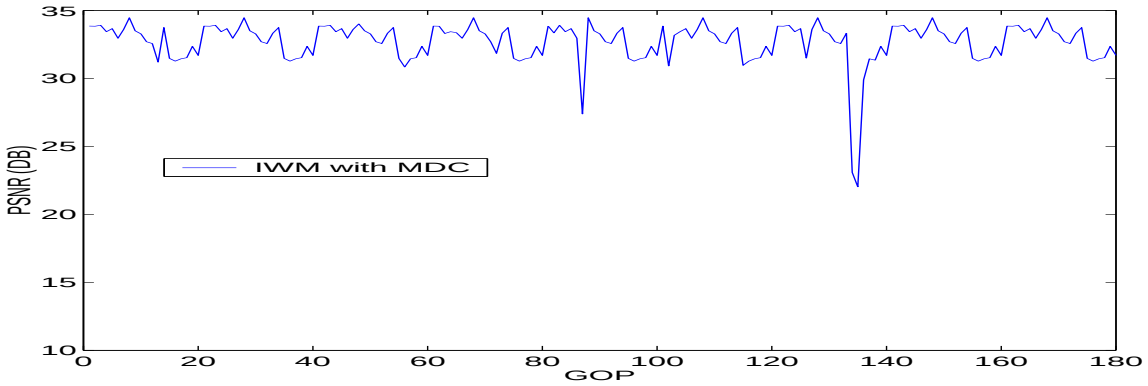
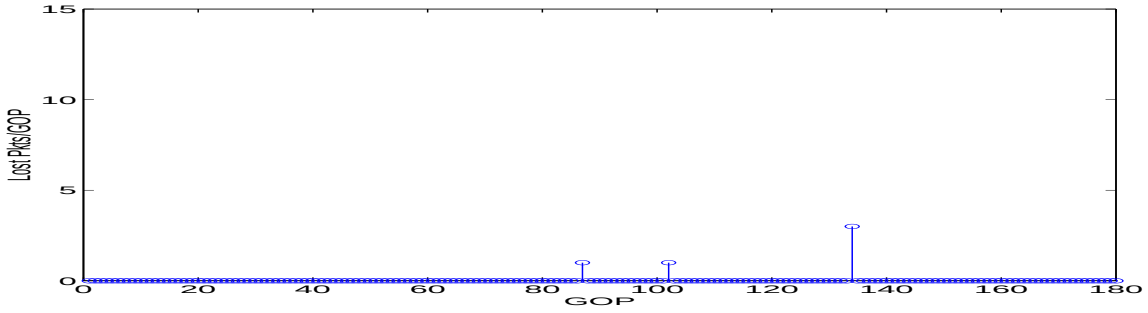


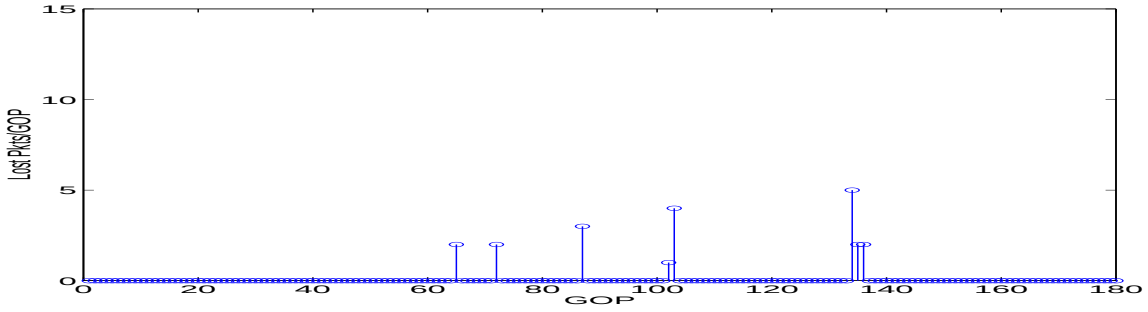
Figure 3.11. PSNR performance evaluation for 802.11g with static nodes



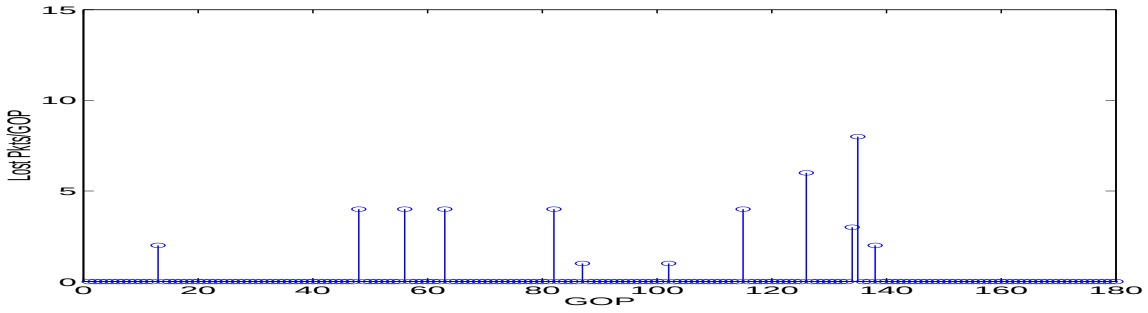
(a)



(b)

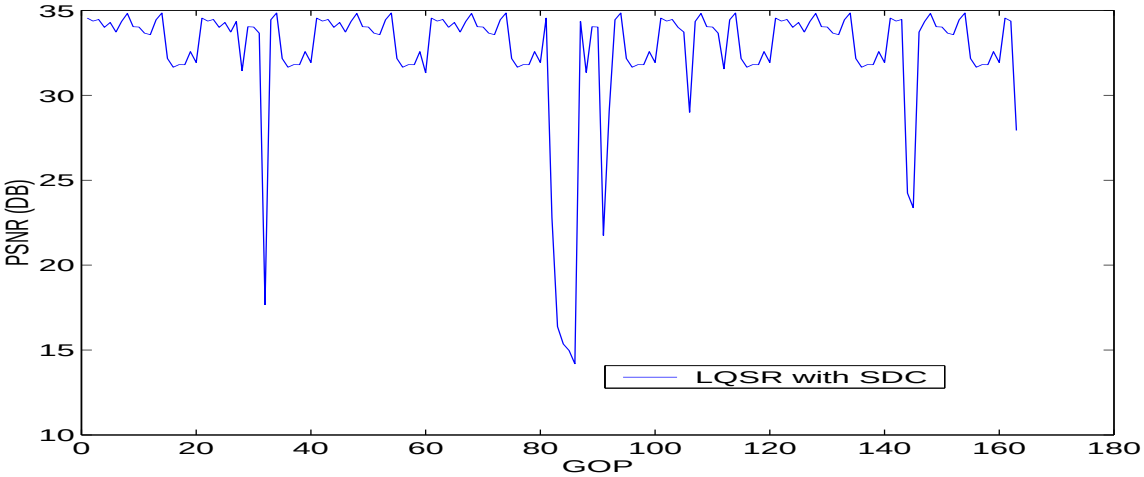


(c)

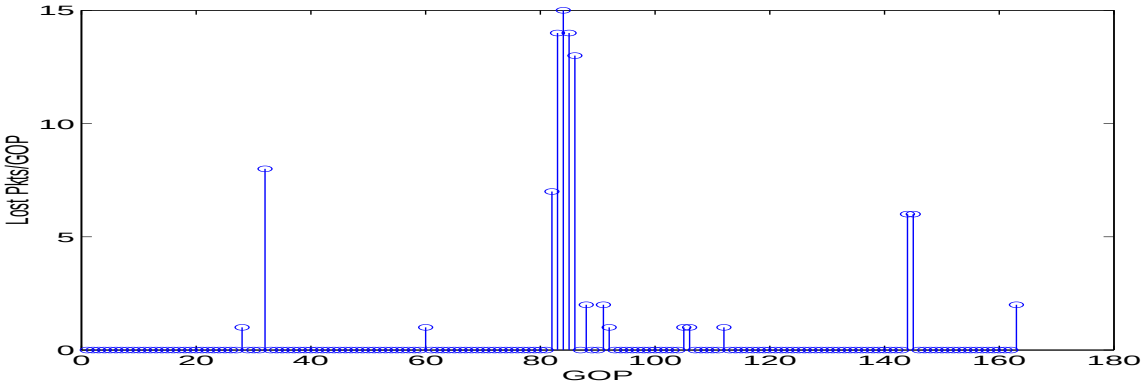


(d)

Figure 3.12. Performance evaluation for 802.11g with static nodes (a) PSNR of the received frames using IWM and MDC; (b) Number of Frames that both descriptions are lost; (c) Lost packets per GOP for substream 0; (d) Lost packets per GOP for substream 1.

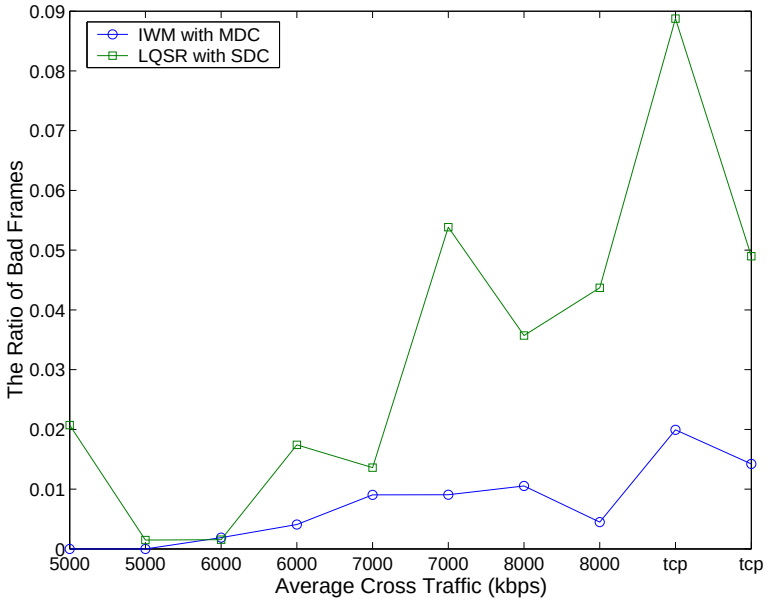


(a)

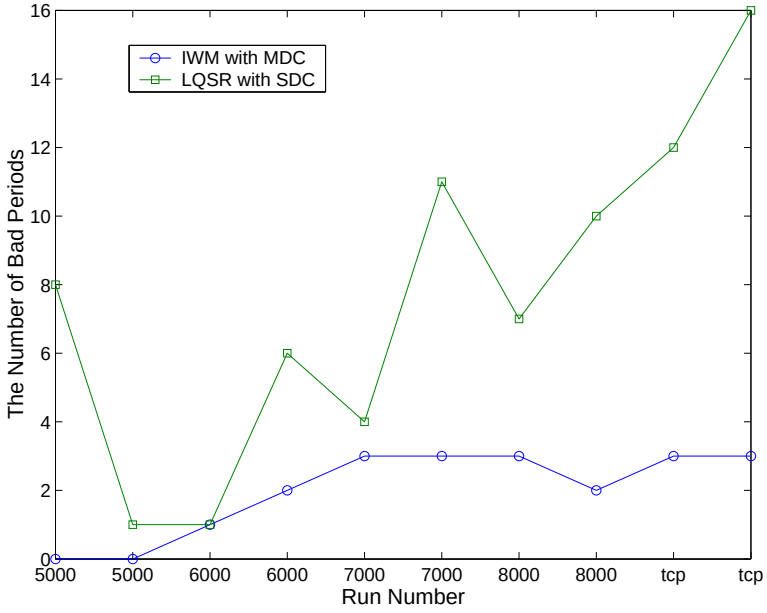


(b)

Figure 3.13. Performance evaluation for 802.11g with static nodes (a) PSNR of the received frames using LQSR and SDC; (b) Lost packets per GOP for the stream.



(a)



(b)

Figure 3.14. Performance Evaluation for 802.11a with static nodes (a) The ratio of bad frames; (b)The number of bad periods.

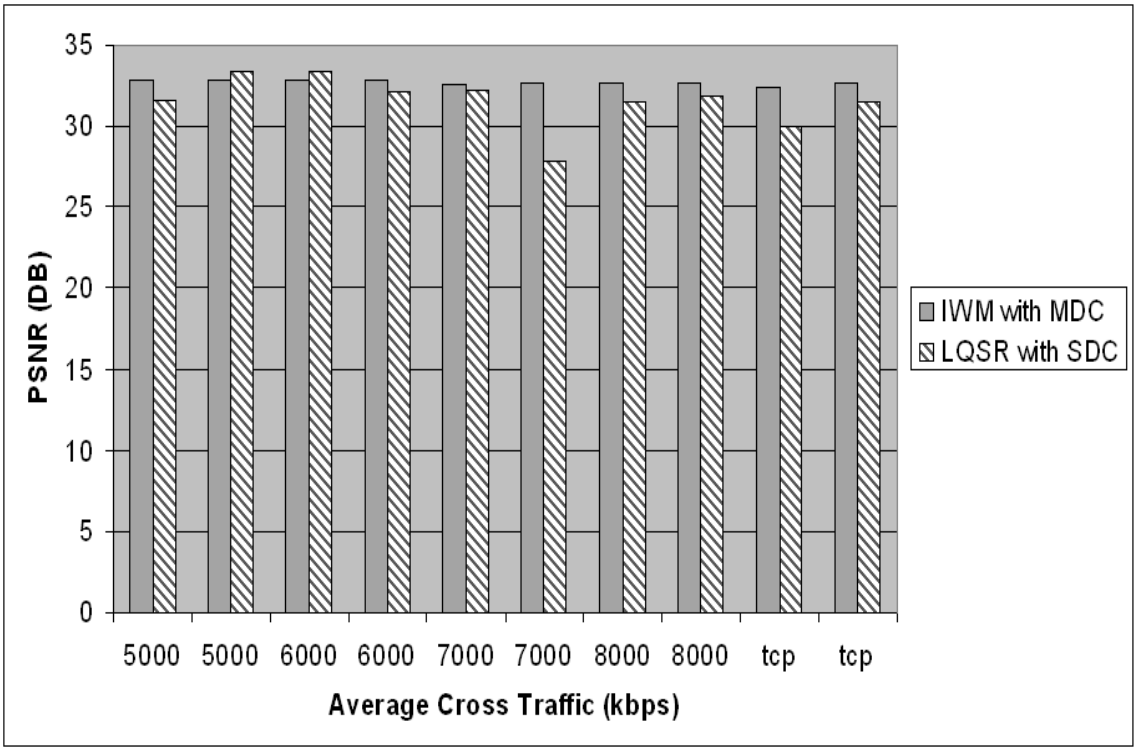
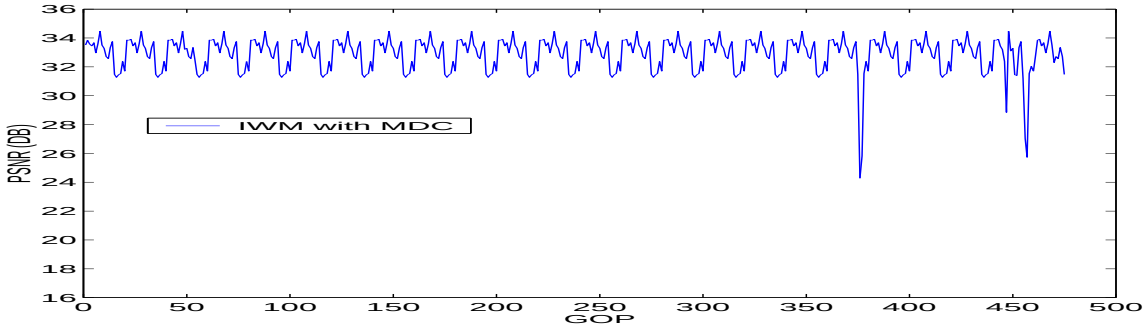
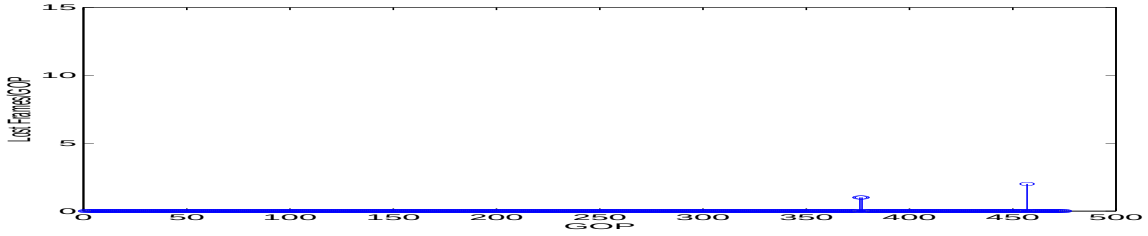


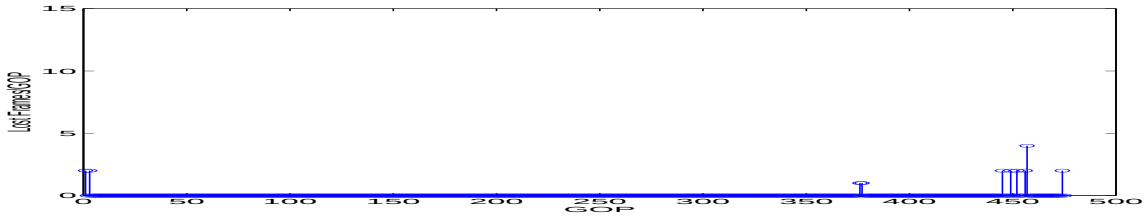
Figure 3.15. PSNR performance evaluation for 802.11a with static nodes



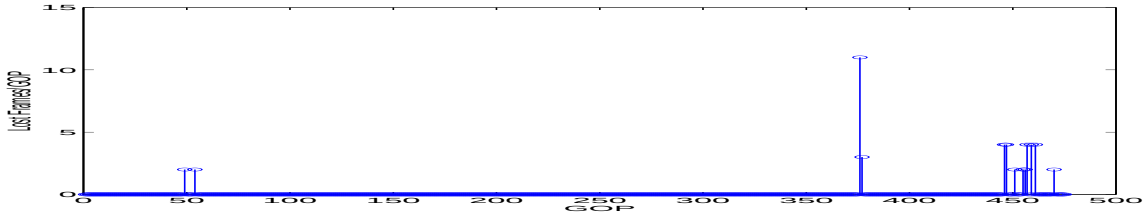
(a)



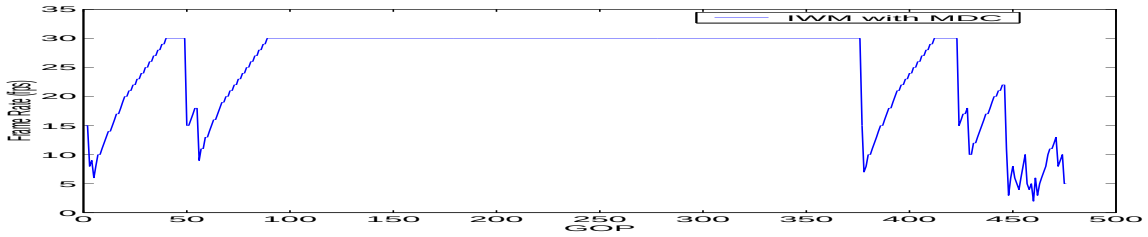
(b)



(c)

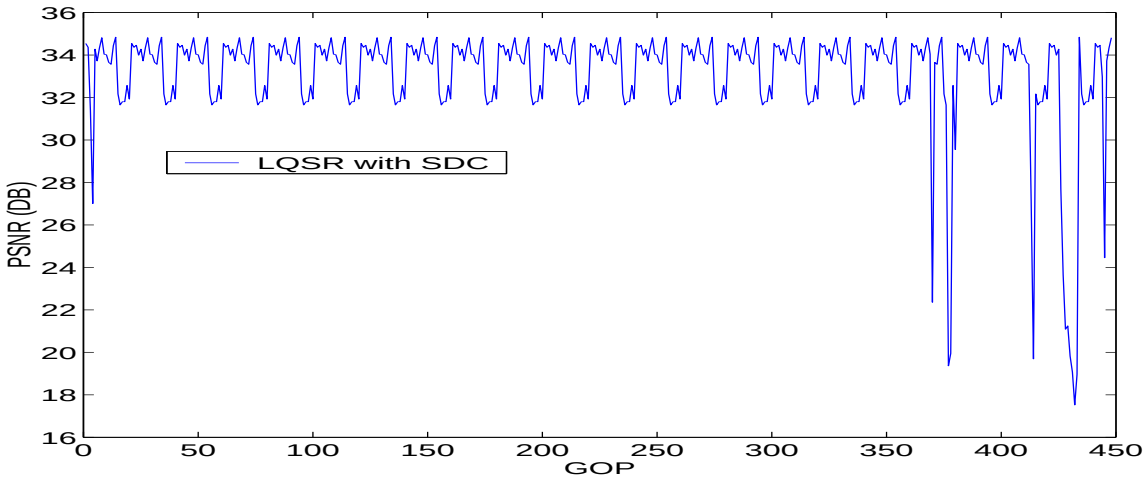


(d)

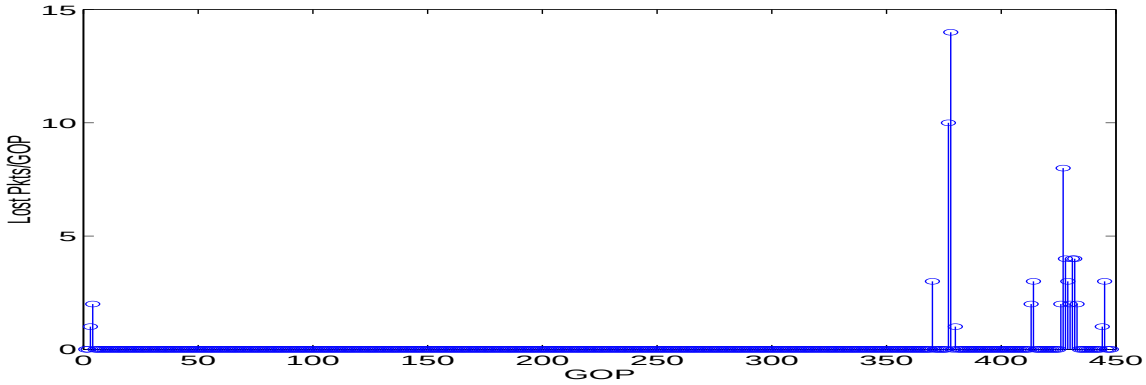


(e)

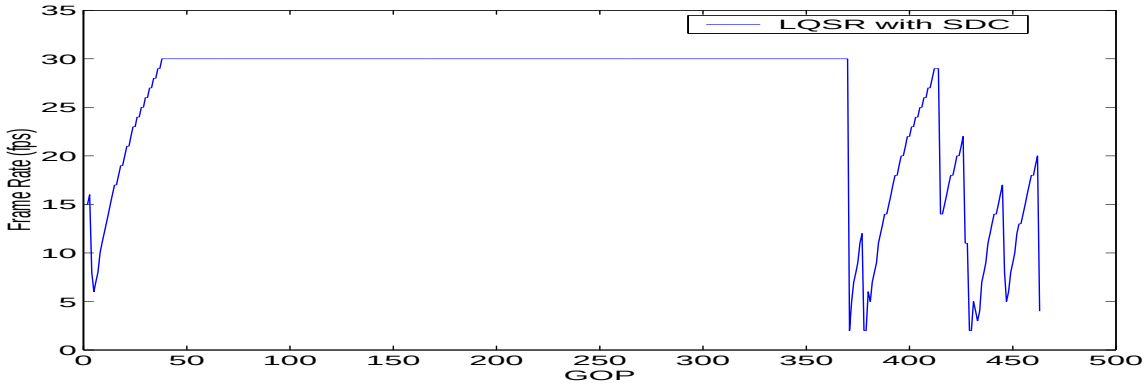
Figure 3.16. Performance Evaluation for 802.11a with static nodes (a) PSNR of the received frames using IWM and MDC; (b) Number of Frames that both descriptions are lost; (c) Lost frames per GOP for substream 0; (d) Lost frames per GOP for substream 1; (e) Sending frame rate.



(a)

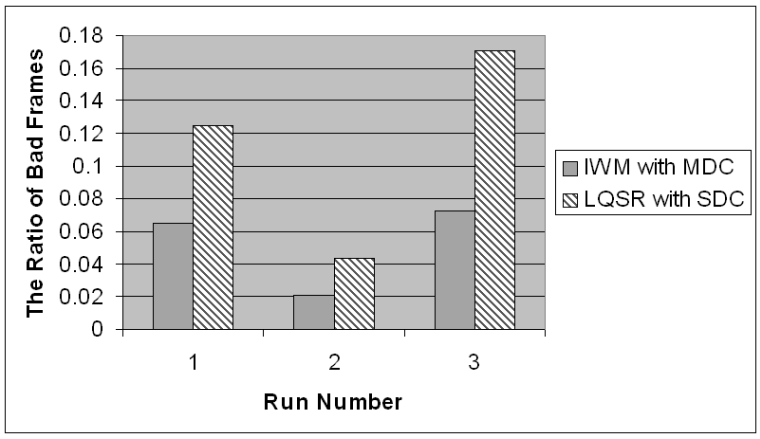


(b)

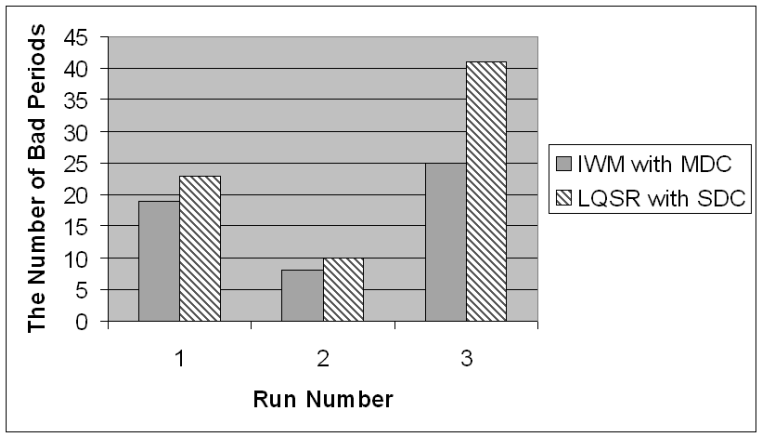


(c)

Figure 3.17. Performance Evaluation for 802.11a with static nodes (a) PSNR of the received frames using LQSR and SDC; (b) Lost frames per GOP for the stream; (c) Sending frame rate.



(a)



(b)

Figure 3.18. Performance Evaluation for 802.11a with moving node (a) The ratio of bad frames; (b)The number of bad periods.

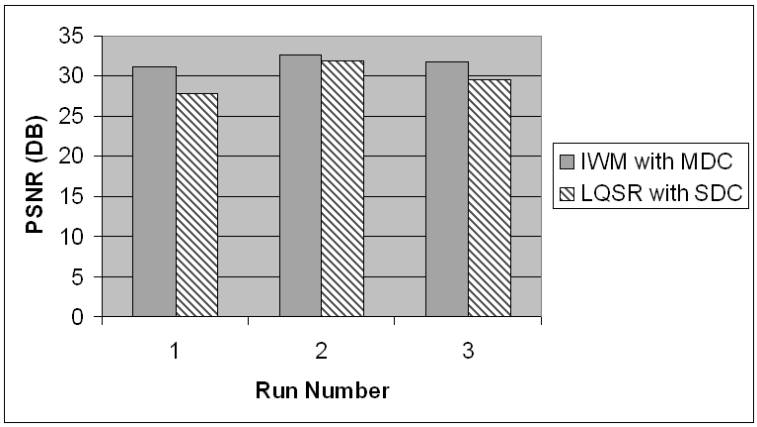
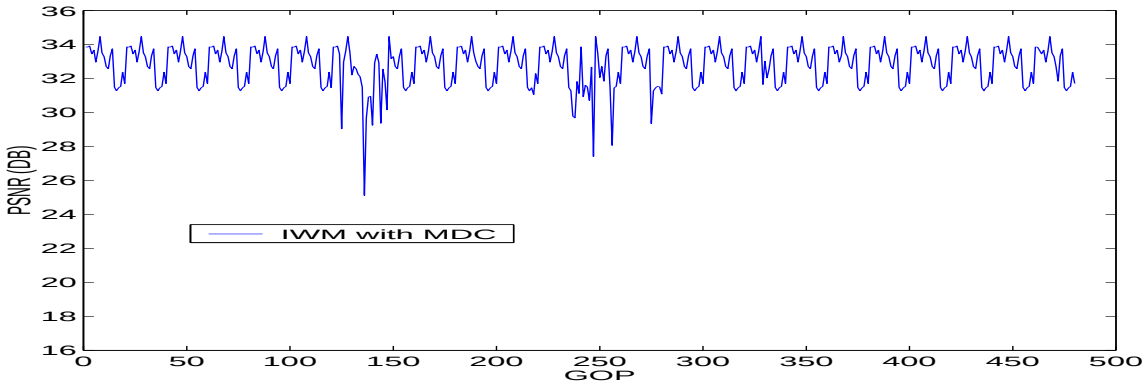
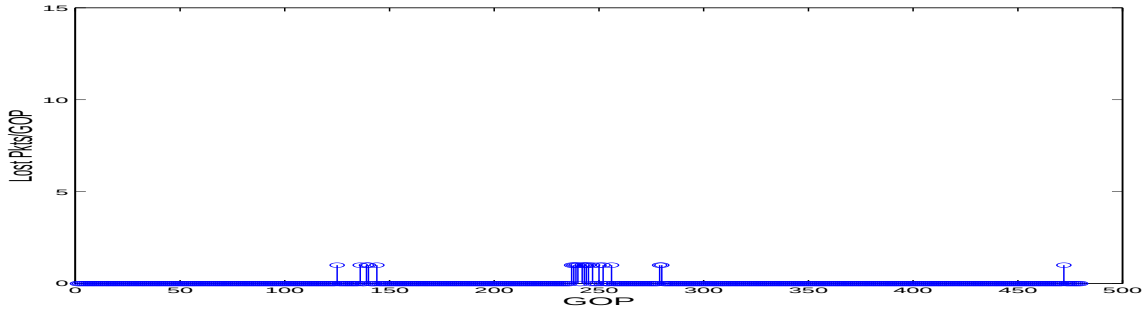


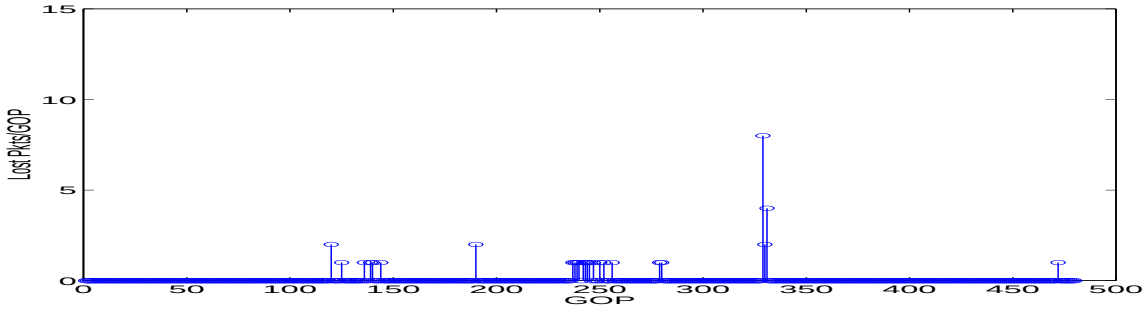
Figure 3.19. PSNR performance evaluation for 802.11a with moving node



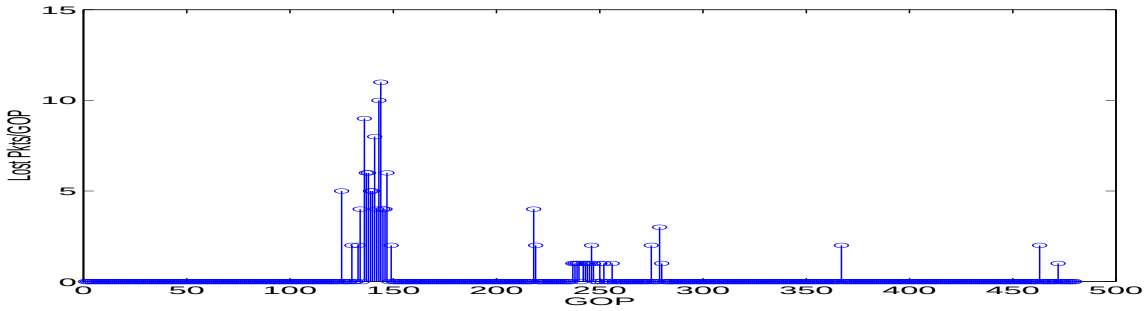
(a)



(b)

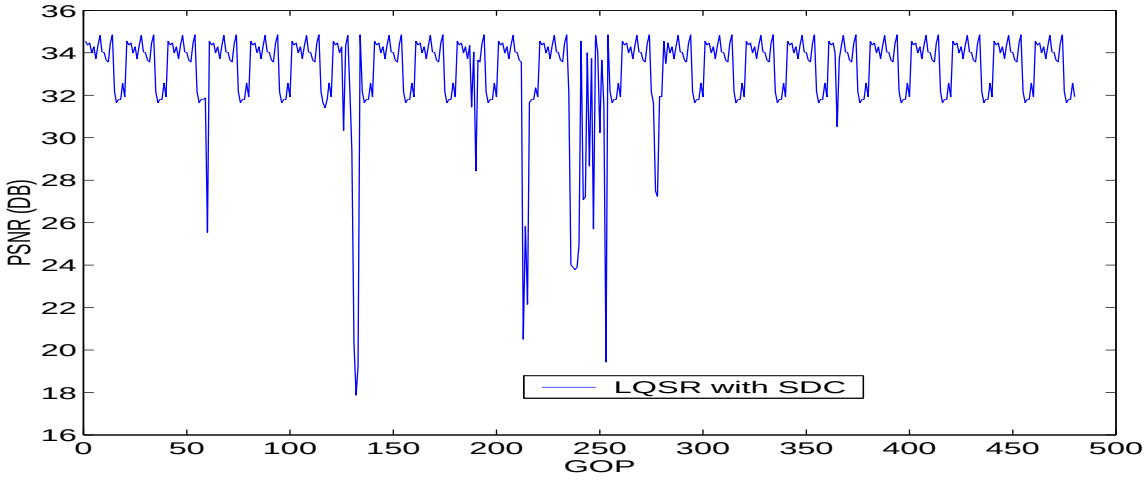


(c)

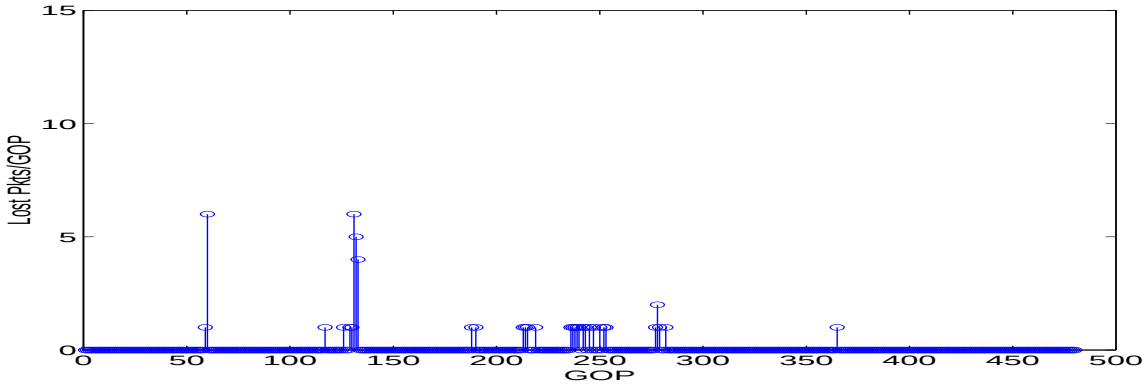


(d)

Figure 3.20. Performance Evaluation for 802.11a with moving node (a) PSNR of the received frames using IWM and MDC; (b) Number of Frames that both descriptions are lost; (c) Lost packets per GOP for substream 0; (d) Lost packets per GOP for substream 1.



(a)



(b)

Figure 3.21. Performance Evaluation for 802.11a with moving node (a) PSNR of the received frames using LQSR and SDC; (b) Lost packets per GOP for the stream.

Chapter 4

Multiple Tree Video Multicast in Wireless Ad Hoc Networks

In this chapter, we first introduce an architecture for multiple tree video multicast communication over wireless ad hoc networks. The basic idea is to split the video into multiple parts and send each part over a different tree, which are ideally disjoint with each other so as to increase robustness to loss and other transmission degradations. We show that for a given level of tree connectivity, the difference between the node densities required for single and double tree schemes is small. Thus building multiple trees does not increase the cost of network deployment significantly.

We then propose Serial MDTMR, which constructs two disjoint multicast trees sequentially in a distributed way, to facilitate multiple tree video multicast. This scheme results in reasonable tree connectivity while maintaining disjointness of two trees.

However Serial MDTMR has a larger routing overhead and construction delay than conventional single tree multicast routing protocols, as it constructs the trees in a sequential manner. To alleviate these drawbacks, we further propose Parallel

MNTMR in which nearly disjoint trees are constructed in parallel, and in a distributed way. Using the Parallel MNTMR, each receiver is able to always connect to two trees, regardless of the node density. Simulations show that multiple tree video multicast with both Serial MDTMR and Parallel MNTMR improve video quality significantly compared to single tree video multicast; at the same time routing overhead and construction delay of Parallel MNTMR is approximately the same as that of a single tree multicast protocol.

The rest of this chapter is organized as follows. In Section 4.1, we introduce a multiple tree video multicast framework. We discuss the feasibility of multiple tree multicast protocols in Section 4.2. We propose Serial MDTMR in Section 4.3, and introduce Parallel MNTMR and Parallel MNTMR with local subtree optimization in Section 4.4 and 4.5 respectively. Simulation results are represented in Section 4.6 and Section 4.7 concludes this chapter.

4.1 Multiple Tree Video Multicast Framework

Our proposed multiple tree multicast video communication system consists of two parts: a multicast routing protocol to construct multiple trees, and a scheme to distribute video packets into different trees. For the latter part, we employ MDC video to form multiple video streams, and transmit different video streams through different trees. If packet drop over two trees are not correlated, when some packets in one tree do not arrive at the destination on time, the receiver continues to decode and display packets corresponding to the other description on the other tree, resulting in acceptable video quality without interruption [86]. In this chapter, we assume that the network is lightly loaded, i.e. mobility and poor channel condition rather than congestion are major reasons for packet drop. In this case, multiple tree multicast

with MDC effectively alleviates undesirable effects caused by packet drop due to mobility and poor channels.

Our proposed multiple tree multicast packet forwarding works as follows. The application layer protocol sets a tree-flag in each packet's header to determine the tree to which the packet should be forwarded. The multiple tree multicast protocol forwards the packet in different trees according to the tree-flag as follows: when a node receives a data packet, it checks the node's *Forwarding Table* for the forwarding status and *Message Cache* to avoid forwarding duplicate data packet. The node forwards a non-duplicate packet forwarded in tree-y, if it is a forwarder for tree-y. Each packet flows along the corresponding tree from the sender to the receivers, but is not constrained to follow pre-set branches in the tree, as in the *tree flood* approach [24] or the *forwarding group flooding* approach [22]. Thus our packet forwarding scheme utilizes the broadcast nature of wireless ad hoc networks to obtain extra diversity gain without using extra network resources. Our packet forwarding scheme does not support packet forwarding across the trees, since nodes in one tree are unaware of the status of nodes in the other tree.

4.2 Feasibility of Multiple Tree Multicast Protocols

In this section, we first define the level of tree connectivity as the expected value of the ratio between the average number of receivers connected to each multicast tree and the total number of receivers. As we have argued, tree diversity is an effective technique to reduce the effects of tree failure, but it also reduces connectivity level as compared to single tree scheme for the same node density. If double tree scheme

requires a significant increase in node density in order to keep a high connectivity level, it may be too expensive to implement in practical situations.

Through analysis, we show that for a given level of tree connectivity, the difference between node densities required for single and double tree schemes is no larger than a logarithm factor of node density required for double tree schemes. This means that the required density for double tree scheme is not significantly larger than that of single tree scheme, and that tree diversity is a feasible technique to improve the robustness of multicast video transmission over wireless ad hoc networks.

4.2.1 Related Works and Problem Formulation

In recent years, there has been a number of papers on the problem of path connectivity in wireless ad hoc networks [87][88][89][90][91]. The problem can be described as follows: If two nodes are to be chosen at random, what is the probability of having a path between them? Let r denote the radio link range, λ the density of nodes, i.e. in the number of nodes per unit area, n the total number of nodes, and m the total number of receivers. We assume a Poisson Boolean model $B(\lambda, r)$, where nodes are distributed according to a Poisson point process of density λ in a square field of size A.

Philips et. al. [87] have shown that choosing $r = \sqrt{\frac{(1-\epsilon)\ln(A)}{\pi\lambda}}$ for any $\epsilon > 0$ leads to an asymptotically disconnected network as $A \to \infty$. They conjectured that choosing $r = \sqrt{\frac{(1+\epsilon)\ln(A)}{\pi\lambda}}$ would lead to asymptotic connectivity. Gupta and Kumar [88] have proved that $r = \sqrt{\frac{\ln(n)+c(n)}{\pi\lambda}}$ leads to asymptotic connectivity if and only if $c(n) \to \infty$ as $n \to \infty$. Dousse et. al. have stated that there exists one critical density λ_c, such that if the density $\lambda < \lambda_c$, all the connected clusters are almost surely bounded; otherwise, almost surely there exists one unique unbounded super connected cluster[89].

In this section, we first define tree connectivity as follows [92]:

Definition 4.1: Tree connectivity P is defined as

$$P \triangleq E[\frac{N}{M}] \tag{4.1}$$

where M is the product of the total number of receivers and the number of trees, $N = \sum_{i=1}^{m} n_i$, with n_i denoting the number of trees that receiver i connects to, and m denoting the number of receivers. For single tree scheme, $M = m$, and for double tree scheme, $M = 2m$. It can be shown that in general $0 \leq p \leq 1$. Given a random topology with n nodes, one random sender and m random receivers, N is the sum of all receivers connected to each multicast tree, and $E[N]$ is the expected value of N over all topologies. The definition of tree connectivity is a natural extension of that of path connectivity. It is actually asymptotic average ratio between receivers which connect to each multicast tree, and the total number of receivers, as the total number of nodes grows. Tree connectivity is a measure of the tree construction ability of a multicast routing protocol. For example, if we want to connect one sender to 20 receivers via 2 trees, and the resulting trees connect 18 receivers to 2 trees and 2 receivers to 1 tree, the tree connectivity $P = (18\times2+2)/(2\times20) = 0.95$. Alternatively if only 2 receivers are connected to 2 trees and 18 receivers are connected to 1 tree, the tree connectivity $P = (18 + 2 \times 2)/(2 \times 20) = 0.55$. Clearly, it is desirable to design tree construction scheme with as high tree connectivity level, P, as possible.

In this section, for a given level of tree connectivity, we study the relation between node density required for single and double tree schemes. The problem is formulated as follows: given tree connectivity level P, what is the relation between D_1 and D_2, the minimal node density to assure that tree connectivity above P for single and double disjoint multicast tree schemes?

We make the following assumptions:

1. We assume that node density of the network is higher than the critical density λ_c, i.e. the network is in the supercritical phase, and there exists one infinite unbounded super-cluster in the network. We use θ to denote the fraction of nodes belonging to the unbounded super-cluster. θ can also be defined as the probability of an arbitrary node belonging to the unbounded cluster, and is called percolation probability. Note that in the supercritical phase, θ is close to 1.

2. The total number of nodes is infinite and the size of the deployment area is infinite.

3. The ratio of the number of receivers to the total number of nodes is small.

We have summarized symbols used in this chapter in Table 4.1.

4.2.2 Single Multicast Tree Construction Schemes

We have the following result for single tree schemes.

Theorem 4.1: The tree connectivity level for a single tree scheme is given by

$$P_1 \approx \theta_1^2 \tag{4.2}$$

where θ_1 denotes the fraction of nodes belonging to the super-cluster in the single tree case.

The proof is shown in the Appendix B.1.

4.2.3 Double Disjoint Tree Construction Schemes

We establish the following results for double disjoint tree schemes.

Table 4.1. Symbols used in Chapter 4

Symbols	Definition
n	the total number of nodes
m	the total number of receivers
r	radio link range
θ	fraction of nodes belonging to the infinite super-cluster
S	the sender
R_i	Receiver i
$X(S)$	the number of nodes of the cluster to which the sender belongs
H	the set of nodes of the infinite super-cluster
P	tree connectivity level
N	$N = \sum_{i=1}^{m} n_i$
n_i	the number of trees that R_i connects to
M	the product of the total number of receivers and the number of trees
D_1, D_2	node density required for single and double tree schemes respectively to achieve the same tree connectivity level
P_1, P_2	tree connectivity level corresponding to single tree scheme with D_1, and double tree scheme with D_2
θ_1, θ_2	fraction of nodes belonging to the super-cluster corresponding to D_1 and D_2 respectively

Claim 4.1: Define receiver sets $A = \{R : \text{Ne(R)} \geq 2, R \in H\}$ and $B = \{R : R \in H'\}$, where Ne(R) denotes the number of receiver R's neighbors, H and H' denote the connected infinite supercluster and the one after removing all the middle nodes of the first tree respectively. Then there exists at least one tree scheme that makes $A \approx B$.

The proof of the claim is shown in the Appendix B.2.

Theorem 4.2: Consider an infinite wireless network, with nodes assumed to be distributed according to two-dimensional poisson process. Let D_1 denote the required node density to achieve a given tree connectivity level, P, in a single tree case. If $D_1 > \lambda_c$, there exists at least one double disjoint tree whose required node density

D_2 to achieve P satisfies

$$D_2 - \frac{\ln(\pi D_2 r^2 + 1)}{\pi r^2} \leq D_1 \leq D_2 \tag{4.3}$$

where r is the radio link range.

The detailed proof is shown in Appendix B.3.

We can see from **Theorem 4.2** that the difference between D_1 and D_2 is only a logarithm factor of D_2, which is small compared to the value of D_1 and D_2. The difference is negligible as $D_1, D_2 \to \infty$, which are requirements for keeping the network connected as the number of total nodes $n \to \infty$ [87][88]. Thus we conclude that the required density for double disjoint tree schemes is not significantly larger than that of single tree schemes, and that tree diversity is a feasible technique to improve the robustness of multicast video transmission over wireless ad hoc networks.

4.3 Serial Multiple Disjoint Tree Multicast Routing (Serial MDTMR)

Due to the nature of MDC, the less correlated packet drop between two trees, the more robust the video multicast. We assume that the network is lightly loaded, i.e. mobility and poor channel conditions rather than congestion are major causes of packet drop. In this case, if two trees do not share any middle nodes, packet drop over two trees are independent. Thus our main objective in the design of Serial MDTMR is to construct two node-disjoint multicast trees.

The proposed serial MDTMR constructs two node-disjoint trees in a distributed way. First we build a shortest path multicast tree. Then after requiring all the middle nodes in the first tree not to be middle nodes of the second tree, we construct another shortest path tree. Since these two trees do not share middle nodes at all,

they are node disjoint. Since Serial MDTMR is a way of constructing two disjoint multicast trees, it can be easily applied on top of any suitable single tree multicast routing protocol. Without loss of generality, we design the detailed Serial MDTMR based on ODMRP [22], since ODMRP has been demonstrated to perform well and is well known [23]. By comparing Serial MDTMR and ODMRP, it is easy to quantify the performance gain obtained by the multiple tree multicast routing. We can also design detailed Serial MDTMR based on other multicast routing protocols [23-30], taking advantage of their individual strengths. For example, we could apply a *local repair* scheme similar to [24] to maintain the tree structure with less control overhead. When a middle node or receiver detects that it is disconnected from the corresponding multicast forwarding tree tree-x, where x is 0 or 1, it initiates a *local repair* of tree-x, which searches the neighborhood of the middle node or receiver in order to find a new upstream node to reconnect the middle node or receiver to tree-x. To keep two trees disjoint, the middle node or receiver only selects a node, which is not a forwarding node for tree-(1-x), as its new upstream node.

Similar to ODMRP, group membership and multicast trees in Serial MDTMR are established and updated by the source on demand. When a multicast source has packets to send, it periodically triggers a two step multicast tree construction/refresh process. In the first step, the multicast source broadcasts to the entire network a JOIN REQUEST message, which includes the tree ID. When a node receives a non-duplicate JOIN REQUEST message for the first tree, it stores the upstream node ID, and rebroadcasts the packet. When the JOIN REQUEST message reaches a multicast receiver, the receiver unicasts a JOIN ACK message to the multicast source via the reverse shortest path. When a middle node in the reverse path receives a non-duplicate JOIN ACK message, it updates its corresponding forwarding state in the Forwarding Table, and forwards the message to its upstream node. Each middle

node of the tree only forwards the JOIN ACK message once in one tree construction cycle.

After receiving the first JOIN ACK message, the multicast source waits for a short time period before broadcasting another round of JOIN REQUEST message for the second tree in order to ensure the disjointness of two trees. When a node receives a non-duplicate JOIN REQUEST message, it forwards the packet only if it is not a middle node of the first tree in this round. When the JOIN REQUEST message reaches a receiver, the receiver unicasts back a JOIN ACK message to the multicast source to set up the second tree.

We compare tree connectivity of single shortest path tree and Serial MDTMR as a function of node density through simulations, as shown in Figure 4.1. Note that for Serial MDTMR, tree connectivity is averaged across two trees. The total number of nodes is 1000, with 50 receivers. The nodes are randomly distributed according to a two-dimensional poisson process. The results are averaged over 5000 runs. As seen, there is only a small performance gap between the two schemes when node density is larger than 7 nodes per neighborhood. For example, when node density is 8.2 nodes per neighborhood, tree connectivity of a single tree scheme and Serial MDTMR is around 0.99 and 0.95 respectively. Therefore, Serial MDTMR achieves reasonable tree connectivity, while maintaining the disjointness of two trees, through a very simple approach.

Figure 4.2 shows simulations results relating the required node density for single tree D_1 and Serial MDTMR D_2 for various tree connectivity levels, ranging from 0 to 1. The value of tree connectivity level for each point can be obtained from Figure 4.1, using the curve Serial MDTMR and node density D_2. It also shows corresponding lower bounds and upper bounds for node density D_1 according to Equation (4.3) provided by Theorem 4.2. As seen, Serial MDTMR curve fits in

between the lower and upper bounds quite well, which means that in terms of tree connectivity, the performance of Serial MDTMR is close to that of an ideal double disjoint tree construction scheme.

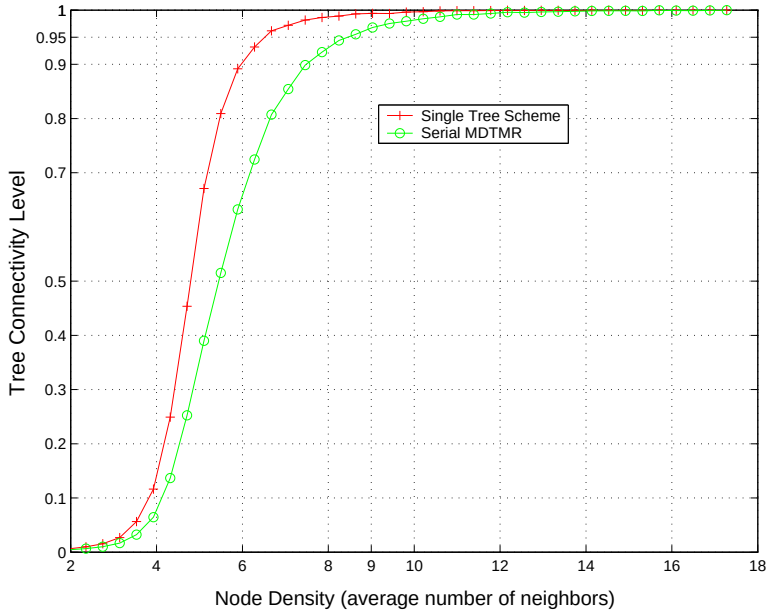


Figure 4.1. Tree connectivity of Serial MDTMR

4.4 Parallel Multiple Nearly-Disjoint Tree Multicast Routing (Parallel MNTMR)

Serial MDTMR achieves reasonable tree connectivity, while maintaining the disjointness of two trees. However its routing overhead and construction delay are potentially twice as much as that of a parallel scheme that would build two trees simultaneously. In this section, we propose a novel parallel double tree scheme, named Parallel MNTMR, to overcome the above disadvantages of Serial MDTMR. We have four main design goals for the Parallel MNTMR:

- *Low routing overhead and construction delay*: The routing overhead and con-

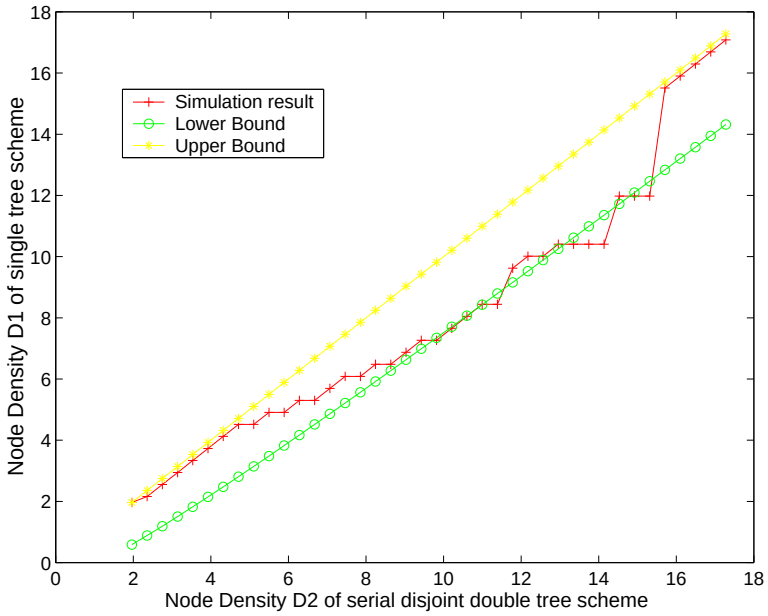


Figure 4.2. Comparison of node densities of Serial MDTMR with lower bounds and upper bounds

struction delay of Parallel MNTMR should be similar to that of a typical single tree multicast routing protocol.

- *High tree connectivity*: if a receiver is connected to the sender, it should be able to be connected to both trees.

- *Low tree similarity*: The ratio of the number of shared nodes of two trees to the number of nodes of the smaller tree should be minimized.

- *Distributedness*: The protocol should be fully distributed.

To measure the level of disjointness of two trees, we define tree similarity, S, between two trees as the ratio of the number of shared nodes to the number of middle nodes of the tree with a smaller number of middle nodes. Tree similarity between two disjoint trees is zero, and between two identical trees is one. The lower tree similarity between two trees, the lower correlated packet drop across two trees, and hence, the more effective multiple tree video multicasting is in achieving high video quality.

Ideally, we desire a multicast routing protocol to achieve both a high tree connectivity level and a low tree similarity level in video applications space. However, if the node density is low, it might not be possible to maintain both a high tree connectivity and a low tree similarity. In this case, Parallel MNTMR would first try to achieve high tree connectivity, and then try to make tree similarity as low as possible. We consider tree connectivity to be more important than tree similarity in the sense that a receiver first needs to connect to the sender in order to receive video packets. On the other hand, for sufficiently high node density or sufficiently large radio range, we would expect a routing protocol to be able to achieve both a high tree connectivity level and a low tree similarity level.

4.4.1 Overview

In a general single-tree multicast protocol, e.g. ODMRP, when a multicast source has packets to send, it triggers a multicast tree construction process by broadcasting a Join Query (JQ) message to its neighbors. Each node further forwards its earliest received JQ message to its neighbors, until the JQ message arrives at the receivers. Each receiver sends back a Join Reply (JR) message to the sender to construct the multicast tree.

In Parallel MNTMR, we apply similar JQ and JR processes to construct two nearly-disjoint trees in parallel. We require each node to forward the JQ message at most once in each JQ process, thus the amount of routing overhead of Parallel MNTMR is similar to that of a corresponding single tree multicast protocol.

The basic idea behind parallel tree construction is to classify all the nodes randomly into one of two categories, i.e. group 0 or group 1, based on uniform distribution. The protocol could potentially build tree-0 purely from nodes in group-0, and tree-1 purely from nodes in group-1, resulting in two node-disjoint trees. However this

approach achieves low tree connectivity when the node density is not high enough, since it is equivalent to partitioning the network into two networks with half of the node density. Route query messages may sometimes even be blocked in the middle of the network, causing some receivers, which are physically connected to the sender, to be connected to neither tree.

To increase the tree connectivity level, Parallel MNTMR forces each node, which is physically connected to the sender, to forward a JQ message once in a JQ process. We define a *pure JQ message* as one whose route only consists of nodes in the same group, and a *mixed JQ message* as one whose route consists of nodes in both groups. We also classify JQ messages based on the group of the last hop of the message. We call a JQ message a *group-0 message*, if the last hop belongs to group-0; otherwise we call it a *group-1 message*. We summarize the types of JQ messages in Table 4.2.

During the JQ process, nodes store selected JQ messages according to the *JQ message storing condition*, to be described shortly, in their *JQ Message caches*, so that they can use them later for upstream nodes selection in the JR process. A node forwards the earliest received JQ message of the same group as the node belongs to immediately, if it receives one. Otherwise it forwards the earliest received JQ message of the other group, after a short delay d from receiving it. The timeout of the JQ-delay timer should be large enough to differentiate between *pure JQ messages* and *mixed JQ messages*, at the same time as being small enough to reduce the overall delay in constructing the trees. Assuming t_p is the propagation delay of one hop, the delay of a *pure JQ message* traversing n hops is nt_p. The delay of a n-hop *mixed JQ message* is $nt_p + md$, where m is the number of group node changes within the message, since an extra delay d is incurred every time the current node and the last hop of the message are not in the same group. Thus *pure JQ messages* have lower overall delay, and are therefore selected and forwarded with a priority over *mixed JQ*

messages; as will be seen shortly, this improves the disjointness of the constructed trees in the JR process.

After receiving JQ messages, a receiver selects one upstream node for each tree according to the *upstream node selection rule*, to be described shortly, and sends back two JR messages to the sender via selected upstream nodes to initiate the process of constructing two trees. When a middle node receives a non-duplicate JR message, it also selects its upstream node according to the *upstream node selection rule*, and forwards the JR message to the selected upstream node, until the JR message reaches the sender. The basic idea behind the *upstream node selection rule* is to encourage nodes close to each other select the same upstream node for the same tree, and not to select middle nodes of the other tree. This is because ideally, one would like to make (a) middle nodes of one tree to serve as many nodes in a given neighborhood as possible, and (b) two trees to share as few middle nodes as possible. This is accomplished by synchronizing the selection of upstream nodes for different nodes in a distributed fashion.

Table 4.2. Classification of JQ messages

Type of JQ messages	Definition
Pure JQ message	A JQ message which is only forwarded by nodes in the same group
Mixed JQ message	A JQ message which is forwarded by nodes in both groups
Group-i JQ message	A JQ message whose last hop is a group-i node

4.4.2 Conditions and Rules

Parallel MNTMR applies the following conditions and rules on each node to control the flow of JQ and JR messages, in order to construct two trees with both high tree connectivity and low tree similarity. Without loss of generality, we assume the current node a is in group x, where x could be 0 or 1.

- *JQ message storing condition*: During the JQ process, node a stores received JQ messages in its *JQ Message cache*, so that it can later use them to select its upstream node in the JR process. However if node a stores every received JQ message, it is likely that the protocol constructs trees with loops. For instance, if node b receives a JQ message forwarded by node a, node a is a candidate upstream node for node b; similarly node a could receive a JQ message forwarded by node b, making node b a candidate upstream node for node a. This could potentially result in node a selecting node b and node b selecting node a as their upstream nodes at the same time, thus forming a loop.

  In order to obtain two loop-free trees in the JR process, each node only stores JQ messages satisfying the *storing condition* into its *JQ Message Cache*. Basically the storing condition helps node a eliminate those nodes which are possible offsprings of node a, as candidate nodes for its upstream node, thus avoiding loops in the constructed trees. A JQ message received by node a satisfies the *storing condition*, either if it is the first JQ message that node a receives in the current JQ round, or if the following two conditions are satisfied: (a) the number of hops it has travelled is no larger than that of the first received JQ message of node a plus one, and (b) the JQ message has not been forwarded by node a. Condition (a) helps node a eliminate those nodes, which have a much longer distance from the sender than the shortest distance, as candidate nodes for its upstream node, while condition (b) guarantees the trees to be loop-free.

- *JQ message forwarding condition*: Before node a in group x can decide as to whether or not to forward a JQ message, it has to determine whether a received JQ message is a group-x JQ message. If a JQ message satisfies the *forwarding condition*, it is forwarded immediately. Otherwise, it is either forwarded after a short delay, or not forwarded at all if node a has already forwarded a JQ message in this JQ round. The *forwarding condition* results in *pure group-x JQ messages*

being selected and forwarded with a priority over *mixed JQ messages*, thus helping the protocol construct trees that are as disjoint as possible. Formally a JQ message satisfies the *forwarding condition*, if the following two conditions hold true: (a) node a has not forwarded a JQ message in this JQ round, and (b) the message's last hop is the sender or a group-x node.

- *Upstream node selection rule*: The objective of the *upstream node selection rule* is to maximize the disjointness of two trees. Let JQM_a denote the set of all the messages in the *JQ Message Cache* of node a. If there exist both group-0 and group-1 JQ messages in JQM_a, node a selects last hops of the earliest received group-0 and group-1 JQ messages as upstream nodes for tree-0 and tree-1 respectively. Otherwise, we assume all the JQ messages in JQM_a are group-y JQ messages. In this case, if $|JQM_a| > 1$, node a selects last hops of the earliest and the second earliest received JQ messages as upstream nodes for tree-y and tree-$(1-y)$ respectively; otherwise if JQM_a only has one element, the last hop of the only JQ message is selected as upstream nodes for both tree-0 and tree-1.

  Using the *upstream node selection rule*, when nodes select an upstream node for tree-y, they are likely to select the same upstream node for tree-y as other close-by nodes would, and thus avoid the upstream node for tree-$(1-y)$ chosen by other nodes; this increases the likelihood of disjointness of two trees. The *upstream node selection rule* tends to synchronize different nodes' selection. It is effective since neighbors are likely to have received similar JQ messages with similar arrival times. We examine the effect of the *Upstream node selection rule* in the discussion section.

  Note that receivers and middle nodes share the same *upstream node selection*

rule. Receivers need to apply the rule twice for tree-0 and tree-1 separately, and middle nodes only apply the rule once for the specific tree they belong to.

4.4.3 Detailed Double Nearly-Disjoint Tree Construction

Similar to ODMRP, when a multicast source has packets to send, it triggers a multicast tree construction process by broadcasting a JQ message to its neighbors. When a node receives a group-y JQ message, if the message satisfies the *storing condition*, the node stores it into the *JQ Message Cache* for later usage in the JR process; otherwise, the message is simply discarded. If the message also satisfies the *forwarding condition*, the current node forwards the JQ message to its neighbors immediately; otherwise if the JQ message is the earliest received JQ message in the current JQ round, the node sets a JQ-delay timer. When the JQ-delay timer expires, if the node has not forwarded a JQ message in this JQ round, it forwards the earliest received JQ message at that time. The JQ-delay scheme tends to make *pure JQ messages* be selected and forwarded with a priority over *mixed JQ messages* in the distributed tree construction process. We provide pseudocode for processing JQ messages in Algorithm 2. We also provide flow diagram for processing JQ messages in Figures 4.3 and 4.4.

When a receiver receives a group-y JQ Message, if the message is a *pure JQ message*, and the node has not initiated a JR message in this JQ round for tree-y, it selects the last hop of this JQ message as its upstream node for tree-y, and unicasts a JR message to the sender via the selected upstream node, in order to set up tree-y. All nodes, receiving and forwarding the JR message for tree-y, become middle nodes for tree-y. The receiver also sets a timer upon receiving the earliest JQ message. When the timer expires, for each tree for which it has not already initiated a JR message, the receiver selects an upstream node according to the *upstream node selection rule*

and unicasts a JR message to the sender via the selected upstream node to construct that tree.

When a middle node receives a non-duplicate JR message for tree-x, it selects an upstream node according to the *upstream node selection rule*, and forwards the JR message to the upstream node. In the end, we obtain one tree mainly consisting of group-0 nodes and another mainly consisting of group-1 nodes. Therefore these two trees are likely to be nearly disjoint.

Algorithm 2 Processing JQ messages

```
if (The JQ message satisfies the storing condition) then
    Store the message into the JQ Message Cache
    if (The message satisfies the forwarding condition) then
        Forward the JQ message now
    else if The JQ message is the earliest received one in the current JQ round then
        Set a JQ-delay timer
    end if
    if (The JQ message is a group-y pure JQ message) And (The node is a receiver)
    AND (Not initiated a JR message for tree-y) then
        Select the upstream node
        Unicast a JR message to the sender for tree-y
    end if
else
    Discard the JQ message
end if
```

We visualize the tree construction process in Figure 4.5. The network consists of one sender (S), two receivers (R1 and R2), and five other nodes. The dashed lines denote the underlying topology of the network, dot-dashed arrows denote the construction of the first tree, and solid arrows denote the construction of the second

tree. We assume that nodes 1, 3, 4 belong to group-0, and nodes 2 and 5 belong to group-1.

Using our scheme, both R1 and R2 select node 4 as their upstream node for tree-0, and node 5 as their upstream node for tree-1. Let JQ_4 and JQ_5 denote node sets of last hops of JQ messages stored in JQ Messages Caches of nodes 4 and 5 respectively. $JQ_4 = \{1, 2\}$ and $JQ_5 = \{2, 3\}$. According to the *upstream node selection rule*, node 4 selects node 1 as the upstream node for tree-0, since node 1 is a group-0 node, while node 2 is a group-1 node. Using the same rule, node 5 selects node 2 as the upstream node for tree-1. Thus we obtain two disjoint trees, where middle nodes of tree-0 are nodes 1 and 4, and those of tree-1 are nodes 2 and 5. Note that each node learns the group of its candidate upstream nodes through JQ messages.

In our current implementation, we apply the periodic update scheme of ODMRP [22] to maintain the tree structure, since node movement causes unpredictable tree structure breakage. It is possible to apply other tree maintainence schemes, such as the adaptive demand-driven multicast routing (ADMR) [24], to reduce the amount of control overhead. When a node becomes a forwarder in tree-y, it sets its forwarding status to be 1 for tree-y. Once the forwarding status of a node is updated, it expires after a pre-specified time, i.e. JQ refresh period. The protocol runs JQ and JR processes every JQ refresh period to update forwarding status of forwarders, in order to maintain the tree structure during video transmission.

4.4.4 Discussion

In this section, we argue that the proposed Parallel MNTMR achieves the three design goals we introduced eariler. Firstly, the Parallel MNTMR builds two trees simultaneously, and each node forwards the JQ message at most once in one JQ round, therefore the routing overhead and the construction delay is similar to that

of a typical single tree multicast routing protocol. Secondly, as long as a receiver is connected to the sender, the protocol requires it to send JR messages for both trees, therefore the tree connectivity is the same as that of a single tree protocol. Thirdly, regarding the disjointness of the two trees constructed by Parallel MNTMR, we propose the following claim:

Claim 4.2: Given any two nodes N_a and N_b, which are middle nodes for tree-0 and tree-1 respectively, let JQ_a and JQ_b denote node sets of last hops of JQ messages stored in the JQ Message Caches of nodes N_a and N_b respectively. We sort nodes in JQ_a and JQ_b according to the arrival time of corresponding JQ messages. Let nodes N_c and N_d denote upstream nodes obtained by the Parallel MNTMR of nodes N_a and N_b respectively. We have $N_c \neq N_d$, if the first two nodes of JQ_a and JQ_b are the same. ■

The proof is shown in Appendix B.4. Intuitively, *claim 4.2* shows that if two nodes in different trees share the same first two JQ messages in their JQ message caches, they will not select the same node as their upstream nodes. Thus for many scenarios, the Parallel MNTMR is likely to maintain disjointness between two trees.

We now use the example in Figure 4.5 to demonstrate that the Parallel MNTMR reduces the number of shared nodes between two trees in other conditions. We use a three-bit code to denote the classification of nodes 1, 2, and 3, with the x^{th} bit representing the class to which node x belongs. For example, code 001 shows that nodes 1 and 2 are group-0 nodes and node 3 is a group-1 node. We enumerate all possible classifications of the nodes and all possible arrival sequences at node 4 and 5 of JQ messages forwarded by these three nodes manually, and summarize the results in Table 4.3. From Table 4.3, we see that averaged over all possible classification of all nodes, the probability that two nodes share a upstream node using Parallel MNTMR is 1/6, while choosing at random would have resulted in 1/4.

Table 4.3. Analysis of the scenario shown in Figure 4.5

Classification of nodes 1, 2 and 3	Probability that node 4 and 5 share the same upstream node
000	1/6
001	0
010	0
011	0
100	3/6
101	0
110	3/6
111	1/6
Average	1/6

4.5 Parallel MNTMR with Local Subtree Optimization (Parallel MNTMR-LSO)

As we have shown in the previous section, the proposed Parallel MNTMR achieves similar received video quality of a multiple tree multicast protocol, with similar control overhead of a single tree multicast protocol. However the two trees built by MNTMR are not guaranteed to be fully disjoint, reducing the effects of tree diversity. This effect is more serious when the node density of the network is low. We expect to further improve the received video quality, if we could reduce the number of shared nodes between two trees, i.e. making them more disjoint. In this section, we propose a local subtree optimization process with varying transmission power, to reduce the number of shared nodes for multiple tree multicast protocols. Specifically we apply this technique in conjunction with Parallel MNTMR.

The basic idea behind the local subtree optimization process is that a shared node a between two trees triggers a search in its local area to obtain a non-shared node as a replacement for itself in one tree, e.g. tree-0. The substitute should be able to connect the parent and children of node a in tree-0, in order to maintain the connectivity of tree-0. We then replace node a with the newly discovered non-shared

node, as the connection between the parent and children of the shared node a in tree-0. To increase the probability of finding a substitute for the shared node a, the local search is operated with higher transmission power, which is equivalent to increasing local area's node density.

We now describe the local subtree optimization process as follows. When a node, a, detects it is a shared forwarding node of two trees, it sends out a Tree-Opt-Inform message to its upstream node, u, of tree-0. [1]

If node u, receiving the first Tree-Opt-Inform message, is not a shared node itself, it stores the downstream node's ID into a local cache, and sets a tree-opt timer. Then node u waits for a small time period to receive and store more Tree-Opt-Inform messages from its shared downstream nodes. When the tree-opt timer expires, node u triggers a local subtree optimization process, by broadcasting a Tree-Opt-Query message carrying IDs of all the stored shared nodes. The time-to-live (ttl) bit of each Tree-Opt-Query message is set to be 1, to ensure the Tree-Opt-Query message to be forwarded only once in the upstream node's local area. Note that in the case that both the parent node and the child node are shared nodes, the protocol only optimizes for the parent node, since optimizing for both nodes simultaneously breaks tree structure unnecessarily.

We classify nodes into three categories: category 1, nodes that are not forwarders in tree-1; category 2, shared nodes; category 3, all other nodes. Ideally we wish to find nodes in category 1 as substitutes for shared nodes in tree-0, in order to reduce the number of shared nodes. It is useless to select a node from category 3 into tree-0, because the node will still be a shared node. Original shared nodes behave as backup parents, in a sense that they are still parents for their children, which can not find

[1] Without loss of generality, we assume every shared node to select tree-0 to optimize. This way, the parent only needs to search once on behalf of all its shared nodes. Future work could involve finding a scheme that chooses intelligently as to which tree to modify.

other category 1 nodes as their parents. We implement the above idea by using the forwarding status of shared nodes, where forwarding status has been described in Section 4.4.3.

In practice, when a node receives a non-duplicate Tree-Opt-Query message, with $ttl = 1$, if it is a category 1 node, it forwards the message immediately, in order to possibly reach children of shared nodes early, i.e. with a priority; otherwise if it is a category 3 node, it simply does nothing with the message; otherwise it is a shared node, which forwards the message after a short delay and sets its forwarding status for tree-0 to be expired in a short time period, behaving as a backup parent. By doing this, a child of a shared node would select a category 1 node as its new parent with a priority, if it could find one; otherwise it has to select a shared node as its parent, and send a message to the selected shared node to inform it to change back to a normal forwarder, by updating its forwarding status. If every child of a shared node a finds a new parent, i.e. node a does not receive any update message from its children, the forwarding status of node a expires shortly, and node a is no longer a middle node in tree-0. A child of a shared node does not need to explicitly inform its original parent if it finds a new parent, thus reducing the control overhead.

When a child of a shared node receives the earliest Tree-Opt-Query message, it unicasts a Tree-Opt-Reply message to the last hop of the Tree-Opt-Query message. When a node receives the Tree-Opt-Reply message, it updates its forwarding status for tree-0. We provide pseudocode for processing Tree-Opt-Query messages in algorithm 3.

Figure 4.6 shows one example of Parallel MNTMRLSO scheme. There are seven nodes in the network, one sender, two receivers and four other nodes. Nodes 1 and 2 are middle nodes for tree-0 and tree-1 respectively, and node 4 is a middle shared by tree-0 and tree-1. We use solid lines and dashed lines to represent tree-0 and tree-1

respectively. We use dashed-dot lines to represent the route for control messages. In step 1, shared middle node 4 sends a Tree-Opt-Inform message to its upstream node in tree-0, node 1. In step 2, node 1 broadcasts a Tree-Opt-Query message, after waiting for a short period. In our example, nodes 2, 3, and 4 receive the Tree-Opt-Query message. Since node 2 is a category 3 node, it simply discards the message. Node 3 is a category 1 node. In step 3, it forwards the Tree-Opt-Query message immediately, and both R1 and R2 receive the message. In step 4, both R1 and R2 unicast a Tree-Opt-Reply message to node 3. After receiving the message, node 3 becomes a forwarding node in tree-0, and connects node 1 and receivers R1 and R2. In step 5, node 4 forwards the Tree-Opt-Query after a short delay. However neither R1 and R2 responds to this message. The old forwarding status stored in node 4 will expire soon, and node 4 is disconnected from receivers R1 and R2.

Our goal of the local subtree optimization is to eliminate shared nodes without breaking the tree connection. Our scheme allows children of those shared nodes to select non-shared nodes, which are not forwarding nodes of tree-1, as upstream nodes for tree-0 with a priority. In the worst case, original shared nodes are selected, and the connection of the parent node and children nodes of shared nodes is maintained.

In a typical wireless ad hoc network scenario, the distribution of nodes are not uniform in the local space scale, i.e. node density is high in some area, and is low in some other area. The chances that shared nodes are in low density area is high, thus it is likely that the local subtree optimization could not find alternative nodes for shared nodes. To improve the performance of the local tree optimization process, Tree-Opt-Query and Tree-Opt-Reply messages are transmitted with a higher power than the transmission power used in the normal process, which is equivalent to increasing node density of local area, thus increasing the chances of eliminating shared nodes. Note that increasing the transmission power has some undesirable effects on the entire network, e.g. increasing the power consumption of individual nodes and increasing the

Algorithm 3 Processing non-duplicate Tree-Opt-Query messages

```
if (The ttl bit of the Tree-Opt-Query message > 0) then
  if (The node a category 1 node) then
    Forwards the Tree-Opt-Query message now
  else if (The node is category 2 node) then
    Forwards the Tree-Opt-Query message with a small delay
    Sets the node's forwarding status for tree-0 to be expired soon
  else
    Do nothing
  end if
end if
if (The node is a child of a shared node) then
  Unicasts a Tree-Opt-Reply message to the upstream node
end if
```

level of interference. However our main design goal of the system is to maximize the reliability of video multicast applications by exploiting the tree diversity. We observe that there is a trade-off between transmission power and reliability. In our local subtree optimization scheme, we select to raise the transmission power only in a small portion of nodes to increase the reliability level. Ideally, we should find the minimum transmission power of each node to maintain two disjoint trees, however the search of the minimum transmission power increases the routing overhead significantly. Thus we fix the transmission power at two levels in our proposed scheme. The common JQ and JR messages, and common data packets are transmitted with the low transmission power, while control packets used in the tree-opt process and data packets forwarded by the nodes obtained through the tree-opt process are transmitted with the high transmission power.

Our local subtree optimization scheme with varying transmission power is differ-

ent from traditional transmission power control schemes in wireless ad hoc networks [94][95][96][97]. The main objectives of traditional schemes are to minimize energy consumption and increase network throughput. In contrast, the objective of our proposed scheme is to minimize the number of shared nodes between two trees through increasing the transmission power of a small portion of forwarding nodes.

4.6 Simulation Results

We compare the performance of our proposed multiple tree multicast communication with that of multicast communication using ODMRP [22] through detailed packet-level simulations in various mobility and communication scenarios.

4.6.1 Simulation Setup

We use a simulation model based on NS-2 [76]. We briefly introduce NS simulation settings in 2.4.1. A detailed description of the simulation environment and the models is available in [72].

The random waypoint model [72] is used to model mobility. We only consider the continuous mobility case with zero pause time. To change the mobility level of the network, we vary the maximum speed from 2.5 m/s to 15 m/s. For each maximum speed, we randomly generate 30 different scenarios, and average the simulation results over those 30 scenarios.

In each run, we simulate a 50 node wireless ad hoc network within a 1500 $\times$ 300 square meter area. Each simulation is 900 seconds long, and results are averaged over 30 runs. The movement of the nodes and application-layer communication traffic are generated in advance so that they can be replayed identically for different multicast communication protocols.

We randomly choose one sender and eight receivers. For MDC we encode one frame into two packets, while for SDC we encode one frame into one packet. We set the frame rate to 8 fps, and GOP size to 15. For fairness, we set the Peak Signal to Noise Ratio (PSNR) of MDC and SDC to be approximately the same, i.e. 33 dB, where PSNR is a measure of video quality. To achieve approximately same quality, standard MPEG QCIF sequence Foreman is coded with MP-MDVC [65] at 64.9 kbps for MDC, and with Matching Pursuit Codec [66] at 41.2 kbps for SDC. We use the coded video traces to set size of video packets of the simulations. We consider interactive video applications in which the playback deadline of each packet is 150 milliseconds (ms) after it is generated.

We evaluate the performance of the multiple tree multicast communication using the following metrics:

a. **The ratio of bad frames:** In multicast scenario, the ratio of bad frames is the ratio of the number of bad frames experienced in all the receivers to the total number of frames that should have been decoded in all the receivers. For example, in a multicast group with 1 sender and 5 receivers, each video frame should be decoded a total of 5 times across 5 receivers. If 100 video frames are transmitted, and the receivers experience 50 bad frames in total, the ratio of bad frames is 0.1.

b. **The number of bad periods:** A bad period consists of contiguous bad frames. This metric reflects the number of times that received video is interrupted by the bad frames.

c. **Normalized packet overhead:** The total number of control packets transmitted by any node in the network, divided by the total number of video frames received by all the receivers. This metric represents the control packet overhead of the routing protocol normalized by the successful video frames received.

d. **Forwarding efficiency:** The total number of data packets transmitted by any node in the network, divided by the total number of packets received by all the receivers. This metric represents the efficiency of multicast forwarding of the routing protocol. For video applications, forwarding efficiency is more important than the control packet overhead, since the size of a video packet is generally much larger than the size of a control packet.

e. **Average hops of each packet:** The average number of hops that each packet takes. This metric represents the quality of the multicast trees as constructed by the multicast routing protocol.

f. **Tree similarity:** The ratio of the number of shared nodes to the number of middle nodes of the tree with a smaller number of middle nodes. This metric shows the level of disjointness of two trees.

4.6.2 Simulation Results for Serial MDTMR and Parallel MNTMR

We compare the following four schemes:

- Multiple tree video multicast with Parallel MNTMR and MDC;
- Multiple tree video multicast with Serial MDTMR and MDC;
- Single tree video multicast with ODMRP [22] and MDC;
- Single tree video multicast with ODMRP and SDC.

The reason we choose ODMRP as a benchmark for single tree protocols is that it outperforms many single tree protocols [23]. Through forwarding group flooding

technique, ODMRP uses redundant links inside the mesh structure to forward data packets, thus increasing packet delivery ratio.

ITAMAR is another multiple tree protocol, which builds edge disjoint or nearly-disjoint trees. However there are two obvious advantages of our proposed techniques as compared to ITAMAR[29]: first, our protocols are distributed, rather than centralized, and hence do not require the knowledge of network topology in advance; second, our protocols' overhead is $O(n)$, rather than $O(n^2)$ of ITAMAR[29], where n is the number of total nodes.

For fair comparison, all of three multicast routing protocols use 3 seconds for the JOIN REQUEST flooding interval, and use 4.5 seconds as a forwarding state lifetime.

Figures 4.7(a) and 4.7(b) show the result of the ratio of bad frames and the number of bad periods of the four schemes respectively. As expected, both the number of bad frames and the number of bad periods increase with maximum speed. As seen, performance of multiple tree multicast with Parallel MNTMR is close to Serial MDTMR, and they both perform much better than the other two schemes with ODMRP. Shown in Figure 4.8, two trees obtained by Parallel MNTMR only share approximately eight percent of nodes, which means they are nearly disjoint. This explains the reason the two multiple tree protocols perform similarly. The combination of our proposed multiple tree multicast protocols, e.g. Parallel MNTMR or Serial MDTMR, and MDC reduces contiguous packet loss caused by broken links of multicast tree, since links of two nearly-disjoint trees fail nearly independent, resulting in much better received video performance than that with ODMRP and MDC. By comparing ODMRP with MDC and ODMRP with SDC respectively, we conclude MDC by itself could also reduce scattered packet loss caused by wireless channel error, or packets collision, thus reducing both the ratio of bad frames and the number of bad periods.

We plot PSNR and loss traces of a randomly selected receiver using Parallel MNTMR with MDC and ODMRP with SDC in Figures 4.9 and 4.10 respectively. Every node moves randomly with a maximum speed 5.0 m/s. For MDC, it can be seen in Figure 4.9(a) that PSNR drops gracefully, when there is packet loss only in one substream. As seen in Figure 4.9, in Parallel MNTMR, most of the time, packet losses of two substreams do not overlap, thus reducing both the number and the amount of PSNR drops. The PSNR curve of ODMRP with SDC shown in Figure 4.10(a) has more frequent and severe drops than that of Parallel MNTMR with MDC; this is because PSNR drops for every packet drop in SDC video, and would drop severely when there is a burst of packet loss. We also visually examine the reconstructed video sequences under different schemes. For the video sequence obtained via Parallel MNTMR with MDC, we experience 6 short periods of distorted video in 900 seconds, while for the video sequence obtained via ODMRP with SDC, we experience 16 longer periods of more severely distorted video in the same time period.

Figure 4.11(a) shows the normalized control packets for the four schemes. Simulation results show that the number of normalized control packets of Parallel MNTMR is very similar to that of ODMRP, and is about 50 percent lower than that of Serial MDTMR. In order to construct double disjoint trees, Serial MDTMR has to broadcast JQ message twice in each routing cycle, while both Parallel MNTMR and ODMRP only broadcast once. Let n and m denote the number of nodes and receivers respectively, and k_1 and k_2 denote the number of middle nodes in tree-0 and tree-1 respectively. The number of control messages in one JQ round of ODMRP, Parallel MNTMR and Serial MDTMR are $n+m+k1$, $n+(m+k1)+(m+k2)$, and $2n+(m+k1)+(m+k2)$ respectively. In typical multicast scenarios, $(m + k_i) \ll n$, $i = 0, 1$, or else simple broadcasting scheme is more efficient than multicasting. Thus we see that Parallel MNTMR has approximately the same control overhead as ODMRP, and they both have significantly lower overhead than Serial MDTMR.

Figure 4.11(b) shows that the number of the normalized forwarded data packets is almost the same for all four schemes with Parallel MNTMR being slightly worse. This indicates that the performance gain of Parallel MNTMR and Serial MDTMR is not at the expense of forwarding a packet more times than ODMRP, rather by the combined effect of independent trees and MDC. Shown in Figure 4.12, the average number of hops travelled by each packet using Parallel MNTMR is similar to Serial MDTMR, and is only approximately four percent higher than that using ODMRP.

Figures 4.13(a) and 4.13(b) compare Serial MDTMR with MDC, Parallel MNTMR with MDC, ODMRP with MDC and ODMRP with SDC as a function of cross traffic level in the network. The maximum speed of each node is 5.0 m/s. There are four UDP flows in the network, connecting randomly selected senders and receivers. Each flow has the same rate, i.e. one fourth of the total flow rate. With the increase of cross traffic rates, the possibility of packet drop due to collision and congestion increases, thus both the ratio of bad frames and the number of bad periods of all four protocols increase. As seen, the performance of Serial MDTMR and Parallel MNTMR with MDC is better than ODMRP with MDC and ODMRP with SDC under varying cross traffic levels. From simulation results shown in Figure 4.13, our schemes work well, when the network is lightly loaded or medium loaded. When the network is heavily loaded, none of existing protocols would work. In that scenario, we could potentially combine our protocols with some multicast rate control schemes to reduce the network load.

Figure 4.14 shows the comparison of four schemes as a function of the number of receivers ranging from five to eighteen. Two multiple tree streaming schemes outperform ODMRP with SDC and ODMRP with MDC for different number of receivers.

We also compare four schemes as a function of node density of the network, and show the results in Figures 4.15(a) and 4.15(b). We fix the size of the area to be 1000 $\times$ 1000 square meters, and vary the node density from 2.5 to 9.0 neighbors per node by varying the number of nodes in the network. The maximum speed of each node is 5.0 m/s. As seen, the performance of two multiple tree streaming schemes outperform that of ODMRP with SDC for all node densities. When the node density is smaller than 6 neighbors per node, the performance of Parallel MNTMR with MDC and Serial MDTMR with MDC improve with node density, since our proposed multiple tree protocols build more disjoint trees with higher node density. However when the node density is higher than 6 neighbors per node, the performance of two multiple tree streaming schemes gets worse with higher node density, because of the increase in routing overhead caused by larger number of nodes in the network. From Figure 4.16, the control overhead of Serial MDTMR grows faster than that of Parallel MNTMR with the increase of node density, which explains the reason that Parallel MNTMR outperforms Serial MDTMR when node density is higher than 6 neighbors per node.

4.6.3 Simulation Results for Parallel MNTMR-LSO

To test Parallel MNTMR-LSO, we simulate a 50 node wireless ad hoc network within a 800 $\times$ 800 square meter area. Low transmission power is 0.1154, which radio range is 200 meters, and high transmission power is 0.2818, which radio range is 250 meters. All other settings are the same parameters described in Section 4.6.1.

We evaluate the performance of the multiple tree multicast communication with Parallel MNTMR-LSO using the following metrics: the ratio of bad frames, the number of bad periods, normalized packet overhead, forwarding efficiency, average hops of each packet, and tree similarity, which are defined in Section 4.6.1.

We compare the following two schemes:

- Multiple tree multicast communication with Parallel MNTMR and MDC;
- Multiple tree multicast communication with Parallel MNTMR-LSO and MDC.

Figures 4.17(a) and 4.17(b) show the result of the ratio of bad frames and the number of bad periods of the two schemes respectively. Over scenarios with maximum speed lager than 2.5 m/s, both the ratio of bad frames and the number of bad periods of multiple tree video multicast with Parallel MNTMR-LSO are around 10 to 20 percent lower than those of multicast with Parallel MNTMR. Shown in Figure 4.18, the level of tree similarity is reduced almost 50 percent, which means the two trees obtained by Parallel MNTMR-LSO is much more disjoint than those obtained by Parallel MNTMR. This is the main reason the received video quality improves with Parallel MNTMR-LSO.

Figure 4.19(a) shows the normalized control packets for the two schemes. The number of normalized control packets of Parallel MNTMR-LSO is about 10 percent higher than that of Parallel MNTMR, since Parallel MNTMR-LSO uses extra control packets for the local subtree optimization. Also there are about 10 percent middle nodes of tree-0 forward data packets with the high transmission power. This shows that Parallel MNTMR-LSO improves the received video quality at the expense of extra control overhead and interference comparing to Parallel MNTMR. Thus Parallel MNTMR-LSO is more suitable for ad hoc networks with a low node density, for which the Parallel MNTMR could not build two trees disjoint enough to satisfy the reliability requirements of video multicast applications.

Figure 4.19(b) shows that the number of the normalized forwarded data packets is almost the same for the two schemes with Parallel MNTMR-LSO being slightly worse. Shown in Figure 4.20, the average number of hops travelled by each packet using Parallel MNTMR-LSO is only approximately five percent shorter than that using Parallel MNTMR. This indicates that the performance gain of Parallel MNTMR-

LSO is mainly achieved by forwarding MDC packets over two trees more disjoint than those obtained by Parallel MNTMR.

4.7 Conclusions

In this chapter, we study the problem of real-time video multicast communication over wireless ad hoc networks. We have proposed multiple tree video multicast with MDC to provide robustness for video multicast applications. Specifically, we first propose a simple distributed protocol, Serial MDTMR, which builds two disjoint trees in a serial fashion. This scheme results in good tree connectivity while maintaining disjointness of two trees. In order to reduce the routing overhead and construction delay of Serial MDTMR, we further propose Parallel MNTMR, which constructs two nearly disjoint trees simultaneously in a distributed way. Simulation shows that video quality of multiple tree multicast video communication is significantly higher than that of single tree multicast video communication, with similar routing overhead and forwarding efficiency.

The Parallel MNTMR can not guarantee node disjointness between the obtained trees, especially when node density of the network is low, which reduces the robustness of video multicast applications. To alleviate this problem, we further propose a local tree optimization process with varying transmission energy for Parallel MNTMR, which reduces the number of shared nodes of two trees significantly and enhances the received video quality with slightly larger routing overhead compared to the original Parallel MNTMR.

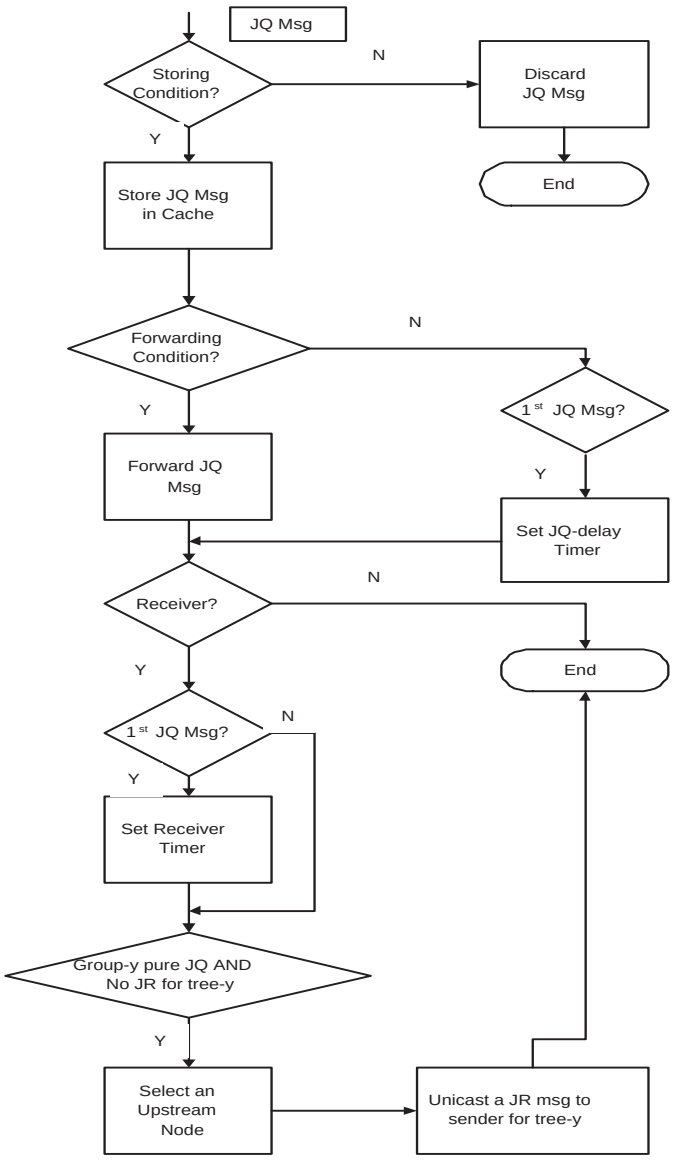
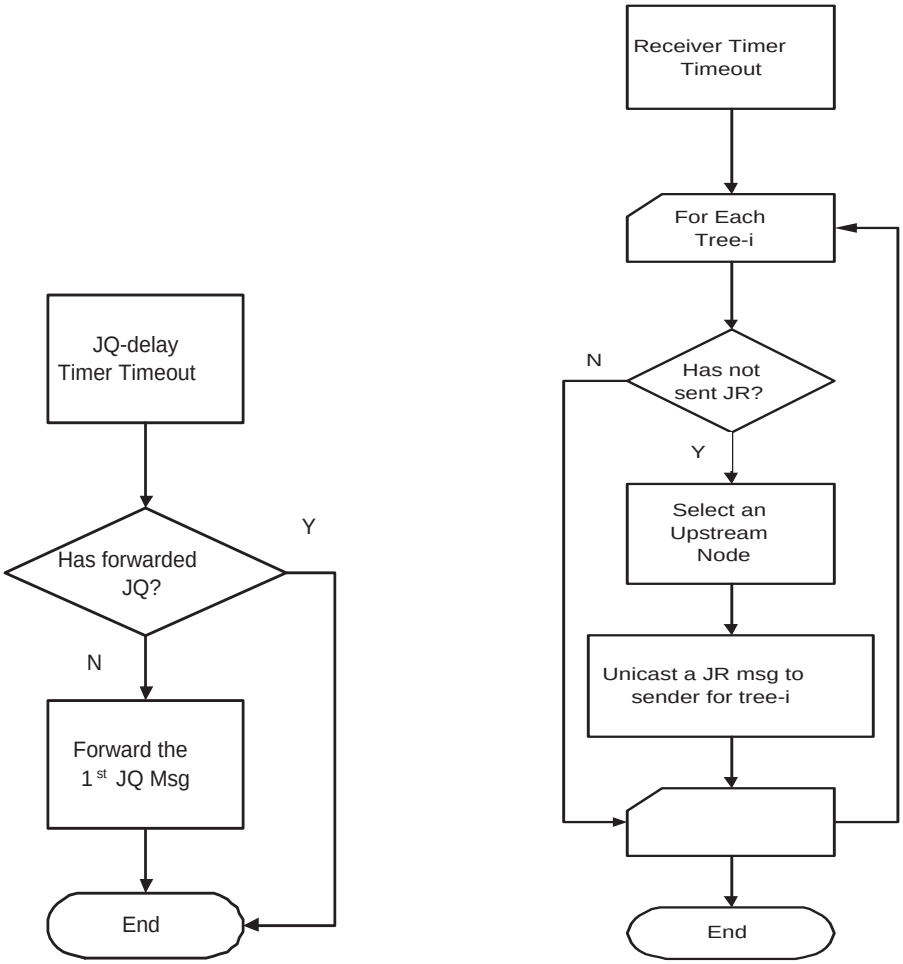


Figure 4.3. Flow diagram for processing JQ messages for Parallel MNTMR



(a) (b)

Figure 4.4. Auxiliary flow diagrams for processing JQ messages for Parallel MNTMR: (a)JQ-delay timer timeout; (b)Receiver Timer timeout.

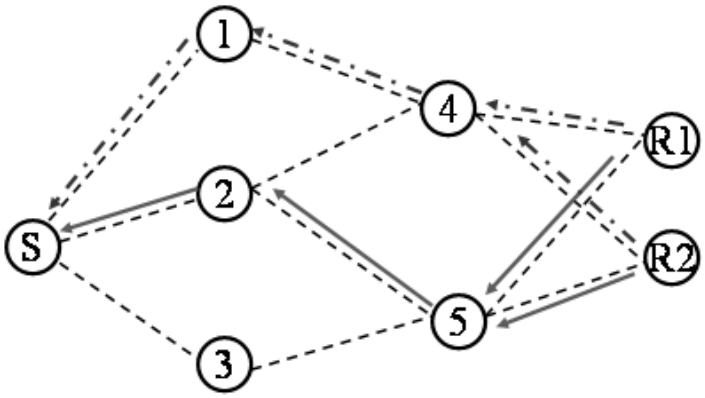


Figure 4.5. Double Nearly-Disjoint Tree Construction

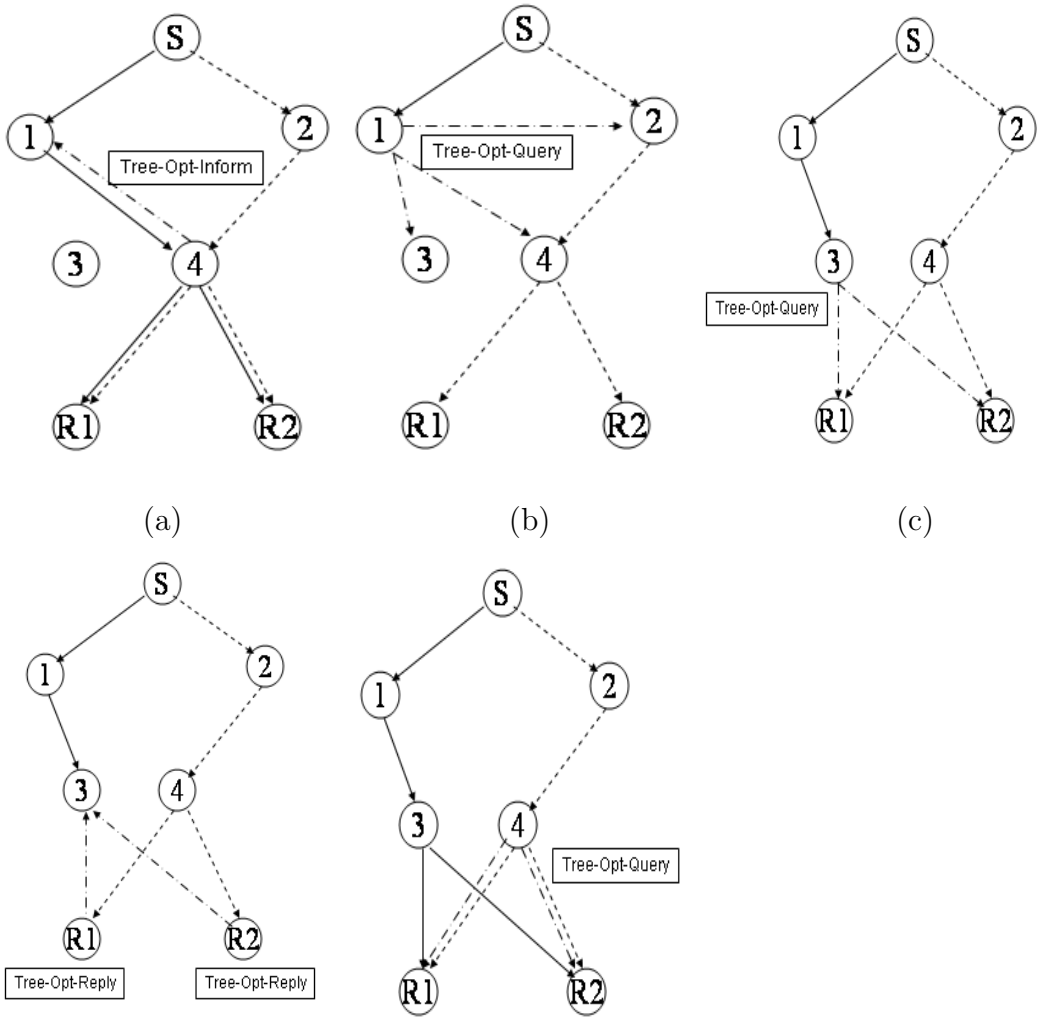
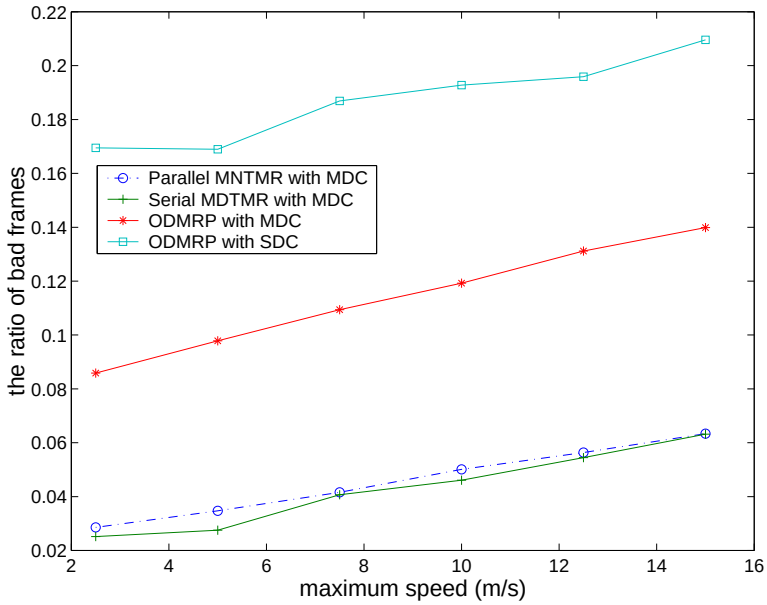
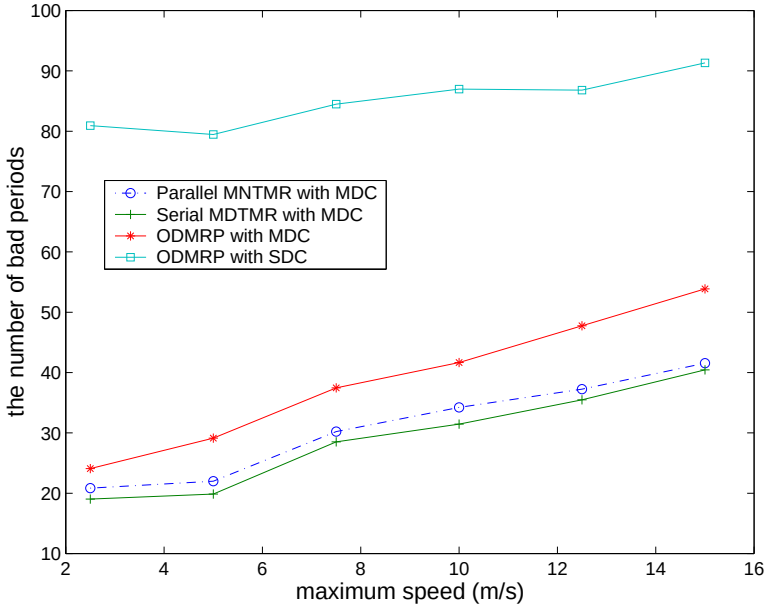


Figure 4.6. Demonstration of Parallel MNTMR-LSO scheme: (a)step 1; (b)step 2; (c)step 3; (d)step 4; (e)step 5



(a)



(b)

Figure 4.7. Performance evaluation for multiple tree video multicast Parallel MNTMR, Serial MDTMR and ODMRP: (a) The ratio of bad frames; (b)The number of bad periods.

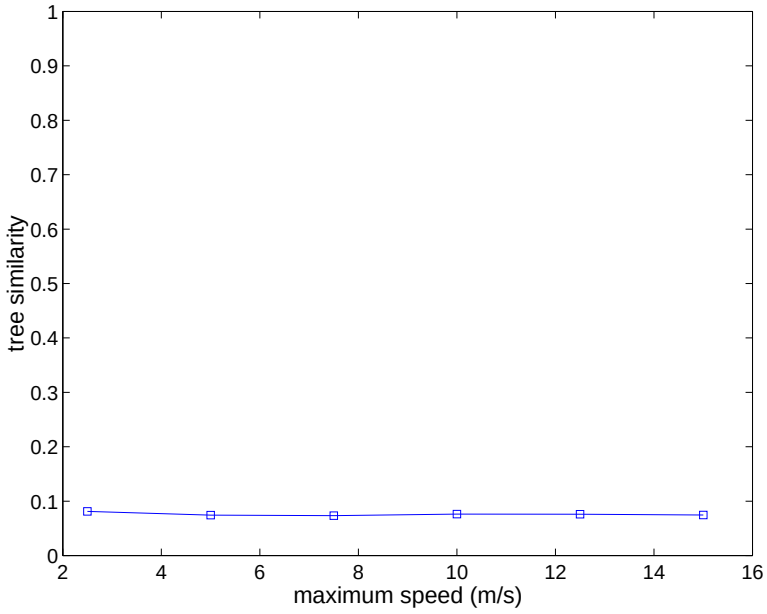
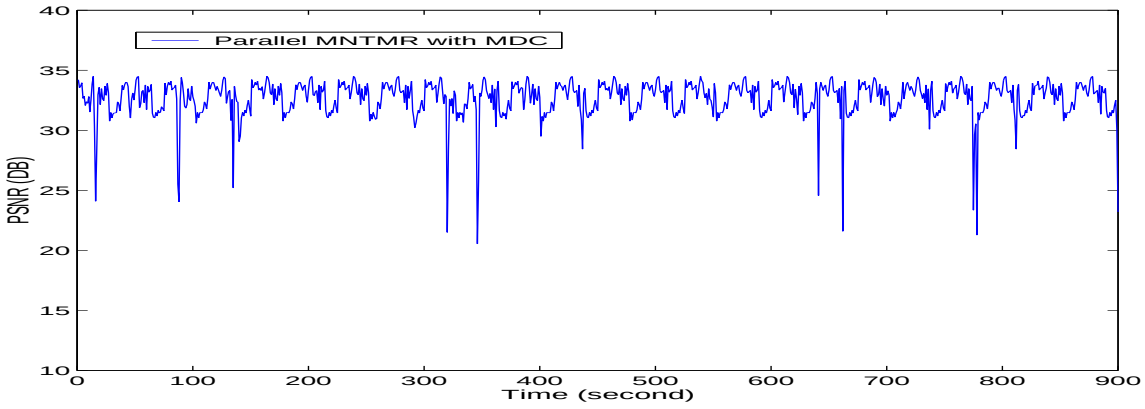
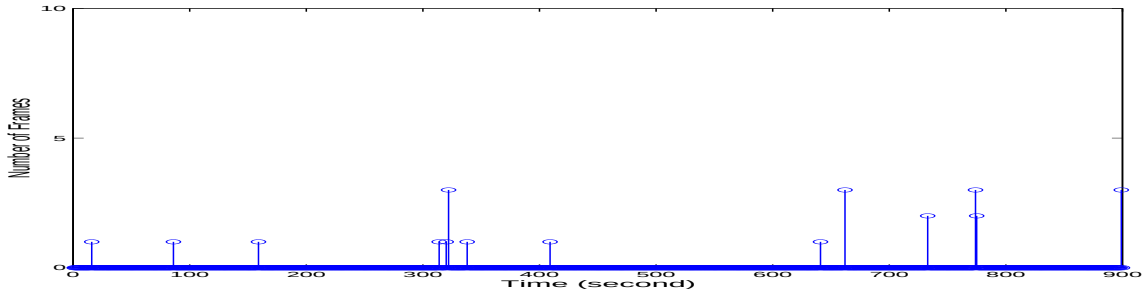


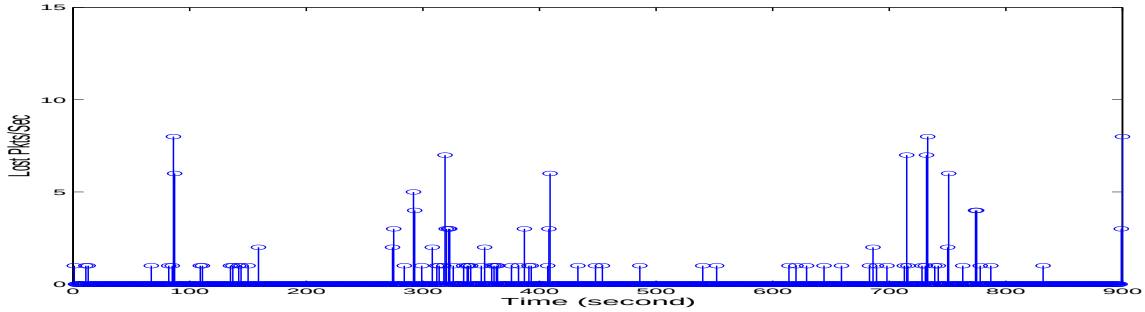
Figure 4.8. Tree Similarity of Parallel MNTMR



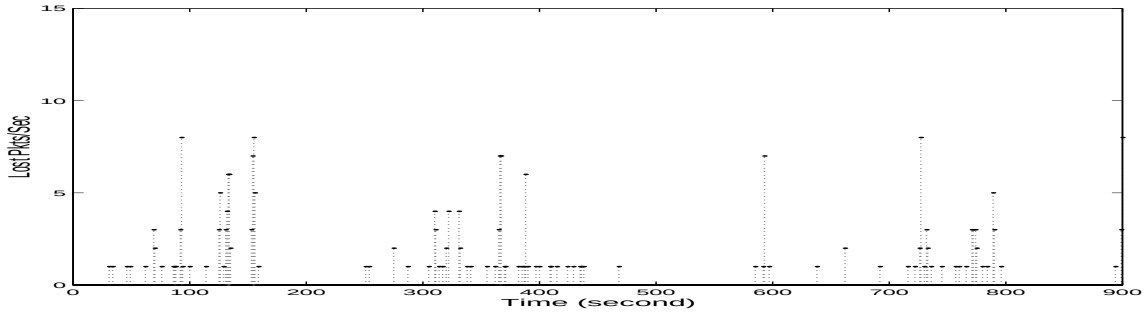
(a)



(b)

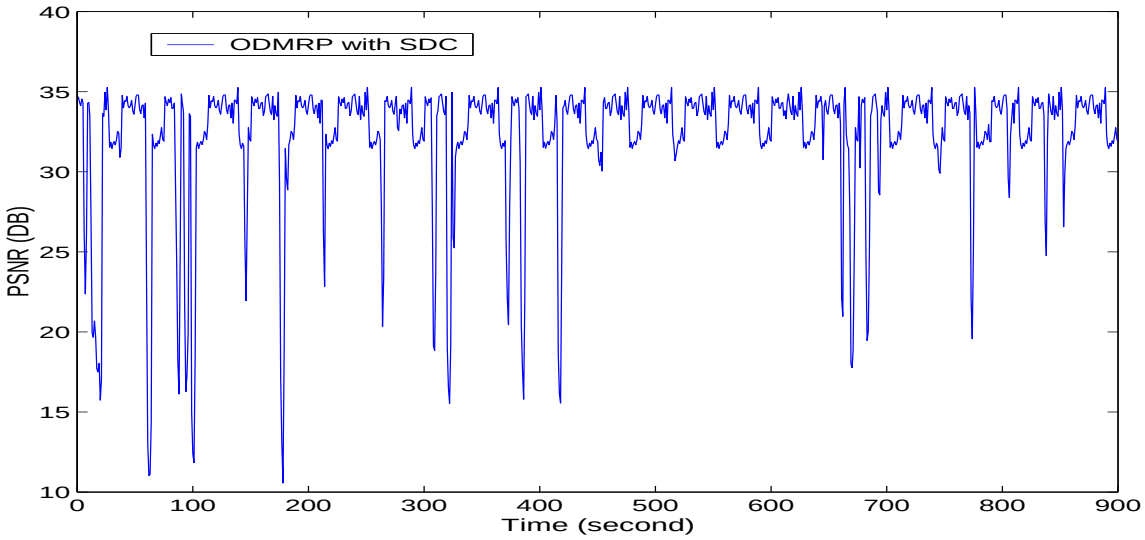


(c)

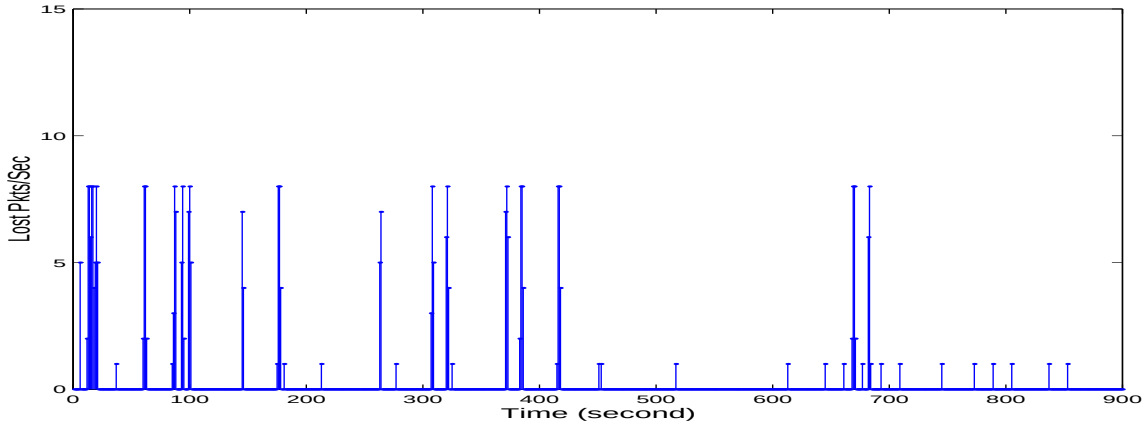


(d)

Figure 4.9. (a) PSNR of the received frames using Parallel MNTMR and MDC; (b) Number of Frames that both descriptions are lost; (c) Lost packets per second for substream 0; (d) Lost packets per second for substream 1.

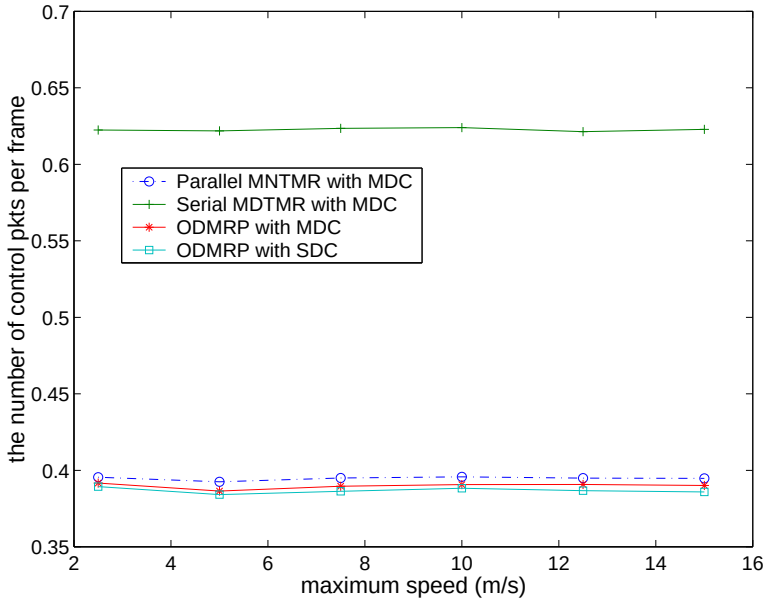


(a)

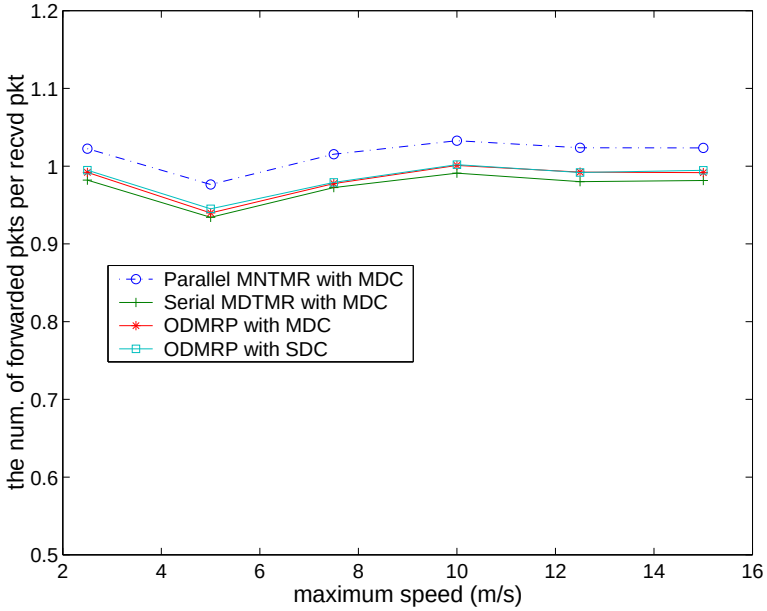


(b)

Figure 4.10. (a) PSNR of the received frames using ODMRP and SDC; (b) Lost packets per second for the stream.



(a)



(b)

Figure 4.11. Performance evaluation for multiple tree protocols Parallel MNTMR, Serial MDTMR and ODMRP: (a) The normalized control packets; (b) The normalized forwarded data packets.

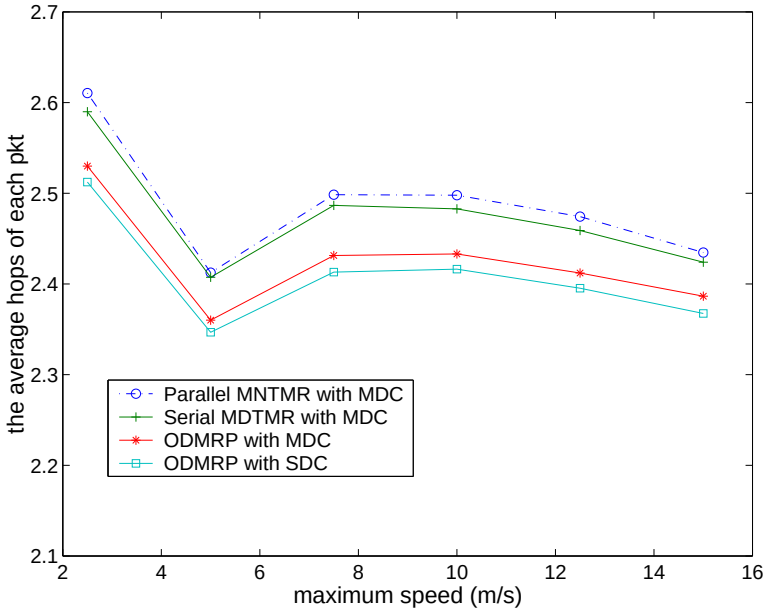
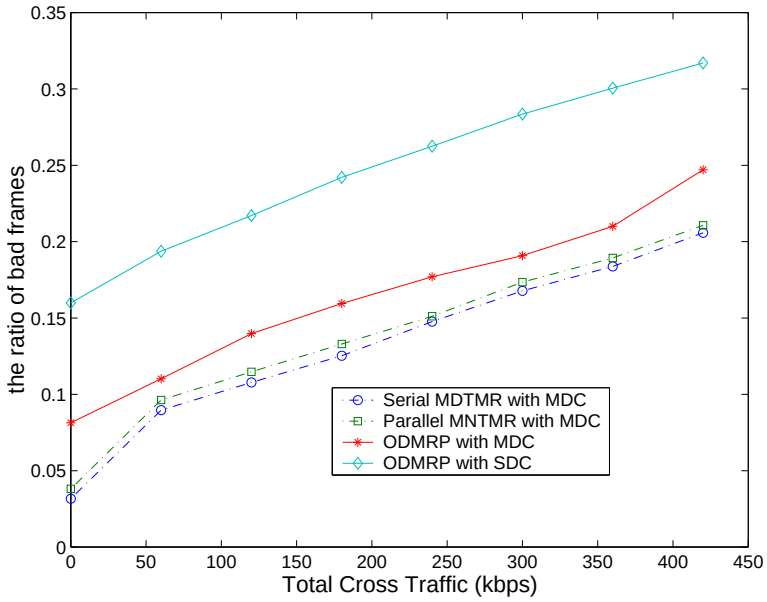
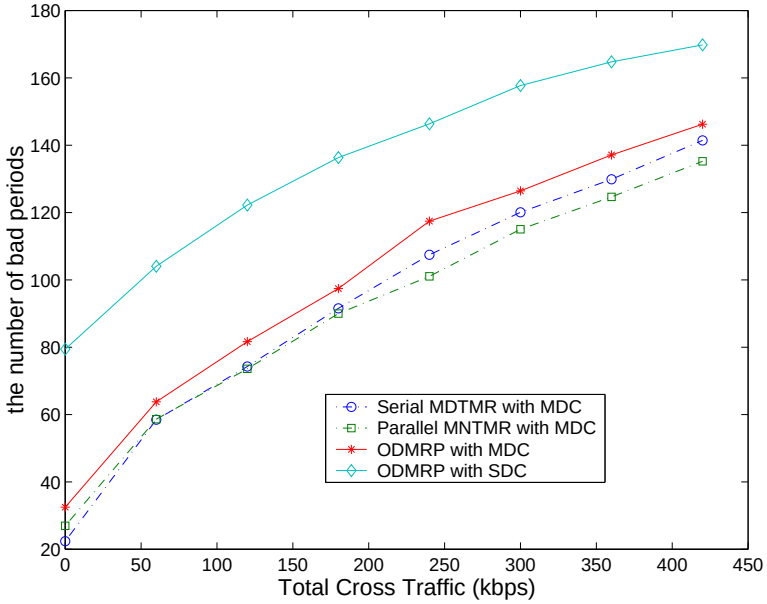


Figure 4.12. Performance evaluation for multiple tree protocols Parallel MNTMR, Serial MDTMR and ODMRP: The averaged number of hops.

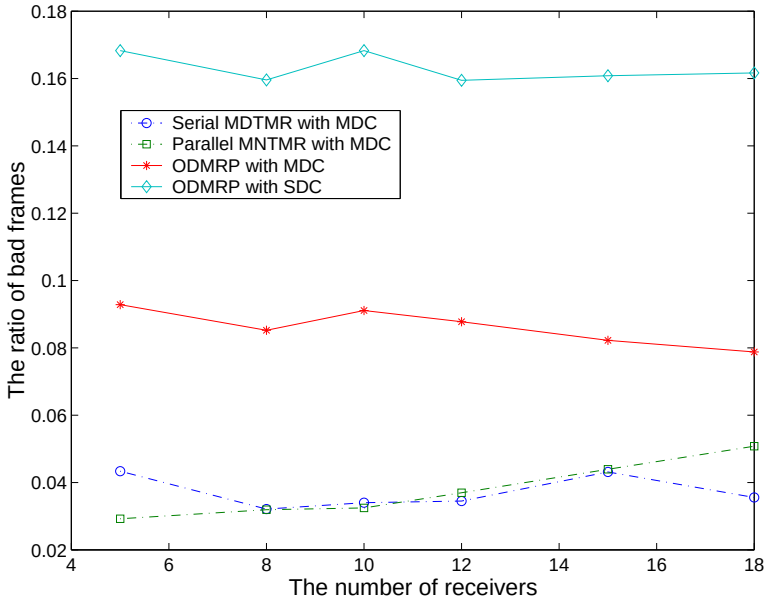


(a)

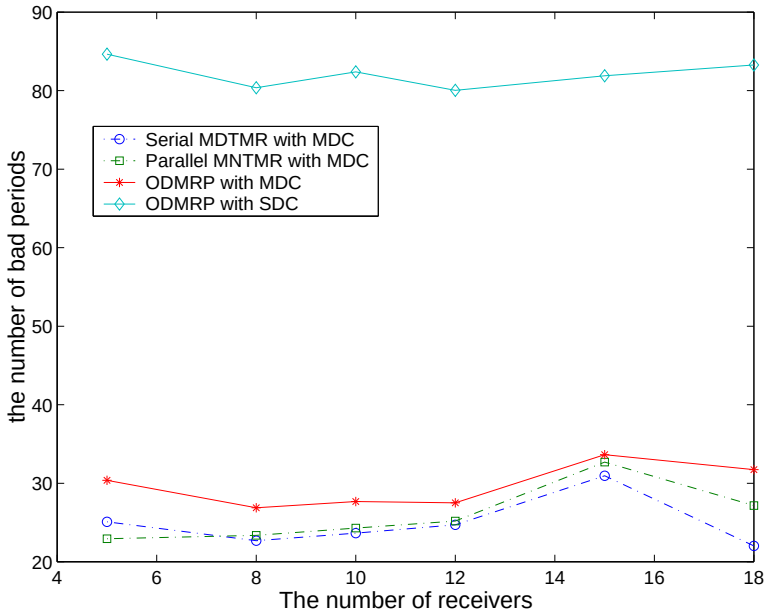


(b)

Figure 4.13. Performance evaluation for multiple tree video multicast Parallel MNTMR, Serial MDTMR and ODMRP with cross traffic: (a) The ratio of bad frames; (b)The number of bad periods.

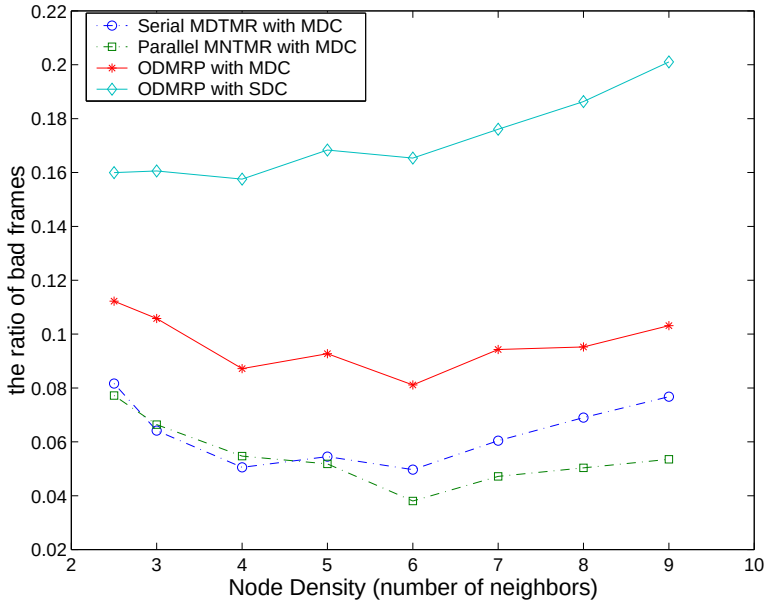


(a)

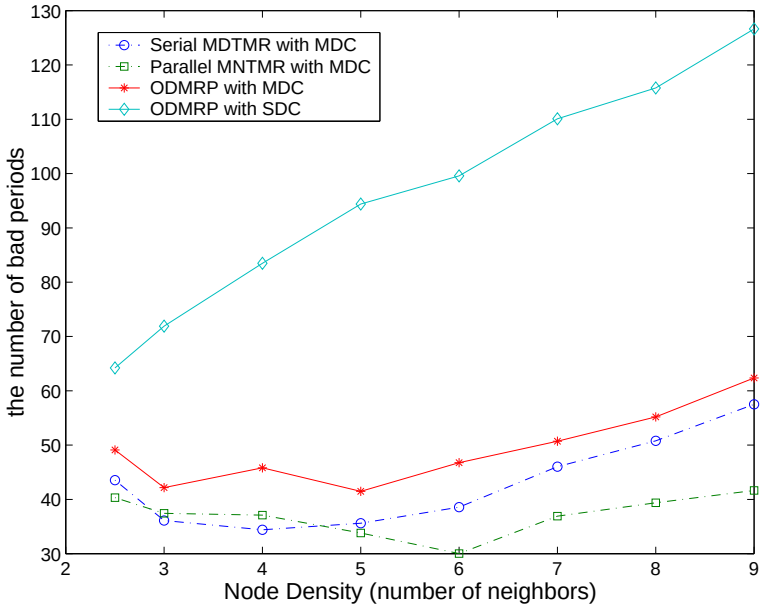


(b)

Figure 4.14. Performance evaluation for multiple tree video multicast Parallel MNTMR, Serial MDTMR and ODMRP with varying number of receivers: (a) The ratio of bad frames; (b)The number of bad periods.



(a)



(b)

Figure 4.15. Performance evaluation for multiple tree video multicast Parallel MNTMR, Serial MDTMR and ODMRP with varying node density: (a) The ratio of bad frames; (b)The number of bad periods.

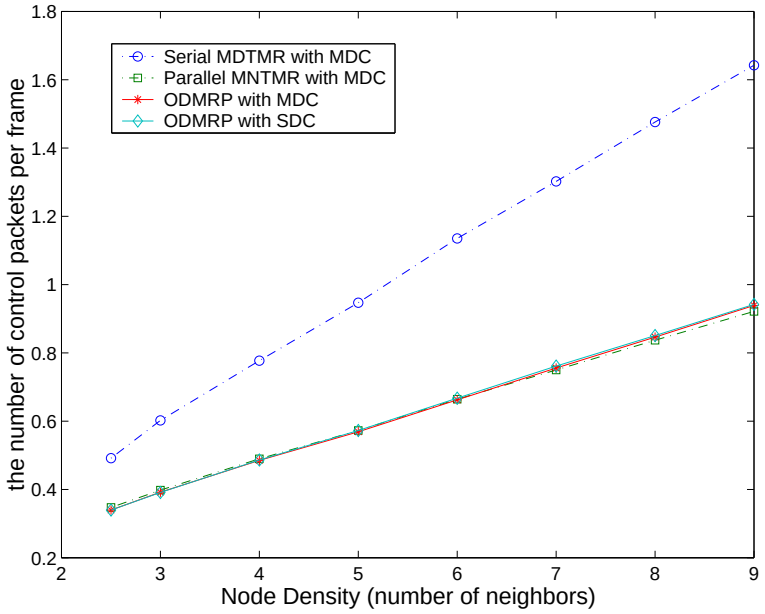
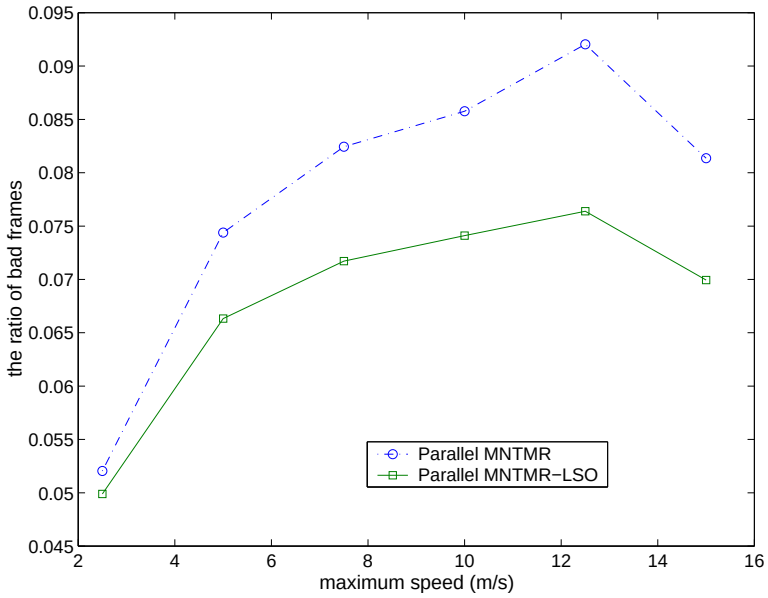
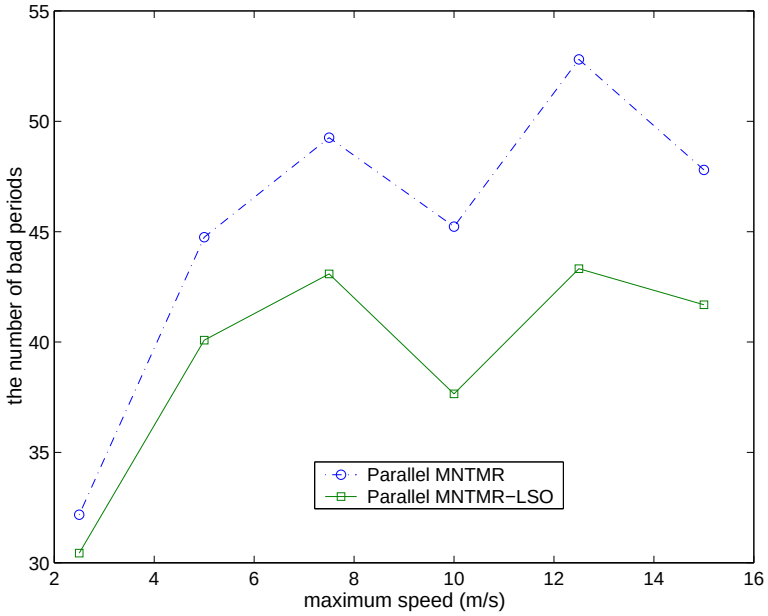


Figure 4.16. Performance evaluation for multiple tree protocols Parallel MNTMR, Serial MDTMR and ODMRP with varying node density: The normalized control packets.



(a)



(b)

Figure 4.17. Performance evaluation for multiple tree video multicast with Parallel MNTMR and Parallel MNTMR-LSO: (a) The ratio of bad frames; (b)The number of bad periods.

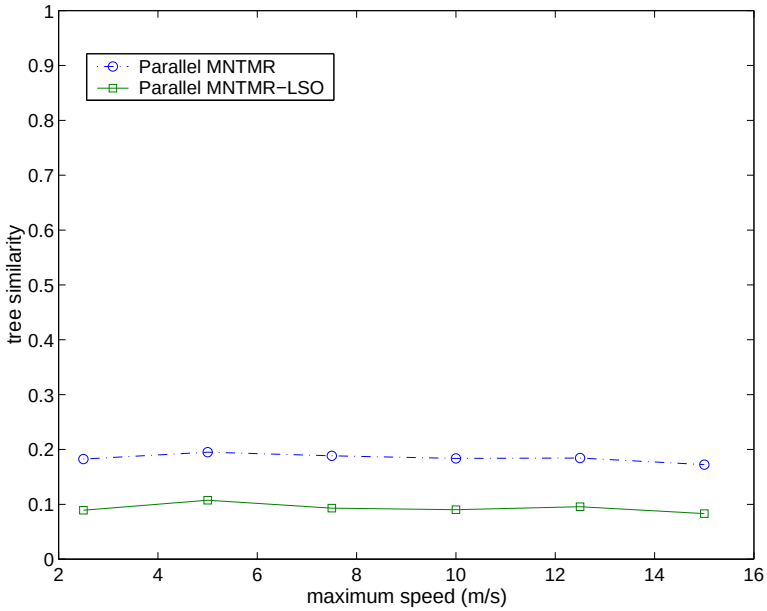
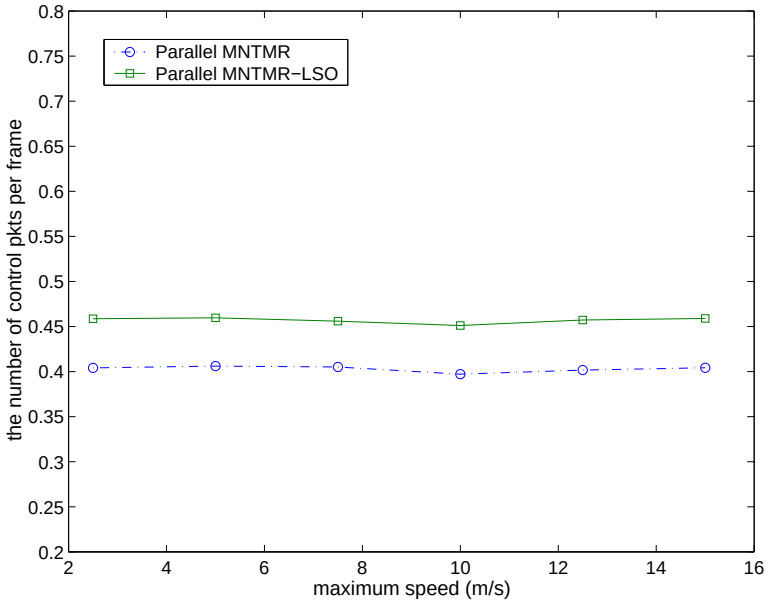
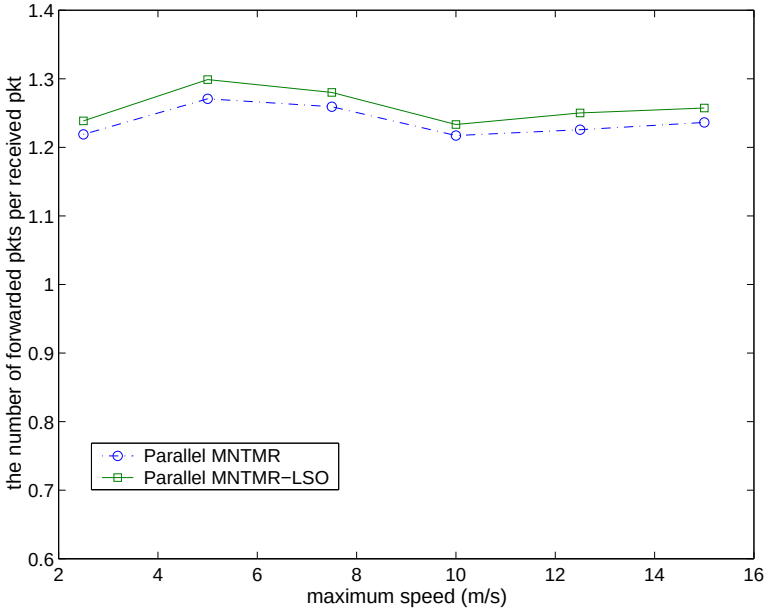


Figure 4.18. Tree Similarity of Parallel MNTMR and Parallel MNTMR-LSO

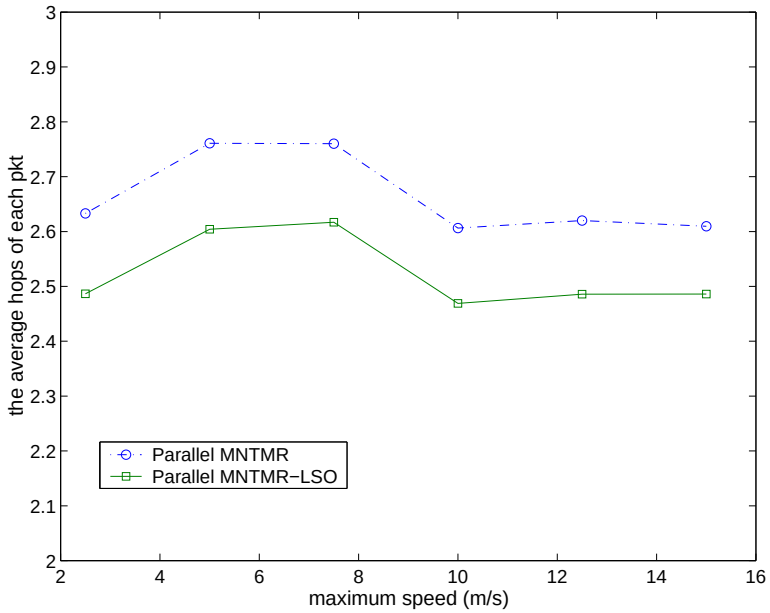


(a)



(b)

Figure 4.19. Performance evaluation for Parallel MNTMR and Parallel MNTMR-LSO: (a) The normalized control packets; (b) The normalized forwarded data packets.



(c)

Figure 4.20. Performance evaluation for Parallel MNTMR and Parallel MNTMR-LSO: The averaged number of hops.

Chapter 5

Summary and Future Work

5.1 Summary

In this dissertation, we have designed and developed a framework for multipath unicast and multicast video streaming over wireless ad hoc networks. Transmitting video streams through multiple paths combats unpredictable packet loss in wireless ad hoc networks. Given multiple paths, a video stream can be divided into different substreams and different substreams can be transmitted over different paths simultaneously. With careful design, each substream could experience relatively independent packet loss. The receiver could still get acceptable video quality with part of video packets received. Our proposed framework combines approaches from network routing protocols to video error control schemes in order to improve the quality of received video.

In Chapter 2, we have presented the general architecture of multipath video streaming over wireless ad hoc networks. There are four main components in the architecture: a video encoder with error control, a traffic allocator, a multipath unicast/multicast routing, and a rate control scheme. Within this framework, we pro-

pose a protocol named RMPSR to support multipath streaming. RMPSR builds and maintains multiple nearly disjoint route sets for video communication. RMPSR uses alternative sub-routes of each route set to salvage packets dropped in the middle of the network. We examine the performance of RMPSR through extensive NS simulations.

One main disadvantage of RMRSR is that it does not consider the interference between node disjoint paths. In Chapter 3, we propose a multipath routing protocol, which selects two node-disjoint paths with minimum concurrent packet drop probability of all path pairs. The proposed protocol both optimizes the worst case video quality of MDC streaming and improves performance of video streaming with FEC. We formulate the path selection problem as an optimization problem. We model the effects of interference between different wireless links, and estimate the concurrent packet drop probability of two node-disjoint paths, given an estimate of cross traffic flows' rates, and bit rate of the video flow. We show that the above optimization is an NP-hard problem. Then we propose one heuristic interference aware multipath routing protocol based on our path selection model. The proposed protocol obtains a path with approximately minimum packet drop probability as the first path. After updating all the link metrics, such as flow rate, of the network graph, the protocol finds the path with approximately minimum packet drop probability based on the new graph as the second path. The performance of the proposed scheme is shown to be close to that of the "optimal routing", and much better than that of the node-disjoint multipath routing and the shortest-widest routing through extensive NS simulations and real experiments.

In Chapter 4, we study the problem of multicast streaming. We first show that for a given level of tree connectivity, the difference between node density required for single and double tree schemes is small. Thus building multiple trees does not increase the cost of network deployment significantly. We then propose Serial MDTMR, which constructs two disjoint multicast trees sequentially in a distributed way, to facilitate

multiple tree video multicast. This scheme results in reasonable tree connectivity while maintaining disjointness of two trees. However Serial MDTMR has a larger routing overhead and construction delay than conventional single tree multicast routing protocols, as it constructs the trees in a sequential manner. To alleviate these drawbacks, we further propose Parallel MNTMR in which nearly disjoint trees are constructed in parallel, and in a distributed way. Using the Parallel MNTMR, each receiver is able to always connect to two trees, regardless of the node density. Simulations show that multiple tree video multicast with both Serial MDTMR and Parallel MNTMR improve video quality significantly compared to single tree video multicast; at the same time routing overhead and construction delay of Parallel MNTMR is approximately the same as that of a single tree multicast protocol.

5.2 Recommendations for Future Work

5.2.1 Multipath Streaming over Multiple Channel Wireless Ad Hoc Networks

Currently, we only select two paths with minimum concurrent packet drop probability in single channel wireless ad hoc networks. It would be interesting if we extend our framework to multiple channel wireless ad hoc networks. Each node in a multiple channel wireless ad hoc network is equipped with multiple 802.11 Network Interface Cards (NICs), each of which is tuned to a different non-overlapping radio channel. The NICs can be the same type, e.g. all 802.11g cards or all 802.11a cards, or different types, e.g. some 802.11g cards and some 802.11a cards. 802.11g and 802.11a have 3 and 8 non-overlapping channels respectively. For direct communication, two nodes need to have a common channel assigned to their interfaces and be within communication range of each other. Two links interfere with each other, if they use the same

channel and are within the interference range of each other. Two links using different channels can transmit packets simultaneously without interference.

Multiple channel wireless ad hoc networks have higher throughput compared to single channel wireless ad hoc networks. Packet drop over two links with different channels are independent, which suggests that multipath streaming in multiple channel wireless ad hoc networks can improve the performance more compared to multipath streaming in single channel wireless ad hoc networks.

The path selection problem in a multiple channel wireless ad hoc network is similar to that of a single channel wireless ad hoc network, except we need to redefine the condition under which two links interfere with each other. We use a triplet $(n_i, n_j; k_1)$ to represent a link in multiple channel wireless ad hoc networks, where n_i, n_j and k_1 are the source node, destination node and channel number of the link respectively.

Definition 5.1: Two links $(n_i, n_j; k_1)$ and $(n_x, n_y; k_2)$ interfere with each other, if

- $k_1 = k_2$;
- $d_{ix} \leq \omega$ or $d_{iy} \leq \omega$ or $d_{jx} \leq \omega$ or $d_{jy} \leq \omega$, where ω is the interference range.

The extension of the Optimal Multipath Selection Problem in a single channel wireless ad hoc network to a multiple channel wireless ad hoc network is straightforward. PDP of two paths can be computed by Equations (3.5), (3.6), (3.7) and (3.14), except a link's *interfering link set* is determined by Definition 5.1. Then OMR can be extended to a multiple channel wireless ad hoc network by enumerating all possible pairs of node-disjoint paths from a source N_S to a destination N_D, and choosing the best one based on PDP.

Compared to single channel wireless ad hoc networks, the search space of multiple channel wireless ad hoc networks has one more degree of freedom, i.e. channels.

Thus the OMR in multiple channel wireless ad hoc networks is significantly more complicated than that in single channel wireless ad hoc networks.

We propose Multiple Channel IWM (MC-IWM) for multiple channel wireless ad hoc networks. We apply a centralized approach. We assume that flow rate and packet loss probability of broadcast packets of each link, are distributed over the whole network periodically. Thus the sender knows both the topology of the network and characteristics of each link. In this case, the sender is able to compute the PDP given any two paths in the network. The basic idea behind MC-IWM is to select two paths which share as little common channels and nodes as possible and each path has a small PDP.

The optimization of finding the first path is formulated as follows.

$$\underset{x_{ij}}{\text{Minimize}} \sum_{l_{ij} \in E} x_{ij} P_{\text{drop}}(l_{ij}) + \text{ch\_ecost}$$

such that the constraint in Equation (3.1) is satisfied. $P_{\text{drop}}(l_{ij})$ as defined by Equation (3.6) denotes the PDP over link l_{ij}. $P_{\text{drop}}(l_{ij})$ is estimated through the procedure described in Section 3.1.6. $\text{ch\_ecost} = \max(0, n_{c1} - N_{c1}) \times a_1$ is a penalty factor for using extra channels, where n_{c1} is the number of channels used by the first path and N_{c1} is an offset. There is a trade off between the PDP and channel independence between two paths. If the first path uses more channels, generally the available bandwidth increases, thus reducing the PDP. However the second path has less number of unused channels to select, raising the probability it shares some channel with the first path. Thus the first path should neither use too few channels, causing a high PDP, nor too many channels, losing channel independence with the second path. We need to study how to set parameter a_1 to reach a good trade off for different network scenarios. We obtain the first path by solving the above OSPF-like Weighted Path Cost routing problem using the Dijkstra's algorithm.

Given the first path, for computing the second path, we define a link cost as follows:

$$C_{mn} = P_{\text{drop}}(l_{mn}) + \text{nd\_cost} + \text{Int\_cost} \tag{5.1}$$

where

$$\text{nd\_cost} = \begin{cases} b_1 \gg 1 & \text{destination node of link } l_{mn} \text{ in path 1} \\ 0 & \text{otherwise} \end{cases}$$

is a penalty factor to maintain the node-disjointness between two paths.

$$\text{Int\_cost} = \begin{cases} b_2 \gg 1 & \text{the channel used by link } l_{mn} \text{ is also used by path 1} \\ 0 & \text{otherwise} \end{cases}$$

is a penalty factor to maintain channel independence between two paths.

The optimization problem to find the second path is formulated as follows:

$$\underset{y_{mn}}{\text{Minimize}} \sum_{l_{mn} \in E} y_{mn} C_{mn}$$

such that the constraints in Equation (3.2) are satisfied. We also apply the Dijkstra's algorithm to solve this optimization problem.

In practice, there are a few difficulties of implementing MC-IWM in an actual multiple channel wireless ad hoc network.

- Case 1: A node's NICs are with the same type, i.e. all 802.11a or all 802.11g. In this case, two NICs interfere with each other even if they are assigned to non-overlapping channels[13]. This may be caused by the proximity of two NICs. We need to study how to overcome this when implementing MC-IWM.

- Case 2: A node's NICs are with different types, i.e. some 802.11a and some 802.11g. Since in ad hoc mode, a 802.11a NIC has much larger bandwidth than a 802.11g NIC, the MC-IWM will select channels with 802.11a for path 1 and channels with 802.11g for path 2, causing very unbalanced paths. We need to improve and test MC-IWM to address this issue in practice.

5.2.2 Multipath Streaming over Wireless Ad Hoc Networks with Advanced Technologies

Optimum path selection problem in wireless ad hoc networks with other advanced techniques, such as directional antennas, multi-input multi-output (MIMO) systems (802.11n), is another interesting research topic.

A directional antenna is an antenna which transmits or receives maximum power in a particular direction. Directional antennas have a number of advantages over omnidirectional antennas in ad hoc networking. By focusing energy only in the intended direction, directional antennas can increase the potential for spatial reuse and provide longer transmission range for the same amount of power, which increases capacity of an ad hoc network. In order to fully utilize the potential of directional antenna, MAC layer and routing protocols of ad hoc networks need to redesigned. It would be interesting to see how to modify these protocols in order to better support video applications.

The incoming standard 802.11n builds upon MIMO technique, which uses multiple transmitter and receiver antennas to allow for increased data throughput through spatial multiplexing and increased range by exploiting the spatial diversity. It is claimed that the throughput of 802.11n can be up to 600 Mbps. 802.11n might be the future of Wi-Fi, so it is important to study video streaming, possibly multipath streaming, in this kind of networks.

Video streaming over heterogeneous wireless multi-hop networks is another interesting research problem. In a heterogeneous wireless multi-hop network, devices apply different wireless techniques, thus having drastically different bandwidth, transmission range, power usage, and et al. It is very possible, in one wireless network, different wireless techniques, such as WiMax, Wi-Fi (802.11 series), and Ultra-Wideband net-

work (UWN), coexist with each other. For example, in a multi-hop wireless network covering the entire Bay Area, computers in the city, such as San Francisco, connect with each other using Wi-Fi; computers in the suburbs are connected using WiMax; computers or PDAs connect with the display device through UWN. A video program streamed from a server in SF to a client in Berkeley, is first transmitted using Wi-Fi, then WiMax, and finally UWN. How to optimize the experience of video applications in these heterogeneous wireless multi-hop networks is an interesting problem to explore.

5.2.3 Scalability Issue in Wireless Ad Hoc Networks

MAC layer design for wireless mesh networks is an important research area. The MAC protocol has to be distributed, cooperative, taking into account multi-hop communication and mobility. To the best of my knowledge, the scalability issue of multi-hop wireless networks has not been fully solved yet, i.e. with the increase of the size of the network, the end-to-end throughput drops drastically. How to fundamentally improve the scalability of IEEE 802.11 MAC, i.e. CSMA/CA, is a challenging task. Besides CSMA/CA, distributed schemes based on TDMA or CDMA might be promising to improve throughput and scalability of wireless mesh networks. MAC protocol design for multi-channel wireless mesh networks are also interesting research problems. In order to support multimedia communication better, the development of MAC protocols with multiple QoS metrics such as packet loss rate, delay, and delay jitter is an important topic.

It has been many years since the development of wireless ad hoc networks, wireless sensor networks and wireless mesh networks. However the deployment of these kinds of wireless networks is slow in practice. One important reason is lack of killer applications for mass users. Until recently, the driving application was instant deployment

in an unfriendly, remote infrastructure-less area, such as battle field, Mars exploration, disaster recovery and et al. Thus finding, designing and deploying interesting applications is also an interesting and important problem.

Bibliography

[1] P. Mohapatra, and S. Krishnamurthy, "Ad Hoc Networks: technologies and protocols", Spinger Science+Business Media, Inc., 2004.

[2] c. Perkins and P. Bhagwat. "Highly Dynamic Destination-Sequenced Distance-Vector Routing (DSDV) for Mobile Computers." *In Proceddings of ACM SIGCOMM*, pp. 234-244, 1994.

[3] T. Clausen and P. Jacquet. "Optimized Link State Routing Protocol (OLSR)." *IETF Internet RFC3626*, http://www.ietf.org/rfc/rfc3626.txt, 2003.

[4] J. Moy, "OSPF Version 2", *RFC 1583*, March 1994.

[5] D. Johnson and D. Maltz. "Dynamic Source Routing in Ad Hoc Wireless Networks." *Mobile Computing*, Chapter 5. Kluwer Academic, 1996.

[6] J. Broch, D.A. Maltz, D.B. Johnson, et al. "A performance comparison of multihop wireless ad hoc network routing protocols", *ACM MobiCom 1998*, pp.85-97.

[7] C. Perkins and E. Royer. "Ad Hoc On-Demand Distance Vector Routing." *IEEE Workshop on Mobile Computing Systems and Applications,* pp. 90-100, 1999.

[8] Z. Haas and M. Pearlman. "The Performance of Query Control Schemes for the Zone Routing Protocol." *IEEE/ACM Transactions on Networking,* 9(4):427-438, 2001.

[9] Y. Ko and N. Vaidya. "Location-Aided Routing (LAR) in Mobile Ad Hoc Networks". *IEEE/ACM Mobicom*, pp. 66-75, 1998.

[10] B. Karp and H. Kung. "GPSR: Greedy Perimeter Stateless Routing for Wireless Networks." *IEEE/ACM Mobicom*, pp. 243-254, 2000.

[11] D. De Couto, D. Aguayo, J. Bicket, and et al, "High-throughput path metric for multi-hop wireless routing". *ACM Mobicom*, 2003.

[12] R. Draves, J. Padhye, and B. Zill, "Comparison of Routing Metrics for Static Multi-hop Wireless Networks", *ACM Sigcomm'04*, Portland, August 2004.

[13] R. Draves, J. Padhye, and B. Zill, "Routing in Multi-Radio, Multi-Hop Wireless Mesh Networks", *ACM MobiCom'04*, Philadelphia, September 2004.

[14] S.J. Lee and M. Gerla, "Split multipath routing with maximally disjoint paths in ad hoc networks," *ICC 2001*, pp.3201-3205.

[15] P. Pham and S. Perreau, "Performance Analysis of Reactive Shortest Path and Multi-path Routing Mechanism With Load Balance", *IEEE Infocom 2003*.

[16] A. Nasipuri and S.R. Das, "On-demand multipath routing for mobile ad hoc networks," *Proceedings of the IEEE International Conference on Computer Communication and Networks (ICCCN'99)*, Boston, October, 1999, pp.64-70.

[17] K. Wu and J. Harms, "On-demand multipath routing for mobile ad hoc networks," *Proceedings of 4th European Personal Mobile Communication Conference (EPMCC 01)*, Vienna, Austria, Feb. 2001, pp.1-7.

[18] M. Marina and S. Das, "On-demand Multipath Distance Vector Routing in Ad Hoc Networks." *International Conference for Network Protocols*, 2001.

[19] K. Obraczka, and G. Tsuduk. "Multicast routing issues in ad hoc networks", *IEEE International Conference on Universal Personal Communications*, 1998.

[20] S. Mueller, R. Tsang, and D. Ghosal. "Multipath Routing in Mobile Ad Hoc Networks: Issues and Challenges", *Lecture Notes in Computer Science*, Edited by M. Calzarossa and E. Gelenbe, 2004.

[21] P. Papadimitratos, Z.J. Haas, and E.G. Sirer, "Path Set Selection in Mobile Ad Hoc Networks," ACM Mobihoc 2002, Lausanne, Switzerland, June 9-11, 2002.

[22] S. Lee, M. Gerla, and C. Chiang. "On-Demand Multicast Routing Protocol", *Proceedings of IEEE WCNC'99*, New Orleans, LA, Sep. 1999, pp. 1298-1302.

[23] S. Lee, W. Su, et al. "A Performance Comparison Study of Ad Hoc Wireless Multicast Protocols", *Proceedings of IEEE INFOCOM 2000*, Tel Aviv, Israel, Mar. 2000.

[24] J. Jetcheva and D. Johnson. "Adaptive Demand-Driven Multicast Routing in Multi-hop Wireless Ad Hoc Networks", ACM Mobihoc 2001.

[25] S. Park, and D. Park. "Adaptive core multicast routing protocol", *Wireless Networks 2004.*

[26] P. Sinha, R. Sivakumar, and V. Bharghavan. "MCEDAR: multicast core extraction distributed ad hoc routing", *Proceedings of IEEE WCNC'99*, New Orleans, LA, Sep. 1999, pp. 1313-1317.

[27] L. Ji, and M. Corson. "Differential Destination Multicast - A MANET multicast routing protocol for small groups", *Proceedings of IEEE Infocom'01*, 2001, pp. 1192-1202.

[28] J. Xie, R. Talpade, et al. "AMRoute: ad hoc multicast routing protocol", *Mobile Networks and Applications*, v.7 n.6, p.429-439, December 2002.

[29] S. Sajama, and Z. Haas. "Independent-tree ad hoc multicast routing (ITAMAR)", *ACM Mobile Networks and Applications*, Oct. 2003.

[30] C. Toh, G. Guichala, and S. Bunchua. "ABAM: OnDemand Associativity-Based Multicast Routing for Ad Hoc Mobile Networks", *In Proceedings of IEEE Vehicular Technology Conference, VTC 2000*, pages 987–993, September 2000.

[31] C. Wu and Y. Tay. "AMRIS: A multicast protocol for ad hoc wireless networks", *in Military Communications Conference Proceedings. 1999*, vol. pp. 25-29. New York: IEEE, 1999.

[32] Y. Wang and Q. F. Zhu. "Error control and concealment for video communication: a review", *in Proc. IEEE*, vol.86, issue 5, pp.974-997, May 1998.

[33] J. Apostolopoulos, "Reliable video communication over lossy packet networks using multiple state encoding and path diversity", *Visual Communications and Image Processing (VCIP)* 2001.

[34] S. Mao, S. Lin, S. Panwar, and etc. "video transport over ad hoc networks: multistream coding with multipath transport", *IEEE Journal on Selected Areas in Communications*, Dec. 2003, pp. 1721 - 1737.

[35] T. Nguyen and A. Zakhor. "Multiple Sender Distributed Video Streaming", *IEEE Transactions on Multimedia*, Vol. 6, No. 2, April 2004, pp. 315 - 326.

[36] E. Setton, X. Zhu and B. Girod. "Congestion-optimized multi-path streaming of video over ad hoc wireless networks", *International Conference on Multimedia and Expo (ICME) 2004*.

[37] T. Nguyen and A. Zakhor, "distributed video streaming over internet", *in Multimedia Computing and Networking 2002, Proceedings of SPIE*, San Jose, California, January 2002, Vol. 4673, pp. 186-195.

[38] T. Nguyen, and A. Zakhor, "distributed video streaming with forward error correction", *Packet Video 2002*, Pittsburgh, April 2002.

[39] T. Nguyen, and A. Zakhor, "Path Diversity with Forward Error Correction (PDF) System for Packet Switched Networks", *INFOCOM 2003*, April 1-5, San Francisco CA, USA.

[40] J. Apostolopoulos and S.J. Wee, "Unbalanced Multiple Description Video Communication Using Path Diversity", *International Conference on Image Processing (ICIP)*, October 2001.

[41] J. Apostolopoulos, W. Tan, S.J. Wee, G.W. Wornell. "Modeling Path Diversity for Multiple Description Video Communication", *IEEE Inter. Conf. on Acoustics, Speech, and Signal Processing (ICASSP)*, May 2002.

[42] J. Apostolopoulos, T. Wong, W. Tan, and S. Wee, "On multiple description streaming in content delivery networks", *IEEE INFOCOM*, 2002, pp. 1736-1745.

[43] A. Begen, Y. Altunbasak, and O. Ergun. "Multi-Path Selection for Multiple Description Encoded Video Streaming", *IEEE Internationl Conference on Communications (ICC)*, May 2003.

[44] A. Begen, Y. Altunbasak, and O. Ergun. "Fast heuristics for multi-path selection for multiple description encoded video streaming," *in Proc. IEEE Int. Conf. Multimedia and Expo (ICME)*, Baltimore, MD, July 2003.

[45] S. Mao, Y. Hou, X. Cheng, H. Sherali, and S. Midkiff. "Multipath Routing for Multiple Description Video in Wireless Ad Hoc Networks", *IEEE Infocom*, March 2005.

[46] Z. Ma, H. Shao and C. Shen. "A New Multi-path Selection Scheme for Video Streaming on Overlay Netwoks", *IEEE Internationl Conference on Communications (ICC)*, June 2003.

[47] J. Chen, S. Chan, and V. Li. "Multipath routing for video delivery over bandwidth-limited networks", *IEEE Journal on Selected Areas in Communications*, Vol. 22, No. 10, pp. 1920 - 1932, December 2004.

[48] J. H. Kim, R. M. Mersereau, and Y. Altunbasak, "Network-adaptive video streaming using multiple description coding and path diversity," *in Proc. of IEEE Intl. Conf. on Multimedia and Expo (ICME)*, Baltimore, MD, July 2003.

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

[50] M. Bansal and A. Zakhor, "Path Diversity for Overlay Multicast Streaming", *Packet Video Workshop*, Irvine, CA, Dec. 2004

[51] M. Bansal and A. Zakhor, "Path Diversity Based Techniques for Resilient Overlay Multimedia Multicast", *Picture Coding Symposium (PCS)*, San Francisco, CA, Dec. 2004.

[52] S. Mao, S. Lin, S. Panwar, and Y. Wang, "reliable transmission of video over ad-hoc networks using automatic repeat request and multi-path transport", *Proceedings of the 2001 Fall IEEE Vehicular Technology Conference*, Atlantic City, NJ, October 2001, pp.615-619.

[53] S. Lin, Y. Wang, S. Mao, and S. Panwar. "video transport over ad-hoc networks using multiple paths", *invited paper in the Proceedings of the 2002 IEEE International Symposium on Circuits and Systems*, Scottsdale, Arizona, May 2002, pp. 57-60.

[54] S. Mao, S. Lin, D. Bushmitch, S. Narayanan, S. Panwar, Y. Wang, and R. Izmailov, "Real time transport with path diversity," *the Second New York Metro Area Networking Workshop (NYMAN)*, New York, NY September 3, 2002.

[55] Y. Wang, S. Panwar, S. Lin, and S. Mao, "Wireless video transport using path diversity: Multiple description vs. layered coding," *in Proceedings of the 2002 IEEE International Conference on Image Processing (ICIP)*, pp.21-24, Rochester, NY, September 2002.

[56] S. Mao, X. Cheng, Y. Hou, and H. Sherali, "Multiple description video multicast in wireless ad hoc networks," *in Proceedings of the First International Conference on Broadband Netowrks (BROADNETS 2004)*, pp.671-680, San Jose, CA, October 25-29, 2004.

[57] Y. J. Liang, E. Setton, and B. Girod, "Channel-Adaptive Video Streaming Using Packet Path Diversity and Rate-Distortion Optimized Reference Picture Selection," *Proceedings IEEE Fifth Workshop on Multimedia Signal Processing*, St. Thomas, Virgin Island, Dec. 2002.

[58] X. Zhu, S. Han and B. Girod, "Congestion-Aware Rate Allocation For Multipath Video Streaming Over Ad Hoc Wireless Networks", *IEEE International Conference on Image Processing*, vol. 4, pp. 2547-2550, Singapore, October 2004.

[59] E. Setton, X. Zhu and B. Girod, "Minimizing Distortion for Multipath Video Streaming over ad hoc Networks", *IEEE International Conference on Image Processing*, vol. 3, pp. 1751-1754, Singapore, Octobor 2004.

[60] X. Zhu and B. Girod, "A Distributed Algorithm for Congestion-Minimized Multi-Path Routing over Ad Hoc Networks", *IEEE International Conference on Multimedia and Expo*, pp. 1484 - 1487, Amsterdam, The Netherlands, July 2005.

[61] J. Chakareski, S. Han, and B. Girod, "Layered coding vs. multiple descriptions for video streaming over multiple paths," *Multimedia Systems*, Springer, online journal publication, January 2005.

[62] H. Man and Y. Li, "A multipath video delivery scheme over diffserv wireless LANs", *Proceedigns of SPIE*, pp. 1148-1158, 2004.

[63] A. Miu, J.G. Apostolopoulos, W. Tan, and M. Trott, "Low-latency Wireless Video over 802.11 Networks using Path Diversity", *IEEE International Conference on Multimedia and Expo (ICME)*, July 2003.

[64] A. Miu, G. Tan, H. Balakrishnan, J.G. Apostolopoulos, "Divert: Fine-grained Path Selection for Wireless LANs", *ACM MobiSys*, Boston, MA, June 2004.

[65] X. Tang, and A. Zakhor, "Matching pursuits multiple description coding for wireless video," *IEEE Trans. on Circuits and Systems for Video Technology*, pp. 566 -575, June 2002.

[66] R. Neff and A. Zakhor. "Very Low Bit-Rate Video Coding based on Matching Pursuits", *IEEE Transactions on Circuits and Systems for Video Technology*, February 1997, vol. 7, no. 1, pp. 158-171.

[67] Y. Wang, and S. Lin, "Error-resilient video coding using multiple description motion compensation," *IEEE Trans. on Circuits and Systems for Video Technology*, pp. 438-452, June 2002.

[68] V. Goyal, "Multiple description coding: compression meets the network", *IEEE Signal Processing Magazine*, Vol. 18, No. 5, Sept. 2001, pp. 74 - 93.

[69] L. Rizzo, "Effective erasure codes for reliable computer communication protocols",*Computer Communication Review*, vol. 27, no. 2, pp. 24-36, April 1997.

[70] S. Mao. "Real-time Multimedia Transport using Multiple Paths", *Ph.D. Dissertation*, Polytechnic University, NY, January 2004.

[83] D. Chiu and R. Jain, "Analysis of the Increase/Decrease Algorithms for Congestion Avoidance in Computer Networks", *Journal of Computer Networks and ISDN Systems*, Vol. 17, No. 1, June 1989, pp. 1-14.

[72] J. Broch, D.A. Maltz, D.B. Johnson, et al. "A performance comparison of multihop wireless ad hoc network routing protocols", *ACM MobiCom 1998*, pp.85-97.

[73] D B. Johnson, D A. Maltz, and Y. Hu, "The dynamic source routing protocol for mobile ad hoc network," Internet-Draft, draft-ietf-manet-dsr-09.txt, April 2003.

[74] W. Wei, and A. Zakhor, "Robust Multipath Source Routing Protocol (RMPSR) for Video Communication over Wireless Ad Hoc Networks", *International Conference on Multimedia and Expo (ICME) 2004*.

[75] S. Blake, D. Black, M. Carlson, et al. "An Architecture for Differentiated Services", RFC 2475, December 1998.

[76] ns-2: network simulator. http://www.isi.edu/nsnam/ns/

[77] H. Luo, S. Lu, and V. Bhargavan, "A New Model for Packet Scheduling in Multihop Wireless Networks", *ACM Mobicom*, 2000, pp. 76-86.

[78] K. Jain, J. Padhye, V. Padmanabhan, and L. Qiu, "Impact of Interference on Multi-hop Wireless Network Performance", *ACM MobiCom*, Sep. 2003.

[79] R. Gupta, J. Musacchio, and J. Walrand, "Sufficient Rate Constraints for QoS Flows in Ad-Hoc Networks", *under revision at Ad-Hoc Networks Journal*

[80] A. Bruce McDonald and T.F. Znabi, "A path availability model for wireless ad hoc networks", Proc. IEEE WCNC, New Orlean, USA, (Sept. 1999), pp.35-40.

[81] S. Jiang, D. He, and J. Rao, "A Prediction-based Link Availability Estimation for Mobile Ad Hoc Networks", IEEE Infocom 2001.

[82] R. Ogier, M. Lewis, and F. Templin. "Topology Dissemination Based on Reverse-Path Forwarding (TBRPF). *IETF Internet RFC 3684*, http://www.ietf.org/rfc/rfc3684.txt.

[83] D. M. Chiu and R. Jain, "Analysis of the Increase and Decrease Algorithms for Congestion Avoidance in Computer Networks", *Computer Networks and ISDN Systems*, Vol. 17, 1989, pp. 1-14.

[84] G.Bianchi, "Performance Analysis of the IEEE 802.11 Distributed Coordination Function", *IEEE Journal on Selected Areas in Communicatioins"*, vol. 18, no. 3, March 2000.

[85] Z. Jia, R. Gupta, J. Walrand, and P. Varaiya, "Bandwidth guaranteed routing for ad-hoc networks with interference consideration", ISCC 2005.

[86] W. Wei and A. Zakhor, "Multipath Unicast and Multicast Video Communication over Wireless Ad Hoc Networks", *International Conference on Broadband Networks (Broadnets) 2004*, San Jose, CA, USA, Oct. 2004 *(invited)*, pp. 496 - 505.

[87] T. Philips, S. Panwar, A. Tantawi. "Connectivity Properties of a Packet Radio Network Model", *IEEE Trans. On Information Theory*, Sep. 1989

[88] P. Gupta, and P. Kumar. "Critical Power for Asymptotic Connectivity in Wireless Networks". *Stochastic Analysis, Control, Optimization and Applications: A Volume in Honor of W.H. Fleming* 1998

[89] O. Dousse, P. Thiran and M. Hasler, "Connectivity in ad-hoc and hybrid networks", *IEEE Infocom 2002*

[90] D. Yu, and H. Li. "On the Definition of Ad Hoc Network Connectivity", *ICCT 2003*

[91] L. Shu, M. Desai, and R. Mangoubi. "Average Connectivity Properties of Wireless Ad Hoc Networks." *WCNC 2003*

[92] W. Wei and A. Zakhor, "Connectivity for Multiple Multicast Trees in Ad Hoc Networks", in International Workshop on Wireless Ad-hoc Networks (IWWAN), Oulu, Finland, June 2004.

[93] R. Meester and R.Roy. "Continuum percolation", *Cambridge University Press, Cambridge*, 1996

[94] M. Krunz, A. Muqattash, and S. Lee. "Transmission power control in wireless ad hoc networks: challenges, solutions, and open issues", *IEEE Network*, vol. 18, no. 5, Oct. 2004, pp. 8-14.

[95] J. Gomez, A. Campbell, et al. "PARO: Supporting Dynamic Power Controlled Routing in Wireless Ad Hoc Networks", ACM/Kluwer Journal on Wireless Networks (WINET), vol. 9, no. 5, 2003, pp. 443-460.

[96] J. Monks, V. Bharghavan, W. Hwu, "A power controlled multiple access protocol for wireless packet networks", IEEE Infocom 2001, pp. 219-228.

[97] A. Muqattash, M. Krunz, "Power controlled dual channel (PCDC) medium access protocol for wireless ad hoc networks", IEEE Infocom 2003, pp. 470-480.

[98] W. Cui, I. Stoica, and R. Katz, "Backup Path Allocation Based On a Correlated Link Failure Probability Model in Overlay Networks", *ICNP* 2002.

[99] M. Garey, and D. Johnson, "Computers and intractability: A guide to the theory of NP-Completeness", W. H. Freeman Company, 1979.

[100] T. Ballardie, P. Francis, and J. Crowcroft. "Core based trees (CBT) an architecture for scalable inter-domain multicast routing", *Computer Communication Review* Oct. 1993

[101] P. Gupta, and P. R. Kumar, "The Capacity of Wireless Networks", *IEEE Trans. on Information Theory*, Vol. 34, No. 5, pp. 910-917, 2000.

Appendix A

Proofs for Chapter 3

This appendix contains the proofs for Chapter 3.

A.1 Proof of Claim 3.1 in Section 3.1.2: The NP-hardness of the Optimal Path Selection Problem

*Claim 3.1:*The optimal multipath selection over wireless ad hoc networks defined by Equations (3.1), (3.2) and (3.3) is NP-hard.

Let A_i denote the event that $P^i_{S,D}$ does not drop packets, for $i = 1, 2$, and L_{ij} denote the event that link (i, j) does not drop packets.

$$\begin{aligned} P_{\text{drop}}(P^1_{S,D}; P^2_{S,D}) &= P(\overline{A_1 \bigcup A_2}) \\ &= P(\overline{(\bigcap_{(i,j)\in L^1_{S,D}} L_{ij}) \bigcup (\bigcap_{(m,n)\in L^2_{S,D}} L_{mn})}) \\ &= P(\bigcup_{(i,j)\in L^1_{S,D},(m,n)\in L^2_{S,D}} (\overline{L}_{ij} \bigcap \overline{L}_{mn})) \end{aligned} \tag{A.1}$$

By assuming that the packets drop probability of each link is small, we can ap-

proximate $P_{\text{drop}}(P^1_{S,D}; P^2_{S,D})$ as follows:

$$P_{\text{drop}}(P^1_{S,D}; P^2_{S,D}) \approx \sum_{(i,j)\in L^1_{S,D}} \sum_{(m,n)\in L^2_{S,D}} P(\overline{L}_{ij}, \overline{L}_{mn}) \tag{A.2}$$

The simplified optimization problem, which uses Equation (A.2) as its metrics and keeps all the constraints of the original optimization problem in Equations (1-3), has been shown to be an Integer Quadratic Programming problem [46][98], which is a known NP-hard problem as proved in [99].

Appendix B

Proofs for Chapter 4

This appendix contains all the proofs for Chapter 4.

B.1 Proof of Theorem 4.1 in Section 4.2.2: The form of tree connectivity level for a single tree scheme

Theorem 4.1: The tree connectivity level for a single tree scheme is given by

$$P_1 \approx \theta_1^2 \tag{B.1}$$

where θ_1 denotes the fraction of nodes belonging to the super-cluster in the single tree case.

We use S to denote the sender, R_i to denote Receiver i, $i = 1, \ldots, m$, $X(S)$ to denote the number of nodes of the cluster to which the sender belongs, and H to denote the set of nodes of the infinite super-cluster. By the definition of tree

connectivity level, we get:

$$P_1 = E[\frac{N}{m}]$$

where N denotes the number of receivers connecting to the tree, and $M = m$ for single tree schemes. By the property of conditional expectation, we further get

$$P_1 = \frac{1}{m}\{\theta_1 \times E(N|S \in H) + (1 - \theta_1) \times E(N|S \notin H)\} \tag{B.2}$$

where θ_1 denotes the fraction of nodes belonging to the super-cluster in the single tree case. Under the assumption of a dense ad hoc network, it has been shown that $\theta_1 \approx 1 - \exp(-\pi D_1 r^2)$ [93], where D_1 is node density of the network.

Using the definition of $N = \sum_{i=1}^{m} n_i$, we get:

$$\begin{aligned} E(N|S \in H) &= E(\sum_{i=1}^{m} n_i | S \in H) \\ &= m\theta_1 \end{aligned} \tag{B.3}$$

We now show that $E[N|S \notin H] \to 0$ as $n \to \infty$. Using the definition of conditional expectation, we get:

$$E(N|S \notin H) = \sum_{i=1}^{m} iP(N = i|S \notin H) \tag{B.4}$$

Besides the infinite size supercluster H, there are infinite number of finite size small clusters in the network. Let k denote the number of nodes of the largest finite cluster, and $X(S)$ denote the number of nodes of the cluster $C(S)$ to which the sender S belongs. Note that when $X(S) \leq i$, at most there are $i - 1$ receivers belonging to $C(S)$, thus $P(N = i|S \notin H, X(S) \leq i) = 0$. Then Equation (B.4) can be further expressed as

$$E(N|S \notin H) = \sum_{i=1}^{m}\{i \sum_{j=i+1}^{k} [P(N = i|S \notin H, X(S) = j) \cdot P(X(S) = j)]\} \tag{B.5}$$

Also from [93], and the fact that $k \ll D_1$, we have

$$\begin{aligned} P(X(S)=j) &= \exp\left[-4\pi r^2 D_1 + (j-1)\ln(\frac{D_1}{j-1})\right] \\ &< \exp\left[-4\pi r^2 D_1 + (k-1)\ln(\frac{D_1}{k-1})\right] \\ &= P(X(S)=k) \end{aligned} \tag{B.6}$$

Taking into account Equation (B.6), Equation (B.5) can be simplified as

$$E(N|S \notin H) < \sum_{i=1}^{m}[i \sum_{j=i+1}^{k} P(N=i|S \notin H, X(S)=j)] \cdot P(X(S)=k) \tag{B.7}$$

It is easy to show that

$$P(N=1|S \notin H, X(S)=j) \approx \frac{m(j-1)}{n} \tag{B.8}$$

and

$$2P(N=2|S \notin H, X(S)=j) \approx \frac{m(m-1)(j-1)^2}{n^2} \tag{B.9}$$

given that $j \leq k \ll n$. By assumption 3, $m \ll n$, and comparing the right hand side of Equations (B.8) and (B.9), we get:

$$2P(N=2|S \notin H, X(S)=j) \ll P(N=1|S \notin H, X(S)=j) \tag{B.10}$$

From (B.10) and by induction, we get:

$$(i+1)\sum_{j=i+2}^{k} P(N=i+1|S \notin H, X(S)=j) \ll i \sum_{j=i+1}^{k} P(N=i|S \notin H, X(S)=j) \tag{B.11}$$

where $i = 1, 2, \ldots, m-1$.

Considering Equation (B.11) and (B.7), we get:

$$E(N|S \notin H) < m\sum_{j=2}^{k} P(N=1|S \notin H, X(S)=j) \cdot P(X(S)=k)$$

Substituting Equation (B.8), we get:

$$E(N|S \notin H) < m\sum_{j=2}^{k} \frac{m(j-1)}{n} \cdot P(X(S)=k) \tag{B.12}$$

By assumption, $m \ll n$ and $k \ll n$, and $D_1 \to \infty$ in order to keep connectivity as $n \to \infty$ [88]. Therefore, using Equation (B.6), we conclude that $P(X(S) = k) \to 0$ as $D_1 \to \infty$. From Equation (B.12), we obtain

$$E(N|S \notin H) \to 0 \tag{B.13}$$

as $n \to \infty$.

By the dense network assumption, $\theta_1 \approx 1$. Taking into account Equations (B.2), (B.3), and (B.13), we get:

$$P_1 \approx \theta_1^2 \tag{B.14}$$

■

B.2 Proof of Claim 4.1 in Section 4.2.3

Claim 4.1: Define receiver sets $A = \{R : \text{Ne(R)} \geq 2, R \in H\}$ and $B = \{R : R \in H'\}$, where Ne(R) denotes the number of receiver R's neighbors, H and H' denote the connected infinite supercluster and the one after removing all the middle nodes of the first tree respectively. Then there exists at least one tree scheme that makes $A \approx B$.

It is easy to show that $B \subseteq A$. We only need to show that $A \subseteq B$ approximately. Without loss of generality, we assume that the sender belongs to set H' and is well connected. A well connected node has so many neighbors that losing a few of them does not affect its connectivity status. If not, we can select a well connected node belonging to set H', and build a core based tree[100].

We construct the first tree in such a way that each receiver belonging to set A is only a leaf node, which means that each receiver only connects to one neighbor in the first tree. We define $K =$ { middle nodes of the first tree } Now we want to

show there is only approximately one neighbor of each receiver belonging to set K. We pick an arbitrary receiver R belonging to set A. By assumption 3 in Chapter 4, the distribution of receivers is scattered, so quite likely there are no other receivers, which are in the two-hop neighborhood of receiver R. Thus quite likely receiver R does not share any neighbors with other receivers, and hence there is approximately only one neighbor of receiver R belonging to K. Thus after removing all the middle nodes of the first tree, receiver R still has at least one neighbor left. It has been shown in [89] that for high density networks, the number of independent paths from receiver R to the sender is equal to $Ne(R)$. Thus receiver R still has at least one path connecting to the sender, which belongs to set H'. Therefore $R \in H'$ approximately. Thus $A \approx B$. ■

B.3 Proof of Theorem 4.2 in Section 4.2.3: The relation between node densities required for single and double tree schemes

Theorem 4.2: Consider an infinite wireless network, with nodes assumed to be distributed according to two-dimensional poisson process. Let D_1 denote the required node density to achieve a given tree connectivity level, P, in a single tree case. If $D_1 > \lambda_c$, there exists at least one double disjoint tree whose required node density D_2 to achieve P satisfies

$$D_2 - \frac{\ln(\pi D_2 r^2 + 1)}{\pi r^2} \leq D_1 \leq D_2 \tag{B.15}$$

where r is the radio link range.

We use S to denote the sender, R_i to denote Receiver i, $i = 1, \ldots, m$, m and n to denote the number of receivers and nodes respectively, and H and H' to denote

the connected infinite supercluster and the one after removing all the middle nodes of the first tree respectively. From Equation (4.1), tree connectivity of a double tree scheme is expressed as $P_2 = E[\frac{N}{2m}]$, where $N = \sum_{i=1}^{m} n_i$, with n_i denoting the number of disjoint trees that receiver i connects to. We define N_1 and N_2 as the number of receivers which belong to H and H' respectively. The fraction of nodes belonging to H is $\theta_2 \approx 1 - \exp(-\pi D_2 r^2)$ [93].

By the definition of N_1 and N_2,

$$N = N_1 \times 1(S \in H) + N_2 \times 1(S \in H') \tag{B.16}$$

Similar to Equation (B.3) in Appendix B.1,

$$E[N_1 \times 1(S \in H)] = m\theta_2^2 \tag{B.17}$$

We also have

$$E[N_2 \times 1(S \in H')] = E[1(S \in H')]E[\sum_{i=1}^{m} 1(R_i \in H')] \tag{B.18}$$

Since $S, R_1, \ldots R_m$ are independent identically distributed, $P(S \in H') = P(R_i \in H'), i = 1, \ldots m$. We denote this probability as $P(R \in H')$, and simplify Equation (B.18) as

$$\begin{aligned} E[N_2 \times 1(S \in H')] &= mP(S \in H')P(R \in H') \\ &= mP^2(R \in H') \end{aligned} \tag{B.19}$$

From *Claim 4.1*, we get:

$$\begin{aligned} P(R \in H') &\approx P(\mathrm{Ne(R)} \geq 2, R \in H) \\ &= P(\mathrm{Ne(R)} \geq 2) - P(\mathrm{Ne(R)} \geq 2, R \notin H) \end{aligned} \tag{B.20}$$

From the definition of point poisson process, we have

$$P(\mathrm{Ne(R)} = 0) = \exp(-\pi D_2 r^2) \tag{B.21}$$

$$P(\text{Ne(R)} = 1) = \pi D_2 r^2 \exp(-\pi D_2 r^2) \tag{B.22}$$

Thus,

$$\begin{aligned} P(\text{Ne(R)} \geq 2) &= 1 - P(\text{Ne(R)} = 0) - P(\text{Ne(R)} = 1) \\ &= 1 - \exp(-\pi D_2 r^2) - \pi D_2 r^2 \exp(-\pi D_2 r^2) \end{aligned} \tag{B.23}$$

Also,

$$\begin{aligned} P(\text{Ne(R)} \geq 2, R \notin H) &= P(\text{Ne(R)} \geq 2 | R \notin H) \cdot P(R \notin H) \\ &= P(\text{Ne(R)} \geq 2 | R \notin H) \cdot \exp(-\pi D_2 r^2) \\ &\ll \exp(-\pi D_2 r^2) \end{aligned} \tag{B.24}$$

The last inequality holds because in a high density network, when a node does not belong to the super-cluster, very likely it is isolated, i.e. has no neighbors [93]. Substituting (B.23) and (B.24) into (B.20), we get:

$$P(R \in H') \approx 1 - \pi D_2 r^2 \exp(-\pi D_2 r^2) - \exp(-\pi D_2 r^2) \tag{B.25}$$

And from (B.16), (B.17), and (B.19), we get:

$$\begin{aligned} P_2 &= E[\frac{N}{2m}] \\ &= \frac{\theta_2^2}{2} + \frac{P^2(R \in H')}{2} \\ &\geq P^2(R \in H') \end{aligned} \tag{B.26}$$

Letting $P_1 = P_2$, and using Equation (B.1), we get $P_1 = \theta_1^2 = P_2 \geq P^2(R \in H')$. Substituting Equation (B.25) for $P(R \in H')$, we get:

$$(\pi D_2 r^2 + 1) \exp(-\pi D_2 r^2) \geq \exp(-\pi D_1 r^2) \tag{B.27}$$

Therefore,

$$D_2 - \frac{\ln(\pi D_2 r^2 + 1)}{\pi r^2} \leq D_1 \leq D_2 \tag{B.28}$$

■

B.4 Proof of Claim 4.2 in Section 4.4.4

Claim 4.2: Given any two nodes N_a and N_b, which are middle nodes for tree-0 and tree-1 respectively, let JQ_a and JQ_b denote node sets of last hops of JQ messages stored in the JQ Message Caches of nodes N_a and N_b respectively. We sort nodes in JQ_a and JQ_b according to the arrival time of corresponding JQ messages. Let nodes N_c and N_d denote upstream nodes obtained by the Parallel MNTMR of nodes N_a and N_b respectively. We have $N_c \neq N_d$, if the first two nodes of JQ_a and JQ_b are the same.

We prove the claim through enumerating all possible scenarios. We list all the scenarios in Table B.1.

Table B.1. All scenarios of Claim 4.2

Scenario Number	Types of Nodes in JQ_a	Types of Nodes in JQ_b
1	All group-0 nodes	All group-0 nodes
2	All group-0 nodes	Both group-0 and group-1 nodes
3	All group-1 nodes	All group-1 nodes
4	All group-1 nodes	Both group-0 and group-1 nodes
5	Both group-0 and group-1 nodes	All group-0 nodes
6	Both group-0 and group-1 nodes	All group-1 nodes
7	Both group-0 and group-1 nodes	Both group-0 and group-1 nodes

Let message sets JQM_a and JQM_b denote JQ Message Caches of nodes N_a and N_b respectively. In scenarios 2, 6 and 7, according to the *upstream node selection rule*, N_c is the last hop of the first received group-0 JQ message in JQM_a, and N_d is the last hop of the first received group-1 JQ message in JQM_b. Therefore $N_c \neq N_d$.

In scenario 1, using the *upstream node selection rule*, N_c is the last hop of the first received group-0 JQ message, which is also the first received JQ message in JQM_a. N_d is the last hop of the second received JQ message in JQM_b. Since the first two JQ messages of JQM_a and JQM_b are the same, $N_c \neq N_d$. Similarly in scenario 3, we arrive at the same conclusion.

In scenario 4, N_c is the last hop of the second received JQ message. Since the first two JQ messages of JQM_a and JQM_b are the same, the first two JQ messages of JQM_b are group-1 JQ messages. Thus N_d is the last hop of the first group-1 JQ message, which is also the first JQ message. Therefore $N_c \neq N_d$. We could arrive at the same conclusion in scenario 5 in a similar fashion.

Therefore for all seven possible scenarios, $N_c \neq N_d$. ■