The Dissertation Committee for Eric John Rozner
certifies that this is the approved version of the following dissertation:

Combatting Loss in Wireless Networks

Committee:

Lili Qiu, Supervisor

Lorenzo Alvisi

Ranveer Chandra

Gustavo de Veciana

Yin Zhang

Combatting Loss in Wireless Networks

by

Eric John Rozner, B.S.; M.S.C.S.

DISSERTATION

Presented to the Faculty of the Graduate School of

The University of Texas at Austin

in Partial Fulfillment

of the Requirements

for the Degree of

DOCTOR OF PHILOSOPHY

THE UNIVERSITY OF TEXAS AT AUSTIN

December 2011

Dedicated to my wife Rebecca.

Acknowledgments

Graduate school has been a long and hard journey. Getting my doctorate is one of the hardest things I have ever done. It truly would not have been possible without an incredible support system. I owe gratitude to many people— too many to list here. To all those that helped me through my journey, I thank you all very much.

First, I would like to thank my wife, Rebecca. She had unbounded patience for my studies and was understanding when deadlines approached. She has made just as many, if not more, sacrifices than I have during this journey. I could not have done it without her and am eternally grateful for everything she has done for me. She has further enriched my life by adding many wonderful pets that helped reduce my stress and give me a rewarding hobby to focus on outside of my studies.

My family has also been extremely supportive. I would like to thank my parents, Nancy and Roy, for emphasizing the importance of education at a young age and their constant support and encouragement throughout my undergraduate and graduate studies. Their support is incredibly reassuring and comforting to me. I would also like to thank my sisters, Holly and Sarah, for empathizing with me when needed. I must also thank my extended family, specifically my Uncle Jerry, Aunt Sharon, Katie, Steven and Bradley. I cannot thank Jerry and Sharon enough for everything they have done for me over the years. They are both role-models and I am so happy that they are part of my life. Katie is amazing, and I am happy she has done such a great job keeping in touch with me over the years. Her phone calls always cheer me up. Finally, I would like to thank my in-laws. Mark, Lani and Nicole have made me feel like family from day one. I am grateful to have such wonderful in-laws and want to thank them for all their support, hospitality and great food over the years.

Next, I must thank all my friends and colleagues. There are too many people to list here, but I have thoroughly enjoyed all your company, support and fun times over the years. I would especially like to thank Edmund– you are a great friend and I am grateful that you have been there for me through thick and thin. To all the co-authors on my papers: thank you all your hard work and for helping me learn, teach and have fun with you along the way.

Last, and certainly not least, I must express my extreme gratitude for my mentors. To Lili, I appreciate you always pushing me to be my best. I may not have always been the easiest student to advise, but you have been consistently clear, kind, supportive and intelligent. I hope that one day I can be as good of a mentor and researcher as you. I must also thank others that I have worked closely with: Yin, Vishnu, Ram, Ranveer, and Kun. I have learned a lot about tackling and presenting research problems and solutions from each of you. Finally, I would like to thank the other members of my committee: Gustavo and Lorenzo. Gustavo, your insights and questions have had an impact on me. I will try to focus on what I am most proud of and use that to drive future work. Lorenzo, thank you so much for all the advice and support over the years; you always had the exact right thing to say at the most needed times.

Combatting Loss in Wireless Networks

Eric John Rozner, Ph.D.
The University of Texas at Austin, 2011

Supervisor: Lili Qiu

The wireless medium is lossy due to many reasons, such as signal attenuation, multi-path propagation, and collisions. Wireless losses degrade network throughput, reliability, and latency. The goal of this dissertation is to combat wireless losses by developing effective techniques and protocols across different network layers.

First, a novel opportunistic routing protocol is developed to overcome wireless losses at the network layer. Opportunistic routing protocols exploit receiver diversity to route traffic in the face of loss. A distinctive feature of the protocol is the performance derived from its optimization can be achieved in real IEEE 802.11 networks. At its heart lies a simple yet realistic model of the network that captures wireless interference, losses, traffic, and MAC-induced dependencies. Then a model-driven optimization algorithm is designed to accurately optimize the end-to-end performance, and techniques are developed to map the resulting optimization solutions to practical routing configurations. Its effectiveness is demonstrated using simulation and testbed experiments.

Second, an efficient retransmission scheme (ER) is developed at the link layer for wireless networks. Instead of retransmitting lost packets in their original forms, ER codes packets lost at different destinations and uses a single retransmission to potentially recover multiple packet losses. A simple and practical protocol is developed to realize the idea, and it is evaluated using simulation and testbed experiments to demonstrate its effectiveness.

Third, detailed measurement traces are collected to understand wireless losses in dynamic and mobile environments. Existing wireless drivers are modified to enable the logging and analysis of network activity under varying end-host configurations. The results indicate that mobile clients can suffer from consecutive packet losses, or burst errors. The burst errors are then analyzed in more detail to gain further insights into the problem. With these insights, recommendations for future research directions to mitigate loss in mobile environments are presented.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Motivation

Wireless networks are ubiquitous in today's world and networks using the IEEE 802.11 protocol [1] have been adopted for a wide range of scenarios. Many different classes of these networks exist, such as ad-hoc networks, multiple-hop wireless mesh networks, and Wireless Local Area Networks (also known as WLANs or Wi-Fi hotspots). Wireless ad-hoc networks consist of multiple devices that may be fixed or mobile and may need to route traffic over multiple wireless hops with intermittent connectivity in order to communicate. For instance, ad-hoc networks have been used for battlefield communications, vehicular networks, delay tolerant networks, and sensor networks. Wireless mesh networks have been successfully deployed to cover large, dedicated areas. To support large coverage areas, the networks consist of many static mesh nodes that communicate with one another to route traffic over multiple wireless hops. Only a few of the mesh nodes may be outfitted with a connection to a back-end network (for instance, the Internet). Therefore the rest of the mesh nodes need to cooperate to route traffic so that every node in the mesh can gain access to the back-end network. The infrastructure for these networks is typically cost-effective, leading them to be deployed over traditional approaches such as wired networks. For example, many cities and towns have implemented mesh networks to blanket certain areas, such as downtown regions and parks (a partial list can be found in [3]). Mesh networks can also be deployed in low-income areas in order to provide free or heavily-subsidized Internet connectivity [147]. Furthermore, there has been a trend towards deploying these networks in developing countries [117, 155], where wiring costs are prohibitive. Beyond these uses, mesh networks are also being deployed commercially [97]. Finally, perhaps the most common type of 802.11 network is the WLAN. These networks consist of one or more access points (APs), each having a dedicated back-end Internet connection. Access points provide Internet connectivity to wireless and mobile devices over a single hop connection. These networks have been adopted in many different environments. They are commonly found in homes, office buildings, campuses, coffee shops, restaurants, airports, malls, and hotels.

The challenge all of these networks face, regardless of their deployment or intended use, is coping with with a problem fundamental to all wireless communications: loss. The wireless medium is lossy due to many prevailing factors. For instance, the medium is inherently lossy and phenomena such as signal attenuation and multi-path propagation can cause wireless packets to become corrupted. Furthermore, wireless transmissions can be subject to collisions from imperfect packet scheduling or hidden terminals. All of these effects can combine to create very high loss on wireless links. Some deployments report average loss rates of up to 20-40% [7, 124], while others show that lossy links are quite common [32]. The effects of loss become even worse for multiple hop wireless networks because each hop is susceptible to loss. This high loss rate makes it difficult for wireless technologies to realize their full potential.

Loss causes many problems for wireless networks. Three main indicators of network performance suffer in the face of loss: latency, throughput, and the successful delivery of packets. For example, additional latency is imposed on data packets that are lost in wireless networks. To cope with loss, retransmissions are often employed. Not only does the retransmission impose extra latency on the packet itself, but it also imposes latency on other traffic in the network. The scheduler in the 802.11 standard uses head-of-line blocking, so all packets in the transmit queue must wait for the retransmissions to be sent. Throughput can also be affected by loss. The wireless medium is a shared broadcast medium. Therefore, retransmissions to one node take up air-time that could be used to transmit data to other nodes in the network. Furthermore, a single client with high loss can impact the performance of the whole network when it reduces its sending rate [17], regardless of the link quality of the other nodes. Finally, lossy wireless links can cause packets to be lost in transit, forcing upper-layers and applications to add redundancy and robustness on an end-to-end basis.

The negative impact on latency, throughput and reliability can wreak havoc on the higher network layers. For example, it has been widely shown that TCP performs poorly in wireless environments [15,16,56,159], mainly because losses in wireless environments are often due to poor signal, rather than network congestion. Furthermore, interactive applications typically tolerate small latencies and loss. The effects from packet losses in wireless environments has been shown to degrade video quality [87, 88], as well as VoIP calls [106]. Lastly, applications that deal with safety and monitoring must be robust and cannot afford to lose packets.

In order to avoid degraded network performance, the goal of this dissertation is to combat loss at varying network layers in order to allow wireless technologies to realize their full potential. Attacking loss at each level of the network stack provides increased robustness: if a coping mechanism fails in one layer, the adjacent lower layer can be employed to mitigate the problem. For the purposes of this dissertation, loss is tackled in the Network Layer and below.

1.2 Review of 802.11

This dissertation focuses on combatting losses in 802.11 networks [1]. Here, a brief overview of 802.11 provides the necessary background for the rest of the document. The 802.11 standard specifies two types of coordination functions that dictate how nodes can access the wireless medium. The distributed coordination function (DCF) allows nodes to access the wireless medium in a distributed manner through CSMA/CA (carrier-sense multiple access with collision avoidance), while the point coordination function (PCF) enrolls a point coordinator (typically an access point) to to coordinate communication in the network. Wireless devices rarely support PCF, and as a result, it is not widely used in practice. Therefore, this summary focuses on DCF.

Prior to a node transmitting data, it must first carrier sense the medium as idle. There are two sensing techniques: physical carrier-sensing and virtual carrier-sensing. In physical carrier-sensing, a sender determines the channel to be idle when the total energy received is less than the clear-channel assessment threshold.

Virtual carrier-sensing relies on a Network Allocation Vector (NAV), which is a field contained in the 802.11 packet header that indicates the total transmission time of a packet. When nodes overhear a packet, they set their local NAV counter to the value in the packet header. Virtual carrier-sense classifies the medium as idle only if the NAV counter is zero. A node can only classify the medium as idle when the virtual-carrier sensing and the physical carrier-sensing indicate the medium is idle. There are two types of data packets a node can send in 802.11, broadcast (which includes multicast) and unicast. A brief description of each is listed below.

We first review broadcast transmissions as specified by the IEEE 802.11 standard. Before transmission, a sender first checks to see if the medium is available using the sensing techniques outlined in the previous paragraph. In this case, a sender may begin transmission using the following rule: If the medium has been idle for longer than a distributed inter-frame spacing time (DIFS) period, transmission can begin immediately. Otherwise, a sender waits for DIFS and then selects a random back-off interval uniformly chosen between $[0, CW_{\min}]$, where $CW_{\min}$ is the minimum contention window. The sender begins to count down its randomly chosen interval. If, however, the medium no longer remains idle during the countdown, the sender freezes its back-off counter and the countdown doesn't resume until the medium is again sensed idle. Once the countdown reaches zero, the sender can send the packet. A special destination address is used in the packet header to denote a broadcast packet, and all nodes try to receive the packet. However, due to scalability concerns, the sender does not try to ensure that all nodes receive the packet. Therefore, no retransmissions are provided for broadcast packets.

With 802.11 unicast transmissions, a station waits for the medium to become idle as before. When it transmits the packet, it waits for an ACK. If the receiver successfully receives the packet, it waits for a short inter-frame spacing time (SIFS) and then transmits an ACK frame. If the sender does not receive an ACK (e.g., due to a collision or poor channel condition), it retransmits the packet using binary exponential back-off, where its contention window is doubled every time after a failed transmission until it reaches its maximum value, denoted as CW_{max}. In 802.11, the packet is retransmitted in its original form and up to a pre-specified number of times. After the successful reception of a packet, the sender reduces its contention window to $CW_{\min}$.

Finally, in order to avoid collisions due to hidden terminals, a sender can employ the Request-To-Send/Clear-To-Send (RTS/CTS) mechanism for unicast packets. Instead of sending a data packet, the sender first sends a RTS packet when it wins contention. The RTS packet contains a NAV value indicating the length of the packet transmission (including the RTS/CTS and ACK overhead). When the receiver gets the RTS packet, it replies with a CTS packet, which also contains an updated NAV. Any other node receiving the RTS/CTS frames updates its NAV value accordingly and thus defers to the upcoming transmission. Once the sender receives the CTS, it sends the data.

1.2.1 The 802.11n Standard

The IEEE 802.11n standard [2] was introduced as an amendment to the 802.11a and 802.11g standards. In 802.11n, physical-layer data rates of 600 Mbps are possible, whereas only 54 Mbps was possible in the the legacy 802.11 standards. The increase in throughput in 802.11n is contributed to two factors: (i) the use of multiple-input and multiple-output (MIMO) antennas at both the transmitter and the receiver, and (ii) the use of MAC and PHY layer aggregation techniques.

The 802.11n standard utilizes MIMO technology to achieve high data rates. At a high level, the sender and receiver can be affixed with multiple antenna pairs, and the data rates scale linearly with the number of antenna pairs. In Table 1.1, the data rates are enumerated for up to 3 spatial streams. The table lists the data rates defined in the 802.11n standard, and the various rates are indexed by a Modulation and Coding Scheme (MCS) index value. Therefore, the MCS index is used when describing a particular rate in 802.11n. While the table shows up to 3 spatial streams, in the 802.11n standard, up to 4 spatial streams are allowed. Further reading on MIMO and spatial diversity can be found in [53] and [2].

The second enhancement in 802.11n is the use of frame aggregation to improve efficiency in the MAC layer. As described in Section 1.2, each data packet in 802.11 must incur some constant-in-time overheads (DIFS, preamble, SIFS, contention time and acknowledgment frames). However, as the data rate increases, the time to send the packet becomes increasingly smaller and the overhead associated with the fixed inter-frame spacing intervals becomes increasingly higher. This overhead severely limits the throughput that can be achieved in practice. Therefore,

MCS index	Spatial streams	Modulation type	Coding rate	Data rate (Mbit/s)			
				20 Mhz channel		40 Mhz channel	
				800 ns GI	400 ns GI	800 ns GI	400 ns GI
0	1	BPSK	1/2	6.5	7.2	13.5	15
1	1	QPSK	1/2	13	14.4	27	30
2	1	QPSK	3/4	19.5	21.7	40.5	45
3	1	16-QAM	1/2	26	28.9	54	60
4	1	16-QAM	3/4	39	43.3	81	90
5	1	64-QAM	2/3	52	57.8	108	120
6	1	64-QAM	3/4	58.5	65	121.5	135
7	1	64-QAM	5/6	65	72.2	135	150
8	2	BPSK	1/2	13	14.4	27	30
9	2	QPSK	1/2	26	28.9	54	60
10	2	QPSK	3/4	39	43.3	81	90
11	2	16-QAM	1/2	52	57.8	108	120
12	2	16-QAM	3/4	78	86.7	162	180
13	2	64-QAM	2/3	104	115.6	216	240
14	2	64-QAM	3/4	117	130	243	270
15	2	64-QAM	5/6	130	144.4	270	300
16	3	BPSK	1/2	19.5	21.7	40.5	45
17	3	QPSK	1/2	39	43.3	81	90
18	3	QPSK	3/4	58.5	65	121.5	135
19	3	16-QAM	1/2	78	86.7	162	180
20	3	16-QAM	3/4	117	130.7	243	270
21	3	64-QAM	2/3	156	173.3	324	360
22	3	64-QAM	3/4	175.5	195	364.5	405
23	3	64-QAM	5/6	195	216.7	405	450

Table 1.1: 802.11n rate table for up to 3 streams.

the 802.11n standard allows multiple data packets to be combined into a single aggregated frame. This aggregated frame incurs the fixed time overheads only once, allowing the cost to be amortized over all the packets contained within the frame. There are two types of aggregation techniques: Aggregation of MAC Service Data Units at the top of the MAC (A-MSDU) and Aggregation of MAC Protocol Data Units at the bottom of the MAC (A-MPDU). The A-MPDU technique allows each of the aggregated data frames to be individually acknowledged and thus requires the use of the 802.11n Block ACK scheme [2].

1.3 Related Work

Due to the significant impact of losses in wireless networks, there exists a large body of literature in this area. This section provides a brief overview of the related work for each layer of the network stack in order to set the context for the proposed approaches. Chapter 2 presents a more detailed look at related work.

1.3.1 Network Layer

Mitigating loss in the Network Layer has typically centered around designing routing protocols for wireless mesh networks that account for loss when routing traffic. Early on, researchers found that the hop count metric, as typically used in AODV [112] and DSR [66], did not provide good performance in wireless networks. This is because not all hops are created equal: some hops have higher loss than others, which impacts the performance of a given hop. Therefore, Couto et al. [32] proposed routing traffic based on a routing metric called ETX, which explicitly accounts for the expected number of transmissions needed to forward a packet over a wireless link based on the probability of a packet getting lost on the link. Since then, researchers have proposed many other metrics to route data [13, 35, 36, 47, 58, 138, 158], such as the expected transmission time, which explicitly accounts for multiple rates and interfering links (WCETT) [35], or signal quality [36,47,58].

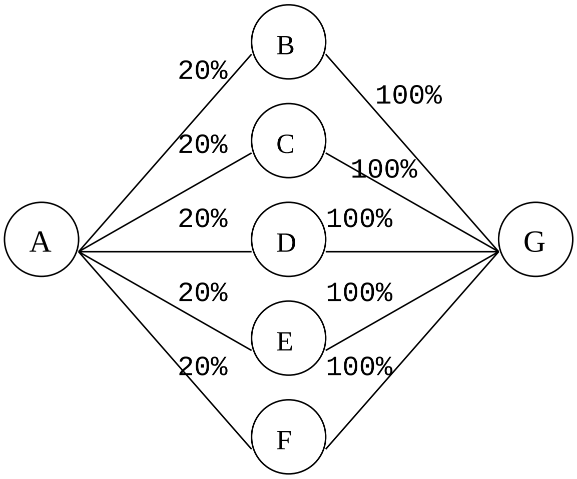


Figure 1.1: Opportunistic routing can take advantage of multiple weak links. The source A has weak wireless connectivity to each of the five intermediate nodes, with a delivery rate of 20%. All the intermediate nodes have 100% delivery rate to the destination G.

While these metrics and routing protocols are designed explicitly for wireless networks, they still treat wireless transmissions as point-to-point communication links. However, the wireless medium is a broadcast medium and links do not have to be treated as point-to-point. A class of *opportunistic routing* protocols [20, 24, 72, 91, 129, 161] exploit the broadcast nature of the wireless medium and defer route selection until after packet transmissions. This can cope well with unreliable and unpredictable wireless links. There are two major benefits in opportunistic routing. First, it can combine multiple weak links into one strong link. For example, in Figure 1.1, traditional routing requires the source A to pick one intermediate node as the next hop. Multiple packet transmissions are required to send data to the next hop, due to the loss on the link. For example, in Figure 1.1 five transmissions are required, on average, to send a packet over the first hop. However, in opportunistic routing, the source can specify a set of nodes as the next hop and transmit its data. If any one of these nodes receives the data, it can forward it to the destination. This essentially combines multiple weak links into one strong link. For example, in Figure 1.1 only 1.48 transmissions are required, on average, to send a packet to at least one of the forwarders on the first hop (assuming independent packet loss). Second, opportunistic routing takes advantage of unexpectedly short or unexpectedly long transmissions. A traditional routing protocol has to trade off between link quality and the amount of progress each transmission makes. For example, consider the network shown in Figure 1.2, where A sends data to D along the path $A - B - C - D$. If B is used as the next hop and the quality of link $A - B$ is good, then no retransmissions are required to deliver the packet to B. But the progress made is small. Alternatively, if C is chosen as the next hop, a large progress is made if the packet reaches C. However if the quality of link $A - C$ is poor, multiple transmissions are required to deliver the packet to C. In comparison, opportunistic routing does not commit to B or C before transmissions. Among the nodes that receive the packet, the one closest to the destination is chosen to forward.

Early opportunistic routing protocols required forwarders to coordinate amongst

Figure 1.2: Opportunistic routing can maximize the progress each transmission makes.

themselves to determine which forwarder would actually send a received packet. The sender of each individual packet was typically determined by strict timing constraints and forwarder priorities [20]. However, MORE [24] is an opportunistic routing protocol that applies network coding to ease the coordination between potential forwarding nodes. MORE puts multiple packets into a batch and sends out packets that are random linear combinations of the batch. Since random linear coding generates linearly independent coded packets with high probability, the forwarding nodes in MORE require no coordination. Instead, each node computes *how much* traffic it should forward given a received packet and independently generates random linear combinations of all the packets it has received from a current batch of packets.

1.3.2 Link Layer

Techniques to overcome loss and support reliable communications in the Link Layer have centered around adding redundancy to cope with wireless loss or trying to avoid loss in the first place. A brief explanation of works within each class is provided below.

Adding redundancy: Perhaps the most widely known mechanism to recover from packet losses is the simple retransmission technique employed by the IEEE 802.11

MAC layer [1], as described in Section 1.2. As an alternative to retransmissions, forward error correction (FEC) coding can also be employed to add redundancy in the face of loss. There have been various schemes for FEC coding in the wired environment (*e.g.*, [19, 121, 122, 139]), but schemes have also been developed for the wireless environment (*e.g.*, [80, 96, 109, 123, 162, 168]). For instance, in [96], McKinley et al. dynamically adjust the level of FEC redundancy based on observed channel quality.

Avoiding Loss: Various schemes try to prevent loss from happening in the first place. These schemes aim to prevent losses arising from packet collisions. A widely-known scheme is the RTS/CTS mechanism in the IEEE 802.11 MAC layer, as outlined in Section 1.2. Since RTS/CTS only applies to unicast traffic, other schemes have been developed to extend channel reservation for broadcast traffic [34, 59, 143, 144].

1.3.3 Physical Layer

There has been a large body of work to measure and increase link robustness in the Physical Layer. A brief description of each is provided here.

Characterizing Wireless Links: Numerous studies have shown that loss is prevalent in wireless networks [7, 32, 39, 120]. Furthermore, additional studies have shown that wireless losses exhibit bursty patterns [23, 65, 79, 99, 137]. In these works, burstiness is found to occur in challenged environments: signal strength is low or near the receive sensitivity threshold.

Increasing Link Robustness: There are many classes of work that aim to create a better wireless link. In the interest of brevity, a short summary is provided here (a more detailed description can be found in Section 2.3). One class of schemes aim to pick the best data rate for a wireless link [33, 103, 116, 151, 156]. Other schemes allow the partial reception of uncorrupted bits in errored frames [64, 89]. Finally, other well-known techniques, such as utilizing diversity [11, 63, 101, 146] or using interference cancellation [148, 150], can also reduce loss and make a link more robust.

1.4 Challenges

We now describe the challenges for combating loss in wireless networks and highlight why the previous work is not sufficient.

1.4.1 Network Layer

Opportunistic routing protocols leverage receiver diversity to effectively route traffic over multiple hops in the face of loss. There are two key factors that determine the performance of opportunistic communication in wireless mesh networks: (i) routes (*i.e.*, for a given flow how much traffic node j should forward upon receiving a packet from another node i), and (ii) rate limits (*i.e.*, how fast each traffic source can inject traffic into the network). Routes determine how effectively we take advantage of communication opportunities and how efficiently we utilize network resources and exploit spatial reuse. Rate limits ensure that traffic sources do not send more than what paths can support. Without appropriate rate limits, the network throughput can degrade drastically under traditional shortest-path routing [85]. Rate limiting is even more critical for opportunistic routing due to its use of broadcast transmissions: (i) broadcast transmissions do not perform exponential back-off (*i.e.*, its contention window does not increase upon packet losses) and thus are more likely to cause network congestion; and (ii) broadcast transmissions preclude the use of 802.11's synchronous ACK mechanism, and receivers' feedback has to be sent above the MAC layer, which can easily get lost during network congestion and cause unnecessary retransmissions and serious throughput degradation. Previous approaches to opportunistic routing use a set of heuristics to determine the routes that are utilized [20, 24, 72, 91, 129, 161], and only [129] attempted to rate limit traffic. Each node makes these decisions locally and does not incorporate network-wide information, like interference, into its decisions.

To this end, a class of related works has formulated opportunistic routing as an optimization problem in order to provide interference-aware route selection to reduce collision losses and compute safe sending rates to avoid losses due to congestion [93, 115, 135, 141, 165]. However, these works use a simplistic model of interference to derive their routes and sending rates. In order to ensure that the derived configurations are feasible and the optimized results are effective, an accurate network model should be employed.

However, accurate optimization of opportunistic communication in an IEEE 802.11 network is challenging for the following four reasons. *First*, the dynamic and incidental nature of communication opportunities makes it difficult to estimate their impact on the resulting network performance. *Second*, optimization of opportunistic routing places stringent requirements on a network model: the model should (i) specify the region of feasible network configurations using a compact representation so that we can optimize the objective within the feasible region as defined by these constraints, (ii) accurately estimate performance on every link in the network (as opposed to only a small number of links on specified routes, as in [85], for the purpose of optimizing rate limiting alone), and (iii) be accurate across a wide range of traffic conditions, including high traffic load, which is common in opportunistic routing. *Third*, the non-convex interference relationships among different links and the huge search space of possible opportunistic routes and rate limits impose significant challenges on the optimization procedure itself. It is unclear whether one should resort to a less accurate model that is convex and easier to optimize globally, or use a more accurate but non-convex model and settle for a local optimum. *Fourth*, to be valuable in practice, the resulting optimization solution should be easy to implement, using only a small number of control knobs.

1.4.2 Link Layer

Providing reliability in an efficient manner in the Link Layer is difficult because the wireless medium is inherently lossy and also suffers from packet losses due to collisions. Consider a WLAN environment where an access point may have multiple client flows to serve, where each client may have different loss rates and link qualities. Previous approaches providing reliability, such as the IEEE 802.11 retransmission technique or FEC coding, achieve robustness on a *per node* basis. That is, they treat the wireless medium as a point-to-point link from the access point to the client and try to provide reliability over that link.

A key challenge in providing Link Layer reliability in wireless domains is how to achieve robustness over *multiple nodes* simultaneously and efficiently in the network. When providing reliability over multiple nodes, the previous schemes simply iterate the reliability scheme over all links. But since the wireless domain is a broadcast medium, the point-to-point abstraction no longer holds. How to exploit the broadcast property of the wireless medium to provide robustness to many nodes at the same time is a challenging problem.

The technique must provide the same reliability guarantees as existing approaches, while providing a mechanism that exploits the broadcast properties of the wireless medium in order to simultaneously recover loss for multiple nodes. Algorithms for recovering lost packets to multiple nodes simultaneously must be done in a simple and effective way. Ideally, the approach should work with off-the-shelf hardware so it can be used in today's networks. Furthermore, the scheme should be lightweight so that it can be employed by resource-constrained wireless devices such as smart-phones.

Finally, the solution should work with broadcast, as well as unicast, data traffic. Broadcast traffic is becoming increasingly popular. For example, it is used for content dissemination, such as broadcast video, file sharing and location-aware notifications (such as advertisements). A scheme that provides retransmissions for broadcast must be able to retransmit its data in an efficient way in order to scale with the number of broadcast clients.

1.4.3 Physical Layer

We examine loss in the Physical Layer with the motivation that many applications are pushing the wireless medium towards its limits. For instance, consider the home of the not-too-distant future that may be equipped with multiple high-definition televisions, laptops and tablet PCs. The devices could be used for wirelessly streaming television and movie programming to the televisions, but can also be used for live, interactive applications like gaming, security monitoring and video conferencing. Therefore, the wireless links need high throughput to support high-definition streams, but also very low latency to provide interactivity.

Combating loss in this scenario is difficult because high-throughput schemes rely on aggregating multiple packets together to reduce MAC-layer inefficiency. However, creating larger packets makes it difficult for retransmissions to adhere to the strict latency requirements imposed by the interactive nature of the content. The problem is significantly worsened when coupled with the fact wireless losses are typically bursty in nature [23, 65, 79]. This implies these losses must be eliminated in order to support high-throughput, low latency applications on wireless links.

To eliminate these burst losses, we must first understand the details of what happens during the burst. This can prove difficult, especially because most off-the-shelf hardware does not readily provide adequate low-level details when packets are lost. Therefore, a platform must be developed that allows analysis of corrupted packets so that root causes of this packet loss can be flushed out.

Leveraging previous related work is difficult because of our unique usage scenario. Previous work typically does not jointly consider strict latency requirements and high throughput requirements. Furthermore, in our scenario, signal strength is typically strong and the wireless endpoints typically have a clear line-of-sight. Much of the previous work focuses on more challenged environments where signal strengths are typically low and therefore requirements for performance are not as strict.

1.5 Thesis Contributions

Given a brief overview of the related work and the challenges addressed in the previous section, an overview of contributions of this thesis are now presented. Specifically, approaches for combating loss in each layer in the networking stack are outlined below.

1.5.1 Network Layer

Our goal is to optimize opportunistic routing, but ensure it is being driven by an accurate model of broadcast interference. With an accurate model of interference, we can be confident that we're optimizing the actual network performance, rather than utilizing a set of heuristics or an inaccurate model that may not directly correlate to system-wide network performance. If an optimization framework is used to select routes and enforce safe sending rates, the underlying model must be accurate so that the results being installed into the network can actually be achieved. To achieve our goal, we develop our scheme with the following steps:

1. *Interference model for IEEE 802.11 broadcast traffic.* The complex interference, traffic, and MAC-induced dependencies in the network are often the underlying cause of unexpected behavior. We develop a simple yet accurate model to capture these dependencies for broadcast transmissions. We use measurements from a given network to estimate link loss rate, carrier sense probability, and conditional collision loss probabilities to seed our model. Our model derives the relationships between sending rates, loss rates, and throughput to capture the effects of carrier sense and collisions. Despite its simplicity, the model captures real-world complexities such as hidden terminals, non-uniform traffic, multi-hop flows, non-binary and asymmetric interference.

2. *General optimization framework.* We then develop a general framework to jointly optimize routes and rate limits for opportunistic communication. The framework uses opportunistic constraints to probabilistically characterize the available communication opportunities. It allows the use of different wireless interference models so that we can evaluate the effectiveness of our model.

3. *Iterative procedure for non-convex optimization.* Since our model is non-convex, we develop an iterative optimization procedure to find a local optimal solution. Our algorithm is flexible and can accommodate different performance objectives. For comparison, we explore an alternative approach that uses a widely used conflict-graph-based interference model [62] that is less accurate [85, 113], but convex, and thus allows global optimization. Our

results show that our approach of combining a more accurate model with non-convex optimization yields better and more accurate performance.

4. *Practical installation of routes and rate limits.* We develop a practical opportunistic routing protocol that implements the opportunistic routes and rate limits optimized by our algorithm in real networks.

Finally, we evaluate our scheme in a network simulator, as well as a testbed. We find our model is highly accurate, typically estimating derived performance within 20%. The performance of the protocol is significantly higher than other schemes (its throughput is 2-13 times ETX's throughput and 1.5-10 times MORE's throughput). Furthermore, we evaluate the performance of our scheme in dynamic and uncontrolled environments.

1.5.2 Link Layer

Next, loss is dealt with in the Link Layer so single wireless hops, such as WLANs, can also employ mechanisms to combat loss. The Link Layer typically recovers from loss by retransmitting lost packets in their original forms. However, instead of retransmitting packets in their original form, packets lost at different destinations can be coded into a single retransmission in an attempt to recover multiple packet losses simultaneously.

Given this framework for retransmitting packets to multiple nodes simultaneously, we develop an Efficient Retransmission (ER) protocol to realize the benefits in 802.11 networks. Our goal is to develop a protocol that is simple to implement, has low overhead, codes multiple retransmissions effectively and can be run on off-the-shelf hardware. In designing ER, we consider the following key questions:

- First, how should the receivers give timely feedback to the sender without incurring much overhead? In ER, receivers send periodic feedback with aggregated information about which packets they've received in order to amortize the cost of providing feedback.

- Second, when to classify a packet as lost? In ER, senders keep track of the average time needed for receivers to provide feedback and use a simple round-trip timeout to determine that a packet is lost.

- Third, when the medium is available for the sender to transmit, which packet should it send – a new packet or a lost packet? This decision must be made carefully as to balance the trade-off between the amount of coding possible and the latency imposed on individual packets. In ER, senders store packets to be retransmitted in a separate queue so that coding decisions can be made over all lost packets easily. A scheduler in ER determines if a retransmitted or a lost packet is to be sent. It ensures outstanding packets do not exceed a pre-specified delay threshold and also ensures the retransmission queue does not overflow.

- Fourth, which set of packets should be coded together to minimize the number of retransmissions? We formally study this problem and show it is NP-hard. Therefore, we design several practical heuristics to code effectively and use empirical evaluation to study their effectiveness.

Finally, we develop and implement ER to provide reliable unicast, broadcast, and multicast in WLANs. Our extensive simulation and testbed experiments show that ER significantly reduces the number of retransmissions compared to the existing retransmission scheme, which retransmits lost packets by themselves.

1.5.3 Physical Layer

Finally, a case study is presented in the physical layer with the goal of measuring, understanding and eliminating losses. Our goal is to see if strict performance requirements can be imposed on the wireless link. We are concerned with having high throughput, but also very low latency. Our study focuses on the 802.11n standard, which is designed to give much higher throughput than the previous 802.11a/b/g standards. The 802.11n standard achieves high throughput by exploiting spatial diversity and eliminating MAC-layer inefficiencies. We examine the trade-offs of these approaches and find that network loss must be minimized to achieve our goals. Our study consists of three main parts:

- First, we modify existing wireless drivers to export fine-grained loss information. We integrate application layer measurement tools into the driver to ensure that we can measure losses specific to the wireless link, and avoid counting queue drops at the sender's buffer as wireless losses. We use the test platform to study throughput and latency over wireless links. Our case study focuses on a living-room like environment, where the sender and receiver are close to each other and even have a line-of-sight path. Our studies find that high throughput and low latency can be achieved in static environments. However, when client mobility is introduced, significant latency can incur. We vary the configurations at the sender and receiver to see if these latencies can be eliminated.

- Second, we make further modifications to the wireless driver in order to export information about wireless losses. We debug the burst losses in detail to try to examine what is happening during a burst loss. We find that a significant amount of losses are not typical data packet corruption, meaning that it isn't possible to recover only a few bits that may be in error. Instead, the whole aggregate packet must be recovered.

- Last, we provide recommendations for minimizing loss in mobile environments. We utilize the information obtained in our debugging step in order focus on what may be causing the problem. Based on our analysis, we find that increasing the robustness of the preamble is important to mitigating loss. Therefore, we make several recommendations of future research directions to minimize loss. We analyze the potential of several schemes that show promise in reducing loss: adding a preamble acknowledgement to avoid losing a whole aggregated packet, changing the preamble structure to make packet detection more robust, adding a midamble to help reduce CRC errors and finally utilizing multiple radios to increase receiver robustness.

Throughout our approach, we log and analyze extensive network traces. We use the traces to present our case study and further understand losses in the Physical Layer.

1.6 Thesis Outline

The outline of this thesis is as follows. Chapter 2 details the related work. Chapter 3 describes how loss is mitigated in the Network Layer through the use of opportunistic routing. Chapter 4 shows how loss is dealt with at the Link Layer through an Efficient Retransmission scheme. Chapter 5 presents a case study for measuring, understanding and minimizing loss in the Physical Layer. Finally, Chapter 6 concludes the thesis.

Chapter 2

Related Work

This chapter presents related work. Section 2.1 presents related work to our optimized opportunistic routing protocol in the Network Layer. Section 2.2 details related work to ER, our Link Layer approach. And finally, Section 2.3 examines related work for the Physical Layer project.

2.1 Network Layer

There are three main categories of related work when considering our model-based optimized opportunistic routing protocol: (i) design of traditional routing protocols, (ii) design and analysis of opportunistic routing protocols, and (iii) wireless network modeling.

2.1.1 Traditional Routing in Wireless Mesh Networks

Routing has been an active area in wireless networking research. Most of the original work in this area targeted high-mobility scenarios such as battlefield networks. Therefore, the focus was on establishing and maintaining routes under frequent and unpredictable changes in network connectivity. A number of on-demand routing protocols have been proposed for this purpose, as exemplified by DSR [66] and AODV [112], where packets are routed along paths with the shortest hop count.

Recently, wireless mesh networks [70, 126, 133] have emerged as a new dominant application of multihop wireless networks. Nodes in such networks have little or no mobility and often are not constrained by short battery-life or limited computational power. Therefore, the primary focus becomes improving network performance. Researchers have found the hop-count metric, as used in DSR and AODV, does not provide good performance since not all hops are equal. To address this issue, various link-quality metrics have been proposed. For example, Couto et al. [32] propose a routing metric, called ETX, which is based on link loss rate. Awerbuch et al. [13] develop a routing algorithm that chooses a path with the smallest transmission time. Draves et al. [35] design a routing metric, called ETT, based on the expected transmission time of a packet over the link. The authors further generalize the metric for multiple-radio multiple-channel networks. Several other routing schemes [36, 47, 58] propose the use of signal-to-noise ratio (SNR) as a link quality metric. These metrics quantify the quality of links using link loss rate, packet transmission time, or signal-to-noise ratios. Others use ETX scaled by factors such as modulation or the number of neighbors [35, 138, 158].

2.1.2 Design and Analysis of Opportunistic Routing Protocols

ExOR [20] is a seminal opportunistic routing protocol. In ExOR, a sender broadcasts a batch of packets. Each packet contains a list of nodes that can potentially forward it. To maximize the progress of each transmission, the forwarding nodes relay data packets in the order of their proximity to the destination in terms of ETX metric [32], which quantifies the number of transmissions required to deliver a packet from the forwarder to the destination. ExOR imposes strict timing constraints among the forwarders to avoid redundant transmissions. Specifically, it uses a batch map, which records the list of packets each node has received. Every forwarding node only forwards data that has not been acknowledged by the nodes with smaller ETX to the destination.

Since ExOR, many other opportunistic routing protocols have been proposed. ROMER [161] tries to forward packets simultaneously along multiple paths. It incorporates a credit based scheme to limit the number of transmissions a packet is allowed before reaching the destination. SOAR [129] is an opportunistic routing protocol designed to work with multiple flows in the network. It features adaptive forwarding path selection to exploit diversity and reduce redundant retransmissions, priority timer-based forwarding, local retransmissions, and adaptive flow control to avoid excessive network congestion. Afanasyev and Snoeren develop a scheme called modrate [5] to jointly couple physical-layer rate selection and overhearing opportunities.

MORE [24] applies network coding to opportunistic routing to avoid the need for forwarders to coordinate on which specific packet they need to send. Since random linear coding generates linearly independent coded packets with high probability, the forwarding nodes in MORE require no coordination. Instead, each node computes how much traffic it should forward and independently generates random linear combinations of all the packets it has received from the current batch. By obviating the needs for strict coordination, MORE can out-perform ExOR. However, the performance of MORE can degrade significantly when there are more than a few flows in the network. This is because (i) it lacks rate limiting and causes network congestion, and (ii) its routes only try to minimize the number of transmissions and do not take wireless interference into account.

Since then, several improvements upon MORE have been proposed. MIXIT [72] applies opportunistic routing at the symbol level, rather than the packet level. CodeOR [90] improves upon MORE by supporting multiple outstanding batches in a streamlined fashion. SlideOR [91] alleviates the problem of determining when to start transmitting the next batch of packets by encoding packets in overlapping sliding batches such that coded packets from one batch can be used to decode packets within another batch. CCACK [76] and SOR [82] develop new schemes for online assistance in how much traffic should be forwarded, rather than the offline approaches used in [24] and [20].

Other approaches examine how forwarders should be selected in opportunistic routing. Zhong *et al.* [167] show the routing metric used to select and prioritize forwarding nodes is important. They develop a new routing metric, called EAX, to account for inter-candidate communication in opportunistic routing, and show that EAX out-performs the ETX metric. Ferriere *et al.* [37] also point out the limitation of using ETX for selecting forwarding nodes. In particular, they show single-path metrics, such as ETX, ignore the nodes with low delivery rate to each of its neighbors even though collectively the links to multiple neighbors can form a strong wireless link. Based on this insight, they develop least-cost opportunistic routing, which quantifies the forwarding cost as the cost of reaching any neighbor in the direction towards the destination.

In addition to opportunistic routing protocols for mesh networks, researchers have also designed opportunistic routing protocols for ad-hoc and sensor networks. For example, [29] and [153] both dynamically select forwarding nodes based on recent link quality. However, in both protocols, only one forwarding node is selected before transmissions, and they cannot take advantage of transmissions reaching nodes other than the previously selected forwarder. [26] balances the energy consumption rates of different nodes in a sensor network by opportunistically incorporating forwarders' energy consumption.

There have been several studies analyzing the performance of opportunistic routing. For example, [163] develops a methodology for estimating the maximum throughput given forwarding paths and traffic demands, and [164] extends the work to multi-radio multi-channel wireless network. Both works assume the opportunistic routes are given, where nodes only forward the traffic that are not received by nodes closer to the destinations, so they cannot optimize routes. Such selected routes are not optimal for two reasons: (i) as shown in [37], prioritizing forwarders according to ETX, as done in [20, 24, 163, 164], is not optimal since the single path metric, like ETX, does not capture the anycast performance in the opportunistic routes, and (ii) due to wireless interference, we may sometimes prefer nodes farther away from the destination to forward traffic that has been received by nodes closer to the destination (in other words, none of least-cost path metrics can optimize end-to-end throughput).

A few studies (*e.g.*, [93, 115, 135, 141, 165]) propose optimization frame-works for opportunistic routing. The studies aim to maximize network throughput, but differ in their derivations. Like our protocol, these approaches can optimize throughput, proportional fairness, or energy [166] by constructing a linear program that takes into account broadcast interference, multi-rate, multi-power and flow control.

2.1.3 Wireless Network Modeling

Significant research has been done on wireless network modeling. One class of work focuses on asymptotic performance bounds (*e.g.*, [44,50,51,83]). The seminal work by Gupta and Kumar analyzes the capacity of a wireless network under certain traffic patterns and topologies [51]. Other researchers have since extended this work to other traffic patterns [83], mobility [50], and network coding [44]. These models provide useful insights as a network scales, but cannot be applied to a specific network. Another large class of models predict performance for a given scenario (*e.g.*, [18, 41, 43, 71, 113, 120]). They differ in their generality: some assume that everyone is within communication range of each other [18, 41, 43, 77], while others assume restricted traffic demands (*e.g.*, a single flow [41, 43], two flows [120], sending to a single neighbor [42], adding one new flow at a time [130], or one-hop demands [71, 113]).

2.2 Link Layer

The existing work of supporting reliable communication in the Link Layer uses one or a combination of the following techniques: (i) retransmissions, (ii) forward error correction (FEC), (iii) network coding, and (iv) channel reservation for reducing collision losses.

2.2.1 Retransmission

Retransmission is the most commonly used approach to recover packet errors and losses. Retransmissions require feedback from the receivers, specifying which packets are required for retransmissions. The feedback can be either ACKs or negative ACKs (NACKs) [1, 78]. The retransmission mechanism of 802.11 was presented in Section 1.2.

Reliable multicast has been proposed in SMACK [38]. The protocol works by having each receiver send a binary response of packet reception on an dedicated OFDM sub-channel.

2.2.2 FEC

FEC has been used to provide reliable unicast and multicast communication in both wireless networks (*e.g.*, [80, 96, 109, 123, 162, 168]) and wireline networks (*e.g.*, [19, 121, 122, 139]). For example, in [96], McKinley et al. dynamically adjust the level of FEC redundancy based on observed channel quality. [80] points out that many existing FEC-based works incorrectly assume independent packet losses, and studies the impact of spatial and temporal correlation of packet losses on FEC schemes. In addition to network performance, [168] analyzes the tradeoff between improving multicast throughput and minimizing power consumption when using FEC techniques.

2.2.3 Network coding

The pioneering work by Ahlswede et al. [8] shows that allowing relay nodes to encode and decode traffic can achieve maximum multicast rate, and this is generally more efficient than only allowing the relay nodes to forward traffic. Since then, lots of progress has been made in applying network coding to wireless and wireline networks (*e.g.*, [73, 75, 84, 86]). In particular, COPE [73] develops a practical network coding scheme for unicast in multi-hop wireless networks and [84] further extends the idea to broadcast. Both works focus on multihop wireless networks, and use network coding for the initial transmissions. COPE relies on MAC-layer retransmissions to recover packet losses, while [84] does not consider loss recovery. Finally, other schemes have used coding for loss recovery [67, 81, 160].

2.2.4 Channel reservation

One of the major sources of packet losses in wireless networks comes from packet collisions. For unicast traffic, binary exponential back-off and RTS/CTS are used to reduce collision losses and avoid hidden terminals. Due to expensive feedback, neither schemes are applicable to multicast/broadcast traffic [1] and the collision losses of multicast/broadcast traffic can be quite high. Motivated by this observation, several channel reservation schemes have been proposed to reduce collision losses for multicast traffic, such as Broadcast Support Multiple Access (BSMA) [143], Broadcast Medium Window (BMW) [144], Batch Mode Multicast MAC protocol (BMMM) [59], and Leader based Priority Ring Multicast Protocol (LPRMP) [34].

2.3 Physical Layer

There has been a wide variety of research that aims to prevent loss in the Physical Layer. Here, we classify previous works into three main categories: (i) characterization and measurements of wireless losses, (ii) efforts to build a more robust wireless link and (iii) automatically diagnosing faults in wireless networks.

2.3.1 Characterization of Wireless Losses

In this section, a brief summary of loss measurements in wireless networks is provided. In [7], the authors examine link loss in an 802.11b urban mesh network. They find loss rates stay mostly stable from one second to the next, links can exhibit varying degrees of bursty behavior, and a large number of links suffer from intermediate loss rates due to multi-path. In [118], the authors find the link loss variation is likely due to external interference rather than multi-path.

Numerous other works study loss. In [39], the authors characterize errors in WLANs by evaluating the effects of interference and attenuation due to distance and physical obstructions. In [136], the authors find some links exhibit correlation in packet delivery. This finding is also observed in [142]. In [54], the authors examine loss characteristics at a sub-frame level and find three main loss patterns consistent across various chipsets: the slope-line pattern, the saw-line pattern and the finger pattern.

Other works study burst losses in wireless networks. Burstiness in wireless losses is a well-known problem [23, 65, 79]. For instance, evidence of burstiness in wireless mesh networks is found in [7]. Subsequent works have found burstiness in indoor static wireless networks [120], industrial environments [154], as well as environments without RF interference [49]. In [99], the authors find burstiness exists in WLAN environments on the order of tens of frames. In their experiments, nodes are separated by 15 meters with no line-of-sight. Static and mobile receivers are evaluated and it is found that mobile receivers suffer from more frequent bursty losses than static receivers. They also found, in contrast to our results, that bursts from the static nodes were longer than bursts from the mobile nodes. In [137], the authors describe an algorithm, β, to measure link burstiness. Furthermore, they investigate the causes of link burstiness and show that a possible cause is variation in the signal-to-noise ratio in 802.15.4 networks. They find that bursty links are often links on the edge of reception sensitivity.

2.3.2 Building a Robust Wireless Link

There has been a wide variety of schemes that aim to build a dependable wireless link. The schemes fall into these main categories: autorate mechanisms to achieve the best rate, partial packet recovery to make use of the uncorrupted parts of a frame in error, diversity techniques, and interference cancellation.

Autorate There has been a very large body of work to pick the best modulation and coding schemes on a given wireless link. For example, there exists some work on 802.11a/b/g networks [21, 68, 103, 156], on 802.11n networks [33, 111], and some taking a cross layer approach [116, 151]. Each of these schemes try to find the rate that provides the highest throughput on the wireless link and do not necessary try to minimize loss.

Partial Packet Recovery Other schemes allow for partial packet reception [55,61, 64,89]. That is, they keep the error-free portions of the packet and only require the lost portions to be retransmitted. The 802.11n standard uses a similar approach in its packet aggregation technique: only packets that are lost within an aggregated frame need to be retransmitted.

Spatial Diversity Spatial diversity has been employed in order improve the quality and reliability of a wireless link. Information from multiple antennas can effectively be combined to reconstruct the transmitted wireless signal. Some schemes have leveraged receiver diversity to reduce retransmissions in WLAN environments. To reduce retransmissions in the down-link direction (from AP to client), PRO [92] defines an opportunistic retransmission protocol. It leverages overhearing nodes to retransmit data on behalf of a source when the original transmission fails. Schemes have also been proposed in the up-link direction. In MRD [98], erroneous packets received at an AP are compared with the same reception of the packet at different APs. A search over the erroneous portions is conducted in an effort to satisfy the CRC checksum. In SOFT [157], the confidence measure on individual bits is used to more efficiently combine multiple corrupted packets into a single correct packet. Diversity has been applied over multiple antennas for a single node [63, 101]. For example, techniques such as diversity combining (for example, Maximal Ratio Combining) and using multiple transmit antennas [11, 145, 146] can be employed.

Interference Cancellation Finally, there has been a large body of work that tries to mitigate collisions through interference cancellation [22, 57, 148, 150]. These techniques provide mechanisms to retrieve wireless data when the receiver receives data from multiple concurrent wireless transmissions. Some of these works have focused on WLANs [48, 52], while others have focused on cellular networks. An overview of schemes developed for cellular networks can be found in [12].

2.3.3 Diagnosing Wireless Losses

In [4], an architecture for detecting and diagnosing faults in WLAN environments is presented. A system-wide framework is deployed to monitor and assist wireless traffic when nodes are experiencing disconnectivity or performance issues. In [14], wired desktop clients are also employed to monitor and assist wireless clients.

In WiFiProfiler [25], clients cooperate with one another to diagnose and resolve network problems. The system contains a sensing component to passively monitor the connectivity status and configuration information of a node, a communicate component to facilitate client cooperation, and a diagnosis component to determine the likely cause of a fault.

The wit framework [95] merges multiple monitor traces to infer the status of wireless packets. Similarly, Jigsaw [28] is a distributed wireless monitoring platform that allows the merging of network events at several locations to present a unified view of all activity on an 802.11 network. In [27], the framework is utilized, along with network-side information, to reconstruct sources of delay in an 802.11 network. This information is then used to determine and diagnose faults.

2.4 Contrasting Completed Work from Related Work

In this section, we present a brief overview of how the proposed work differs from the related work.

2.4.1 Network Layer

Previous opportunistic routing schemes try route their traffic based on heuristics [20, 24, 129, 161]. However, these heuristics may not correlate well to actual network performance. In comparison to the pre-existing opportunistic routing schemes, our approach directly optimizes end-to-end performance by computing interference-aware opportunistic routes and rate limits. The performance optimized by our approach can be realized in a real network and is significantly better than the existing schemes. Also, as detailed in Section 3.4, the protocol to realize our approach in real networks is built on top of the MORE code-base. Therefore, schemes that improve MORE [76,82,90,91] can be used to make our routing protocol more efficient.

Like our work, some schemes propose to optimize opportunistic routing [93, 115, 135, 141, 165]. However, the interference model used in these works is the conflict-graph model described in Section 3.2. This model assumes packet transmissions can be precisely controlled and, hence, over-estimates performance in real networks (our results in Section 3.6 confirm this model significantly over-estimates the actual performance of the network). Furthermore, the interference models used only provide an aggregate view of broadcast interference: broadcast transmissions are assumed to interfere if any one of their receivers is interfered with by the other transmission. But providing a simple binary answer of interference does not fully characterize the impact of interference on different receivers and is therefore inadequate for use in optimizing opportunistic routing. Different from these works, we show that to achieve accurate optimization of network performance it is essential to use an accurate network model that captures the non-convex relationship between the performance of different wireless links. We develop an accurate model of broadcast interference to capture these relationships. Since the relationships are non-linear and non-convex, we also develop iterative procedure for non-convex optimization. The algorithm provides us with a local optimal solution and can be used to answer the question: is it better to use a global solution with a less accurate model (the conflict-graph model), or instead find a local optimal solution with an accurate model? To help answer this, in addition to a new interference model and model-based optimization, our work goes beyond theoretical analysis (the primary focus of the above works) by developing a practical routing protocol to realize the performance gains in a real IEEE 802.11 network. This allows us to answer the previously posed question. We find our model is highly accurate: it achieves within 20% of the estimated performance, while the conflict-graph approach consistently over-predicts. Furthermore, our model provides performance benefits: 2-13 times ETX's throughput, 1.5-10 times MORE's throughput and 10%-46% better throughput than the conflict-graph model.

Finally, Section 2.1.3 lists previous work that has focused on modeling wireless networks. Most of the models in that section predict performance under a given scenario and cannot support optimization without enumerating all possible network configurations, which is prohibitive due to an intractable search space. To facilitate optimization, we need a model that can specify the entire region of feasible network configurations using a compact set of constraints, which can then be incorporated into the optimization procedure to optimize the desired objective within the feasible region. Two existing models are in this category: (i) the conflict-graph-based model [62], and (ii) the unicast interference model [85]. We discuss why [85] is insufficient for optimizing opportunistic routing in Section 3.2.1.

2.4.2 Link Layer

Much of the related work in this area is complementary to our approach. The SMACK [38] scheme that provides reliable broadcast can be utilized to implement receiver feedback in ER. SMACK, however, may require changes to current off-the-shelf devices, whereas ER is designed to be used with commonly deployed hardware. The FEC protocols are complementary to loss recovery schemes using retransmissions with or without source/network coding, and can be used in combination with ER. Furthermore, the channel reservation schemes reduce collision losses, while the retransmissions can recover both collision and other wireless medium related losses (*e.g.*, those due to fading and low SNR). Finally, ER can be augmented to protocols such as COPE to provide coding for retransmitted packets.

The schemes most similar to ER are [67, 81, 160]. These schemes also employ coding for retransmitting data to multiple users. None of these works, however, provide a detailed protocol description nor implement their scheme. In ER, we develop a practical protocol that can be utilized in today's IEEE 802.11 networks. We implement and evaluate our approach in a simulator and on a testbed, whereas the previous approaches study and derive the problem analytically.

2.4.3 Physical Layer

In contrast to works that have measured and characterized wireless losses, our work investigates losses when the signal quality is high and benefits from a strong line-of-sight. We find that moderate mobility can cause burst losses even if the signal isn't low or near the edge of the reception sensitivity. The previous approaches have only identified burst losses when the signal strength is low or highly variable. We find these losses occur independent of the receiver and sender configurations. Furthermore, our case study presents an in-depth analysis to characterize these burst losses, gains insights into why they may be happening, and makes recommendations to rectify the problem.

Many of the previous schemes aiming to build a better wireless link try to minimize the number of corrupted bits within a packet. We share this goal and can harness existing schemes for this problem. However, through our analysis of burst errors in dynamic environments, we find many of the burst errors result from the packets getting lost completely (as opposed to be received with a few corrupted bits). This wreaks havoc on wireless links that must provide high throughput and low latency. Our motivating example and prime interest is to understand the bounds of wireless communication so as to enable a rich ecosystem of wireless technologies. We therefore make recommendations to curtail wireless losses without sacrificing the need for high throughput and low latency.

Chapter 3

Network Layer

In this chapter, our scheme for optimizing opportunistic routing is presented. The contents of this chapter represent work that appeared in the proceedings of the ACM SIGMETRICS 2011 conference [127].

3.1 Optimization Framework

Overview: We adopt network coding introduced by MORE [24] to prevent forwarders from forwarding redundant information without fine-grained coordination. In this framework, an opportunistic route for a flow f from a source s to a destination d is defined by the sending rate at s, denoted as $T(f, s)$, and the forwarding rate at node j upon receiving a packet from node i, denoted as $F(f, i, j)$. Single path routing is a special case of opportunistic routing where $F(f, i, j) = 1$ if j is i's next hop and 0 otherwise, whereas the opportunistic route for a given packet is determined on the fly based on who receives the transmission and the $F(f, i, j)$ values at the nodes. Below we present our general framework for jointly optimizing opportunistic routes and rate limits (*i.e.*, $T(f, s)$ and $F(f, i, j)$). Our optimization outputs $T(f, i)$ and $Y(f, d, i, j)$, which will be converted to $F(f, i, j)$ using the credit computation described in Section 3.4.

$Flows$	the set of unicast or multicast flows
$src(f)$	source of flow f
$dest(f, d)$	d-th destination of flow f
$Demand(f)$	traffic demand of flow f, *i.e.*, the amount of traffic f desires to send
$G(f)$	throughput of flow f
$T(f, i)$	node i's sending rate for flow f
$Y(f, d, i, j)$	information receiving rate along link $i-j$ for d-th destination in flow f ($d = 1$ for unicast)
$P(i, j)$	loss rate of link $i - j$ (including both collision and inherent wireless medium loss)
$\mathcal{N}(i)$	a subset of i's neighbors
$S(i, \mathcal{N}(i))$	success rate from node i to i's neighbor set $\mathcal{N}(i)$

Table 3.1: Notations for optimizing opportunistic routing.

Without loss of generality, we focus on multicast flows, since unicast flows are a special case of multicast with one receiver in each multicast group. The main design issue becomes how fast each traffic source should send traffic and how much traffic an intermediate node should forward to achieve high performance. This can be formulated as an optimization problem that maximizes total network throughput subject to information conservation constraints, opportunistic constraints, and interference constraints. Figure 3.1 shows the resulting formulation, and Table 3.1 specifies the variables in the formulation. Below we explain the formulation.

Optimization objective: The first term in the objective, shown in Figure 3.1, $\sum_{f \in Flows} G(f)$, reflects the goal of maximizing the total throughput over all flows. The second term in the objective, $-\beta \sum_{f,i} T(f, i)$ represents the total amount of

$\triangleright\, Input:\ Flows,\ Demand(f)$
$\triangleright\, Output:\ T(f,i),\ Y(f,d,i,j)$

$$\textbf{maximize:} \sum_{f \in Flows} G(f) - \beta \sum_{f,i} T(f,i)$$

subject to:

$$\begin{array}{ll}
\text{[C1]} & G(f) \leq Demand(f) \qquad\qquad (\forall f) \\
\text{[C2]} & G(f) \leq \sum_{k} Y(f,d,k,dest(f,d)) \qquad (\forall f,d) \\
\text{[C3]} & Y(f,d,k,src(f)) = 0 \qquad\qquad (\forall f,d,k) \\
\text{[C4]} & Y(f,d,dest(f,d),k) = 0 \qquad\qquad (\forall f,d,k) \\
\text{[C5]} & \sum_{k} Y(f,d,k,i) \geq \sum_{j} Y(f,d,i,j) \\
 & \qquad (\forall i \neq src(f) \text{ and } i \neq dest(f,d)) \\
\text{[C6]} & S(i,\mathcal{N}(i))T(f,i) \geq \sum_{k \in \mathcal{N}(i)} Y(f,d,i,k) \quad (\forall i, \mathcal{N}(i)) \\
\text{[C7]} & interference\ constraints\ on\ T(f,i)
\end{array}$$

Figure 3.1: Problem formulation to optimize multicast throughput of opportunistic routing.

wireless traffic. Including both terms reflects the goals of (i) maximizing total throughput and (ii) preferring the least amount of traffic among all solutions that support the same total throughput (*e.g.*, avoiding loops and unnecessary traffic). Since the first objective is more important, we use a small weighting factor $\beta = 1e - 5$ for the second term just for tie breaking (*i.e.*, only when the first objective is the same, we prefer the one with the least traffic).

To compute the first term, for a unicast flow f, $G(f)$ is its throughput. For a multicast flow f, $G(f)$ is the throughput of the bottleneck receiver. While here we focus on total throughput, our framework can be directly applied to optimizing other linear objectives. For example, our evaluation also considers optimizing a linear approximation of proportional fairness, defined as $\sum_{f \in Flows} \log G(f)$, which strikes a good balance between fairness and throughput [119]. We can also maximize total revenue if the revenue of a flow is a linear function of its throughput. In addition, it is easy to apply our framework (with small modifications) to optimize total throughput over all receivers in the multicast groups.

Throughput constraints: To ensure $G(f)$ is the throughput of flow f, it has to satisfy constraints (C1) and (C2) in Figure 3.1. Constraint (C1) indicates that the throughput of a flow should be no more than its traffic demand (*i.e.*, total amount of information a source desires to send). Constraint (C2) ensures that $G(f)$ is no more than the total amount of information delivered from all links incident to the destination of flow f. For a multicast flow f, $G(f)$ should be no more than the total amount of information delivered to each destination in the flow f. Note that we do not need a lower bound on $G(f)$ since the objective is to maximize $G(f)$.

Information conservation constraints: To handle lossy wireless links, we distinguish traffic and information sent along a link. A feasible routing solution should satisfy information conservation. This property is given by constraints (C3–C5) in Figure 3.1. Constraint (C3) ensures no incoming information to a traffic source, constraint (C4) ensures no outgoing information from a destination, and constraint (C5) represents flow conservation at an intermediate node i, *i.e.*, the total amount of incoming information is no less than the total amount of out-going information.

Opportunistic constraints: Opportunistic routing exploits the wireless broadcast medium by having different nodes extract information from the same transmission. We formally capture this notion using opportunistic constraints, which relate traffic volume to the amount of information delivered.

For ease of explanation, we first consider one sender sending to two receivers, and then generalize it to an arbitrary number of receivers. Consider a sender s and denote the link loss rates from s to its neighbors r_1 and r_2 as $P(s, r_1)$ and $P(s, r_2)$, respectively. It is evident that for a given flow the amount of information delivered to a neighbor is bounded by the product of the sending rate and link delivery ratio. Therefore we have $(1 - P(s, r_1))T(f, s) \geq Y(f, d, s, r_1)$ and $(1 - P(s, r_2))T(f, s) \geq Y(f, d, s, r_2)$. In addition, since there is overlap between the information delivered to r_1 and r_2 and we are only interested in the non-overlapping information (*i.e.*, when redundant information is delivered to both nodes, it should only count once). The total non-overlapping information delivered to r_1 and r_2 should satisfy the following constraints:

$$(1 - P(s, r_1)P(s, r_2))T(f, s) \geq \sum\nolimits_{i \in \{1,2\}} Y(f, d, s, r_i)$$

where the left hand-side represents the total amount of traffic successfully delivered to at least one of the receivers, and the right hand-side represents the total non-overlapping information delivered to the receivers.

Now we consider a general setting, where a sender s has N neighbors. We enumerate all possible subsets of its neighbors. For each neighbor set $\mathcal{N}(i)$, we require:

$$S(i, \mathcal{N}(i))T(f, i) \geq \sum\nolimits_{k \in \mathcal{N}(i)} Y(f, d, i, k)$$

where $S(i, \mathcal{N}(i)) = 1 - \prod_{k \in \mathcal{N}(i)} P(i, k)$ gives the probability for i's transmission to reach at least one node in $\mathcal{N}(i)$. The constraint indicates the total traffic successfully delivered to at least one neighbor in $\mathcal{N}(i)$ should be no less than the total non-overlapping information delivered to $\mathcal{N}(i)$. This results in (C6) in Figure 3.1. When i has many (say, K) neighbors, we limit the number of such constraints by only enumerating neighbor sets of size 1, 2, and K (*i.e.*, we enumerate only O(K^2), instead of O(2^K) neighbor sets).

Interference constraints: Wireless interference has significant impact on wireless network performance. In particular, nearby senders carrier sense and defer to each other. Moreover, since carrier sense is not perfect, there may be multiple overlapping nearby transmissions and cause collisions. These effects can further constrain the amount of traffic on each link and introduce strong inter-dependency between sending rates, loss rates, and throughput. We address this issue in Section 3.2 by developing the constraints that capture the relationships between $T(f, i)$ and $P(i, j)$.

3.2 Broadcast Interference Model

In this section, we first motivate the need for a better interference model and then present our new model.

3.2.1 Motivation for a Better Model

Despite significant research on modeling the impact of wireless interference, none of the existing models fulfill our need for optimizing opportunistic routing. To support optimization, we need a model that specifies the feasible region of network configurations using a compact representation. The following two models fall into this category.

Conflict-graph-based model: The first model, proposed in [62], is a conflict-graph-based model that represents wireless links as vertices and draws a conflict edge between two vertices if the corresponding wireless links interfere. Based on this definition, it is clear that links corresponding to an independent set in the conflict graph can be active simultaneously. Therefore, the interference constraints are the schedule restrictions imposed by the independent sets, which can be expressed as a set of linear constraints.

There are two limitations in applying the conflict-graph-based model for optimizing opportunistic routing. First, the model in [62] assumes perfect scheduling, *i.e.*, packet transmissions at different nodes can be precisely controlled and it overestimates the performance in real networks as we will show in Section 3.6. Second, the conflict-graph-based model is a link-based model, while opportunistic routing uses broadcast transmissions and requires a node-based broadcast model. Existing broadcast extensions of the conflict-graph model provide only an aggregate answer of whether two broadcast transmissions interfere or not. For example, some extensions [115, 134, 163, 165] conservatively consider two broadcast transmissions to interfere if any one of their receivers is interfered by the other transmission, while other extensions [163] consider broadcast transmissions to interfere if all of their receivers are interfered by the other transmission. A single aggregate answer on whether broadcast transmissions interfere does not fully characterize the impact of interference on different receivers and is therefore inadequate for use in optimizing opportunistic routing.

IEEE 802.11 unicast model: The other model, proposed in [85], models interference among unicast transmissions in IEEE 802.11. Since opportunistic routing uses broadcast traffic, we need to develop interference models for broadcast transmissions. Furthermore, as broadcast transmissions does not perform binary backoff to limit the sending rate, it is necessary to have an accurate model even for high traffic load and channel occupancy, which induces high collision losses, and the linear approximation used in [85] becomes inaccurate under high collision losses. In addition, [85] is used for rate limiting unicast transmissions when given specified routes. Therefore it suffices to accurately estimate the sending rates and loss rates on a small number of links used for routing. In contrast, for the purpose of route optimization, we need to accurately estimate the performance for *all* receivers of a given sender, which is much more challenging.

Modeling goals and strategy: We design our model specifically for IEEE 802.11 broadcast traffic. We observe that wireless interference affects IEEE 802.11 traffic in two important ways: (i) nearby senders cannot transmit simultaneously due to carrier sense, and (ii) transmissions may sometimes result in collisions due to imperfect carrier sense. We model these effects by developing the relationships between sending rates, loss rates, and throughput, which can be incorporated into our optimization framework and facilitate model-driven optimization. While this chapter applies the model to optimizing opportunistic routing, the model is useful in other contexts (*e.g.*, optimizing network topology and network planning). Our model is general and captures real-world complexities (*e.g.*, hidden terminals, multi-hop flows, non-binary interference, and heterogeneous traffic), which is confirmed by simulation and testbed experiments using multihop networks in Section 3.6. Compared with [85], both our sender model (Section 3.2.3.1) and loss model (Section 3.2.3.2) are much more refined and do not involve any linear approximation. As a result, our model can accurately estimate the loss rates for all receivers even under heavy traffic loads, which is essential for the optimization of opportunistic routing.

3.2.2 Background

We first review the broadcast transmissions as specified by the IEEE 802.11 standard [1]. Before transmission, a sender first checks to see if the medium is available using carrier-sensing. A sender determines the channel to be idle when the total energy received is less than the clear-channel assessment threshold. In this case, a sender may begin transmission using the following rule: If the medium has been idle for longer than a distributed inter-frame spacing time (DIFS) period, transmission can begin immediately. Otherwise, a sender waits for DIFS and then waits for a random backoff interval uniformly chosen between $[0, CW_{\min}]$, where $CW_{\min}$ is the minimum contention window.

3.2.3 Our New Model

We develop a simple interference model to capture the interdependency between broadcast sending rates, loss rates, and throughput. Such interdependency can be captured using $O(N)$ constraints, where N is the total number of nodes in the network. These constraints can then be incorporated into the optimization problem as interference constraints shown in Figure 3.1. We present methods to measure the input parameters of the model in Section 3.4.

Our model consists of two main components: (i) a *sender model* that captures the effects of carrier-sensing on a sender's sending rate, and (ii) a *loss model* that captures both inherent loss (*i.e.*, packet loss under no interference) and the effects of overlapping packet transmissions on the collision loss rates for different links.

3.2.3.1 Broadcast Sender Model

Modeling the effects of carrier sense on traffic rates: We divide time into *variable-length slots* (VLS) for each sender i. A variable-length slot may last for either IEEE 802.11 slot time T_{slot} or the transmission time of a packet followed by a DIFS duration. The former occurs when i senses a clear channel but either has no data to transmit or has data but cannot transmit due to a non-zero backoff counter. The latter occurs when i either transmits a packet or waits for a transmission from another sender to complete.

Let τ_i be the probability for i to start a new packet transmission in a variable-length slot. Clearly, τ_i depends on (i) how often i has data to send, and (ii) the random backoff interval (*i.e.*, $CW_{\min}$). As derived in [18], when i has saturated traffic demand (*i.e.*, it always has data to transmit), on average i performs one transmission every $CW_{\min}/2+1$ variable-length slots (since there is no exponential backoff for broadcast traffic, we have $CW_{\min}/2$ slots for backoff plus 1 slot for the transmission). Therefore, the transmission probability τ_i is bounded by the following *feasibility constraint*:

$$\tau_i \leq \tau_{\max} \triangleq \frac{1}{CW_{\min}/2+1} \quad (\text{for } \forall i) \tag{3.1}$$

We assume pairwise interference, *i.e.*, the interference relationship between two links is independent of activities on other links. Previous works show that pairwise interference is good approximation in real networks [6, 105]. Hence this assumption has been widely used in the literature (*e.g.*, [18, 41, 43, 85, 120]). Moreover, for the purpose of optimizing the performance of multi-hop wireless networks, it is often more important to capture the interference relationship among links that are not too far apart. For these links, pairwise interference is likely to be an even better approximation.

Under the pairwise interference model, whether sender i carrier-senses (and thus defers to) an ongoing transmission of sender j only depends on nodes i and j and is independent of if other senders are transmitting. Let D_{ij} be this conditional deferral probability (*i.e.*, probability for node i to defer to node j when node j is transmitting). For convenience, let $D_{ii} = 1$. Let T_i be sender i's sending rate over all flows ($T_i = \sum_f T(f,i)$), VLS_i be its expected VLS duration, and P_i^{idle} be the idle probability of node i. T_i, VLS_i and τ_i have the following approximate relationship, called the *throughput constraints*:

$$\begin{aligned} T_i &= (EP \times \tau_i)/VLS_i && (3.2) \\ VLS_i &= T_{\text{slot}} P_i^{\text{idle}} + (T_{\text{xmit}} + T_{\text{DIFS}})(1 - P_i^{\text{idle}}) && \\ &= T_{\text{slot}} + (T_{\text{xmit}} + T_{\text{DIFS}} - T_{\text{slot}})(1 - P_i^{\text{idle}}) && (3.3) \\ P_i^{\text{idle}} &= \prod_j (1 - D_{ij} \times \tau_j \times \frac{VLS_i}{VLS_j}) && (3.4) \end{aligned}$$

where EP is the expected packet payload size, EH is expected header size, $T_{\text{xmit}} = (EP + EH)/rate$ is the expected packet transmission time, and T_{slot} is an IEEE 802.11 slot time. Eq. (3.2) computes throughput as the total amount of payload transmitted during one VLS divided by the expected VLS duration. Eq. (3.3) computes expected VLS duration as idle probability times an idle slot duration plus transmission (including collision) probability times a transmission duration. Finally, Eq. (3.4) gives the probability that i finds the medium is idle, where $\tau_j \times \frac{VLS_i}{VLS_j}$ is the probability for j to start transmission in i's VLS, $D_{ij} \times \tau_j \times \frac{VLS_i}{VLS_j}$ is the probability that i defers to j's transmission, and $\prod_j (1 - D_{ij} \times \tau_j \times \frac{VLS_i}{VLS_j})$ is the probability that i does not defer to any node in the network including its own transmission (*i.e.*, i senses the medium is idle).

Reducing the number of model parameters: To better facilitate model-driven optimization, we transform the relationships in (3.1)–(3.4) into the following equivalent constraints that apply directly to the traffic rates $\{T_i\}$ by eliminating $\{\tau_i\}$ and $\{P_i^{\text{idle}}\}$.

- *Feasibility constraint.* According to Eq. (3.2), we have: $\tau_i = \frac{T_i * VLS_i}{EP}$. As a result, Eq. (3.1) is equivalent to:

$$\frac{T_i}{EP} \leq \frac{\tau_{\max}}{VLS_i} \quad (\text{for } \forall i). \tag{3.5}$$

- *Throughput constraint.* With $\tau_i = \frac{T_i * VLS_i}{EP}$, Eq. (3.4) becomes: $P_i^{\text{idle}} = \prod_j \left(1 - \frac{D_{ij} * T_j * VLS_i}{EP}\right)$. So Eq. (3.3) becomes:

$$\begin{aligned} VLS_i = T_{\text{slot}} \quad + & \quad (T_{\text{xmit}} + T_{\text{DIFS}} - T_{\text{slot}}) * \\ & \left[1 - \prod_j \left(1 - \frac{D_{ij} * T_j * VLS_i}{EP}\right)\right]. \end{aligned} \tag{3.6}$$

Eq. (3.5) and (3.6) fully capture the relationships in (3.1)–(3.4) but have fewer variables. Moreover, note that when traffic rates $\{T_j\}$ are given as inputs, Eq. (3.6) contains only a single variable: VLS_i. This allows us to numerically derive VLS_i and partial derivatives $\frac{\partial VLS_i}{\partial T_j}$ from the given $\{T_j\}$ (as described in Section 3.3.2). We will therefore use (3.5) and (3.6) in our model-driven optimization.

3.2.3.2 Broadcast Loss Model

Integrating inherent loss and collision loss: To estimate loss rates $P(i, j)$ from traffic rates T_i, we distinguish between two types of loss: inherent wireless medium loss (*i.e.*, loss rate under no interference) and collision loss. The former is denoted as $P^{\text{raw}}(i, j)$ for link $i - j$ and can be periodically measured. The latter depends on two factors: (i) how often transmissions from different nodes overlap and (ii) how often such overlapping transmissions result in a collision. To capture the first effect, we introduce $O(i, k)$ to denote the probability for an i's transmission to overlap with a k's transmission (conditioned on i's transmission) and derive its value based on the deferral probability. To capture the second effect, we observe that the pairwise interference model indicates there is a constant conditional collision loss probability L^k_{ij} (*i.e.*, the probability that a transmission on link $i-j$ collides with an overlapping transmission from node k). We assume that inherent wireless medium loss and collision loss are independent, which has been commonly used (*e.g.*, [85,113]). We then compute the combined loss rate as follows:

$$P(i, j) = 1 - (1 - P^{\text{raw}}(i, j)) \times \prod_{k \neq i} \left[1 - L^k_{ij} \times O(i, k)\right]$$

This is because a packet is delivered when it is not lost due to either inherent loss or collision loss. To ensure no collision, the packet should not collide with any node's transmission. Since $L^k_{ij} \times O(i, k)$ is the collision loss probability with node

k's transmission, $\prod_{k \neq i} \left[1 - L_{ij}^{k} \times O(i,k)\right]$ is the probability that the link has no collisions with any other node in the network.

Estimating overlap probabilities: We next estimate the overlap probabilities $O(i,j)$, which depends on whether i and j can carrier sense each other. Our model has two salient features: (i) it supports both symmetric and asymmetric deferral (*e.g.*, node i defers to node j but not vice versa), and (ii) it handles non-binary deferral (*e.g.*, node i sometimes defers to j and sometimes does not).

To provide both features, our modeling strategy is to divide time into regions to which one of the following four cases applies:

- Case 1: i and j can both carrier sense each other;
- Case 2: neither i nor j can carrier sense each other;
- Case 3: i can carrier sense j but j cannot carrier sense i; and
- Case 4: i cannot carrier sense j but j can carrier sense i.

Let $Q_c(i,j)$ be the probability for Case c to occur. Let $O_c(i,j)$ be the probability for a transmission of i to overlap with any transmission of j under Case c. We then have:

$$O(i,j) = \sum_{c=1}^{4} (Q_c(i,j) \times O_c(i,j)). \tag{3.7}$$

Assuming whether i can carrier sense j is independent of whether j can carrier sense i, we derive $Q_c(i,j)$ as follows.

$$\begin{aligned}
Q_1(i,j) &= D_{ij} \times D_{ji}, \\
Q_2(i,j) &= (1-D_{ij}) \times (1-D_{ji}), \\
Q_3(i,j) &= D_{ij} \times (1-D_{ji}), \\
Q_4(i,j) &= (1-D_{ij}) \times D_{ji}.
\end{aligned}$$

In Section 3.2.3.3, we show $O_c(i,j)$ can be computed as follows.

$$\begin{aligned}
O_1(i,j) &= \tau_j, \\
O_2(i,j) &= 1-(1-\theta_j)\exp\left[-T_{\text{xmit}}/IPD_j\right], \\
O_3(i,j) &= 1-\exp\left[-T_{\text{xmit}}/IPD_j\right], \\
O_4(i,j) &= \frac{\theta_j}{\theta_j+(1-\theta_j)\exp\left[-T_{\text{xmit}}/IPD_j\right]},
\end{aligned}$$

where $\theta_j = \frac{T_j}{rate} \times \frac{EP+EH}{EP}$ is the fraction of time j is transmitting (either payload or header) and $IPD_j \triangleq \frac{1-\theta_j}{\theta_j} \times T_{\text{xmit}}$ is j's expected inter-packet delay.

3.2.3.3 Deriving Overlap Probabilities

In this section, we derive the overlap probabilities $O_c(i,j)$ $(c = 1,2,3,4)$ used in Eq. (3.7). Let $\theta_j = \frac{T_j}{rate} \times \frac{EP+EH}{EP}$ be the fraction of time j is transmitting (either payload or header) and $IPD_j \triangleq \frac{1-\theta_j}{\theta_j} \times T_{\text{xmit}}$ be the expected inter-packet delay of j. We assume that j's inter-packet delay has an exponential distribution.

Case 1: i and j can both carrier sense each other. In this case, i's transmission overlaps with j's transmission if and only if they both start transmitting within the same idle slot. At a given idle slot, the probability for both i and j to start transmitting is simply $\tau_i \times \tau_j$. Therefore, we have: $O_1(i,j) = \frac{\tau_i \times \tau_j}{\tau_i} = \tau_j$.

Case 2: neither i nor j can carrier sense each other. In this case, in order for i's transmission not to overlap with any of j's transmissions, two conditions must hold: (C1) i's transmission must start when j is not transmitting anything, and (C2) j's next transmission starts at least T_{xmit} after i's transmission. Therefore, $O_1(i,j) = 1 - \textsf{prob}\{C1 \wedge C2\} = 1 - \textsf{prob}\{C1\} \cdot \textsf{prob}\{C2|C1\}$.

The probability for C1 to hold is simply $\textsf{prob}\{C1\} = (1-\theta_j)$. To derive $\textsf{prob}\{C2|C1\}$, we assume the inter-packet delay of j has an exponential distribution with mean equals IPD_j. The memoryless property of exponential distribution ensures that if j is not transmitting when i's transmission starts, the delay between the start time of i's current transmission and the start time of j's next transmission also has an exponential distribution with the same mean IPD_j. The probability for this delay to exceed T_{xmit} is simply $\textsf{prob}\{C2|C1\} = \exp\left[-T_{\text{xmit}}/IPD_j\right]$. Therefore, we have

$$\begin{aligned} O_2(i,j) &= 1 - \textsf{prob}\{C1\} \cdot \textsf{prob}\{C2|C1\} \\ &= 1 - (1-\theta_j)\exp\left[-T_{\text{xmit}}/IPD_j\right]. \end{aligned}$$

Case 3: i can carrier sense j but j cannot carrier sense i. In this case, i always starts transmission when j is idle. In order for i's transmission not to overlap with j's transmission, the delay between the start time of i's transmission and the start time of j's next transmission must exceed T_{xmit}. As derived in Case 2, the probability for this to occur is simply $\mathsf{prob}\{C2|C1\} = \exp\left[-T_{\text{xmit}}/IPD_j\right]$. Therefore, we have:

$$O_3(i,j) = 1 - \exp\left[-T_{\text{xmit}}/IPD_j\right].$$

Case 4: i **cannot carrier sense** j **but** j **can carrier sense** i**.** There are three possible relative positions of i and j's transmissions: (S1) i transmits in the middle of j's transmission, (S2) i starts and finishes its transmission during j's idle time, and (S3) i starts transmitting when j is idle but does not finish before j starts transmitting. Since j can carrier sense i, (S3) is not possible. So we only need to consider (S1) and (S2). (S1) causes i's transmission to overlap with j's transmission, whereas (S2) does not result in any overlapping transmission.

It is evident that $\mathsf{prob}\{S1\} = \theta_j$. Meanwhile, note that $\mathsf{prob}\{S2\}$ is identical to $\mathsf{prob}\{C1 \wedge C2\}$ in Case 2. So we have $\mathsf{prob}\{S2\} = (1-\theta_j)\exp\left[-T_{\text{xmit}}/IPD_j\right]$. Therefore, we have:

$$\begin{aligned} O_4(i,j) &= \frac{\mathsf{prob}\{S1\}}{\mathsf{prob}\{S1\} + \mathsf{prob}\{S2\}} \\ &= \frac{\theta_j}{\theta_j + (1-\theta_j)\exp\left[-T_{\text{xmit}}/IPD_j\right]}. \end{aligned}$$

3.2.3.4 Deriving Expected VLS Duration

In this section, we present the details on how to derive VLS_i from given traffic rates $\{T_j\}$ according to Eq. (3.6).

Let

$$f_i(x) \triangleq x - T_{\text{slot}} \quad - \quad (T_{\text{xmit}} + T_{\text{DIFS}} - T_{\text{slot}}) * \left[1 - \prod_j \left(1 - \frac{D_{ij} * T_j * x}{EP}\right)\right] \tag{3.8}$$

According to Eq. (3.6), we have $f_i(VLS_i) = 0$. That is, $x = VLS_i$ is a root of $f_i(x)$.

Moreover, we need $x = VLS_i \in \left[0, \frac{EP}{\max_j(D_{ij} * T_j)}\right]$ to make sure that each $\left(1 - \frac{D_{ij} * T_j * x}{EP}\right) \geq 0$ in Eq. (3.6).

Theorem 1. *$f_i(x)$ is convex when $x \in \left(-\infty, \frac{EP}{\max_j(D_{ij} * T_j)}\right]$.*

Proof. Clearly, $f_i(x)$ is continuously differentiable. Let $f_i'(x)$ be the derivative of $f_i(x)$. We have:

$$f_i'(x) = 1 \quad - \quad (T_{\text{xmit}} + T_{\text{DIFS}} - T_{\text{slot}}) * \sum_k \left[\frac{D_{ik} * T_k}{EP} \prod_{j \neq k} \left(1 - \frac{D_{ij} * T_j * x}{EP}\right)\right] \tag{3.9}$$

It is easy to verify that $f_i'(x)$ is monotonically increasing for $x \in \left(-\infty, \frac{EP}{\max_j(D_{ij} * T_j)}\right]$. Therefore, $f_i(x)$ is convex over interval $x \in \left(-\infty, \frac{EP}{\max_j(D_{ij} * T_j)}\right]$. □

Lemma 1. *$f_i(x)$ has at most 2 roots for $x \in \left(-\infty, \frac{EP}{\max_j(D_{ij} * T_j)}\right]$.*

Proof. Since $f_i'(x)$ is continuous and monotonically increasing with $x \in \left(-\infty, \frac{EP}{\max_j(D_{ij} * T_j)}\right]$, $f_i'(x) = 0$ has at most one solution. So we only need to consider two cases:

- *Case 1*: $f_i'(x) \neq 0$ for $\forall x \in \left(-\infty, \frac{EP}{\max_j(D_{ij}*T_j)}\right]$. In this case, $f_i'(x)$ must be either always positive or always negative (because any change of signs would immediately imply a root of $f_i'(x)$ due to the continuity of $f_i'(x)$). As a result, $f_i(x)$ must be monotonic for $x \in \left(-\infty, \frac{EP}{\max_j(D_{ij}*T_j)}\right]$. Therefore, $f_i(x)$ can have at most one root over interval $\left(-\infty, \frac{EP}{\max_j(D_{ij}*T_j)}\right]$.
- *Case 2*: $\exists x_0 \in \left(-\infty, \frac{EP}{\max_j(D_{ij}*T_j)}\right]$ such that $f_i'(x_0) = 0$. Since $f_i'(x)$ is monotonically increasing, we have $f_i'(x) > 0$ with $x > x_0$ and $f_i'(x) < 0$ with $x < x_0$. By the same reasoning as in Case 1, we know that $f_i(x)$ has at most one root on each side of x_0. So $f_i(x)$ has at most 2 roots.

Combining Case 1 and Case 2, we complete our proof that $f_i(x)$ has at most 2 roots for $x \in \left(-\infty, \frac{EP}{\max_j(D_{ij}*T_j)}\right]$. □

Theorem 2. *$f_i(x)$ has at most one root for $x \in \left[0, \frac{EP}{\max_j(D_{ij}*T_j)}\right]$.*

Proof. Since $f_i(x)$ is a polynomial, we only need to consider the case when its degree is at least 2. This implies that there exist at least 2 different j's such that $D_{ij}T_j > 0$. In this case, it is easy to verify that $\lim_{x\to-\infty} f_i(x) = +\infty$. Meanwhile, we have $f_i(0) = -T_{\text{slot}} < 0$. Since $f_i(x)$ is continuous, it has at least one root for $x \in (-\infty, 0)$. However, according to Lemma 1, $f_i(x)$ has at most 2 roots for $x \in \left(-\infty, \frac{EP}{\max_j(D_{ij}*T_j)}\right]$. Therefore, $f_i(x)$ has at most one root for $x \in \left[0, \frac{EP}{\max_j(D_{ij}*T_j)}\right]$. □

Given Theorem 1 and 2, we can apply any classic univariate root-finding algorithm to numerically compute the root of $f_i(x)$ over interval $x \in \left[0, \frac{EP}{\max_j(D_{ij}*T_j)}\right]$

```
▷ $\mathbf{T}$: traffic rates, $\mathbf{Y}$: information, $\mathbf{P}$: loss rates
initialization: $\mathbf{T}^* = \mathbf{0}, \mathbf{Y}^* = \mathbf{0}, thruput^* = 0$
for $k = 1$ to $KMAX$
    $\mathbf{P}^* =$ estimate_loss($\mathbf{T}^*$)
    [$\mathbf{VLS}^*, \frac{\partial \mathbf{VLS}^*}{\partial \mathbf{T}^*}$] = estimate_VLS_and_partial_derivatives($\mathbf{T}^*$)
    derive linearized interference constraints in Eq. (3.10) using $\mathbf{VLS}^*$ and $\frac{\partial \mathbf{VLS}^*}{\partial \mathbf{T}^*}$
    construct a linear program (LP$_k$) from Figure 3.1 by adding linearized
    interference constraints (3.10), and fixing loss rates $\mathbf{P} = \mathbf{P}^*$ as constants
    solve (LP$_k$); let ($\mathbf{T}^{\text{opt}}, \mathbf{Y}^{\text{opt}}$) be the optimal solution
    $\alpha = \alpha_{\max}$; $succ$ = false
    while ($\alpha \geq \alpha_{\min}$) and ($succ$ = false) // line search for a better solution
        $\mathbf{T} = (1 - \alpha) \times \mathbf{T}^* + \alpha \times \mathbf{T}^{\text{opt}}$
        $feasible$ = test_traffic_rates_feasibility($\mathbf{T}$)
        if ($feasible$)
            [$thruput, \mathbf{Y}$] = compute_OR_thruput_from_traffic_rates($\mathbf{T}$)
            if ($thruput > thruput^*$)
                $thruput^* = thruput$; $\mathbf{T}^* = \mathbf{T}$; $\mathbf{Y}^* = \mathbf{Y}$;
                $succ$ = true; break
            end
        end
        $\alpha = \alpha/2$
    end
    if ($succ$ = false), break; end
end
return ($thruput^*, \mathbf{T}^*, \mathbf{Y}^*$)
```

Figure 3.2: Iterative optimization of opportunistic routing.

and let the solution be VLS_i (if such a solution exists). In our current implementation, we use the Matlab built-in function fzero to compute the root VLS_i through bisection search.

3.3 Model-Driven Optimization

3.3.1 Iterative Model-driven Optimization

A key challenge in optimizing opportunistic routing is that the relationships between $\{T_i\}$, $\{VLS_i\}$ and $\{P(i,j)\}$ are not linear or convex. To address this challenge, we perform optimization in an iterative fashion, as illustrated in Figure 3.2. To decouple the non-linear inter-dependency between $\{T_i\}$, $\{VLS_i\}$, and $\{P(i,j)\}$, during each iteration, we perform the following steps:

1. We first fix traffic rates $\{T(f,i)\}$ to their values $\{T^*(f,i)\}$ obtained in the previous iteration and estimate the loss rates $\{P^*(i,j)\}$ as described in Section 3.2.3.2.

2. We then numerically compute VLS_i^* and partial derivatives $\frac{\partial VLS_i^*}{\partial T_k^*}$ from $\{T_j^*\}$ according to Eq. (3.6). The key observation we leverage is that when $\{T_j\}$ are given, Eq. (3.6) only contains a single variable, *i.e.*, VLS_i. We present the details of this step later in Section 3.3.2.

3. We then approximate the non-linear interference constraints given in Eq. (3.5) and (3.6) using linear constraints. This can be achieved by computing the first-order approximation to the R.H.S. of (3.5) as a Taylor expansion at the current T_i^* and then replacing equality with "$\leq$". Specifically, we use the following linearized interference constraints:

$$\frac{T_i}{EP} \leq \frac{\tau_{\max}}{VLS_i^*} - \frac{\tau_{\max}}{(VLS_i^*)^2} \sum_k \frac{\partial VLS_i^*}{\partial T_k^*} \times (T_k - T_k^*), \qquad (3.10)$$

   where VLS_i^* and $\frac{\partial VLS_i^*}{\partial T_k^*}$ are computed in step 2.

4. We then treat loss rates $P^*(i,j)$ as constants in Figure 3.1. We also add the linearized interference constraints given in Eq. (3.10) to the LP formulation in Figure 3.1, yielding a linear program (LP_k) that can be solved efficiently by LP solvers like `cplex`.

5. Since the linearized interference constraints are only an approximation to the true interference constraints, the optimal solution to (LP$_k$) may be infeasible

under IEEE 802.11. We therefore perform a line search between the old solution and the optimal solution to (LP$_k$) to find a new set of traffic rates that are both feasible and improves the total throughput. During the line search, we need two capabilities: (i) to test whether a set of traffic rates are feasible under 802.11 (line 11 in Figure 3.2), and (ii) to find the maximum total throughput of opportunistic routing under such traffic rates. The former is performed as described in Section 3.3.2. The latter can be achieved by treating $T(f, i)$ as constants while solving the LP in Figure 3.1.

The iterative process continues until it reaches a solution that cannot be further improved upon after enough attempts. Since the total throughput will strictly increase over each iteration, the process is guaranteed to converge. In our experiments, we conservatively limit the maximum number of iterations to 30. Our experience suggests that typically the iteration stops much earlier.

3.3.2 Technical Details

Our model-driven optimization framework above makes use of the following three key capabilities: (i) estimating VLS_i from traffic rates $\{T_j\}$, (ii) testing the feasibility of given traffic rates $\{T_j\}$, and (iii) computing partial derivatives $\frac{\partial VLS_i}{\partial T_k}$. Below we present details on how to support these capabilities using our model.

Estimating VLS_i from traffic rates $\{T_j\}$: To numerically derive VLS_i from given traffic rates $\{T_j\}$, let $f_i(x) \triangleq x - T_{\text{slot}} - (T_{\text{xmit}} + T_{\text{DIFS}} - T_{\text{slot}}) * \Big[1 - \prod_j \Big(1 - \frac{D_{ij} * T_j * x}{EP}\Big)\Big]$. According to Eq. (3.6), $x = VLS_i$ is a root of $f_i(x)$. Moreover, we need $x \in \Big[0, \frac{EP}{\max_j(D_{ij} * T_j)}\Big]$ to ensure $1 - \frac{D_{ij} * T_j * x}{EP} \geq 0$ in Eq. (3.6). In Appendix 3.2.3.4, we prove that when $x \in \left[0, \frac{EP}{\max_j(D_{ij}*T_j)}\right]$, $f_i(x)$ is convex (see Theorem 1) and has at most one root (see Theorem 2). Therefore, we can apply any classic univariate root-finding algorithm (*e.g.*, Matlab's fzero function) to numerically compute the root of $f_i(x)$ over interval $x \in \left[0, \frac{EP}{\max_j(D_{ij}*T_j)}\right]$ and let the solution be VLS_i (if a root exists).

Testing the feasibility of traffic rates $\{T_j\}$: To test whether traffic rates $\{T_j\}$ are feasible, we first numerically compute VLS_i from Eq. (3.6) by finding a root of $f_i(x)$ over $x \in \left[0, \frac{EP}{\max_j(D_{ij}*T_j)}\right]$ as described above. If no solution is found, or if the solution VLS_i violates Eq. (3.5), then traffic rates $\{T_j\}$ are infeasible. Otherwise, $\{T_j\}$ are feasible.

Computing partial derivatives $\frac{\partial VLS_i}{\partial T_k}$: Eq. (3.6) also allows us to compute the partial derivatives $\frac{\partial VLS_i}{\partial T_k}$ for given traffic rates $\{T_j\}$, which allows us to linearize the non-linear interference constraints (see Section 3.3.1). Specifically, we have $\frac{\partial VLS_i}{\partial T_k} = \frac{N_{ik}}{1-M_i}$, where $M_i \triangleq (T_{\text{xmit}} + T_{\text{DIFS}} - T_{\text{slot}}) \times P_i^{\text{idle}*} \times \sum_j \frac{D_{ij}T_j}{EP-D_{ij}T_jVLS_i}$, $N_{ik} \triangleq (T_{\text{xmit}} + T_{\text{DIFS}} - T_{\text{slot}}) \times P_i^{\text{idle}*} \times \frac{D_{ik}VLS_i}{EP-D_{ik}T_kVLS_i}$, and $P_i^{\text{idle}*} \triangleq \prod_j(1 - \frac{D_{ij}T_jVLS_i}{EP})$.

3.4 Protocol Implementation

Overview: We develop a practical opportunistic routing protocol to install the opportunistic routes and rate limits computed by our optimization algorithm. It is built on top of MORE [24], which sits between the IP and 802.11 MAC layers. It differs from MORE in that it uses interference modeling and optimization to derive rate limits and opportunistic routes for a given performance objective. As in MORE, it leverages intra-flow network coding to carry out the derived routes (*i.e.*, an intermediate forwarder transmits random linear combinations of the packets it receives for a given flow at the rate derived from our optimization).

As most opportunistic routing protocols, we target medium to large transfers. A traffic source divides data packets into batches, and broadcasts a random linear combination of the original packets at the rate computed according to Figure 3.2. Upon receiving encoded packets, an intermediate node generates a random linear combination of all the innovative packets it has from the current batch. Each intermediate node uses the algorithm described in Figure 3.2 to determine how much traffic it should forward. After receiving enough innovative packets, the destination extracts the original data packets and sends an end-to-end ACK using MAC-layer unicast. When the source receives the ACK, it moves to the next batch. Below we describe several key steps in our protocol: (i) measuring inputs to seed our interference model, (ii) computing opportunistic routes and rate limits for each flow, (iii) routing traffic according to the derived sending rates and routes, (iv) supporting multicast, and (v) enhancing the reliability of end-to-end ACKs.

Measuring inputs: Our optimization procedure requires three inputs: traffic demands, interference measurements, and link loss rates. As reported in [40, 85], wireless traffic exhibits temporal stability and we can estimate current traffic demands based on previous demands. In our evaluation, we also test the sensitivity to the demand estimation error. To obtain interference measurements, we conduct pairwise broadcast measurements [6] and compute the carrier sense probability and conditional collision loss probabilities as in [85]. The measurement takes $O(n^2)$ time for an n-node network. In our 21-node testbed, each pair broadcasts for 30 seconds, and the entire measurement takes around 2 hours. To minimize measurement overhead, we conduct pairwise broadcast measurement infrequently, around once a week. Note that recent works have developed efficient online techniques to measure interference when a network is in use (*e.g.*, [9, 10]). These techniques can be incorporated into our implementation to further reduce measurement overhead. In addition, we compute the inherent wireless link loss rates using more frequent broadcast measurements conducted at the beginning of each experiment. The latter was based on more frequent measurement because it is more light-weight (only requiring $O(n)$ measurements) and existing routing protocols, such as [20, 24, 32], all use frequent loss measurements for route selection.

Deriving opportunistic routes and rate limits: The optimization procedure to compute opportunistic routes and rate limits can be done at a central location and then the optimized results can be distributed to all other nodes. We use this approach in our implementation. The amount of information to distribute is very small compared to data traffic: the optimization input is around 2 KB per node and the optimization output is within a 100 bytes per node. Alternatively, the computation can also be done in a fully distributed fashion, similar to link-state protocols like OSPF, where every node implements the same algorithm over the same data to arrive at the same results. Such computation happens once every several minutes. For instance, default SNMP polling intervals are typically 5 minutes, so the optimization can rerun when the traffic demands and network topology change. The optimization is fairly efficient (*e.g.*, it takes around 3 seconds to optimize routes and rate limits for 16 flows under the 5×5 grid topologies used in our simulation).

Enforcing derived routes and rate limits: As mentioned in Section 3.1, opportunistic routes are defined by $F(f, i, j)$. Here we compute $F(f, i, j)$ based on $T(f, i)$ and $Y(f, d, i, j)$ using a credit-based scheme. When node j receives a packet from node i, it increments its credit, which denotes how many packets j can transmit. If its credit is at least 1, j generates and transmits a random linear combination of the packets from the current batch buffered locally, and decrements the credit by 1. This process repeats until the credit goes below 1. The credit computation in our protocol differs from MORE [24] in two main aspects. First, our protocol computes credit to ensure the traffic and information sending rates conform to the derived T and Y. Second, unlike MORE, which treats all transmissions equally if coming from nodes with larger ETX to the destination, our protocol differentiates transmissions coming from different neighbors as follows. Upon receiving a packet from i, j increments its credit by $C \times R$, where C reflects the fraction of useful information contained in each packet received from i and R reflects the amount of redundancy j should include to compensate for loss to its neighbors. Specifically, we have $C = \frac{Y(f,d,i,j)}{T(f,i)(1-P(i,j))}$, and $R = \frac{T(f,j)}{\sum_k Y(f,d,j,k)}$. For example, when j receives a packet from a downstream node i, $C = 0$ to prevent j from sending non-innovative packets; when receiving a packet from an upstream node i, j updates its credit according to how much new information is involved in the packet and its loss rate to its forwarders.

Supporting multicast extension: Our previous description applies to the unicast case. A few modifications are required to support multicast. First, since a single packet carries a different amount of information for different destinations in the same multicast group, a node j increments its credit by $C \times R$, where $C = \frac{\max_d Y(f,d,i,j)}{T(f,i)(1-P(i,j))}$ and $R = \frac{T(f,j)}{\sum_k \max_d Y(f,d,j,k)}$. Second, when some destinations receive enough innovative packets, the encoded packets from the current batch should only be delivered to those who have not received all packets. To adapt to the changes in the set of destinations that need the packets, we dynamically re-adjust credit increment based on the remaining receivers who have not finished.

Enhancing ACK reliability: The destination sends an end-to-end ACK to the source upon receiving enough innovative packets for decoding so that the source can move on to the next batch. To ensure the reliability of ACKs, we keep retransmitting ACKs until they are received. To expedite ACK transmissions, ACKs do not perform binary backoff so that they have higher priority over retransmitted data. For fair comparison, we apply the same optimizations to MORE.

3.5 Evaluation Methodology

We evaluate our approach using extensive simulation and testbed experiments. Our evaluation consists of four parts. First, we compare the fidelity of the conflict-graph (CG) based model and our new model by quantifying their under-prediction and over-prediction errors. We use a conservative CG model, which considers two broadcast transmissions to interfere if any one of their receivers is interfered by the other transmission.

Second, we compare the performance of our opportunistic routing protocol using either the conflict-graph-based model or the new broadcast interference model, with the following existing routing protocols: (1) shortest-path routing using the ETX routing metric, which minimizes the total number of expected transmissions from a source to its destination [32], (2) shortest-path routing with rate limit optimization as developed in [85], and (3) MORE, a state-of-art opportunistic routing protocol.

We compare total network throughput under 1–16 simultaneous flows. We also compare in terms of the proportional fairness metric [74], which is defined as: $\sum_{f \in Flows} \log G(f)$, where $G(f)$ is flow f's throughput. This metric strikes a balance between increasing network throughput and maintaining fairness among the flows. Higher values are more desirable. Unless otherwise noted, each flow sends saturated CBR traffic.

Third, we evaluate the multicast performance of one multicast group with a varying group size, and measure the average throughput of the bottleneck receiver. As in [24], we extend the shortest path routing to support multicast by generating a multicast tree as a union of shortest paths towards all destinations and sending one copy of traffic along the links that are shared by multiple destinations. It saves the traffic on shared point-to-point links as in wire-line multicast routing but does not leverage the broadcast nature of wireless links (*e.g.*, a node still needs to send traffic separately to reach each of its next hops). Shortest path with rate limit [85] takes a routing matrix R as part of the input, where R_{id} is the fraction of flow d that traverses link i. To support multicast, we derive a multicast routing tree R, where

$R_{ig} = 1$ if link i appears in multicast group g's routing tree.

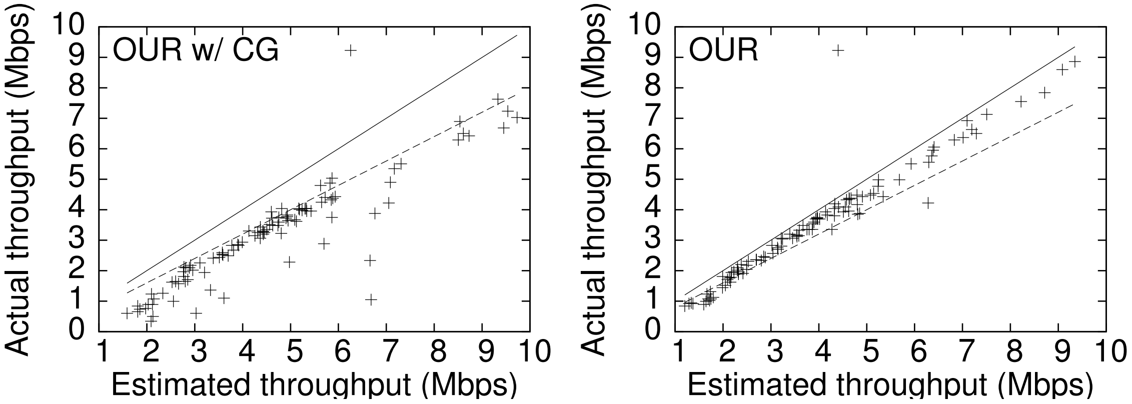
Fourth, we evaluate the sensitivity of our protocol against (i) errors in the input traffic demands, (ii) unknown external interference, and (iii) errors in link loss estimation.

For simulation, we implement all the protocols in Qualnet 3.9.5 [114]. For testbed experiments, we use the shortest path routing and MORE implementations publicly available [102]. In particular, the shortest path routing is the Click implementation released as part of MORE source code. We calculate ETX according to [32] and configure the link weight accordingly. The shortest path with rate limiting is based on the shortest path code but the rate limit of each flow is computed using the algorithm in [85]. We extend MORE to implement our protocol as described in Section 3.4. Both MORE and our protocol use 64 packets as the batch size for network coding. All these routing protocols are implemented using Click [30] and the MadWiFi driver [94] in the testbed.

Qualnet simulation: In simulation, we use 802.11a with a fixed MAC rate of 6 Mbps. The communication range is 230 meters, and interference range is 253 meters. These are the default values in Qualnet under transmission power of 10dB, and we use them in the conflict-graph model to determine if two nodes interfere. As in [85], we seed the new interference model by having two senders broadcast simultaneously and measuring the resulting sending rates and receiving rates. Unless noted otherwise, we use saturated UDP traffic with 1024-byte payloads.

For each scenario, we conduct 20 random trials. In each trial, flow sources and destinations are picked randomly and the simulation time is 20 seconds. We extend Qualnet to generate directional inherent packet losses, which are uniformly distributed between 0 and 90%. We consider two types of topologies: 5×5 grid and 25-node random topologies, each occupying a $750 \times 750\ m^2$ area.

Testbed experiments: Our testbed consists of 21 nodes located on two floors inside an office building. Each node runs Linux and is equipped with a NetGear WAG511 NIC. Unless otherwise specified, we use 802.11a to minimize interference with campus wireless LAN traffic, which uses 802.11g. This allows us to evaluate in a controlled environment. We use 20 mW transmission power and 6 Mbps transmission rate so that the network paths are up to 7 hops. Among the node pairs that have connectivity, 47.8% of them have links with loss $\leq 20\%$. All the routing protocols require estimation of link loss rates, which are measured by having one sender broadcast at a time and the other nodes measure the receiving rates. The loss measurements were collected before the experiments. In addition, our protocol and shortest path with rate limiting require interference measurement, which we collected once per week. As in simulation, we conduct 20 random trials for each scenario. Each trial lasts one minute. Other settings are consistent with the simulation. Finally, in Section 3.8, we further evaluate using 802.11b, which competes with campus WLAN traffic, in order to assess the sensitivity against unknown external interference.



(a) CG model

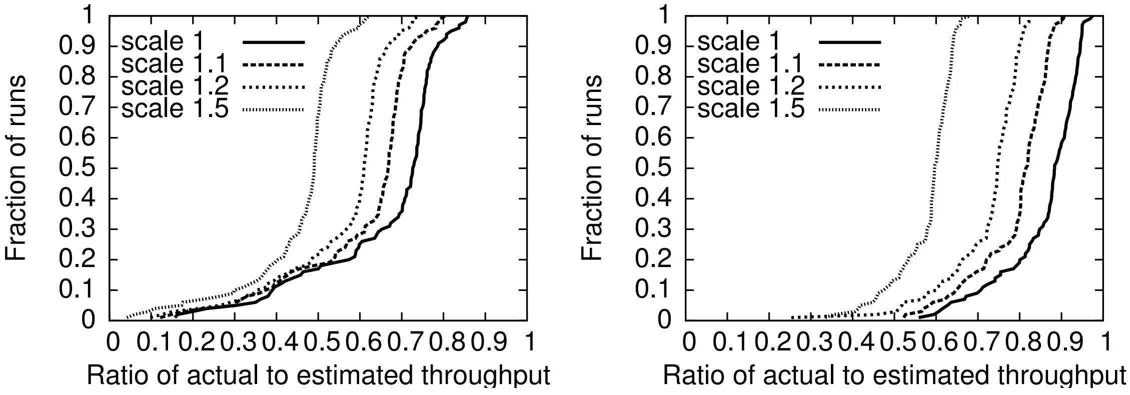
(b) Our model

Figure 3.3: Scatter plots of actual versus estimated throughput in simulation (25-node random topologies).

3.6 Model Validation

We adopt the evaluation methodology presented in [85] to quantify the accuracy of our model. In particular, to evaluate the over-prediction of our model, we install the estimated throughput to the network to see if it can be satisfied. To evaluate the under-prediction error, we uniformly scale each flow throughput by the same factor and check if the scaled demand is achievable. If the scaled demand is achieved in the network, it indicates that the under-prediction error is at least the scaling factor. We vary the scaling factor from 1.1, 1.2, 1.5, corresponding to a load increase of 10%, 20%, and 50%, and vary the number of flows from 1 to 16.

Simulation results: We first evaluate how often the models over-predict. In Figure 3.3, we plot the estimated throughput versus the actual throughput using the CG model and our model in 25-node random topologies. For reference, we plot lines $y = x$ and $y = 0.8x$. Here, the CG model significantly over-predicts the



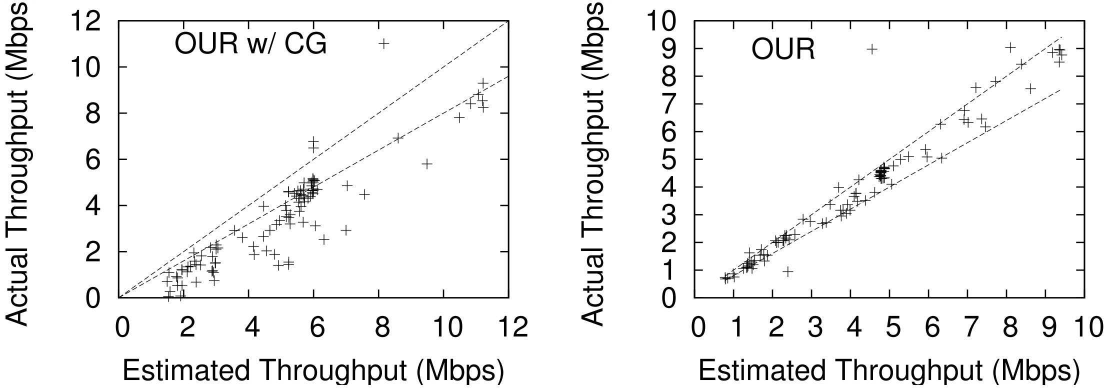
(a) CG model

(b) Our model

Figure 3.4: CDF of ratios between actual and estimated throughput in simulation (25-node random topologies).

actual throughput obtained, whereas the actual performance under our model is mostly within 80% of the estimated throughput. The CG model experiences significantly higher over-prediction errors since it assumes perfect scheduling, whereas our model explicitly models the interference between broadcast transmissions in IEEE 802.11, thereby achieving higher accuracy. Moreover, the amount of over-estimation by CG heavily depends on the network topology (*e.g.*, whether the network has hidden terminals) and simply scaling down the performance estimated by CG by a constant factor does not work. Both CG and our models have part of their over-prediction errors coming from the delay in end-to-end ACKs, during which time the source keeps retransmitting the current batch. This effect is not modeled. The use of a larger batch size can reduce the gap between the model estimation and actual performance at the cost of a larger header size and longer delay.

Next we quantify under-prediction errors. In Figures 3.4(a) and (b), we plot



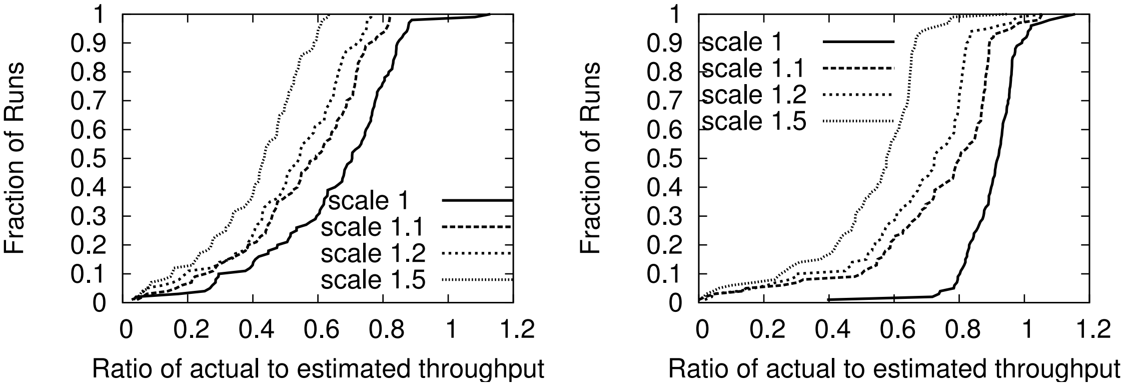
(a) CG model

(b) Our model

Figure 3.5: Scatter plots of actual versus estimated throughput in the testbed.

CDFs of the ratios between actual and estimated throughput in random topologies for the CG model and our model, respectively. Consistent with the scatter plots, the CG model mostly over-predicts, and virtually none of the scaled demands are satisfied. In comparison, using our model with a scale factor of 1, 80% of the runs have actual throughput within 80% accuracy of the estimation. Increasing the scale factor to 1.1 causes 65% of the actual throughput to be within 80% accuracy. After a further increase of the scale factor to 1.2, only 11% of actual throughput falls into 80% accuracy. This indicates that the demands scaled up by 20% can rarely be satisfied and shows our model has low under-prediction errors.

Testbed results: Next we validate our model and the CG model using testbed experiments. Figures 3.5(a) and (b) show the scatter plots of the CG model and our model, respectively. Figures 3.6(a) and (b) plot the CDFs of the ratios between actual and estimated throughput using different scale factors. As in simulation, the scatter plots from testbed experiments show a good match between actual and



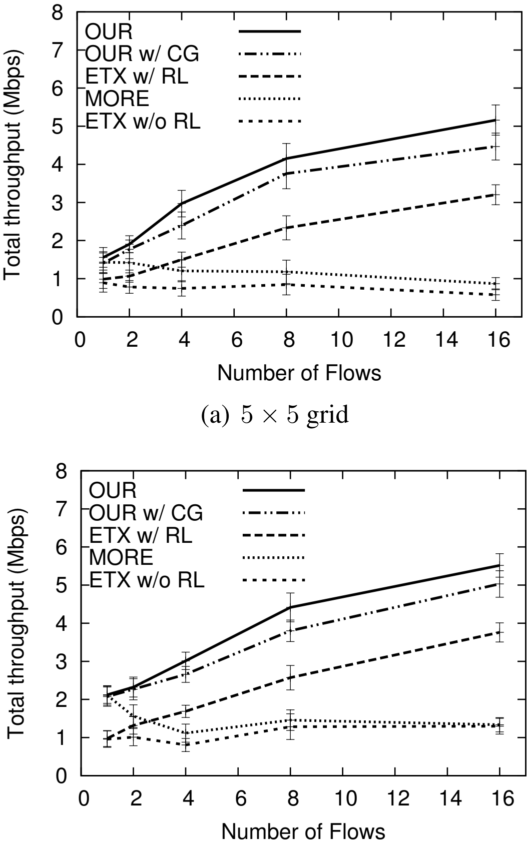
(a) CG model

(b) Our model

Figure 3.6: CDF of ratios between actual and estimated throughput in the testbed.

estimated throughput using our model and a significant over-estimation in the CG model. Scaling the demands by 1.1 leads to only 50% of the demands being satisfied and scaling the demands by 1.2 leads to only 29% of the demands being satisfied. These results indicate low over-prediction and under-prediction error. There are a few points in the testbed results where the actual throughput is higher than the estimated throughput. These cases arise from loss fluctuation: we use loss measurements to seed our model and derive opportunistic routes and rate limits, but the actual link loss rates in the experiment improve and support higher throughput.

Summary: The simulation and testbed results demonstrate that our model is accurate. It rarely over-estimates or under-estimates performance by more than 20%. In comparison, the CG model consistently over-predicts network throughput due to its assumption of perfect scheduling. These results highlight the importance of model fidelity on performance predictability.



(a) 5×5 grid

(b) 25-node random topology

Figure 3.7: Total unicast throughput in simulation (25-node random topologies).

3.7 Performance Comparison

In this section, we compare the performance of different routing protocols using simulation and testbed experiments.

3.7.1 Simulation Results

Total throughput of unicast flows: Figures 3.7(a) and (b) show the total throughput for 5×5 grid and 25-node random topologies, respectively. The error bars on the graph show the standard deviation of the sample mean.

We make several observations. First, in all cases our protocol using our model yields the best performance. It out-performs ETX by 76%-799% in the grid topology and by 117%-327% in the random topologies. Its gain over ETX with rate limiting ranges from 57%-99% in the grid topology and 46%-117% in the random topology. Its gain over MORE increases rapidly with the number of flows, ranging from 34% (2 flow) to 146% (4 flows) to 501% (16 flows) in the grid topology, and from 50% (2 flows) to 169% (4 flows) to 311% (16 flows) in random topologies. It out-performs the one with CG, the second best performing protocol by up to 24% in the grid topologies and 16% in the random topologies. Its performance benefit comes from three main factors: (i) taking advantage of opportunistic transmissions to cope with lossy wireless links, (ii) using interference-aware rate limiting to avoid network congestion, and (iii) using interference-aware opportunistic routing to maximize spatial reuse.

Second, comparing MORE against shortest path rate limiting, we observe that MORE out-performs the latter under 1 or 2 flows by leveraging opportunistic transmissions to recover losses. As the number of flows increases, the performance of MORE degrades and becomes significantly worse than shortest path with rate limiting due to lack of rate limiting. The impact of rate-limiting on opportunistic routing is even higher than shortest path routing because opportunistic routing uses

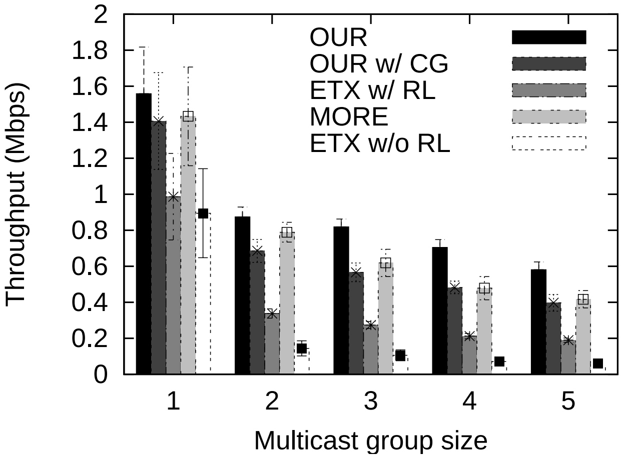
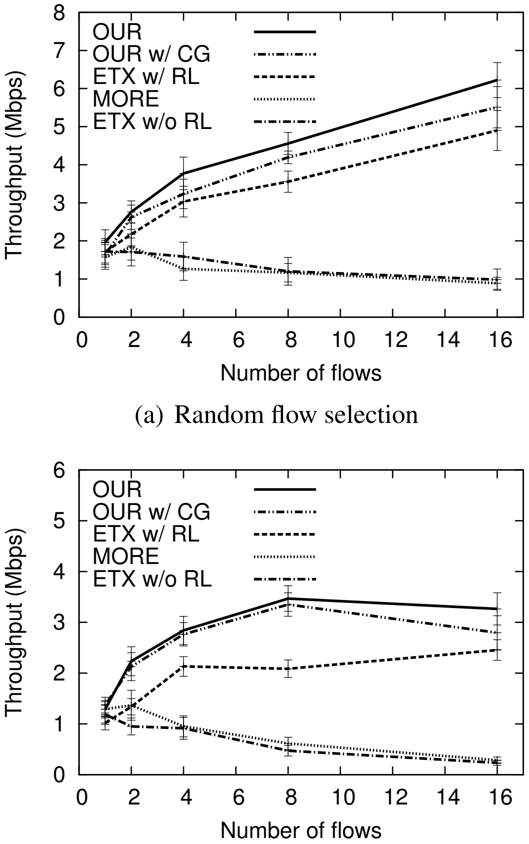


Figure 3.8: 802.11a multicast throughput in a 5×5 grid.

broadcast transmissions, which do not have exponential backoff and are more likely to cause network congestion. Further, congestion on the data path may corrupt end-to-end ACKs in opportunistic routing and lead to unnecessary retransmissions and throughput degradation. In contrast, shortest path routing uses unicast transmissions, whose MAC-layer ACKs are given higher priority and hence more reliable.

Multicast flows: We now evaluate the performance of multicast flows. Figure 3.8 shows the throughput of the bottleneck receiver in a multicast group as we vary the group size from 1 to 5. As in unicast flows, our protocol consistently out-performs the alternatives. It improves the one with CG by 10%-46%, MORE by 8%-47%, shortest path rate limiting by 58%-232%, and shortest path by 74%-894%. The larger performance gain over both versions of shortest path is because our protocol effectively exploits the broadcast nature of the wireless medium to reduce the number of transmissions. When sending to multiple neighbors, it can use one broadcast



(a) Random flow selection

(b) Flows with ETX > 1.25

Figure 3.9: Total unicast throughput in the testbed.

transmission to reach all the neighbors. In comparison, while shortest path routing uses a multicast tree to compress the traffic on a shared link, the links from one sender to different neighbors are considered different and multiple transmissions are required to reach them. For the same reason, MORE consistently out-performs both versions of shortest path routing. Our protocol still out-performs the one with CG and MORE by using a more accurate model to jointly optimize rate limit and opportunistic routes.

3.7.2 Testbed Results

Throughput of unicast flows: Figure 3.9(a) shows the total throughput of different protocols in the testbed, which has up to 7 hops. The relative rankings of the routing schemes are consistent with the simulation. As before, our protocol yields the best performance. the links in our testbed tend to be binary: either low loss or close to no connectivity. Among the node pairs that have network connectivity, 47.8% of them have loss rate within 20%. So the benefit of opportunistic routing is smaller in the testbed than in simulation. MORE performs close to shortest path routing, and significantly worse than shortest path with rate limiting; similarly, the gap between our protocol and shortest path routing also becomes smaller. These results confirm the intuition that opportunistic routing is most useful under lossy medium.

To understand how opportunistic routing performs under more lossy wireless medium, we conduct another set of experiments where we pick only flows whose ETX between source and destination is at least 1.25. Figure 3.9(b) summarizes the results. In this case, our protocol out-performs shortest path with rate limiting by 26%-67%, and out-performs shortest path by 9%-1303%. It provides similar performance to MORE with 1 flow and up to 1047% improvement over MORE with 16 flows. Furthermore, MORE also out-performs shortest path routing by up to 43%. However, its performance is still worse than shortest path with rate limiting as the number of flows reaches 4 or higher. These results are consistent with the simulation, and highlight the importance of optimizing rate limiting and opportunistic routing.

Proportional fairness of unicast flows: Next we consider maximizing propor-

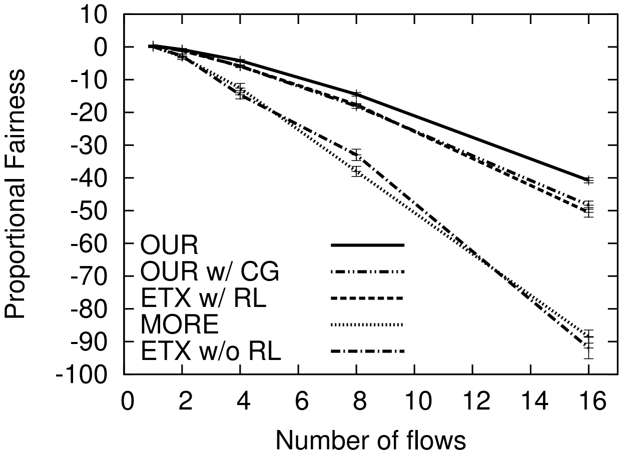


Figure 3.10: Unicast proportional fairness in the testbed.

tional fairness. Since this objective is non-linear, in order to optimize it, we first approximate it using a piecewise linear, increasing, convex function as follows. We select s points on $\log(x)$, and approximate $\log(x)$ using s line segments, each connecting two adjacent points. We perform two different point selections and observe similar performance. In the interest of space, below we present results from only one selection: $x = 0.001$, 0.01, 0.1, $\sqrt{0.1}$, 1, $\sqrt{10}$, 10. When a flow's throughput is 0, its log value is undefined, so we set its throughput to 1 Kbps. Figure 3.10 shows the proportional fairness as we vary the number of unicast flows in the testbed. As in simulation, the three routing schemes that support rate limiting significantly out-perform MORE and shortest path without rate limiting since the latter two can easily cause starvation. Among those that support rate limit, our protocol performs the best due to its opportunistic routing and high-fidelity model.

Multicast flows: Finally, we evaluate the performance of multicast in our testbed.

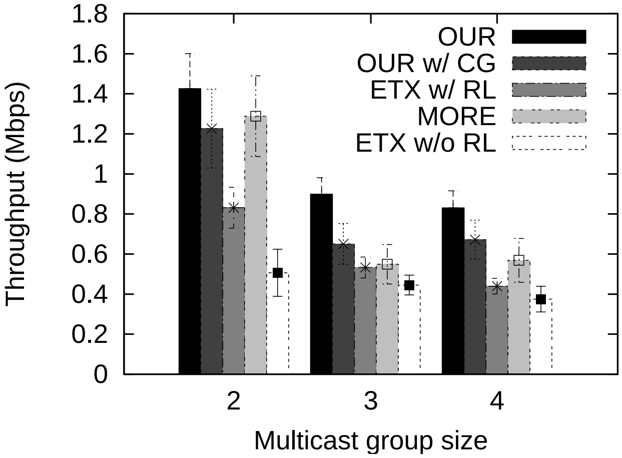


Figure 3.11: 802.11a multicast throughput in the testbed.

Figure 3.11 shows the throughput of the bottleneck multicast receiver in one multicast group, where the multicast group size is varied from 2 to 4. Our protocol performs the best. It out-performs the one with CG by 16%-38%, MORE by 10%-63%, shortest path with rate limiting by 68%-89%, and shortest path routing by 101%-181%. In addition, by leveraging the broadcast wireless medium, all types of opportunistic routing, including MORE, out-perform both versions of shortest path routing. These results suggest opportunistic routing is even more useful to multicast, and the effective optimization of multicast routes and rate limiting continues to be important.

3.7.3 Summary of Performance

The simulation and testbed results show that our protocol consistently out-performs the alternatives. By leveraging opportunistic transmissions and effective route optimization, it significantly out-performs state-of-the-art shortest path routing protocols. By using a high fidelity network model to jointly optimize rate limits and opportunistic routes, it significantly out-performs state-of-the-art opportunistic routing protocols. These benefits suggest that all the design components in our protocol, including opportunistic routing, network model, and joint rate limit and route optimization, are essential and help improve the performance.

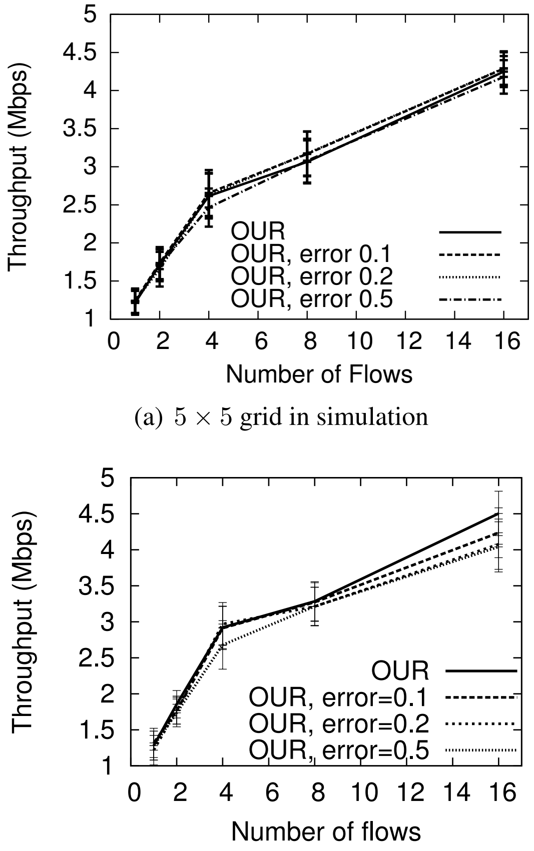
3.8 Evaluation of Sensitivity

In this section, we evaluate the sensitivity of various protocols under inaccurate traffic demands, loss fluctuation and unknown external interference.

3.8.1 Impact of Inaccurate Traffic Demand

Methodology: We first evaluate the performance under inaccurate traffic demand estimation, since in practice traffic demands fluctuate and may not be known exactly. The actual traffic demands are uniformly distributed between 0 and the maximum link throughput. To simulate demand estimation error, we inject errors into the actual demands and feed the salted demands to our optimization framework while imposing the actual demands to the network for evaluation. The error injected is uniformly distributed between 0-10%, 0-20%, and 0-50%. To protect against estimation error, our protocol slightly over-provisions by scaling the derived sending rates from the optimization output by a factor of 1.1.

Simulation: Figure 3.12(a) shows the total throughput versus the number of flows. We see similar performance across different error ranges. This indicates that our protocol is fairly robust against demand estimation errors, because for the purpose



(a) 5×5 grid in simulation

(b) 802.11a Testbed

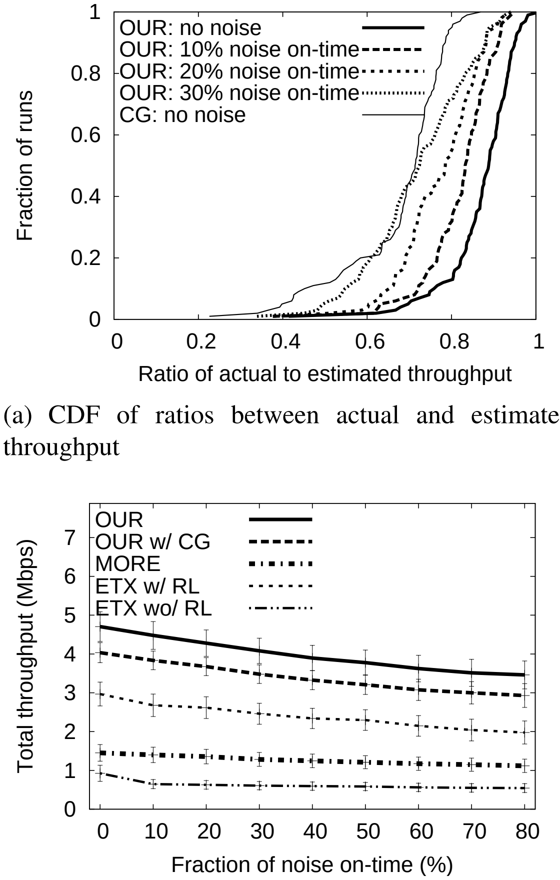
Figure 3.12: Total throughput under inaccurate traffic demand estimates.

of performance optimization, the spatial traffic demand distribution is more important than the exact demand values.

Testbed: Figure 3.12(b) shows the performance of our protocol when we feed inaccurate traffic demands as input to our optimization. As in simulation, it is robust to the inaccuracy in traffic demand estimation in testbed. Its performance under no error is close to that under the relative error of 0.5.

3.8.2 Impact of Unknown External Interference and Loss Fluctuation

3.8.3 Simulation



(a) CDF of ratios between actual and estimated throughput

(b) Total throughput of 8 flows

Figure 3.13: Simulation results under 2 noise sources with varying on-time in 25-node 802.11a random topologies.

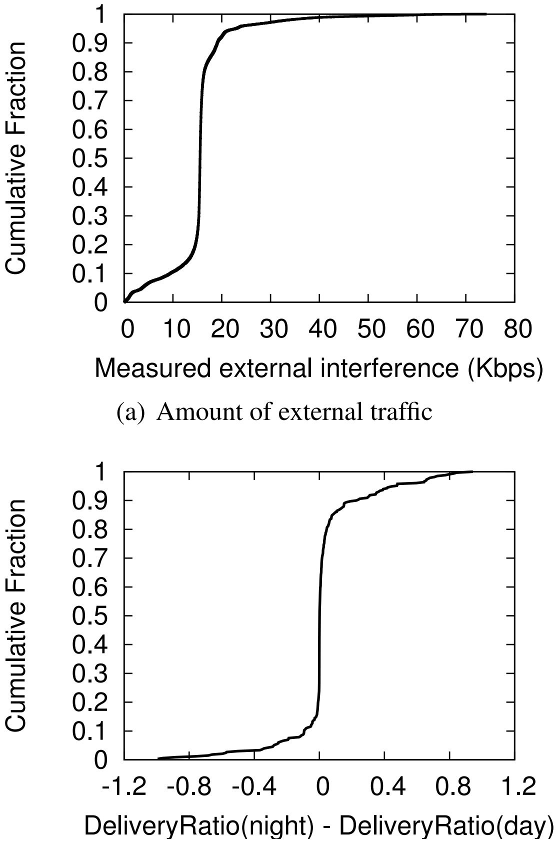
Methodology: We create external interference by randomly placing two external noise sources in 25-node random topologies. All protocols compute routes and rate limits without considering the external noise, and we measure the throughput of using the derived routes and rate limits under external noise. The noise sources have uniformly distributed on and off time, where the average on-time is 0.25 second and the total simulation time is 20 seconds. We vary the average off-time so that every noise source is on 10% to 80% of time. During on-time, each noise source broadcasts 802.11 packets (with 1024-byte payload) as fast as possible.

Model validation: First, we compare actual throughput under external noise versus estimated throughput derived without considering the noise sources. As shown in Figure 3.13(a), the accuracy of our protocol degrades gracefully as we increase the on-time of each noise source. The fractions of runs that achieve within 30% error are 99% under 10% noise on-time, 76% under 20% noise on-time, and 56% under 30% noise on-time. Moreover, even with 30% noise on-time, it achieves much higher predictability than the one with CG model under no external noise.

Performance comparison: As shown in Figure 3.13(b), the ranking of different protocols remains the same across all noise levels. Our protocol consistently out-performs all other protocols. Even when every noise source is active 80% of time, it out-performs the one with CG by 18%, shortest path with rate limiting by 75%, MORE by 209%, and shortest path without rate limiting by 535%. Moreover, the performance of different protocols degrades smoothly as the on-time of each noise source increases.

3.8.4 Testbed

Methodology: We also evaluate the sensitivity in an 802.11b testbed consisting of 22 nodes. As before, we randomly select flows in our network. As common practice, we run the link loss measurements at night, which has low network activity.

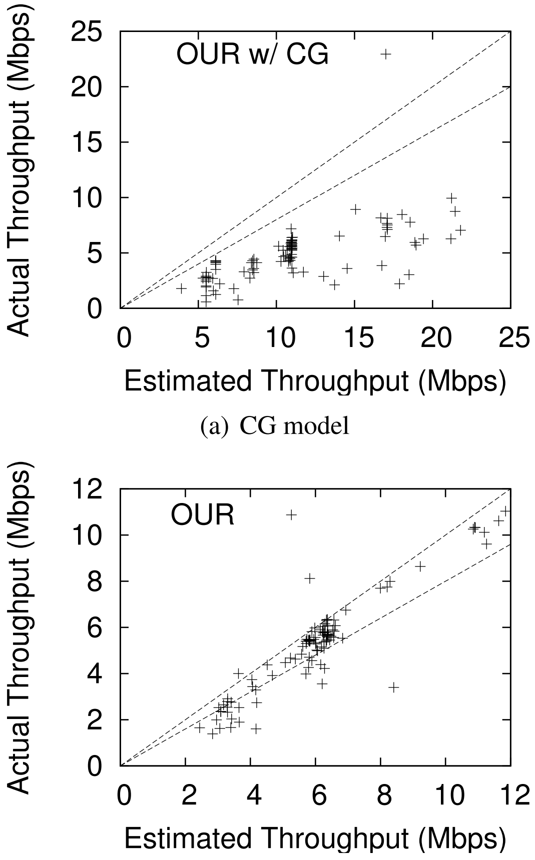


(a) Amount of external traffic

(b) CDF of loss fluctuation

Figure 3.14: Amount of external traffic from the campus network and loss fluctuation in our 802.11b testbed.

Then we run all evaluation during the day. This allows us to evaluate the sensitivity against unknown external interference and loss fluctuation. In particular, our building has an active 802.11g campus network, whose traffic directly interferes with our wireless mesh traffic. We treat traffic from the campus network as unknown external interference. Figure 3.14(a) plots the CDF of the average campus network traffic measured by all mesh nodes in promiscuous mode every 30 seconds. The



(a) CG model

(b) Our model

Figure 3.15: Scatter plots of actual versus estimated throughput in our 802.11b testbed under unknown external interference and loss fluctuation.

median and mean are both 15.5 Kbps. Moreover, loss fluctuates from nights to daytime. Figure 3.14(b) plots a CDF of $DeliveryRatio(night) - DeliveryRatio(day)$ over all links that have $\geq 5\%$ delivery rates. We observe loss fluctuation, because during the day time (i) more people sit near mesh nodes and cause more attenuation, and (ii) more people move around and close/open doors and cause frequent changes to the RF environment.

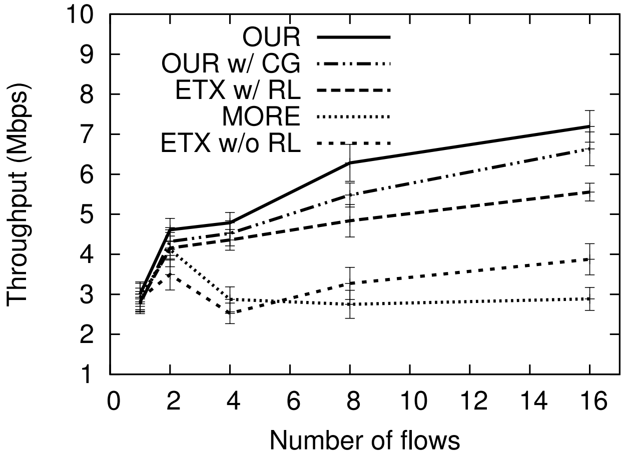


Figure 3.16: Unicast throughput in our 802.11b testbed under unknown external interference and loss fluctuation.

Model validation: Figure 3.15 shows the scatter plot of actual versus estimated throughput from the 802.11b testbed. We also plot $y = x$ and $y = 0.8x$ for reference. Our protocol continues to exhibit high predictability: 78% of runs have within 20% error.

Performance comparison: As shown in Figure 3.16, our protocol continues to perform the best. Different from simulation, MORE sometimes performs worse than shortest path without rate limiting because the network congestion in MORE is more severe in a dense network like our 802.11b testbed.

3.9 Summary

In this chapter, we present the first protocol that can accurately optimize the performance of opportunistic routing in IEEE 802.11 networks. Our framework consists of three key components: (i) a simple yet accurate wireless network model, (ii) a novel algorithm for optimizing different performance objectives, and (iii) an opportunistic routing protocol that effectively maps solutions resulted from our optimization into practical routing configurations. Through testbed implementation and simulation, we show that the performance of our protocol is close to our estimation, and is much better than state-of-the-art shortest path routing and opportunistic routing protocols. Moreover, it is robust against inaccuracy introduced by a dynamic network and consistently out-performs the existing schemes. To further enhance the robustness against traffic and topology variations, in the future we plan to extend the robust traffic engineering techniques developed in the Internet to optimize wireless networks. In particular, a traffic engineering system usually collects a set of traffic matrices and uses their convex combination to cover the space of common traffic patterns for optimization. These new demand constraints are compact and can be easily incorporated into our framework. We plan to extend this technique to cope with both traffic and topology variations in wireless networks.

Chapter 4

Link Layer

After detailing an optimized opportunistic routing protocol to combat losses in the Network Layer, we now present a solution to mitigate loss in the Link Layer. The contents of this chapter are based on work published in [128].

4.1 Introduction

This chapter presents *ER*, an efficient retransmission mechanism to support reliable unicast, broadcast, and multicast in WLANs. The design of ER can also easily be extended to multihop wireless networks. We illustrate the idea of ER using the following two simple examples.

Consider two clients $C1$ and $C2$ associated with an access point (AP). The AP has two packets to send: $p1$ destined to $C1$ and $p2$ destined to $C2$. The links $AP - C1$ and $AP - C2$ both have 50% loss rates. Using the traditional unicast in IEEE 802.11 [1], on average 4 transmissions are required to successfully send packets to both clients. Due to the broadcast nature of wireless medium, $C1$ may lose $p1$ but receive $p2$; similarly, $C2$ may lose $p2$ but receive $p1$. Whenever this case occurs, ER reduces the number of transmissions by letting AP retransmit $p1 + p2$, which is $p1$ xor-ed with $p2$, instead of sending $p1$ and $p2$ separately. Then $C1$ can extract $p1$ by xoring $p2$ with $p1 + p2$, and similarly $C2$ can extract $p2$ by xoring $p1$ with $p1 + p2$. In this way, the AP reduces the number of transmissions (including the original transmissions) from 4 to 3 to successfully deliver both packets.

Now consider a multicast example. Since broadcast is a special case of multicast, in this chapter we consider multicast without loss of generality. Suppose the AP wants to send both packets $p1$ and $p2$ to the clients $C1$ and $C2$. If both clients only receive one packet and the packets they receive are different, then AP can retransmit $p1 + p2$ instead of retransmitting them separately, thereby using one transmission to recover two packet losses.

In both unicast and multicast examples, ER takes advantage of wireless broadcast medium and minimizes the number of transmissions by effectively combining packets. As we will show later, the coding benefit further increases with the number of clients and/or the number of packets.

The design of ER is inspired by several recent works on network coding, in particular, COPE [73]. ER complements the previous work in several important ways. First, the existing network coding approaches target multihop wireless networks and there are no coding opportunities for single hop paths. Instead we show that the coding benefit also exists in widely-used single-hop WLANs. Second, the existing coding schemes, such as COPE, maximize efficiency by coding the original transmissions destined to different receivers, but relies on MAC-layer retransmissions to recover lost packets. In comparison, ER improves the efficiency of MAC-layer retransmissions by reducing the number of retransmissions required to recover the losses. Therefore ER can be applied to wireless LANs and multihop wireless networks to achieve efficient retransmission. When combined with COPE, it helps achieve high efficiency in both original transmissions and retransmissions of lost packets. Third, the coding opportunities in the previous work are determined by traffic demands. For example, coding opportunities arise in COPE when traffic heading towards different directions meet at the same intermediate router. Instead the coding opportunities in ER are determined by the loss patterns – more coding opportunities arise when different receivers lose different sets of packets. So understanding the coding benefits under realistic packet loss characteristics is an interesting and open question.

In this chapter, we develop and implement ER to provide reliable unicast, broadcast, and multicast in WLANs. An important component in the design of ER, as well as other network coding schemes, such as COPE [73] and broadcast coding [84], is which set of packets should be coded together to minimize the number of required transmissions. We formally study the problem, and show it is NP-hard. We describe several practical heuristics and use empirical evaluation to study their effectiveness. Our extensive simulation and testbed experiments show that ER significantly reduces the number of retransmissions compared to the existing retransmission scheme, which retransmits the lost packets by themselves.

The rest of the chapter is organized as follows. In Section 2.2, we review the existing work on providing reliable communication in WLANs. In Section 4.2, we present our approach. We describe our simulation methodology and results in Section 4.3, and present the implementation and experimental results in Section 4.4. We conclude in Section 3.9.

4.2 Our Approach

ER can be applied to single-hop WLANs and multihop wireless networks in the same way. In the following description, a sender refers to an AP in a wireless LAN, or refers to a traffic source or an intermediate router in a multihop wireless network.

4.2.1 Overview

First, we consider unicast transmissions. In ER, a sender maintains two packet queues: one for new packets and the other for retransmission packets. In the new packet transmission mode, the sender sends packets from its new packet queue, following 802.11's contention mechanism. Since ER is a replacement of the MAC-layer retransmission in 802.11, the sender disables the default MAC-layer retransmission by setting the MAC retry count (*i.e.*, the maximum number of retransmissions at MAC-layer) to 0. ER retransmits the packet above the MAC-layer until its receiver acknowledges the packet or the retry count in ER is reached. To provide the same level of reliability, the retry count in ER is set to the original MAC retry count.

The receivers periodically send feedback of which packets are received successfully. Based on the feedback, the sender puts the packets that require retransmissions into the retransmission queue. In the retransmission mode, the sender examines all the packets in its retransmission queue to determine which sets of packets to code together in order to minimize the number of retransmissions. The sender uses MAC-layer unicast to send both new and retransmitted packets, while all the other nodes use promiscuous mode monitoring so that they can receive packets destined to other nodes, which is necessary to create coding opportunities. Unicast is used in this case because its binary exponential backoff can help reduce collision losses under high load and it also allows the use of RTS/CTS to avoid hidden terminals (if needed).

ER can be applied to multicast traffic in a similar manner. As in unicast, the sender also maintains two queues and switches between them for sending new and retransmitted packets. The receivers report to the sender the set of packets that they receive, and a packet is retransmitted until all its receivers acknowledge the packet or the retry count in ER is reached. Multicast packets can be sent either using MAC-layer multicast or MAC-layer unicast with promiscuous monitoring. Our implementation uses the latter approach: packets are unicast to one of the receivers in the multicast group, and the other receivers in the group use promiscuous mode monitoring to extract their data. We choose this implementation because it unifies the unicast and multicast implementation and also allows potential use of exponential backoff and RTS/CTS. However this choice is not fundamental, and ER can also be built on top of MAC-layer multicast.

Note that ER can be applied to both encrypted and unencrypted data packets. When encryption (*e.g.*, WPA) is used, the sender xors encrypted packets and adds ER's header in plain text to specify which packets are combined, and the receiver uses the ER's header information to extract the new packet and then decrypt its content. Therefore the benefit of ER extends to corporate wireless networks using WPA.

Several important design issues should be addressed in order to realize ER.

- First, how should the receivers give timely feedback to the sender without incurring much overhead?
- Second, when to retransmit data? This question involves two parts: (i) how should the sender determine that a packet requires a retransmission? (ii) when the medium is available for the sender to transmit, which packet to send – a new packet or a lost packet? The answers to these questions affect the retransmission delay, the number of unnecessary retransmissions, and the potential coding benefit.
- Third, which set of packets should be coded together to minimize the number of retransmissions?

To address the above issues, ER consists of the following three components: (i) a light-weight receiver feedback scheme, (ii) a scheduling algorithm to determine which packets need retransmissions and when to transmit a new or lost packet, and (iii) a coding algorithm to optimize which set of packets to be coded together.

4.2.2 Receiver Feedback

Our receiver feedback scheme is built on COPE [73], where a node sends reception reports to inform which set of packets it has recently received. As in COPE, we use selective/cumulative ACKs to minimize the impact of ACK losses. Specifically, the report contains two fields: (i) the starting sequence number of the out of order ACKs ($start$), and (ii) a bit-map of out of order ACKs. All the packets up to $start$ are assumed to be received, and i-th position in the bitmap is 1 if and only if the $start + i$-th packet is received. Our implementation differs from COPE in the following ways. First, to increase the reliability of feedback, we send feedback using MAC-layer unicast, which will automatically retransmit lost feedback. Second, the length of bitmap increases from 1 byte in COPE to 8 bytes in ER so that it is more resilient to high ACK losses at a cost of a small increase in ACK overhead. We find this small cost increase is worthwhile since its benefits under ACK losses is significant. Third, COPE has separate ACKs and reception reports, where the former acknowledge the receipt of packets destined to itself and the latter acknowledge the receipt of packets destined to other nodes; furthermore these two types of reports are sent in different time scales. For the purpose of ER, the difference between ACK and reception reports is no longer necessary because in order to determine which packets to retransmit and how to code them, the sender needs both ACKs and reception reports. Therefore, our implementation unifies ACK and reception reports. Finally, when retransmitting a multicast packet, the sender specifies the nodes to which multicast packets are destined; and only the nodes that are specified as destinations will send feedback. In this way, we can reduce the receiver feedback especially when the multicast group is large but only a small number of nodes need the packet.

4.2.3 Scheduling Algorithm

Next we need to decide (i) when a packet needs a retransmission and (ii) when the medium is available for the sender to transmit, which packet should the sender transmit – a new packet or a lost packet?

```
if ($T$ is the first RTT measurement)
  SRTT = T;
  RTTVAR = T/2;
  RTO = SRTT + K*RTTVAR;
else
  $RTTVAR = (1 - \beta) \times RTTVAR + \beta \times |SRTT - T|$;
  $SRTT = (1 - \alpha) \times SRTT + \alpha \times T$;
  $RTO = SRTT + K * RTTVAR$;
end
```

Figure 4.1: Estimation of RTO.

To address the first question, we use a standard approach to estimate retransmission timeout (RTO), similar to TCP. Specifically, for every packet that has not been retransmitted, a node measures the time difference between when the packet is transmitted and when the corresponding ACK in ER is received. Let T denote the measured round-trip time of the current packet. Then the node updates its RTO based on smoothed RTT and RTT variance as shown in Figure 4.1. RTO is initialized based on the MAC data rate. Our evaluation uses $K = 4$, $\alpha = 1/8$, and $\beta = 1/4$ as in TCP [110].

To answer the second question, we make the following observation. If the sender retransmits a packet whenever the retransmission queue is non-empty, it achieves lowest retransmission delay. On the other hand, such aggressive retransmission would reduce or even eliminate coding opportunities. In the extreme, there is only one packet in the retransmission queue, and the packet has to be sent by itself and results in 0 coding gain. To strike a good balance between low delay and high coding gain, we use the following heuristic: retransmit the packet when the retransmission queue reaches a certain threshold or the packets in the retransmission queue timeout. The first condition increases the coding gain, and the second condition bounds retransmission delay. Our evaluation uses 25 as the threshold for the retransmission queue, and uses 250 msec as the timeout.

4.2.4 Coding Problem and Algorithms

Another important design issue is how to code packets together to minimize the number of transmissions. In this section, we first formally study the coding problem and show that it is NP-hard to solve. Then we describe several practical coding algorithms.

4.2.5 Problem Specification

First, we introduce some notation. Let $N(i)$ denote the set of nodes that need packet i, and $H(i)$ denote the set of nodes that have packet i. A sender only codes packets together if the coded packet can be decoded right after its reception. This condition is commonly used in existing coding algorithms to simplify decoding algorithms [73, 84]. Under the above condition, two packets i and j can be coded if and only if $N(i) \subseteq H(j)$ and $N(j) \subseteq H(i)$, which we call *coding condition*. Essentially it means that i and j can be coded if and only if any nodes that need j have i, and any nodes that need i have j. To show the forward direction holds, i and j can be coded means that any node in $N(i)$ can decode the packet $P_i + P_j$

immediately after its reception; since nodes in $N(i)$ do not have P_i, the only way for decoding to succeed is that $N(i)$ have P_j so that they can xor $P_i + P_j$ with P_j. Similarly for N(j). To show the reverse direction holds, since $N(i) \subseteq H(j)$, every node in $N(i)$ has P_j. Then after receiving the coded packet $P_i + P_j$, it can extract P_i by xoring $P_i + P_j$ with P_j. Similarly for $N(j)$.

Based on the coding condition, we construct the following coding graph. Each packet is denoted by a vertex in the coding graph. For any two packets that can be coded together, we draw an edge between their corresponding vertices. It is not difficult to see that a transmission can be decoded if and only if the transmission only involves packets corresponding to a clique in the coding graph, where a clique is a set of vertices such that there is an edge between every pair of the vertices. This is a simple generalization of the coding condition from 2 packets to N packets. Therefore the coding problem, *i.e.*, transmitting a given set of packets using a minimum number of transmissions, is essentially finding a minimum clique partition [69], which is stated as follows. Given a graph $G = (V, E)$, where V are vertices and E are edges in G, partition V into a minimum number of disjoint subsets V_1, V_2, ..., V_k such that the subgraph induced by V_i is a complete graph. Next we show the coding problem is NP-hard. We prove this by reducing the minimum clique partition problem, which is known to be NP-hard, to the coding problem.

Given a graph $G = (V, E)$ for a minimum clique partition problem, we construct the coding problem that consists of three types of input: (i) a set of packets, (ii) for each packet i which clients need it – $N(i)$, and (iii) for each packet i which clients have it – $H(i)$. For each vertex i in G of the minimum partition problem, we create a corresponding packet P_i in the coding problem. Each packet P_i is needed by a distinct receiver R_i, *i.e.*, $N(i) = \{R_i\}$. Based on the coding condition, we assign $H(i)$ as follows:

$$H(i) = \{R_j | \forall j \ s.t. (i, j) \in E\}$$

To show the above assignment of $H(i)$ satisfies the coding condition, we need to show that (i) any two adjacent nodes i and j satisfy $N(i) \subseteq H(j)$ and $N(j) \subseteq H(i)$, and (ii) any nodes that satisfy $N(i) \subseteq H(j)$ and $N(j) \subseteq H(i)$ are adjacent. The former holds because for $\forall \ (i, j) \in E$, $N(i) = \{R_i\} \subseteq \{R_k | \forall k \ s.t. \ (k, j) \in E\} = H(j)$, similarly for $N(j) \subseteq H(i)$. The latter holds because for $\forall N(i) = \{R_i\} \subseteq H(j) = \{R_k | \forall k \ s.t. \ (k, j) \in E\}$, we have $(i, j) \in E$. Therefore with the above construction, finding a minimum clique partition is essentially finding the optimal solution to the coding problem. Hence the coding problem is NP-hard.

4.2.6 Coding Algorithms

We describe three practical heuristics to solve the coding problem. Given the NP-hard nature of the problem, these heuristics are not guaranteed to yield optimal results. However as we will show in Section 4.3 and Section 4.4, they work well in practice. In addition, we also present an exhaustive search algorithm. While the algorithm is guaranteed to give an optimal solution, it is computational very expensive and can only run on small-sized problems. So it just serves as an interesting baseline comparison.

Sort by time: The heuristic described in COPE [73] can be directly applied here. This heuristic is greedy in nature. Packets are sorted according to their arrival time with the first packet being the one that arrives the earliest. Every time the sender starts with the first packet in the queue, and iteratively combines with subsequent packets in the queue as long as the combined packet can be decoded (*i.e.*, all the receivers of the combined packet already have all but one packets in the combined packet).

Sort by utility: We find the order in which packets are examined for potential coding is important. The previous heuristic codes the packet in the order of their arrival time. In the sort-by-utility heuristic, each packet is assigned a utility, defined as the number of receivers that need the packet. Intuitively, the packet that is required by more receivers is more important, and should be transmitted earlier. Therefore we examine the packet in the non-increasing order of utility and using arrival time for tie-break. Specifically, the sender starts with the packet having the highest utility, and iteratively codes subsequent packets as long as the combined packet can be decoded. Note that this algorithm is useful for broadcast and multicast. In unicast, each packet is needed by one client, and has the same utility of 1. So it is equivalent to the sort-by-time heuristic under unicast.

Maximum clique: As shown in Section 4.2.5, the coding problem can be cast as finding a minimum clique partition in a coding graph. Therefore another approach is to employ heuristics for minimum clique partition. One of the commonly used heuristics to minimum clique partition is to first find a maximum clique in the graph; then remove the clique from the graph and find another maximum clique, and iterate. Note that the maximum clique problem itself is NP-hard, but has a simple heuristic, which starts with the vertex of highest degree and iteratively adds additional vertices to the clique as long as they maintain the clique property – there is an edge between every pair of vertices in a clique.

Exhaustive search: We develop an exhaustive search algorithm to minimize the number of retransmissions. This algorithm is computationally very expensive and is not for practical use. Instead it serves as an interesting baseline comparison to quantify the effectiveness of the other coding algorithms.

First, we introduce a few notations. Let M denote the number of packets required for retransmissions. Let S denote a state, indicating for each packet i which nodes need it and which nodes have it, namely $(N(i), H(i))$. The exhaustive search algorithm first generates all possible packet combinations. There are 2^M packet combinations, since each packet either belongs to a packet combination or not. The goal is to find a smallest number of packet combinations that converts the current state to the state where every node gets the packets it needs (*i.e.*, $N(i) = \{\}$ for every i). To identify a minimum set of packet combinations, we build the following coding tree. The root of the tree is the current state. Starting from the root, we try every packet combination. A packet combination is considered useful if it allows at least one receiver to get a packet it needs if there is no loss. For each useful packet combination, we add a child node to the root; we also label the edge of to the child with the packet combination and label the node with the state after all nodes receive the packet combination. Packets combinations that are not useful are simply ignored. After going through all the packet combinations, we then repeat the process – for each of the child nodes we identify the useful packet combinations and add them to the next level of the tree. The process continues until we reach a state where every node gets the packet that it needs. The depth of the tree at that node is the minimum number of transmissions required (assuming the depth of a root is 0). Moreover, the packet combinations marked along the path from the root to that node are the set of packets to transmit that minimizes the number of transmissions.

4.3 Simulation Methodology and Results

In this section, we first describe our simulation methodology and then present performance results.

4.3.1 Simulation Methodology

To evaluate the performance of various retransmission schemes presented in Section 4.2.6, we simulate the behavior of the algorithms under both unicast and multicast using a variety of network topologies. In our simulation, we generate network topologies consisting of a sender and a varying number of receivers with varying loss rates.

We consider both homogeneous and heterogeneous loss rate assignments between a sender and each of its receivers. In homogeneous cases, the loss rates between the sender and all its receivers are the same, and are varied from 10% to 90%. In heterogeneous loss cases, we assign the loss rates between the sender and its receivers randomly chosen between 0 and an upperbound, where the upperbound is varied from 10% to 90%. Therefore some receivers may see loss rate as low as 0, while other receivers may see loss rates close to the upperbound. For both homogeneous and heterogeneous cases, we generate losses using Bernoulli and Gilbert models. In the Bernoulli model, each packet is dropped with a fixed probability determined by the loss rate of the link. In the Gilbert model, the link moves between a good state and a bad state, where no packets are dropped at the good state and all packets are dropped at the bad state. Following [104, 108], we use 35% as the probability of remaining in the bad state. The other state-transition probabilities are determined to match the average loss rate with the loss rate assigned to the link.

The high-level simulation evaluates a simplified scheduling algorithm, where a sender sends a constant-sized batch of packets at a time before starting retransmissions. Unless otherwise specified, the batch size is 20. In addition, we also evaluate the impact of varying batch sizes. Note that the batch-based scheduling tries to approximate the effect of the scheduling algorithm presented in Section 4.2.3. The simulation does not directly evaluate the latter scheduling, because it requires modeling timing dynamics, which the high-level simulation does not model. The testbed evaluation will directly evaluate the scheduling algorithm in Section 4.2.3.

We use *retransmission ratio* to quantify the performance of different retransmission schemes. The retransmission ratio is defined as the total number of retransmissions using the current scheme divided by the total number of retransmissions using a basic retransmission scheme, which retransmits each lost packet by itself without coding and corresponds to the retransmission scheme in IEEE 802.11. A lower retransmission ratio indicates fewer retransmissions, and hence is preferred. We calculate the retransmission ratio for every 200 new packets that sender sends to each of the clients. Then we compute the average and standard deviation of retransmission ratios over 10 runs. Under all cases, the standard deviation of retransmission ratios is low – typically around 0.02 and no more than 0.08 over all the runs. So in the interest of space and clarity, we only present the average retransmission ratios in the following evaluation results.

To ensure the same level of reliability, all retransmission schemes use an unlimited number of retransmissions so that they all achieve 100% delivery rate. Our results of bounded retransmissions are qualitatively similar. The only difference is that under extremely high loss rates (*e.g.*, 90%), the retransmission ratio under bounded retry count approaches 1 because the numbers of retransmissions under both the basic and coding algorithms are determined by the retry count. In such cases, the coding based retransmission schemes deliver more packets successfully. Therefore in the interest of brevity, we will focus on the performance of unbounded retry count in this section.

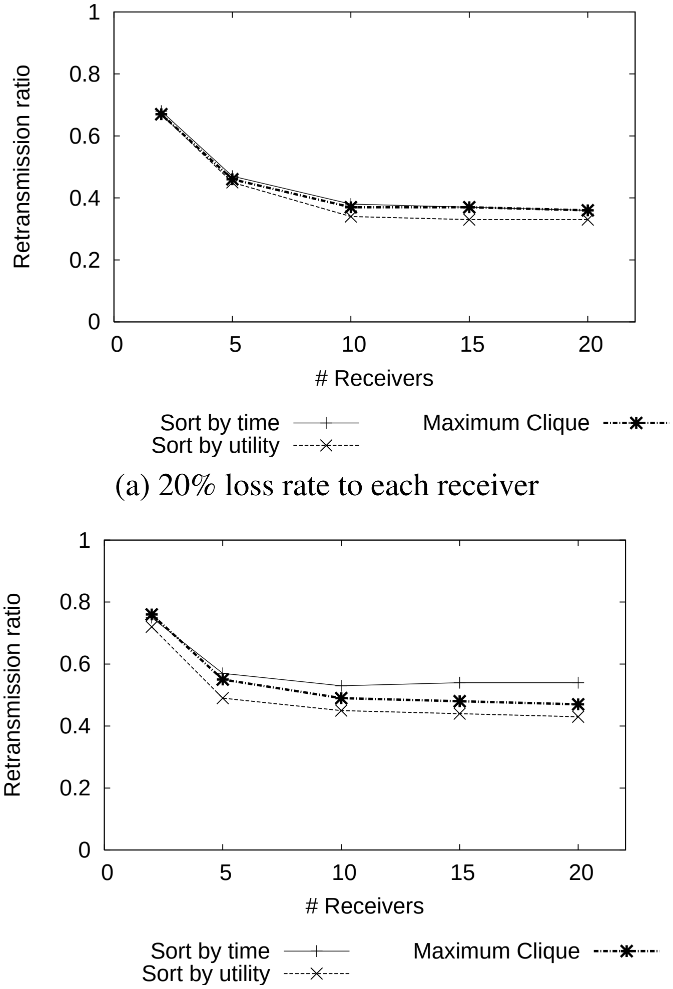
4.3.2 Simulation Results

First we present the simulation results of multicast by varying the number of receivers, loss rates, and batch sizes. Then we present the unicast performance results.

4.3.3 Multicast Results under Homogeneous Loss Rates

Varying the number of receivers: Figure 4.2 and Figure 4.3 show retransmission ratios with a varying number of clients under Bernoulli and Gilbert loss models, respectively. We make the following observations.

First, in all cases the coding-based retransmissions yield retransmission ra-

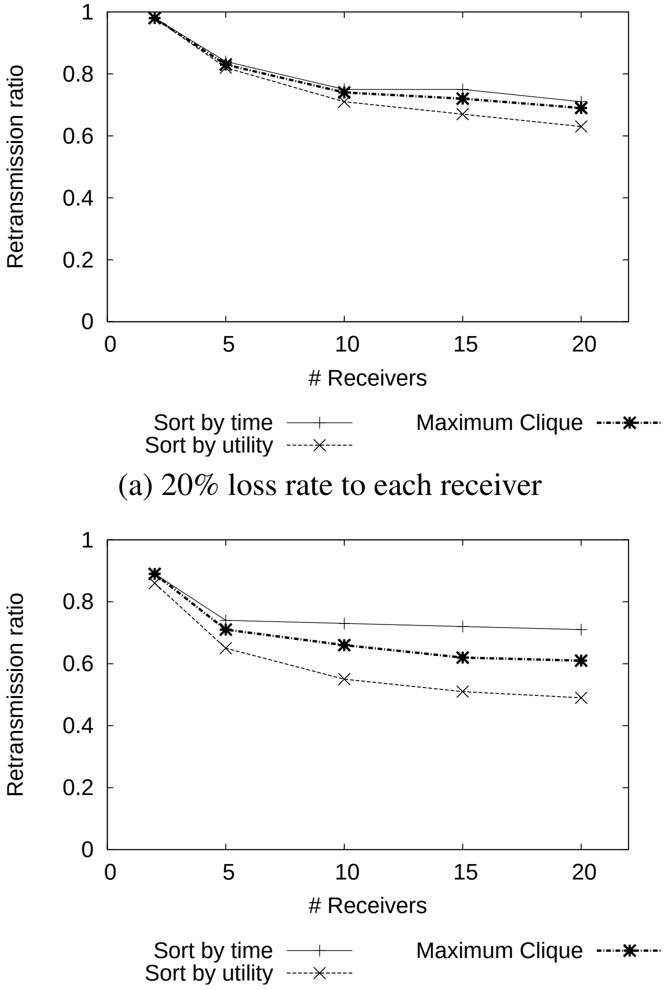


(a) 20% loss rate to each receiver

(b) 50% loss rate to each receiver

Figure 4.2: Multicast comparison under a varying number of receivers with homogeneous Bernoulli losses.

tios below 1. This indicates that the coding-based retransmissions is more efficient than the basic retransmission. The lowest ratios achieved are around 0.4, reducing the total number of retransmissions by 60%.



(a) 20% loss rate to each receiver

(b) 50% loss rate to each receiver

Figure 4.3: Multicast comparison under a varying number of receivers with homogeneous Gilbert losses.

Second, the retransmission ratios decrease with the number of receivers, which suggests that the benefit of coding-based retransmissions increases with the number of receivers. This is because a larger number of receivers makes it easier to find receivers that lose different packets and create coding opportunities.

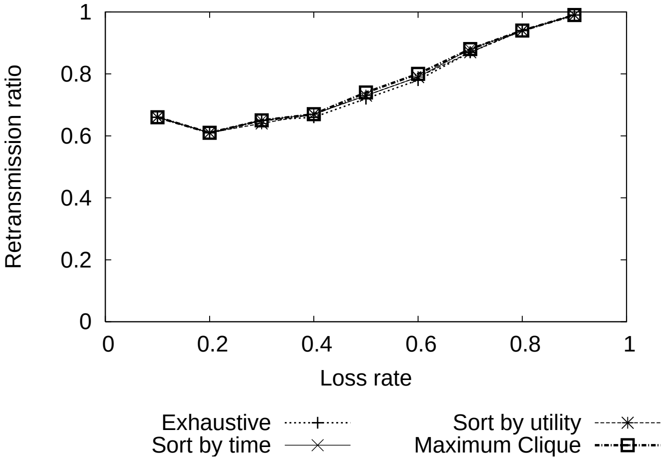


Figure 4.4: Multicast with 3 clients under a varying loss rate with homogeneous Bernoulli losses.

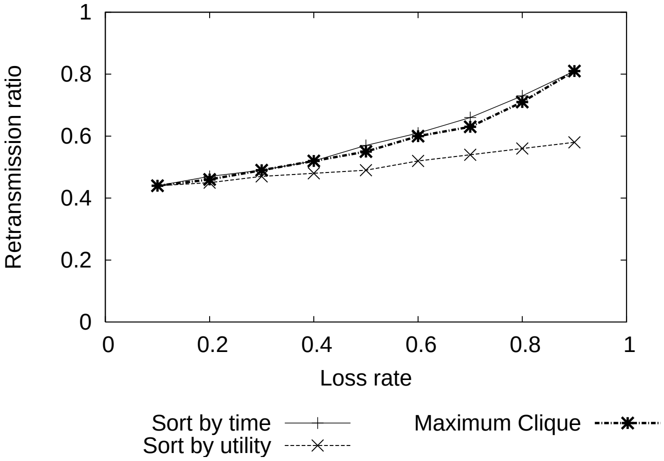


Figure 4.5: Multicast with 5 clients under a varying loss rate with homogeneous Bernoulli losses.

Third, comparing the three different coding algorithms, we observe the sort-by-utility algorithm out-performs the maximum clique, which out-performs the sort-by-time. Their performance difference is larger under the Gilbert loss model than under the Bernoulli loss model. The good performance of the sort-by-utility algorithm is likely because packets lost at many nodes are harder to find other packets to code with (in the extreme, the packets lost at all nodes have to be retransmitted by itself); sending them earlier makes it easier to find packets to code with since there are more candidates to choose from. In addition, sending them earlier helps to create coding opportunities for future retransmissions as coding opportunities arise after enough packets are received. The larger benefit under the Gilbert loss model is likely because utility distribution is more skewed under the Gilbert loss model and the sort-by-utility algorithm makes a larger difference. In the interest of brevity, below we present the results under the Bernoulli loss model, and comment on the the Gilbert results whenever their difference is significant.

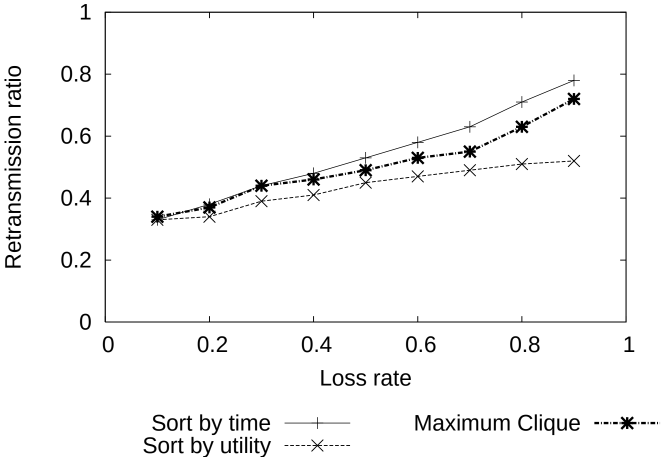


Figure 4.6: Multicast with 10 clients under a varying loss rate with homogeneous Bernoulli losses.

Varying loss rates: Next we evaluate the performance by varying loss rates. Figures 4.4, 4.5, and 4.6 summarize the results under 3, 5, and 10 receivers, respectively. For 3 receivers, we also plot the results of the exhaustive search; the results of the exhaustive search under a higher number of receivers are not available due to its high computational complexity. Under 3 receivers, the practical coding schemes perform almost the same as the exhaustive search, with all curves overlapping with each other. This further confirms the effectiveness of the coding heuristics. Under 5 and 10 receivers, the retransmission ratios of three coding heuristics are between 0.35 and 0.8, cutting the number of retransmissions by 20% to 65%. The sort-by-utility continues to perform the best. In all cases, the ratios are lowest under low packet loss rates because the packets lost at different receivers are more likely to be different under low loss rates and create more coding opportunities.

Varying batch sizes: We further evaluate the impact of batch sizes. As shown in Figure 4.7, with an increasing batch size, the retransmission ratio decreases and coding benefit increases. When the batch size is 5 packets, the coding-based retransmission schemes already achieve the ratio below 0.6. When the batch size increases to 50, the ratios are as low as 0.3. This shows that there is a tradeoff between packet delay and bandwidth saving. The good news is that only a small batch (or delay) is needed to achieve significant saving.

4.3.4 Multicast Results under Heterogeneous Losses

So far we consider similar loss rates between the sender and all its receivers. In the following evaluation, we consider heterogeneous loss rates. We assign the average loss rate to each client, randomly chosen between 0 and the loss bound. In this case, the difference between loss rates across different clients is up to the loss

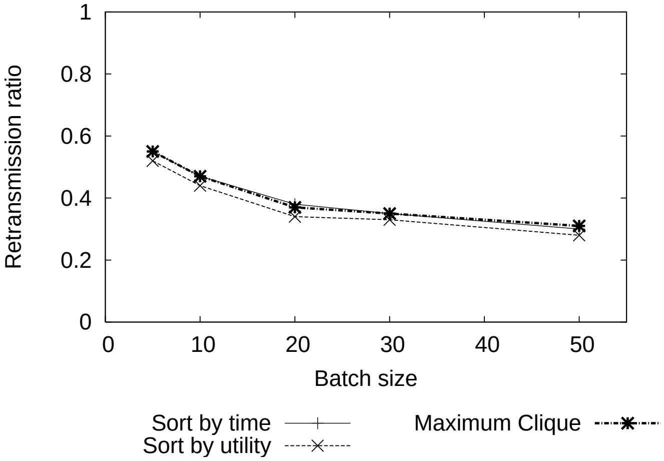
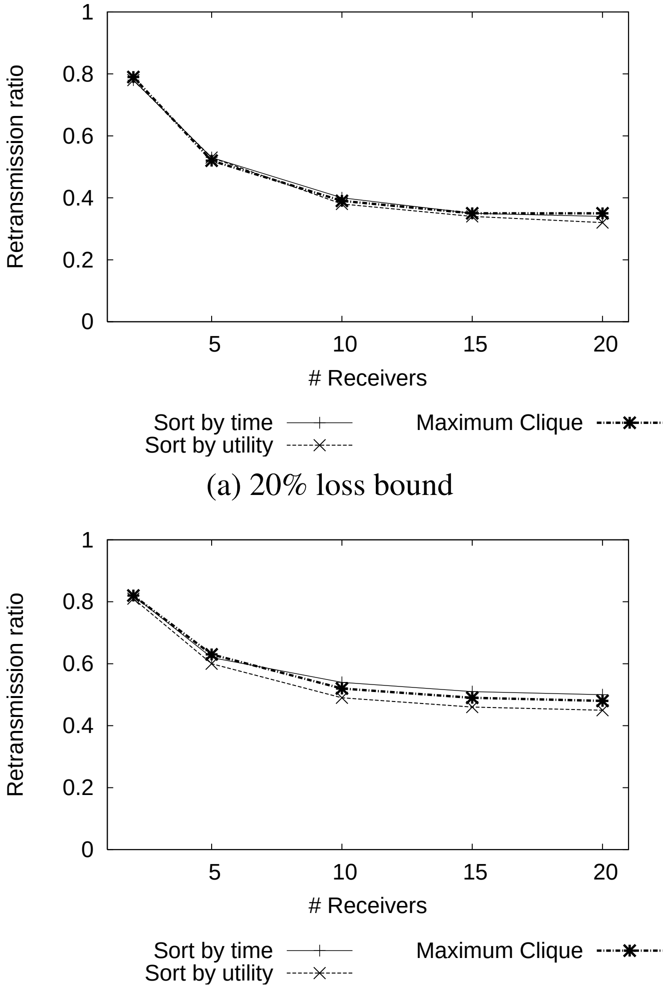


Figure 4.7: Multicast comparison under a varying batch size with 10 receivers and 20% homogeneous Bernoulli loss rates.

bound.

Varying the number of receivers: Figure 4.8(a) and (b) show the results under 20% and 50% loss bounds, respectively, where the number of receivers varies from 2 to 20. As we can see, the coding-based retransmission schemes significantly outperform the basic retransmission, with retransmission ratios ranging between 0.4 and 0.8. However, the difference across different coding algorithms is small under Bernoulli losses. The difference under the Gilbert loss model (not shown) is larger, with the same ranking as before and the sort-by-utility out-performing the sort-by-time by up to 25%.

Varying loss bounds: We further evaluate the heterogeneous cases by varying the loss bound. Figure 4.9 summarizes the results under 5 and 10 receivers. As the loss bound increases, loss heterogeneity increases, which increases the retransmission

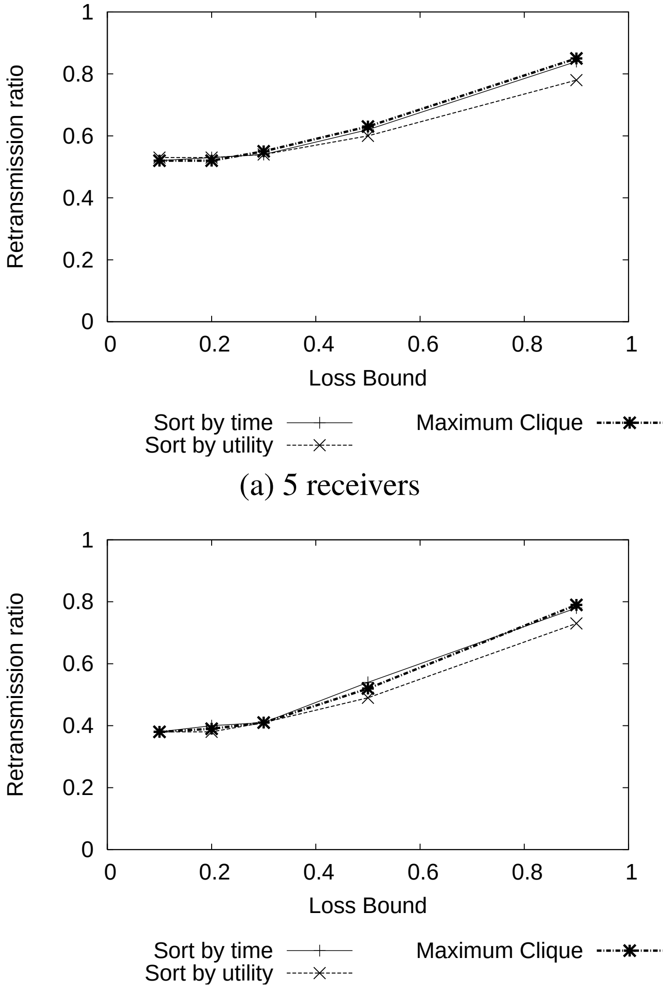


(a) 20% loss bound

(b) 50% loss bound

Figure 4.8: Multicast comparison under a varying number of receivers with heterogeneous Bernoulli losses.

ratio and decreases the coding benefit. This is expected because under higher loss heterogeneity most of the retransmissions are sent to one or few receivers and such imbalanced retransmission load makes it hard to find coding opportunities.



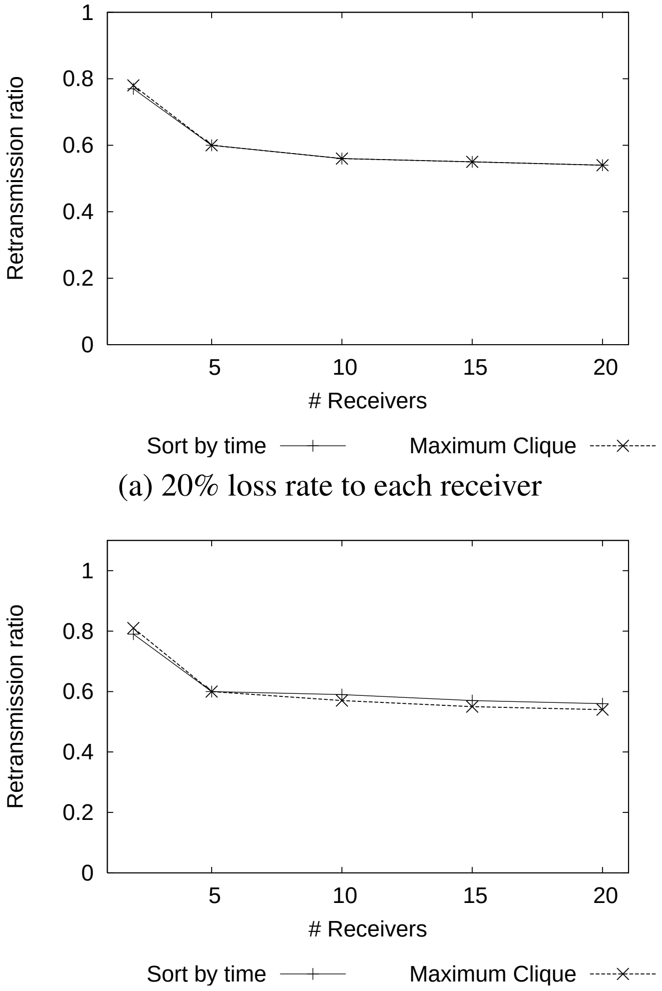
(a) 5 receivers

(b) 10 receivers

Figure 4.9: Multicast comparison under a varying loss bound with heterogeneous Bernoulli losses.

4.3.5 Unicast Results under Homogeneous Losses

In the following two sections, we evaluate the performance of unicast retransmission schemes under homogeneous and heterogeneous losses. Since the sort-by-utility and sort-by-time algorithms are equivalent under unicast, we only compare the sort-by-time and maximum clique with the basic retransmission.



(a) 20% loss rate to each receiver

(b) 50% loss rate to each receiver

Figure 4.10: Unicast comparison under a varying number of receivers with homogeneous Bernoulli losses.

Varying the number of receivers: Figure 4.10(a) and (b) show the results under a varying number of receivers when the loss rate to each receiver is 20% and 50%, respectively. In both cases, the coding-based schemes achieve retransmission ratios between 0.6 and 0.8. As in multicast cases, with an increasing number of receivers, the retransmission ratio decreases and the coding benefit increases.

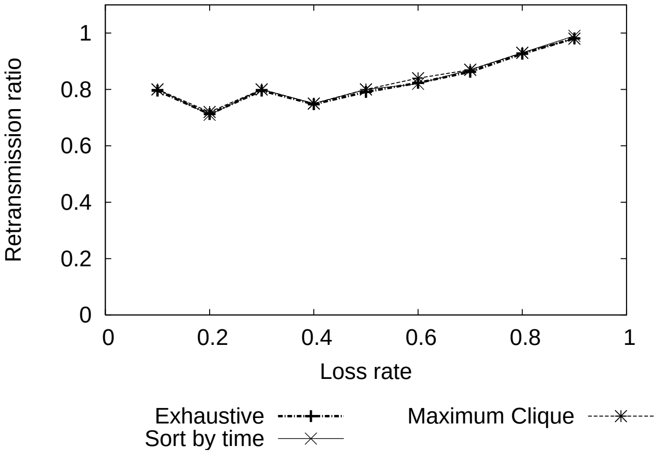


Figure 4.11: Unicast comparison for 3 clients under a varying loss rate with homogeneous Bernoulli losses.

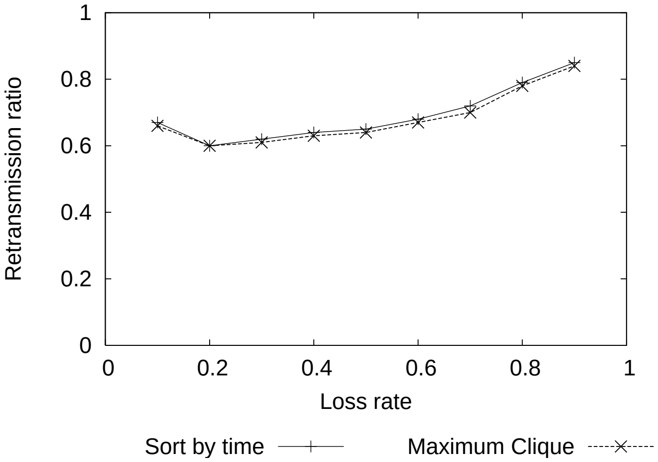


Figure 4.12: Unicast comparison for 5 clients under a varying loss rate with homogeneous Bernoulli losses.

Varying loss rates: Figures 4.11, 4.12, and 4.13 show the results under a vary-

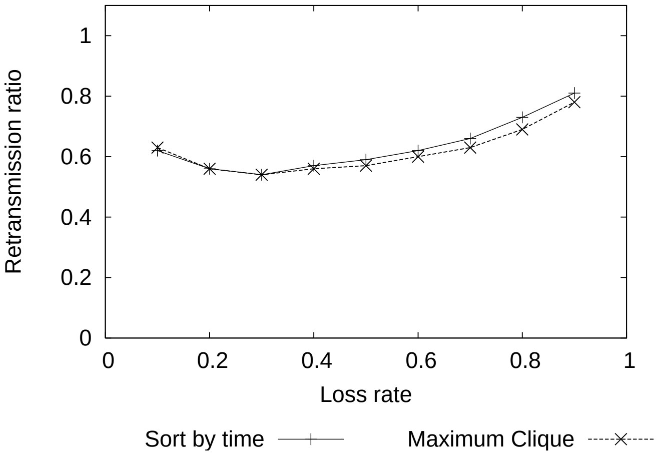


Figure 4.13: Unicast comparison for 10 clients under a varying loss rate with homogeneous Bernoulli losses.

ing loss rate for 3, 5, and 10 receivers, respectively. Under 3 receivers, the coding heuristics are compared against the exhaustive search, and they all perform similarly, indicating the effectiveness of the heuristics. Compared with the multicast performance in Figures 4.4, 4.5, and 4.6, the coding benefit of unicast retransmissions under the corresponding loss rates are smaller. This is because coding gain in multicast cases arises whenever receivers obtain different sets of packets, whereas coding in unicast not only requires the above condition but also requires that packets that the receivers lose are destined to them (*i.e.*, receivers do not care if they lose packets destined to other nodes). The additional coding constraint reduces the coding opportunities.

Varying batch sizes: Figure 4.14 shows the result of varying batch size. As the batch size increases, the retransmission ratio decreases and coding benefit increases. This is consistent with multicast results, since a larger batch size has more packet

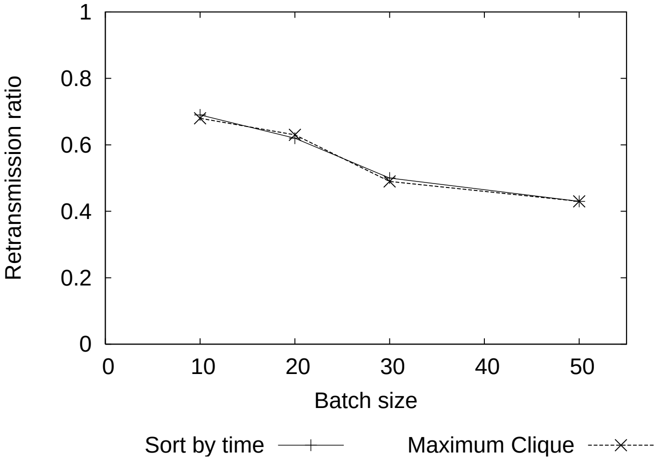


Figure 4.14: Unicast comparison under a varying batch size with 10 receivers and 20% homogeneous Bernoulli loss rates.

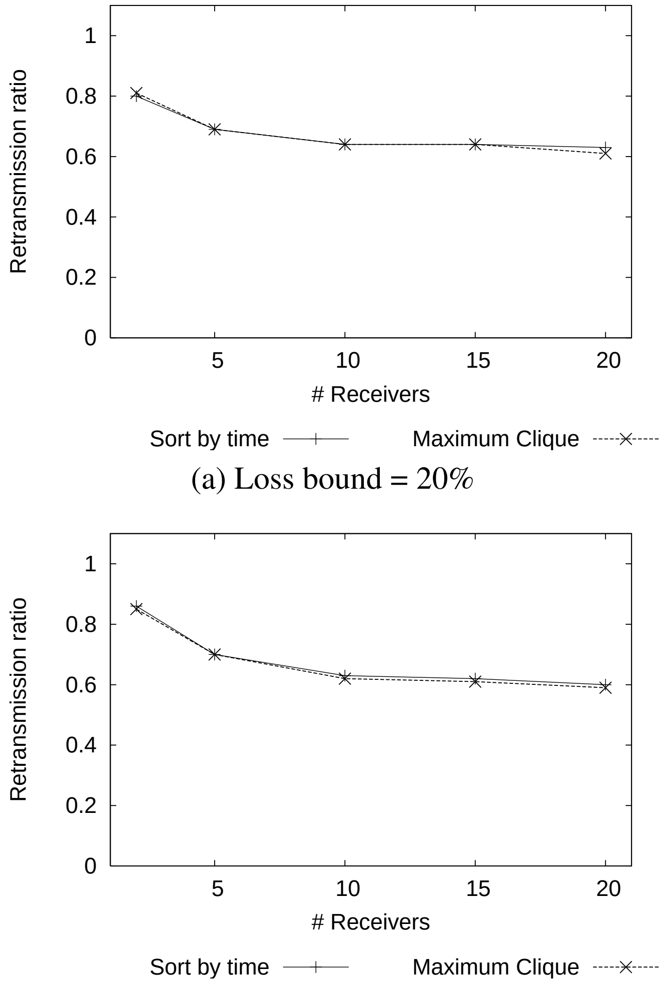
combinations to choose from and increases the coding benefit.

4.3.6 Unicast Results under Heterogeneous Losses

We also evaluate the retransmission schemes under unicast traffic using heterogeneous losses.

Varying the number of receivers: First we vary the number of receivers with the loss bound of either 20% or 50%. As shown in Figure 4.15, in both cases, the retransmission ratio is between 0.6 and 0.8. As in multicast cases, the lower retransmission ratios (or higher coding benefit) is achieved under a larger number of receivers due to more coding opportunities.

Varying loss bounds: We further evaluate the performance by varying the loss bounds while setting the number of receivers to 5 or 10. Figure 4.16 shows that

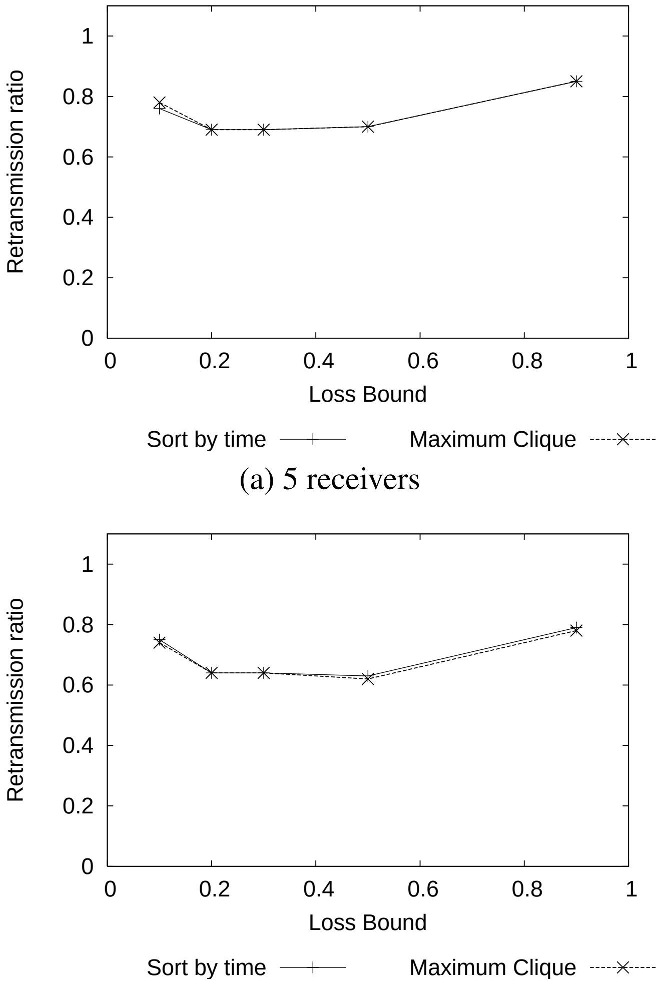


(a) Loss bound = 20%

(b) Loss bound = 50%

Figure 4.15: Unicast comparison under a varying number of receivers with heterogeneous Bernoulli losses.

the retransmission ratio initially decreases and then increases with the loss bound. The later increase is due to the same reason as in the multicast cases, where under higher loss bounds most retransmissions are towards one or few receivers and hard to code them with other packets. The initial decrease is likely because coding unicast retransmissions requires an additional constraint that different nodes miss their own packets, and increasing the loss bound initially helps to increase the likelihood of satisfying this constraint.



(a) 5 receivers

(b) 10 receivers

Figure 4.16: Unicast comparison for unicast under a varying loss bound with heterogeneous Bernoulli losses.

4.3.7 Summary

The simulation results show that coding-based retransmissions are effective in reducing the number of retransmissions required to recover packet losses. Their performance benefit increases with the number of receivers and the batch size. Moreover their performance gain is larger for multicast traffic. Comparing different coding-based heuristics, the sort-by-utility performs the best.

4.4 Implementation and Testbed Experiments

In addition to high-level simulation, we also implement different retransmission mechanisms in a wireless testbed. Testbed experiments are valuable because they allow us to evaluate the ER protocol under realistic scenarios. In this section, we first describe our testbed implementation and evaluation methodology, and then present the performance results.

4.4.1 Implementation

Our implementation is built on the COPE source code [31], which performs network coding at intermediate nodes in multihop wireless networks. We make the following modifications to support ER. We modify the receiver feedback scheme in COPE as described in Section 4.2.2. We implement the scheduling algorithm described in Section 4.2.3 to determine whether a packet needs a retransmission and when a retransmission should be sent. In addition, we disable MAC-layer retransmissions used in COPE. Instead, we implement two retransmission mechanisms above the MAC-layer: (i) the basic retransmission, which keeps sending a lost packet until all the intended receivers acknowledge it or the maximum retry count is reached, and (ii) the coding-based retransmission using the sorted by time heuristic. The maximum retry count is 7 in both basic retransmission and the coding-based retransmission to achieve similar level of reliability, and 7 is commonly used retry count in IEEE 802.11. We plan to evaluate the performance of other coding heuristics as part of our future work.

4.4.2 Experiment Methodology

We set up a wireless testbed that consists of 7 DELL Dimension 1100 PCs. The testbed spans one floor of an office building. Each machine has a 2.66 GHz Intel Celeron D Processor, and runs Fedora Core 4 Linux. Each is equipped with 802.11 a/b/g NetGear WAG511 using MadWiFi. RTS/CTS is disabled as in the default setting. Our experiments use 802.11b. To avoid interference with resident wireless networks, we run our experiments during nights and weekends. We use 1 AP as a sender, and use up to 6 clients as receivers. The loss rates between the AP and clients are generated in a controlled manner to evaluate the performance under various loss scenarios. We impose a specific loss rate on each wireless link by artificially dropping traffic at the receivers, and the dropped packets are not acknowledged by the receiver's feedback in ER. Unless otherwise specified, the packets are dropped using the Bernoulli loss model and all clients experience similar loss rates. For each scenario (*i.e.*, a given number of receivers and loss rate), we run seven times, where each time we obtain the retransmission ratio (defined in Section 4.3.1) by letting the AP send 1000 packets. We then plot retransmission ratios using errorbars, where the center of an errorbar corresponds to the mean and the length of the errorbar is twice the standard deviation over seven runs. In addition, we compare the total throughput of ER and the basic retransmission by running a 30-second UDP transfer, and report throughput ratio, defined as the ratio of the ER's throughput against that of the basic retransmission scheme. A higher throughput ratio indicates a larger performance gain from ER.

4.4.3 Experiment Results

4.4.3.1 Multicast Evaluation

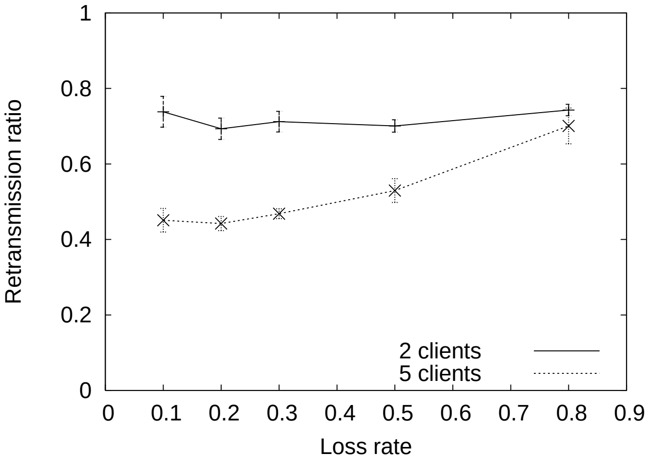


Figure 4.17: Multicast experiment results under a varying loss rate.

We first evaluate the multicast performance of ER by varying the loss rates. Figure 4.17 summarizes the results under 2 and 5 clients. The retransmission ratio is between 0.7 – 0.8 for 2 receivers, and between 0.4 – 0.7 for 5 receivers. These results are consistent with the simulation results, indicating that the benefit of ER extends to real wireless networks.

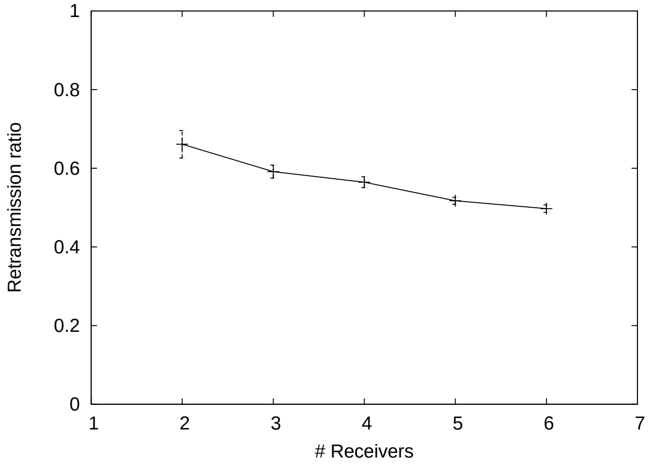


Figure 4.18: Multicast experiment results under a varying number of clients.

We further evaluate the performance using a varying number of receivers while keeping the loss rate to each client around 50%. As shown in Figure 4.18, the retransmission ratio decreases from 0.7 to 0.5 as the number of receivers varies from 2 to 6. This shows that the benefit of ER increases with the number of receivers, which is consistent to the simulation results.

Finally we compare the throughput of ER and basic retransmission by varying the loss rate to each client. As shown in Figure 4.19, the throughput gain from ER can be quite significant: up to 21% gain for 2 clients, and up to 50% gain for 5 clients. Moreover, the throughput gain tends to increase with loss rate, because ER improves the efficiency of retransmission, which is more important under high loss rates.

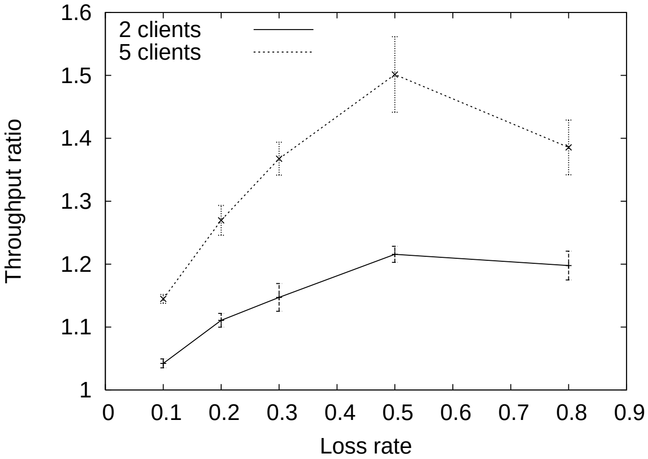


Figure 4.19: Compare throughput against the basic scheme for multicast under a varying loss rate.

4.4.3.2 Unicast Evaluation

Next we evaluate the unicast performance of ER. Figure 4.20 shows that the retransmission ratios are between 0.6 and 0.8 as the loss rate varies from 0.1 to 0.8. The retransmission ratio is lower under 5 clients than under 2 clients, as we would expect.

To further quantify how the loss patterns affect the coding benefit, we impose a different loss characteristic – only packets destined to the receivers are dropped according to the specified loss rate, while all the other packets incur no artificial losses. In this case, the packets lost at different receivers are guaranteed to be different, and this increases the coding opportunities. Therefore we observe a lower retransmission ratio, between 0.2 and 0.8, as shown in Figure 4.21.

We also evaluate the performance by varying the number of clients and

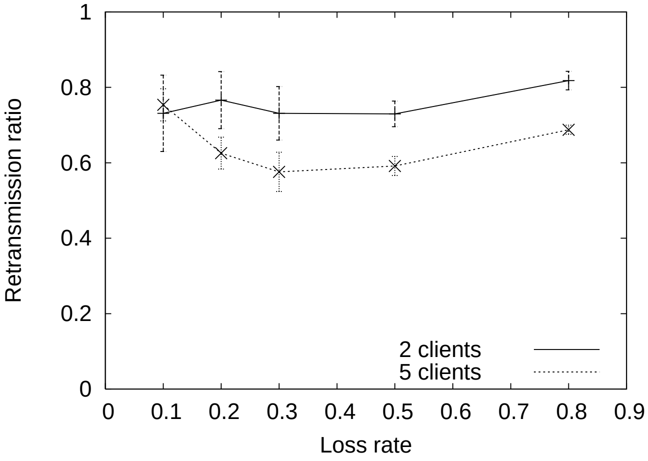


Figure 4.20: Compare retransmission mechanisms for unicast under a varying loss rate.

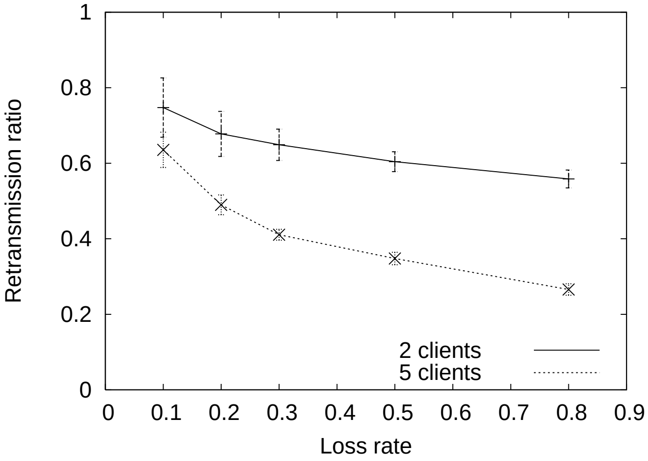


Figure 4.21: Compare retransmission mechanisms for unicast under a varying loss rate, where only packets destined to the receivers are dropped.

keeping the loss rate to each client to be around 0.5. As shown in Figure 4.22, the retransmission ratio is between 0.6 and 0.7. Moreover, the ratio tends to decrease with the number of receivers, as we would expect.

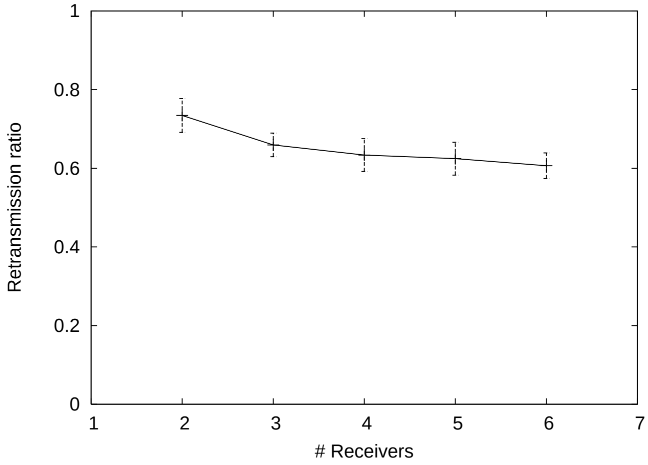


Figure 4.22: Unicast experiment results under a varying number of clients.

Finally we evaluate the performance in terms of throughput ratio by varying the loss rate to each client. As shown in Figure 4.23, the improvement of ER under unicast is generally less than under multicast, as we would expect. Nevertheless, we observe that throughput improves by up to 17% for 2 clients and up to 25% for 5 clients. As in multicast, the performance gain of ER under unicast also increases with loss rates, since ER helps to reduce more packet retransmissions under higher loss rates.

4.5 Summary

In this chapter, we develop ER to efficiently support retransmissions in wireless networks for both unicast and broadcast/multicast traffic. ER reduces the number of required transmissions to recover packet losses by coding packets lost at different receivers. Using simulation and experiments, we show that ER is effective

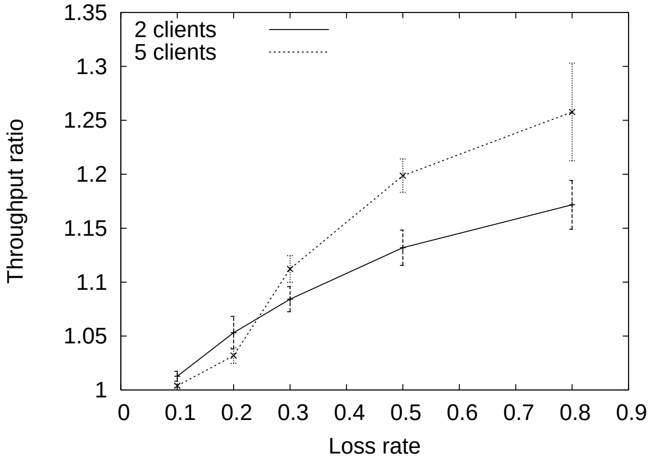


Figure 4.23: Compare throughput against the basic scheme for unicast under a varying loss rate.

over a wide range of scenarios.

Chapter 5

Physical Layer

After investigating how to reduce loss in the Link Layer, we now turn our attention to the Physical Layer. We present our case study below.

5.1 Introduction

Motivation: Our motivation is to enable the next generation of wireless applications. We envision bandwidth-intensive, real-time application systems that need to rely on a robust wireless link. Specifically, we are interested in supporting a high throughput link with low latency. There are a variety of potential uses for such a link. For example, high definition security and video conferencing systems may depend on real-time information. Commercial products have shown high-definition video can be streamed over high-throughput channels [100, 125, 152], but we want to know if we can support these videos in *real-time*. One potential near-term application is high-definition gaming systems. For instance, the Nintendo Wii U [107] has a wireless controller with a screen to stream video content. If multiple video streams (say up to four) are included, each with a 20 Mbps bitrate (1080p video under the H.264/AVC codec [140]), then 80 Mbps total is required to support all four clients. Furthermore, latency requirements are typically low for these systems, and the given that all nodes may be sharing the same frequency, a delay constraint of 4 ms could be imposed. [1]

Approach: However, under traditional 802.11a/b/g technologies, 80 Mbps is not achievable. The maximum physical layer data rate is 54 Mbps, but that number is not even achieved in practice due to MAC-layer overheads. Therefore, we focus our attention on the 802.11n standard [2]. Products today support 3x3 MIMO communication with physical-layer bitrates up to 450 Mbps (although the standard allows for up to 4x4 MIMO and 600 Mbps). The goal of our study is to ask ourselves if we can achieve high throughput and low latency with today's 802.11n technologies. To this end, we present a case study of 802.11n technologies in a living room environment. Our goal is to see when the signal strength is strong, what bounds in terms of throughput and latency can 802.11n technologies provide. In our approach, we examine the performance of the wireless link under a variety of 802.11n configurations. The 802.11n standard provides much more flexibility than previous standards. Nodes can support multiple rates, multiple spatial streams, multiple channel widths, and multiple preamble formats. We examine the performance of the link while varying these parameters.

Findings: In our experiments, we find that static and stable wireless environments can come close to meeting both high throughput and low latency demands. However, when a typical use case is encountered, client mobility, we find that significant delays can be imposed on wireless traffic. We examine the source of these

[1] 1080p60 video displays 60 frames per second, which equates to 16 ms per frame. If there are four clients, then a new frame is received every 4 ms

delays and find that they are due to burst errors. Burst errors are compounded in 802.11n technologies because the standard relies on packet aggregation in order to reduce MAC-layer inefficiencies. When multiple aggregated packets are lost consecutively, the latency on the channel can reach up to 40-60 ms. We find that this happens regardless of the client's configuration. To that end, we debug the burst losses and find that a significant number of them are not due to a few corrupt bits within the payload of the packet. Instead, consecutive aggregate packets are getting lost.

From this debugging, we then set out to make recommendations to reduce loss in mobile environments. Our recommendations center around ensuring the packets do not get completely lost. We outline our recommendations and benchmark possible solutions. To further the case study, we examine previous work to see if it is effective in mitigating the problem.

5.2 Methodology

In this section, we discuss the platform developed to measure the results, the hardware and client configurations used, and describe how the tests were performed.

Configurations: In order to ensure our results are general, we first employ a variety of 802.11n chipsets. Each chipset is manufactured by a different vendor. Specifically, we evaluate the Atheros AR9380 chipset, the Quantenna QHS600 solution, the Marvell Smart Wi-Fi 3x3 450 Mbps Dual-band Access Point and the Broadcom Intensi-fi 802.11n Dual-Band 3x3 chip. Each chip is capable of 3x3 MIMO communication. Our goal is to see if there are any main differences between the performance of each chip-maker and if there are any problems that are consistent across chip-makers.

In 802.11n, clients have a variety of configuration options. For instance, cards can be configured to work with 20/40 Mhz channels, varying levels of packet aggregation, data rates and spatial streaming. Furthermore, optional 802.11n features, like the Greenfield preamble, can be enabled. In our tests, we look to enable and disable these features and measure the performance under varying client configurations. We want to ensure that potential limitations are not a product of an incorrect client configuration. For all tests, we disable beam-forming and retransmissions, and set the guard interval to the long guard interval.

Platform: While we investigate the performance of a variety of chipsets, we focus mostly on results obtained from the Atheros and Marvell cards. We find these cards to be the most stable (for instance, the Quantenna system's processor was not fast enough to handle high 3x3 data rates) and the best performers. Our results presented typically focus on those two chipsets, but the trends that we see were witnessed across all chipsets. We specifically focus on Atheros because it gave us the most control over the wireless card. The Atheros chipset was installed in a Linux laptop with the `ath9k` open-source driver. The other chipset solutions were provided as stand-alone boxes, and therefor getting detailed per-packet information from them proved challenging.

We modified the `ath9k` driver to support our needs. We provided functionality to easily change wireless settings. For instance, on the sender, we allowed different fixed packet aggregation sizes. The hardware in the Atheros chipset limited the number of packets in an aggregate to 32. However, our modifications allowed smaller aggregates to be fixed so we could evaluate different packet sizes. We further modified the rate-control algorithm to send at a pre-specified, fixed wireless rate. By altering the rate, we also dictate how many spatial streams are used for communication.

The main goal of our work was to measure the throughput and latency of the wireless link. Measuring the throughput is easy, and we used the `iperf` open-source application [60] to measure saturated UDP throughput. We saturate the medium because we want to quantify the exact behavior of the link. If the medium isn't saturated, potential packet losses could occur during the inter-packet arrival times that aren't in use. We minimize this effect by constantly sending consecutive packets. To measure the latency, we make several modifications to the driver. We integrate the driver with `iperf`-specific sequence numbers. We intercept each packet before it is sent on the wireless card and replace the `iperf` sequence number embedded in each packet with our own counter in the MAC layer. This ensures that any packets dropped in any buffers between the application layer and MAC layer will not be counted as wireless losses. Each individual packet is also modified to contain information about the AMPDU (the aggregated packet it is a part of). We also add a AMPDU sequence number and the AMPDU size, allowing the receiver to detect how many AMPDUs were lost and how many packets may have been lost inside a partially-received AMPDU aggregated packet. During the transfer, we log all outgoing packets and AMPDUs at the sender and modify the driver to provide debug-level output about which packets are lost. We can correlate the two traces to obtain insights about the loss patterns in the network.

Testing: We conduct tests by having the sender send UDP traffic to the receiver. The sender sends 1500 byte packets. We place our two nodes in close proximity (about 3 feet apart) in order to achieve high received signal strength. All tests have a direct line-of-sight from the sender to the receiver and the sender sends with maximum transmit power. When we do mobility experiments, we place the receiver on a cart and walk slowly (about 1 m/s) away from the sender and then back (at most 15-20 ft away). We repeat this motion for the total length of the experiment. To ensure there is limited interference, all of our results are run late and night and during the weekends. There are few people in the office during this time. We have tried different mobility patterns and have witnessed consistent trends. Furthermore, we monitor the wireless channels to ensure they aren't being used and pick the cleanest channel. We also repeated some experiments late at night in an underground parking garage (4 stories underground) to confirm the results we saw were consistent with previous results done in the office.

5.3 Measurements

In this section, we present the measurement results from our case study. We first briefly measure throughput and then focus on latency.

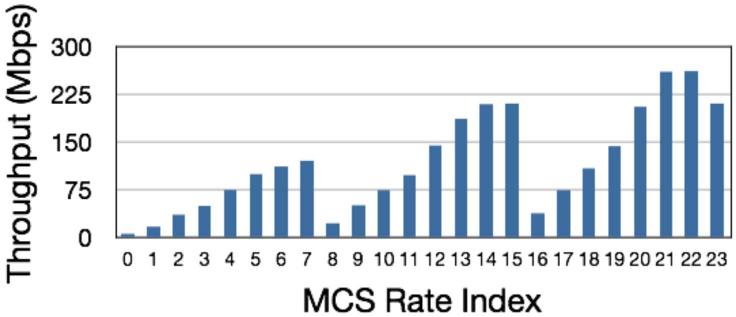


Figure 5.1: Measuring Marvell 802.11n throughput with strong signal strength.

5.3.1 Throughput

We measure the throughput in an ideal, static environment. The sender and receiver are close by, as described in Section 5.2. We send saturated UDP traffic, with packet sizes of 1500 bytes. We let the driver aggregate as many packets as possible into its AMPDUs. We plot the performance of the Marvell card in Figure 5.1 for each MCS index from 0 to 23. MCS rate indices 1-7 use one spatial stream, indices 8-15 use two spatial streams, and indices 16-23 use three spatial streams. We see that the throughput increases within each spatial stream. We see that there are quite a few rates that have no troubles supporting high throughput. There are many streams that provide over 80 Mbps throughput (5-7, 11-14, and 18-23). We also found that MCS indices higher than 19 were less stable across all the chipsets (the same applies for MCS 15): the losses they suffered were more sporadic and inconsistent in nature, and furthermore the Quantenna chipset typically maxed out at 150 Mbps. Since we are interested in low latencies and controlled experiments, this thesis will only focus on the most robust and consistent rates.

Next, we look what happens when we impose artificial limits on the AMPDU size. We pick five MCS rate indices that perform well (11, 12, 18-20) and plot

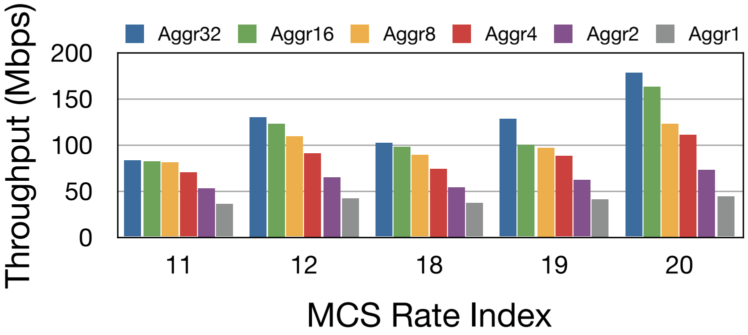


Figure 5.2: Measuring 802.11n throughput when varying aggregate sizes.

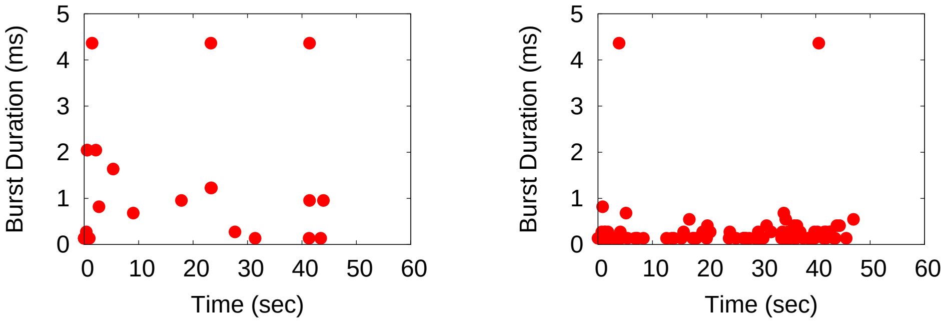
the throughput of each for a varying aggregate size. The results are in Figure 5.2. For MCS11, an aggregate size of 8 represents a small reduction in throughput. However, for higher rates, the overhead becomes much higher (17% for MCS12, 13% for MCS18, 25% for MCS19 and 31% for MCS20). Without any aggregation (aggregation size set to 1), the throughput of the slowest rate (MCS11) drops by 57% to only 36 Mbps.

Summary: We can see that 802.11n provides high throughput under high signal strength environments. There are multiple rates available that can provide high throughput, so it is important to see the latency provided under each rate. We see that packet aggregation can have a potentially large impact on performance, and that some aggregation should be used in order to maintain efficiency.

5.3.2 Latency

In order to effectively support real-time communications, the latency of a wireless link must be low. In the previous subsection, we saw that packet aggregation is necessary to maintain high MAC efficiency, and thus throughput. In this section we analyze how wireless losses, coupled with packet aggregation, impact the latency on a wireless link. We look at a variety of environments (static versus mobile) and client configurations.

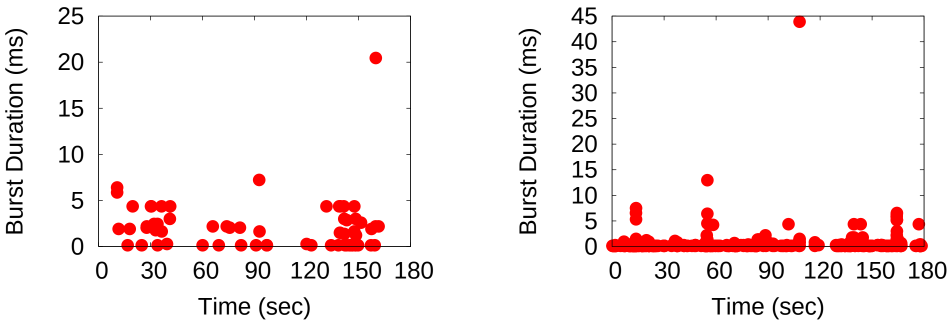
5.3.2.1 Environment Comparison



(a) Atheros chipset (b) Marvell chipset

Figure 5.3: Time-series loss for static environment for two different vendors.

Static Nodes: For these tests, we fix the rate to MCS11, which gives a throughput of around 90 Mbps, and send saturated UDP traffic from the sender to the receiver. Of all the rates, we found that MCS11 seemed to be the most stable– it provides high throughput consistently under a variety of chipsets and still allows for additional receiver diversity since it only utilizes 2 spatial streams. Packet sizes are 1500 bytes and they are typically aggregated into an AMPDU that contains 32 packets. In MCS11, the maximum aggregate size is 32 and the one aggregated packet takes roughly 4 ms to transfer. The graphs in Figure 5.3 show a time-series of the loss of a 60 second transfer for the Atheros and Marvell chipset. Each point represents a burst loss, or a series of consecutively lost packets. We can see that bursts occur infrequently, and when they do happen, they typically have a short duration.



(a) Atheros chipset

(b) Marvell chipset

Figure 5.4: Time-series loss for mobile environment for two different vendors.

Mobile Nodes: Next, we make the client nodes mobile. As detailed earlier, we slowly move the clients (about 1 m/s) away from the sender and back. We repeat this movement throughout the run. The results are shown in Figure 5.4. We can see that now the burst losses increase in size and frequency. In the Atheros case, the worst burst loss is 5 AMPDUs long, which equates to roughly 20 ms of wasted air time. In the Marvell case, the worst case has up to 10 AMPDUs lost consecutively, which equates to roughly 40 ms of wasted time. The periods with no loss in the Atheros graphs represent starting and stopping mobility. We can see that mobility imposes significant delay on the wireless link. In the worst case, up to 20-40 ms of channel time is wasted because consecutive AMPDUs are lost.

5.3.2.2 Rate Comparison

Now we only examine the performance of the Atheros chipset because we were able to modify the driver to give us fine-grained trace collection. The performance of the other chipsets are comparable and are omitted for brevity. First we

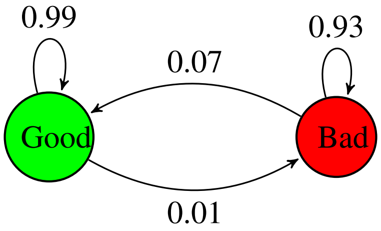


Figure 5.5: Gilbert-Elliott model for mobile environment with Atheros chipset.

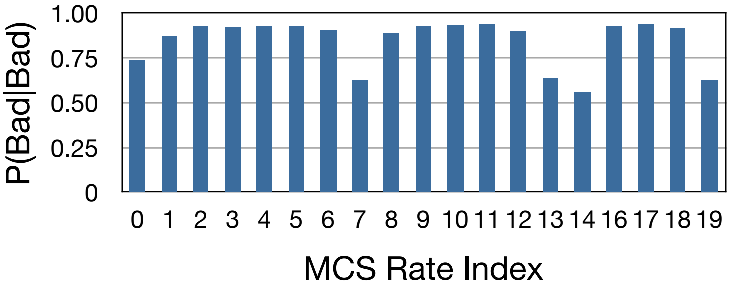


Figure 5.6: Varying rate.

need to define a metric that quantifies burstiness. A traditional model to classify burstiness is the Gilbert-Elliot model [45]. This model is a Markov model with two states: one for *good* (a packet is delivered successfully) and the other for *bad* (a packet is lost). The state transitions provide a notion of burstiness, especially the transition from the bad state to the bad state, referred to as P(bad|bad). The Gilbert-Elliot model from the Atheros mobility results is presented in Figure 5.5. Here, we can see the effects of a bursty channel: the probability of staying in a bad channel state is 93%. We also note that overall the loss is low. Typically the loss in all experiments is no greater than 5%.

Equipped with the Gilbert-Elliot model, we can now present the results of varying the data rate. The results are in Figure 5.6. Here, we only plot the probability of a burst loss– that is the conditional probability of staying in a lossy state when already in a lossy state. We can see that independent of rate, the likelihood to stay in a lossy state is high. Most rates have a 90% chance of staying in a lossy state. It is interesting that a few points in this graph have lower probabilities to stay in a lossy state (MCS 7, 13-14, and 19). This is a result of using less robust modulation techniques, which increases packet CRC errors. The packet CRC errors tend to be less bursty in nature because they corrupt only a portion of the data payload, rather than affecting the whole aggregate. This will be examined in further detail when we debug the losses in the next section.

Varying the rate also varies the amount of diversity the receiver can use. The receiver constantly receives on all three of its antennas. For example, when the sender uses a rate with only 1 stream (say MCS 4), the receiver is using all three of its antennas to receive the packet. It is interesting to note that even with low modulation techniques and one stream, we still witness burst losses.

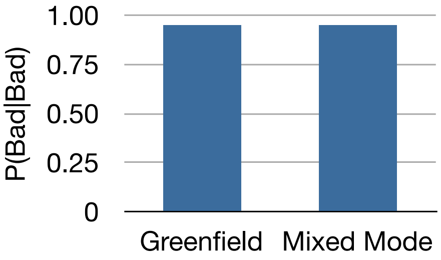
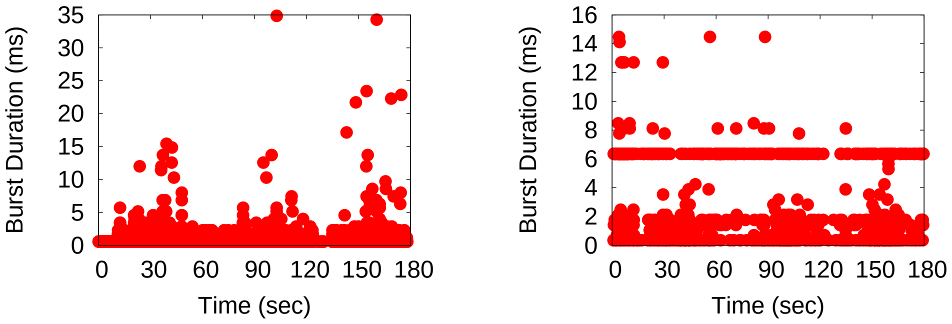


Figure 5.7: Varying 802.11n preamble modes.

5.3.2.3 Preamble Setting Comparison

802.11n comparison: In 802.11n, there are two preambles. There is an optional Greenfield preamble that is not backwards compatible with 802.11a/b/g, and a mandatory Mixed Mode preamble designed to be backwards compatible. We are interested to see if the preamble makes a difference. Figure 5.7 shows the conditional probability of staying in a bad state for each preamble setting. We see that there is little difference between the two preambles. Given that the channels we use for evaluation are clean, the Greenfield preamble should not suffer from any worse loss than the Mixed Mode preamble. However, the Mixed Mode preamble is slightly longer and has more overhead, so there is a possibility that its preamble could be more susceptible to loss. This result seems to indicate the Mixed Mode preamble isn't any less robust than the Greenfield preamble.



(a) 802.11a

(b) 802.11n

Figure 5.8: Time-series loss for mobile environment for 802.11a and 802.11n.

802.11a comparison: We were also interested if the performance of 802.11a would exhibit similar burst losses. The preamble for 802.11a is simpler and shorter than the 802.11n Mixed Mode preamble. Furthermore, 802.11a uses 20 Mhz channels, where as our results up to this point have used 40 Mhz channels. Since 802.11a provides significantly lower data rates than 802.11n, we were interested in seeing if the best 802.11a data rate performed in a similar manner to its equivalent 802.11n data rate. We set the 802.11a data rate to 54 Mbps and set the 802.11n sender to 20 Mhz and the equivalent modulation rate using 1 spatial stream. The results for latency are shown in Figure 5.8. We can see that 802.11a suffers from burst errors, causing latencies on the order of 35 ms.

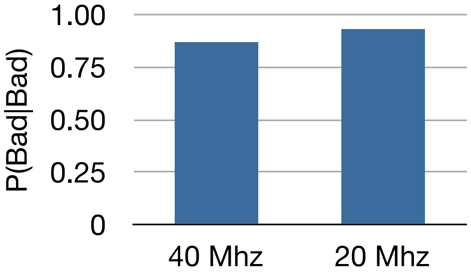
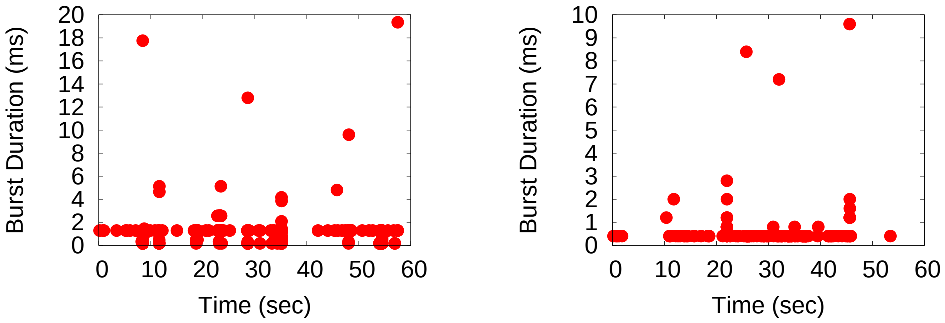


Figure 5.9: Varying 802.11n channel width.

5.3.2.4 Channel Width Comparison

Next we vary the channel width under 802.11n. We present the results of the probability of transitioning back into a bad state from a bad state in Figure 5.9. In this graph, we fix the modulation rate to MCS11 for the 40 Mhz channel and fix the modulation rate to MCS3 for the 20 Mhz channel (both MCS values use the same modulation type and coding rate). The probability for 40 Mhz channels is 0.88 and the probability for 20 Mhz channels is 0.93. Also note that the results from 802.11n in Figure 5.8 also use a 20 Mhz channel. These results confirm burstiness regardless of channel width.



(a) 8 packets

(b) 1 packet

Figure 5.10: Time-series loss for mobile environment with varying aggregation size.

5.3.2.5 Aggregate Size Comparison

Finally, we vary the maximum number of packets that can be included in an aggregate. Figure 5.10 plots the time-series for burst loss under (a) 8 packets in an aggregate and (b) 1 packet in an aggregate (this effectively disables aggregation). We can see that under 8 packets in an aggregate, unacceptable latencies are still encountered. However, when aggregation is disabled, the delay due to burst errors is a maximum of about 10 ms. This corresponds to 24 packets lost in a row. The maximum latency decreases because the frame size is smaller, and thus frames are sent more quickly. Therefore, there is a higher probability to successfully receive a packet in the midst of a long duration of burst errors. Disabling aggregated frames is not an acceptable solution, however, because the throughput suffers significantly. Figure 5.2 shows a 40-75% reduction in throughput when disabling aggregation.

5.3.2.6 Summary of Measurements

We find that supporting a high-throughput, low latency link encounters a number of challenges. The 802.11n standard provides high throughput, but can suffer from increased latencies under a mobile environment. In this section, we have analyzed a variety of client configurations and found burst errors to occur in each scenario. Latencies of 20-40 ms are often encountered on the wireless link.

5.4 Loss Analysis

In this section, we try to dig deeper to understand the burst losses in more detail. We make modifications to the `ath9k` driver in order to obtain additional loss information. Specifically, we enable counters in the driver that track status errors returned by the firmware. There are two counters that we found to be consistently updated during our tests: `ATH9K_RXERR_CRC` and `ATH9K_RXERR_PHY`. The`ATH9K_RXERR_CRC` error indicates that the packet was received, but some of the payload is corrupt (and thus the packet does not pass the cyclic redundancy check). The `ATH9K_RXERR_PHY` errors indicate that the packet was detected, but was unable to be received correctly. The `ATH9K_RXERR_PHY` errors can occur for a variety of reasons: OFDM frequency and timing synchronization errors and incorrect checksum in the signal field of the preamble that can be caused by false preamble detection, weak signal, interference, and/or incorrect AGC settings that can lead to ADC saturation.

We keep track of these counters during our tests. When losses occur, we

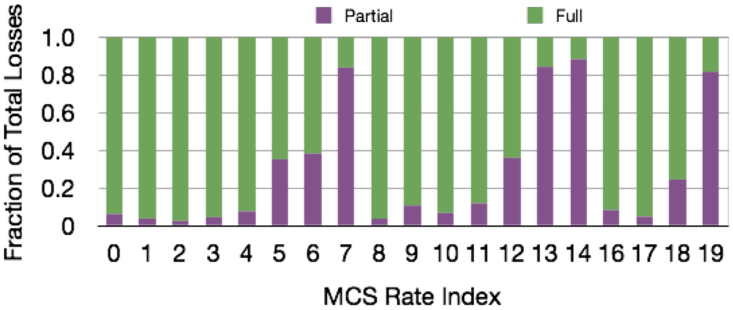


Figure 5.11: Fraction of losses that are a total aggregate loss or a partial packet loss.

can compare the number of losses we've observed to the counters in the driver. If the sum of the two counters is less than the number of packets, then we know that some packets were not detected at all. We use these three indicators (deemed `CRC`, `PHY` and `None`) to better understand what is happening under bursty losses.

5.4.1 Partial or Full Losses?

The first question we wanted to ask was what was happening when a string of packets were lost. Say for instance that we loss 32 packets in a row. Are these losses 32 packets with `CRC` errors inside of an aggregate, or is the whole aggregate lost? We define losses due to `CRC` errors as *partial* losses and losses due to the whole aggregate being lost as *full* losses. In Figure 5.11, we examine the fraction of packet losses that are a whole aggregate loss versus the fraction of packet losses that are partially lost. We see that for rates that use a high modulation type, the partial packet losses can dominate (MCS rates 5-7, 13-15 and 19 all use 64-QAM). The rates that do not use 64-QAM typically have much fewer CRC errors. For these rates, the burst losses are dominated by full aggregate losses; the full aggregate losses typically account for around 90% of the total losses.

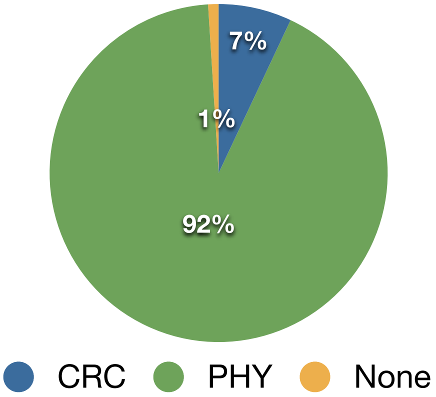


Figure 5.12: Analysis of the type of loss for MCS 11.

5.4.2 Full Loss Analysis

We are interested to see what is happening during full packet losses. Therefore, we zoom in on a specific rate, MCS 11, and analyze the type of losses experienced. In Figure 5.12, we show the fraction of total losses that CRC, PHY and None account for. There are several things to note from this graph. First, we see a correspondence between the partial losses in Figure 5.11 and fraction of CRC errors in this plot. Second, we notice that the fraction of CRC errors is relatively small compared to the fraction of PHY errors. This implies that the aggregated frames are not being received at the receiver. The last classification, None, account for only a small fraction of losses in this test.

Next, we show the same analysis for all of the single stream rates in 802.11n. The results are presented in Figure 5.13. The channel used in these experiments is 20 Mhz in order to compare against 802.11a, which we will analyze next. The results show that again, PHY errors dominate. We the modulation type increases, so

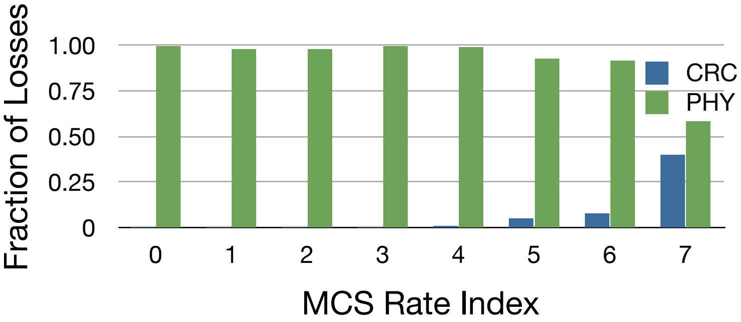


Figure 5.13: Analysis of losses for 802.11n.

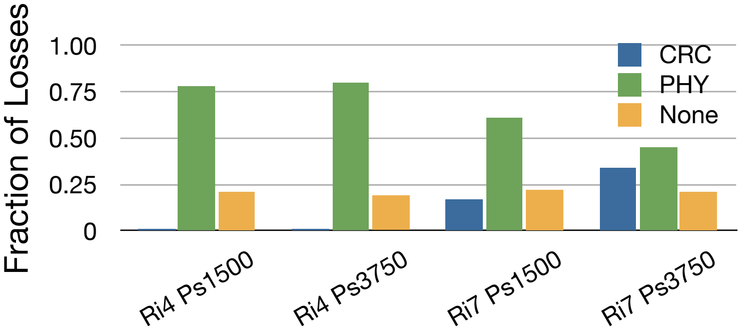


Figure 5.14: Analysis of losses for 802.11a.

does the fraction of CRC errors. Note that the fraction of None errors were quite small, typically less than 2%, so they were omitted from this graph for clarity.

The last set of full loss analysis is presented in Figure 5.14. Here, we plot the fraction of losses for each loss type under 802.11a. The x-axis looks at two different data rates: rate index 4 (24 Mbps) and rate index 7 (54 Mbps). For each rate index, we examine two different packet sizes, 1500 bytes and 3750 bytes. We denote rate index as “Ri” in the legend, and packet size as “Ps”. There are a few things to note here. First, we typically see a small fraction of CRC errors for rate index 4. As we increase the rate, we see more CRC errors. Furthermore, with the larger packet size, there are also more CRC errors. The interesting result here is that there is a much larger fraction of None errors. Typically about 20% of the errors are None errors. This is in contrast to the 802.11n results, which had very few of these errors. This may indicate that the added receiver diversity in 802.11n is making the receiver more robust towards packet detection.

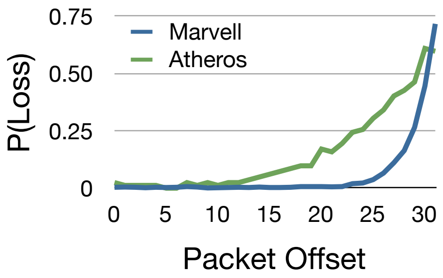


Figure 5.15: Probability density function of packets in an aggregate with CRC errors.

5.4.3 Partial Loss Analysis

We were also interested to see what the partial losses look like. Here, an aggregated frame is being received with some CRC errors. This implies that at least one packet inside the aggregate is corrupted. We examine all aggregate packets in the trace that suffer from CRC errors and then plot the probability of each packet offset within the aggregate being exposed to error. The results are presented in Figure 5.15. Here, we show the PDF for both Atheros and Marvell chipsets for MCS11. Other rates provided consistent results. We see that regardless of the chipset, the most packet losses are concentrated near the end of the aggregate. This is likely because the channel starts to become stale from the initial channel estimate during the preamble. We find these results to be consistent with the results presented in [54].

5.4.4 Summary

We find that packet lost in burst errors are typically due to full aggregated frames being lost. Furthermore, when the whole aggregated packet is lost, it is not because every packet within the aggregate is corrupted. Rather, the aggregate is not even detected or suffers from an error after detection.

5.5 Recommendations

Given the analysis in the previous section, in this section we present some recommendations for reducing loss. We use our findings in the previous section to recommend future research directions. First, we examine a technique to reduce CRC errors in packets. Next, we focus on the fact that a high number of burst losses are due to whole aggregates being lost. Therefore, we examine the potential of two techniques that aim to eliminate the possibility of a whole aggregate getting lost. Afterwards, we present a short case study to analyze the effectiveness of prior work to solve our problem.

5.5.1 Addition of a Midamble

Discussion: In Section 5.4.3, we found that CRC errors within an aggregated packet are typically distributed toward the end of the aggregate. More than likely, this trend is due to the fact that the channel conditions, which are derived during the preamble, can become stale at the end of the aggregate [54]. In order to avoid the channel conditions becoming stale, a mid-amble can be added to the middle of the aggregated packet. The mid-amble will allow the receiver to re-synchronize with the sender. This can help reduce the loss at the end of the packet to rates similarly seen at the beginning of the packet (so the loss at offset 31 will become the loss at offset 15 in Figure 5.15).

We analyze the possible benefits of adding a mid-amble via a simple, custom-built simulator. The simulator models a single wireless link, and uses the measurement data seeded from Figure 5.15 to impose loss on packets. That is, from the trace we get the probabilities of suffering a CRC error for each packet within an aggregate, and then use an uniform distribution to determine if a packet suffers a CRC error. In this analysis, we omit PHY errors. When the mid-amble technique is used, we reset the probability of suffering a CRC error: for packet x, the probability of losing the packet to a CRC error is determined by x's position in the aggregate if $x < 16$ and $x - 16$'s position in the aggregate if $x \geq 16$. We set the data rate to MCS 11, which allows for 32 packets in an aggregate.

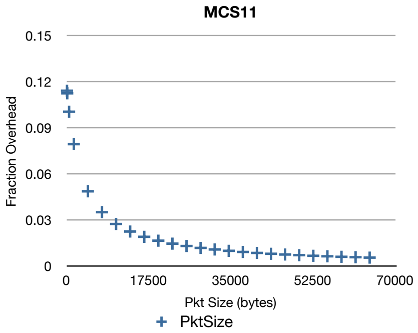


Figure 5.16: Overhead of the midamble scheme compared to the base.

Results: To analyze the scheme, we measure its overhead and the throughput benefit compared to a scheme without the midamble. The overhead is plotted in Figure 5.16. For MCS 11, we vary the packet size and plot the fraction of overhead

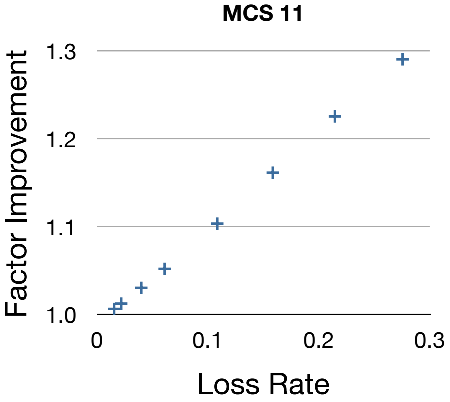


Figure 5.17: Throughput improvement of midamble scheme over the base.

the midamble scheme consumes. When the packet size is small, the overhead is still less than 12%. When the packet size is large (64 Kb is the maximum aggregate size), the overhead shrinks to less than 1%.

The throughput improvement of the midamble scheme is plotted in Figure 5.17. For varying overall loss rates, we plot the throughput improvement of the midamble scheme over the baseline approach (no midamble). As we can see, the improvement scales linearly with the loss. This is because most of the losses in the baseline scheme are concentrated near the end of the aggregate. The midamble technique reduces the losses at the end of the aggregate to be similar to the losses experienced during the first half of baseline aggregate. Since these losses are very small, the midamble scheme is effective in increasing the throughput by being more robust to loss.

5.5.2 Changing the Preamble

Discussion: In 802.11, the preamble consists of two parts: a short training sequence and a long training sequence. The short training sequence is used for signal detection, automatic gain control, diversity selection, timing synchronization and coarse frequency offset. From our previous results, we find that packets that are not detected are the potential cause of the burst losses.

In 802.11, only 12 sub-carriers are used to send the short training sequence [1]. The sub-carriers that are not used are filled with 0 values. The zero padding allows for the frequency offset to be calculated: more padding allows a larger frequency offset to be tolerated [149]. The data sent on the 12 sub-carriers consists of a PN sequence. Autocorrelation techniques, such as Schmidl and Cox [132], are typically used to detect the packet by correlating incoming symbols by the PN sequence used in the preamble. When the autocorrelation values spike, it triggers the packet detection algorithm.

Therefore, a simple approach is to decrease the amount of zero padding in the preamble. This reduces the amount of frequency offset a radio can tolerate. A radio that uses this approach would have to be calibrated, potentially increasing its cost. The benefit is that we can use more sub-carriers to include a longer PN sequence in the preamble. A larger PN sequence will give larger autocorrelation spikes, which should help detect packets.

Results: We provide a micro-benchmark to these scheme by implementing two different preambles in GNU Radio [46]. One preamble consists of a PN sequence

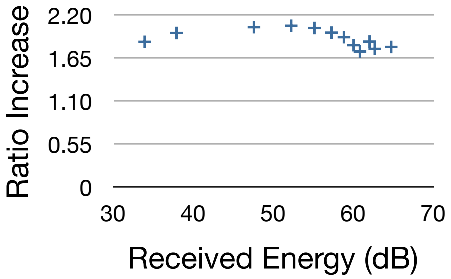


Figure 5.18: Improvement of the autocorrelation values for a training sequence with no zero-padding over one with zero-padding in every-other sub-carrier.

with every-other sub-carrier set to a 0. The other preamble consists of a PN sequence without any zeros. We manually tune the frequency offset of the radios and then run a transfer from the sender to the receiver. We vary the transmit power at the sender, and for each preamble scheme, we log the autocorrelation values that are used to detect the packet. Figure 5.18 shows the results. It shows the improvement over the magnitude of the spike for the PN sequence without any zeros over the magnitude of the spike for the PN sequence that is zero padded. The expected improvement would be a factor of two, since each sub-carrier that does not have a zero can increase the autocorrelation value. In these results, we see the improvement is close to the expected value.

We are also interested in how changing the PN sequence can help with packet detection. There are two metrics that are important: the number of false positives (how many times we detect a packet when one isn't there) and the number of false negatives (how many times we do not detect a packet when one is present). The number of false positives is plotted in Figure 5.19 and the number of false negatives is plotted in Figure 5.20. The plots are generated over a packet

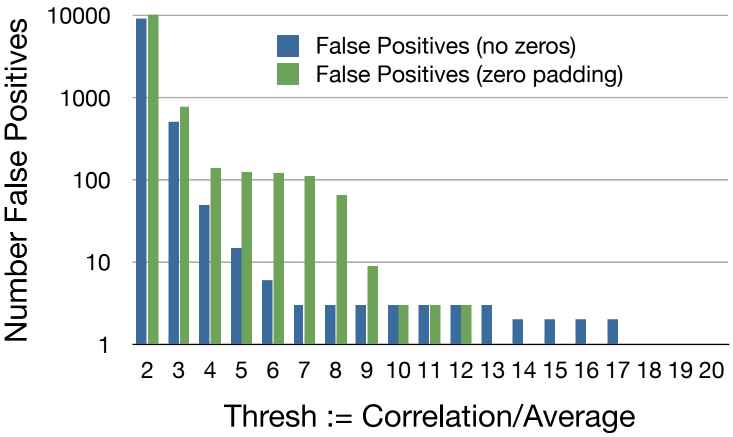


Figure 5.19: False Positives.

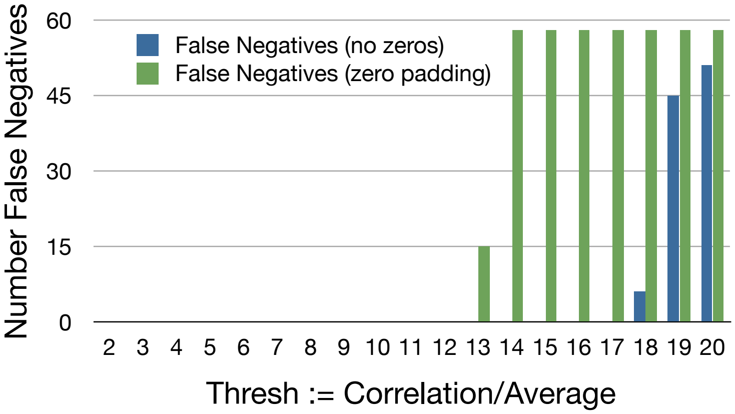


Figure 5.20: False Negatives.

trace containing 58 packets. The y-axis shows the number of packets in error (false positives or negatives) and the x-axis shows a threshold used in packet detection. Note that Figure 5.19 has a log scale. The default OFDM implementation in GNU Radio averages the incoming correlation values over every sample with an EWMA (each new sample is weighted $\alpha = 0.001$). The ratio of the correlation value to the average is taken, and when that ratio exceeds a threshold, it triggers a packet detection.

In the figures, we can see that for a given threshold, the preamble with no zero padding performs better in terms of minimizing false positives and false negatives. This allows more flexibility and robustness when designing packet detection schemes. This also has an impact on performance. False negatives have an obvious impact on performance: missed preambles equate to a lost packet. False positives can also have an impact on performance: false positives can hurt performance when the demodulator tries to lock onto a false preamble, potentially missing a true preamble in the process [131]. While these occurrences may be low, it is none-the-less important to minimize their possibility if minimizing loss is at a premium. Finally, it may look as though certain threshold values well (10-12 in the graphs). However, under low energy levels (the lowest two levels in Figure 5.18), this threshold is too large, and thus it results in 100% false negative detection (graph omitted for brevity).

5.5.3 Preamble ACK Scheme

Another scheme that has the potential to help mitigate full aggregate loss is to have an explicit acknowledgement after the preamble is sent. This scheme is effective because it works regardless of whether there is a PHY error or if the packet is simply not detected.

To examine the effectiveness of the approach, we implemented such a scheme in a simple simulator. We assume that there is separate and independent preamble loss and data loss. For a packet to be successfully received, it must first successfully receive the preamble, and then successfully receive the payload. If the preamble is

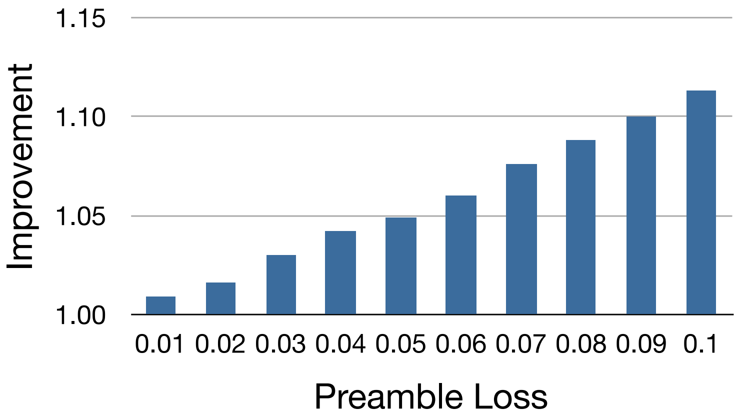


Figure 5.21: Improvement of preamble acknowledgement scheme over legacy approach.

not detected, or if the receiver indicates a PHY error, it will not send an acknowledgement for the preamble. Then, the sender will retransmit the preamble until the receiver successfully receives the preamble. After the preamble is successfully received, the data will be transmitted.

Our results indicate that the improvement of the preamble acknowledgement scheme is a function of the preamble loss rate. This is because the scheme that doesn't acknowledge the preamble will transmit the whole packet when the preamble is lost. In the preamble acknowledgement scheme, the node will retransmit the preamble. This takes a small portion of time compared to the data packet. Therefore, not much time is wasted when a preamble is lost. If the preamble acknowledgement scheme suffers from a data loss, it is no worse off than the normal scheme. Figure 5.21 show the throughput improvement when the data loss probability is fixed to 0.10 and the preamble loss rate is varied. The loss distribution in this graph is uniform, but we find the trends still hold for other loss distributions, as

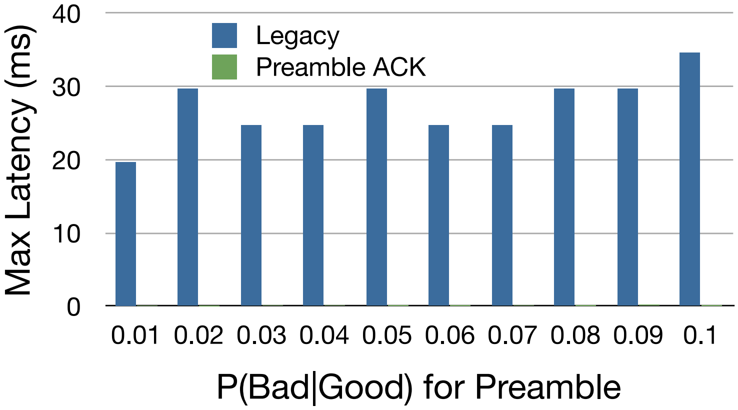


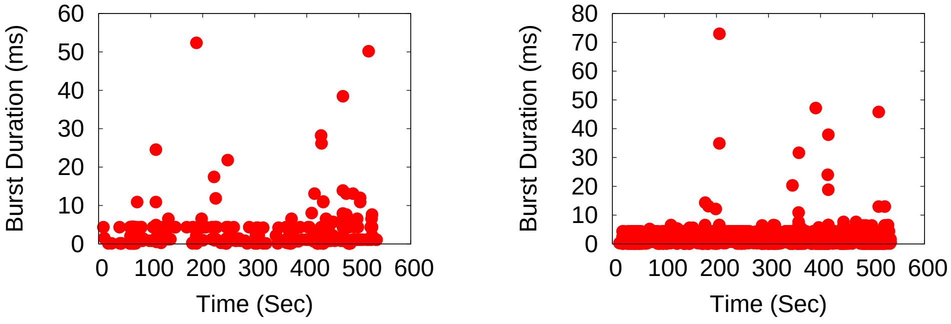
Figure 5.22: Maximum latency of preamble acknowledgement scheme and legacy approach.

well as trace-based Gilbert-Elliot model-based runs (see below).

Figure 5.22 shows the maximum latency encountered for each scheme during a five minute simulated run. In this graph, the sender uses MCS 11, 40 Mhz channel width, and Greenfield preambles. The data loss rate is fixed to 0% and we use the Gilbert-Elliot model to dictate the preamble loss rate. We seed the transition probabilities from the bad state (P(G|B) and P(B|B)) from our loss measurements and artificially vary the other transition probabilities (P(B|G) and P(G|G)). The plot shows the maximum latency experienced as a function of the probability of transitioning into a bad state from a good state (P(B|G)). The reason that the maximum latency is not monotonically increasing is because the results depend on the randomness of each run. While the latencies in the legacy case are on the order of 20-30 ms, the latencies of the Preamble Acknowledgement scheme are quite low. In fact, they are typically less than 1 ms. This is because the time to send a preamble and the time to acknowledge the preamble are very small. Since the data loss is set to zero, retransmitting the preamble multiple times will still result in latencies < 1 ms.

5.5.4 Multiple Radio Diversity

In this subsection, we implement the multi-radio diversity (MRD) scheme described in [98]. Briefly, MRD increases the robustness of a wireless link by employing multiple radios. The radios combine the packets received into one stream and export the unified stream to the higher layer. Therefore, in order for a packet to be successfully received by a node, it only has to be received by at least one of its radios.



(a) Radio 1 (b) Radio 2

Figure 5.23: Time-series loss for each radio in MRD case study.

In our tests, we employ two radios at the receiver. We then conduct the mobility experiments with MCS 11. The results are gathered over 10 minutes of mobility. We use the same trace to compare MRD versus the single radio case. That is, we log the burst errors on the primary radio and use these for the single radio results. We then post-process and combine the trace on the secondary radio

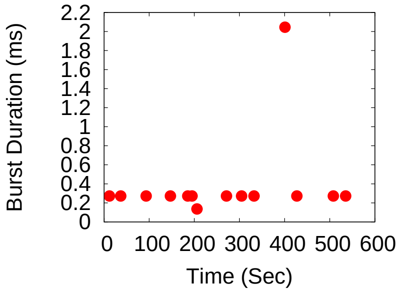


Figure 5.24: Time-series for MRD scheme.

to approximate MRD. The results are shown in Figure 5.23 (time-series for each radio) and in Figure 5.24 (results for MRD). Here, we can see the single radio results, as before, suffer from bursts of up to 50 ms. However, these bursts are greatly reduced in MRD. In fact, the largest burst size is only 2 ms. Over the whole 10 minute run, there were only 14 instances of loss. This indicates that MRD is an effective solution to providing low latency, high throughput wireless links. The reason why MRD is effective is because the loss on the two radios appears to be mostly independent. For instance, let A be the event a packet is lost on the first radio, and let B be the event that a packet is lost on the second radio. We measured P(A) = 0.00103 and P(A|B) = 0.00107. The results were similar for P(B) and P(B|A), helping confirm that each radio sees independent losses.

The main trade-off, of course, is that MRD requires multiple radios. Therefore, this approach may prove ineffectual for mobile or constrained clients because multiple radios require increased energy, size, cost and add additional complexity on the receiver. Future research directions should aim to reduce these overheads.

5.5.5 Summary

In this section, we present recommendations for reducing wireless loss. Techniques for future research directions are proposed based on the findings in our loss analysis. We provide micro-benchmarks that indicate adding a midamble, changing the preamble, or providing a preamble acknowledgement have the potential to reduce loss. Furthermore, we perform a case study that shows previous work on multi-radio diversity provides an effective solution to the problem if multiple radios are a practical design constraint.

5.6 Physical Layer Conclusion

In this section we presented a case study for physical layer losses in wireless networks. We were motivated by the potential for a high throughput, low latency wireless link, and conducted measurements and analysis to assess its viability. We found that mobility can impose latencies on the link that may not support real-time communications. Further analysis of the losses indicated that the burst errors were typically due to whole aggregated packets being lost. Therefore, we presented recommendations for mitigating this type of loss.

Chapter 6

Conclusion

Wireless networks are ubiquitous in today's world. A wide variety of networks are being deployed and the networks are being relied on for more and more applications. A fundamental problem that all wireless networks must face is loss. Loss can greatly impact the performance of a network and must be reduced or eliminated in order for wireless networks to reach their full potential.

In order to avoid degraded network performance, the goal of this dissertation is to combat loss at varying network layers in order to allow wireless technologies to realize their full potential. Attacking loss at each level of the network stack provides increased robustness: if a coping mechanism fails in one layer, the adjacent lower layer can be employed to mitigate the problem. For the purposes of this dissertation, loss is tackled in the Network Layer and below.

In the Network Layer, we develop an accurate model for IEEE 802.11 broadcast transmissions and design a model-driven optimization algorithm to jointly optimize opportunistic routes and rate limits. Our evaluations show the model is highly accurate: the predicted performance is within 80% of the achieved performance most of the time. Furthermore, we evaluate the effectiveness of our approach through simulations and a testbed implementation. We find that our protocol performs significantly better than state-of-the-art shortest path routing and opportunistic routing protocols (*e.g.*, its total throughput is 2-13 times ETX's throughput and 1.5-10 times MORE's throughput). Furthermore, we evaluate its performance in dynamic and uncontrolled environments, and find it is robust against inaccuracy introduced by a dynamic network and consistently out-performs the existing schemes.

In the Link Layer, an Efficient Retransmission (ER) scheme is proposed to reduce the number of retransmissions by coding together multiple retransmissions in a single packet. The design a protocol and implement ER in real IEEE 802.11 networks in order to evaluate its performance. We find the approach can effectively reduce the number of retransmissions in the network and also lead to throughput gains.

Finally, a case study is discussed in the Physical Layer to investigate loss patterns of packets in dynamic environments. We impose strict performance requirements on the wireless link, demanding high throughput and low latency. We first find when these requirements can be met and then investigate problems that occur under moderate mobility and high signal strength. We measure and analyze losses under a variety of configurations, and find that burst losses can cause high delays regardless of the configuration. We debug the burst losses to gain further insights into their possible causes and find that most burst losses are due to consecutive aggregated packet being lost. Based on these findings, we make recommendations to help reduce losses. Future research directions can focus on making the preamble more robust to mitigate loss, and we conclude by briefly examining the potential of several promising techniques: adding a preamble acknowledgement to avoid losing a whole aggregated packet, changing the preamble structure to make packet detection more robust, adding a midamble to help reduce CRC errors and finally utilizing multiple radios to increase receiver robustness.

Bibliography

[1] IEEE Standard for Wireless LAN medium access control (MAC) and physical layer (PHY) specifications. 1999.

[2] IEEE Standard for Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications Amendment 5: Enhancements for Higher Throughput. 2009.

[3] 802.11b community network list. http://www.toaster.net/wireless/community.html.

[4] A. Adya, P. Bahl, R. Chandra, and L. Qiu. Architecture and techniques for diagnosing faults in IEEE 802.11 infrastructure networks. In *Proc. of ACM MobiCom*, September 2004.

[5] M. Afanasyev and A. C. Snoeren. The importance of being overheard: throughput gains in wireless mesh networks. In *Proceedings of the 9th ACM SIGCOMM Conference on Internet Measurement*, IMC, pages 384–396, New York, NY, USA, 2009. ACM.

[6] S. Agarwal, J. Padhye, V. N. Padmanabhan, L. Qiu, A. Rao, and B. Zill. Estimation of link interference in static multi-hop wireless networks. In *Proc. of Internet Measurement Conference (IMC)*, 2005.

[7] D. Aguayo, J. Bicket, S. Biswas, G. Judd, and R. Morris. Link-level measurements from an 802.11b mesh network. In *Proc. of ACM SIGCOMM*, Aug. 2004.

[8] R. Ahlswede, N. Cai, S. R. Li, and R. W. Yeung. Network information flow. *IEEE Transactions on Information Theory*, Jul. 2000.

[9] N. Ahmed, U. Ismail, S. Keshav, and K. Papagiannaki. Online estimation of RF interference. In *Proc. of ACM CoNEXT*, Dec. 2008.

[10] N. Ahmed and S. Keshav. Smarta: A self-managing architecture for thin access points. In *Proc. of ACM CoNEXT*, Dec. 2006.

[11] S. Alamouti. A simple transmit diversity technique for wireless communications. *IEEE Journal on Selected Areas in Communications*, 16(8):1451–1458, 1998.

[12] J. Andrews. Interference cancellation for cellular systems: a contemporary overview. In *IEEE Wireless Communications Magazine*, volume 12(2), April 2005.

[13] B. Awerbuch, D. Holmer, and H. Rubens. Provably secure competitive routing against proactive Byzantine adversaries via reinforcement learning. In *JHU Tech Report Version 1*, May 2003.

[14] P. Bahl, R. Chandra, J. Padhye, L. Ravindranath, M. Singh, A. Wolman, and B. Zill. Enhancing the security of corporate wi-fi networks using DAIR.

In *Proceedings of the 4th international conference on Mobile systems, applications and services*, MobiSys, pages 1–14, New York, NY, USA, 2006. ACM.

[15] A. Bakre and B. Badrinath. I-TCP: indirect TCP for mobile hosts. In *Proceedings of the 15th International Conference on Distributed Computing Systems*, pages 136 –143, May 1995.

[16] H. Balakrishnan, S. Seshan, E. Amir, and R. H. Katz. Improving TCP/IP performance over wireless networks. In *Proceedings of the 1st annual international conference on Mobile computing and networking*, MobiCom, pages 2–11, New York, NY, USA, 1995. ACM.

[17] G. Berger-Sabbatel, F. Rousseau, M. Heusse, and A. Duda. Performance anomaly of 802.11b. In *Proc. of IEEE INFOCOM*, June 2003.

[18] G. Bianchi. Performance analysis of the IEEE 802.11 distributed coordination function. *Proc. of IEEE Journal on Selected Areas in Communications*, March 2000.

[19] Y. Birk and D. Crupnicoff. A multicast transmission schedule for scalable multirate distribution of bulk data using nonscalable erasure correcting codes. In *Proc. of IEEE INFOCOM*, Apr. 2003.

[20] S. Biswas and R. Morris. ExOR: opportunistic multi-hop routing for wireless networks. In *Proc. of ACM SIGCOMM*, Aug. 2005.

[21] J. Camp and E. Knightly. Modulation rate adaptation in urban and vehicular environments: cross-layer implementation and experimental evaluation. In *Proceedings of the 14th ACM international conference on Mobile computing and networking*, MobiCom, pages 315–326, New York, NY, USA, 2008. ACM.

[22] P. Castoldi. *Multiuser Detection in CDMA Mobile Terminals.* Artech House, Inc., Norwood, MA, USA, 2002.

[23] A. Cerpa, J. L. Wong, M. Potkonjak, and D. Estrin. Temporal properties of low power wireless links: modeling and implications on multi-hop routing. In *Proceedings of the 6th ACM international symposium on Mobile ad hoc networking and computing*, MobiHoc, pages 414–425, New York, NY, USA, 2005. ACM.

[24] S. Chachulski, M. Jennings, S. Katti, and D. Katabi. Trading structure for randomness in wireless opportunistic routing. In *Proc. of ACM SIGCOMM*, Aug. 2007.

[25] R. Chandra, V. N. Padmanabhan, and M. Zhang. WiFiProfiler: cooperative diagnosis in wireless LANs. In *Proceedings of the 4th international conference on Mobile systems, applications and services*, MobiSys, pages 205–219, New York, NY, USA, 2006. ACM.

[26] C. Chen, D. Aksoy, and T. Demir. Processed data collection using opportunistic routing in location aware wireless sensor networks. In *Proc. of International Conference on Mobile Data Management*, May 2006.

[27] Y.-C. Cheng, M. Afanasyev, P. Verkaik, P. Benkö, J. Chiang, A. C. Snoeren, S. Savage, and G. M. Voelker. Automating cross-layer diagnosis of enterprise wireless networks. In *Proceedings of the 2007 conference on Applications, technologies, architectures, and protocols for computer communications*, SIGCOMM, pages 25–36, New York, NY, USA, 2007. ACM.

[28] Y.-C. Cheng, J. Bellardo, P. Benkö, A. C. Snoeren, G. M. Voelker, and S. Savage. Jigsaw: solving the puzzle of enterprise 802.11 analysis. In *Proceedings of the 2006 conference on Applications, technologies, architectures, and protocols for computer communications*, SIGCOMM, pages 39–50, New York, NY, USA, 2006. ACM.

[29] R. Choudhury and N. Vaidya. MAC-layer anycasting in wireless ad hoc networks. In *Proc. of Hot Topics in Networks (HotNets)*, Nov. 2003.

[30] Click modular router. http://pdos.csail.mit.edu/click/.

[31] COPE source code. `http://piper.csail.mit.edu/dokuwiki/doku.php?id=cope`.

[32] D. D. Couto, D. Aguayo, J. Bicket, and R. Morris. A high-throughput path metric for multi-hop wireless routing. In *Proc. of ACM MobiCom*, 2003.

[33] R. Daniels, C. Caramanis, and R. Heath. Adaptation in convolutionally coded MIMO-OFDM wireless systems through supervised learning and SNR ordering. *Vehicular Technology, IEEE Transactions on*, 59(1):114 –126, 2010.

[34] P. Ding, J. Holliday, and A. Celik. MAC layer multicast protocol to increase reliability in WLANs. In *IEEE International Conference on Dependable Systems and Networks*, DSN, June 2004.

[35] R. Draves, J. Padhye, and B. Zill. Routing in multi-radio, multi-hop wireless mesh networks. In *Proc. of ACM MobiCom*, Sept. - Oct. 2004.

[36] R. Dube, C. Rais, K.-E. Wang, and S. Tripathi. Signal stability based adaptive routing (SSA) for ad-hoc mobile networks. In *IEEE Personal Comm*, Feb. 1997.

[37] H. Dubois-Ferrier, M. Grossglauser, and M. Vetterli. Least-cost opportunistic routing. In *Proc. of Allerton Conference on Communication, Control, and Computing*, Sept. 2007.

[38] A. Dutta, D. Saha, D. Grunwald, and D. Sicker. SMACK: a smart acknowledgment scheme for broadcast messages in wireless networks. In *Proceedings of the ACM SIGCOMM 2009 conference on Data communication*, SIGCOMM, pages 15–26, New York, NY, USA, 2009. ACM.

[39] D. Eckhardt and P. Steenkiste. Measurement and analysis of the error characteristics of an in-building wireless network. In *Conference proceedings on Applications, technologies, architectures, and protocols for computer communications*, SIGCOMM, pages 243–254, New York, NY, USA, 1996. ACM.

[40] H. Feng, Y. Shu, S. Wang, and M. Ma. SVM-based models for predicting WLAN traffic. In *Proc. of IEEE ICC*, 2006.

[41] Y. Gao, J. Lui, and D. M. Chiu. Determining the end-to-end throughput capacity in multi-hop networks: Methodolgy and applications. In *Proc. of ACM SIGMETRICS*, June 2006.

[42] M. Garetto, T. Salonidis, and E. Knightly. Modeling per-flow throughput and capturing starvation in CSMA multi-hop wireless networks. In *Proc. of IEEE INFOCOM*, March 2006.

[43] M. Garetto, J. Shi, and E. Knightly. Modeling media access in embedded two-flow topologies of multi-hop wireless networks. In *Proc. of ACM MobiCom*, Aug. - Sept. 2005.

[44] M. Gastpar and M. Vetterli. On the capacity of wireless networks: the relay case. In *Proc. of IEEE INFOCOM*, June 2002.

[45] E. N. Gilbert. Capacity of a burst-noise channel. *Bell System Technical Journal*, 39:1253–1265, 1960.

[46] Gnu radio. `http://gnuradio.org/`.

[47] T. Goff, N. Abu-Aahazaleh, D. Phatak, and R. Kahvecioglu. Preemptive routing in ad hoc networks. In *Proc. of ACM MobiCom*, July 2001.

[48] S. Gollakota and D. Katabi. Zigzag decoding: combating hidden terminals in wireless networks. In *Proceedings of the ACM SIGCOMM 2008 confer-*

ence on Data communication, SIGCOMM, pages 159–170, New York, NY, USA, 2008. ACM.

[49] J. Gross, A. Willig, and T. Berlin. Measurements of a wireless link in different RF-isolated environments. Technical report, In Proc. of European Wireless, 2002.

[50] M. Grossglauser and D. N. C. Tse. Mobility increases the capacity of ad hoc wireless networks. In *Proc. of IEEE INFOCOM*, April 2001.

[51] P. Gupta and P. R. Kumar. The capacity of wireless networks. *IEEE Transactions on Information Theory*, 46(2), March 2000.

[52] D. Halperin, T. Anderson, and D. Wetherall. Taking the sting out of carrier sense: interference cancellation for wireless LANs. In *Proceedings of the 14th ACM international conference on Mobile computing and networking*, MobiCom, pages 339–350, New York, NY, USA, 2008. ACM.

[53] D. Halperin, W. Hu, A. Sheth, and D. Wetherall. 802.11 with multiple antennas for dummies. *SIGCOMM Computer Communication Review*, 40:19–25, January 2010.

[54] B. Han, L. Ji, S. Lee, B. Bhattacharjee, and R. R. Miller. All bits are not equal - a study of IEEE 802.11 communication bit errors. In *INFOCOM*, pages 1602–1610, 2009.

[55] B. Han, A. Schulman, F. Gringoli, N. Spring, B. Bhattacharjee, L. Nava, L. Ji, S. Lee, and R. Miller. Maranello: practical partial packet recovery for

802.11. In *Proceedings of the 7th USENIX conference on Networked systems design and implementation*, NSDI, pages 14–14, Berkeley, CA, USA, 2010. USENIX Association.

[56] G. Holland and N. Vaidya. Analysis of TCP performance over mobile ad hoc networks. In *Proc. of ACM MobiCom*, 1999.

[57] J. Hou, J. Smee, H. Pfister, and S. Tomasin. Implementing interference cancellation to increase the EV-DO Rev A reverse link capacity. In *IEEE Wireless Communications Magazine*, volume 44(2), Feb 2006.

[58] Y. C. Hu and D. B. Johnson. Design and demonstration of live audio and video over multi-hop wireless networks. In *Proc. of MILCOM*, Oct. 2002.

[59] L. Huang, A. Arora, and T. Lai. Reliable MAC layer multicast in IEEE 802.11 wireless networks. In *Proc. of ICPP*, Aug. 2002.

[60] iperf. `http://sourceforge.net/projects/iperf/`.

[61] A. Iyer, G. Deshpande, E. Rozner, A. Bhartia, and L. Qiu. Fast resilient jumbo frames in wireless LANs. In *Proc. of IWQoS*, June 2009.

[62] K. Jain, J. Padhye, V. N. Padmanabhan, and L. Qiu. Impact of interference on multi-hop wireless network performance. In *Proc. ACM MobiCom*, 2003.

[63] W. Jakes. A comparison of specific space diversity techniques for the reduction of fast fading in uhf mobile radio systems. *IEEE Transactions on Vehicular Technology*, Nov. 1971.

[64] K. Jamieson and H. Balakrishnan. PPR: partial packet recovery for wireless networks. In *Proceedings of the 2007 conference on Applications, technologies, architectures, and protocols for computer communications*, SIGCOMM, pages 409–420, New York, NY, USA, 2007. ACM.

[65] C. Jiao, L. Schwiebert, and B. Xu. On modeling the packet error statistics in bursty channels. In *Proceedings of the 27th Annual IEEE Conference on Local Computer Networks*, LCN, pages 0534–, Washington, DC, USA, 2002. IEEE Computer Society.

[66] D. B. Johnson, D. A. Maltz, and J. Broch. DSR: The dynamic source routing protocol for multihop wireless ad hoc networks. In *Ad Hoc Networking*, 2001.

[67] M. Jolfaei, S. Martin, and J. Mattfeldt. A new efficient selective repeat protocol for point-to-multipoint communication. In *IEEE International Conference on Communications. ICC. Geneva.*, volume 2, pages 1113 –1117 vol.2, May 1993.

[68] G. Judd, X. Wang, and P. Steenkiste. Efficient channel-aware rate adaptation in dynamic environments. In *Proceeding of the 6th international conference on Mobile systems, applications, and services*, MobiSys, pages 118–131, New York, NY, USA, 2008. ACM.

[69] V. Kann. Minimum clique partition. `http://www.csc.kth.se/~viggo/wwwcompendium/node25.html`.

[70] R. Karrer, A. Sabharwal, and E. Knightly. Enabling large-scale wireless broadband: The case for TAPs. In *Proc. of HotNets*, Nov. 2003.

[71] A. Kashyap, S. Das, and S. Ganguly. A measurement-based approach to modeling link capacity in 802.11-based wireless networks. In *Proc. of ACM MobiCom*, Sept. 2007.

[72] S. Katti, D. Katabi, H. Balakrishnan, and M. Medard. Symbol-level network coding for wireless mesh networks. In *Proceedings of the ACM SIGCOMM 2008 conference on Data communication*, SIGCOMM, pages 401–412, New York, NY, USA, 2008. ACM.

[73] S. Katti, H. Rahul, W. Hu, D. Katabi, M. Medard, and J. Crowcroft. XORs in the air: Practical wireless network coding. In *Proc. of ACM SIGCOMM*, Sept. 2006.

[74] F. Kelly, A. Maulloo, and D. Tan. Rate control in communication networks: shadow prices, proportional fairness and stability. In *Journal of the Operational Research Society*, volume 49, 1998.

[75] R. Koetter and M. Medard. An algebraic approach to network coding. *IEEE/ACM Transactions on Networking*, pages 782–795, Oct. 2003.

[76] D. Koutsonikolas, C.-C. Wang, and Y. C. Hu. CCACK: efficient network coding based opportunistic routing through cumulative coded acknowledgments. In *Proc. of IEEE INFOCOM*, 2010.

[77] A. Kumar, E. Altman, D. Miorandi, and M. Goyal. New insights from a fixed point analysis of single cell IEEE 802 .11 wireless LANs. In *Proc. of IEEE INFOCOM*, March 2005.

[78] J. Kuri and S. Kasera. Reliable multicast in multi-access wireless LANs. *Wireless Networks*, pages 359–369, Apr. 2001.

[79] A. Kpke, A. Willig, and H. Karl. Chaotic maps as parsimonious bit error models of wireless channels. In *In Proceedings of Infocom*, pages 513–523, 2003.

[80] J. Lacan and T. Perennou. Evaluation of error control mechanisms for 802.11 b multicast transmissions. In *Proc. of WiNMee*, Apr. 2006.

[81] P. Larsson and N. Johansson. Multi-user ARQ. In *IEEE 63rd Vehicular Technology Conference*, volume 4, pages 2052 –2057, May 2006.

[82] P. Lee, V. Misra, and D. Rubenstein. On the robustness of wireless opportunistic routing toward inaccurate link-level measurements. In *Proc. of COMSNETS*, pages 1 –10, Jan. 2010.

[83] J. Li, C. Blake, D. S. J. D. Couto, H. I. Lee, and R. Morris. Capacity of ad hoc wireless networks. In *Proc. of MobiCom*, July 2001.

[84] L. Li, R. Ramjee, M. Buddhikot, and S. Miller. Network-coding based broadcast in mobile ad-hoc networks. In *Proc. of IEEE INFOCOM*, Apr. 2007.

[85] Y. Li, L. Qiu, Y. Zhang, R. Mahajan, and E. Rozner. Predictable performance optimization for wireless networks. In *Proc. of ACM SIGCOMM*, Aug. 2008.

[86] Z. Li, B. Li, and L. C. Lau. On achieving maximum multicast throughput in undirected networks. *IEEE Transactions on Information Theory & IEEE/ACM Transactions on Networking (Special Issue on Networking and Information)*, June 2006.

[87] Y. Liang, J. Apostolopoulos, and B. Girod. Analysis of packet loss for compressed video: does burst-length matter? In *IEEE International Conference on Acoustics, Speech, and Signal Processing. (ICASSP)*, volume 5, pages V – 684–7 vol.5, 2003.

[88] Y. J. Liang and B. Girod. Network-adaptive low-latency video communication over best-effort networks. *IEEE Transactions on Circuits and Systems for Video Technology*, 16:72–81, 2006.

[89] K. C.-J. Lin, N. Kushman, and D. Katabi. ZipTx: Harnessing partial packets in 802.11 networks. In *Proceedings of the 14th ACM international conference on Mobile computing and networking*, MobiCom, pages 351–362, New York, NY, USA, 2008. ACM.

[90] Y. Lin, B. Li, and B. Liang. CodeOR: opportunistic routing in wireless mesh networks with segmented network coding. *IEEE International Conference on Network Protocols (ICNP).*, pages 13–22, Oct. 2008.

[91] Y. Lin, B. Liang, and B. Li. SlideOR: online opportunistic network coding in wireless mesh networks. In *Proc. of IEEE INFOCOM*, March 2010.

[92] M.-H. Lu, P. Steenkiste, and T. Chen. Design, implementation, and evaluation of an efficient opportunistic retransmission protocol. In *Proc. of ACM MobiCom*, 2009.

[93] D. Lun, M. Medard, and R. Koetter. Network coding for efficient wireless unicast. *International Zurich Seminar on Communications*, 2006.

[94] Madwifi. `http://madwifi.org`.

[95] R. Mahajan, M. Rodrig, D. Wetherall, and J. Zahorjan. Analyzing the MAC-level behavior of wireless networks. In *Proc. of SIGCOMM*, 2006.

[96] P. McKinley, C. Tang, and A. Mani. A study of adaptive forward error correction for wireless collaborative computing. *IEEE Transactions on Parallel and Distributed Systems*, pages 936–947, Sept. 2002.

[97] Meraki networks. `http://www.meraki.com`.

[98] A. K. Miu, H. Balakrishnan, and C. E. Koksal. Improving loss resilience with multi-radio diversity in wireless networks. In *Proc. of ACM MobiCom*, 2005.

[99] A. K. Miu, G. Tan, H. Balakrishnan, and J. Apostolopoulos. Divert: Fine-grained path selection for wireless LANs. In *Proc. of ACM MobiSys*, 2004.

[100] Monsoon HAVA wireless HD streaming device. `http://www.myhava.com/press_releases.html`.

[101] J. Moon and Y. Kim. Antenna diversity strengthens wireless LANs. *Communication Systems Design*, Jan. 2003.

[102] MORE source code. `http://people.csail.mit.edu/szym/more/README.html`.

[103] R. T. Morris, J. C. Bicket, and J. C. Bicket. Bit-rate selection in wireless networks. Technical report, Master's thesis, MIT, 2005.

[104] E. M. Nahum, C. C. Rosu, S. Seshan, and J. Almeida. The effects of wide-area conditions on WWW server performance. In *Proc. of ACM SIGMETRICS*, June 2001.

[105] D. Niculescu. Interference map for 802.11 networks. In *Proc. of Internet Measurement Conference (IMC)*, Nov. 2007.

[106] D. Niculescu, S. Ganguly, K. Kim, and R. Izmailov. Performance of VoIP in a 802.11 wireless mesh network. In *In Proc. IEEE INFOCOM*, 2006.

[107] Nintendo Wii U. `http://e3.nintendo.com`.

[108] V. N. Padmanbhan, L. Qiu, and H. Wang. Server-based inference of Internet performance. In *In Proc. of IEEE INFOCOM*, March 2003.

[109] R. Patra, S. Nedevschi, S. Surana, A. Sheth, L. Subramanian, and E. Brewer. WiLDNet: design and implementation of high performance WiFi based long distance networks. In *Proc. of NSDI*, Apr. 2007.

[110] V. Paxson and M. Allman. Computing TCP's retransmission timer. *IETF Internet DRAFT*, 2000. http://www3.ietf.org/proceedings/00jul/I-D/paxson-tcp-rto-01.txt.

[111] I. Pefkianakis, Y. Hu, S. H. Wong, H. Yang, and S. Lu. MIMO rate adaptation in 802.11n wireless networks. In *Proceedings of the sixteenth annual international conference on Mobile computing and networking*, MobiCom, pages 257–268, New York, NY, USA, 2010. ACM.

[112] C. E. Perkins and E. M. Royer. Ad hoc on-demand distance vector routing. In *Proc. of the Workshop on Mobile Computing Systems and Applications*, Feb. 1999.

[113] L. Qiu, Y. Zhang, F. Wang, M. K. Han, and R. Mahajan. A general model of wireless interference. In *Proc. of ACM MobiCom*, Sept. 2007.

[114] The Qualnet simulator from Scalable Networks Inc. `http://www.scalable-networks.com/`.

[115] B. Radunovic, C. Gkantsidis, P. Key, and P. Rodriguez. An optimization framework for opportunistic multipath routing in wireless mesh networks. *The 27th Conference on Computer Communications. IEEE INFOCOM*, pages 2252–2260, April 2008.

[116] H. Rahul, F. Edalat, D. Katabi, and C. G. Sodini. Frequency-aware rate adaptation and MAC protocols. In *Proceedings of the 15th annual international conference on Mobile computing and networking*, MobiCom, pages 193–204, New York, NY, USA, 2009. ACM.

[117] B. Raman and K. Chebrolu. Experiences in using Wi-Fi for rural internet in India. In *IEEE Wireless Communications Magazine*, volume 45(1), Jan. 2007.

[118] B. Raman, K. Chebrolu, D. Gokhale, and S. Sen. On the feasibility of the link abstraction in wireless mesh networks. In *Proc. of IEEE INFOCOM*, 2008.

[119] B. Randunovi and J. Y. L. Boudec. Rate performance objectives of multihop wireless networks. In *Proc. of IEEE INFOCOM*, Apr. 2004.

[120] C. Reis, R. Mahajan, M. Rodrig, D. Wetherall, and J. Zahorjan. Measurement-based models of delivery and interference. In *Proc. of ACM SIGCOMM*, 2006.

[121] L. Rizzo. Effective erasure codes for reliable computer communication protocols. *ACM SIGCOMM Computer Communication Review*, pages 24–36, Apr. 1997.

[122] L. Rizzo and L. Vicisano. A reliable multicast data distribution protocol based on software FEC techniques. In *Proc. of HPCS*, June 1997.

[123] L. Rizzo and L. Vicisano. RMDP: an FEC-based reliable multicast protocol for wireless environments. *ACM SIGMOBILE Mobile Computing and Communications Review*, pages 23–31, Apr. 1998.

[124] M. Rodrig, C. Reis, R. Mahajan, D. Wetherall, and J. Zahorjan. Measurement-based characterization of 802.11 in a hotspot setting. In *Proc. of ACM SIGCOMM Workshop on Experimental Approaches to Wireless Network Design and Analysis (E-WIND)*, Aug. 2005.

[125] Roku streaming player. `www.roku.com/`.

[126] MIT Roofnet. http://www.pdos.lcs.mit.edu/roofnet/.

[127] E. Rozner, M. K. Han, L. Qiu, and Y. Zhang. Model-driven optimization of opportunistic routing. In *Proceedings of the Joint International Conference on Measurement and Modeling of Computer Systems*, SIGMETRICS, pages 269–280, New York, NY, USA, 2011. ACM.

[128] E. Rozner, A. P. Iyer, Y. Mehta, L. Qiu, and M. Jafry. ER: efficient retransmission scheme for wireless LANs. In *Proceedings of the 2007 ACM CoNEXT conference*, CoNEXT, pages 8:1–8:12, New York, NY, USA, 2007. ACM.

[129] E. Rozner, J. Seshadri, Y. A. Mehta, and L. Qiu. SOAR: Simple opportunistic adaptive routing protocol for wireless mesh networks. *IEEE Transactions on Mobile Computing*, 8:1622–1635, 2009.

[130] T. Salonidis, M. Garetto, A. Saha, and E. Knightly. Identifying high throughput paths in 802.11 mesh networks: a model-based approach. In *Proc. of IEEE ICNP*, Oct. 2007.

[131] L. Scalia, J. Widmer, and I. Aad. On the side effects of packet detection sensitivity in IEEE 802.11 interference management. In *World of Wireless Mobile and Multimedia Networks (WoWMoM), 2010 IEEE International Symposium on a*, pages 1 –7, june 2010.

[132] T. Schmidl and D. Cox. Robust frequency and timing synchronization for ofdm. *Communications, IEEE Transactions on*, 45(12):1613 –1621, Dec 1997.

[133] Seattle wireless. `http://www.seattlewireless.net`.

[134] S. Sengupta, S. Rayanchu, and S. Banerjee. An analysis of wireless network coding for unicast sessions: The case for coding-aware routing. In *Proc. of IEEE INFOCOM*, Apr. 2007.

[135] F. Soldo, A. Markopoulou, and A. Toledo. A simple optimization model for wireless opportunistic routing with intra-session network coding. In *Proc. of IEEE NetCod*, June 2010.

[136] K. Srinivasan, M. Jain, J. I. Choi, T. Azim, E. S. Kim, P. Levis, and B. Krishnamachari. The kappa factor: Inferring protocol performance using interlink reception correlation. In *Proc. of ACM MobiCom*, Sept. 2010.

[137] K. Srinivasan, M. A. Kazandjieva, S. Agarwal, and P. Levis. The beta-factor: measuring wireless link burstiness. In *Proceedings of the 6th ACM conference on Embedded network sensor systems*, SenSys, pages 29–42, New York, NY, USA, 2008. ACM.

[138] A. P. Subramanian, M. M. Buddhikot, and S. Miller. Interference aware routing in multi-radio wireless mesh networks. In *IEEE Workshop on Wireless Mesh Networks (WiMesh)*, Sept. 2006.

[139] L. Subramanian, I. Stoica, H. Balakrishnan, and R. Katz. OverQos: an overlay based architecture for enhancing internet QoS. In *Proc. of NSDI*, Mar. 2004.

[140] G. J. Sullivan, P. Topiwala, and A. Luthra. The h.264/avc advanced video coding standard: Overview and introduction to the fidelity range extensions. In *SPIE conference on Applications of Digital Image Processing XXVII*, pages 454–474, 2004.

[141] J. J. T. Ho and H. Viswanathan. On network coding and routing in dynamic wireless multicast networks. In *Proc. of Workshop on Information Theory and its Applications*, 2006.

[142] C. Tang and P. K. McKinley. Modeling multicast packet losses in wireless lans. In *Proceedings of the 6th ACM international workshop on Modeling analysis and simulation of wireless and mobile systems*, MSWIM, pages 130–133, New York, NY, USA, 2003. ACM.

[143] K. Tang and M. Gerla. Random access MAC for efficient broadcast support in ad hoc networks. In *Proc. of Wireless Communications and Networking Conference (WCNC)*, Sept. 2000.

[144] K. Tang and M. Gerla. MAC reliable broadcast in ad hoc networks. In *Proc. of IEEE MILCOM*, Oct. 2001.

[145] V. Tarokh, H. Jafarkhani, and A. R. Calderbank. Space-time block codes from orthogonal designs. *IEEE Trans. Inform. Theory*, 45:1456–1467, 1999.

[146] V. Tarokh, N. Seshadri, S. Member, and A. R. Calderbank. Space-time codes for high data rate wireless communication: Performance criterion and code construction. *IEEE Trans. Inform. Theory*, 44:744–765, 1998.

[147] Technology for all project. `http://tfa.rice.edu/`.

[148] D. N. C. Tse, P. Viswanath, and L. Zheng. Diversity-multiplexing tradeoff in multiple-access channels. *IEEE Trans. Inform. Theory*, 50:1859–1874, 2004.

[149] R. Van Nee. A new OFDM standard for high rate wireless lan in the 5 GHz band. In *IEEE 50th Vehicular Technology Conference. VTC - Fall.*, volume 1, pages 258 –262 vol.1, 1999.

[150] A. J. Viterbi. Very low rate convolutional codes for maximum theoretical performance of spread-spectrum multiple-access channels. *IEEE Journal on Selected Areas in Communications*, 8:641–649, May 1990.

[151] M. Vutukuru, H. Balakrishnan, and K. Jamieson. Cross-layer wireless bit rate adaptation. In *Proceedings of the ACM SIGCOMM 2009 conference on Data communication*, SIGCOMM, pages 3–14, New York, NY, USA, 2009. ACM.

[152] Western Digital WD TV Live Plus 1080p HD Media Player. `http://wdc.com/en/products/products.aspx?id=320/`.

[153] C. Westphal. Opportunistic routing in dynamic ad hoc networks: The OPRAH protocol. In *Proc. of IEEE MASS*, Oct. 2006.

[154] A. Willig, M. Kubisch, C. Hoene, A. Wolisz, and S. Member. Measurements of a wireless link in an industrial environment using an ieee 802.11-compliant physical layer. *IEEE Transactions on Industrial Electronics*, 43:1265–1282, 2002.

[155] Wireless Africa. `http://wirelessafrica.meraka.org.za/wiki/index.php/Wireless_Africa_Home%_Page`.

[156] S. H. Y. Wong, H. Yang, S. Lu, and V. Bharghavan. Robust rate adaptation for 802.11 wireless networks. In *Proceedings of the 12th annual international conference on Mobile computing and networking*, MobiCom, pages 146–157, New York, NY, USA, 2006. ACM.

[157] G. R. Woo, P. Kheradpour, D. Shen, and D. Katabi. Beyond the bits: cooperative packet recovery using physical layer information. In *MobiCom*

'07: Proceedings of the 13th annual ACM international conference on Mobile computing and networking, pages 147–158, New York, NY, USA, 2007. ACM.

[158] Y. Yang, J. Wang, and R. Kravets. Designing routing metrics for mesh networks. In *IEEE Workshop on Wireless Mesh Networks (WiMesh)*, Sept. 2005.

[159] R. Yavatkar and N. Bhagawat. Improving end-to-end performance of TCP over mobile internetworks. In *Proceedings of the 1994 First Workshop on Mobile Computing Systems and Applications*, pages 146–152, Washington, DC, USA, 1994. IEEE Computer Society.

[160] S. Yong and L. B. Sung. XOR retransmission in multicast error recovery. In *Proceedings of IEEE International Conference on Networks (ICON)*, pages 336 –340, 2000.

[161] Y. Yuan, H. Yuan, S. H. Wong, S. Lu, and W. Arbaugh. ROMER: resilient opportunistic mesh routing for wireless mesh networks. In *Proc. of IEEE WiMESH*, Sept. 2005.

[162] S. Yuk and D. Cho. Parity-based reliable multicast method for wireless LAN environments. In *Proc. of VTC*, May 1999.

[163] K. Zeng, W. Lou, and H. Zhai. On end-to-end throughput of opportunistic routing in multirate and multihop wireless networks. In *Proc. of IEEE INFOCOM*, Apr. 2008.

[164] K. Zeng and Z. Y. W. Lou. Oportunistic routing in multi-radio multi-channel multi-hop wireless networks. In *Proc. of IEEE INFOCOM*, March 2010.

[165] X. Zhang and B. Li. Optimized multipath network coding in lossy wireless networks. In *Proc. of Int. Conf. on Distributed Computing Systems*, 2008.

[166] F. Zhao, D. Lun, M. Medard, and E. Ahmed. Decentralized algorithms for operating coded wireless networks. *Information Theory Workshop, 2007. ITW '07. IEEE*, pages 472–477, Sept. 2007.

[167] Z. Zhong and S. Nelakuditi. On the efficacy of opportunistic routing. In *Proc. of IEEE SECON*, June 2007.

[168] Z. Zhou, P. K. McKinley, and S. M. Sadjadi. On quality-of-service and energy consumption tradeoffs in FEC-encoded wireless audio streaming. In *Proc. of IWQoS*, June 2004.