MITIGATING CONGESTION IN WIRELESS NETWORKS

A Dissertation in
Computer Science and Engineering
by
Raju Kumar

© 2010 Raju Kumar

Submitted in Partial Fulfillment
of the Requirements
for the Degree of

Doctor of Philosophy

August 2010
The dissertation of Raju Kumar was reviewed and approved* by the following:

Thomas La Porta  
Distinguished Professor of Computer Science and Engineering  
Dissertation Advisor, Chair of Committee

Guohong Cao  
Professor of Computer Science and Engineering

Sencun Zhu  
Associate Professor of Computer Science and Engineering & College of Information Sciences and Technology

Aylin Yener  
Professor of Electrical Engineering

Raj Acharya  
Professor of Computer Science and Engineering  
Head of the Department of Computer Science and Engineering

*Signatures are on file in the Graduate School.
Cellular networks and Wireless LAN provide ubiquitous wireless connectivity today. These one-hop wireless networks have been extended to multi-hop wireless networks like mesh, ad hoc, and sensor networks. Since the wireless medium is broadcast in nature, multi-hop wireless networks are susceptible to congestion due to excessive traffic in a part of the network. In this dissertation, five algorithms are proposed to ameliorate the impact of congestion on different wireless networks under separate scenarios.

An important aspect of congestion in wireless networks is its indiscriminate impact. Data in sensor networks may not all be equally important and congestion in such a network may lead to low priority data being delivered while high priority data is dropped en route to the destination. A differentiated routing scheme is proposed to address this issue.

Network coding, heralded as a means to better utilize the medium’s capacity as compared to traditional routing’s approach of store-and-forward, is also susceptible to congestion. Channelization, or reserved access to the medium, has been proposed as a means to eliminate congestion by assigning parts of the medium to different links. In this dissertation, channelization schemes are proposed for intra-flow network coding of multicast flows in a wireless network. These channelization schemes effectively restore the performance of network coding.

Congestion in wireless networks has also been addressed by using appropriate transmission rates in a multi-rate wireless MAC. But a multi-rate medium raises the issue of rate selection, the problem of selecting which transmission rate to use. When inter-flow network coding is used to deliver unicast traffic in such a network, it requires some packets to be multicast. A rate selection scheme is proposed to determine the appropriate transmission rate while taking network coding into account and ensuring that the throughput on the multicast link is maximized. Finally, all prominent rate selection schemes are hop-by-hop. We propose an end-
to-end rate selection scheme for networks that leverage opportunistic reception as a means to increase the medium capacity to counter congestion.
# Table of Contents

List of Figures ix  
List of Tables xii  
Acknowledgments xiii  

Chapter 1  
Introduction 1  
1.1 Routing .................................................. 2  
1.2 Network Coding ........................................... 3  
   1.2.1 Network Coding and Selective Channelization ...... 5  
   1.2.2 Network Coding and Cooperative Channelization ... 6  
1.3 Multi-Rate Wireless Networks and Rate Selection .......... 6  
   1.3.1 Rate Selection for Network Coding .................. 7  
   1.3.2 End-to-End Rate Selection for Opportunistic Reception 8  
1.4 Outline ................................................... 9  

Chapter 2  
Mitigating Performance Degradation in Congested Sensor Networks 10  
2.1 Introduction .............................................. 10  
2.2 Related Work ............................................ 12  
2.3 Congestion Aware Algorithms ................................ 14  
   2.3.1 Overview ............................................. 14  
   2.3.2 Congestion Aware Routing (CAR) ....................... 15  
      2.3.2.1 High Priority Routing Network Formation ........ 17  
      2.3.2.2 Dynamic Conzone Discovery ..................... 19
### Chapter 2

<table>
<thead>
<tr>
<th>Subsection</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.3.2.3</td>
<td>Differentiated Routing</td>
<td>21</td>
</tr>
<tr>
<td>2.3.2.4</td>
<td>Enhancements</td>
<td>23</td>
</tr>
<tr>
<td>2.4</td>
<td>Performance Evaluation</td>
<td>24</td>
</tr>
<tr>
<td>2.4.1</td>
<td>Simulation Setup</td>
<td>25</td>
</tr>
<tr>
<td>2.4.2</td>
<td>Simulation Results</td>
<td>29</td>
</tr>
<tr>
<td>2.4.2.1</td>
<td>Varying Transmission Range</td>
<td>30</td>
</tr>
<tr>
<td>2.4.2.2</td>
<td>Varying Low Priority Data Rate</td>
<td>34</td>
</tr>
<tr>
<td>2.4.2.3</td>
<td>Varying High Priority Data Rate</td>
<td>35</td>
</tr>
<tr>
<td>2.5</td>
<td>Summary and Future Work</td>
<td>35</td>
</tr>
</tbody>
</table>

### Chapter 3

**Channelization for Network Coding in Wireless Networks**

<table>
<thead>
<tr>
<th>Subsection</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.1</td>
<td>Introduction</td>
<td>37</td>
</tr>
<tr>
<td>3.2</td>
<td>Motivation</td>
<td>39</td>
</tr>
<tr>
<td>3.2.1</td>
<td>Channelization</td>
<td>40</td>
</tr>
<tr>
<td>3.2.2</td>
<td>Channelization for Network Coding</td>
<td>40</td>
</tr>
<tr>
<td>3.3</td>
<td>Gain Analysis</td>
<td>41</td>
</tr>
<tr>
<td>3.3.1</td>
<td>One-Hop Network</td>
<td>42</td>
</tr>
<tr>
<td>3.3.2</td>
<td>Multi-Hop Network</td>
<td>46</td>
</tr>
<tr>
<td>3.4</td>
<td>Network Coding Analysis</td>
<td>47</td>
</tr>
<tr>
<td>3.4.1</td>
<td>Routing</td>
<td>48</td>
</tr>
<tr>
<td>3.4.2</td>
<td>Network Coding</td>
<td>48</td>
</tr>
<tr>
<td>3.5</td>
<td>Channelization</td>
<td>50</td>
</tr>
<tr>
<td>3.5.1</td>
<td>When To Channelize</td>
<td>50</td>
</tr>
<tr>
<td>3.5.2</td>
<td>How To Channelize</td>
<td>53</td>
</tr>
<tr>
<td>3.6</td>
<td>Network Coding Scheme</td>
<td>54</td>
</tr>
<tr>
<td>3.7</td>
<td>Results</td>
<td>55</td>
</tr>
<tr>
<td>3.8</td>
<td>Related Work</td>
<td>58</td>
</tr>
<tr>
<td>3.9</td>
<td>Summary</td>
<td>59</td>
</tr>
</tbody>
</table>

### Chapter 4

**Cooperative Channelization in Wireless Networks with Network Coding**

<table>
<thead>
<tr>
<th>Subsection</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.1</td>
<td>Introduction</td>
<td>61</td>
</tr>
<tr>
<td>4.2</td>
<td>Network Model</td>
<td>62</td>
</tr>
<tr>
<td>4.2.1</td>
<td>Medium Access Control</td>
<td>63</td>
</tr>
<tr>
<td>4.2.2</td>
<td>Transmission Probability Assignment</td>
<td>64</td>
</tr>
<tr>
<td>4.2.3</td>
<td>Network Coding</td>
<td>66</td>
</tr>
<tr>
<td>4.2.4</td>
<td>Metrics</td>
<td>66</td>
</tr>
<tr>
<td>4.3</td>
<td>Hyperarc Health</td>
<td>67</td>
</tr>
</tbody>
</table>
## List of Figures

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1</td>
<td>Canonical Network Coding Butterfly Network Example [33].</td>
<td>4</td>
</tr>
<tr>
<td>2.1</td>
<td>A critical area of a sensor network may generate high priority data at a high rate. This causes congestion in a part of the network exacerbated by the presence of low priority data being routed in that area.</td>
<td>16</td>
</tr>
<tr>
<td>2.2</td>
<td>Presence of congestion with AODV routing in a network subjected to high priority data rate of 30 packets/sec (pps) and background low priority traffic rate of 0.5 pps. Thin lines represent low priority traffic while thick lines represent high priority traffic.</td>
<td>17</td>
</tr>
<tr>
<td>2.3</td>
<td>In a dense deployment, multiple nodes can be parents of a node. Each parent lies on a different shortest path route to the sink. This structure is used for shortest multipath routing.</td>
<td>18</td>
</tr>
<tr>
<td>2.4</td>
<td>Conzone discovery algorithms in CAR for node x.</td>
<td>20</td>
</tr>
<tr>
<td>2.5</td>
<td>Routing algorithm for CAR for low and high priority data inside the conzone.</td>
<td>23</td>
</tr>
<tr>
<td>2.6</td>
<td>Simulation scenario.</td>
<td>24</td>
</tr>
<tr>
<td>2.7</td>
<td>Routing views for all data and packet drop analysis for high priority packets for AODV and CAR for range = 130 m, low priority data rate = 0.5 pps, high priority data rate = 30 pps.</td>
<td>25</td>
</tr>
<tr>
<td>2.8</td>
<td>Varying transmission range, low priority data rate = 0.5 pps, high priority data rate = 30 pps.</td>
<td>28</td>
</tr>
<tr>
<td>2.9</td>
<td>Maximum node energy used: Varying transmission range, low priority data rate = 0.5 pps, high priority data rate = 30 pps.</td>
<td>31</td>
</tr>
<tr>
<td>2.10</td>
<td>Varying low priority data rate, Transmission range = 130 m, High priority data rate = 30 pps.</td>
<td>32</td>
</tr>
<tr>
<td>2.11</td>
<td>Varying high priority data rate, Transmission range = 130 m, Low priority data rate = 0.5 pps.</td>
<td>33</td>
</tr>
<tr>
<td>3.1</td>
<td>Network Coding and Channelization</td>
<td>43</td>
</tr>
</tbody>
</table>
3.2 Number of Transmissions: Number of Flows = 1, Number of Packets = 100, Packet Size = 512 Bytes ............................... 52
3.3 Varying Link Error Rate: Number of Flows = 1, Number of Packets = 50, Packet Size = 512 Bytes ............................... 56
3.4 Varying contention Level: Number of Flows = 1, Number of Packets = 50, Packet Size = 512 Bytes, Link Error Rate = 0.4 ....... 58

4.1 Four one-hop multicast flows in a 2-hop neighborhood, \( q_{\text{min}} = 1.0, q_{\text{max}} = 3.0 \) .................................................. 71
4.2 Four one-hop multicast flows in a 2-hop neighborhood, \( q_{\text{min}} = 1.0, q_{\text{max}} = 3.0 \) .................................................. 72
4.3 Two-hop multicast flow, \( q_{\text{min}} = 2.0, q_{\text{max}} = 6.0 \) ........................................... 75
4.4 Two-hop multicast flow, \( q_{\text{min}} = 2.0, q_{\text{max}} = 6.0 \) ........................................... 76
4.5 25 flows over a 50 node network, \( q_{\text{min}} = 4.0, q_{\text{max}} = 12.0 \) ......................... 80
4.6 Channelizations for 25 flows over a 50 node network, \( q_{\text{min}} = 4.0, q_{\text{max}} = 12.0 \) ........................................... 81
4.7 Varying Probability, \( q_{\text{min}} = 2.0, q_{\text{max}} = 6.0 \) ........................................... 84

5.1 Node \( n_3 \) has to deliver \( \text{pkt}_0 \) of flow 0 to \( n_1 \), \( n_4 \) has to deliver \( \text{pkt}_1 \) of flow 1 to \( n_2 \) .................................................. 90
5.2 Experiment vs. simulation results for 1500Byte packets .................. 92
5.3 Small scenarios for inter-flow network coding of 2 flows (not to scale) 94
5.4 Gain over routing for packets delivered to all hops for small scenarios 107
5.5 Gain over routing for packets delivered to all hops for 3-node scenarios with varying SNR combinations - \( \text{SNR}_1 = \text{SNR} \) on \( n_0 \to n_1 \), and \( \text{SNR}_2 = \text{SNR} \) on \( n_0 \to n_2 \), Load = 1500pps, Maximum Gain - NCRS-MinRS: 16%, NCRS-MaxRS: 21% ....................... 109
5.6 Gain over routing for packets delivered to destinations for random-ized grids and random flows ........................................ 111

6.1 Opportunistic reception example - Links are annotated with packet loss probability ................................................. 121
6.2 Wireless network - Flows, Conflict Graph, and Maximal Cliques in conflict graph .................................................. 125
6.3 Opportunistic reception in flow \( r \): Destination \( D \) receives packets at rate \( x_r \), Source \( S \) transmits packets at rate \( x_r / \beta_r \), Intermediate node \( R_b \) forwards packets at rate \( x_r \cdot \alpha_r(j+1) / \beta_r \) ......................... 127
6.4 Packet error rate for 1500Byte packets with IEEE 802.11a trans-mission rates .......................................................... 133
6.5 Convergence of Received Rate and Transmission Rates .................. 135
6.6 Single Flow - Received Rate Gain ........................................ 137
6.7 Single Flow - Avg. Weighted Transmission Rate Ratio ............ 138
6.8 Single Flow - Normalized number of Hops ............................. 141
6.9 Multiple Flows - Received Rate Gain .................................... 143
# List of Tables

<table>
<thead>
<tr>
<th>Table</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1</td>
<td>Summary of Schemes</td>
<td>24</td>
</tr>
<tr>
<td>4.1</td>
<td>Notation ($A^i_v$ denotes $A$ in node $v$ for flow $i$)</td>
<td>65</td>
</tr>
<tr>
<td>4.2</td>
<td>Abbreviations</td>
<td>82</td>
</tr>
<tr>
<td>5.1</td>
<td>Transmission Rates and Rate Adaptation for 1500Byte packets</td>
<td>93</td>
</tr>
<tr>
<td>5.2</td>
<td>Transmission Rates [Mbps] for Hyperarcs in Small Scenarios</td>
<td>112</td>
</tr>
<tr>
<td>6.1</td>
<td>Throughput [pps] for IEEE 802.11a transmission rates for 1500Byte packets</td>
<td>134</td>
</tr>
<tr>
<td>6.2</td>
<td>Example: Packet Error Rates (PER) for Transmission Rates</td>
<td>136</td>
</tr>
<tr>
<td>6.3</td>
<td>Example: Transmission Rates [Mbps] for Baselines and NUM-RS</td>
<td>139</td>
</tr>
<tr>
<td>6.4</td>
<td>$b_l$ of maximal cliques for a 8-hop flow</td>
<td>140</td>
</tr>
</tbody>
</table>
Acknowledgments

First and foremost, I would like to thank Dr. Thomas La Porta who has been a patient advisor throughout the Ph.D. program. His constant guidance was instrumental in all components of this dissertation. His approach to research problems has been a considerable influence on the problems that I explore in this dissertation.

I would also like to thank Dr. Guohong Cao, Dr. Sencun Zhu, and Dr. Aylin Yener for serving on my Ph.D. Committee and providing valuable insights on a range of issues related to the dissertation. I am also grateful to Dr. Srikanth Krishnamurthy for his assistance with this dissertation.

It was my privilege to work with Patrick Traynor and JaeSheung Shin in the first few years of the program. Thereafter, Hosam Rowaihy, Heesook Choi, JaeSheung Shin, Felipe de Mello, Srikar Tati, and Sharanya Eswaran provided vital help with the research problems that I explore in this dissertation. I would also like to extend my gratitude to Networking and Security Research Center members who assisted me in different ways over the years.

Finally, I would like to thank Mom, Bhaiya, and Bhabhiji for their unrelenting support throughout this period, and Ram, Manikant, and Ankur for the incredible assistance that they have provided over the years.
Dedication

To Ghar
Chapter 1

Introduction

The emergence of Cellular Networks and Wireless LANs has led to ubiquitous connectivity to desired sources of information. These one-hop wireless networks have been extended to multi-hop networks in which a collection of devices with wireless interfaces discover other such devices in the vicinity and form a network to transfer data from one node to another.

A range of multi-hop wireless networks exist today - primarily mesh, ad hoc, and sensor networks. Mesh networks [12], comprised of static or mobile mesh clients, have been proposed to provide wireless access across multiple hops. Characterized by a few wireless hops at maximum and a backhaul link to a gateway network, mesh networks have been extended to ad hoc and sensor networks. Ad hoc networks consist of several nodes that transfer data on a peer-to-peer basis as compared to the centralized approach of cellular networks. The form factor of these ad hoc network nodes was reduced to create sensor nodes which can perform tasks like sensing temperature, humidity, etc. Networks of such nodes have often been deployed to monitor different features of the deployed environment [11]. While the nodes in an ad hoc network are constrained in terms of battery power, sensor network nodes have more severe constraints in terms of processing power, memory, and energy available.

Mesh, ad hoc, and sensor networks are the primary domains along which research in multi-hop wireless networks has been conducted. These form the set of networks that we investigate. We concentrate on the issue of congestion in these wireless networks in this dissertation. Congestion leads to increased delay in ac-
cess to medium, increased errors due to collisions, highly occupied queues, buffer overflows, etc. Besides causing data to be lost, in wireless multi-hop networks, congestion also results in wastage of energy and reduction of network lifetime. As a result, addressing the issue of congestion is not just vital to prevent performance degradation but also to avoid a breakdown of the network itself.

The goal of this dissertation is to recognize performance degradation due to congestion in multi-hop wireless networks that employ routing or network coding, analyze the source of congestion and propose and verify the efficacy of solutions via analysis and simulations.

We first propose a differentiated routing scheme to address congestion in sensor networks in §1.1. We find network coding, heralded as a means to improve network throughput compared to routing, to be susceptible to congestion and explore channelization schemes to augment network coding in §1.2. Since wireless networks are comprised of links that exhibit different and time-varying characteristics, multirate MAC protocols like IEEE 802.11(b/a/g) are widely used. We propose rate selection schemes for such networks to better utilize network capacity in §1.3.

1.1 Routing

Nodes in a sensor network have diverse capabilities - sensing temperature, noise, audio levels, capturing video, etc. There are two primary approaches to data generation in sensor networks - periodic, and triggered by events such as intrusion, fire, flood, node failure, etc. If data generation is based on triggers, given the high deployment density of sensor networks, multiple sensor nodes are triggered into generating data. All such data is generally collected at special nodes called sink(s) in the network for analysis at a later point in time. Though in-network processing [41] can delay congestion, with sufficient events in a network, congestion will still occur.

We believe that in spite of the eventual aim of low cost sensor nodes, deploying hundreds of such nodes in an environment like the Great Duck Island [72] will incur a significant cost. As a result, such a high cost network needs to be leveraged to gather different types of data - temperature, audio, video, intrusion, etc. Some of these data may not be as important as others. Specifically, periodic data may not
be as important as data generation triggered based on event detection. Hence in Chapter 2 we consider two categories of data - high and low priority.

Routing all data generated from an area to a sink (e.g., as a response to a fire) leads to congestion in the region connecting the event area to the sink if the datarate at which the event needs to be reported is sufficiently high. Since the paths from a set of nodes in a region to a sink span the same geographical area and hence interfere with each other, the congestion problem becomes worse. Due to these interfering paths, a routing layer based solution is needed to address the issue of congestion. We propose *Congestion Aware Routing (CAR)* [57] in Chapter 2 as a differentiated routing scheme to address this problem. CAR discovers the congestion zone (*conzone*) and reserves it to deliver only high priority packets. Low priority packets generated outside the conzone are delivered to their destination by using nodes that do not lie in the aforementioned congested area. Low priority packets that are generated inside a conzone are routed out efficiently. Our results show that CAR improves the data delivery characteristics of high priority data and even improves the fraction of low priority data delivered. It also decreases energy consumption in nodes that were otherwise congested and used a larger amount of energy.

1.2 Network Coding

Network Coding was proposed by Ahlswede et al. [7] as a conceptual departure from the approach of routing. In routing, nodes store a received packet, determine the next-hop destination and forward the packet accordingly. This store-and-forward approach of routing was replaced by the store-modify-forward approach of network coding in which intermediate nodes in the network are allowed to modify data before forwarding. This has led to a large collection of follow up work which concentrates on schemes for effective ways to encode and decode such data [38, 55, 68]. Significantly, network coding has been shown to achieve the multicast capacity of the channel while routing fails to do so.

Figs. 1.1(a) and 1.1(b) compare the performance of routing and network coding using the canonical butterfly network example [33]. Nodes $N_1$ and $N_2$ need to deliver packets $P_1$ and $P_2$, respectively, to nodes $N_5$ and $N_6$. If routing is used,
6 multicast transmissions are required as shown in Fig. 1.1(a). But as shown in Fig. 1.1(b) if network coding is used, only 4 multicast transmissions are required. The gain brought by network coding, defined as the ratio of number of transmissions for routing to that for network coding, is $1.5 \times$. Hence network coding requires significantly less network bandwidth to support the same set of flows as compared to routing. As a result, network coding can be employed to postpone the onset of congestion.

Congestion is born out of competition for medium access. Severe competition for medium access drastically reduces the effective network bandwidth utilization over contention based MAC protocols like CSMA/CA [98], IEEE 802.11 [1], etc. To reduce the loss of available bandwidth due to conflict resolution, channelization based schemes have been proposed in works like [85, 86]. In such schemes, network access resources like time-slots (Time Division Multiple Access (TDMA)) or frequencies (Frequency Division Multiple Access (FDMA)) are assigned to nodes or links in the network, as appropriate. Since the networks that we are interested in are multi-hop, this assigning of network resources has to be a distributed scheme.

Though network coding serves to increase the effective throughput of a network, at high loads, it may struggle to retain its efficacy. Channelization has been proposed as a solution for networks facing high loads. In Chapters 3 and 4 we consider TDMA slots as channels. With such a TDMA MAC, channelization refers
to allotting slots for transmission to nodes (or links) and is hence also referred to as scheduling. Scheduling every node (or link, as appropriate) in a multi-hop network is not feasible (in spite of taking spatial reuse [74] into account) as it requires a large number of slots in a frame. Such a large frame size leads to large delays in data delivery and even buffer overflow due to a larger packet arrival rate than the packet departure rate. In addition, channelization of a node takes away resources from the set of two-hop neighbors. The tradeoff of channelization is that it improves the performance of the channelized node or link, but it compresses the rest of the traffic into a smaller available bandwidth. As a result, we need to use channelization only when doing so has sufficient benefits.

In §1.2.1 we summarize a channelization scheme proposed for intra-flow network coding of multicast flows (cf. Chapter 3). We subsequently enhance this channelization scheme in Chapter 4 (summarized in §1.2.2).

### 1.2.1 Network Coding and Selective Channelization

The drawbacks of exhaustive channelization lead us to propose the concept of selective channelization [56] in Chapter 3 in which we recognize the hyperarcs i.e., multicast links, in the network that need to be channelized. Selective channelization takes into account the effectiveness of network coding in addressing the errors over each hyperarc. This scheme is based upon our analysis of the gains brought by network coding as compared to routing over a single hop flow with multiple destinations.

We propose two channelization trigger schemes - counting and model based. The counting based scheme adopts the approach of actually maintaining the metric of average number of transmissions required to deliver a packet to all destinations of a hyperarc. In contrast, the model based scheme estimates the contention error rates at all the destinations of the hyperarc and uses our network coding analysis to estimate the number of transmissions that network coding would require due to errors resulting from contention only (i.e., not link errors). Channelization is triggered when the average number of transmissions to deliver a packet exceeds a pre-defined threshold. Our results show that the model based scheme enables us to channelize fewer hyperarcs in the network while keeping the performance level
of the network the same as the count based scheme.

1.2.2 Network Coding and Cooperative Channelization

Excessive competition for medium access leads to performance degradation in a neighborhood. It is vital to note that this onset of degradation arises out of competition. If there is a set of links competing for access in a neighborhood, channelizing some of them will lead to the performance of the others being restored to acceptable levels. This insight is used in our cooperative channelization [59] scheme for network coding in wireless networks proposed in Chapter 4.

Since performance degradation due to congestion arises out of competition for channel access among nodes in a neighborhood, cooperative channelization decisions are made while taking the neighborhood into account. We define a composite health metric that includes factors for the impact of collisions, queue occupancy, network coding opportunity, and effect of channelization on next-hop downstream hyperarcs. The cooperative channelization algorithm uses this health metric to compare need for channelization of different multicast links. This enables cooperative channelization to efficiently distribute network resources to improve the overall system performance. In contrast, selective channelization [56] does not consider delays or queue occupancy and is a selfish scheme that determines need for channelization of hyperarcs in isolation. Our results show that, in general, channelization combined with network coding during congestion improves performance of the affected hyperarcs. When combined with network coding, cooperative channelization also decreases delivery delay as compared to the selective channelization algorithm.

1.3 Multi-Rate Wireless Networks and Rate Selection

Packet reception in wireless networks is governed by several factors - signal propagation characteristics, separation, antenna type, background noise, interference, collisions, multi-path fading, etc. As a result, in practical wireless networks, channel conditions on and among links often vary widely. IEEE 802.11, a widely
prevalent PHY and MAC protocol for wireless networks, attempts to improve the performance of a link under any given condition by transmitting at appropriate transmission rates and modulation schemes. For example, IEEE 802.11b provides four transmission rates - 1, 2, 5.5, and 11Mbps; IEEE 802.11a provides eight transmission rates - 6, 9, 12, 18, 24, 36, 48, and 54Mbps. Selecting a transmission rate incurs an inherent tradeoff wherein an increasing transmission rate results in decreasing packet delivery probability. These factors can be combined into throughput, frequently the metric for network performance. As link characteristics vary, a different transmission rate may increase throughput.

Though IEEE 802.11 does not specify an algorithm to select a transmission rate for a link, or to adapt it as the channel conditions vary, this problem has received considerable attention [16, 20, 39, 44, 45, 62, 82, 91]. This problem of adapting transmission rate to varying link characteristics has been referred to as rate adaptation. For any given channel condition, the component of rate adaptation that selects the transmission rate is referred to as rate selection.

We propose a rate selection scheme for inter-flow network coding of unicast flows in Chapter 5 and summarize it in §1.3.1. We then propose an end-to-end rate selection scheme for opportunistic reception in wireless networks in Chapter 6 and summarize it in §1.3.2.

1.3.1 Rate Selection for Network Coding

Network coding enables the delivery of different packets to distinct neighbors with a single transmission. In a network using inter-flow network coding of unicast flows, coded packets must be multicast for network coding to gain efficiency. Moreover, it is essential that uncoded packets be overheard by neighbors to enable inter-flow coding i.e., uncoded packets may need to be multicast.

A rate adaptation algorithm may determine different transmission rates from a node to its different neighbors. In a network using inter-flow network coding of unicast flows, rate selection refers to the problem of selecting a transmission rate for all multicast recipients, or any neighbors which may benefit from overhearing. If rate selection for uncoded packets precludes overhearing in a neighborhood, few inter-flow coding opportunities will arise. Rate selection for coded packets has to
address the following tradeoff - if a low transmission rate is chosen, all destinations may receive a transmission successfully but the transmission time will be longer; if a higher rate is chosen, the transmission time will decrease but destinations with poor link quality may not receive the transmission.

In Chapter 5 we propose Network Coding aware Rate Selection (NCRS) [60], a rate selection algorithm to maximize throughput in a multi-hop multi-rate wireless network in which network coding is used. Virtually all prior work focuses on maximizing coding gain i.e., maximize the reduction in network traffic due to network coding. Often these methods use the lowest transmission rate available to enable network coding to the largest extent possible. NCRS selects transmission rates to maximize throughput while taking network coding into account. In some cases this entails setting rates that fully leverage network coding, and in others coding is largely precluded and yet, throughput is maximized.

We first identify the prevalent fundamental building blocks in a network that enable network coding. We analyze the performance of network coding and routing over these building blocks in terms of throughput in a multi-rate environment. Based on this analysis, we propose NCRS as a linear programming problem to maximize total throughput on a multicast link. Our results illustrate the wide applicability of NCRS by examining several scenarios that contain multicast links which support different transmission rates on component unicast links. In addition, for a large network, we show that NCRS outperforms the alternative rate selection schemes and is also more robust than these alternatives in that it works for a wide range of channel characteristics.

1.3.2 End-to-End Rate Selection for Opportunistic Reception

Traditional routing in wireless networks forwards packets in a hop-by-hop manner on a selected route. By virtue of the wireless medium being broadcast in nature, some transmissions may be serendipitously received multiple hops downstream with a non-negligible probability. This phenomenon, referred to as opportunistic reception, occurs frequently in a multi-rate wireless network as illustrated in Afanasyev et al. [6]. Opportunistic reception has recently been leveraged by Biswas
et al. [18] and Chachulski et al. [21], among others.

In Chapter 6, we propose a low overhead end-to-end rate selection algorithm called NUM-based Rate Selection (NUM-RS) [58] that accounts for opportunistic reception in a multi-rate wireless network. NUM-RS uses end-to-end semantics to select both source rates and link transmission rates. It also considers tradeoffs of link occupancy time with opportunistic reception when setting link rates.

We leverage the Network Utility Maximization (NUM) framework, first proposed by Kelly et al. [49, 50] for wired networks. The incorporation of NUM enables a distributed scheme to determine the rates at which traffic for different flows should be injected onto links while maintaining proportional fairness [49]. Thus NUM-RS inherits salient attributes of NUM - feasibility, fairness, and the ability to implement a distributed, low overhead protocol.

We extend the NUM framework as applied to wireless networks to explicitly take the characteristics of a multi-rate wireless network and opportunistic reception into account and to enable end-to-end rate selection. We then design a protocol to implement NUM-RS in a wireless network and analyze its overhead. We conclude that NUM-RS imposes a marginal increment in overhead as compared to other NUM based schemes. Our results show that in networks where source rates are determined using the NUM framework, existing hop-by-hop approaches for rate selection are sub-optimal. NUM-RS increasingly outperforms other hop-by-hop rate selection schemes as scope for opportunistic reception increases.

1.4 Outline

The rest of this dissertation is organized as follows. In Chapter 2 we present our approach to mitigate performance degradation in congested sensor networks that use routing. In Chapter 3 we present selective channelization for intra-flow network coding of multicast flows in wireless networks. Subsequently, we improve on selective channelization by proposing cooperative channelization in Chapter 4. Next, we propose a rate selection scheme for inter-flow network coding of unicast flows in Chapter 5. Finally, in Chapter 6 we propose an end-to-end rate selection scheme for wireless networks that leverage opportunistic reception. We conclude this dissertation and discuss future research directions in Chapter 7.
Data generated in wireless sensor networks may not all be alike; some data may be more important than others and hence may have different delivery requirements. In this chapter we address differentiated data delivery in the presence of congestion in wireless sensor networks. We propose a class of algorithms that enforce differentiated routing based on the congested areas of a network and data priority. The basic protocol, called Congestion Aware Routing (CAR), discovers the congested zone of the network that exists between high priority data sources and the data sink, and using simple forwarding rules dedicates this portion of the network to forwarding primarily high priority traffic. We present extensive simulation results for CAR.

2.1 Introduction

Sensor network deployments may include hundreds or thousands of nodes. Since deploying such large-scale networks has a high cost, it is increasingly likely that sensors will be shared by multiple applications and gather various types of data: temperature, the presence of lethal chemical gases, audio and/or video feeds, etc. Therefore, data generated in a sensor network may not all be equally important.

With large deployment sizes, congestion becomes an important problem. Congestion may lead to indiscriminate dropping of data (i.e., high priority packets may
be dropped while low priority packets are delivered). It also results in an increase in energy consumption to route packets that will be dropped downstream as links become saturated. As nodes along optimal routes are depleted of energy, only non-optimal routes remain, further compounding the problem. To ensure data with higher priority is received in the presence of congestion due to low priority packets, differentiated service must be provided.

In this chapter we are interested in congestion that results from excessive competition for the wireless medium. Existing schemes detect congestion while considering all data to be equally important. We characterize congestion as the degradation of service to high priority data due to competing low priority traffic. In this case, congestion detection is reduced to identifying competition for medium access between high and low priority traffic.

Congestion becomes worse when a particular area is generating data at a high rate. This may occur in deployments in which sensors in one area of interest are requested to gather and transmit data at a higher rate than others (similar to bursty convergecast [95]). In this case, routing dynamics can lead to congestion on specific paths. These paths are usually close to each other, which leads to an entire zone in the network facing congestion. We refer to this zone, essentially an extended hotspot, as the congestion zone or conzone.

In this chapter, we examine data delivery issues in the presence of congestion. We propose the use of data prioritization and a differentiated routing protocol and/or a prioritized medium access scheme to mitigate its effects on high priority traffic. We strive for a solution that accommodates both low and high priority traffic when the network is static or near static and enables fast recovery of low priority traffic in networks with mobile high priority data sources. Our solution uses a differentiated routing approach to effectively separate high priority traffic from low priority traffic in the sensor network. High priority traffic has exclusive use of nodes along its shortest path to the sink; low priority traffic is routed over uncongested nodes in the network but may traverse longer paths.

Our contributions in this chapter are –

- **Design of Congestion Aware Routing (CAR):** CAR is a network layer solution to provide differentiated service in congested sensor networks. CAR also prevents severe degradation of service to low priority data by utilizing
uncongested parts of the network.

We compare CAR to an AODV [77] scheme enhanced with priority queues (referred to as AODV+PQ). CAR leads to a significant increase in the successful packet delivery ratio of high priority data, and a clear decrease in the average delivery delay compared to AODV+PQ. CAR also provides low jitter. Moreover, it uses energy more uniformly in the deployment and reduce the energy consumed in the nodes that lie on the conzone which leads to an increase in connectivity lifetime. In the presence of sufficient congestion, CAR also allows an appreciable amount of low priority data to be delivered.

The rest of this chapter is organized as follows. §2.2 presents related work. Details of CAR are presented in §2.3. Simulation details and results are presented in §2.4. Finally, §2.5 presents summary and future directions.

2.2 Related Work

An obvious solution to enhance service to high priority data is to use priority queues to provide differentiated services (see [10, 70, 95]). However, in such schemes, though high priority packets get precedence over low priority packets within a node, at the MAC layer they still compete for a shared channel with low priority traffic sent by surrounding nodes. As a result, without a routing scheme to address the impact of congestion and hotspots in the network, local solutions like priority queueing are not sufficient to provide adequate priority service to important data.

QoS in sensor networks has been a focus of current research (e.g., [10, 36, 97]). SPEED [36] provides soft real-time guarantees for end-to-end traffic using feedback control and location awareness. It also concludes that local adaptation at the MAC layer alone is insufficient to address the problem of hotspots and that routing is essential to the solution. Akkaya et al., [10] propose an energy aware QoS routing protocol to support the delivery of real-time data in the presence of interfering, non real-time data by using multiple queues in each node in a cluster based network; they do not consider the impact of congestion in the network and the interference that non real-time traffic can cause to real-time data. Zhang et al. [97] propose a generic model to achieve multiple QoS objectives.
Degrading service to one type of data to provide better service to another has been used in schemes like RAP [70] and SWAN [9]. Similar to these works, we segregate data; however, instead of real-time delivery demands, we use data priority as the basis for our segregation.

Approaches like 802.11e [5] and other differentiated MAC schemes that assign higher priority to important data (e.g., VoIP for 802.11e) via MAC layer mechanisms succeed at providing better service to high priority data by assigning them preferential medium access. Funneling-MAC [8], proposed by Ahn et al., addresses the issue of increased traffic intensity in the proximity of a sink by using a schedule-based and contention-based MAC hybrid. As with data aggregation schemes like [71, 83], it serves to delay the occurrence of congestion. Back-pressure and rate-limiting (also used in SPEED [36] and Fusion [40]) are essential to avoid situations where network capacity is less than the amount of traffic being injected into the medium. Rangwala et al. [80] propose Interference-Aware Fair Rate Control (IFRC) which employs schemes to achieve fair and efficient rate-limiting. It uses a tree rooted at each sink to route all data. When congestion occurs, the rates of the flows on the interfering trees are throttled. But these schemes do not adopt differentiated routing. Also, in a large network which is under congestion in a constrained area, our approach leverages the large uncongested parts of the network that is often underutilized to deliver low priority traffic.

RAP [70], SPEED [36] and MMSPEED [32] use velocity monotonic scheduling. Applications assign an expected speed to each data packet which is then ensured by these schemes. The speed that the application should assign to a packet if the network is congested is unclear. These schemes spread traffic around hotspots, but they do not give preference to high priority data. In fact, if low priority data has led to a hotspot in an area, routes for high priority data that later enter the network will circumvent this hotspot. This will increase the number of hops over which this data has to be routed and increase the energy consumed in the network. In the worst case, no path for high priority data may be found and these packets will be dropped. Additionally, MMSPEED [32] achieves reliability by duplicating packets and routing them over different paths to the destination.Duplication of packets in congested networks may further precipitate congestion. Also, these schemes do not explicitly separate low and high priority traffic generated in the same area.
Our schemes are different from these schemes because we use differentiated routing to provide best possible service to high priority data while trying to decrease the energy consumption in the conzone.

Congestion in sensor networks has been addressed in works like CODA [88], Fusion [40] and by Ee et al. [28]. Though these schemes take important steps to mitigate congestion in sensor networks, they treat all data equally. These schemes are complementary to the capability provided by CAR. Similarly, our solutions do not preclude the use of priority queues which can be added as a simple extension.

Existing work on congestion in sensor networks has two aspects - detection and mitigation. As mentioned earlier, we do not concern ourselves with congestion detection schemes in this chapter. Most mitigation schemes differ in how they invoke back-pressure and rate-limiting. Fusion’s [40] mitigation scheme (other than back-pressure and rate-limiting) is assigning preferential medium access to parents in the routing tree. This assumes that all data in a network be destined to a single sink which might not always be the case. In contrast, in our scenario, low priority data can be be sent from any node to any other node As a result, Fusion’s preferential MAC scheme is not applicable. Also, congestion in Fusion occurs due to accumulation of packets close to the sink. In contrast, we address the degradation of performance of high priority data delivery due to an extended hotspot in the network resulting from competition for medium access between low and high priority data. Also, Fusion does not do data differentiation based on priorities or provide differentiated routing.

### 2.3 Congestion Aware Algorithms

In the following subsection we introduce the network scenario and present an overview of our scheme which is then detailed in the subsection that follows it.

#### 2.3.1 Overview

An example of the problem scenario we consider is shown in Fig. 2.1. An important event occurs in one portion of the sensor field, which we call the critical area. This critical area will typically consist of multiple nodes. In such a scenario, there
is a data processing center for collecting sensitive information from the critical area. Such data is assigned a higher priority than other data. There might also be several nodes collecting different types of low priority information from other parts of the network. In the presence of this background low priority traffic, without differentiating between the two priority classes, congestion will degrade the service provided to high priority data. This may result in high priority data being dropped or delayed so long that it is of no use to the data processing center. We refer to the area that contains the shortest paths from the critical area to the sink as the conzone. High priority data would ideally traverse the conzone but will face competition for medium access due to low priority traffic.

Our basic solution, called Congestion Aware Routing (CAR) operates solely in the network layer. Packets are classified as high or low priority by the data sources, and nodes within a conzone only forward high priority traffic; low priority traffic is routed out of and/or around the conzone. In effect, we segment the network into two parts using forwarding rules. One limitation with this system is that it requires some overhead to discover the conzone. While this overhead is reasonable, it may still be too heavyweight if the data source is moving often and the conzone is changing frequently or if the high priority traffic is short-lived. Hence CAR is designed for static or nearly-static networks with long-lived high priority flows.

**2.3.2 Congestion Aware Routing (CAR)**

CAR is comprised of three steps – high priority network formation, conzone discovery, and differentiated routing. The combination of these functions segments the network into on-conzone and off-conzone nodes. Only high priority traffic is routed by on-conzone nodes. Note that the protocol specifically accommodates low priority traffic, albeit with less efficient routes than high priority traffic.

For the purposes of this discussion, we assume that there is one high priority sink and a contiguous part of the network (critical area) that generates high priority data in the presence of network-wide background low priority traffic. We also assume that nodes are location-aware (as in [36, 46]) and densely deployed with uniform distribution.

Since nodes in the scenario in Fig. 2.1 send all high priority data to a single
Figure 2.1. A critical area of a sensor network may generate high priority data at a high rate. This causes congestion in a part of the network exacerbated by the presence of low priority data being routed in that area.

If standard ad hoc routing schemes (e.g., AODV [77] or DSR [43]) are used to route the burst of high priority data instead of the tree-based routing scheme, congestion occurs. Fig. 2.2 shows the congestion zone that is formed when AODV is used for routing all data in a deployment of 120 nodes. There is one high priority sink and two low priority sinks as shown in the figure. Only critical area nodes send high priority data, while all other nodes in the network send low priority data to either of the low priority sinks. We do not show a similar figure for Directed
Diffusion [41] (using the One-Phase-Pull-Filter) because the control overhead of the initial flooding required with such a large number of data sources of low and high priority traffic was prohibitive and led to no high priority data to be delivered.

Figure 2.2. Presence of congestion with AODV routing in a network subjected to high priority data rate of 30 packets/sec (pps) and background low priority traffic rate of 0.5 pps. Thin lines represent low priority traffic while thick lines represent high priority traffic.

We now present the algorithms used by CAR to build high priority routing networks, to perform dynamic conzone discovery, and to provide differentiated routing. This is followed by the description of two enhancements of basic CAR.

2.3.2.1 High Priority Routing Network Formation

After deployment of sensor nodes, the high priority data collection center (the sink) initiates the process of building the high priority routing network (HiNet). This network covers all nodes because, at the time of deployment, the sink will usually have no information on the whereabouts of the critical area nodes. Also, based on the locations of events that can occur during the lifetime of the network, different nodes may constitute the critical area.

Since all high priority data is destined to a single sink, the HiNet is based on a minimum distance spanning tree rooted at the sink. As with TAG [71], this
structure ensures that all nodes have shortest path routes to the sink. However, instead of every node having a single parent, as in other tree-based schemes, we allow nodes to have multiple parents. A node that has multiple neighbors with depths (number of hops to the sink) less than its own, considers them all as parents (see Fig. 2.3). We leverage this property to support multi-path forwarding, thus providing load-balancing and making the routing network more resilient to failures.

Figure 2.3. In a dense deployment, multiple nodes can be parents of a node. Each parent lies on a different shortest path route to the sink. This structure is used for shortest multipath routing.

We now consider the HiNet formation process. Once the sink discovers its neighbors, it broadcasts a “Build HiNet” message (containing ID and depth of the node) asking all nodes in the network to organize as a graph. Once a neighboring node hears this message, it checks if it has already joined the HiNet (i.e., if it knows its depth); if not, it sets its depth to one plus the depth in the message received and sets the source of the message as a parent. This node then re-broadcasts the Build HiNet message, with its own ID and depth. If a node is already a member of the graph, it checks the depth in the message, and if that depth is one less than its own, then the source of the message is added as a parent. In this case, the message is not re-broadcast.

If a node receives a Build HiNet message with a depth value less than that of its parent’s depth, it updates its own value to the received value plus one. It then
removes all current parents and adds the source of the message as a new parent. Finally, the Build HiNet message is re-broadcast with the new depth value. In this fashion, the Build HiNet message is sent down the network until all nodes become part of the graph. Similar to TAG [71], the Build HiNet message can be periodically broadcast to maintain the topology and adapt to changes caused by the failure or addition of nodes.

2.3.2.2 Dynamic Conzone Discovery

Nodes discover if they are on the conzone by using the conzone discovery mechanism. After building the HiNet, the next task is to dynamically discover the conzone. The conzone is formed when one area is generating high priority data. We refer to this area as the critical area. This conzone discovery is done dynamically because the critical area can change during the lifetime of the deployment and is triggered when an area starts generating high priority data.

The conzone can be discovered and destroyed either from the critical area nodes to the sink or vice-versa. The conzone discovery algorithms allow nodes, in a distributed fashion, to determine if they are on a potentially congested path between the critical area and the sink. If they are, they mark themselves as “on-conzone”. The conzone discovery schemes are summarized in Fig. 2.4.

For brevity, we only present conzone discovery from the critical area to the sink in detail. In this case, critical area nodes detect an event that triggers discovery. A conzone must be then discovered from that neighborhood to the sink for delivery of high priority data. To do this, critical area nodes broadcast ”discover conzone to sink” (ToSink) messages. This message includes the ID of the source and its depth and is overheard by all neighbors. The depth is included here to ensure that nodes do not respond to ToSink messages heard from their parents. When a node hears more than \( \alpha \) distinct ToSink messages coming from its children, it marks itself as on-conzone and propagates a single ToSink message. This message is overheard by neighbors who mark this neighbor as being on the conzone in their neighborhood table. In our scheme, this threshold \( \alpha \) is a linear function of the neighborhood size (i.e., number of nodes within communication range) and of the depth of the node in the HiNet as shown in Eqn. 2.1. For node \( x \) with depth \( d_x \) and neighborhood size \( n_x \), we have:
Local variables:

Off-conzone parents: \( P_{off} = \{p_1, p_2, ..., p_n\} \)
Off-conzone siblings: \( S_{off} = \{s_1, s_2, ..., s_m\} \)
On-conzone parents: \( P_{on} = \{\} \)
On-conzone siblings: \( S_{on} = \{\} \)
Children: Children = \( \{c_1, c_2, ..., c_k\} \)
Node’s on-conzone status: On_Conzone = FALSE

ToSink messages received:
ToSink received = 0
ToSink threshold: \( \alpha_x = \beta_{d_x} \cdot d_x \cdot n_x \)

Conzone Discovery From Critical Area To Sink:
if node \( x \) receives ToSink from child \( c_i \) then
  if On_Conzone == FALSE then
    if ToSink_received > \( \alpha_x \) then
      On_Conzone = TRUE
      if \( x \) is not sink then
        broadcast ToSink with \( d_x \)
    else
      ToSink_received ++
  else if node \( x \) receives ToSink from parent \( p_j \) then
    \( P_{off} = \{p_j\} \); \( P_{on} = \{p_j\} \)
else if node \( x \) receives ToSink from sibling \( s_l \) then
  \( S_{off} = \{s_l\} \); \( S_{on} = \{s_l\} \)

Conzone Discovery From Sink To Critical Area:
if node \( x \) receives FromSink from parent \( p_i \) then
  \( P_{off} = \{p_i\} \); \( P_{on} = \{p_i\} \)
if On_Conzone == FALSE then
  if \( x \) has a critical area child \( c_j \in \) critical area then
    On_Conzone = TRUE
    if \( x \) is not a critical area node then
      broadcast FromSink with \( depth_x \)
  else if node \( x \) receives FromSink from sibling \( s_l \) then
    \( S_{off} = \{s_l\} \); \( S_{on} = \{s_l\} \)

Figure 2.4. Conzone discovery algorithms in CAR for node \( x \).
\[ \alpha_x = \beta d_x \cdot d_x \cdot n_x \] (2.1)

Since the depth and neighborhood size can vary for different nodes, \( \alpha \) is set accordingly. Setting \( \beta \) correctly for different depths ensures that the conzone is of the appropriate width. As \( \beta \) becomes smaller, the conzone becomes wider. Depth must also be taken into account because if \( \alpha \) is the same for different depths, the conzone will become very narrow as it approaches the sink. Note that due to the assumption of uniform deployments, neighborhood size is related to the number of children by a constant factor. Hence Eqn. 2.1 can be adapted to use number of children, but we use neighborhood size instead.

An important goal of the conzone discovery algorithm is to split the parents and siblings (nodes with the same depth) in the HiNet into on-conzone and off-conzone neighbors. Initially, all parents and siblings are marked as off-conzone. Since a node will forward a ToSink message only if it becomes on-conzone, when a node hears such a broadcast from its parent(s) or sibling(s), it marks that neighbor as on-conzone.

Since the presence of a conzone leads to sub-optimal routing for low priority data due to on-conzone nodes being dedicated to serving high priority data, after high priority stream comes to an end, the conzone is destroyed by flooding a "destroy conzone" message in the conzone.

### 2.3.2.3 Differentiated Routing

Once the conzone is discovered, high priority data is routed in the conzone and low priority data is routed off the conzone. Since the critical area is part of the conzone, all high priority data will be generated inside the conzone. Hence, routing of high priority data is simple; a node always forwards the data to one of its on-conzone parents. This parent is chosen randomly from the on-conzone parent list to balance the load among them. If for some reason the links to all parents are broken for example, because of node failures, the node will forward the data to a sibling which is on the conzone. If that is impossible, it will forward the data to any of its neighbors hoping that it can return to an on-conzone node.

Low priority data generated inside the conzone is routed out using the following
approach. When an on-conzone node gets a low priority message, it forwards it to an off-conzone parent, if there are any. Otherwise the low priority data is forwarded to an off-conzone sibling. If there are no parents or siblings that are off-conzone, we resort to the following method. After discovering the conzone, the sink sends a message through the conzone which contains the coordinates of a line that cuts the conzone in half. This line connects the sink to the center of the critical area. Using this information and its own coordinates, a node can determine on which half of the conzone it lies and hence route low priority data to the parent that is closest to the conzone boundary, i.e., farthest from the line. With the assumption of uniform deployment density, this ensures that all low priority data generated inside the conzone is routed out efficiently and along the shortest path.

The routing scheme described above is highly efficient for low priority traffic flowing in the same direction as the high priority traffic. Though it is not optimal for low priority traffic flowing in different directions, it will still correctly deliver the data while keeping the routing out cost low.

It is important to note here that to keep the routing overhead low, low priority routing decisions inside the conzone are static. So, once a node decides to which neighbor it is going to forward low priority data, it uses the same neighbor for all low priority packets. If that neighbor fails, an alternative must be found using the same scheme. In-conzone routing for both low and high priority data is summarized in Fig. 2.5.

Low priority data generated outside the conzone or routed out of the conzone has to be routed to the appropriate low priority sink without using the conzone nodes. Hence routing low priority data outside the conzone can use any of the known routing schemes, such as AODV, with modifications to prevent low priority data from being routed from an off-conzone node into the conzone. We used AODV in the off-conzone nodes to route low priority data with the modification that the on-conzone nodes do not propagate route request or reply messages for low priority data. Using this modified routing scheme, low priority data generated outside or routed out of the conzone is routed to its destination via off-conzone nodes only.
Routing Low Priority Data:
if $P_{off} \neq \{\} \text{ then}$
\hspace{1em} send data to any $p \in P_{off}$
\hspace{1em} \textbf{else if} $\exists$ a sibling $s \in S_{off}$ \textbf{then}
\hspace{1em} send data to $s$
\hspace{1em} \textbf{else}
\hspace{2em} send data to the farthest parent $p$ from dividing line

Routing High Priority Data:
if $P_{on} \neq \{\} \text{ then}$
\hspace{1em} send data to any $p \in P_{on}$
\hspace{1em} \textbf{else if} $\exists$ a sibling $s \in S_{on}$ \textbf{then}
\hspace{1em} send data to $s$
\hspace{1em} \textbf{else}
\hspace{2em} send data to any $u \in P_{off} \cup S_{off}$

**Figure 2.5.** Routing algorithm for CAR for low and high priority data inside the conzone.

### 2.3.2.4 Enhancements

In CAR, low priority data generated inside the conzone requires the conzone nodes to dedicate some of their resources to route such data out of the conzone. As an enhancement to better serve high priority data, on-conzone nodes stop generating or forwarding any low priority data. We call this enhancement CAR+.

Due to the shared nature of the wireless channel, high priority messages can be dropped by the critical area nodes themselves due to collisions with other low priority data from neighboring nodes. This is especially true if the amount of low priority traffic surrounding the critical area is large. As a second improvement, we disable generating and forwarding of low priority data in all nodes that are within the communication range of any critical area node. Since nodes know their neighbors and their status, once a node discovers that one of its neighbors is on the critical area, it disables generation and forwarding of any low priority data. We call this enhancement CAR++.
Table 2.1. Summary of Schemes

<table>
<thead>
<tr>
<th>Scheme</th>
<th>Summary</th>
</tr>
</thead>
<tbody>
<tr>
<td>CAR</td>
<td>For static or nearly-static conzone and long-lived high priority flows</td>
</tr>
<tr>
<td>CAR+</td>
<td>Conzone nodes drop all low priority data</td>
</tr>
<tr>
<td>CAR++</td>
<td>Conzone nodes and neighbors of critical area drop all low priority data</td>
</tr>
</tbody>
</table>

2.4 Performance Evaluation

In this section, we describe our simulation setups used to test CAR and discuss the results in detail. Table 2.1 provides a brief summary of our proposed schemes. Note that while CAR is designed for near-static networks, a scheme called MAC-Enhanced CAR (MCAR)\(^1\) was proposed to address the issue of mobile high priority data sources. The results in this chapter include results for MCAR. For details, refer to Kumar et al. [57].

---

\(^1\)The design, simulation and implementation results for MCAR were contributed by Riccardo Crepaldi, Albert F. Harris III and Michele Zorzi from the Department of Information Engineering at the University of Padova.
2.4.1 Simulation Setup

The simulations were conducted in NS-2 [3] with a deployment area of 560 m × 280 m. In this area 120 nodes are placed in a 15 × 8 grid, as shown in Fig. 2.6, with separation between neighboring nodes along both axes being 40 m. Note that we use grids as deployments in this chapter to emulate uniformly dense deployments and such grids are not a requirement of our algorithms. As long as the neighborhood relationships are similar, the results will not differ significantly from those presented in this chapter.

Two low priority sinks receive all low priority data while a single sink receives all high priority data. Three nodes form the critical area and send high priority data. The rest of the nodes, other than the three sinks and the three critical area nodes, send low priority data to either of the low priority sinks (see Fig. 2.6). This low priority data serves as the background traffic in our simulations. Note
that the high priority sources in our simulations were placed at the edge of the deployment to get a sufficient number of hops from them to the high priority sink. In a large deployment of hundreds of nodes, these high priority sources need not be at the edge of the deployment. Results were recorded when the system reached steady state. CAR uses AODV to route low priority data outside the conzone with a modification to ensure that off-conzone nodes do not route such data into the conzone.

IEEE 802.11 is used as the MAC layer operating at 11 Mbps. 802.11 is a CSMA/CA MAC layer that uses RTS/CTS to avoid the hidden terminal problem. Sensor network MAC schemes like S-MAC [93] and B-MAC [78] employ CSMA/CA and cut down the overhead of RTS/CTS. However, in dense networks that are under high congestion, these schemes will need to use RTS/CTS for each data packet to avoid the hidden terminal problem. Hence 802.11 is a reasonable approximation of S-MAC [93] under congested conditions. Actually, results in [93] show that a node that forwards sufficient traffic uses less energy with 802.11 than with S-MAC. Since nodes in the congested part of the network will often forward a significant amount of traffic, 802.11 is more-energy efficient than S-MAC for congested networks.

We compare CAR (and its improvements) to AODV and to an enhanced version of AODV that we implemented, referred to as **Priority Queue Based AODV** (AODV+PQ). AODV+PQ maintains two queues at each node. The first is a high priority queue. Messages in this queue are transmitted if present. The second queue is a low priority queue. When the high priority queue is empty, messages from this queue are transmitted. This policy provides absolute privilege to high priority data within a node. AODV+PQ is a simple generalization of priority queue based schemes such as the ones used in [10, 70, 95].

We also generated results for DSR [43] and Directed Diffusion [41], but do not present them here. In our environment of large multi-hop networks, DSR fails to route any high priority data successfully. DSR is intended to work over networks with a small number of hops, as reported in [26]. Similarly, Directed Diffusion was unable to route any high priority data successfully due to the large control overhead involved in the initial flooding that is required to set up the data paths. The One-Phase-Pull-Filter was used in the simulations and though it is expected to
route low priority packets successfully, our simulations showed that as the number of senders in the deployment was increased beyond 10, Directed Diffusion failed to route any data. As with DSR, Directed Diffusion is not intended for such applications. It was mainly designed to work in cases where the number of sinks and senders is small.

We do not compare our work to solutions that propose rate-limiting, back-pressure or throttling of senders as mitigation schemes (Fusion [40] and CODA [88]) because they treat all data equally and do not utilize the uncongested parts of the network to deliver low priority data. Also Fusion’s prioritized medium access scheme is designed for situations in which all data in the network is routed to the same sink. This might not always be true, e.g., 3 sinks are used in an experiment in CODA [88]. We also do not compare our schemes with SPEED [36] because its Stateless Non-deterministic Geographic Forwarding scheme is agnostic of data priority. If low priority data leads to an extended hotspot in the network, high priority flows that later enter the network will be forced to circumvent the hotspot. In contrast, we provide shortest paths for high priority flows and force low priority data to circumnavigate the conzone. In the worst case, SPEED will drop high priority packets due to unavailability of a suitable downstream node if all neighbors are sufficiently congested with low priority traffic. These are the scenarios in which we try to prevent degradation of service to high priority data.

A routing scheme has different aspects - route formation overhead, quality of routes, route maintenance overhead, etc. For this work, only the quality of routes is vital. The rest of the aspects of a routing scheme are not central to the problem of performance in congested networks. Since AODV uses shortest paths, we selected the same. Also, AODV is a widely used routing scheme and several other routing schemes are based on it. Hence like SPEED, we compare the performance of our schemes to AODV.

In our simulations, CAR builds the conzone from the critical area to the sink. Nodes are added to the conzone if they receive at least $\alpha = 2$ ToSink messages. For example, with transmission range of 130 m, the neighborhood size of a node away from the edge of the deployment is equal to 36. For a node at depth 3, $\beta_3$ is set to 0.018. For nodes with depth 2, $\beta_2$ is set to 0.027 and so on.

We now provide a high level comparison of CAR with AODV. Figs. 2.7(a)
and 2.7(b) depict the routing of low and high priority packets by AODV and CAR. CAR+ and CAR++ look similar to CAR as they all route only high priority traffic in the conzone. Lightly shaded edges denote low priority data while heavily shaded edges denote high priority data. The thickness of edges is directly proportional to the number of packets routed over them. A circle around a node denotes that the node has dropped high priority packets; the radius of the circle is directly proportional to the number of high priority packets dropped. Also, the larger nodes in Fig. 2.7(b) denote the nodes that belong to the discovered conzone.

From Fig. 2.7(a), we observe that AODV routes both low and high priority data delivery fraction

Figure 2.8. Varying transmission range, low priority data rate = 0.5 pps, high priority data rate = 30 pps.
data along the same paths and these paths may not be the shortest possible. As a result, many high priority packets are dropped at the critical area nodes themselves. Additionally in CAR, a conzone is formed (this can be observed in Fig. 2.7(b)). The set of shortest paths from the critical area to the sink route all high priority data while the rest of the network routes low priority data. Low priority data generated inside the conzone for CAR is effectively routed out using the minimum number of hops inside the conzone. It can be seen from Fig. 2.7(b) that CAR effectively performs differentiated and multipath routing and successfully routes routed-out or off-conzone generated low priority data around the conzone.

In the examples of Figs. 2.7(a) and 2.7(b), AODV routes only 5.7% of high priority data successfully while CAR delivers 96.2% of such data. The most prominent reason for AODV dropping packets, based on our analysis shown in Fig. 2.7(c), is that the MAC layer fails to route a packet after several retransmission attempts (MAC Callback). This is due to congestion which makes it difficult for a node to capture the channel to transmit data. Also, while AODV delivers 78% of low priority data, CAR delivers 89% of such data.

Note that in all simulations the queue size at each node was set to 1000 packets while the size of the packets was 50Bytes. Note that RAP [70] uses a queue size of 300 packets. Since the queues in nodes are almost always non-empty during congestion, the rate of “MAC Callback” errors will remain the same. AODV benefits more than CAR-based schemes with a large queue size because it drops more high priority packets due to buffer overflow (see Fig 2.7(c)). With a small queue, AODV will drop more data due to buffer overflow while CAR-based schemes will drop much less data.

### 2.4.2 Simulation Results

We analyze two aspects of the CAR-based schemes: feasibility and performance. For feasibility, we have analyzed the delays to form the routing network (HiNet for CAR, CAR+, and CAR++ ), discover the conzone and destroy it. The delays for HiNet formation, conzone discovery and destruction tend to decrease as the number of hops from the critical area to the sink decreases. HiNet formation delay stays under 11 seconds at maximum and decreases as the transmission range
increases. The conzone discovery and destruction delays were found to be less than 1 second. Since CAR is meant for static networks with comparatively long-lived high priority streams, these delays are small compared to the duration of high priority data flood. In low mobility scenarios the delay required for some nodes to refresh their parents and children will be less than the HiNet formation delay and the delay for nodes that move from off-conzone areas to the conzone to switch to conzone mode will be small as well.

In our simulations for CAR, this flood duration is set to 50 seconds. CAR is hence a feasible solution that can quickly adjust to different events that require the rediscovery of the conzone.

In the following we present results comparing the performance of our schemes for varying ranges, varying low priority data rates and varying high priority data rates.

2.4.2.1 Varying Transmission Range
In this group of simulations, the transmission ranges were varied between 90, 130, 170, and 210 m. As the transmission range increases, the number of hops from the edge of the network to the sink decreases from 6 to 3. The low priority (LP) data rate of each node, other than the critical area nodes and the sinks, was set to 0.5 packets/sec (pps) while the high priority (HP) data rate of critical area nodes was set to 30 pps. These simulations show the gains of CAR schemes as the node density of a deployment increases.

We first make some general observations on the behavioral differences between CAR and AODV+PQ. Priority queues provide better service to high priority data compared to AODV. However, because each node makes the best decision locally, such a scheme may not be able to provide better service globally. Consider the case in which a node has an empty high priority queue but a non-empty low priority queue. This node will start injecting low priority traffic into the network which, due to the shared medium, may degrade the service provided to high priority packets in nearby nodes. CAR and its enhancements, on the other hand, separate the traffic into two regions and hence eliminate most of the interference that can be caused by having both low and high priority traffic routed on the same paths.

Fig. 2.8(a) plots the fraction of high priority data delivered to the sink. As
the transmission range increases, the network becomes more congested and more collisions occur. As a result, the performance of AODV degrades severely and it routes less than 10% of high priority data successfully. On the other hand, AODV+PQ and CAR-based schemes route a higher fraction of the data, although CAR-based schemes route more high priority data than AODV+PQ for all ranges. At ranges larger than or equal to 130 m, CAR-based schemes route more than 90% of the data. We note that CAR++ routes more data than CAR+, which in turn routes more data than CAR.

Fig. 2.8(b) shows the fraction of low priority data routed successfully. Although our focus is to provide better service to high priority data in the presence of congestion, CAR also effectively utilizes the uncongested off-conzone nodes to prevent severe degradation of low priority data. Hence in addition to improving high priority delivery, CAR also enhances delivery of low priority traffic as the range increases. AODV delivery ratio decreases sharply as the range increases while AODV+PQ routes the highest percentage of low priority data. Note that, since AODV+PQ routes less high priority data (see Fig. 2.8(a)) and more low priority data than CAR-based schemes, it is clear that priority queue based schemes alone are not sufficient to provide better service to critical data.

Figure 2.9. Maximum node energy used: Varying transmission range, low priority data rate = 0.5 pps, high priority data rate = 30 pps.
CAR routes more low priority data than AODV as range increases since it prevents low priority data from entering the conzone and getting dropped. AODV + PQ routes more low priority data than CAR because it does not as aggressively degrade service to low priority data as CAR. At large ranges (i.e., in networks with few hops from sink to critical area) AODV+PQ routes more low priority data and approximately the same amount of high priority data as CAR. This is because, in CAR, congestion may occur in off-conzone areas as low priority data from the conzone is routed out into such areas. Note that CAR+ and CAR++ deliver less low priority data compared to CAR because they, by design, drop more such data.

![Graphs showing data delivery fraction and delay](image)

(a) HP data delivery fraction  
(b) LP data delivery fraction  
(c) HP data delivery delay

**Figure 2.10.** Varying low priority data rate, Transmission range = 130 m, High priority data rate = 30 pps.
Fig. 2.11. Varying high priority data rate, Transmission range = 130 m, Low priority data rate = 0.5 pps.

Fig. 2.8(c) shows that as the range increases, the average high priority data delivery delay for AODV increases while such delay for AODV+PQ and CAR-based schemes decreases. This is due to increasing congestion that AODV faces. Further, the jitter introduced by data forwarding is always less for CAR-based schemes as compared to AODV and AODV+PQ (not shown).

In our simulations, we observed that despite shorter paths being available, AODV and AODV+PQ do not necessarily route data along such paths. CAR, CAR+, and CAR++ schemes always find a shorter path. Routing along shortest paths has several implications including less overall network energy usage.
Fig. 2.9 shows the maximum energy used by any node in the deployment. This includes the energy used to route all possible traffic, both low and high priority. The energy used by AODV and AODV+PQ is more than that for CAR. CAR+ uses less energy and CAR++ uses the minimum energy among all the schemes.

2.4.2.2 Varying Low Priority Data Rate

In this set of simulations, the range is set to 130 m and the high priority data rate of each critical area node is set to 30 pps while the low priority data rate is varied. These simulations compare the performance of CAR schemes with AODV and AODV+PQ as the network has to contend with increasingly intense background low priority traffic.

As the low priority data rate increases, the fraction of high priority packets routed by AODV sharply falls to zero (see Fig. 2.10(a)). Although AODV+PQ performs better than AODV it still faces the same fate. In contrast, though the fraction of data routed successfully to the sink by the CAR-based schemes decreases, these schemes still route more than 60% of the data even when AODV and AODV+PQ do not route any data at all. Note that the curve for CAR++ overlaps the curve for MCAR in Figs. 2.10(a) and 2.10(c).

Also, the fraction of low priority data successfully routed by AODV drops (see Fig. 2.10(b)). For CAR it decreases from around 90% to 85%. For CAR+ and CAR++ this fraction stays almost constant.

The delays for AODV and CAR increase while they stay almost constant for CAR+, CAR++ (see Fig. 2.10(c)). The standard deviations of the delivery delays (not shown), exhibit higher variations for AODV and AODV+PQ. Such high variations correspond to larger jitter values, which are a problem for real-time data delivery.

Since AODV routes a very small fraction of high priority packets and a smaller fraction of low priority packets, the maximum energy used by AODV stays the same as the rate of low priority data varies (not shown). The energy consumed in the CAR-based schemes increases as the low priority data rate becomes larger. In all cases it is lower than AODV.
2.4.2.3 Varying High Priority Data Rate

In the final set of simulations, the high priority data rate was varied and the low priority data rate was fixed to 0.5 pps. The communication range was fixed at 130 m. These simulations compare the performance of CAR schemes with AODV and AODV+PQ for high priority data rates ranging from same as the low priority datarate of 0.5 pps to 30 pps.

As shown in Fig. 2.11(a), though the fraction of high priority data successfully delivered is 100% for all schemes when the high priority data rate is low, as it increases, the fraction of high priority data routed decreases faster for AODV and AODV+PQ than for CAR-based schemes. Note that the curve for CAR++ overlaps the curves for CAR+ in Figs. 2.11(a) and 2.11(c). Fig. 2.11(b) depicts the low priority data delivery fraction as the high priority data rate was varied. These results are consistent with those already presented.

Though average delivery delay increases for all schemes as the high priority data rate increases as shown in Fig. 2.11(c), AODV and AODV+PQ have higher variation in delivery delay as compared to CAR-based schemes (not shown). The CAR variants also provide even energy consumption as compared with AODV and AODV+PQ (not shown).

Note that CAR does not require the critical area to be on a horizontal line from the sink. The simulation setup used in this chapter uses such critical areas for the simplicity of presentation. CAR’s component algorithms can handle critical areas located anywhere in the network - not necessarily on a horizontal line with the high priority sink.

2.5 Summary and Future Work

In this chapter, we addressed data delivery issues in the presence of congestion in wireless sensor networks. We proposed Congestion Aware Routing (CAR) which is a differentiated routing protocol and uses data prioritization.

Our extensive simulations show that as compared to AODV and AODV+PQ, CAR and its variants increase the fraction of high priority data delivery, decrease delay and jitter for such delivery while using energy more uniformly in the deploy-
ment. CAR also routes an appreciable amount of low priority data in the presence of congestion.

Because of the low jitter rates and maintainable delay, CAR and its variants appear suitable to real-time data delivery. To ensure QoS for video streams, reactive dropping methods could be combined into the routing protocol. Our future work will look at the effectiveness of such techniques in sensor network environments. We will explore the interactions of differentiated routing and multiple conzones which may be overlapping or disjoint in CAR and its two enhancements. Finally, we will also explore the impact of different sizes and shapes of conzones on data delivery in the future.
Chapter 3

Channelization for Network Coding in Wireless Networks

Network coding is increasingly being investigated as an alternative to routing to increase throughput in packet networks. Like most data transfer schemes, the effectiveness of network coding may be limited by extreme congestion. When using network coding, these congested conditions are mitigated somewhat, but may still occur. We propose a selective channelization scheme in which links that experience congestion at a level that cannot be overcome by network coding are given reserved communication resources. This method has the following benefits. First, the algorithm proposed allows network coding full opportunity to overcome congestion before performing channelization, thus reducing the number of reserved resources used. Second, when triggered, the channelization of severely congested links greatly improves the end-to-end performance of flows that traverse the channelized link.

To determine the point at which channelization should be triggered, we perform a thorough analysis of potential coding gains in a network facing errors due to collisions, and determine the point at which network coding loses its effectiveness.

3.1 Introduction

Network coding, proposed by Ahlswede et al. [7], introduced the concept of combining packets before forwarding. By combining packets, less transmission capacity in the network is required to deliver data than with the traditional store-and-
forward approach used by routing schemes. Due to the inherent broadcast nature of the medium, network coding has been effectively applied to the problem of data delivery in wireless networks [48].

Network coding has been widely used to deliver multicast data. On a multicast link, called a hyperarc, network coding gains can arise for one of two reasons - multi-path and independent errors at destinations. If a node is a destination of multiple hyperarcs (e.g., in the canonical butterfly network example for network coding [33]), gains due to multi-path may arise. Otherwise, coding opportunities will arise out of independent errors in packet delivery to the destinations of the hyperarc. For example, consider a source that multicasts two packets $P_1$ and $P_2$ to nodes $N_1$ and $N_2$. Suppose that node $N_1$ receives only packet $P_1$ while node $N_2$ receives only packet $P_2$. The source may then multicast $P_1$ combined (using e.g., bitwise-XOR) with $P_2$ so that both nodes can recover their missing packet in a single transmission. This simple example shows the basic operation of network coding to overcome errors on a multicast flow.

Several factors impact the extent to which network coding gains are achieved when dealing with error recovery, such as link error rates (LER), frequency of collisions, maximum number of packets that can be coded, distribution of errors among destinations of a hyperarc, etc. While network coding greatly reduces the number of transmissions required to deliver data from a source to multiple destinations, its effectiveness is limited under conditions of extremely high errors, as with most data transfer methods. Though it still improves delivery efficiency when compared to routing under such conditions, it is unable to prevent the number of transmissions from increasing drastically.

In this chapter, we introduce a technique to assist network coding under conditions of extreme contention. Specifically, we address the delivery of multicast traffic using network coding in the presence of intense background traffic. When most errors occur due to collisions, channelization of a link will drastically reduce the error rate by removing errors due to collisions. We use this as motivation to introduce a selective channelization scheme used in concert with network coding. Network coding is relied upon to overcome errors caused by impaired links and collisions in the majority of the network operation; under periods of high collision rates, the affected links are channelized, i.e., assigned TDMA slots, to reduce
the probability of collision to zero, leaving network coding to contend only with remaining errors.

With selective channelization only severely congested hyperarcs in a flow are selected for channelization. As a result very few resources are required for reserved channels. Because network coding naturally reduces congestion, channelization is required at a higher threshold of offered load than when traditional routing protocols are used.

The contributions of this chapter are:

• Provide an in-depth analysis of potential network coding gains under congested conditions

• Design of selective channelization policies based on our analysis

• Simulation results to show gains brought by selective channelization with network coding

The rest of this chapter is organized as follows: §3.2 presents motivations for this work; §3.3 presents an analysis of expected network coding gains and sheds insight into the potential impact of channelization; §3.5 introduces our selective channelization policies; §3.6 lays out the details of our network coding scheme; results are discussed in §3.7; related work is presented in §3.8; §3.9 summarizes the chapter.

3.2 Motivation

Multicast traffic forms both the background and main traffic in this chapter. 802.11 does not protect multicast traffic by performing an RTS/CTS exchange before transmission. As a result, many collisions occur when the intensity of the background traffic is high. Channelization is one solution to address the issue of large-scale collisions. In this section we provide a brief motivation for selective channelization applied to network coding. We develop a formal model in the next section.
3.2.1 Channelization

In this chapter, we define channels as TDMA slots that are organized into frames which repeat cyclically. If a link is channelized, it has at least one slot in a frame reserved for its use among its two hop neighbors. The remaining slots that are not reserved may be used for transmissions by other links. The other links access these slots using a Slotted ALOHA-based protocol.

We explain this model with a simple example that is for illustrative purposes only. Assume a frame includes 10 slots. If a node is backlogged, \( i.e., \) it has data to send at the start of the frame, and it does not have a channelized link, it will send in each slot with a probability of 0.1 until it empties its queue. Now consider a case in which a node has a channelized link, \( i.e., \) it has a time-slot in each frame reserved for its exclusive use. The remaining nodes in the neighborhood will now contend for the other 9 slots in each frame.

The benefit for a link that is channelized is that it will not experience contention or collisions during its slot. Thus it is highly advantageous during periods of high congestion when Slotted ALOHA will collapse. A naive approach to solve high collision rates is to channelize all transmissions in a flow or network. This solution suffers from high inefficiencies because in a sizable network, the large number of channels required to allow all flows to be scheduled (in spite of spatial reuse) would be impractical from a standpoint of resources required. Even if ample resources are available, large frame sizes, \( i.e., \) frames comprised of many slots, would be required to ensure all links may have a channel. Such large frame sizes will lead to long delays even in a lightly loaded network.

To avoid the problem of large frame sizes we propose selective channelization in which only a portion of the links are channelized.

3.2.2 Channelization for Network Coding

Opportunity for network coding to assist in overcoming errors arises out of independent losses in the delivery of data to the destinations of a hyperarc. We consider losses due to link errors and collisions only in this chapter. Other reasons for packet loss can be dealt with by using the solutions proposed in prior literature.

While network coding improves the throughput of the network by reducing the
number of transmissions required [7, 55], under high error rates the number of transmissions required to deliver data will increase drastically. If most of these errors are due to collisions, channelization may be used to mitigate the loss and reduce error rates to the point where network coding regains its effectiveness; if the errors are due to link impairments, channelization will not help the situation.

Channelization may be applied to networks that use routing as well as those that use network coding. The benefit of combining network coding with channelization is that fewer links will require channelization due to the reduced transmissions required when using network coding. \textit{We aim to activate channelization only in circumstances when network performance is highly degraded due to collisions and cannot be overcome by network coding. Our main challenge is to determine when to trigger channelization to fully leverage the benefits of network coding.}

### 3.3 Gain Analysis

In this section we analyze the potential performance of a simple network using routing and network coding. This analysis, while based on some simplifying assumptions, sheds light on the potential gains of network coding, shows regions of operation where its effectiveness may be limited, and provides insight as to when channelization will be beneficial.

At high collision error rates, even with network coding, the number of transmissions for data delivery may reach unacceptable levels. Let the metric for the efficiency of the data delivery scheme (\textit{i.e.}, routing or network coding) be the number of transmissions required to deliver 100% of the data to all the destinations for all flows. The gain $g_h$ achieved over a single hyperarc $h$ is

$$g_h = \frac{r_h}{n_h} \forall h \in H$$  \hspace{1cm} (3.1)\]

where $r_h$ is the number of total transmissions without network coding, and $n_h$ is the total number of transmissions required with network coding. We next analyze the features of network coding gain and its interaction with channelization for one-hop and multi-hop networks.
3.3.1 One-Hop Network

Consider a one hop network with \( N \) destinations. Since we are interested in channelization, we focus on losses due to collisions only and not due to link errors. Let the number of packets that need to be delivered to all the destinations be \( B \). Note that network coding does not allow combining more packets than the number of destination nodes. Hence the following analysis assumes \( B \leq N \). Let the probability of a destination receiving a packet correctly be \( p_i \), \( i.e. \), the probability of a collision is \( p_b = 1 - p_i \). We assume that errors due to collisions at each destination are independent. While this is not realistic depending on network topology, the assumption is sufficient for allowing us to determine thresholds for triggering channelization.

After transmitting a group of packets, all patterns of data delivery to the destinations can be characterized by \( l \) - the number of packets delivered to all destinations, \( j \) - the number of packets that are delivered to none of the destinations, and \( k = (B - l - j) \) - number of packets that are delivered to some but not all destinations. The performances of routing and network coding differ when \( k > 1 \).

The expected number of transmissions (denoted by \( T(N, B)_{\text{routing}} \) for routing and \( T(N, B)_{\text{nc}} \) for network coding) required to deliver \( B \) packets to all \( N \) destinations is given by Eqn. 3.2.

\[
T(N, B)_{\text{scheme}} = B(p_i^N)^B + \sum_{l=0}^{B-1} \binom{B}{l} X_1 + \sum_{l=0}^{B-1} \binom{B}{l} X_2
\]  

\[X_1 = (B + T(N, m))(p_i^N)^l(p_b^N)^m\]  

\[X_2(l, j, k) = p_i^{Nl} \sum_{j=0}^{B-l-1} p_b^{Nj} [1 - p_i^N - p_b^N]^k \binom{B-l}{j} X_3(l, j, k)\]  

The difference in the value of \( T(N, B)_{\text{routing}} \) and \( T(N, B)_{\text{nc}} \) stems from the term \( X_3 \) as derived in §3.4. This term accounts for the reductions in transmissions when coding is possible.

When \( l = B \) all packets reach all destinations and when \( j = B \), none of the packets reach any destination. In these two cases network coding will not assist in
recovering from errors because there is no scope for coding. When \( k = 0 \), a subset of packets reaches all destinations, while the remaining packets do not reach any destination. When \( k = 1 \), one packet reaches some, but not all destinations. But this single packet cannot be combined with any other packet. Hence network coding will not provide a benefit for error recovery when \( k \leq 1 \). When \( k > 1 \) different destinations receive different sets of packets. In these cases it may be possible to combine packets and thus achieve network coding gains. Even when \( k > 1 \) coding is not always possible, and if it is, different gains arise from different combinations of received packets. The term \( X_3 \) quantifies these gains.

(a) Avg. Number of Transmissions Per Packet with Perfect Feedback: \( N=3, B=3, \) Packet Size = 512 Bytes

(b) Channelization Gain over Network Coding

Figure 3.1. Network Coding and Channelization
If the network is operating in a region where the term for $k > 1$ dominates, this means some destinations are receiving packets while others are not, and hence network coding may provide a benefit. More specifically, if this term is large, destinations have received different packets and combining packets for retransmissions may be possible. If the network is operating in a region where terms of Eqn. 3.2 for $l = B$ or $j = B$ dominate, the scope for network coding is small. Specifically, if $j = B$, there is little opportunity for network coding due to collisions and channelization gains are vital for the performance of the network.

To validate these equations we run an NS-2 [3] simulation with $N = 3$ and $B = 3$ to determine the number of packets required to deliver data for routing and network coding under different collision probabilities. We compare these results with those obtained by Eqn. 3.2 in Fig. 3.1(a). In the first set of simulations, we set the destinations so that their collisions are independent. In the second set, two of the destinations are assumed to share 50% of their neighbors with the third destination and hence their packet reception is correlated 50% of the time with the third destination. Our results in Fig. 3.1(a) indicate our model provides a good estimate of the expected number of packets to be transmitted. The values resulting from the model match the empirical results closely for low collision rates. At higher rates, the model diverges but is still within 30% for network coding with independent errors with a probability of collisions of 50% and within 15% for dependent errors when the probability of collisions is below 60%.

The formula for the gain (by using Eqn. 3.1 and $r_h = T(N, B)_{routing}$ and $n_h = T(N, B)_{nc}$) brought about by network coding is theoretical in nature. In practice, these gains are lower due to the limited number of packets that can be combined and the limited number of destination nodes of a multicast hyperarc. In addition, implicit to the concept of achieving such a theoretical gain is the presence of a perfect feedback channel to convey the failure of data delivery to any destination of the hyperarc. Without this feedback, the source of the hyperarc has to employ schemes like ACKs and heuristic based guessing schemes [48] to multicast the correct combination of packets. These imperfect feedback schemes further erode the gains of network coding.

Using Eqns. 3.1 and 3.2 for $N = 3$ and $B = 3$ we find that the network coding gain over routing increases as the error rate increases until an error rate of 0.21, and
then remains at a value of around 1.21 thereafter. Since our goal in this chapter is to present the benefits of combining channelization and network coding, we use topologies that can isolate the gains related to collisions during data transfer. The topologies we use to do this are not necessarily the best for network coding in general - e.g., network coding brings large gains when a node is at the junction of numerous flows. Therefore, the absolute gains of network coding over routing may be smaller than those reported in other papers. If optimal topologies are used, these gains will be restored and the benefits we show in this chapter will still be achieved.

As discussed above, initially as the error rate increases, network coding gains increase. If a source of errors is removed, e.g., if collisions are eliminated via channelization, network coding gains may also decrease because the scope for network coding has been reduced. However, this loss in coding gain may be offset by a gain due to channelization, i.e., fewer packets are required to be transmitted because channelization has removed errors. We should channelize only if doing so has an overall gain.

Let the error rate due to collisions be $e_1$ and that due to link errors be $e_2$ for a link. Hence the combined error rate is $1 - (1-e_1)(1-e_2) = e_1+e_2-e_1*e_2$. Let the gain function for network coding over routing be $g_{nc}(e)$ and that for channelization gain over network coding be $g_{ch}(e)$ where $e$ is the combined error rate. When the total error rate is $(e_1+e_2-e_1*e_2)$, the network coding gain without channelization is $g_{nc}(e_1 + e_2 - e_1 * e_2)$. Upon channelizing this link, errors due to collisions are removed. Hence network coding gains arise out of link errors only, i.e., network coding gain is now $g_{nc}(e_2)$. In this case the channelization gain is $g_{ch}(e_1)$. Since network coding and channelization gains are independent, the total gain for the link is their product - $g_{nc}(e_2) * g_{ch}(e_1)$.

For a larger overall gain due to channelization, we require

$$g_{nc}(e_1 + e_2 - e_1 * e_2) < g_{nc}(e_2) * g_{ch}(e_1)$$

$$g_{nc}(e_1 + e_2 - e_1 * e_2)/g_{nc}(e_2) < g_{ch}(e_1)$$

$$g_{ch}(e_1) > 1.24 \quad [\text{From Eqns. 3.1,3.2}]$$

$$e_1 > 0.14 \quad [\text{From Fig. 3.1(b)}]$$

Hence in our sample network, if channelization is triggered on a link that has
a collision probability of more than 0.14, it will always be beneficial. Our channelization policies (discussed in the next section) trigger channelization at higher estimated collision rates. Thus our approach to channelization will always increase the joint gain.

### 3.3.2 Multi-Hop Network

The gains of network coding in a multi-hop network are limited by the fraction of multicast hyperarcs (like COPE [48], unicast hyperarcs do not bring network coding gain in our scheme) and scope for network coding. Suppose that a flow involves \( m \) hops (i.e., hyperarcs) and that network coding brings gains \( g_i \) where \( i \in [1, m] \) over these hops. Let maximum gain over any hyperarc be \( g_{\text{max}} \). Let hop \( i \) transmit \( n_i \) packets when routing is used. The gain \( G \) for the end-to-end flow is

\[
G = \frac{\sum_{i=1}^{m} n_i}{\left( \sum_{i=1}^{m} \left( n_i/g_i \right) \right)}
\]

\[
\leq \frac{\sum_{i=1}^{m} n_i}{\left( \sum_{i=1}^{m} \left( n_i/g_{\text{max}} \right) \right)}
\]

\[
\leq g_{\text{max}}
\]

As a result, the total gain is not multiplicative (or even additive) of individual gains. In fact, the total gain is less than the maximum gain achieved over any hyperarc.

Now let the fraction of hyperarcs that bring gain be \( f \) and let \( n_i = n \) and \( g_i = g \ \forall i \in [1, m] \). Hence

\[
G = \frac{\sum_{i=1}^{m} n_i}{\left( \sum_{i=1}^{m} \left( n_i/g_i \right) \right)}
\]

\[
=(mn)/\left( \sum_{i=1}^{m} \left( n/g_i \right) \right)
\]

\[
=m/((1-f)m + \sum_{i=1}^{m} (1/g_i))
\]

\[
=m/((1-f)m + fm/g) = g/((1-f)g + f)
\]
As \( f \) increases, \( G \) increases. In fact, if a large fraction of the hyperarcs on a flow have little network coding gain, the end-to-end flow will also have low coding gains. As a result, the fraction of hyperarcs that bring gain is also an important contributor to total gain. Hence our approach to channelization helps the network in two respects. First, it increases the joint gain of hyperarcs on which network coding gains are already present (note that we never reduce joint gain over a hyperarc). Second, since network coding does not bring any gains for unicast links, channelization of such a link will bring pure channelization gains. As a result, channelization is a compelling scheme to improve the performance of networks employing network coding.

### 3.4 Network Coding Analysis

Here we analyze network coding gain over a single hyperarc based on the average number of transmitted packets. Let there be \( N \) destinations on this hyperarc on which \( B \) packets are to be delivered. For simplification, assume that all the destinations of the hyperarc have independent packet loss probability of \( p_b \). Let \( p_i \) denote the probability that the channel is idle (i.e., \( p_i = 1 - p_b \)). All possible outcomes of a set of transmissions can be classified into the following groups:

1. All \( B \) packets are received at all \( N \) destinations
2. Only \( l(< B) \) packets are received at all \( N \) destinations.
   (a) All \( m = (B - l) \) packets not received at any destination
   (b) Only \( j(< m) \) packets not received at any destination. Hence \( k = (m - j) \) packets are received by at least one but not all destinations.

Hence the average number of transmissions \( T(N, B) \) is defined as:

\[
T(N, B) = B(p_i^N)^B + \sum_{l=0, m=B-l}^{B-1} \binom{B}{l} X_1 + \sum_{l=0}^{B-1} \binom{B}{l} X_2 \tag{3.8}
\]

The second term of Eqn. 3.8 represents case 2(a) and \( X_1 \) is defined as:

\[
X_1 = (B + T(N, m))(p_i^N)^l(p_b^N)^m \tag{3.9}
\]
The third term of Eqn. 3.8 represents case 2(b) (when some of \( m = (B - l) \) packets are received by some of the destinations) and \( X_2 \) is defined as:

\[
X_2(l, j, k) = p_i^{N_l} \sum_{j=0}^{B-l-1} p_b^{N_j} [1 - p_i^N - p_b^N]^k \left( \frac{B-l}{j} \right) X_3(l, j, k) \tag{3.10}
\]

Eqn. 3.8 is a general definition of the average number of transmissions. The differences in the expected number of transmissions for routing and network coding arise in the expression for \( X_3 \). We now analyze expressions for \( X_3 \) for routing and network coding.

### 3.4.1 Routing

For routing, the source has to retransmit \( k \) packets which need to be delivered to only some of the destinations and hence:

\[
X_3^{\text{rout}}(l, j, k) = B + T(N, j) + \sum_{a=1}^{k} T(n_a, 1)
\]

\[
\approx B + T(N, j) + kT(N, 1)
\]

where \( n_a \) is the number of destinations that do not receive packet \( a \). For simplification, we assume that \( T(n_a, 1) = T(N, 1) \) \( \forall n_a \in [1, N - 1] \).

We can compute \( T(N, 1) \) (note that it is the same for routing and network coding) which is the average number of transmitted packets for a source to deliver a packet to \( N \) destinations as:

\[
T(N, 1) = p_i^N + [1 + T(N, 1)] p_b^N
\]

\[
+ \sum_{k=1}^{N-1} \binom{N}{k} p_i^{N-k} p_b^k (1 + T(k, 1))
\]

\[
= p_i^N + p_b^N + \sum_{k=1}^{N-1} \binom{N}{k} p_i^{N-k} p_b^k (1 + T(k, 1))
\]

\[
1 - p_b^N
\]

### 3.4.2 Network Coding

With \( N \) destinations and \( B \) packets, the delivery of packets can result in any of \( 2^{NB} \) possibilities. Also, the number of cases when \( B - l \) packets are not received by any
node is $\sum_{l=1}^{B-1} \binom{B}{l} = 2^B - 2$ while the number of cases when some of $B - l$ packets are received by some destinations is

$$Y_b = 2^{NB} - \left( \sum_{l=1}^{B-1} \binom{B}{l} + 2 \right) = 2^{NB} - 2^B.$$  

For ease of analysis, we use a network coding policy in which the source tries to find a network coding packet that codes the maximum number of lost packets. Also, if different packets are lost at the same destination, they cannot be combined. We designate the first destination as a basis. For $i \in [1, k - 1]$ lost packets at the basis user, one encoded packet and $i - 1$ native packets need to be delivered. The encoded packet is generated by combining a packet lost at the basis user with other $k - i$ packets that the basis user received but other users might not have received. The number of cases in which we can apply network coding with the policy described above, $Y_i$, is defined as

$$Y_i = \binom{k}{i} (2^{N-1} - 2)^{-1} \sum_{x_1=0}^{N-2} \left( \begin{array}{c} N-1-x_1 \\ x_1 \end{array} \right) \prod_{s=2}^{N-1-x_1} \left( \sum_{x_s=1}^{x_1} \left( \begin{array}{c} N-1-x_1-x_s \\ x_s \end{array} \right) \right) \right]$$

(3.12)

For each case, the number of packets that the source retransmits is $(i - 1)T(N, 1) + T(N, 1) = iT(N, 1)$. Also, the total number of cases that we can apply network coding is defined as $Y_{nc} = \sum_{p=1}^{k-1} Y_p$. Hence, the number of ways in which we cannot apply network coding is $Y_b - Y_{nc}$. In these cases, the source should retransmit $k$ packets to the destinations as routing. The number of transmissions is $kT(N, 1)$ based on earlier-mentioned assumption $T(ga, 1) \approx T(N, 1)$.

For $k$ packets that at least one, but not all, destination receives, all possibilities can be classified into two groups. The first group, defined by $X_{no-nc}$, consists of the cases that the source cannot apply network coding.

$$X_{no-nc} = (1 - \frac{Y_{nc}}{Y_b})(B + T(N,j) + kT(N,1)).$$  

(3.13)

The other group, denoted by $X_{nc}$, is for the source to apply network coding.

$$X_{nc} = \sum_{i=1}^{k-1} \frac{Y_i}{Y_b} (B + T(N,j) + iT(N,1))$$

(3.14)

Based on these two groups, we can evaluate $X_3$ as:

$$X_{3nc}^{nc}(l,j,k) = (X_{no-nc} + X_{nc})$$

(3.15)
3.5 Channelization

We assume an underlying TDMA MAC layer and hence channelization refers to the allocation of slots (\textit{i.e.}, channels) in a frame to hyperarcs. We use the canonical two-hop reuse constraint of channelization [74].

Our channelization is per flow, \textit{i.e.}, the same physical hyperarc may be assigned multiple channels - one for each flow that utilizes it, thereby isolating the performance of one flow from another. Suppose there are $s$ slots in a TDMA frame. Out of these slots, $s_{\text{chan}}$ slots are marked for channelization (\textit{i.e.}, reservation) while the rest of the slots are used by unchannelized hyperarcs to contend for transmission using a Slotted ALOHA-based mechanism. These contention channels are also used to carry control packets.

In the following subsections, we describe our algorithms for deciding when to channelize and how to channelize.

3.5.1 When To Channelize

As discussed, one benefit of applying network coding is to reduce the number of transmissions required to overcome errors. However, even with network coding the number of transmissions required to deliver data may rise dramatically due to high error rates. To address this, we propose to integrate network coding with channelization. We propose two channelization schemes of varying complexity and adaptation to network coding. These methods adhere to the following philosophy. A node determines if the number of transmissions required to successfully deliver packets exceeds a threshold. If it does, the node channelizes the link, \textit{i.e.}, we trigger channelization if the threshold is exceeded by errors due to collisions.

- \textit{Count Based Channelization:} In this first scheme a node simply maintains an average of the number of transmissions required to successfully deliver a packet. When this average becomes larger than a threshold, this hyperarc is channelized. This approach is accurate in determining the cost of transmitting a packet while accounting for network coding, but does not separate errors due to collisions from those due to link errors. Therefore, it may channelize links to no avail in some circumstances.
• **Model-Based Channelization using Network Coding Estimate:** In this method we attempt to isolate the impact of collisions on the number of transmissions. We do this by setting a target threshold based on $T(N,B)_{nc}$ from Eqn. 3.2. Based on this equation we determine a target $p_b$ at which the threshold is exceeded. Nodes gather information, as described below, from their neighbors to estimate $p_b$, and if this value is above the target $p_b$, the link is channelized. This method isolates collisions from link errors and accounts for the benefits of network coding. Thus, if the collision rate is such that network coding is still operating effectively, this method will not trigger channelization.

**How to estimate $p_b$ of neighbors:** Each node appends a flag in each transmitted packet indicating whether it has more packets waiting to be transmitted in the contention slots. Based on such overheard information from all neighbors, a node can calculate its own collision probability, $p_b$. Let $n_{cont}$ be the number of neighbors that have data to send in one of the $s_{cont}$ channels. Each node maintains a weighted moving average of $n_{cont}$ by counting the number of neighbors that transmitted packet with the aforementioned flag set. $p_b$ can be estimated as $1 - [1 - p_{send}]^{n_{cont}}$ where $p_{send}$ is the probability of a backlogged node transmitting in a contention slot. The local $p_b$ in each node is maintained as a weighted moving average as well. This $p_b$ is also appended to all transmitted packets. Using overhearing, nodes gather the $p_b$ of all neighbors. This value of $p_b$ will tend to be aggressive because a node does not contend with itself but the information received from its neighbors includes its own transmissions.

**Comparison of Channelization Schemes:** To understand the behavior of these two channelization schemes we perform two simulations in NS-2 [3]. In both cases we have a single hop multicast flow from a source to three destinations. For comparison, we apply methods similar to that described above onto a network that uses traditional routing. The count based scheme is easily extended for routing. For the model based scheme, we determine a target $p_b$ using $T(N,B)_{routing}$ from Eqn. 3.2 at which the threshold is reached.
We use this scheme with routing to show the benefits of combining network coding and channelization.

For a fair comparison of the channelization policies, we set the threshold as $M$ transmissions per packet. For the model-based policies if the value of $T(N, B)$ is larger than $M * B$, channelization is invoked. We use $M = 3.0$ in these simulations. For this threshold, Fig. 3.1(a) shows that the $p_b$ at which model based channelization schemes trigger are 0.57 and 0.71 for routing and network coding, respectively. This shows that the network can tolerate more load (higher collisions) using network coding than with routing alone.
In the first simulation, there are link errors but no collisions. Fig. 3.2(a) shows that as the link error rate increases, the number of transmissions required to deliver the same amount of data also increases. The vertical arrows show the points at which the count algorithms trigger the channelization of the hyperarc due to the crossing of the threshold. The results show that this channelization does not reduce the number of transmissions because the errors are not due to collisions (curves for count and model based schemes for both routing and network coding overlap in Fig. 3.2(a)). Note that neither of the model-based schemes trigger. This shows that the count based channelization algorithm has false positives in the presence of sufficient link errors and wastes resources by channelizing when there is no need.

In the second simulation, there are no link errors and losses occur only due to collisions. As the probability of collision increases, both count and model based algorithms eventually trigger channelization and improve performance as shown in Fig. 3.2(b). Both routing schemes channelize before any scheme employing network coding because, without the reduction in transmissions achieved through network coding, the target collision probability is reached at a lower contention level. The results for both the count and model-based algorithm using $T(N, B)_{nc}$ indicate they trigger at higher background loads because of the reduction in transmissions due to network coding. Note that in this example, they trigger at virtually the same load. In this case, the count algorithm does not result in false positives as all errors are due to collisions.

These results indicate that under conditions in which errors occur due to both link impairments and collisions, the model-based algorithm using $T(N, B)_{nc}$ as its estimate will perform best. The count algorithm is likely to be too aggressive in cases of high link errors.

### 3.5.2 How To Channelize

There are several existing mechanisms to channelize links. We define an efficient method below that fits well with our selective channelization algorithm. This method is based on similar assumptions as other previously proposed algorithms use for similar purposes (like Bao et al. [14] and Tang et al. [85]).

Since our channelization scheme is per-flow, we append channel information to
each entry in the multicast tables of all the nodes. When routes are destroyed, channelization information is lost, and channels are implicitly unchannelized.

If the hyperarc on which a packet is to be forwarded is not channelized and the selective channelization policy indicates it should be channelized, the node will attempt to channelize the hyperarc. Hence it needs to be aware of the channels used by its two hop neighbors. Since explicit messages to gather two-hop channel utilization information will further burden a network, an implicit mechanism is used. To each data packet that a node broadcasts, it appends bitmaps of the channels that it uses and that its neighbors use (based on best available information). These bitmaps keep the overhead of such piggybacked transmission to a minimum. Each node keeps track of such 1-hop and 2-hop information from all of its neighbors. When a node needs to channelize an outgoing hyperarc, it uses this locally available information to rule out the channels that are being used in the neighborhood. It then picks a random channel from among the available reservation channels.

A drawback of using implicit information is that, at low loads, the information may be stale. Even with fresh neighborhood information, reservation conflicts may arise. If a conflict is detected, a node retains the channel for the hyperarc with a probability of 0.5. If the channel is not retained, the hyperarc is assigned another available reservation channel, based again on the locally stored information. If no reservation channel is available, contention slots are used to transmit packets meant for this hyperarc. This resolution mechanism does not impose any overhead of explicit messages.

### 3.6 Network Coding Scheme

There are several proposed ways in which network coding can be implemented. The network coding scheme that we employ uses most of the salient features of COPE [48]. COPE uses a feedback mechanism built using asynchronous ACKs such that a source, in ideal circumstances, knows which destinations have received which packets and can perform packet combination intelligently. This feedback mechanism is hop-by-hop. The quality of this feedback channel is vital for the performance of network coding.
The focus of our work is on channelization for network coding and not network coding itself. Hence we assume the presence of a perfect feedback channel, i.e., a node is always aware of whether a downstream node received its transmission or not. In fact, channelization aids the design of the feedback scheme. Nodes upstream of the source of a channelized hyperarc will be able to overhear all transmissions that may have been otherwise lost due to contention errors. While this will not make the feedback channel perfect, it will make it more reliable.

Briefly, our network coding scheme adopts the following approach. Each node stores all data packets for all flows that it forwards. Nodes also decode all packets that they receive. Hence we perform encoding as well as decoding on a per-hop basis. A source of a hyperarc estimates which destinations of the hyperarc have received which packets based on the feedback. Any packet that has been received by all destinations of the hyperarc need not be retransmitted in any combination. Among the remaining packets, multiple packets can be combined only if the destinations that do not have these packets have no common nodes (this policy is from COPE [48]). As a result, combining more packets than the number of destinations of a hyperarc is not possible. Packets are combined using XORs as is widely prevalent in network coding literature.

3.7 Results

The results presented here were obtained using NS-2.30 [3]. An 802.11 air interface operating at 11Mbps was used for the non-TDMA results. NS-2 provides a single-hop TDMA-MAC that uses the 802.11 physical layer module. We extended this MAC layer for multi-hop networks for the TDMA results. The COPE-like network coding scheme presented in Section 3.6 was also implemented in NS-2.

We simulate multicast flows from a source to multiple destinations across multiple hops. Our measured multicast flow has seven destinations. The source transmits to three destinations three hops away and two destinations each one hop and two hops away. Hence, there are three hyperarcs in this flow. Because the source and intermediate nodes have at most three downstream nodes, we set the limit on the number of packets that can be combined to three as network coding does not combine more packets than the number of downstream nodes. Also, the number
of contention and reservation channels was each set to 4 in these simulations.

In the first simulation, the hyperarc with the flow source as its root is subjected to severe contention, the second hyperarc is under mild contention, and the third hyperarc had little contention. Throughout the simulation, the offered load is kept constant. In these simulations, the threshold for triggering channelization is set to \( M = 2.0 \) transmissions per packet. These results are an average of 5 runs.

![Graph showing number of transmissions and channelizations vs. link error rate](image)

**(a) Number of Transmissions**

**(b) Number of Channelizations**

**Figure 3.3.** Varying Link Error Rate: Number of Flows = 1, Number of Packets = 50, Packet Size = 512 Bytes

Fig. 3.3(a) shows that as the link error rate increases, the number of transmis-
sions increases for all schemes. The network coding based schemes using TDMA outperform routing and network coding over 802.11. Even at low error rates, the high contention hyperarc is channelized by both the model based and count based schemes. At sufficiently high link error rates, the other two hyperarcs are also channelized by the count based scheme (Fig. 3.3(b)). This illustrates the drawback of false positives in the count based scheme. On the other hand, the model based scheme successfully realizes that only one of the links has high contention error and channelizes only this link for all link error rates. From Fig. 3.3(a), we observe that the performance of both the count based and model based schemes is almost the same even though the count based scheme uses more channels.

In the next set of simulations, the number of links under high contention were varied. The link error rate was set to 0.4 throughout these simulations. The channelization trigger threshold was set to $M = 3.0$ transmissions per packet.

Results in Figs. 3.4(a) and 3.4(b) were averaged over 5 runs. As the number of hyperarcs under high contention error is increased, both routing and network coding over unchannelized TDMA face performance degradation (Fig. 3.4(a)). Both the count based and model based schemes still maintain their performance level in spite of the increasing number of contention links.

When there are no links under high contention, and just one link is under mild contention, the count based channelization scheme channelizes links (Fig. 3.4(b)), indicating false positives. As the number of links under contention increases, the number of channelized links increases accordingly for the model based scheme. The count based scheme is always more aggressive in terms of channelization even though both schemes result in a similar level of performance.

In summary, the model based scheme results in the same performance in terms of number of transmissions as the count based scheme. In the presence of link errors, the count based scheme is prone to false positives. The model based channelization scheme effectively detects the need for channelization, thereby preventing performance degradation while using fewer channels at the same time.
Figure 3.4. Varying contention level: Number of Flows = 1, Number of Packets = 50, Packet Size = 512 Bytes, Link Error Rate = 0.4

3.8 Related Work

Various approaches have been adopted to increase throughput of wireless networks. Ahlswede et al. [7] proposed the concept of network coding and have shown that the multicast capacity of a network cannot be achieved by traditional store-and-forward approach of routing. This capacity can be realized by using network coding. Linear network coding has been shown to be an effective way to approach this capacity [55, 68]. Random linear network coding proposed by Ho et al. [38]
approaches this multicast capacity as the code length increases.

Network coding is increasingly being adopted to address problems of bandwidth constraints in different areas. Gkantsidis et al. [34] proposed a scheme which uses network coding to enhance delivery of blocks of a file in a peer-to-peer network. Research has also been conducted to bring network coding in the realm of practical use by Chou et al. [24] among others. Katti et al. [48] proposed COPE to leverage scope for combining packets in wireless networks by using opportunistic policies. Katti et al. [47] and Zhang et al. [96] have also proposed to use network coding at the physical layer to avoid dropping packets due to collisions in the medium at a receiver.

Another way to improve network capacity is through scheduling and spatial reuse of channels. Specially at high contention levels for medium access, scheduling has been an important mechanism to avoid large-scale contention. Since the number of channels at disposal is limited, spatial reuse of multiple available channels has been intensively researched. But optimal link or broadcast scheduling in wireless networks has been known to be an NP-complete problem [13, 30]. Works such as [52, 85] and [86] have investigated this spectrum of research. Distributed schemes for link and node scheduling were also proposed in Bao et al. [14]. Note that in this chapter we have considered link channelization as compared to broadcast or node channelization.

To the best of our knowledge, our work is the first to foray into channelization schemes specific to network coding.

### 3.9 Summary

Even when network coding is employed, under highly congested circumstances the number of transmissions required to successfully deliver data may become prohibitive. We provide an analysis of the gains to expect out of network coding in a multicast setup. Based on this analysis, we propose a selective channelization scheme to recognize the opportunity to use channelization to prevent performance degradation of network coding in the face of high contention errors. Our results show that our model based channelization trigger criteria is an effective scheme to use fewer channels while maintaining the performance of network coding.
In this chapter we address congestion of multicast traffic in multi-hop wireless networks through a combination of network coding and resource reservation. Network coding reduces the number of transmissions required in multicast flows thus allowing a network to approach its multicast capacity. In addition, it efficiently repairs errors in multicast flows by combining packets lost at different destinations. However, under conditions of extremely high congestion the repair capability of network coding is seriously degraded. In this chapter we propose cooperative channelization, in which portions of the transmission media are allocated to links that are congested to the point where network coding cannot efficiently repair loss. A health metric is proposed to allow comparison of need for channelization of different multicast links. Cooperative channelization considers the impact of channelization on overall network performance before resource reservation is triggered. Our results show that cooperative channelization improves overall network performance while being well suited for wireless networks using network coding.
4.1 Introduction

Congestion in multi-hop wireless networks is a significant problem and manifests itself in several forms - large scale collisions, buffer overflows, large delivery delays, etc. We consider congestion of multicast traffic in such networks in this chapter. In congested wireless networks the performance of multicast flows degrades quickly. A packet lost at any of the next hop nodes in a multicast tree requires retransmission.

One approach to alleviate congestion is to design medium access control (MAC) protocols that effectively avoid collisions by backing off during congested periods. These protocols typically lead to large backlogs, and hence delays, in nodes. Another approach is to increase system bandwidth by using multiple radios, frequencies, multiple data rates, etc. to postpone congestion.

Another approach is network coding [7] which applies naturally to multicast data and has been shown to approach the multicast capacity of networks. In addition, it can typically recover from errors on multicast links, termed hyperarcs, using fewer (re)transmissions than routing. For example, collisions at destinations of a hyperarc may lead to successful reception of different packets at different destinations. We refer to this phenomena as discrepancy errors. Network coding may combine the different lost packets into fewer transmissions than if routing is used to recover from the errors. This decreases congestion in the network, thereby improving overall performance. However, under sufficient congestion even network coding cannot keep the number of transmissions to deliver a packet below an acceptable threshold.

Another approach to avoid collisions is channelization i.e., scheduling transmissions in the network while using a TDMA MAC. Exhaustive channelization i.e., scheduling all transmissions, will require prohibitively large frame sizes in spite of medium reuse due to requirement of multiple slots by each node in a neighborhood. This will result in large delays even at low loads.

In this chapter, we propose a resource reservation scheme called cooperative channelization to determine need for channelization of hyperarcs in congested multi-hop wireless networks that employ network coding. Since performance degradation due to congestion arises out of competition for channel access among nodes in a neighborhood, cooperative channelization decisions are made while taking the
neighborhood into account. Cooperative channelization considers losses due to collisions as well as delays and queue occupancy as factors for channelization. In contrast, selective channelization [56] does not consider delays or queue occupancy and is a selfish scheme that determines need for channelization of hyperarcs in isolation.

The contributions of this chapter are:

- Definition of a composite health metric that includes factors for the impact of collisions, queue occupancy, network coding opportunity, and effect of channelization on next-hop downstream hyperarcs;

- Cooperative channelization algorithm that efficiently distributes network resources to improve the overall system performance;

- An evaluation of the metrics and algorithms, including justification for each component in the composite health metric.

Our results show that, in general, channelization combined with network coding during congestion improves performance of the affected hyperarcs. When combined with network coding, cooperative channelization proposed in this chapter decreases delivery delay by up to 20% compared to the selective channelization algorithm with network coding. Finally, the TDMA MAC protocol used is well suited for networks using network coding. It allows aggressive transmission of packets, that while leading to more collisions, leverages the repair capabilities of network coding to improve network performance.

The rest of this chapter is organized as follows: §4.2 presents the network model; §4.3 presents components of the health metric. The cooperative channelization algorithm is presented in §4.4. Results are discussed in §4.5; related work is presented in §4.6. §4.7 summarizes the chapter.

### 4.2 Network Model

We use a Slotted ALOHA-like MAC with probabilistic transmission in this chapter. We refer to it as TDMA MAC for simplicity. Frames are divided into multiple slots. Resource reservation or channelization corresponds to assigning slots in this TDMA
frame to nodes for exclusive access. Nodes may probabilistically transmit in slots that have not been channelized in a neighborhood. This probability is based on the load on the medium in the neighborhood.

In this section, we first present our TDMA MAC protocol in §4.2.1. Nodes are assigned a transmission probability assignment for using it in §4.2.2. A brief description of the network coding scheme used is presented in §4.2.3 and evaluation metrics used are explained in §4.2.4. A summary of the frequently used symbols used in this chapter is presented in Table 4.1. Note that detailed examples of channelization are provided in the next section (cf. §4.3.2 and §4.3.4).

### 4.2.1 Medium Access Control

A TDMA frame consists of $s_{control}$ control slots and $s_{data}$ data slots. In the rest of this chapter, all references to slots stand for data slots unless specified otherwise. We define a data slot that is reserved for a specific flow as being channelized. Nodes transmit in unchannelized slots based on a transmission probability (described in subsection 4.2.2).

Let the set of 1-hop and 2-hop neighbors of node $v$ be represented by $\gamma_1(v)$ and $\gamma_2(v)$, respectively. Data slots available in the network are denoted by elements of set $C = \{1, 2, \ldots, s\}$. Slots free to be used by node $v$ for contention access are denoted by $C_v$. $L_v$ is the set of slots reserved, i.e., channelized, for flows of node $v$. $O_v$ is the set of slots that are unavailable to node $v$ because they are reserved for flows of other nodes in the 2-hop neighborhood. Hence

$$C_v \cup L_v \cup O_v = C, \ \forall v$$  \hspace{1cm} (4.1)

Considering spatial reuse constraint of channels, for any two-hop neighbors $v$ and $w$, we have

$$L_v \subseteq O_w, \ \forall v \in \gamma_2(w)$$  \hspace{1cm} (4.2)

To begin with, all slots are unreserved. When channelization occurs (as discussed later), we use the channelization scheme proposed in [56] to reserve slot(s) for the relevant flow. Note that a slot can be reserved in one two-hop neighborhood and used for contention based access in another two-hop neighborhood. To ensure
the availability of at least one data slot for contention access for each node, the first data slot in $C$ is never considered for channelization.

### 4.2.2 Transmission Probability Assignment

A node $v$ is assigned a transmission probability $q_v$, i.e., the probability that it transmits in an available unchannelized slot. We set $q_v$ in a node dynamically based on the current load in the neighborhood.

A node may forward data for multiple flows. To isolate the impact of flows from one another in a node we use per-flow queues within nodes and per-flow channelization (i.e., in a node, a slot assigned for a flow will not be used for another flow). The data rate of flows is specified in terms of number of slots in a TDMA frame required to keep the waiting queue empty (assuming this is the only flow) i.e., packets-per-TDMA-frame (ppf).

Let there be $f$ flows in the network. Let flow $i$ require $d_v^i$ slots in node $v$ and be assigned $e_v^i$ channelized slots (If flow $i$ does not traverse node $v$, $d_v^i = 0$). $e_v^i \leq \lceil d_v^i \rceil$ ($d_v^i$ need not be an integer). Flows may be completely channelized, not channelized at all, or partially channelized. For fully channelized flows, $e_v^i = \lceil d_v^i \rceil$. For unchannelized flows, $e_v^i = 0$. For partially channelized flows $e_v^i < \lceil d_v^i \rceil$, and the unchannelized load is

$$
\chi_v^i = \begin{cases} 
    d_v^i - e_v^i, & \text{if } d_v^i \geq e_v^i \\
    0, & \text{otherwise}
\end{cases}
$$

(4.3)

Packets for a flow in a node may be transmitted in slots that have been reserved for it or in any of the available contention slots. For example, if a flow in node $v$ has $d = 4$ ppf and is assigned $e = 1$ slot only, $1/4$-th of the packets for this flow are marked as “channelized” and transmitted in the channelized slot while the rest are marked as “unchannelized” and contend for transmission in the slots of $C_v$.

We use a transmission probability assignment scheme that provides transmission probability proportional to the traffic load in the node. Nodes maintain the set of unchannelized slots in their 2-hop neighborhood (i.e., $C_v$). Since the set of slots available for contention access at a node and its neighbors may be different, all nodes append $h_v = \sum_{i=1}^f \chi_v^i/|C_v|$, the load on each unchannelized slot in the
Table 4.1. Notation ($A^i_v$ denotes $A$ in node $v$ for flow $i$)

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>$C_v$</td>
<td>Contention slots available to node $v$</td>
</tr>
<tr>
<td>$\gamma_1(v)$</td>
<td>1-hop neighbors of node $v$</td>
</tr>
<tr>
<td>$\gamma_2(v)$</td>
<td>2-hop neighbors of node $v$</td>
</tr>
<tr>
<td>$d^i_v$</td>
<td>Load</td>
</tr>
<tr>
<td>$e^i_v$</td>
<td>Number of slots assigned</td>
</tr>
<tr>
<td>$h_v$</td>
<td>Load on an unchannelized slot due to node $v$</td>
</tr>
<tr>
<td>$q_v$</td>
<td>Transmission probability in contention slots for node $v$</td>
</tr>
<tr>
<td>$q^i_v$</td>
<td>Transmission probability in contention slots in node $v$ for flow $i$</td>
</tr>
<tr>
<td>$q_{min}$</td>
<td>Minimum sum of transmission probabilities in a 2-hop neighborhood</td>
</tr>
<tr>
<td>$q_{max}$</td>
<td>Maximum sum of transmission probabilities in a 2-hop neighborhood</td>
</tr>
<tr>
<td>$p^i_u$</td>
<td>Probability of node $u$ receiving a transmission from $v$ in a contention slot</td>
</tr>
<tr>
<td>$X^i_v$</td>
<td>Expected no. of transmissions with network coding due to collisions</td>
</tr>
<tr>
<td>$R^i_v$</td>
<td>Expected no. of media access attempts</td>
</tr>
<tr>
<td>$G^i_v$</td>
<td>Network coding opportunity estimate</td>
</tr>
<tr>
<td>$J^i_v$</td>
<td>Impact on next-hop downstream hyperarcs</td>
</tr>
<tr>
<td>$H^i_v$</td>
<td>Health</td>
</tr>
<tr>
<td>$\eta^i_v$</td>
<td>Number of next hop nodes</td>
</tr>
<tr>
<td>$p^i_v$</td>
<td>Number of next-hop downstream hyperarcs</td>
</tr>
</tbody>
</table>

node, to each transmitted packet. Each node collects $h_{max}$, the maximum $h$ among all nodes in its 2-hop neighborhood. This requires each node to append its $h_v$ and maximum $h_u$ where $u$ is a 1-hop neighbor to outgoing packets. It then computes $q_v$ as

$$q_v = \left[ q_{min} + h_v \cdot (q_{max} - q_{min}) / h_{max} \right] / |\gamma_2(v)|$$  \hspace{1cm} (4.4)

where $q_{min}$ ($q_{max}$) is the minimum (maximum) sum of transmission probabilities in a 2-hop neighborhood. Hence $q_{min}$ and $q_{max}$ are parameters of this transmission probability assignment algorithm. The minimum bound is used to prevent a node from receiving too few slots in a congested area. The maximum bound is used to
avoid allowing a node to transmit in too many slots and therefore make it difficult for new flows to enter the network.

The distribution of $q_v$ among flows traversing node $v$ is proportional to the unchannelized load for the flow (i.e., $\chi_v^i$). Hence $q_v$ is divided among flows in a node as

$$q_v^i = \chi_v^i \cdot q_v / (h_v \cdot |C_v|)$$

(4.5)

where $q_v^i$ is the transmission probability for flow $i$ in node $v$. So the total transmission opportunity for flow $i$ in node $v$ is $(e_v^i + q_v^i \cdot |C_v|)$ ppf.

### 4.2.3 Network Coding

Our network coding scheme is based on COPE [48]. COPE employs “opportunistic listening” and probabilistic packet delivery information based on ETX [27] to determine packets delivered to neighboring nodes. “Opportunistic coding” uses this information to transmit packets that can be decoded by all neighbors. COPE operates in a hop-by-hop manner and hence only considers coding opportunities in a one-hop neighborhood. We use COPE’s underlying theme of keeping track of packets missing at next hop nodes and transmitting coded packets to address gaps in delivered packet sequences efficiently. Like COPE, we combine packets using XOR.

We only use intra-flow coding opportunities. The work presented in this chapter will be extended for inter-flow coding in the future. We assume the presence of a perfect feedback system that enables nodes to keep track of packets delivered to next hop nodes - e.g., by overhearing transmissions of downstream nodes. While this is a simplified model, it allows us to concentrate on the performance of network coding.

### 4.2.4 Metrics

Congestion on the wireless medium will manifest itself in two ways: excessive collisions or backoff.

A highly aggressive transmission strategy may result in an increased number of
collisions which will increase delivery delay. We capture this effect with the number of transmissions required to deliver a packet. This metric is insufficient by itself because if an aggressive backoff mechanism is used, few collisions will occur, but the medium will not be efficiently utilized and packets will encounter large delays.

Second, due to large backoff, or if the arrival rate of packets into a node is higher than the resources available to forward them, buffers may fill up, resulting in increased delay. We use number of backoffs per packet as a metric to capture this phenomenon. A backoff is defined as a node desiring to transmit a packet but being precluded from doing so because there are either no slots available (e.g., when traffic load is more than available transmission opportunity in the TDMA frame), or the node’s backoff mechanism has been triggered.

The delivery delay, and hence throughput, observed in a system is dependent on both - number of transmissions to deliver a packet, and number of backoffs. Normalized delay, our third metric, combines these two factors and is representative of delivery delay (and inverse of throughput) in the system. We use this as the paramount metric to represent delivery delay in the network. Note that all these metrics are per-packet.

We use UDP traffic to emulate constant load and implement a retransmission layer over it. Each flow transmits packets periodically based on the load rate. The retransmission layer employs network coding to determine if a new packet needs to be transmitted or a coded packet can be transmitted. System metrics like delay and application-level throughput for a flow depend on its rate and normalized delay per packet. When different UDP flows transmit at different rates, combining them into a metric requires eliminating rates to maintain fairness. Accordingly, we use number of transmissions required, backoffs, and normalized delay as the performance metrics.

4.3 Hyperarc Health

Network coding alone fails to prevent performance degradation of flows when subjected to large scale losses due to collisions, link errors, and large delays due to inability to access media, etc. Losses due to collisions, and delays due to backoffs can be addressed by adding channelization.
We propose a health metric to classify hyperarcs according to their need for channelization. We start with the expected number of transmissions per packet due to collisions and add a factor that enables accounting for backoffs which includes a measure of inability to access the media due to either backoff or overload.

Given that there may be competition to reserve slots, i.e., there may be fewer slots available to channelize than there are flows with degraded performance in a neighborhood, we add two additional parameters to skew our choice of hyperarcs to channelize. First, if a hyperarc has a high potential to benefit from network coding under varying conditions, we decrease its likelihood of being channelized relative to hyperarcs that will likely not benefit from the repair characteristics of network coding. Second, if channelizing a hyperarc has the potential to flood next-hop downstream hyperarcs with greatly increased load, we decrease its likelihood of channelization. These two parameters do not preclude channelization, but instead tend to give preference to hyperarcs that must rely on channelization for performance improvements and those that will not impact the performance in other neighborhoods.

We combine these four factors into a metric that represents the health of a hyperarc. Hyperarcs that have a small value for the health metric are considered sick and are candidates for channelization. Note that the analysis related to the health metric in the next subsections is only relevant for TDMA MAC and does not apply to 802.11.

4.3.1 Expected Number of Transmissions Due to Collisions

Since channelization serves to remove only errors due to collisions, here we only consider packet loss due to collisions. Let $p_u^v$ denote the probability of node $u$ receiving a transmission from node $v$ in spite of collisions. A low value of average $p_u^v$ at destinations $u$ of a hyperarc originating at $v$ will indicate high loss rate due to collisions. A node $u$ can estimate $p_u^v$ as

$$p_u^v = (1 - h'_u \cdot q_u) \cdot \prod_{w} (1 - h'_w \cdot q_w), w \in \gamma_1(u) - \{v\}$$ (4.6)
\[
    h'_u = \begin{cases} 
    h_u, & \text{if } h_u < 1 \\
    1, & \text{otherwise}
    \end{cases}
\] (4.7)

For a hyperarc with \( N \) destinations which delivers \( B \) packets to all the destinations by using intra-session network coding for error recovery, we present the detailed derivation of \( T(N, B) \), the number of transmissions required, in §3.4 (from our previous work on selective channelization [56]).

Hence, the expected number of transmissions per packet due to collisions for flow \( i \) in node \( v \) when using network coding can be represented by \( X^i_v \) where,

\[
    X^i_v = T(|\eta^i_v|, |\eta^i_v|, avg_{u \in \eta^i_v}(p^u_v))_{nc}/|\eta^i_v|
\] (4.8)

where \( \eta^i_v \) is the set of next hop nodes for flow \( i \) in node \( v \) and \( avg() \) is the average function. \( T()_{nc} \) is Eqn. 3.8 (cf. §3.4) for network coding with average probability of reception at next hop nodes \( avg_{u \in \eta^i_v}(p^u_v) \) as an additional input. The use of \( X \) as one of the channelization criteria makes our channelization scheme specific to network coding.

To calculate \( X^i_v \), \( p^u_v \) needs to be collected for all destinations \( u \) of a hyperarc. This requires every node to inform the source of each incoming hyperarc of a different receiving probability. To reduce this overhead, nodes append \( t_u \) to packets, where

\[
    t_u = (1 - h'_u \cdot q_u) \cdot \prod_{w} (1 - h'_w \cdot q_w), w \in \gamma_1(u)
\] (4.9)

Upon receiving \( t_u \), node \( v \) can compute \( p^v_u \) as

\[
    p^v_u = t_u/(1 - h'_v \cdot q_v)
\] (4.10)

This requires each node to append only one value to an outgoing packet rather than one for each incoming hyperarc.

While a metric similar to \( X^i_v \) was used as the sole criteria for channelization in selective channelization [56], here we estimate \( p^v_u \) for a more generic network. Moreover, we consider three additional components in the health metric as described next.
4.3.2 Expected Number of Backoffs

The impact of backoffs on network coding is different from the impact of collision errors. Backoffs preclude all next hop nodes from receiving a transmission. Unlike collisions, which afford the possibility of coding due to discrepancy errors, backoffs cannot be overcome by network coding directly. Channelization can help alleviate this problem by reserving more slots for a flow, thereby reducing the number of backoffs.

We define $R^i_v$ as the ratio of load to service for flow $i$ in node $v$ where,

$$R^i_v = \begin{cases} 
\frac{d^i_v}{(e^i_v + q^i_v \cdot |C_v|)}, & \text{if } d^i_v > (e^i_v + q^i_v \cdot |C_v|) \\
1, & \text{otherwise}
\end{cases} \quad (4.11)$$

$R$ is indicative of the number of attempts for a packet to be transmitted in spite of backoffs. $RX$ is the number of attempts for a packet to be delivered in spite of collisions and backoffs i.e., $RX$ is representative of normalized delay. For example, if each packet must be transmitted 2 times due to collisions ($X = 2$), and each time a packet is queued for transmission it requires 3 backoffs ($R = 3$), then the normalized delay to deliver a packet is 6. Our initial definition of the health metric (we augment the health metric in subsections to follow) is

$$H^i_v = 1/(R^i_v \cdot X^i_v) \quad (4.12)$$

As $H$ increases, data delivery performance over the hyperarc increases and the need for its channelization decreases. While selective channelization used an upper threshold on $X^i_v$ to determine if channelization is required, here we use a lower threshold on $H^i_v$ to determine the same. Also note that if $d^i_v \leq (e^i_v + q^i_v \cdot |C_v|)$, queue backlog is unlikely to be severe and hence $R$ is assigned a constant value to annul its impact on the health metric.

We now present simulation results to illustrate the importance of $R$ as a component of the health metric. We use a deployment (Fig. 4.1(a)) where nodes within communication range are connected by a link. There are four one-hop multicast flows - $0 \rightarrow \{1,2,3\}$, $1 \rightarrow \{0,4,5\}$, $2 \rightarrow \{0,6,7\}$, $3 \rightarrow \{0,8,9\}$ (referred to as flows 1-4 in the rest of this subsection). To provide contention in the medium surrounding some of the destinations of these multicast flows, six unicast flows are used as shown in
Fig. 4.1(a) with dashed arrows. This network is designed so that all destinations of the four multicast flows have independent sources of collision errors. We will present results for a more general network with random flows in Section 4.5.

Flow 1 is assigned a data rate of approximately 4ppf while flows 2-4 are assigned data rates of approximately 2ppf each. Data rates of unicast flows are close to 1ppf. All flows enter the network at 50s and channelization is triggered after $d_v$ stabilizes. The simulations are stopped at 80s. Though the simulation durations are small, they are sufficient to allow us to make important observations regarding the impact of $R$ on the health metric.

Simulations were conducted in NS-2 and use the 802.11 physical layer at 1Mbps. The packet size used is 512Bytes and the number of slots in a frame is set to 8 ($s_{data} = 7, s_{control} = 1$). We compare results for TDMA with no channelization
Figure 4.2. Four one-hop multicast flows in a 2-hop neighborhood, $q_{min} = 1.0$, $q_{max} = 3.0$

(“TDMA + No Chann.”) and TDMA with channelization of any one of the multicast hyperarcs with 2 slots in Figs. 4.1 and 4.2. “TDMA + Flow A” refers to flow “A” being channelized with 2 slots. For completeness, we also compare results to a standard 802.11 scheme which uses the 802.11 physical layer as well as the 802.11 MAC layer. This standard 802.11 does not use channelization. Note that all these configurations use network coding.

Results in Figs. 4.2(a)-4.2(c) show values of the performance metrics for each of the multicast hyperarcs, for all four multicast flows together (“4 Flows”) and for all flows including the unicast flows (“All”). These results were averaged over 5 simulation runs. We confirmed that the loads generated by each flow conformed to our profile and that transmission probabilities were assigned correctly.

In terms of normalized delay (Fig. 4.2(a)) and backoffs (Fig. 4.2(b)) 802.11 performs poorly for most of the multicast flows, all multicast flows and all flows...
together. For all flows together “TDMA + No Chann.” performs better than 802.11 in terms of normalized delay but slightly worse than channelization of any one of the multicast flows. Channelization of a flow improves its performance but degrades the performance of other flows. However, the overall performance for all four flows and all flows including the unicast flows is best when flow 1 is channelized (Fig. 4.2(a)).

Note that though the number of transmissions for 802.11 is lower than when flow 1 is channelized (Fig. 4.2(c)), this result does not present the complete picture. Excessive backoff employed by 802.11 (Fig. 4.2(b)) leads to few collisions. These backoffs increase the delivery delay of packets as shown by normalized delay (Fig. 4.2(a)).

To better understand how hyperarcs are selected for channelization, we examine $X$ and $H(= 1/(RX))$ for “TDMA + No Chann.” from a single simulation run that can be used to make channelization decisions in Figs. 4.1(b) and 4.1(c), respectively. These metrics are plotted against simulation timeline.

According to $X$ (Fig. 4.1(b)), flows 2-4 require the largest number of transmissions due to collisions to deliver a packet to all destinations followed by flow 1. In contrast, according to $H(= 1/(RX))$ (Fig. 4.1(c)), flow 1 has the smallest value of health followed by flows 2-4 which have similar values. As shown in Fig. 4.2(a), selecting flow 1 for channelization leads to minimum normalized delay. This illustrates that the inclusion of $R$ guides us to the correct hyperarc to channelize.

The overall delay decrement due to channelization is not large in these simulations because we channelize only 2 slots in a network with 16 nodes and 10 flows. Moreover, as we show later, the factor $R$’s primary role is to detect need for channelization of a hyperarc which would otherwise not be selected based only on $X$ (as is done by selective channelization).

### 4.3.3 Network Coding Opportunity

The health metric defined in the previous subsection, $H_i^v = 1/(R_i^v \cdot X_i^v)$, is insufficient as a measure of network coding’s recovery capability. If two hyperarcs experience the same congestion level, and therefore have the same health metric value, but only one can be channelized, it is advantageous to channelize the hy-
perarc that may benefit least from network coding. Therefore, in addition to the estimate of performance of network coding under current conditions, we use an estimate of coding opportunities over a hyperarc under varying conditions.

We use a network coding scheme based on COPE [48] and allow intra-flow coding only (cf. subsection 4.2.3). Packets are coded using XOR. Hence a coded transmission cannot enable a node to decode more than one packet out of it. The maximum reduction of retransmissions due to discrepancy errors over a hyperarc for flow $i$ originating at node $v$ is $(|\eta_v^i| - 1)$, where, as earlier, $\eta_v^i$ is the set of destinations of the hyperarc.

The larger the number of next hop nodes for a hyperarc, the more coding opportunities will arise due to discrepancy errors. We need an estimate for coding opportunities for a hyperarc under varying conditions. Since we want to use this estimate only as a bias and not as an over-riding metric, we use the following estimate

$$ G_v^i = 1 + |\eta_v^i| / |\gamma_1|_{\text{max}} $$

(4.13)

where $|\gamma_1|_{\text{max}}$ is the maximum number of one-hop neighbors in the network. Hence $G_v^i \in [1, 2]$. We limit its value within this range so that its impact on the overall health metric is moderate. $|\gamma_1|_{\text{max}}$ can be easily collected in the network during initialization. We augment $H_v^i$ with $G_v^i$ as

$$ H_v^i = G_v^i / (R_v^i \cdot X_v^i) $$

(4.14)

Hence at best $G$ can bias a hyperarc to be channelized even though it has $1/2$ of the $RX$ value of another hyperarc. Note that $G$ for all multicast hyperarcs in previous simulations is the same as they all have 3 destinations each. Hence it does not affect comparison of $H$ in Fig. 4.1(c) or results in Figs. 4.2(a)-4.2(c).

### 4.3.4 Impact on Next-Hop Downstream Hyperarcs

Channelizing a hyperarc has a direct impact on its neighborhood and the downstream hyperarcs of the same flow and their neighborhood. By channelizing a congested hyperarc near the source of a flow, the rate of packets arriving at next-hop downstream hyperarcs may be drastically increased. This may cause these
downstream hyperarcs to congest their neighborhoods and experience degraded performance. If a congested hyperarc near the destination(s) of the flow is channelized, it will tend to only impact its local neighborhood. Therefore, the challenge is to skew the health metric so that hyperarcs near the destinations of flows are more likely to be channelized.

Similar to $G$, we keep the range of metric for impact on next-hop downstream hyperarcs between 1 and 2 and define it as

$$J^i_v = 1 + \frac{\rho^i_v}{|\gamma_1|_{max}}$$  \hspace{1cm} (4.15)

where $\rho^i_v$ is the number of next-hop downstream hyperarcs for flow $i$ in node $v$ and as in previous subsection, $|\gamma_1|_{max}$ is the maximum number of one-hop neighbors in the network. $J^i_v$ is combined into the health metric as

Figure 4.3. Two-hop multicast flow, $q_{min} = 2.0$, $q_{max} = 6.0$
In the previous simulations, each multicast flow spans only one hyperarc. To test the impact of $J$ we conduct simulations with a 2-hop multicast flow $0 \rightarrow \{1,2,3\} \rightarrow \{6,7,8\}$ (Fig. 4.3(a)). The load on downstream hyperarc will depend on load and performance of the upstream hyperarc for this flow. Simulation settings were same as in previous simulations with the only difference being the number of slots in a frame being set to 6 ($s_{data} = 5$, $s_{control} = 1$). We refer to $0 \rightarrow \{1,2,3\}$ as “hyperarc 1” and $3 \rightarrow \{6,7,8\}$ as “hyperarc 2” in the rest of this subsection. Five unicast flows are used to cause independent contention at the destinations of these hyperarcs as shown in Fig. 4.3(a) with dashed arrows. The data rate of the multicast flow is approximately 3ppf while that of unicast flows is close to 2ppf.
We present results in Figs. 4.3 and 4.4 for 802.11, TDMA with no channelization (“TDMA + No Chann.”), and when any one of the multicast hyperarcs is channelized with 2 slots. “TDMA + Hyperarc A” refers to hyperarc A being channelized with 2 slots. Once again, all these configurations use network coding. These results are averaged over 5 runs and include results for each of the hyperarcs and also for all flows including unicast flows (“All”). Note that some bars for 802.11 in Figs. 4.4(a) and 4.4(b) are cut off at 6.0.

Channelization of hyperarc 2 leads to minimum normalized delay (Fig. 4.4(a)). Though it does require more transmissions than 802.11 (Fig. 4.4(c)), it is still the least among the TDMA schemes. As expected, the excessive backoff of 802.11 (Fig. 4.4(b)) reduces collisions but is unable to keep normalized delay low (Fig. 4.4(a)).

Fig. 4.3(b) shows that according to $X$, both hyperarcs are equally good candidates for channelization. But according to $H$ (Fig. 4.3(c)), hyperarc 2 is the best candidate for channelization. Results in Fig. 4.4(a) show that channelizing hyperarc 2 decreases normalized delay more than channelizing hyperarc 1. Note that both hyperarcs have 3 destinations each and hence have the same $G$. Also, $R$ (not shown) is 1 for both hyperarcs. As a result, the only difference in the value of $H$ for these hyperarcs is due to $J$. Hence this simple simulation illustrates the need for $J$ in the health metric.

### 4.4 Cooperative Channelization

The health metric allows us to compare need for channelization of different hyperarcs. In this section we introduce the cooperative channelization algorithm to decide which specific hyperarcs to channelize.

#### 4.4.1 Cooperative Channelization

The health of a hyperarc determines its expected performance. Larger health denotes better data delivery characteristics and a smaller need for channelization. Based on the health metric we use the following two rules for cooperative channelization:
• If the health of a hyperarc is less than a threshold \( H_{thr} \), it is labeled as sick.

• If a sick hyperarc has the lowest value of health in its 2-hop neighborhood, it is channelized if slots are available.

Nodes must determine if an outgoing hyperarc requires channelization. This requires knowing the health of the sickest hyperarc in the 2-hop neighborhood. This information is collected via piggybacking the health of sickest local hyperarc and sickest 1-hop hyperarc in outgoing packets.

Computing health metric and triggering channelization requires information from neighbors. This information is gathered via information piggybacked to data packets. Results presented in this chapter take the delay and overhead in gathering this information into account and demonstrate the feasibility of cooperative channelization in multi-hop wireless networks.

### 4.4.2 Number of Slots to Assign

Once a node decides that it should channelize a hyperarc, it needs to decide how many slots to assign to it. This can be done by allotting anywhere from 1 slot to as many as needed or possible. But assigning slots in bulk to a hyperarc may lead to other hyperarcs in the neighborhood being left with few slots to contend on, thereby drastically degrading their health.

We adopt the approach of assigning 1 slot every time a hyperarc meets the criteria for channelization. If the health of the hyperarc is not fully assuaged by this assignment, channelization will again be triggered for it and it will be assigned 1 more slot. This incremental assignment avoids drawbacks of bulk channelization and poor channelization decisions due to race conditions (channelization criteria being met at approximately the same time in multiple nodes in a neighborhood).

### 4.4.3 Thresholds

For the selective channelization algorithm we use a threshold \( X_{thr} \). If a hyperarc’s \( X_i \) exceeds this threshold, it is channelized. Like cooperative channelization, each time a hyperarc meets criteria for channelization, it is assigned 1 more slot. For a
fair comparison, the threshold $H_{\text{thr}}$ for cooperative channelization is set comparatively. Since $X_{\text{thr}}$ is the acceptable number of transmissions per packet, we use the following threshold for cooperative channelization

$$H_{\text{thr}} = G_{\text{thr}} \cdot J_{\text{thr}} / X_{\text{thr}}$$

(4.17)

where $G_{\text{thr}}$ is a threshold for $G_i^v$ and $J_{\text{thr}}$ is a threshold for $J_i^v$. These thresholds imply that a hyperarc that has sufficient normalized delay (i.e., $RX > X_{\text{thr}}$) will be favorably considered for channelization if its network coding opportunity estimate is less than $G_{\text{thr}}$ and its impact on next-hop downstream hyperarcs is less than $J_{\text{thr}}$.

### 4.5 Results

We perform simulations using NS-2 with an 802.11 physical layer operating at 1Mbps. Data packet size is set to 512Bytes and the number of slots in a frame is 12 ($s_{\text{data}} = 11$, $s_{\text{control}} = 1$). The link error rate in these simulations is set to 0; i.e., losses occur only due to collisions and inability to access media.

We present results for a scenario in which 25 multicast flow trees are randomly deployed in a network of 50 nodes in a 1000m×1000m area with a receiving range of 250m. The maximum number of 1-hop neighbors in this deployment is 10 (i.e., $|\gamma_1|_{\text{max}} = 10$). Different flows have different data rates. The ratio of the largest data rate to the smallest is 5 : 1 with the smallest data rate being approximately 1ppf. Among these 25 flows, there are a total of 85 hyperarcs. The maximum number of hops from a source to destination in any flow is set to 3. The number of destinations of each hyperarc varies from 1 to 5. These results are averaged over 5 runs. $X_{\text{thr}}$ was set to 10, while $G_{\text{thr}}$ and $J_{\text{thr}}$ were set to 1.2. We compare several different schemes that use network coding. For brevity we use abbreviations summarized in Table 4.2 to refer to schemes in the results presented next. Note that all these schemes, including 802.11, use network coding.
4.5.1 Multi-Hop Flows

We now discuss results for the multi-hop flows presented in Fig. 4.5. According to the most vital metric - normalized delay, all TDMA schemes perform better than 802.11 (Fig. 4.5(a)). Channelization schemes do better than TDMA with no channelization and cooperative channelization outperforms selective channelization.
81

(a) Selective Channelization

(b) Cooperative Channelization

(c) Cooperative Channelization + Lim.

Figure 4.6. Channelizations for 25 flows over a 50 node network, $q_{min} = 4.0$, $q_{max} = 12.0$

The 802.11 system suffers from aggressive backoff compared to the TDMA systems (Fig. 4.5(b)). Even though it avoids collisions as is evidenced by low number of actual transmissions (Fig. 4.5(c)), its normalized delay is much higher than the TDMA systems (Fig. 4.5(a)). Also, channelization may increase backoffs (Fig. 4.5(b)) though it more than makes up for it by decreasing number of transmissions (Fig. 4.5(c)).

The TDMA systems are more aggressive at transmitting, and therefore experience more collisions. This can be seen in their larger number of actual transmissions compared to 802.11. However, they experience smaller normalized delay because they do not employ backoff, but rather transmit with a probability proportional to their offered load.
Vitally, 802.11 does worse than TDMA with no channelization in terms of normalized delay (Fig. 4.5(a)). This shows that the TDMA MAC is better suited for congested multi-hop wireless networks that carry multicast traffic. The TDMA systems allow aggressive transmission. Resulting discrepancy errors are then efficiently repaired by network coding.

The effectiveness of channelization in reducing collisions is also apparent by the drastically reduced number of actual transmissions in Fig. 4.5(c) when compared to the non-channelized TDMA system. We now discuss the insights as to why cooperative channelization outperforms selective channelization.

The first observation from Fig. 4.5(d) is that cooperative channelization reserves more slots than selective channelization. Additional simulations confirm that it is not simply more reservations that lead to improved performance of cooperative channelization, but smarter selections of hyperarcs to channelize.

As a first test, we limit the number of channelizations that can be performed by cooperative channelization to be the same as number of channelizations for selective channelization (labeled “Lim.”). “Lim.” still outperforms selective channelization.

Fig. 4.6 shows the channelizations on the topology of the network for one of the simulation runs. Edges in the figures join nodes within communication range and channelized slots are marked next to the source of corresponding hyperarc. Cooperative channelization, with or without limitations on the number of slots it can reserve, channelizes different hyperarcs from selective channelization. Note that each channelization for “Lim.” (Fig. 4.6(c)) is a subset of the channelization decisions for “Coop.” (Fig. 4.6(b)). Through further simulations we determined there are two factors that lead to smarter hyperarc selection - the definition of the health metric, and cooperative nature of the selection of hyperarcs to channelize.
We first consider the impact of health metric as a measure of congestion. Selective channelization makes channelization decisions based solely on $X$, while with cooperative channelization the health metric incorporates $R$. To quantify the impact of this parameter, we obtained results with a modified health metric $H = 1/(RX)$ (labeled “1/(RX)”). This results in better overall performance in terms of normalized delay (Fig. 4.5(a)) than when only $X$ is considered, i.e., selective channelization. Thus the health metric with $R$ selects hyperarcs that may have fewer collisions than when selective channelization is used, but more overall performance degradation when backoffs are considered.

Cooperative channelization assigns slots to nodes that have been overlooked by selective channelization (Fig. 4.6). For example, in Fig. 4.6(b) the node surrounded by a box channelizes slots 11 and 6 using cooperative channelization. Selective channelization does not channelize at this node (Fig. 4.6(a)). This is because cooperative channelization considers $R$ in addition to $X$ and thus detects need for channelization that is missed by selective channelization.

Next, we consider the impact of $G$ and $J$ components. Results in Fig. 4.5(a) show that “1/(RX)” outperforms selective channelization but not cooperative channelization. This underlines the observation that while $R$ is a vital component of the health metric, the impact of $G$ and $J$ in biasing the health metric towards channelization of hyperarcs with little network coding opportunity or a smaller tendency to increase load on next-hop downstream hyperarcs is important as well.

Finally, with cooperative channelization, only the sickest hyperarc in a 2-hop neighborhood is channelized. This tends to spread channelization out across neighborhoods as shown in Figs. 4.6(b) and 4.6(c). Selective channelization attempts to channelize all hyperarcs experiencing high collision rates in a congested area (Fig. 4.6(a)). This leads to exhaustion of available slots, thus hurting performance in the neighborhood.

Fig. 4.5(e) shows normalized delay of each system with respect to selective channelization. Note that all results in this chapter include network coding (including 802.11). 802.11 and TDMA with no channelization incur 38% and 15% more delay, respectively, than selective channelization. Cooperative channelization incurs 14% less delay than selective channelization.
4.5.2 Varying Transmission Probability

As explained in Section 4.2, the sum of transmission probabilities in a two-hop neighborhood ranges from $q_{\min}$ to $q_{\max}$. In this set of simulations the scenario and traffic load are same as in the previous subsection and the transmission probability range is varied. In effect, this varies the aggressiveness of the TDMA transmissions.

We define a transmission factor in the TDMA system, $\delta$. The parameters for the transmission probability assignment algorithm are $\delta \cdot q_{\min}$ and $\delta \cdot q_{\max}$. As the transmission factor increases, the transmission probabilities increase, resulting in increasing $X$ values and decreasing $R$ values. The results are summarized in Fig. 4.7. Note that varying transmission probability does not have an impact on 802.11.

At a transmission factor of 1.0, none of the hyperarcs experience $X > X_{\text{thr}}$. As a result, no channelization is triggered by selective channelization (Fig. 4.7(c)). At such low transmission probabilities the backoff in nodes is excessive and nodes are overloaded, thus leading to high values of $R$. Since cooperative channelization takes $R$ into account, it recognizes sick hyperarcs in the network and channelizes them.
This reduces normalized delay (Fig. 4.7(a)) as well as transmissions (Fig. 4.7(b)). Again, 802.11’s backoff reduces transmissions (Fig. 4.7(b)) but is outperformed in normalized delay (Fig. 4.7(a)) by the TDMA based schemes.

As the transmission factor increases, $X$ increases and $R$ decreases. Hence the difference in number of channelizations between the selective and cooperative channelization algorithms decreases (Fig. 4.7(c)). At a high transmission factor of 2.5 the number of channelizations are almost the same. However, the distribution of these reserved slots among the hyperarcs is different among the schemes. As a result, even with similar number of channelizations cooperative channelization outperforms selective channelization (Fig. 4.7(a)).

Selective channelization decreases normalized delay by only as much as TDMA with no channelization up to transmission factor of 1.6. At transmission factor 1.6, cooperative channelization incurs 20% less delay than selective channelization. This highlights one of the vital strengths of cooperative channelization - detecting need for channelization that would be otherwise missed by selective channelization.

4.6 Related Work

Network coding [7] allows combining or modifying packets before forwarding. This enables it to reach the multicast capacity of a network which would otherwise not be possible with routing based solutions. Network coding can also be used to reduce the number of transmissions required as it can deliver different packets to different destinations via a single coded transmission. Koetter et al. [55] and Li et al. [68] have shown that linear network coding can be used to achieve the multicast capacity of the network.

Network coding has also moved from the realm of theoretical promise to performance improvements in actual experiments. COPE, a network coding scheme for wireless networks, was proposed by Katti et al. [48] and uses opportunistic listening and coding. Chaporkar et al. [22] show that network coding may reduce throughput as compared to plain routing because of a decrease in spatial reuse of medium resources. The authors propose a scheduling algorithm jointly designed with network coding to enhance the performance of network coding.

Contention based schemes like 802.11 have often been used in networks under
low load. Schedule based MAC schemes have been proposed to address high loads. With the constraint of spatial reuse the problem of optimal scheduling becomes difficult. Optimal scheduling of links or nodes in wireless networks is an NP-complete problem [30, 79]. Link scheduling has also been addressed in [89] for different interference models. Scheduling for single channel networks has also been explored in works like [14] and [79]. Network resource reservation in multichannel scenarios has been explored in [85] and [86].

Brar et al. [19] show that 802.11’s backoff scheme is unsuitable for wireless mesh networks and use Spatial TDMA [74]. [19] also presents an algorithm to efficiently schedule all transmissions in the network within a provable approximation factor of the optimal schedule. Probabilistic transmission in TDMA slots has been used in works like [29] and [75]. We adopt a mixed approach in this work - some transmissions in the network use reserved or scheduled slots, while others probabilistically contend for the medium.

4.7 Summary

In this chapter, we addressed the problem of resource reservation in a TDMA MAC for a wireless network that uses network coding to deliver multicast data. Our cooperative channelization algorithm determines if a hyperarc in the network needs to be channelized using the proposed health metric. Results show that cooperative channelization incurs up to 20% less delivery delay than selective channelization.
Network Coding Aware Rate Selection in Multi-Rate IEEE 802.11

Network coding has been proposed as an alternative to the conventional store-and-forward routing paradigm for data delivery in networks. When deployed in a multi-rate wireless network, network coding has to interact with rate adaptation. When multicasting packets (a requirement of network coding) in a multi-rate IEEE 802.11 wireless network, one must use care when selecting the transmission rate to use. We refer to this problem as rate selection. We analyze the performance of network coding for a small set of scenarios representative of common topologies in a network that lead to coding opportunities. Based on this analysis, we present our Network Coding aware Rate Selection (NCRS) algorithm which takes into account transmission rates used for unicast links to all multicast targets. Simulation results show that in a multi-hop wireless network, network coding with NCRS achieves up to 24% more gain over routing than network coding with other rate selection algorithms.

5.1 Introduction

Network coding [7] has been proposed as a method to increase the multicast capacity of wireless networks. In contrast to the store-and-forward paradigm of routing, network coding allows nodes to combine packets before forwarding them. In essence, network coding enables the delivery of different packets to distinct
neighbors with a single transmission. Coded packets must be multicast for network coding to gain efficiency. Moreover, it is essential that uncoded packets be overheard by neighbors to enable inter-flow coding i.e., uncoded packets may need to be multicast.

Unfortunately, in practical wireless networks, channel conditions on and among links often vary widely. IEEE 802.11, a widely prevalent PHY and MAC protocol for wireless networks, attempts to improve the performance of a link under any given condition by transmitting at appropriate transmission rates and modulation schemes. Selecting a transmission rate incurs an inherent tradeoff wherein increasing rate results in decreasing packet delivery probability. These factors can be combined into throughput, frequently the metric for network performance. As link characteristics vary, a different transmission rate may increase throughput. This problem of adapting transmission rate to varying link characteristics has been referred to as rate adaptation [16, 20, 39, 44, 62, 82, 91].

A rate adaptation algorithm may determine different transmission rates from a node to its different neighbors. In a network using network coding, we refer to the problem of selecting a transmission rate for all multicast recipients, or any neighbors which may benefit from overhearing, as the rate selection problem. If rate selection for uncoded packets precludes overhearing in a neighborhood, few inter-flow coding opportunities will arise. Rate selection for coded packets has to address the following tradeoff - if a low transmission rate is chosen, all destinations may receive a transmission successfully but the transmission time will be longer; if a higher rate is chosen, the transmission time will decrease but destinations with poor link quality may not receive the transmission.

In this chapter we propose a rate selection algorithm to maximize throughput in a multi-hop multi-rate wireless network in which network coding is used. Virtually all prior work focuses on maximizing coding gain i.e., maximize the reduction in network traffic due to network coding. Often these methods use the lowest transmission rate available to enable network coding to the largest extent possible. Our solution selects transmission rates to maximize throughput while taking network coding into account. In some cases this entails setting rates that fully leverage network coding, and in others coding is largely precluded and yet, throughput is maximized. The contributions of this chapter are -
We first identify the prevalent fundamental building blocks in a network that enable network coding. We analyze the performance of network coding and routing over these building blocks in terms of throughput in a multi-rate environment.

Based on this analysis, we propose Network Coding aware Rate Selection (NCRS) as a linear programming problem to maximize total throughput on a multicast link.

We present simulation results to compare NCRS with other rate selection schemes that try to either maximize network coding gain or use the highest rates supported by the component links of a multicast link to minimize link occupancy time. We illustrate the wide applicability of NCRS by examining several scenarios that contain multicast links which support different transmission rates on component unicast links. In addition, for a large network, we show that NCRS outperforms the alternative rate selection schemes by up to 24% on average in terms of gain over routing. NCRS is also more robust than these alternatives in that it works for a wide range of channel characteristics.

Note that while the NCRS problem formulation is based on the analysis of building blocks that yield coding opportunities, results for large networks show that NCRS indeed improves the data delivery characteristics of such a network.

The rest of this chapter is organized as follows: §5.2 discusses network coding and rate adaptation; §5.3 presents the motivation and design details of our rate selection protocol. We analyze a small scenario for throughput with routing and network coding in §5.4. Based on this analysis we present our rate selection algorithm NCRS in §5.5. Results are presented in §5.6. Related work for network coding in multi-rate MAC environments is discussed in §5.7 and §5.8 summarizes the chapter.

5.2 Background

We now present related work on network coding and introduce the network coding algorithm used in §5.2.1. §5.2.2 presents related work on rate adaptation and details of multi-rate IEEE 802.11g and rate adaptation scheme used.
5.2.1 Network Coding

Several papers have been recently published to propose and fundamentally advance the area of network coding [7, 22, 24, 38, 55, 68]. We use a slightly modified version of COPE [48], a previously proposed network coding protocol for wireless networks. COPE requires nodes to overhear transmissions in their neighborhood. In addition, nodes in COPE need to know which packets have been received by their neighbors. This information is collected in one of the following ways - asynchronous ACKs, packet reception reports, or probabilistic packet delivery information based on Expected Transmission Count (ETX) [27]. Once a node is made aware of the packets available at its neighbors, it codes packets that can be decoded by all neighbors. Note that COPE scans only the 1-hop neighborhood of the transmitting node for opportunity to send coded packets. In addition, encoding and decoding is performed on a per-hop basis.

To illustrate the basic operation of per-hop encoding and decoding refer to Fig. 5.1. Native packets may need to be overheard at the neighbors of a transmitting node to create inter-flow coding opportunity. On the other hand, a coded packet is useful to multiple neighbors, and hence it must be multicast. For example, nodes $n_3$ and $n_4$ have to deliver packets $pkt_0$ (of flow 0) and $pkt_1$ (of flow 1) to nodes $n_1$ and $n_2$, respectively. Say, node $n_3$ transmits $pkt_0$ which is received by $n_0$. Similarly, $n_4$ transmits $pkt_1$ which is received by $n_0$. Now, assume that the intersection node, i.e., $n_0$, combines these packets together into - $pkt_0$ XOR
Node $n_1$ can extract $pkt_0$ from this coded packet only if it has $pkt_1$. This packet can only be received by overhearing $n_4$’s transmission of $pkt_1$. Hence native packet transmissions for flow 1 from $n_4$ must have $n_0$ as a direct target and $n_1$ as an overhearing target.

We use COPE’s underlying philosophy of keeping track of packets at next hop nodes and transmitting coded packets to efficiently address gaps in delivered packet sequences. We assume the presence of a perfect feedback system that enables nodes to keep track of packets delivered to next hop nodes. While this is a simplified model, it allows us to concentrate on the performance of network coding. We combine packets using XOR (like COPE) and ensure that a coded transmission can be decoded at its target nodes. Note that NCRS is not COPE-specific. Instead, it only uses components of COPE that are common with other network coding schemes.

In this chapter, we consider only unicast flows. As a result, there is little opportunity for intra-flow coding and so virtually all coding that occurs is inter-flow coding.

### 5.2.2 Multi-Rate IEEE 802.11g and Rate Adaptation

In this section we provide background on rate adaptation and IEEE 802.11g [1] and discuss details of the rate adaptation algorithm used. When selecting the transmission rate for a multicast link, referred to as a hyperarc with “direct” and “overhearing” targets, it is possible that rate adaptation will select different rates to different targets of the hyperarc. The problem of determining a single transmission rate for all hyperarc targets is referred to as rate selection.

Since IEEE 802.11 does not specify a rate adaptation algorithm, various algorithms have been proposed to address this void. These schemes differ in their approach to channel quality measurement and criteria for switching to a different rate. But at the core, most rate adaptation schemes aim to maximize throughput given a channel condition. AARF [62] employs a threshold on consecutive successful transmissions to probe the higher rate. A binary exponential backoff mechanism is used to determine this threshold. If the probe fails, the threshold is doubled, and so on (up to a maximum of 50). Rate adaptation has been an
active field of research lately and several schemes [16, 20, 39, 44, 82, 91] have been proposed.

IEEE 802.11g, the multi-rate wireless protocol used in this chapter, allows transmissions at 6, 9, 12, 18, 24, 36, 48, and 54Mbps. Since we deal with static networks in this chapter, stable Signal-to-Noise Ratios (SNRs) are observed at nodes. Hence, like [23] and [39], we measure channel conditions at the receiver by measuring SNR of data packets. For the same SNR, transmitting at a lower rate will tend to result in a lower error rate. To verify this, we first plot packet delivery probability vs. SNR in Fig. 5.2 for a unicast link with different transmission rates from an experiment. Note that plots for some of the transmission rates are not shown for brevity. The experimental setup used a Linksys WRT54G with DD-WRT access point as a sender and a Netgear WG111v2 wireless card as receiver. For each result 10000 packets of 1500Bytes were sent from the access point. We then present corresponding results from a NS-2 simulation for comparison in Fig. 5.2. Since the simulation characteristics closely mirror those exhibited by the experiments and

Figure 5.2. Experiment vs. simulation results for 1500Byte packets
Table 5.1. Transmission Rates and Rate Adaptation for 1500Byte packets

<table>
<thead>
<tr>
<th>Mbps</th>
<th>pps</th>
<th>SNR Range</th>
<th>Mbps</th>
<th>pps</th>
<th>SNR Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>6</td>
<td>376</td>
<td>≤ 3.77</td>
<td>24</td>
<td>905</td>
<td>9.99-15.61</td>
</tr>
<tr>
<td>9</td>
<td>508</td>
<td>NA</td>
<td>36</td>
<td>1071</td>
<td>15.61-18.40</td>
</tr>
<tr>
<td>12</td>
<td>616</td>
<td>3.77-8.90</td>
<td>48</td>
<td>1182</td>
<td>18.4-23.10</td>
</tr>
<tr>
<td>18</td>
<td>783</td>
<td>8.9-9.99</td>
<td>54</td>
<td>1222</td>
<td>&gt; 23.10</td>
</tr>
</tbody>
</table>

that presented in prior work [16, 37], we generate further results for rate selection using NS-2.

Though IEEE 802.11g allows transmission rates from 6Mbps to 54Mbps, the maximum achieved throughput does not increase linearly due to protocol overhead. For a given packet size, we can translate these transmission rates into maximum throughput in terms of packets-per-second (pps). We simulate a unicast link with a packet size of 1500Bytes in NS-2 (simulation details presented later in §5.6) with the results shown in Table 5.1.

We now have the packet delivery probability for any given SNR (cf. Fig. 5.2) and the maximum throughputs achieved with all rates (cf. Table 5.1). Therefore it is straightforward to determine the SNR range for which each rate maximizes throughput. This is shown in Table 5.1. Note that 9Mbps is never selected as a transmission rate (similar to an observation made in [62]). We use this generic rate adaptation scheme to determine transmission rates on unicast links. Since we concentrate on static networks in this chapter, any rate adaptation scheme that maximizes throughput will only do as well as our rate adaptation scheme. In addition, our approach to rate selection is orthogonal to the rate adaptation solution by design.

### 5.3 Network Coding and Rate Selection

In this section we present the basic motivation and design of our rate selection protocol. The purpose of this protocol is to adequately deliver packets and to gather SNR information which is then used in the rate selection algorithm (cf. §5.5).

To enable inter-flow coding 2 or more flows need to intersect appropriately i.e., with requisite overhearing requirements as demonstrated with an example
Figure 5.3. Small scenarios for inter-flow network coding of 2 flows (not to scale)

(a) 3 nodes : Flow 0 : \( n_2 \rightarrow n_0 \rightarrow n_1 \), Flow 1 : \( n_1 \rightarrow n_0 \rightarrow n_2 \)

(b) 4 nodes : Flow 0 : \( n_3 \rightarrow n_0 \rightarrow n_1 \), Flow 1 : \( n_1 \rightarrow n_0 \rightarrow n_2 \)

(c) 5 nodes : Flow 0 : \( n_3 \rightarrow n_0 \rightarrow n_1 \), Flow 1 : \( n_4 \rightarrow n_0 \rightarrow n_2 \)

For the frequently occurring instance of coding only two packets, the three sim-
ple scenarios shown in Fig. 5.3 (disregard SNR and rate annotations in the figures for now) exhaustively represent all possible inter-flow coding patterns that may occur in a large network with our per-hop encoding and decoding approach. Solid arrows represent actual next-hop relationships. Dotted arrows denote overhearing required to make network coding at the intersection node $n_0$ feasible.

A rate selection algorithm has to address an overhearing tradeoff and a multicast tradeoff. Consider the 5 node scenario in Fig 5.3(c). Nodes $n_3$ and $n_4$ have to transmit at a rate such that overhearing is successful at $n_2$ and $n_1$, respectively. If these nodes do not enable successful overhearing, coding will not occur at $n_0$. For example, let SNR at $n_0$ and $n_2$ from $n_3$ be 23.3dB and 17.3dB, respectively. If $n_3$ transmits at 54Mbps (corresponding to 23.3dB, cf. Table 5.1), packets will be overheard at $n_2$ with probability 0.22 - (cf. Fig. 5.2). Hence promoting overhearing at $n_2$ may require reducing the transmission rate at node $n_3$. But this may reduce throughput at direct target $n_0$. This is the overhearing tradeoff. To address this tradeoff we maximize total throughput at hyperarc targets - direct and overhearing, as presented in detail in §5.5.

Now consider node $n_0$ in Fig. 5.3(c) which is connected to $n_1$ and $n_2$ with links supporting 15.3dB and 23.3dB SNR, respectively. Coded packets need to be transmitted at a rate such that $n_1$ can receive them. If 54Mbps is used (corresponding to 23.3dB), these packets are not received at $n_1$ and hence the transmission rate needs to be reduced. But if uncoded packets are transmitted instead of coded packets, those destined for $n_2$ can be transmitted at 54Mbps, while those destined for $n_1$ can be transmitted at 24Mbps. Hence $n_0$ can possibly transmit uncoded packets at a higher rate to a single destination or it can transmit coded packets at a lower rate. We refer to this as the multicast tradeoff. Since we code only 2 packets, we adopt the simple policy of coding whenever possible. On a related note, Vieira et al. [87] and Yomo et al. [94] consider the interaction of network coding and multi-rate MAC and recognize that coding more packets may require reducing transmission rate - similar to the multicast tradeoff.

Next, we present details of adapting RTS/CTS exchange for multicast packets in §5.3.1. We then explain how SNR of data packets from neighbors is collected in §5.3.2.
5.3.1 Multicast and RTS/CTS Exchange

As shown earlier, network coding requires multicast of all coded and some native packets. These multicast transmissions are prone to loss due to hidden terminal problems. Hence, in spite of its overhead, the RTS/CTS mechanism is employed to prevent excessive loss of multicast transmissions crucial to network coding. But requiring every destination - direct or overhearing - to reply with a CTS to the RTS is infeasible with respect to the overhead imposed. Hence we employ the common method [48] of selecting one of the destinations as the target for a RTS. Only this node replies with the CTS and sends the ACK after successful reception of the data packet. We refer to this node as cts-node for the transmission. Note that packet reception at non-cts-nodes is still prone to losses due to hidden terminal collisions.

The selection of the cts-node is critical. Assume that the 3-node scenario (cf. Fig. 5.3(a)) occurs in a large congested network and nodes \( n_1 \) and \( n_2 \) are connected to \( n_0 \) by links with SNRs of 15.3dB and 23.3dB, respectively. Let rate selection algorithm in node \( n_0 \) decide to transmit coded packets at 36Mbps. At this rate, packet delivery probability is 0.81 for \( n_1 \) and close to 1 for \( n_2 \) (cf. Fig. 5.2). Hence, if \( n_1 \) is the cts-node for coded packets of node \( n_0 \), there will be more retransmissions per packet but each packet will be delivered to both destinations with a high probability. If \( n_2 \) is the cts-node, few, if any, retransmissions will be required, but \( n_1 \) may not receive some of the packets.

We select the direct target with most packet collisions as the cts-node to minimize losses due to hidden terminal collisions. If all direct targets have similar degrees of loss due to collisions, the direct target with best link conditions (measured by SNR) is selected as the cts-node to minimize the number of retransmissions per packet. If multiple direct targets have the same SNR, one of these is picked randomly as the cts-node. An overhearing destination is not selected as the cts-node to avoid the flow from being penalized to promote overhearing.

5.3.2 Receiver Based SNR Measurement

While we do not modify the RTS and CTS packets, some information is piggy-backed on to data and ACK packets to enable network coding and rate selection.
For rate selection, we must collect the SNR of data packet transmissions at neighbors. Though all neighbors may not be overhearing targets, neighboring nodes overhear all packets. Nodes maintain a per-neighbor exponential weighted average of SNRs of data packets received. This information is collected in one of the following ways -

- Data packets contain exponentially averaged SNR for the directed link from cts-node to the current node.
- ACKs from the cts-node contain exponentially averaged SNR of packets received from source node.
- Overhearing targets and some direct targets may not be selected as the cts-node. Hence exponentially averaged SNRs of transmissions from each neighbor can be piggybacked on to data packets or sent in periodic control packets (similar to packet reception reports in COPE [48]).

5.4 Throughput Analysis

To explore the impact of multi-rate IEEE 802.11g on the gain over store-and-forward routing for network coding, we analyze throughput for routing and network coding for the 5-node scenario (cf. Fig. 5.3(c)). We use this analysis as the basis of our rate selection algorithm presented in §5.5. The analysis for 3 and 4-node scenarios are similar and are not presented for brevity. We verify the analysis presented here in §5.6 to see the impact of our assumptions stated below.

We first analyze components required to evaluate throughput over a hyperarc in §5.4.1. The throughput of routing and network coding on the 5-node scenario is then analyzed in §5.4.2.

Note that the packets-per-second (pps) version of rates is used in the analysis below and the network is assumed to be in steady state. Also, collision loss is not accounted for in this analysis as the interference range is 2× the receiving range and this results in RTS/CTS eliminating hidden terminal problems in all three scenarios in Fig. 5.3. Hence all packet losses in the analysis below are due to low SNR. Though this assumption breaks down in a large network, it provides
us with a stepping stone to the rate selection algorithm proposed in §5.5. Larger simulation scenarios presented in §5.6 indeed incur packet loss due to collisions arising from hidden terminal problems.

5.4.1 Hyperarc Throughput

To estimate throughput over a hyperarc, we need to find the total number of transmissions and account for retransmissions. We first look at retransmissions and subsequently analyze the total number of transmissions.

5.4.1.1 Number of Retransmissions per Packet

For this analysis we assume that the network is loaded with Continuous Bit Rate (CBR) unicast flows. These flows impose traffic at constant rate and employ hop-by-hop retransmissions to ensure data delivery. When an intermediate node receives a packet, it does not immediately forward it. Instead, it is stored in an internal network coding buffer at the routing layer which is separate from the transmission buffer in the link layer. At the load rate determined by rate of the CBR flow, a packet for this flow, referred to as primary-native packet, is extracted from the network coding buffer (possibly coded with packets from other flows) and enqueued into the transmission buffer.

If a packet is lost, hop-by-hop link layer retransmissions are used to deliver the packet. The number of link layer retransmissions per packet is limited to a maximum of 5 in 802.11g. If these link layer retransmissions are unsuccessful, the network coding layer retransmits the primary-native packet possibly coded with another packet. These retransmissions are hop-by-hop as well and are referred to as routing layer retransmissions.

Now, let \( iJ \) denote a hyperarc where node \( n_i \) is the source of the hyperarc and \( J \) is the set of next hop destinations (direct and overhearing) \( n_j \). Let transmission rate on hyperarc \( iJ \) be \( R_{iJ} \) and packet delivery probability to node \( n_j \) be \( q^j_{iJ} \). \( q^j_{iJ} \) depends on \( R_{iJ} \) and the SNR \( SNR_{ij} \) at node \( j \) i.e.,

\[
q^j_{iJ} = q(R_{iJ}, SNR_{ij}) \quad (5.1)
\]

where \( q() \) is a function that translates a rate and SNR tuple to the corresponding
packet delivery probability (derived from Fig. 5.2). We assume that the underlying rate adaptation mechanism does not change transmission rate $R_{iJ}$ during link and routing layer retransmissions.

Note that regardless of the transmission rate of data packets, ACKs are transmitted at 6Mbps. Now, the packet delivery probability for ACK packets at 6Mbps is close to 1 for SNR greater than 4dB. Hence the probability of receiving the ACK packet from the cts-node with the assumption of no collision-related loss is very close to 1 and we assume that ACKs are not lost due to low SNR at the source node. Hence, the expected number of link layer transmissions per packet on hyperarc $iJ$ with $n_j$ being the cts-node is

$$L_{iJ}^j = \sum_{m=1}^{5} m(1-q_{iJ}^j)^{m-1}q_{iJ}^j$$  \hspace{1cm} (5.2)$$

where the limit 5 is due to link layer retransmissions per packet being limited to a maximum of 5 in 802.11.

The probability of link layer retransmissions delivering the packet to the cts-node $n_j$ is

$$q_{iJ}^j = \sum_{m=1}^{5} (1-q_{iJ}^j)^{m-1}q_{iJ}^j$$ \hspace{1cm} (5.3)$$

The probability of these link layer retransmissions delivering the packet to a non-cts-node $n_k$ is

$$q_{iJ}^k = \sum_{m=1}^{5} ((1-q_{iJ}^j)^{m-1}q_{iJ}^j(\sum_{l=1}^{m} (1-q_{iJ}^{k})^{l-1}q_{iJ}^{k}))$$ \hspace{1cm} (5.4)$$

where $\sum_{l=1}^{m} (1-q_{iJ}^{k})^{l-1}q_{iJ}^{k}$ is the probability of $n_k$ receiving a packet when exactly $m$ link layer transmissions are used.

When link layer retransmissions fail to deliver a packet to the cts-node $n_j$, routing layer retransmissions are required. The expected number of such routing layer retransmissions is

$$T_{iJ}^j = \sum_{m=1}^{\infty} m(1-q_{iJ}^j)^{(m-1)}q_{iJ}^j = 1/q_{iJ}^j$$ \hspace{1cm} (5.5)$$
Since retransmissions are designed to guarantee packet delivery to the cts-node, the probability of these routing layer transmissions delivering a packet to the cts-node $n_j$ is assumed to be 1. These retransmissions may not deliver the packet to a non-cts-node. The probability of packet delivery to a non-cts-node $n_k$ after routing layer retransmissions is

$$q_{i,j}^{n_k} = \sum_{m=1}^{\infty} (1 - q_{i,j}^{n_j})^{m-1} q_{i,j}^{n_j} \left( \sum_{l=1}^{m} (1 - q_{i,j}^{n_k})^{l-1} q_{i,j}^{n_k} \right)$$

(5.6)

where $\sum_{l=1}^{m} (1 - q_{i,j}^{n_k})^{l-1} q_{i,j}^{n_k}$ is the probability of $n_k$ receiving a packet when exactly $m$ routing layer transmissions are used.

Now, the expected total number of retransmissions per packet $Z_{i,j}$ on a hyperarc $iJ$ with cts-node being $n_j$ is

$$Z_{i,j} = L_{i,j} T_{i,j}^{j}$$

(5.7)

Hence, if the total number of transmissions on a hyperarc is $N$, the expected number of unique packets received at the cts-node $n_j$ will be $N/Z_{i,j}$, while that at a non-cts-node $n_k$ will be $N q_{i,j}^{n_k}/Z_{i,j}$. Note that all further analysis using $L$, $T$, or $Z$ is for expected values.

### 5.4.1.2 Number of Transmissions on a Hyperarc

The number of transmissions on a hyperarc in a node $n_i$ depends on the node’s transmission probability $p_i$ and medium sharing among flows within the node. Bianchi [15] analyzes transmission probability of nodes in a fully connected network without hidden terminals. But our small scenarios for network coding are not fully connected. Analyzing a node’s transmission probability in a multi-hop multi-rate wireless network which uses IEEE 802.11 (CSMA/CA with RTS/CTS) is difficult. In fact, it is a function of the offered load and congestion in the network.

A node contends for the medium as one on behalf of all flows traversing it. The assigned medium is shared among the flows using a transmission buffer (FIFO buffer with tail-drop). We estimate the number of packets that a node transmits on behalf of a hyperarc using $NumTx()$ in Algorithm 1.
Algorithm 1 \{N^j, \ldots\} \text{NumTx}(\{(B^j, A^j, R^j, L^j, T^j), \ldots\}, p)\\

1: \textbf{for } j \in [1, n] \textbf{ do}\\
2: \quad d^j = \min(T^j B^j, A^j)\\
3: \quad r^j = L^j d^j / R^j\\
4: \textbf{end for}\\
5: \textbf{if } \sum_{j=1}^n r^j \leq p \textbf{ then}\\
6: \quad \textbf{for } j \in [1, n] \textbf{ do}\\
7: \quad \quad N^j = L^j d^j\\
8: \quad \textbf{end for}\\
9: \textbf{else}\\
10: \quad \textbf{for } j \in [1, n] \textbf{ do}\\
11: \quad \quad f^j = L^j d^j / \sum_{m=1}^n L^m d^m\\
12: \quad \textbf{end for}\\
13: \quad D = \sum_{j=1}^n f^j / R^j\\
14: \quad U = 1 / D\\
15: \quad \textbf{for } j \in [1, n] \textbf{ do}\\
16: \quad \quad N^j = f^j U p\\
17: \quad \textbf{end for}\\
18: \textbf{end if}\\
19: \textbf{return } \{N^j, \ldots\}

\text{NumTx}() \text{ takes a set of } n \text{ tuples - one for each hyperarc emanating from the node, and the transmission probability } p \text{ for the node as arguments. A tuple for a hyperarc consists of - incoming load } (\equiv B, \text{ in pps}), \text{ application level constant bit-rate } (\equiv A, \text{ in pps}), \text{ transmission rate } (\equiv R, \text{ in pps}), \text{ number of link layer transmissions per packet } (\equiv L), \text{ and number of routing layer transmissions per packet } (\equiv T). \text{ NumTx}() \text{ returns the set of number of transmissions } \{N^j, \ldots\} \text{ for each outgoing hyperarc } j.

In \text{NumTx}(), \text{ first, for each hyperarc } j, \text{ the load imposed on the link layer is determined as } d^j. \text{ The number of packets entering the routing layer for hyperarc } j \text{ is } B^j. \text{ Due to routing layer retransmissions these packets impose a maximum load of } T^j B^j \text{ packets on the link layer. The link layer drains packets from the network coding buffer at rate } A^j. \text{ Hence the rate of packets being enqueued for flow } i \text{ in the transmission buffer is } d^j = \min(T^j B^j, A^j). \text{ The fraction of time that the node requires to transmit all of the load } d^j \text{ when link layer retransmissions are taken into account is } r^j = L^j d^j / R^j. \text{ But the fraction of time that the node is allotted is}
Hence if \( \sum_{j=1}^{n} r^j \leq p \), all imposed load can be transmitted and the number of total transmissions for hyperarc \( j \) is \( N^j = L^jd^j \).

On the other hand, if \( \sum_{j=1}^{n} r^j > p \), load imposed on the node is more than it can transmit. The fraction of packets \( f^j \) of a hyperarc in the transmission buffer is proportional to the load imposed. Hence \( f^j = L^jd^j / (\sum_{m=1}^{n} L^md^m) \). The average inter-transmission delay is \( D = \sum_{j=1}^{n} f^j / R^j \) and hence the number of packets transmitted per unit time on average is \( U = 1 / D \). Now, for each flow, the number of packets transmitted is proportional to the fraction of packets in the buffer and since the node transmits for only \( p \) fraction of the time, the number of transmissions for hyperarc \( j \) is \( N^j = f^j Up \).

### 5.4.2 Throughput for 5-Node Scenario

Now that we have estimated the number of retransmissions per packet and total transmissions on a hyperarc, we analyze throughput for routing and network coding for the 5-node scenario (cf. Fig. 5.3(c)). We include the number of packets delivered on each hop of the unicast routes in the evaluation metric in this section. We present results for packets delivered to only final destinations in §5.6.

Two unicast flows are deployed in the 5-node scenario (cf. Fig. 5.3(c)) - flow 0: \( n_3 \rightarrow n_0 \rightarrow n_1 \) and flow 1: \( n_4 \rightarrow n_0 \rightarrow n_2 \). We analyze the number of packets of flow 0 received by \( n_0 \) and \( n_1 \) and the number of packets of flow 1 received by \( n_0 \) and \( n_2 \) for routing and network coding per unit time.

#### 5.4.2.1 Routing

The total number of unique packets received with routing is

\[
X = X_0^0 + X_0^1 + X_1^0 + X_2^1
\]

(5.8)

where \( X_i^j \) is the number of unique packets of flow \( j \) received by node \( n_i \) in unit time when routing is used.

Let \( N_{i,j} \) denote the number of transmissions (including retransmissions) on hyperarc \( iJ \) by node \( n_i \). Hence number of transmissions by node \( n_3 \) is

\[
\{ N_{3(0)} \} = \text{NumTx}(\{(B^0, A^0, R_{3\{0\}}, L_{3\{0\}}^0, T_{3\{0\}}^0)\}, p_3) \quad (5.9)
\]
where $A_l$ is the CBR flow rate of flow $l$. Since the incoming load for the routing layer at the source of a flow is the same as its application level load rate, $B_0^0 = A_0^0$.

Similarly, number of transmissions by node $n_4$ is

$$\{N_{4(0)}\} = NumTx(\{(A^1, A^1, R_4^{0}, L_{4(0)}^0, T_{4(0)}^0)\}, p_4) \quad (5.10)$$

The number of unique packets of flow 0 received by node $n_0$ is:

$$X_0^0 = N_{3(0)}/Z_{3(0)}^0 \quad (5.11)$$

Similarly,

$$X_0^1 = N_{4(0)}/Z_{4(0)}^0 \quad (5.12)$$

We now compute the number of transmissions for flows 0 and 1 by node $n_0$ as the following where $M_{i,j}$ is the tuple for hyperarc $iJ$,

$$\{N_{0(1)}, N_{0(2)}\} = NumTx(\{M_{0(1)}, M_{0(2)}, p_0\}) \quad (5.13)$$

where tuple for flow 0 is

$$M_{0(1)} = (X_0^0, A^0, R_{0(1)}, L_{0(1)}^1, T_{0(1)}^1) \quad (5.14)$$

and tuple for flow 1 is

$$M_{0(2)} = (X_0^1, A^1, R_{0(2)}, L_{0(2)}^2, T_{0(2)}^2) \quad (5.15)$$

Hence the number of unique packets of flow 0 received by node $n_1$ is

$$X_1^0 = N_{0(1)}/Z_{0(1)}^1 \quad (5.16)$$

Similarly, the number of unique packets of flow 1 received by node $n_2$ is

$$X_2^1 = N_{0(2)}/Z_{0(2)}^2 \quad (5.17)$$

Hence $X$ in Eqn. 5.8 can be computed to evaluate throughput for routing.
5.4.2.2 Network Coding

Like the previous subsection, the total number of unique packets received with network coding is

\[ Y = Y_0^0 + Y_0^1 + Y_1^0 + Y_2^1 \]  

(5.18)

where \( Y_i^j \) is the number of unique packets of flow \( j \) received by node \( n_i \) in unit time when network coding is used. Again, let \( N_{i,J} \) be the number of transmissions on hyperarc \( iJ \).

Now, node \( n_3 \) has to multicast at rate \( R_{3\{0,2\}} \) to allow node \( n_2 \) to overhear its transmissions. Since \( n_2 \) is an overhearing target, \( n_0 \) is selected as the cts-node. Hence number of transmissions by node \( n_3 \) is

\[ \{N_{3\{0,2\}}\} = NumTx(\{(A^0, A^0, R_{3\{0,2\}}, L_{3\{0,2\}}^0, T_{3\{0,2\}}^0), p_3\}) \]  

(5.19)

Similarly number of transmissions by node \( n_4 \) is,

\[ \{N_{4\{0,1\}}\} = NumTx(\{(A^1, A^1, R_{4\{0,1\}}, L_{4\{0,1\}}^0, T_{4\{0,1\}}^0), p_4\}) \]  

(5.20)

The number of unique packets of flow 0 received by node \( n_0 \) is

\[ Y_0^0 = N_{3\{0,2\}}/Z_{3\{0,2\}}^0 \]  

(5.21)

Note that we do not select overhearing targets as cts-node. As a result we do not need to include \( Z_{3\{0,2\}}^2 \) in Eqn. 5.21.

Similarly,

\[ Y_0^1 = N_{4\{0,1\}}/Z_{4\{0,1\}}^0 \]  

(5.22)

Node \( n_0 \) may be able to code some packets and will have to transmit remaining packets in native form. Let \( \alpha \) be the fraction of CBR rate of flow 0 that is transmitted as coded packets. A packet of flow 0 is coded with a packet of flow 1. Remaining packets of both flows have to be transmitted in native form. Also, without loss of generality, assume that node \( n_2 \) has a higher SNR than \( n_1 \) of data packets received from \( n_0 \). Hence \( n_2 \) is the cts-node for all coded transmissions of \( n_0 \).
The number of coded transmissions $N_c$ and native transmissions for flows 0 and 1 - $N_u^0$ and $N_u^1$, respectively - by node $n_0$ are computed as the following where, again, $M_{iJ}$ is the tuple for hyperarc $iJ$:

\[
\{N_c, N_u^0, N_u^1\} = NumTx(\{M_{0(1,2)}, M_{0(1)}, M_{0(2)}\}, p_0) \tag{5.23}
\]

where tuple for multicast hyperarc $0\{1, 2\}$ is

\[
M_{0(1,2)} = (Y_0^0\alpha, A^0, R_{0(1,2)}, L_{0(1,2)}^2, T_{0(1,2)}^2) \tag{5.24}
\]

and tuple for unicast hyperarc for flow 0 is

\[
M_{0(1)} = (Y_0^0 - Y_0^0\alpha, A^0 - A^0, R_{0(1)}, L_{0(1)}^1, T_{0(1)}^1) \tag{5.25}
\]

and tuple for unicast hyperarc for flow 1 is

\[
M_{0(2)} = (Y_0^1 - Y_0^0\alpha, A^1 - A^0\alpha, R_{0(2)}, L_{0(2)}^2, T_{0(2)}^2) \tag{5.26}
\]

Now, our network coding approach transmits coded packets packets only if they can be decoded at the hyperarc targets. Hence the number of unique packets of flow 0 received by $n_1$ is:

\[
Y_{1}^0 = N_c q_{0(1,2)}^{-1}/Z_{0(1,2)}^2 + N_u^0/Z_{0(1)}^1 \tag{5.27}
\]

Similarly, the number of unique packets of flow 1 received by $n_2$ is:

\[
Y_{2}^1 = N_c/Z_{0(1,2)}^2 + N_u^1/Z_{0(2)}^2 \tag{5.28}
\]

Hence $N^{nc}$ in Eqn. 5.18 can be computed to evaluate throughput for network...
coding.

5.5 Network Coding aware Rate Selection (NCRS)

In this section we present Network Coding aware Rate Selection (NCRS) - based on the analysis in the previous section. We then present two baseline schemes for comparison.

5.5.1 NCRS

The goal of our rate selection algorithm is to maximize total throughput on the targets of a hyperarc. Consider the hyperarc $0\{1, 2\}$ (i.e., multicast link from $n_0$ to $n_1$ and $n_2$). Based on Eqns. 5.27 and 5.28, the total throughput at targets of hyperarc $0\{1, 2\}$ is -

$$\frac{N_c q^\top_{0(1,2)}}{Z_{0(1,2)}^2} + \frac{N_c}{Z_{0(1,2)}^2} = \frac{N_c(1 + q^\top_{0(1,2)})}{Z_{0(1,2)}^2}$$

(5.29)

Based on Eqn. 5.29, NCRS is defined as the following for multicast on hyperarc $iJ$ in node $n_i$

$\text{NCRS}((R_{i(j)}, SNR_{ij})) = \{(\gamma^l_i, R^l)\}$ where $n_j \in J$, and $\gamma^l_i$ is solution of

Maximize $\sum_l \gamma^l_i \delta^l_i$ where $\sum_l \gamma^l_i = 1$,

$$\delta^l_i = R^l(1 + \frac{\sum_{k:\neq m, n_k \in J} q^m_{n_k,iJ}}{Z_{iJ}^m}),$$

(5.33)

$n_m$ is a direct target and is cts-node,

$$\min_j(R_{i(j)}) \leq R^l \leq \max_s(R_{i(s)}),$$

(5.35)

$n_s \in J$ and $n_s$ is a direct target

(5.36)
The formulation of NCRS is a linear programming problem to maximize throughput on hyperarc \( iJ \). It proposes to use each possible rate \( R_l \) for \( \gamma_l \) fraction of time in node \( n_i \) (cf. Eqns. 5.30-5.32). \( \delta_l \) (cf. Eqn. 5.33) corresponds to expected throughput on the hyperarc when \( R_l \) is selected as the transmission rate. We maximize total throughput at direct targets as well as overhearing targets to address the issue of overhearing tradeoff (cf. Eqn. 5.33). The cts-node is selected from among direct target nodes (cf. Eqn. 5.34).

![Graph](image)

(a) 3 nodes, Average Gain - NCRS-MinRS: 9%, NCRS-MaxRS: 24%

(b) 4 nodes, Average Gain - NCRS-MinRS: 8%, NCRS-MaxRS: 20%

(c) 5 nodes, Average Gain - NCRS-MinRS: 7%, NCRS-MaxRS: 20%

Figure 5.4. Gain over routing for packets delivered to all hops for small scenarios

We limit the maximum rate that can be selected to be from among those to direct targets only and not overhearing targets (cf. Eqns. 5.35-5.36). This is because
using a high rate to increase overhearing may be detrimental to the performance of direct targets. A lower rate can be selected to enable better overhearing.

While we use long term estimates of the channel quality, to address rate fluctuations due to variations in link conditions on smaller time-scales, we limit the possible rates to be selected to range from the minimum rate adaptation rate over all targets to the maximum rate adaptation rate among direct targets (cf. Eqns. 5.35-5.36).

Solving the NCRS formulation corresponds to picking $\gamma_i^l = 1$ for which $\delta_i^l$ is maximum. Though the problem of rate selection is based on multicast, our explicit recognition and treatment of the overhearing targets as compared to direct targets makes NCRS specific to network coding.

Note that NCRS does not account for all flows traversing through a node i.e., it does not use $NumTx()$. This simplification is needed to keep the problem tractable.

For comparison, we consider the baseline schemes as explained below.

5.5.2 MinRS and MaxRS

We define Minimum Rate Selection (MinRS) as

$$MinRS(\{ (R_{i(j)}, SNR_{ij}) \}) = \min_j (R_{i(j)}) , n_j \in J \tag{5.37}$$

MinRS selects the minimum transmission rate over the component unicast links of the hyperarc. This approach to rate selection is used in Tan et al. [84], Chou et al. [23], and Yomo et al. [94]. MinRS maximizes combined packet delivery probability on all targets - direct or overhearing. Since packet delivery probability is only a component of throughput, MinRS falls short of NCRS which maximizes throughput. MinRS results in a more balanced distribution of number of packets delivered to the targets of a hyperarc though.

MinRS permits maximum overhearing and as a consequence enables maximum coding opportunity. Still, it allows the links to operate at the highest rate which meets this condition. In COPE [48], the transmission rate was set to the minimum supported by the air interface (i.e., 6Mbps for IEEE 802.11g) in an attempt to maximize the coding gain. Since we compare network coding with rate selec-
Figure 5.5. Gain over routing for packets delivered to all hops for 3-node scenarios with varying SNR combinations - $SNR_1 = SNR$ on $n_0 \rightarrow n_1$, and $SNR_2 = SNR$ on $n_0 \rightarrow n_2$. Load = 1500 pps, Maximum Gain - NCRS-MinRS: 16%, NCRS-MaxRS: 21%

In the context of the comparison schemes against routing with rate adaptation, comparing with COPE with a constant 6Mbps transmission rate is unfair. MinRS allows the use of higher transmission rates while still preserving overhearing. Hence, comparison with MinRS is fair.

For another baseline for comparison, we define Maximum Rate Selection (MaxRS) as

$$MaxRS(\{(R_{i(j)}, SNR_{ij})\}) = \max_j(R_{i(j)}) \text{ where } n_j \in J \text{ and is a direct target} \quad (5.38)$$

MaxRS selects the maximum transmission rate over the component unicast links to direct targets of the hyperarc. Note that for MaxRS, like NCRS, the maximum rate among only direct targets is selected.

MaxRS maximizes throughput at the direct target with best SNR. This may decrease throughput at other targets of the hyperarc. The combined throughput of MaxRS over all targets of the hyperarc will be less than or equal to that for NCRS. Additionally, MaxRS may lead to a more skewed distribution of the number of packets delivered to hyperarc targets.

Note that for any hyperarc $iJ$, the same cts-node is selected for MinRS, MaxRS and NCRS to ensure a fair comparison.
5.6 Results

Simulation results in this chapter are generated using NS-2.34. We use the IEEE 802.11g MAC scheme provided by [2]. Network coding was implemented at the routing layer. The packet size is set to 1500Bytes. The receiving range is set to 85m and interference range is twice the receiving range. All simulation scenarios presented in this section are for static nodes. As a result, signal quality is stable in our simulations. Note that for all results presented in this chapter, the rate selection schemes use network coding. We compare these rate selection schemes to “routing” with rate adaptation.

First, results for the three representative small scenarios (cf. Fig. 5.3) are presented in §5.6.1. We then present results for the 3-node scenario with varying SNR combinations on the multicast hyperarc 0{1,2} in §5.6.2. Finally, we present results for varying number of flows deployed in a 50 node network in §5.6.3.

5.6.1 3, 4, 5-node Scenarios

The SNR and rates selected by rate adaptation on all unicast links are annotated in Fig. 5.3. To compare the rate selection schemes, we use the metric “gain over routing”, defined as the ratio of number of packets delivered to nodes in the network by the relevant rate selection scheme to that for routing, for the same duration. To demonstrate the gain of the rate selection schemes under different network loads, we vary the CBR load rate of the flows in Fig. 5.4. All flows in a simulation are assigned the same CBR load rate.

Simulation results for the 3, 4 and 5-node scenarios are shown in Fig. 5.4. NCRS outperforms MinRS by an average of 9%, 8%, and 7% and MaxRS by an average of 24%, 20%, and 20% for the three scenarios (in order). Note that these simple scenarios serve only as examples to illustrate differences among the rate selection schemes. Larger networks presented in §5.6.3 yield significant gain improvements.

On a side note, for the 3-node scenario MinRS outperforms routing by only 12% on average. This low gain of network coding with MinRS is due to asymmetry in link qualities connecting nodes $n_1$ and $n_2$ to $n_0$ leading to fewer coding opportunities.

For clarity, we do not plot gain over routing from the analysis in §5.4 for MinRS, MaxRS and NCRS in Fig. 5.4. Note that transmission probabilities $p_{i-s}$
Figure 5.6. Gain over routing for packets delivered to destinations for randomized grids and random flows are computed from the simulations and passed as parameters to the analysis. The average deviation of the analysis results from simulation results is 4%, 4%, and 3% over all the rate selection schemes for 3, 4, and 5-node scenarios, respectively. This small disparity between simulations and analysis validates the formulation of NCRS (cf. Eqns. 5.30-5.36) which is based on this analysis.

Table 5.2 shows the transmission rates selected by the rate selection schemes for all hyperarcs. Hyperarc 3\{0, 2\} is an example of NCRS decreasing transmission rate to promote overhearing while hyperarc 4\{0, 1\} demonstrates that NCRS does not increase transmission rate to promote overhearing.
Table 5.2. Transmission Rates [Mbps] for Hyperarcs in Small Scenarios

<table>
<thead>
<tr>
<th>Hyperarc</th>
<th>MinRS</th>
<th>MaxRS</th>
<th>NCRS</th>
<th>cts-node</th>
</tr>
</thead>
<tbody>
<tr>
<td>0{1,2}</td>
<td>24</td>
<td>54</td>
<td>36</td>
<td>2</td>
</tr>
<tr>
<td>3{0,2}</td>
<td>36</td>
<td>54</td>
<td>36</td>
<td>0</td>
</tr>
<tr>
<td>4{0,1}</td>
<td>24</td>
<td>24</td>
<td>24</td>
<td>0</td>
</tr>
</tbody>
</table>

5.6.2 Varying SNR Combinations for 3-Node Scenario

Previous results for the three small scenarios were for the same SNR combination of 15.3dB and 23.3dB for hyperarc 0{1,2}. We now explore gain over routing for the 3-node scenario (cf. Fig. 5.5) for different SNR combinations on hyperarc 0{1,2} such that the rates selected by rate adaptation on the links are different. While there are infinite such combinations possible, we show only some of them here.

Over all these SNR combinations, NCRS improves gain by up to 16% over MinRS and 21% over MaxRS. At best MinRS and MaxRS outperform NCRS by 1% in some cases. This is because NCRS governs performance on only the single multicast hyperarc but not the remaining 4 unicast links in the scenario. Out of the 41 cases considered, NCRS accrues more than 5% gain for 16 of the cases and brings an average improvement of 9% in these cases. This indicates the wide-ranging applicability of rate selection.

Overall, for some combinations MinRS outperforms MaxRS, while for others MaxRS brings more gain over routing than MinRS. Notably, for most of these SNR combinations NCRS does at least as well as the maximum of MinRS and MaxRS. This indicates the flexibility of NCRS in different network conditions.

5.6.3 Randomized Grids

We now present results for randomized 5×10 grids of nodes in Fig. 5.6. Nodes were initially separated from grid neighbors along both axes by 25m. Their location coordinates were then randomized along both axes in both directions. This randomization provides us with links connecting “grid neighbors” with SNRs ranging from 8dB to 27dB i.e., links that support 12Mbps to 54Mbps transmission rates.

Flows in this large network can either be disjoint or have common node(s)/link(s). Flows with multiple common links which are traversed in opposite directions
degenerate to a *chain of nodes* with nodes at extremes sending data to each other. Most coding opportunities stem from these chains. A chain is often used as an example for network coding in COPE [48]. For example, the 3-node scenario in Fig. 5.3(a) is a chain of 3 nodes.

We deploy 4, 8, and 12 flows randomly in different randomized grids. Each flow spans 3 to 5 nodes and is deployed either along a row or a column. Flows of less than 3 nodes do not provide inter-flow coding opportunities and flows of more than 5 nodes can not fit in a column. To provide sufficient coding opportunities, these flows are deployed in pairs as chains. With respect to Fig. 5.3, the intersection of flows can lead to 4 node or 5 node scenarios while the overlap of flows may lead to multiple 3 node scenarios.

Gain over routing in terms of number of packets delivered only to destinations for MinRS, MaxRS and NCRS is presented in Fig. 5.6. As the load or number of flows increase, congestion increases and more multicast packets vital for network coding are lost due to hidden terminal collisions at non-cts-nodes. For the three sets of flows, NCRS brings an average of 18%, 11%, and 24% more gain than MinRS and 28%, 54%, and 55% more gain than MaxRS. Crucially, NCRS outperforms both MinRS and MaxRS over all load levels.

### 5.7 Related Work

Interest in network coding has led to an array of research work [51, 66, 69] to gauge the gains that can be achieved in wireless networks. We now present an overview of research addressing network coding combined with a multi-rate MAC.

The problem of multicast tradeoff was also recognized by Yomo et al. [94]. But it assumes perfect overhearing and only addresses rate selection for the hyperarc rooted at the intersection node. In fact, its solution for rate selection for this hyperarc is the same as MinRS - albeit with a modification. Of the $n$ possible destinations, the transmission rate is selected to correspond to the $k$-th smallest SNR to the destinations. Only the $n - k + 1$ packets destined for the targets with SNR greater than or equal to the threshold SNR are coded. $k$ is dynamically adjusted depending on the conditions on the $n$ links. For coding only 2 packets, as is used in this chapter, always transmitting a coded packet is shown in [94]...
to lead to nearly the same average network capacity as dynamically adjusting $k$. Moreover, Yomo et al. [94] were aware of the overhearing requirement of network coding and left it for future work.

The issue of overhearing for network coding for COPE-like schemes has also been observed in Kim et al. [54]. It addresses the joint problem of rate adaptation and network coding for a star-network that is essentially a generalization of the small scenarios in this chapter. Given transmission rates of all sources in such a topology, only certain coding opportunities may arise due to the overhearing requirements that are met. These opportunities are analyzed in terms of clique partitioning of an undirected graph of destination nodes where these nodes are connected if each can overhear the other’s source’s transmission at the selected transmission rate. When only 2 packets can be coded by an intersection node, as is used in this chapter, and overhearing requirements are met, [54] adopts the approach used in this chapter that coding packets is better than not coding (as indicated by Yomo et al. [94] as well).

For the transmission of coded packets from the intersection node itself, Kim et al. [54] uses MinRS. Though this transmission does not require overhearing, using MinRS is sub-optimal compared to NCRS which maximizes the combined throughput on a hyperarc. In addition, for each star topology, Kim et al. [54] solves the problem of rate selection of all $n$ transmitters at the intersection node. This requires the intersection node to be made aware of link qualities between each of the $n$ sources and their $n-1$ overhearing destinations. In contrast, NCRS determines the transmission rates of all hyperarcs independently and requires very little information overhead in addition to that required by the rate adaptation and network coding schemes (cf. §5.3). Kim et al. [54] also does not show results for an IEEE 802.11 MAC or address the issue of link layer retransmissions in such a network.

While COPE [48] (and hence our network coding scheme) is an inter-session network coding scheme, MORE [21] is an intra-session network coding scheme based on the opportunistic routing and MAC scheme ExOR [18]. Afanasyev et al. [6] propose a rate selection scheme called Modrate to optimize overhearing for ExOR. Modrate minimizes the expected transmission time (extended from ETT [17]) while taking into account all possible paths a packet can take as a
result of overhearing. But a node in ExOR that overhears a packet may forward it. This is in contrast to our inter-session coding where nodes are required to overhear packets to enable coding at the intersection node and do not themselves transmit these packets in any form - native or coded. As a result, Modrate treats all downstream nodes of a flow equally and is different from our NCRS scheme which treats direct and overhearing targets differently. In fact, Afanasyev et al. [6] leave combining Modrate and COPE’s inter-session coding for future work.

Kim et al. [53] also recognize the problem of rate selection and propose different rate selection schemes for different packets - native packets that need to be overheard, native packets that do not need to be overheard, and coded packets. Rate selection for native packets that need to be overheard maximizes throughput at only the direct target while ensuring that the overhearing target receives more than $\beta$ fraction of the packets. For a 5 node scenario, the suggested mechanism to set $\beta$ leads to $\beta \approx 0.89$ i.e., the overhearing target should receive at least 89% of the packets. This limits the range of rates that can be selected. In contrast, our approach maximizes cumulative throughput at all targets without such constraints.

5.8 Summary

We address the problem of rate selection for network coding in multi-hop wireless networks with multi-rate IEEE 802.11g. We analyze the performance of network coding on a representative small scenario and propose a rate selection algorithm NCRS based on this analysis. We show results to illustrate the wide applicability of rate selection for network coding in wireless networks. Additionally, NCRS achieves a gain of up to 24% and 55% on average over MinRS and MaxRS, respectively, for large multi-hop wireless scenarios.
End-to-End Rate Selection for Opportunistic Reception in Multi-Rate Wireless Networks

In this chapter we propose an end-to-end algorithm, called NUM-RS, that is both multi-rate and opportunistic reception-aware, for selecting link transmission rates and source rates in a multi-hop multi-rate wireless network. Prior works on rate selection, including those that are opportunistic reception aware, perform rate selection on a hop-by-hop basis, attempting to maximize the throughput on each link. Our algorithm leverages the Network Utility Maximization (NUM) framework, thus providing end-to-end semantics for rate selection and proportional fairness with low overhead. By using end-to-end semantics NUM-RS considers both source rates and congestion on links used by a flow when selecting link rates. Our results show that NUM-RS increasingly outperforms contemporary hop-by-hop rate selection schemes as the number of hops in the flows increase. For example, 20% and 50% of 8-hop flows exhibit performance gains of at least 36% and 15%, respectively, in terms of data delivered to the destinations.
6.1 Introduction

IEEE 802.11 [1] is a popular wireless standard used for multi-hop wireless networks. As wireless links exhibit different and varying channel characteristics, IEEE 802.11(a/b/g) provides multiple transmission rates - ranging from 1Mbps to 54Mbps from which to choose. A higher transmission rate allows packets to be injected into the medium at a higher rate, but it may also lead to a higher bit error rate depending on the channel conditions.

Though IEEE 802.11 does not specify an algorithm to select a transmission rate for a link, or to adapt it as the channel conditions vary, this problem has received considerable attention [20, 39, 44, 82, 91]. The focus of these rate adaptation algorithms is to change transmission rate as channel characteristics vary. For any given channel condition, the component of rate adaptation that selects the transmission rate is referred to as rate selection. To the best of our knowledge, all rate selection algorithms operate in a hop-by-hop manner.

Traditional routing in wireless networks forwards packets in a hop-by-hop manner on a selected route. By virtue of the wireless medium being broadcast in nature, some transmissions may be serendipitously received multiple hops downstream with a non-negligible probability. This phenomenon, referred to as opportunistic reception, occurs frequently in a multi-rate wireless network as illustrated in Afanasyev et al. [6]. Opportunistic reception has recently been leveraged by Biswas et al. [18] and Chachulski et al. [21], among others.

In this chapter, we propose a low overhead end-to-end rate selection algorithm called NUM-RS that accounts for opportunistic reception in a multi-rate wireless network. NUM-RS uses end-to-end semantics to select both source rates and link transmission rates. It also considers tradeoffs of link occupancy time with opportunistic reception when setting link rates.

We leverage the Network Utility Maximization (NUM) framework, first proposed by Kelly et al. [49, 50] for wired networks. The incorporation of NUM enables a distributed scheme to determine the rates at which traffic for different flows should be injected onto links while maintaining proportional fairness [49]. Thus NUM-RS inherits salient attributes of NUM - feasibility, fairness, and the ability to implement a distributed, low overhead protocol.
Our contributions include -

- We extend the NUM framework as applied to wireless networks to explicitly take the characteristics of a multi-rate wireless network and opportunistic reception into account and to enable end-to-end rate selection.

- We design a protocol to implement NUM-RS in a wireless network and analyze its overhead. We conclude that NUM-RS imposes a marginal increment in overhead as compared to other NUM based schemes.

Our results show that in networks where source rates are determined using the NUM framework, existing hop-by-hop approaches for rate selection are suboptimal. NUM-RS increasingly outperforms other hop-by-hop rate selection schemes as scope for opportunistic reception increases. For example, NUM-RS brings over 36% and 15% gains in terms of data delivered to destinations in 20% and 50% of the flows, respectively, when compared to the best hop-by-hop rate selection approach. In addition, NUM-RS outperforms Modrate [6], a rate selection algorithm specifically designed for opportunistic reception.

The rest of this chapter is organized as follows: §6.2 presents a background on opportunistic reception and its primary example - ExOR [18]. We then present the details of the general opportunistic reception framework that we employ in this chapter.

6.2 Opportunistic Reception

In this section, we first provide a background on opportunistic reception and its primary example - ExOR [18]. We then present the details of the general opportunistic reception framework that we employ in this chapter.

6.2.1 Overview

Delivering data across multiple hops in wireless networks using traditional routing requires a route discovery mechanism based on a metric like estimated transmission
count (ETX) [27], estimated transmission time (ETT) [17], etc. Usually, an intermediate node in the route receives a packet from the node immediately upstream and forwards it to the node immediately downstream. Hence the next-hop node for all packets is determined during the route discovery process. This selection of next-hop node in a route attempts to strike a balance between the geographical distance covered by a transmission and the probability of its reception at the next-hop node. As a result, nodes other than the next-hop node may receive a transmission with non-negligible probability.

Consider the example shown in Fig. 6.1 where the route from source $S$ to destination $D$ uses nodes $R_1$ and $R_2$ while $R_3$ is an off-route node. The links are annotated with their respective packet loss probabilities - e.g., $R_2$ drops the direct transmissions of source $S$ with probability 0.8. The conventional routing approach fails to leverage the broadcast nature of the wireless medium in the following two ways. First, conventional routing does not leverage off-route nodes that may correctly receive a packet that is not received by any on-route node - e.g., $R_1$, $R_2$, and $D$ may not receive the transmission of $S$ but $R_3$ may receive it - this occurs with probability 0.06 in the example in Fig. 6.1. Traditional routing requires any off-route node (e.g., $R_3$) to discard these packets. So in spite of the fact that the packet may have made some progress toward the eventual destination, it is discarded altogether and retransmitted.

Second, conventional routing does not leverage the possibility that a packet may be received multiple hops downstream - e.g., $R_2$ may receive the transmission of $S$. Traditional routing may or may not allow the reception of this packet multiple hops downstream. Even if this packet is received and forwarded by this opportunistic receiver, other upstream nodes (e.g., $R_1$) may potentially receive and forward the packet as well. Subsequently, preventing the forwarding of multiple copies of an opportunistically received packet poses a problem.

6.2.2 ExOR [18]

To resolve these issues, ExOR defers the determination of the next-hop node for a packet until after it has been transmitted. For a batch of packets, the source node adds all nodes in the path up to the destination node in a forwarder list. The
nodes in this list are ordered in increasing order of distance from the destination in terms of the ETX [27] metric. After the source node has transmitted all packets in the current batch, the highest priority forwarder node in the forwarder list forwards all the packets that it has received. A batch map is included in each transmitted packet. For each packet, this batch map stores the highest priority node that may have received the packet. Based on the batch maps received, a node updates its local information and forwards only those packets which have not been forwarded by higher priority forwarder nodes. Essentially, the mechanisms of ExOR are designed to eliminate unnecessary duplicate transmissions and collisions while ensuring that each packet is forwarded by the highest priority node in the forwarder list to have received the packet.

6.2.3 Generalized Opportunistic Reception Framework

The emphasis of this chapter is to propose a rate selection algorithm for opportunistic reception in general. Hence we emulate a generalized approach to opportunistic reception. This allows our rate selection approach to be applicable to different opportunistic reception algorithms [18, 21].

Among all nodes that receive a transmission, the node that is nearest to the destination in terms of ETT [17] is selected to forward the packet next. We assume that this forwarder-selection can be performed at zero cost.

Afanasyev et al. [6] limits the nodes that can be a part of the forwarder list to those that belong to the route selected by the routing scheme. This restriction is referred to as “on-path overhearing”. We adopt the same restriction in this chapter.

Like other opportunistic reception schemes [21], we disable link layer retransmissions for data packets in this chapter. Opportunistic reception implies that if any other node closer to the destination receives the transmission, link layer retransmissions are not required. To prevent the issue of the packet being lost at all forwarder list nodes while the transmitter is left unaware, IEEE 802.11’s ACK mechanism is retained.

In wireless networks the balance between spatial reuse of medium and opportunistic reception is an issue. To prevent hidden terminal problems from eroding
opportunistic reception, we use RTS/CTS in this work - like [25, 42, 65]. A downstream node that is farthest from the destination in terms of ETT may be chosen as the target node to transmit the CTS so as to minimize the impact on spatial reuse and still avoid hidden terminal problems.

6.3 Network Utility Maximization (NUM)

The NUM approach proposed by Kelly et al. [50] was designed for a wired network. We use and extend the notation used therein in this chapter. We summarize Kelly et al.’s approach in §6.3.1. We then summarize extension of NUM to wireless networks by Xue et al. [92] in §6.3.2.

6.3.1 NUM for Wired Networks

Let $J$ be the set of links in the network with $C_j$ being the capacity of link $j \in J$. Each flow $r$ is associated with a route which is a subset of $J$ such that the links in $r$ are connected. Let the set of flows be denoted by $R$. Let $A$ be a matrix such that $A_{jr}$ is 0 if flow $r$ does not use link $j$ and 1 otherwise. Let the amount of resource allocated to flow $r$ be $x_r$ and let $x$ be the vector of these allocations. Note that flow $r$ uses $x_r$ amount of resources on each of its links and the utility of these resources to the system is $U_r(x_r)$. The problem of maximizing network-wide utility is formulated as -
where $Ax \leq C$ is the set of constraints that the sum of the resources of a link allocated to all flows that traverse it is less than the capacity of the link.

When $U_r()$ is differentiable and strictly concave, and the feasible region determined by the constraints is compact, $SYSTEM()$ is solvable using convex optimization. Kelly et al. [49] show that if $U_r()$ is a logarithmic function i.e., $U_r(x_r) = \log(x_r)$, the solution of $SYSTEM()$ is the set of proportionally fair allocations.

The core issue with this formulation is that it requires the knowledge of all $U_r$, $x_r$, matrix $A$, and vector $C$ at a centralized location in the network. This is prohibitively expensive in a multi-hop wireless network. Kelly et al. show that $SYSTEM()$ can be solved in a distributed manner via iteration by solving the problems $USER_r()$ and $NETWORK()$

### $USER_r(U_r; \lambda_r)$:

Maximize $\sum r \in R U_r(x_r)$  
subject to $Ax \leq C$. 
over $x \geq 0$. 

### $NETWORK(A,C; w)$:

Maximize $\sum r \in R w_r log(x_r)$  
subject to $Ax \leq C$. 
over $x \geq 0$. 

The $USER_r()$ subproblem is solved for each flow, with the objective of maxi-
mizing its net utility \( U_r(w_r/\lambda_r) - w_r \) where \( w_r = \lambda_r x_r \) is the price flow \( r \) is required to pay; and \( \lambda_r = dU_r/dx_r \) is the rate of price imposed by the network for the flow. The network solves the \( NETWORK() \) problem with the objective of maximizing the revenue received from the flows, defined as \( \sum_{r \in R} w_r \log(x_r) \).

The distributed algorithm is derived by making use of the Lagrangian multiplier (shadow cost) and driving the gradient to zero. Accordingly, the iterative update of \( x_r \) at the source of flow \( r \) is based on the following

\[
\frac{d}{dt} x_r(t) = \kappa [w_r - x_r(t) \sum_{j \in r} p_j(\sum_{s \in S} x_s(t))] \quad (6.4)
\]

where \( \kappa \) is a constant step-size and \( p_j \) is a cost per unit flow function on link \( j \) with the amount of load \( \sum_{s \in S} x_s(t) \) on link \( j \) as its input. The definition of the price function \( p_j() \) is adopted from Kelly et al. [50] as the following

\[
p_j(y) = (y - C_j + \varepsilon)^+/\varepsilon^2. \quad (6.5)
\]

where \( \varepsilon \to 0 \) is the part of the capacity within which when the link usage is incurred, the price of the link increases.

Note that this distributed iterative process to update \( x_r \) based on Eqn. 6.4 requires collecting \( p_j(\sum_{s \in S} x_s(t)) \) for all links \( j \) in flow \( r \). This renders NUM an end-to-end approach.

### 6.3.2 NUM for Wireless Networks

In wireless networks the capacity of a link depends on the presence of interfering links in its vicinity. Xue et al. [92] propose to use maximal cliques in \( conflict graphs \) of links to characterize wireless capacity constraints in multi-hop wireless networks. For example, the three flows in Fig. 6.2(a) lead to the conflict graph in Fig. 6.2(b) which is comprised of two maximal cliques - \( l_1 = \{n_1 \rightarrow n_2, n_3 \rightarrow n_4, n_4 \rightarrow n_5\} \) and \( l_2 = \{n_3 \rightarrow n_4, n_4 \rightarrow n_5, n_6 \rightarrow n_7\} \). While the NUM framework for wired networks assigns costs to links, Xue et al. assign costs to cliques. In addition, maximal cliques can be used for the capacity constraint component of the \( SYSTEM() \)
problem. We use the notation proposed in Eswaran et al. [31] - an extension of Xue et al. [92].

Like Xue et al. [92], we use the protocol model [35] to characterize the IEEE 802.11 wireless network in this chapter. Links that are sufficiently far apart in a wireless network may transmit concurrently. However, links within interference range are not allowed to do so. Consider a conflict graph which has nodes corresponding to all \((r, j)\) combinations of flow \(r\) and link \(j \in r\). Edges in this graph connect nodes that may not transmit concurrently. Let the set of maximal cliques in this graph be \(L\). Hence, for each maximal clique \(l \in L\), only one of the links may transmit at any time. The fraction of time the medium is busy due to transmissions of nodes of maximal clique \(l\) is

\[
b_l = \sum_{(r, j) \in l} \frac{x_r}{c_{r, j}}
\]

where \(c_{r, j}\) is the capacity of link \(j\) when operated in isolation. Eqn. 6.6 can be used to formulate the \(SYSTEM()\) problem as

\[
SYSTEM(U, A, C):
\begin{align*}
\text{Maximize} & \quad \sum_{r \in R} U_r(x_r) \\
\text{subject to} & \quad b_l \leq 1 - \delta, \quad \forall l \in L. \\
& \quad \text{over} \quad x \geq 0.
\end{align*}
\]

where \(\delta\) is the part of the wireless medium capacity unused due to backoff mechanisms employed by the CSMA/CA component of IEEE 802.11 and control packets like RTS/CTS/ACK. Bianchi et al. [15] indicate that the saturation throughput for IEEE 802.11 is 0.68, i.e., \(\delta = 0.32\).

Note that \(b_l \leq 1 - \delta\) is a convex constraint as \(c_{r, j}\) is a constant. Hence all the constraints are convex and a solution exists and can be found via the Lagrangian method. Let a flow \(r \in l\) if \(\exists\) link \(j \in r\) and \(j \in l\). The update of the source rates follows the following rule [31] -
Figure 6.2. Wireless network - Flows, Conflict Graph, and Maximal Cliques in conflict graph

\[
\frac{d}{dt} x_r(t) = k[w_r - x_r(t) \sum_{l \in \mathcal{L}, x \in l} \sum_{(r,j) \in l} \frac{p_j(b_l)}{c_{r,j}}].
\] (6.8)

where

\[
p_j(y) = \frac{(y - (1 - \delta) + \varepsilon)^+}{\varepsilon^2}.
\] (6.9)
6.4 NUM-based Rate Selection (NUM-RS)

We now present details of our NUM-RS algorithm in §6.4.1. §6.4.2 presents the baseline schemes to which we compare NUM-RS. §6.4.3 compares the overhead of NUM-RS and baseline schemes.

6.4.1 NUM-RS

Multi-rate wireless networks impact the NUM formulation in Eqns. 6.6-6.7 via $c_{r,j}$ - i.e., link capacities in isolation. While for a constant-rate wireless network all $c_{r,j}$ are assigned the same value, multi-rate wireless networks use rate selection to determine the rate at which to transmit.

Since link layer retransmissions are not used in our generalized opportunistic reception framework, the rate at which a flow’s data is injected into the network may not be the same as the rate at which it is received at the destination. We refer to the rate at which the source injects packets in the medium as source rate and the rate at which data is received at the destination as received rate. Given the set of transmission rates employed on the links of flow $r$, source rate and received rate are related by a factor $\beta_r$ explained below.

Consider the example shown in Fig. 6.3 where a flow sends data from source $S$ to destination $D$ through multiple links. We define a link by its source node and immediate downstream node i.e., the intended next-hop target for conventional routing. For example, link $j$ connects intermediate nodes $R_a$ and $R_b$. For simplicity, let $j - 1$ denote the link immediately upstream of link $j$ and $j + 1$ denote the link immediately downstream.

Let the received rate for flow $r$ at destination $D$ be $x_r$ (cf. Fig. 6.3). Out of the multiple discrete transmission rates provided by IEEE 802.11, let transmission rate index $z$ (starting from 0) denote the $z$-th largest transmission rate available. Hence, let the index for transmission rate selected on link $j$ of flow $r$ be $z_{r,j} \in [0, Z - 1]$ where $Z$ is the number of transmission rates available. Let throughput on link $j$ for transmission rate index $z_{r,j}$ be $c_{r,j}(z_{r,j})$ and the packet error rate be $e_{r,j}(z_{r,j})$. While $z_{r,j}$, $e_{r,j}$ and $c_{r,j}$ are discrete variables, they can be adapted into continuous variables using linear interpolation.

Let $j < k$ denote that for flow $r$, $k$ is a downstream link of $j$ - e.g., link $k$
in Fig. 6.3 is a downstream link of link \( j \). Let \( e_{r,j,k}(z) \) denote the packet loss probability of the transmission of source of link \( j \) at the destination of link \( k \) at transmission rate index \( z \) where both links belong to the same flow \( r \) - as illustrated in Fig. 6.3. Hence \( (1 - e_{r,j,k}(z)) \) is the probability of opportunistic reception of \( R_a \)’s transmissions at \( R_t \).

Nodes in opportunistic reception forward only those packets that have not been received by any downstream nodes. The probability of destination of a downstream link \( k \geq j \) receiving a packet that is not received by any further downstream link \( l > k \) is

\[
    p_{r,j,k}(z) = (1 - e_{r,j,k}(z)) \cdot \prod_{l>k} e_{r,j,l}(z) \quad (6.10)
\]

where it is assumed that the reception of a transmission at different nodes is governed by independent error rates - as observed in [73, 81]. While \( (1 - e_{r,j,k}(z)) \) denotes the probability of destination of link \( k \) receiving a packet from source of link \( j \), \( \prod_{l>k} e_{r,j,l}(z) \) is the probability of no nodes downstream of link \( k \)’s destination receiving this packet.

The fraction of the source rate that an intermediate node has to forward is the sum of the transmissions of its upstream nodes that have been received by the node but not by any other downstream nodes. Let \( \alpha_{r,j} \) be the fraction of the source rate that is forwarded by the source of link \( j \). For the first link \( i \) of flow \( r \), \( \alpha_{r,i} = 1 \). Hence \( \alpha_{r,j} \) is defined as -
\[ \alpha_{r,j} = \sum_{k<j} \alpha_{r,k} \cdot p_{r,k,j-1}(z_{r,k}) \quad (6.11) \]

where \((j-1)\) refers to the link immediately upstream of link \(j\) and \(\alpha_{r,k} \cdot p_{r,k,j-1}(z_{r,k})\) is the fraction of the data received from the source of link \(k\) that needs to be forwarded by source of link \(j\) (cf. Fig. 6.3).

The destination of the flow may receive transmission from all the links in the flow. This received rate \(x_r\) for flow \(r\) is related to the source rate by the fraction \(\beta_r\) (cf. Fig. 6.3) -

\[ \beta_r = \sum_{j \in r} \alpha_{r,j} \cdot (1 - e_{r,j,u}(z_{r,j})) \quad (6.12) \]

where \(u\) is the last link in flow \(r\), i.e., the target of link \(u\) is the same as the destination of flow \(r\). \(\alpha_{r,j} \cdot (1 - e_{r,j,u}(z_{r,j}))\) denotes the fraction of transmissions received by the destination node due to transmissions from link \(j\)’s source (cf. Fig. 6.3).

Hence, given the transmission rates of all forwarder nodes, the source rate can be determined from the received rate. This is important because in the NUM formulation we solve for transmission rates and received rates. We then determine the source rates for the flows from the solution of NUM.

Now, the fraction of time that a medium is busy for a maximal clique \(l\) is

\[ b_l = \sum_{(r,j) \in l} \frac{x_r \cdot \alpha_{r,j}}{c_{r,j} \cdot \beta_r} \quad (6.13) \]

where \(x_r\) is the received rate of flow \(r\), \(x_r/\beta_r\) is the source rate of flow \(r\), and \(x_r \cdot \alpha_{r,j}/\beta_r\) is the rate at which the source of link \(j \in r\) transmits. Hence \(x_r \cdot \alpha_{r,j}/(c_{r,j} \cdot \beta_r)\) is the fraction of time that the medium is occupied due to transmission of link \(j\)’s source and \(b_l\) is the fraction of the time that the medium is busy in the vicinity of links in maximal clique \(l\).

Thus our system of equations is -
\(\text{SYSTEM}(U, A, C)\):

Maximize \(\sum_{r \in R} U_r(x_r)\)

subject to \(b_l \leq 1 - \delta, \forall l \in L\).

over \(x \geq 0\),

\(z_{r,j} \geq 0 \forall r, j \in r\),

\(z_{r,j} \leq \arg \min_y \{e_{r,j}(y) \neq 1\} \forall j \in r\).

where \(b_l \leq 1 - \delta\) states that the total fraction of the medium that can be used due to transmissions of links in maximal clique \(l\) should be less than \(1 - \delta\), and \(z_{r,j} \leq \arg \min_y \{e_{r,j}(y) \neq 1\} \forall j \in r\) states that the transmission rate for any link should be less than the minimum rate that fails to deliver any packets to the next-hop node. Note that if a transmission rate used by source of link \(j\) does not deliver any packets to the destination of link \(j\), it will not deliver packets to any other downstream nodes either. Hence the constraint on \(z_{r,j}\) does not limit opportunistic reception.

This end-to-end approach adopted by NUM-RS allows it to consider the utilization of the medium in different parts of the network through the maximal clique medium use constraint. In contrast, hop-by-hop rate selection schemes are oblivious of medium utilization and simply maximize some measure of link throughput.

The constraint \(b_l \leq 1 - \delta\) may not be convex as \(b_l\) combines multiple functions of several \(x_r\)-s and \(z_{r,j}\)-s. As a result, the feasible region may not be compact and a unique solution to \(\text{SYSTEM}()\) may not exist. Moreover, a Lagrangian based approach may only lead to a local maxima - not a global maxima. We assume though that \(\text{SYSTEM}()\) can indeed be solved by convex optimization and leave the proof along lines adopted in Lee et al. [67] and Wang et al. [90] for future work. We will explore convex transformations of the constraint \(b_l \leq 1 - \delta\) like [67, 76, 90] in future work.

Now, consider the Lagrangian -
\[ L(x, z, v; \mu) = \sum_{r \in R} U_r(x_r) - \sum_{l \in L} \mu_l \cdot (b_l - (1 - \delta) - v_l) \quad (6.15) \]

where \( v \) is a vector of slack variables and \( \mu \) is a vector of Lagrangian multipliers.

Hence at a maxima of \( L() \) we have -

\[
\frac{d}{dt} x_r(t) = \kappa_1 \cdot x_r \left( \frac{\partial U_r(x_r)}{\partial x_r} - \sum_{l \in L, r \in l} \mu_l \sum_{(r,j) \in l} \frac{\alpha_{r,j}}{c_{r,j} \cdot \beta_r} \right) \quad (6.16)
\]

\[
\frac{d}{dt} z_{r,j}(t) = \kappa_2 \cdot z_{r,j} \left( \sum_{l \in L, r \in l} \mu_l \cdot x_r \sum_{(r,j) \in l} \gamma \right) \quad (6.17)
\]

where \( \gamma \) is -

\[
\gamma = \frac{\alpha_{r,j}}{c_{r,j}^2 \cdot \beta_r} \frac{\partial c_{r,j}}{\partial z} + \frac{\alpha_{r,j}}{c_{r,j} \cdot \beta_r^2} \cdot \frac{\partial \beta_r}{\partial z} - \frac{1}{c_{r,j} \cdot \beta_r} \frac{\partial \alpha_{r,j}}{\partial z} \quad (6.18)
\]

and \( \mu_l \) is the shadow cost of congestion charged at each clique \( l \) defined as -

\[
\mu_l(t) = p_l \left( \sum_{\forall (r,j) \in l} x_r \cdot \frac{\alpha_{r,j}}{c_{r,j} \cdot \beta_r} \right) \quad (6.19)
\]

where \( p_l \) is from Eqn. 6.9. \( \kappa_1 \) is the step-size for received rates and \( \kappa_2 \) is the step size for transmission rates.

### 6.4.2 Baselines for Comparison

Our NUM-RS algorithm improves on NUM for wireless networks (cf. §6.3.2) by making it multi-rate and opportunistic reception aware. We compare NUM-RS to three schemes in which transmission rates are determined using hop-by-hop rate selection algorithms. To keep the comparisons fair, the received rates in baseline schemes are determined using a multi-rate and opportunistic reception aware NUM
- a straightforward adaptation of NUM-RS which is here on referred to as NUM’
- while keeping the transmission rates fixed. Hence the baseline schemes perform
rate selection hop-by-hop, not end-to-end as in NUM-RS. The convergence and
optimality properties of the regular NUM framework [49, 50] apply to the baseline
schemes.

The baseline schemes are -

6.4.2.1 ORU-RS

Rate selection approaches like RBAR [39] that do not envisage opportunistic re-
ception select a transmission rate so as to maximize link throughput i.e.,

\[ z_{r,j}^* = \arg \max_z \{ c_{r,j}(z) \cdot (1 - e_{r,j}(z)) \} \] (6.20)

where \( z_{r,j}^* \) is the transmission rate selected, \( c_{r,j}(z) \) is the rate at which packets are
injected into the medium and \( (1 - e_{r,j}(z)) \) is the probability of packet reception at
the destination of link \( j \) when the transmission rate index is \( z \).

This approach leads to opportunistic reception unaware rate selection (referred
to as \( ORU-RS \)). Note that opportunistic reception is still enabled in spite of the
rate selection scheme being oblivious of it.

6.4.2.2 Modrate

Modrate [6], an extension of ETT [17], is opportunistic reception aware and adopts
the following approach for rate selection -

\[ z_{r,j}^* = \arg \min_z \{ \sum_{k \geq j} \frac{1}{c_{r,j}(z) \cdot p_{r,j,k}(z)} \} \] (6.21)

where \( z_{r,j}^* \) is the transmission rate selected, \( 1/c_{r,j}(z) \) is the time required to transmit
a single packet and \( p_{r,j,k}(z) \) is the probability of destination of link \( k \) receiving the
transmission of source of link \( j \) such that nodes downstream of link \( k \)’s destination
do not receive it.
6.4.2.3 ORA-RS

Modrate does not differentiate between packets that are delivered one hop away, or opportunistically delivered across several hops. We propose a hop-by-hop rate selection scheme that maximizes the cumulative “distinct” throughput received at all next-hop nodes weighted by the progress in terms of hops \(i.e.,\)

\[
z^*_r,j = \arg \max_z \left\{ \sum_{k \geq j} (k - j + 1) \cdot c_{r,j}(z) \cdot p_{r,j,k}(z) \right\}
\]

(6.22)

where \(z^*_r,j\) is the transmission rate selected, and \((k - j + 1)\) represents the number of links between the source of link \(j\) and destination of link \(k\). \(c_{r,j}(z) \cdot p_{r,j,k}(z)\) is the “distinct” throughput of the source of link \(j\)’s transmission received at link \(k\)’s destination \(i.e.,\) packets not received by any other downstream node. We refer to this opportunistic reception aware rate selection approach as ORA-RS.

6.4.3 Overhead

The maximal clique approach for NUM in wireless networks, explored in detail by Xue et al. [92] and Eswaran et al. [31], includes distributed discovery of maximal cliques and asynchronous message exchanges. Each node exchanges information with nodes up to two hops away. This information is then used to infer the sets of interfering links, enabling the construction of local conflict graphs. Since clique construction depends on network topology as well as channel conditions, it may need to be performed on multiple occasions. The frequency of this protocol depends on the extent of dynamic nature of the network. Conversely, this protocol can also be invoked on-demand, getting called only if a change in topology is detected by the nodes.

NUM’ and NUM-RS require per-flow information (\(e.g.,\) source rates) as well as per-link information (\(e.g.,\) transmission rates). Per-flow information may be stored at the source or destination of the flow, while per link information is stored at the source node of a link. Per flow computation (\(e.g.,\) update of received rate) and per link computation (\(e.g.,\) update of transmission rates) is similarly performed by flow source/destination and source of a link, respectively. To keep the overhead
Regular NUM, as well as NUM’ and NUM-RS, require a “dissemination in maximal cliques” mechanism where a link $j$ needs to inform every interfering link (i.e., links in any maximal clique of which it is a member) of some variable $\rho_{r,j}$. This is accomplished in two ways. First, nodes overhear all transmissions in the neighborhood. Second, in addition to its own information $\rho_{r,j}$, a node randomly selects a few interfering links’ information $\rho_{r,k}$ and piggybacks those onto data packets as well. This provides a way to disseminate information up to two hops away at a low overhead.

Like NUM, NUM’ and NUM-RS also require a “cumulative downstream propagation” mechanism. Say every link $j$ in flow $r$ has a variable $\rho_{r,j}$. This downstream propagation requires link $j$ to forward the value of $\sum_{k \leq j} \rho_{r,k}$ to its immediate downstream link. This cumulative propagation imposes an overhead of only a single variable on each link of the flow.

Since NUM is an iterative algorithm, NUM’ and NUM-RS are iterative algorithms as well. NUM’ iteratively updates received rates (which subsequently determine source rates), while NUM-RS updates both received rates and transmission rates. Some of the overhead incurred by NUM’ and NUM-RS is inherited from NUM and is summarized in §6.4.3.1. The overhead increment incurred by NUM-RS compared to regular NUM can be attributed to its multi-rate and op-

Figure 6.4. Packet error rate for 1500Byte packets with IEEE 802.11a transmission rates
Table 6.1. Throughput [pps] for IEEE 802.11a transmission rates for 1500Byte packets

<table>
<thead>
<tr>
<th>Mbps</th>
<th>6</th>
<th>9</th>
<th>12</th>
<th>18</th>
<th>24</th>
<th>36</th>
<th>48</th>
<th>54</th>
</tr>
</thead>
<tbody>
<tr>
<td>PPS</td>
<td>416</td>
<td>586</td>
<td>736</td>
<td>990</td>
<td>1200</td>
<td>1512</td>
<td>1744</td>
<td>1834</td>
</tr>
</tbody>
</table>

portunistic reception aware capabilities. This overhead increment is incurred by NUM’ as well and is summarized in §6.4.3.2.

6.4.3.1 Overhead Inherited from NUM

- Clique construction and the concomitant overhead is common to NUM, NUM’ and NUM-RS.

- In NUM-RS, Each link needs to compute the congestion cost according to Eqn. 6.19. This requires a link \( j \in r \) to disseminate \( x_r \cdot \alpha_{r,j} / (c_{r,j} \cdot \beta_r) \) in all maximal cliques of which it is a member. This is analogous to exchanging \( x_r/c_{r,j} \) in regular NUM. *Hence this overhead component is same for NUM, NUM’ and NUM-RS.*

- In NUM-RS, once every link is aware of \( x_r \cdot \alpha_{r,j} / (c_{r,j} \cdot \beta_r) \) of every link \( j \) that it interferes with, it performs a cumulative downstream propagation of the congestion cost (cf. Eqn. 6.19) to enable update of \( x_r \) (cf. Eqn. 6.16). Similar actions are required by regular NUM and NUM’. *Hence, all three schemes incur this overhead.*

6.4.3.2 Overhead Increment Due to Multi-Rate and Opportunistic Reception Awareness

- Each link \( j \) in flow \( r \) needs to be aware of \( p_{r,k,j-1}(z_{r,k}) \) where \( k \) is an upstream link. During route discovery, a source node of a link can keep track of the link quality (*e.g.*, in terms of SNR) to every other downstream node. This enables link \( k \) to compute \( p_{r,k,j-1}(z_{r,k}) \) and convey it to link \( j \). *This overhead is incurred by both NUM’ and NUM-RS.*

- Each link \( j \) in flow \( r \) needs to compute \( \alpha_{r,j} \). This requires a cumulative downstream propagation up to link \( j \) of \( \alpha_{r,k} \cdot p_{r,k,j-1}(z_{r,k}) \) of upstream links \( k \in r \). Hence in a \( h \) hop flow, \( h \) cumulative propagations are needed. *This
overhead applies once to NUM’ and per-iteration to NUM-RS due to its end-to-end rate selection approach.

- The destination node of flow $r$ needs to be aware of $\beta_r$ (cf. Eqn. 6.12). This is calculated by cumulative downstream propagation of $\alpha_{r,j} \cdot (1 - e_{r,j,k}(z_{r,j}))$ where $k$ is the last hop in the flow. This imposes an overhead of a single variable per link for NUM’ and NUM-RS.

- Once $x_r$ is updated in the previous step by the destination node, it is propagated upstream by each link as $x_r / \beta_r$. While this information is being propagated upstream, transmission rates of links are updated based on Eqn. 6.18 at no extra communication overhead. Again, both NUM’ and NUM-RS incur this overhead per iteration.

Effectively, our approach of jointly solving for transmission rate and received
Table 6.2. Example: Packet Error Rates (PER) for Transmission Rates

<table>
<thead>
<tr>
<th>PER ↓</th>
<th>6</th>
<th>9</th>
<th>12</th>
<th>18</th>
<th>24</th>
<th>36</th>
<th>48</th>
<th>54</th>
</tr>
</thead>
<tbody>
<tr>
<td>n₁ → n₂</td>
<td>0</td>
<td>0.08</td>
<td>0.51</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>n₂ → n₃</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.93</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>n₂ → n₄</td>
<td>0.06</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>n₃ → n₄</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.16</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>n₄ → n₅</td>
<td>0.14</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>n₅ → n₆</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.02</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

rate in the NUM-RS formulation imposes only a small overhead increment on Xue et al.’s [92] approach.

6.5 Results

In this section we present an evaluation of NUM-RS. We compare its performance in terms of received rate against ORU-RS, Modrate, and ORA-RS. We first discuss the results of a 5-hop scenario to illustrate the convergence of the distributed iterative baseline and NUM-RS schemes (cf. §6.5.1). We then present results for a single flow comprised of different number of hops in §6.5.2. §6.5.3 shows results for multiple such competing flows deployed in a vicinity.

We use NS-3 [4] as the simulation platform. The wireless MAC protocol is IEEE 802.11a. IEEE 802.11a provides 8 transmission rates ranging from 6Mbps to 54Mbps - a subset of the transmission rates available in IEEE 802.11g. The IEEE 802.11a model used in NS-3 is explained in Lacage et al. [61]. Table 6.1 shows the throughput of these transmission rates at very high SNRs for 1500 Byte packets in packets-per-second (referred to as pps hereon). Fig. 6.4 shows the packet error rate vs. SNR for IEEE 802.11a. Note that only the data packet is transmitted at the given rate; control packets may not be transmitted at the same rate as data packets.

Node locations in all simulations are generated randomly while keeping the deployment connected. The source and destination of each flow are randomly chosen. Like Srcr in Roofnet [17], the route from a source to destination is determined using the ETT [17] metric. As observed in Afanasyev et al. [6], ETT calculation
expects a certain transmission rate to be selected for any link. But like Afanasyev et al., we may select a different transmission rate for the source of a link due to the rate selection scheme employed.

Since we measure the capacity of the medium in terms of the rate at which packets can be injected into the medium (but not necessarily received), IEEE’s backoff is already accounted for in our setting. Hence we set $\delta = 0$ in this work.

6.5.1 Example

![Graphs of Cumulative Fraction of Scenarios vs. RecvdRate Gain](image)

(a) NUM-RS vs. ORU-RS
(b) NUM-RS vs. Modrate
(c) NUM-RS vs. ORA-RS

Figure 6.6. Single Flow - Received Rate Gain

We now present results for a 5-hop flow from node $n_1$ to $n_6$ via $n_2$, $n_3$, $n_4$, and $n_5$ (in order) to illustrate the convergence properties of the distributed iterative NUM’ and NUM-RS schemes.

For all relevant links, the packet error rate for different transmission rates is shown in Table 6.2. Links like $n_1 \rightarrow n_3$ that do not deliver any data at any rate,
Figure 6.7. Single Flow - Avg. Weighted Transmission Rate Ratio

\[ i.e., \ e_{r,j}(z) = 1 \text{ for all } z \text{ are omitted.} \] The only transmission rate that results in opportunistic reception is transmission on \( n_2 \rightarrow n_4 \) at 6Mbps which results in packets being dropped with probability 0.06.

For the baseline schemes, NUM’ is used to determine only the received rates. The transmission rate selected by the baseline schemes (and NUM-RS) are shown in Table 6.3.

The convergence of the received rate for the flow using the baseline schemes is shown in Fig. 6.5(a) (all curves overlap). The convergence properties of the regular NUM framework apply directly to the baseline schemes. As a result, the convergence of the distributed iterative approach to the centralized solution is nearly perfect. For this example, all the baseline schemes select the same set of transmission rates and hence they determine the same received rate - 123pps.

NUM-RS determines both received rates and transmission rates using the NUM framework. The convergence of received rates and transmission rates for the 5 links
Table 6.3. Example: Transmission Rates [Mbps] for Baselines and NUM-RS

<table>
<thead>
<tr>
<th>Scheme</th>
<th>Node</th>
<th>$n_1$</th>
<th>$n_2$</th>
<th>$n_3$</th>
<th>$n_4$</th>
<th>$n_5$</th>
</tr>
</thead>
<tbody>
<tr>
<td>ORU-RS</td>
<td>9</td>
<td>18</td>
<td>24</td>
<td>6</td>
<td>18</td>
<td></td>
</tr>
<tr>
<td>Modrate</td>
<td>9</td>
<td>18</td>
<td>24</td>
<td>6</td>
<td>18</td>
<td></td>
</tr>
<tr>
<td>ORA-RS</td>
<td>9</td>
<td>18</td>
<td>24</td>
<td>6</td>
<td>18</td>
<td></td>
</tr>
<tr>
<td>NUM-RS</td>
<td>6</td>
<td>6</td>
<td>6</td>
<td>6</td>
<td>6</td>
<td></td>
</tr>
</tbody>
</table>

for NUM-RS is shown in Figs. 6.5(b) and 6.5(c), respectively. Note that Fig. 6.5(c) plots transmission rate index (cf. §6.4.1) vs. iteration. Transmission rate index of 0 corresponds to 6Mbps, 1 corresponds to 9Mbps, and so on. Vitally, while none of the baseline schemes are able to provide opportunistic reception due to the transmission rates they select (cf. Table 6.3), NUM-RS’ choice of transmission rates do enable opportunistic reception of node $n_2$’s transmissions at node $n_4$.

For NUM-RS, the received rate variable converges to 133pps and is verified by solving $SYSTEM()$ (cf. Eqn. 6.14) in Matlab. We refer to the ratio of the total received rates of all flows, i.e., $\sum_r x_r$, for NUM-RS to that for the baseline scheme as gain. Hence for this example, the gain of NUM-RS over all the baseline schemes is $1.08 \times$.

In the distributed iterative process the received rate variables are initialized to 0 for the baseline schemes as well as NUM-RS. To speed up the convergence of the received rate $x_r$, initially received rate step size (i.e., $\kappa$ for baseline schemes or $\kappa_1$ for NUM-RS) is set to 2.0. When an iteration decrements the received rate $x_r$ for the first time, this step size is reduced to 0.02 to avoid unnecessary oscillations. For NUM-RS, the transmission rates are initialized to values corresponding to those selected by ORA-RS and the transmission rate step size (i.e., $\kappa_2$) is set to 0.02. Fig. 6.5 shows that for the baseline schemes as well as NUM-RS, the distributed iterative process converges to the optimal value in less than 500 iterations.

The results from this example lead to three conclusions - i) the iterative NUM’ indeed quickly converges to the optimal for the baseline schemes; ii) in spite of the fact that the constraint on maximal cliques’ medium use in Eqn. 6.14 is not convex, the NUM-RS framework can still be employed to approach the optimal in terms of received rate; iii) the delay in convergence for the iterative NUM-RS approach is nearly the same as that for NUM’.

The number of maximal cliques in this 5-hop flow is $2 - l_1 = \{n_1 \rightarrow n_2, n_2 \rightarrow$
Table 6.4. $b_t$ of maximal cliques for a 8-hop flow

<table>
<thead>
<tr>
<th>Scheme</th>
<th>Clique →</th>
<th>$l_1$</th>
<th>$l_2$</th>
<th>$l_3$</th>
<th>$l_4$</th>
</tr>
</thead>
<tbody>
<tr>
<td>ORU-RS</td>
<td></td>
<td>0.60</td>
<td>0.99</td>
<td>0.93</td>
<td>0.90</td>
</tr>
<tr>
<td>Modrate</td>
<td></td>
<td>0.60</td>
<td>0.99</td>
<td>0.93</td>
<td>0.90</td>
</tr>
<tr>
<td>ORA-RS</td>
<td></td>
<td>0.57</td>
<td>0.99</td>
<td>0.92</td>
<td>0.77</td>
</tr>
<tr>
<td>NUM-RS</td>
<td></td>
<td>0.90</td>
<td>0.99</td>
<td>0.93</td>
<td>0.99</td>
</tr>
</tbody>
</table>

$n_3, n_3 \rightarrow n_4, n_4 \rightarrow n_5$ and $l_2 = \{n_2 \rightarrow n_3, n_3 \rightarrow n_4, n_4 \rightarrow n_5, n_5 \rightarrow n_6\}$. For these cliques, the fraction of the time that the medium is occupied *i.e.*, $b_t$ (cf. Eqn. 6.13), sheds light on the internals of these rate selection schemes. For all the baseline schemes, $b_{l_1} = 0.99$ and $b_{l_2} = 0.80$. In contrast, for NUM-RS, $b_{l_1} = 0.99$ and $b_{l_2} = 0.90$. Hence, for maximal clique $l_2$, NUM-RS better utilizes the medium. This is an outcome of the incorporation of rate selection with the $b_t$ constraint (cf. Eqn. 6.14) - a shortcoming of the baseline schemes.

Since we have shown that the simulated version of the protocol and the results obtained by Matlab coincide, and because the simulations are very time-consuming, for the remainder of the chapter we present Matlab results.

### 6.5.2 Single Flow

Since the opportunity to leverage opportunistic reception is dependent on the number of hops in a flow, we now simulate a single flow with the number of hops ranging from 2 to 8. Single hop flows do not offer any scope for opportunistic reception and flows longer than 8 hops may not occur frequently in multi-hop wireless networks.

The gain observed for different random scenarios varies significantly. Hence we present the CDF of the gain over each baseline scheme in terms of received rate in Fig. 6.6. Overall, as the number of hops increases, the fraction of flows that incur a gain increases. This is because longer flows offer more transmission rates to tune, increase the impact of opportunistic reception, and are comprised of more maximal cliques. Moreover, the gain of NUM-RS over each of the baseline schemes is nearly the same. Compared to Modrate, for 50%, 40%, 30%, 20%, and 10% of 8-hop flows, NUM-RS brings at least 15%, 19%, 26%, 36%, and 54% gain, respectively.
Figure 6.8. Single Flow - Normalized number of Hops

For one particular 8-hop flow, the received rate for NUM-RS, ORU-RS, Mod-rate, and ORA-RS is approximately 146, 89, 89, and 85pps, respectively. Hence NUM-RS brings a gain of 1.6× over Modrate for this example. We present $b_i$, the fraction of the medium used by the transmissions of links in the 4 maximal cliques that comprise this flow, in Table 6.4. These maximal cliques are ordered in increasing distance from the source of the flow. The largest discrepancy between $b_i$-s for NUM-RS and Modrate (and other baseline schemes) occurs for maximal cliques $l_1$ and $l_4$. $b_{l_1}$ for NUM-RS is 50% higher than that for the other schemes. This shows that even though the baseline schemes may have same $b_i$ for some of the maximal cliques in the network, the fact that they fall short for other maximal cliques impacts overall performance negatively.

For each scenario, we plot the CDF of the ratio of average weighted transmission rate i.e., $\sum_{j \in r} \alpha_{r,j} \cdot c_{r,j}(z_{r,j})$, used by NUM-RS to that used by the baseline scheme in Fig. 6.7. This represents the average transmission rate of all packets. While the gains of NUM-RS over all baseline schemes were similar (cf. Fig. 6.6), it is not so with the weighted transmission rate ratio (cf. Fig. 6.7). In fact, among the baseline schemes, ORU-RS selects the highest weighted transmission rates and ORA-RS selects the lowest weighted transmission rates - as observed by comparing Figs. 6.7(a), 6.7(b), and 6.7(c). This is as expected because ORA-RS may select lower transmission rates to encourage opportunistic reception while ORU-RS does not take opportunistic reception into account.
Depending on opportunistic reception, packets of a flow can take different routes from the source to the destination. To illustrate this, we present the normalized number of average hops that a packet is expected to traverse to reach the destination \( i.e., \) the ratio of expected number of hops to maximum number of hops to the destination, in Fig. 6.8. Overall, as the number of hops increases, the normalized average number of hops decreases. This is because longer flows may lead to more opportunistic reception. Remarkably though, NUM-RS’ selection of transmission rates leads to the largest number of normalized hops - \( i.e., \) NUM-RS leverages opportunistic reception the least. In addition, all of the baseline schemes have nearly the same normalized number of hops. So while we would normally expect ORU-RS to have the highest number of normalized hops - \( i.e., \) the least impact of opportunistic reception, it is NUM-RS that uses opportunistic reception the least. This indicates that even when a rate selection scheme is not opportunistic reception aware - \( e.g., \) ORU-RS, it still fails to outperform NUM-RS.

While Fig. 6.7 shows average weighted transmission rate ratio for all transmitted packets - including those that are lost en route to the destination, Fig. 6.8 shows average number of hops traversed for only packets that are delivered to the destination. This is why in spite of NUM-RS often selecting a lower transmission rate than the baseline schemes (cf. Fig. 6.7), the packets that do reach the destinations follow longer routes on average (cf. Fig. 6.8).

For less than 5% of the scenarios NUM-RS is outperformed by the baseline schemes (cf. Fig. 6.6). Overall though, these results show that the hop-by-hop approach to rate selection may not lead to the best performance and the end-to-end approach of NUM-RS performs best.

### 6.5.3 Multiple Flows

We now present results for the performance of multiple flows such that nodes that comprise the flows form a connected conflict graph in terms of interference. As the number of flows within interference range increases, load in terms of traffic increases. Fig. 6.9 shows the CDF of gains over the baseline schemes as the number of flows in the deployment range from 1 to 3. Each flow spans 2 to 8 hops. As the number of flows increases, the gains decrease. This is because only a subset
of the flows may bring a substantial gain and this gain is then amortized over a larger denominator. Once again, NUM-RS outperforms Modrate as well as other baseline schemes.

![Graphs showing the performance of NUM-RS vs. ORU-RS, NUM-RS vs. Modrate, and NUM-RS vs. ORA-RS.](image)

(a) NUM-RS vs. ORU-RS  
(b) NUM-RS vs. Modrate  
(c) NUM-RS vs. ORA-RS

**Figure 6.9.** Multiple Flows - Received Rate Gain

### 6.6 Related Work

Opportunistic reception has been leveraged in a broad swathe of works like Lane-
man et al. [64, 63] under the umbrella of cooperative diversity. These schemes enable all nodes that receive a packet to relay it to the next-hop. But these approaches suffer from the issue of duplicate transmissions - an issue addressed by ExOR [18]. The cost of ExOR’s transmission ordering and deferment mechanisms to avoid collisions and duplicate transmissions may be significant. This cost is amortized over batches of packets and hence the emphasis that ExOR is a bulk
transfer protocol.

ExOR’s synchronization requirements lead to the restriction that only one of the nodes on the forwarder list can transmit at any instant. This robs wireless networks of a vital feature - spatial reuse of the medium. This issue was remedied by MORE [21] which combines network coding [7] and opportunistic reception. Like ExOR, MORE adopts a batch transmission approach. Intermediate nodes in MORE transmit a random linear combination of all linearly independent packets received for a batch. Once the destination receives as many linearly independent combinations of packets as the batch size, it solves for the native packets and sends an ACK to the source for the entire batch. Unlike ExOR, MORE does not schedule the transmission of intermediate nodes. Instead, it allows a node to transmit a random linear combination only if it has received a linearly independent combination itself.

ExOR and MORE have been evaluated in fixed rate wireless networks. Afanasiev et al. [6] propose Modrate - a rate selection mechanism to maximize overhearing (i.e., opportunistic reception) in multi-rate wireless networks. Modrate maximizes network throughput by jointly optimizing rate selection and overhearing. But it fails to elicit significant improvements from ExOR. This is because the gains of ExOR over traditional routing are largely due to the batch transmission of packets and the use of batch-map in packet headers as bulk-ACKs.

Lee et al. [67] also propose combining NUM with rate selection. The authors recognize the non-convexity of the resulting constraints. These constraints are then transformed into convex versions and a distributed algorithm using the Lagrangian approach is used. The authors also establish the conditions under which the distributed formulation converges. The authors assume that the error probabilities on the component links of a flow are small - an important assumption to keep the constraints convex. But links in wireless networks often have high error rates - as evidenced in Fig. 6.4. Additionally, Lee et al. do not address opportunistic reception. The non-convex constraint in our problem formulation cannot simply employ Lee et al.’s transformations as $b_t$ is a non-linear combination of functions of several variables. Like Lee et al. though, we will explore convex transformations of the NUM-RS problem formulation in future work.
6.7 Summary

In this chapter, we propose NUM-RS - a low-overhead, end-to-end rate selection scheme for a multi-rate wireless network that leverages opportunistic reception and utilizes the NUM framework. Our results show that solving for transmission rates in addition to received rates within the NUM framework leads to substantial gains over three baseline schemes - ORU-RS, Modrate [6], and ORA-RS.
Conclusions

The focus of this dissertation is to address the issue of congestion in wireless networks. Towards this goal, my research takes five important steps in this dissertation (cf. §7.1). Finally, avenues for future work are explored in §7.2.

7.1 Summary of Contributions

- In Chapter 2 [57], we recognize that sensor networks that deliver data of different priority categories - low and high, face performance issues on encountering congestion. This congestion is indiscriminate and may allocate network resources for low priority data instead of high priority data. To assuage this problem, we propose a differentiated routing mechanism called Congestion Aware Routing (CAR). CAR discovers the congestion zone (conzone) and only routes high priority data through it. Low priority data generated outside the conzone is routed to its destination using off-conzone nodes while that generated inside is efficiently routed out of the conzone.

- In Chapter 3 [56], intra-flow network coding of multicast flows in wireless networks is shown to suffer performance degradation when congestion occurs. We propose the concept of selective channelization, analyze the performance of routing and intra-flow network coding for a hyperarc, and propose two channelization schemes to address congestion - count based, and model based. While the count based scheme accurately tracks the performance of network
coding, it is prone to false positives due to link errors. To counter this, the model based scheme uses our analysis of network coding’s performance on a hyperarc to trigger channelization. Overall, selective channelization restores the performance of network coding when faced with congestion.

- While selective channelization [56] considered the performance of hyperarcs in isolation, in Chapter 4 [59] we recognize that congestion arises out of competition for channel access in a neighborhood. As a result, we propose cooperative channelization which uses a health metric to determine a hyperarc’s need for channelization. This health metric takes into account expected number of backoffs, network coding opportunity, and impact of channelization on next-hop downstream hyperarcs - in addition to the sole parameter considered by selective channelization - expected number of transmissions due to collisions. Cooperative channelization channelizes the sickest hyperarc (i.e., lowest value of health metric) in a two-hop neighborhood. Our results show that incorporating these additional factors into the channelization decisions improves the performance of the channelization scheme in congested wireless networks.

- While Chapters 3 and 4 investigated the impact of congestion on intra-flow network coding of multicast flows, we consider inter-flow network coding of unicast flows in Chapter 5 [60]. We address the issue of rate selection for this coding scheme by proposing Network Coding aware Rate Selection (NCRS). We first analyze the performance of network coding on building blocks in a network that enable inter-flow coding of unicast flows. This analysis is then used to formulate NCRS as a linear programming problem to maximize throughput on a hyperarc. Results show the wide applicability of NCRS in addition to performance improvements achieved as compared to other rate selection schemes.

- In Chapter 6 [58] we propose an end-to-end rate selection scheme called NUM-based Rate Selection (NUM-RS) for opportunistic reception in multi-rate wireless networks. We extend prior work on the Network Utility Maximization (NUM) framework in wireless networks to take into account the multi-rate aspect of the medium as well as opportunistic reception. The
end-to-end approach of NUM-RS outperforms other hop-by-hop rate selection schemes including a rate selection scheme designed specifically for opportunistic reception.

7.2 Future Research Directions

Congestion in wireless networks has received significant attention in previous years. This dissertation proposes five solutions to improve the performance of such networks under congestion. Future research work can be partitioned into the following three themes -

• **Implementation:** The focus of my research work has been to study wireless networks, recognize issues related to congestion, and propose effective solutions. I evaluate these solutions with thorough analysis and simulations augmented with limited experiments. Systems level implementation of these schemes will provide further insight as well as validation, and indicate limitations that may be encountered in actual deployments under real conditions. I plan to explore more complete implementations of the channelization [56, 59] and rate selection components [58, 60] of my research work.

• **New Paradigms:** I am very interested in exploring emerging paradigms like network coding, opportunistic reception, multi-user detection, interference cancellation, and dynamic spectrum access, among several others. While these new techniques lead to performance improvement, their interaction with existing techniques and protocols need to be examined to realize their full potential. For example, network coding opportunities may not arise when unicast flows use isolated routes. Hence a routing scheme specific to network coding will need to take into account the tradeoff of congestion and coding gain arising out of paths that intersect and lead to scope for coding. Network coding may also lead to packets being extracted out-of-order and with an unpredictable delay between successive such extracted packets. While some work has recently been conducted on early decoding of packets, the interaction of network coding and TCP needs to be inspected further to sustain TCP’s performance. Interference cancellation can for example be
used to improve tag identification rate in RFID networks which are besieged with collisions when using standard schemes like Slotted Aloha.

- **New Technologies:** I am interested in working with new technologies like IEEE 802.11n, IEEE 802.15.4, ZigBee, IEEE 802.22, etc. For example, while IEEE 802.11 b, a, and g have been primarily used over the last decade to access wireless networks, IEEE 802.11n is being increasingly adopted due to its higher performance. Current convictions about the characteristics of wireless networks were formed using IEEE 802.11 b, a, and g. Working with new technologies like IEEE 802.11n allows us to weigh different concerns facing wireless networks today. In addition, different wireless technologies have been developed for different purposes - Bluetooth for close range, IEEE 802.11 for home/office, Cellular for longer range, 802.15.4 for sensor networks, etc. Most studies recognize and address issues in a particular domain. But since it is evident that a one-size fits all approach will not lend itself to wireless networks, it is important to look at the interaction of different networks.
Bibliography


Vita
Raju Kumar

Education:

• The Pennsylvania State University
  Fall 2004 - Summer 2010
  Ph.D. Candidate in Department of Computer Science and Engineering

• Indian Institute of Technology Kanpur (IIT Kanpur)
  1999-2003
  Bachelor of Technology (B.Tech.) in Computer Science and Engineering

Work Experience:

• Bell Labs, Alcatel-Lucent
  Summer Intern
  Murray Hill, NJ, USA
  23 June 2008 - 22 August 2008

• Adobe Systems, India
  Software Developer
  Noida, India
  9 June 2003 - 30 July 2004

• Sun Microsystems, India
  Summer Intern
  Bangalore, India
  13 May 2002 - 19 July 2002

Honors and Awards:

• AT&T Graduate Fellowship, 2008

• Glenn E. Singley Memorial Fellowship, Graduate Fellowship in College of
  Engineering, The Pennsylvania State University, 2004

• All India Rank 75 in Joint Entrance Examination (IIT JEE) in 1999

• National Talent Search Examination (NTSE), conducted by Government of
  India, Scholar in 1997