Opportunistic Routing and Network Coding in Multi-hop Wireless Mesh Networks

by

A dissertation submitted to the School of Graduate Studies
in partial fulfilment of the requirements for the degree of

Doctor of Philosophy

Department of Computer Science
Memorial University of Newfoundland

April 2018

St. John's, Newfoundland

Abstract

The rapid advancements in communication and networking technologies boost the capacity of wireless networks. Multi-hop wireless networks are extremely exciting and rapidly developing areas and have been receiving an increasing amount of attention by researchers. Due to the limited transmission range of the nodes, end-to-end nodes may situate beyond direct radio transmission ranges. Intermediate nodes are required to forward data in order to enable the communication between nodes that are far apart. Routing in such networks is a critical issue.

Opportunistic routing has been proposed to increase the network performance by utilizing the broadcast nature of wireless media. Unlike traditional routing, the forwarder in opportunistic routing broadcasts date packets before the selection of the next hop. Therefore, opportunistic routing can consider multiple downstream nodes as potential candidate nodes to forward data packets instead of using a dedicated next hop. Instead of simply forwarding received packets, network coding allows intermediate nodes to combine all received packets into one or more coded packets. It can further improve network throughput by increasing the transmission robustness and efficiency. In this dissertation, we will study the fundamental components, related issues and associated challenges about opportunistic routing and network coding in multi-hop wireless networks.

Firstly, we focus on the performance analysis of opportunistic routing by the Discrete Time Markov Chain (DTMC). Our study demonstrates how to map packet transmissions

in the network with state transitions in a Markov chain. We will consider pipelined data transfer and evaluate opportunistic routing in different wireless networks in terms of expected number of transmissions and time slots.

Secondly, we will propose a regional forwarding schedule to optimize the coordination of opportunistic routing. In our coordination algorithm, the forwarding schedule is limited to the range of the transmitting node rather than among the entire set of forwarders. With such an algorithm, our proposal can increase the throughput by deeper pipelined transmissions.

Thirdly, we will propose a mechanism to support TCP with opportunistic routing and network coding, which are rarely incorporated with TCP because the frequent occurrences of out-of-order arrivals in opportunistic routing and long decoding delay in network coding overpower TCP congestion control. Our solution completes the control feedback loop of TCP by creating a bridge between the sender and the receiver. The simulation result shows that our protocol significantly outperforms TCP/IP in terms of network throughput in different topologies of wireless networks.

Acknowledgements

This dissertation could not be realized without help from a wonderful group of people whom I have had the great fortune of meeting.

I am very grateful to my doctoral supervisors, Dr. Yuanzhu Chen and Dr. Cheng Li, who offered me the chance joining their group and helping me to grow as a research scientist. Their valuable guidance, dedication, and encouragement have pushed me far beyond my expectations. They always keep the door open to me whenever I needed to discuss my ideas; their detailed feedback and comments on my papers have greatly improved the quality of my research. I also would like to express my sincere gratitude to Dr. Octavia Dobre, my supervisory committee member, for her excellent support. Without their supervision and constant support in the past five and a half years, I would not be able to bring my research to this successful end. I really appreciate Dr. Yuanzhu Chen's time and patience to guide me back on track when I got lost. During those tough times, his support and belief in me made this difficult journey easier and productive. He has being a great mentor not only for my Ph.D, but also for my whole life.

I am also massively indebted to my supervisor of the master program Dr. Qing Li. He

helped me in the key moment and supported my decision to pursue a PhD degree. His advice on both research as well as on my career has been priceless.

I would like to thank the staff of the Department of Computer Science and the Department of Electrical and Computer Engineering, and all my friends and colleagues in our research group Wireless Networking and Mobile Computing Laboratory (WineMocol), for the help I have received during more than five years and for a pleasant working atmosphere. It was great sharing laboratory with all of you during last five years.

I would like to acknowledge the financial support provided by my supervisors, Natural Science and Engineering Research Council (NSERC), the Department of Computer Science (Graduate Research Award), and IEEE ComSoc (Student Travel Grant to attend IEEE ICC 2016).

I am most grateful to my parents and my wife for giving me support, warmth, comfort, and love all along. Without them I could not possibly have gone this far. Although we are thousands kilometers apart, my heart has never left you.

Thanks for all your encouragement!

Co-Authorship Statements

I, Chen Zhang, hold a principle author status for all the manuscript chapters in this dissertation. However, each manuscript is co-authored by my supervisors, Dr. Yuanzhu Chen and Dr. Cheng Li, whose contributions have expedited the progress of developing the ideas and their formulation, conducting computational experiments, and refinement of the presentation. The contributions for each chapter are mentioned in the following:

- Chapter 2:

  "Joint Opportunistic Routing and Intra-Flow Network Coding in Multi-Hop Wireless Networks: A Survey," accepted in IEEE Network Magazine, 2017.

- Chapter 3:

  "A Markov Model for Batch-Based Opportunistic Routing in Multi-Hop Wireless Mesh Networks," submit to IEEE Transactions on Vehicular Technology, 2018.

  "Analysis of Batched Opportunistic Data Forwarding in Wireless Mesh Networks," published in IEEE Global Communications Conference, 2016.

- Chapter 4:

  "ExOR Compact: Reliable opportunistic data forwarding for wireless mesh networks," published in IEEE International Conference Communications, 2016.

- Chapter 5:

  "TCP Adaptation with Network Coding and Opportunistic Data Forwarding in Multi-Hop Wireless Networks," published in PeerJ Computer Science, 2016.

  "Support of TCP in Wireless Mesh with Unstable Packet Forwarding Capacity," published in IEEE International Conference Communications, 2015.

Chen Zhang	Date

Contents

List of Tables

List of Figures

List of Abbreviations

ACK ..... Acknowledgement

AODV ..... Ad hoc On-Demand Distance Vector

ARQ ..... Automatic Repeat Request

CBR ..... Constant Bit Rate

CTS ..... Clear to Send

DSDV ..... Destination-Sequenced Distance-Vector

DTMC ..... Discrete Time Markov Chain

EAX ..... Expected Any-path Transmission

EDP ..... Expected Distance Progress

ETX ..... Expected Transmission Count

ExNT ..... Expected Number of Transmissions

ExOR ..... Extremely Opportunistic Routing

LCOR ..... Least-Cost Opportunistic Routing

MAC ..... Media Access Control

MORE ..... MAC-independent Opportunistic Routing and Encoding

MTS ..... Minimum Transmission Selection

NACK ..... Negative Acknowledgement

NC ..... Network Coding

NSV ..... Null Space Vector

OAPF ..... Opportunistic Any-Path Forwarding

OR ..... Opportunistic Routing

PNC ..... Physical-layer Network Coding

RLNC ..... Random Linear Network Coding

RTS ..... Request to Send

TCP ..... Transmission Control Protocol

UDP ..... User Datagram Protocol

WMN ..... Wireless Mesh Network

Chapter 1

Introduction

1.1 Background

Did you know there are nearly four billion people for whom connectivity with the Internet is not an option? The multi-hop wireless mesh network can empower users to share photos, videos, music files, apps, and text without ever touching the Internet and meet the demand of a seamlessly connected world. It has emerged as an important technology in the computer networking and becomes an essential part of daily lives of many people around the world such that an unlimited number of people can communicate with one another freely and easily, in an unrestricted way.

A multi-hop wireless mesh network, wireless mesh in short, is a wireless data communication network in which end-to-end nodes are situated beyond direct radio transmission ranges. Intermediate nodes are required to forward data in order to enable communication between nodes that are far apart. Compared to single-hop wireless networks, such as Wi-Fi, Bluetooth, and cellular networks, multi-hop wireless networks can have extended

communication ranges. Furthermore, they can provide coverage in hard-to-wire areas and improve the network throughput by short distance with reliable transmissions. The distinct features and critical design factors of multi-hop wireless networks also pose many problems. Routing is a central problem in wireless mesh. Routing algorithms will select a subset of intermediate nodes to create a path such that these intermediate nodes can forward packets from the source to the destination. The routing process can be decoupled from forwarding, ie., using IP or a variant to forward data independently, e.g., DSDV [79] and AODV [80]. Alternatively, routing can be coupled with forwarding so that forwarding and routing are done by the same module, e.g., DSR [50]. The determination of routing in wireless mesh can incorporate many factors. These include types of topological information, weighted or not and by what, using nodal position or not, number of routes permitted, energy awareness, network coding opportunity, security, and timing of exchanging such information [3, 4, 12, 75].

Most routing solutions are called traditional routing because they follow the Internet architecture of determining routes before forwarding data packets along the routes. Traditional routing pre-selects a fixed routing path before the source node starts transmissions. Each node in a route uses a pre-selected neighbour as the next-hop to forward data packets towards the destination. Once all next-hops have been selected, all packets transmitted between the source and the destination follow the same path. These routing solutions borrow ideas from the routing protocols for wire-line networks, and do not adapt well to the dynamic wireless environment where transmissions failure occur frequently. Wireless mesh networks are built on top of wireless links, and no longer have the constraints of point-to-point links prevalent in the Internet. In particular, wireless links are broadcast and subject to channel fading, which causes an uncertain number of nodes to receive a

single transmission. In wireless networks, when a node transmits a data packet, multiple nodes can overhear the transmission, which can compensate for the transmissions failure to the pre-selected next-hop. However, traditional routing can not utilize this broadcast nature of wireless mesh, because it dictates that all nodes without a matching receiver address should drop packets, and only the node that the routing module decides to be the next hop can keep packets for subsequent forwarding.

Opportunistic routing (OR), also referred as opportunistic data forwarding, diversity forwarding [10] or any-path routing [11, 86], is being investigated to increase the performance of wireless mesh networks by taking advantage of the broadcast nature in wireless mesh. In OR, the next-hop forwarder is decided after the sender broadcasts the data packets. Multiple downstream nodes in addition to that matching receiver will receive these packets and be potential forwarders. One of these potential forwarders, which is 'closest' to the destination, will be selected to forward packets. The on-the-fly selection of the next-hop is the fundamental principle of opportunistic routing. Since multiple downstream nodes are potential next-hop forwarders, opportunistic routing can reduce the possibilities of re-constructing paths or re-transmitting packets due to link breakage on a pre-selected path. A set of intermediate nodes between a source and a destination can be selected as potential forwarders. These selected nodes are called candidate nodes. Candidate selection algorithms choose the candidate nodes and give the proper priority to each candidate. A good routing metric can be used for the selection of candidate nodes and ordering the priority of each node [24, 99, 102]. The highest priority of a node may indicate that this node is 'closest' to the destination. When opportunistic routing was proposed, the typical routing metrics are from traditional routing protocols, for example, the hop count metric or the expected number of transmission. After the selection of can-

didate nodes, opportunistic routing allows any selected node to forward packets according to the coordination algorithm, which is used by nodes to avoid duplicate transmissions and also guarantee that at least one node will forward packets to downstream nodes.

For a better understanding of the inherent benefit of opportunistic routing, we use the example in Figure 1.1, inspired by ExOR [10], to elaborate how opportunistic routing works. It presents the process of opportunistic routing and shows its advantages compared to traditional routing. In this example, the packet delivery probability is shown over the link. Traditional routing chooses the shortest path, $S - A - B - D$, to transmit packets. When S sends packets to node A, data packets are transmitted by broadcasting. Downstream nodes (B or D) may receive some packets. Neither node B nor D is selected as the next hop of the source; even though they receive some packets, these packets have to be retransmitted by node A. Therefore, the traditional routing scheme has many unnecessary transmissions and wastes network resources. In contrast, opportunistic routing can increase the performance of packet transmissions by taking advantage of the broadcast nature of wireless media, specifically long-distance transmissions. ExOR uses the Expected Transmission Count (ETX) [23] as the routing metric, which calculates the average number of transmission that a packet requires to be received by the destination. The ETX value of a corresponding link is the reciprocal of the packet delivery probability of that link. The ETX value on a path can be derived by adding up all ETX values links on this path. The definition of an ETX value of a node is the minimum ETX value of paths from this node to the destination. The intermediate nodes with an ETX value smaller than that of the source will be selected as candidate nodes. In this example, the candidate nodes are $\{A, B, F, D\}$. Each intermediate node will be assigned a priority based on its ETX value. Nodes with a small ETX value will be assigned a high priority.

Therefore, The order of the priority is $D > B > A > F$. After the source node sends packets, multiple candidate nodes are involved in receiving and forwarding packets. In ExOR, the high priority node will forward the received packet first, e.g., if both nodes A and B receive a packet from the source, node B will forward this packet first. When node A overhears node B's transmission, it discards the same packet since node B has forwarded this packet to downstream nodes. If node D receives this packet from the source, neither node A nor B will forward the same packet.

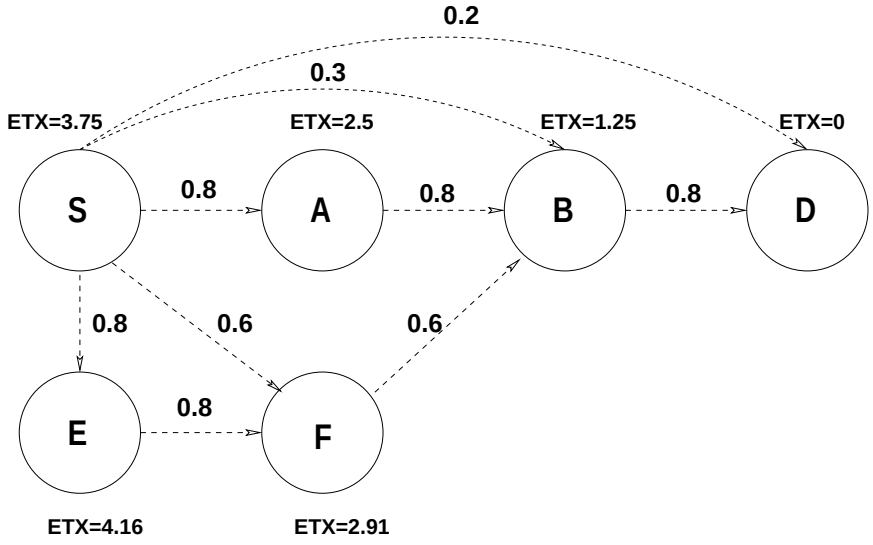

Figure 1.1: An example of opportunistic routing.

With a coordination algorithm, all nodes that have received data packets from upstream nodes will decide whether to forward packets, discard packets or wait for others to transmit. To coordinate transmissions, a node can send extra coordination messages or combine coordination messages with data packets. The coordination message informs other nodes whether it has received packets or not; it is an accurate way to coordinate the nodes in the system if the coordination message is not lost. However, extra co-

ordination messages require network resources and may slow down the transmission of data packets because the data packets are usually sent out only after a node receiving the coordination message. On the other hand, most opportunistic routing protocols join coordination messages with data packets instead. To increase the successful receiving possibility of coordination information and to reduce the amortized overhead, opportunistic routing protocols often transmit a batch of data packets together in a round; packets in that batch always carry the same coordination message, such as ExOR [10], SOAR [83], ROMER [97]. The coordination messages can be received even if only one of the data packets in that batch is received. However, such a reliable coordination mechanism is often complex and requires specialized MAC protocol support, which is difficult to carry out accurately. Furthermore, the coordination mechanism may prevent spatial channel reuse and thus underutilizes the wireless medium. For example, when multiple packets can be simultaneously transmitted by the corresponding nodes, the coordination algorithm still requires the high-priority node to transmit first and the low-priority node to hold the transmission. In general, opportunistic routing often requires a strict coordination on packets in order to avoid redundant transmission [10, 15, 24], this property limits the performance and the use of opportunistic routing [18]. As discussed in [18], network coding can be used to resolve the shortcoming of opportunistic routing.

Network coding is proposed by Ahlswede et al. in 2000 [2]. They prove the value of network coding and provide theoretical bounds on the capacity of wireless networks. It is a networking technique where data packets are encoded and decoded to increase network throughput, reduce delays and improve network robust [44,46,56,63]. Researchers have extended this technique to a variety of areas including content distribution [32], security [14, 45], and distributed storage [48]. Without network coding, data packets are

cached and forwarded to downstream nodes. If a node receives several packets, it forwards them one after another and queues packets that have not received by downstream nodes in the meantime. Therefore, it requires separate transmissions for each packet. In network coding, algorithms are used to code data packets and these coded packets are forwarded to the destination. After receiving coded packets, the destination decodes them using the same algorithm. Network coding applies algebraic algorithms to code data packets; each coded packet randomly combines partial information of different data packets, which further extends the distribution of data packets in the network. In general, two different types of network coding can be applied, namely inter-flow network coding and intra-flow network coding. While the former mixes packets from different sources, the latter focuses on coding packets from the same source. When packets from the same flow are coded, each coded packet is equally beneficial. Nodes do not necessarily forward packets as what are generated by the source; they can transmit a combination of packets. Since no special coded packet is dispensable and the deficit information from lost packets can be easily filled by following coded packets, network coding can eliminate the hop-by-hop feedback mechanism used in traditional forwarding and still achieve reliability in the network. An example of the intra-flow network coding is shown in Figure 1.2.

Figure 1.2: An example of intra-flow network coding.

In this scenario, source node S will send n packets to destination D via intermediate node A. The packet delivery probability is 50% in both directions. Without network

coding, the source node requires a transport layer ACK to learn whether each packet is received by the destination. Thus the expected number of transmissions will be $4n+4n$ (n data packets, n ACKs). Network coding allows both the source and intermediate nodes to make random linear combination of received packets. Once the destination receives n independent coded packets, it can decode n original packets. Then the destination send one ACK for all n packets. The expected number of transmissions will be $4n + 4$.

Generally, opportunistic routing allows multiple nodes to overhear packet transmissions and the next-hop forwarder is decided after the sender broadcasts data packets. The on-the-fly selection of the next hop will increase the network throughput by opportunity gains. However, performing opportunistic routing requires coordination among the links and the design of a specialized MAC protocol, which may prevent spatial channel reuse and thus underutilizes wireless medium. It also requires all of the next-hop nodes to be able to overhear each other, which may not be always possible. Network coding can be used to resolve the shortcomings of opportunistic routing. Network coding increases the transmission capacity of the data communication network as well as its robustness [57]. When packets from the same flow are coded, each coded packet is equally beneficial and inherently different. Therefore, opportunistic routing with network coding does not rely on a complex coordination mechanism to avoid or minimize duplicate transmissions. This simplifies the scheduling since different nodes do not need to exchange information about which packets they have transmitted and received. This property of random network coding eliminates unnecessary feedback and overhearing requirements in opportunistic routing, and makes the design of the MAC layer independent of the other layers.

1.2 Motivation and challenges

1.2.1 Performance analysis of opportunistic routing

Opportunistic routing has shown a great potential performance in simulation and test-bed. To further investigate the advantage of opportunistic routing, many researchers place effort in analytical models. Batch-based coordination is one of the investigated topic in opportunistic routing, where the inherent interaction among packets in a batch affect the performance of opportunistic routing. However, previous works assume that packet transmission is independent, so they only analyze a single packet transmission in the network. Such studies focus on the end-to-end transmission cost of a packet and accumulate this cost for a batch of packets. These analytical models usually require strong assumptions, such as conflict-free schedules access, no interference and no collision. Furthermore, most studies of opportunistic routing only analyze the throughout of the network in terms of the expected number of transmissions but not the transmission time since accumulating transmission time of a batch of packet would yield a poor prediction of packet transmissions where packets are transmitted in a pipelined manner. A more accurate analysis model is needed to better quantify the benefits of opportunistic routing and investigate the effect of batch-based transmissions. Indeed, such analysis model can be considered as an important next step towards modelling general scenarios.

In this dissertation, we will develop an analysis model based on a discrete time Markov model and show how it estimates the performance of opportunistic routing. To the best of our knowledge, this work is the first study investigating the effect of batch-based transmissions. Our model adheres to pipelined data transmission and maps the network state of packet advancement using the Markov model. Furthermore, our model is indepen-

dent of network topologies and candidate selection algorithms. The input parameters of our analysis model are number of nodes, batch sizes, packet delivery probabilities, and pipeline sets.

1.2.2 Reliable opportunistic data forwarding

As stated earlier, opportunistic routing often provides a significant throughput gain compared to traditional routing since it uses multiple potential paths for delivering packets to the destination. When a packet is transmitted, all candidates that successfully receive it will coordinate with each other to determine which one will actually further forward while the others will simply discard the packet. Coordination among these multiple candidates is a crucial issue in order for its overhead not to overwhelm its opportunisty gains. The classic coordination algorithm creates a transmission schedule, where candidates transmit packets in order. High-priority nodes forward packets first, and low-priority nodes only send packets that are not received by higher-priority ones. Only one forwarder is allowed to transmit at any given time; the others listen to learn whether packets are overheard by higher-priority nodes. Candidates go in rounds until all packets reach the destination. Although this transmission schedule can obatin opportunistic throughput gains, it may lose some of the desirable features of the current 802.11 MAC. In particular, the coordination may prevent forwarders from exploiting spatial channel reuse, where multiple packets can be simultaneously transmitted by their corresponding forwarders.

To increase spatial channel reuse for a deeper pipelined transfer; a forwarder should coordinate the priority scheduling only within its transmission range. That is, the coordination of opportunistic data forwarding is limited to the greatest possible range of the

transmitting nodes rather than among the entire set of forwarders. This way, a node's mandatory waiting time is reduced. In this dissertation, we will propose a tighter schedule than ExOR in order to achieve better spatial channel reuse, called *ExOR Compact*. In particular, the schedule is confined to a subset of the forwarders that are within range of the transmitting node so that the waiting time is only a function of this range rather than of the entire forwarder list. It facilitates a pipelined data transfer with higher throughput than ExOR. To moderate the effect brought by the denser network activities, ExOR Compact uses random linear network coding to code packets so that downstream nodes are less sensitive about the lose of specific packets, which potentially alleviates transmission collisions and improves the robustness of data forwarding.

1.2.3 TCP adaptation with opportunistic routing and network coding

The broadcasting nature of wireless links naturally supports both opportunistic routing and network coding, and many studies focus on improving UDP performance in multihop wireless networks. However, opportunistic routing and network coding are inherently unsuitable for TCP for a number of reasons. The frequent dropping of packets and out-of-order arrivals overpower TCP's congestion control. Specifically, opportunistic data forwarding does not attempt to forward packets in the same order as they are injected in the network, so the arrival of packets will be in a different order. Network coding also introduces long coding delays by both the encoding and the decoding processes; besides it is possible along with some scenarios of not being able to decode packets. These phenomena introduce duplicated ACK segments and frequent timeouts in TCP transmissions, which

reduce the TCP throughput significantly.

Majority of previous research on opportunistic routing and network coding was not designed for TCP. Other studies modified TCP protocols by incorporating network coding into TCP protocols; they created different variants of TCP protocols to improve throughput. However, TCP protocols (especially, TCP Reno) are widely deployed in current communication systems, it is not easy to modify all TCP protocols. Therefore, we propose an adaptation layer, called *TCPFender*, functioning below TCP Reno. It adds a thin layer above the network layer to cooperate with TCP's control feedback loop to make the TCP's congestion control work well with opportunistic routing and network coding. TCPFender uses a novel feedback-based scheme to detect the network congestion and distinguish duplicated ACKs caused by out-of-order arrivals in opportunistic routing from those caused by network congestion. With the help of this adaptation, we do not need to make any change of TCP Reno itself while taking advantage of both opportunistic routing and network coding.

1.3 Research contributions

This dissertation presents the following novel contributions to research of computer networking.

- We will propose a discrete time Markov model to study the performance of opportunistic coding in multi-hop wireless mesh networks. It takes into account the interference model, call Protocol Model, to analyze the pipelined data transfer and considers the throughput and the end-to-end delay in both opportunistic routing and traditional routing. We verify the validity of the analytical model by computer

simulation on both NS-2 and Java project.

- We will propose ExOR Compact as a joint opportunistic routing and network coding technique to improve network performance. It creates a regional transmission schedule to confine the waiting time being only a function of the current forwarder rather than of the entire forwarder list and facilitating the pipelined data transfer. We evaluate the performance of ExOR Compact using simulation in NS-2.

- We will propose TCPFender to cooperate with TCP control feedback loop and support TCP to take advantage of both opportunistic routing and network coding. It will compare TCPFender with other baselines in aspects such as throughput, end-to-end delay and evolution of TCP congestion window.

1.4 Thesis outline

The rest of this dissertation is organized as follows.

Chapter 2 provides a review of the studies on opportunistic routing and network coding. We highlight the fundamental components and challenges of opportunistic routing and network coding. By comparing current studies in this area and explaining the motivation and interaction effect of the joint protocols, we discuss the advantage and disadvantage of these two techniques.

In Chapter 3, an analytical model is proposed to study the performance of opportunistic routing in multi-hop wireless mesh networks. The network states of packet advancement are mapped by a Markov chain and the states in the Markov chain denote all valid progression of opportunistic routing. The state transition of a Markov chain

represents packet transmissions in the wireless mesh networks. Our models take into account the theory of the absorbing matrix and enable the estimation of the number of state transitions in an efficient way.

After studying the performance of opportunistic routing, Chapter 4 presents ExOR Compact, a protocol joint opportunistic routing and network coding. It is motivated by improving the spatial channel reuse of opportunistic routing and providing reliable transmissions. We propose an adaptive forwarding schedule that enables nodes in the forwarder list to transmit packets effectively.

In Chapter 5, we propose TCPFender as an adaptation layer to support TCP with opportunistic routing and network coding. TCPFender completes the control feedback loop of TCP by creating a bridge between the adaptation modules of the sender and the receiver. The sender adaptation layer in TCPFender differentiates duplicate ACKs caused by network congestion from those caused by opportunistic data forwarding, and the receiver side releases ACK segments whenever receiving an innovative packet. Our algorithm is designed generally enough to not only support opportunistic routing and network coding, but also any packet forwarding techniques that can cause many dropping packets or out-of-order arrivals.

Finally in Chapter 6, we conclude and summarize the contributions presented in this dissertation, and discuss several potential extensions to our research.

Chapter 2

Related Work

2.1 Opportunistic routing

The basic idea of opportunistic routing is to utilize all downstream nodes as potential forwarders and coordinate transmissions among forwarders. The route of packet transmissions is not predetermined and can be updated according to the change of link qualities for different packets of different flows. In opportunistic routing, a set of intermediate nodes between a source and a destination can be selected as potential forwarders, which are called candidate nodes. Candidate nodes are ordered by their abilities to transmit packets towards the destination according to routing metrics, for example, the hop count metric or the expected number of transmission. The highest ability of a node can indicate that this node is closest to the destination, has highest link quality towards the destination or requires the lowest power consumption, etc. After the selection of candidate nodes, opportunistic routing allows any selected node to participate in forwarding packets according to the coordination algorithm or forwarding schedule, which is abided by forwarders to

avoid duplicate transmissions and also guarantee of at least one node forwarding packets to downstream nodes. With a coordination algorithm, the forwarders that have received data packets from upstream nodes should decide to forward the packets, discard packets, or wait for others' transmissions.

2.1.1 Metrics in OR

A good routing metric has a high impact on candidate selections and the design of the coordination algorithms. In this section, we mention two popular routing metrics in opportunistic routing.

- Expected transmission count (ETX) [23] calculates the average number of transmission that a packet required to be received by the destination. Assume that the link delivery probability between two nodes i and j is $P_{i,j}$. The ETX value of this corresponding link is

$$ETX(i,j) = \frac{1}{P_{i,j}} \tag{2.1}$$

  The ETX value on a path can be derived by adding up all ETX values of each link on this path. The definition of an ETX value of a node is the minimum ETX value of paths from this node to the destination.

- Expected any-path transmissions (EAX) [102] metric is designed for opportunistic routing, which considers multiple paths from the source s to the destination d.

$$EAX(s,d) = \frac{1 + \sum_i^{|C^{s,d}|} EAX(C_i^{s,d}, d) \times p_i \times \prod_{j=1}^{i-1}(1-p_i)}{1 - \prod_i^{|C^{s,d}|}(1-p_i)} \tag{2.2}$$

  Here, the $C^{s,d}$ is the set of candidate nodes from s to d and $\mid C^{s,d} \mid$ is the number of candidate nodes from s to d. $C_i^{s,d}$ is one of the candidate nodes in $C^{s,d}$ with the

priority of i (with 1 being highest priority). The p_i is the packet delivery probability from s to node $C_i^{s,d}$.

2.1.2 Candidate selection in OR

The aim of the candidate selection algorithm is to minimize the transmission cost in terms of the expected transmission transmissions, power consumption or transmission distance, etc. A good candidate selection also needs to assign a priority to each candidate node. In this part, we list two basic and popular candidate selection algorithms in opportunistic routing:

- The candidate selection algorithm in ExOR uses ETX as a routing metric. Each node finds the ETX value of all paths from itself to the destination. This minimum ETX value of a path is the ETX value of this node. The priority of each candidate is decided by the ETX value of the node; a node with a small ETX value has a high priority. Then, all nodes with a smaller ETX value than the source will be selected as candidate nodes.

- The EAX metric is proposed by OAPF [102]. It decides the candidate nodes and priorities based on the packet delivery probability on multiple paths. The selection of candidate nodes begins from the selection of initial candidates $C^{s,d}$, which have a smaller ETX than the source $ETX(s,d) > ETX(j,d)$, where j is a node in the initial candidates $C^{s,d}$. The final candidate nodes will be the subset of the initial candidate set. Note that the candidate selection for all the nodes in $C^{s,d}$ will be finished before s selecting its candidates. Therefore, the candidate selection process is repeated from the destination to the source. After the selection of the

initial candidate set, OAPF candidate selection algorithm selects the best candidate, which can reduce $ETX(s, d)$ the most. This one is added to the final candidate set. This process is repeated until adding any node in the final candidate set can not reduce the expected number of transmission any further.

2.1.3 Forwarding schedule in OR

Forwarding schedule is the coordination mechanism between nodes to clarify whether to forward packets, discard packets or wait for other transmissions. To coordinate between different intermediate nodes, nodes need to send control messages or set timers. The control message informs other intermediate nodes whether it has received packets or not; it is an accurate way to make a coordination between each node given that the control message is not lost. However, the control message requires additional network resources and may slow down the transmission of data packets because the data packets are usually sent out after a node receiving the control message. In addition to control-message based schedules, timer-based forwarding schedules do not rely on any control message, the coordination algorithm is joined with data forwarding instead. When a data packet is sent, the sender sets a timer. If none of downstream candidates receives this packet before the timer expires, the sender will retransmit the previous packet. If one of the downstream candidates receives this packet, the downstream candidates will forward this packet to the destination. If more than one downstream candidate receives this packet, only the highest priority node will forward this packet. The other candidates reset the timers after overhear the transmission from that highest-priority candidate.

- Control-message based schedules require an explicit control packet; for example,

Acknowledgement (ACK) or Request-To-Send (RTS) and Clear-To-Send (CTS). When candidate nodes receive a packet, they will return a short acknowledgement message to the sender. The ACK is sent out in an up-down order, in which the highest priority node sends ACK first and lowest priority node sends the last. The priority of candidates is decided by the routing metric. After downstream nodes send out the acknowledgement, the sender of the data packet and other candidate nodes will know which candidate nodes have received this data packet. Then, the candidate node with the highest priority will forward the packet. The other nodes will discard this packet. This algorithm is first proposed in Selection Diversity Forwarding protocol (SDF) [60].

- Timer-based schedules have no extra control packet and are easy to implement. The overhead for this algorithm is the waiting time. Each candidate node will have a priority based on the routing metric. After a data packet is sent out, candidate nodes will assign a timer if they received the packet. The highest priority node has the shortest timer. After the timer expires, this node will forward the received packets. Since the timer at the high priority node will expire earlier, the high priority node can send packets first and in turn the low priority node can overhear the transmission from the high priority node. If the high priority node already sent out a packet, the low priority node will discard the same packet.

2.1.4 Opportunistic routing protocols

Selection Diversity Forwarding (SDF) [60] incorporates the concept of selection diversity into the framework of routing. The coordination of SDF is that a node performs a

forwarding decision based on multiple responses, which are subsequent acknowledgements from a number of candidate nodes. When a sender forwards a packet, a set of downstream nodes may receive this packet. The header of this packet contains a list of potential receivers and also indicates their priorities. The potential receivers in the packet's header are listed in a consecutive order based on the priority of each receiver. The potential receivers return acknowledgements in this consecutive order. In this way, collisions between acknowledgements are mitigated.

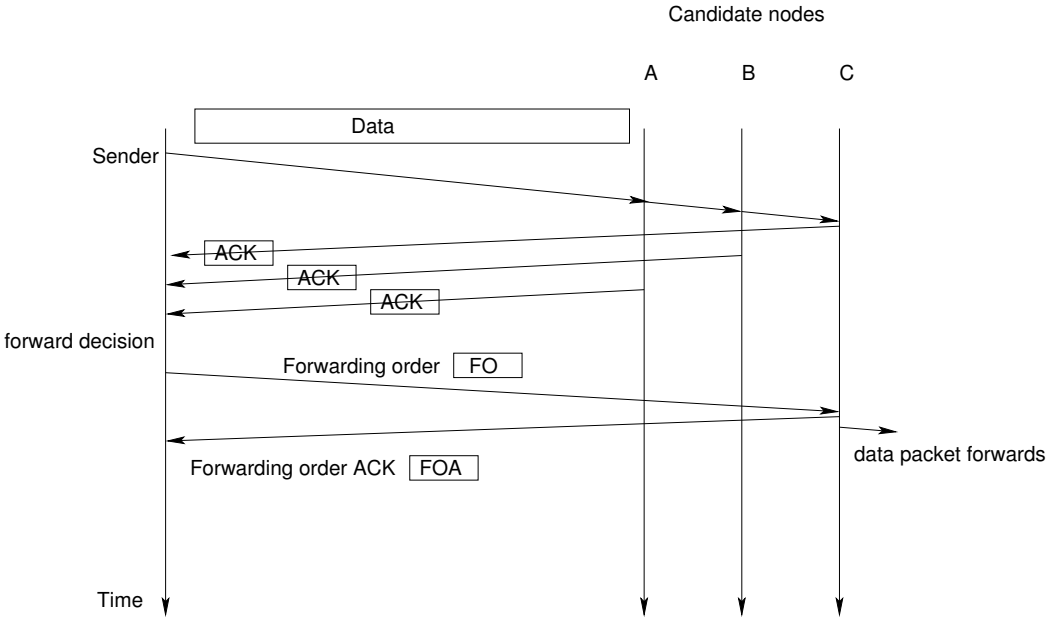

Figure 2.1: An example of of selection diversity forwarding.

After receiving acknowledgements, the sender chooses the 'best' candidate node and sends the Forwarding order that grants the responsibility for onward transmission from this selected node. One example of this process is presented in Figure 2.1. The sender transmits a data packet, and three downstream nodes receive it. Each of them will respond with a respective acknowledgement. After the sender receives all acknowledgements, it

will choose one downstream node as the next-hop forwarder and send the Forwarding order to the selected node C. Finally, the selected node C will return a Forwarding order acknowledgement and further forward the data packet. SDF assumes that control messages are characterized by low energy and short duration compared to data packets. Therefore, the mechanism of SDF can improve the robustness of multi-hop wireless networks and increase the throughput.

The ExOR (Extreme Opportunistic Routing) [10] protocol is an integrated routing and MAC protocol. In wireless networks, when a node transmits a data packet, multiple nodes can overhear the transmission. Traditional IP forwarding dictates that all nodes without a matching receiver address should drop it, and only the node that the routing module decides to be the next hop can keep it for subsequent forwarding . However, ExOR allows multiple overhearing downstream nodes to coordinate forwarding, utilizing the transient high quality of long links. In ExOR, each source nodes specifies a subset of nodes of the network that are on or around the path from the sources to the destination. The nodes are called potential forwarders, and are determined by a link-state-like routing protocol. When the source sends data packets to the destination, the forwarders that are 'close' to the destination have a high priority in forwarding packets towards the destination. When a low-priority forwarder overhears that a high-priority one has forwarded a packet, it knows that it no longer needs to.

To achieve the mechanism above, ExOR has three important features:

- Batch map: The data are grouped in batches for a smaller amortized overhead. The source node collects a batch of packets and prepends an ExOR header to each packet of the batch, which contains a batch ID and a batch map. The batch map

indicates, for each packet in a batch, whether a higher-priority node has received a copy of that packet or not.

- Forwarder list: It contains a copy of the prioritized list of nodes. For a given batch, all nodes have the same forwarder list, which is generated by the source node.
- Forwarding timer: It indicates the time at which the node predicts that it should start forwarding packets to downstream nodes. Each node sets its timer far enough ahead to give higher-priority nodes enough time to send packets. The timer will be adjusted when a node hears other nodes' transmissions.

Simple Opportunistic Adaptive Routing (SOAR) [83] is a proactive link state routing protocol. Every node periodically measures and disseminates link quality in terms of ETX. Based on this information, a sender selects the default path and a list of forwarding nodes that are eligible for forwarding the data packet. It then broadcasts data packets including this information. Upon hearing the transmission, the nodes not on the forwarding list simply discard the packet. Nodes on the forwarding list store packets and set forwarding timers based on their priorities to the destination. A node closer to the destination uses a smaller timer and forwards the packet earlier. Upon hearing this transmission, other nodes will remove the corresponding packet from their queues to avoid duplicate transmissions. Compared to ExOR, SOAR optimizes its candidate selection algorithm and increases the stability of the coordination algorithm. In the candidate selection algorithm of SOAR, each node maintains a routing table with the format: <destination, default path, candidate list>, where the default path is the shortest path from the current node to the destination and the candidate list includes a list of next-hop nodes that are eligible to the following transmission. The candidate selection algorithm of SOAR will

constrain the potential nodes involved in forwarding packets to be near the default path, this candidate selection algorithm simplifies coordination since all the potential nodes are close to nodes on the default path. Therefore they can hear other transmissions with a high probability and utilize these transmissions to coordinate between nodes in a cheap and distributed way.

Cooperative Opportunistic Routing in Mobile Ad hoc Networks (CORMAN) [90] uses a lightweight proactive source routing protocol so that each node has a complete knowledge of how to forward data to all other nodes in the network at any time. Proactive source routing (PSR) [91] is a kind of path finding and link-vector algorithms [76] [30]. When a flow of data packets are transmitted towards the destination, the route information in an intermediate node will be carried by data packets to other intermediate nodes. Each node periodically exchanges route information, which converges after the number of iterations equal to the network diameter. In this way, each node has a spanning tree of the network indicating the shortest paths to all other nodes. If the link conditions change, packets will be forwarded along the new route and such updated information will be rapidly propagated upstream. Therefore, all upstream nodes learn about the new route at a rate much faster than via periodic route exchanges. It takes advantage of opportunistic routing and allows nodes that are not selected on the shortest paths to retransmit data packets if they believe certain packets are missing.

Resilient Opportunistic Mesh Routing (ROMER) [97] tries to forward the packets simultaneously along multiple paths. It incorporates a credit-based approach to limit the number of transmissions that a packet is required to reach the destination along more than two paths. The credit-based approach allows each packet to build its own forwarding candidates on the fly. The packet can be forwarded on different paths offered to resist

against channel fluctuation or node outages.

2.1.5 Analytical models of opportunistic routing

In most works found in the literature, simulation performance and test-bed are used to prove the validation of opportunistic routing. Other research works on developing an analytical model to quantify various transmission cost of opportunistic routing. Most analytical models produce useful opportunistic forwarding metrics to compute the expected number of transmissions in opportunistic routing, like regional opportunistic forwarding metrics: EDP [24], OEOT [99] and end-to-end opportunistic forwarding metrics: ETX [23] and EAX [102]. Other analytical models work on computing the optimal positions of the candidates or developing an unified theoretical framework to quantify various transmission cost of opportunistic routing. Especially, [66] [25] utilize the Markov Chain to formulate the end-to-end transmission cost and provide insighting understanding of opportunistic routing.

Expected Distance Progress (EDP) [24] considers the node positions and link delivery probabilities. It uses the average distance advancement of the transmitted packets as opportunisty metrics. Opportunistic Effective One-hop Throughput (OEOT) [99] characterizes the tradeoff between packet advancement and the medium time cost under different transmission rates. It carries out a study on the impacts of multiple rates, as well as candidate selection. Expected Any-path Transmission (EAX) [102] calculates the expected number of transmission in multi-path transmission, it is a basic opportunistic metric in opportunistic routing. Successful transmission rate (STR) [64] captures the expected successful delivery probability from an intermediate node to the destination and

considers the priorities of nodes based on this possibility. The least-cost opportunistic routing (LCOR) [26] is a candidate selection algorithm to find out the minimum number of transmission. The candidate selection algorithm is repeated over iterations and all possible candidate nodes will be recalculated in each iteration, Therefore, it may cause a large amount of computational delay for ad hoc network. The minimum transmission selection (MTS) [65] is another candidate selection algorithm, which can minimize the expected number of transmission in LCOR by the perfect ACK assumption that all ACKs will be successfully received by corresponding nodes. Compared to LCOR, it has shorter execution time and has the almost same performance in terms of expected number of transmissions.

Luk et al. [74] investigate the effects of node density, transmission power and packet delivery rates by analyzing the average distance gain using lognormal shadowing and Rayleigh fading models. They find that the efficiency of opportunistic routing is high when radio propagation environment is dominated by lognormal shadowing. Opportunistic routing does not perform well where the environment dominated by Rayleigh fading. Cerda-Alabern et al. [16] [17] study the position of the candidates in order to maximize the each transmission progress towards the destination. They propose an equation to analyze the distances of the candidates in opportunistic routing such that per transmission progress towards the destination is maximized. Based on these maximum progress distances, they compute the best position for candidate nodes in order to optimize the packet forwarding towards the destination and also propose a lower bound to the expected number of transmissions required to deliver one packet to destination.

Two analytical models have been proposed in [66] [25] using a Markov Chain to evaluate the performance of opportunistic routing under a perfect coordination, which means

that more than one candidate receives a packet, only the highest priority node forwards this packet while other candidates drop the same packet. [66] utilizes the random walk to formulate the end-to-end transmission cost and also extends the random walk theory from undirected graphs to directed graphs. The model proposed in [25] considers the failure of transmission and limits the number of re-transmission for each intermediate node. Survey of opportunistic routing protocols can be found in [11, 19, 43, 73, 81].

To our best knowledge, all previous analysis models assume that individual transmission of each packet will not affect other packet transmissions thus only analyze one packet transmission in a network, batch-based transmissions are not considered. However, most opportunistic routing protocols require transmission of a batch of packets together to minimize the cost of transmission coordination. Therefore, In Chapter 3, we will propose an analysis model based on the Markov chain to simulate the multi-packet progression process in the network. Compared to previous analysis models of one single packet, our model can consider the interference between packets in a batch and calculate both the expected number of packet transmission and time slots.

2.2 Network coding

Network coding represents an innovative idea introduced by Ahlswede et al. in 2000 [2]. It has emerged as an important potential approach to the operation of communication networks, especially wireless networks. The major benefit of network coding comes from its ability to code data, across time and across flows. It makes data transmission over lossy wireless networks robust and effective. There has been a rapid growth in the theory and potential applications of network coding. These research can be summarized in two

types of network coding: inter-flow network coding that mixes packets from different sources and intra-flow network coding that is about fusing packets from the same source. One of the most popular examples demonstrating the gain of inter-flow network coding is presented in Figure 2.2.

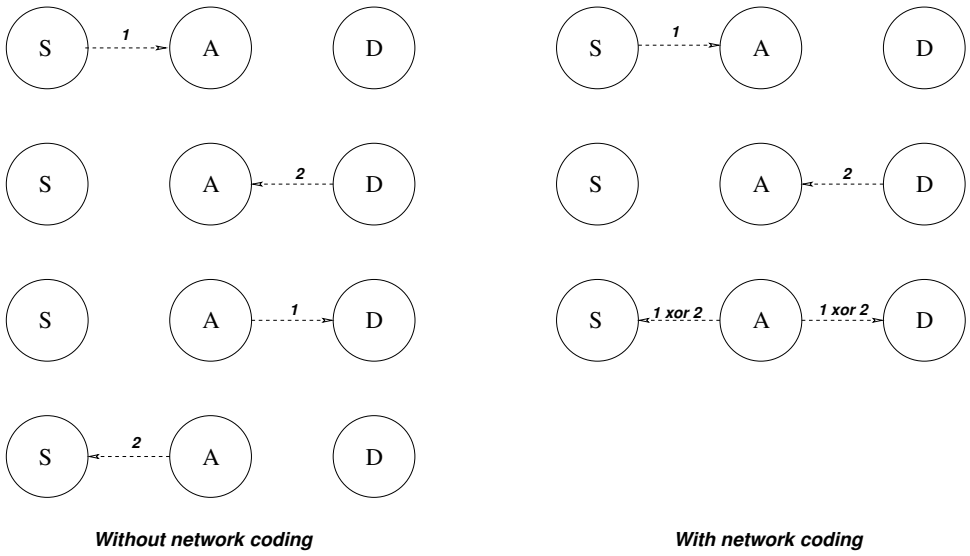

Figure 2.2: An example of inter-flow network coding.

In this scenario, there are three nodes in this topology. The source node S and the destination node D want to exchange packet 1 and 2 via the intermediate node A. Without network coding, four transmissions are required to finish exchanging packets. Network coding can decrease the number of transmissions to three. Specifically, the intermediate node A mixes packets 1 and 2 together and sends a packet (1 xor 2) out. The S and D, which already have packet 1 and 2 separately, are able to decode its required packet 2 and 1 respectively. In general, the linear network coding is different from the xor operation by a linear combination of the data, which interprets data over some finite fields, e.g., $GF(2^s)$. It allows a much larger degree of flexibility to combine packets. With the linear

network coding, coded packets are linear combinations of original packets. The coding process of addition and multiplication is performed over the finite field. The finite field or Galois field is a field that contains a finite number of elements. A finite field is a set on which the operation of multiplication, addition, subtraction and division are defined and satisfy certain basic rules [67]. Addition is the standard bitwise xor. Division is computed by the Euclidian algorithm [28]. Both multiplication and division can be implemented efficiently with s shifts and additions [89].

The primary work of this thesis deals with intra-flow network coding, where packets from the same flow will be coded. Each coded packet contains parts of information from different original packets; no coded packet is special or indispensable. Thus, even though some packets are lost, the sender can keep forwarding coded packets without learning which packets are lost. The subsequent coded packets can compensate for the losses of previous coded packets. The detail discussion about intra-flow network coding as follows.

2.2.1 Encoding at source

Assume that the number of original packets are defined as O^1,O^2,...,O^n and each packet consists of L bits. Original packets will be divided into several symbols with the same length. When original packets to be combined do not have the same size, or the length of the original packets can not be equally divided into symbols with the same length, the shorter packets are padded with trailing 0s. The s consecutive bits of a packets are defined as a symbol over the finite field $GF(2^s)$; therefore, each packet consists of $K = L/s$ symbols. In linear network coding, each packet through the network is associated with a sequence of coefficients $g^1, g^2, ..., g^n$ in $GF(2^s)$. The coded packet

is equal to $C = \sum_{i=1}^{n} g^i O^i$. The summation will occur at every symbol position k; for example, the coded symbol $C_k = \sum_{i=1}^{n} g^i S_k^i$, where S_k and C_k is the kth position of original symbol and coded symbol respectively. We show one example of the encoding process in Figure 2.3.

After packets are coded together, each coded packet contains both a set of coefficients $G = \{g^1, g^2, ..., g^n\}$, called encoding vector and coded data $C = \sum_{i=1}^{n} g^i O^i$, called information vector [28]. The encoding vector is used by receivers to decode original packets. For example, the encoding vector $G = \{0, ..., 1, ..., 0\}$, where the 1 is at the ith position. It means that the information vector is equal to the original packet O^i.

In the linear network coding, the selection of the coefficients for each linear combination is an important issue for practical considerations. A simple algorithm is that each node in the network selects random coefficients over the field $GF(2^s)$ in a completely independent and decentralized manner [37]. With the random network coding, there is a still certain probability of selecting linearly dependent combinations. However, this probability can be reduced by choosing large field size 2^s [28]. Simulation results show that even for small field sizes (for example, $s = 8$), this probability is still negligible [93].

2.2.2 Recoding at intermediate nodes

The encoding process in linear network coding can be performed recursively. Therefore, the intermediate node can recode the coded packet. Consider that a node receives a set of coded packets $(G^1, C^1), ..., (G^m, C^m)$, shown in Figure 2.4, where G^m is the encoding vector and C^m is the information vector. This intermediate node will generate a new set of coefficients $H = \{h^1, h^2, ..., h^m\}$ and compute the linear combination $RC = \sum_{i=1}^{m} h^i C^i$.

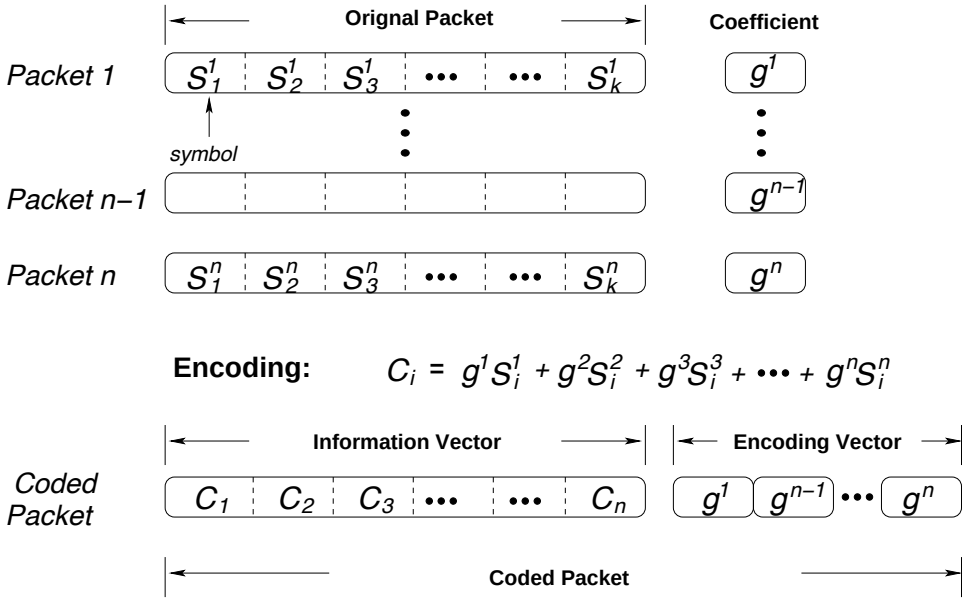

Figure 2.3: An example of encoding in network coding.

The new corresponding encoding vector G' is not simply equal to H, since the coefficients need to be generated with respect to the original packet $O^1, O^2, ..., O^n$, the new encoding vector will be $G' = \sum_{i=1}^{m} h^i G^i$ [28]. This operation can be repeated at intermediate nodes several times in the network.

2.2.3 Decoding at destination

Assume that the destination node receives a set of coded packets $(G^1, C^1), ..., (G^m, C^m)$. In order to decode the original packets, the destination node needs to solve the linear system $C^i = \sum_{i=1}^{n} G^i O^i$, where $G^i = g_i^1 \cdots g_i^n$.

$$\begin{pmatrix} C^1 \\ \vdots \\ C^m \end{pmatrix} = \begin{pmatrix} g_1^1 \cdots g_1^n \\ \vdots \\ g_m^1 \cdots g_m^n \end{pmatrix} \times \begin{pmatrix} O^1 \\ \vdots \\ O^n \end{pmatrix} \tag{2.3}$$

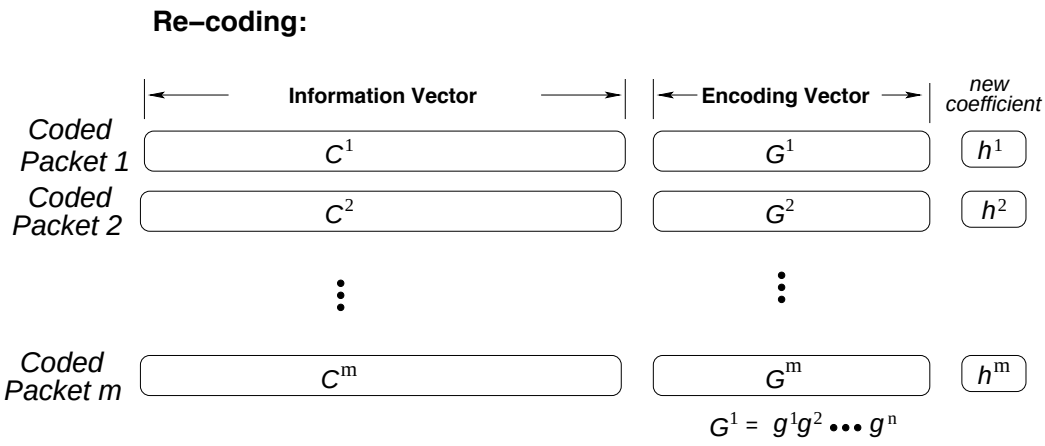

Figure 2.4: An example of recoded in network coding.

The unknowns are O^i and this system is a linear system with m equations and n unknowns. If the number of received packets in the destination node is equal to or larger than the number of unknowns, the destination has a chance of decoding all original packets. Here, the decoding condition that the number of received packets is larger than the number of unknowns ($m < n$) is not sufficient, because some of the coded packets may be linearly dependent.

The destination node will solve a set of linear equations. This process can be done as follows. It stores the encoded vectors from the received coded packets in a decoding matrix [28]. When an encoded packet is received, the encoded vector of this packet will be inserted as the last row of the decoding matrix. Then this matrix will be transformed based on current the encoded vectors, the matrix will be transformed to a reduced row echelon form using Gaussian elimination, where the leading term of each row is 1 and the position of the leading terms moves to the right. A received packet is called innovative if its encoded vector increases the rank of the matrix. As soon as the matrix contains n rows, where n is number of original packets, the destination node can decode all original

Table 2.1: Definitions used in the thesis

Term	**Definition**
Original Packet	An uncoded packet.
Coded Packet	The linear combination of original packets or coded packets.
Innovative Packet	A packet is innovative to a node if this packet is linearly independent from its previously received coded packets.
Coefficient	A value g_i in field $GF(2^s)$ for coding or recoding.
Encoding Vector	A set of coefficients carried by each coded packet.
Information Vector	The coded information from one or more original packets.

packets. The size of the decoding matrix has great impact on the decoding delay at the destination [22]. For practical considerations, this size of decoding matrix should be limited. Therefore, packets are usually grouped into batches or generations, and only packets of the same generation can be combined. The size of batches also has significant impact on the performance of network coding [29].

2.2.4 Network coding protocols

The research of network coding starts after the seminal paper of network information flow by Ahlswede et al. [2]. They propose a new idea that intermediate nodes in the networks do not have to forward the same packets as the source generated, instead, they can forward any combination of these packets by mixing them. It establishes the value of network coding and improves theoretical bounds on the capacity of such networks. The following research [5, 47, 56, 63, 68, 78, 98] shows that the linear coding is sufficient to

achieve the maximun network capacity in multicast traffic transmissions.

Jaggi et al. [46] propose deterministic polynomial time algorithms to design linear codes for directed acyclic graphs with edges of unit capacity. It proves that the coding and decoding can be done in polynomial time. The efficiency of the random linear network coding are proved by [38]. It presents a distributed random linear network coding approach for transmissions and compression of information from multi-source. Each node will independently and randomly select linear mappings from data packet over the finite field. The authors demonstrate that this approach can take advantage of redundant network capacity to improve success decoding probabilities and robustness.

Subsequently, researchers have extended network coding to a variety of areas, including the security [14, 45] and the distributed storage [48]. Additionally, Gkantsidis et al. [32] show how to extend the ideas of network coding to design a peer-to-peer system for content distribution. They improve both the speed of content distribution and the reliability of the system. Some research uses network coding to mix symbols instead of packets [53, 95, 101], i.e. analog network coding [53] encourages strategically picked senders to interfere. Instead of forwarding correct packets, intermediate node can forward the interfering signals. The destination leverages network-level information to cancel the interference and recovers the signal. Their work show that forwarding signals instead of packets can double the capacity of the network. A survey on unicast, multicast and broadcast applications of network coding for wireless networks can be found in [9,27,41,75,77].

2.3 Opportunistic routing with network coding

2.3.1 Intra-flow network coding

Generally, opportunistic routing allows multiple nodes to overhear packet transmissions and be involved in further forwarding packets. With intra-flow network coding, packets from the same flow are coded; each coded packet is equally beneficial and inherently different. Since no special coded packet is dispensable, the deficit information from lost packets can be easily filled by following coded packets. The combination of these two ideas in a natural fashion to provide opportunistic routing without a coordination and exploit opportunisty gains inherent in wireless media. This idea has received considerable attention from the research community and a significant amount of research has been conducted on different points of view [32, 51, 54, 58, 69, 70, 94].

Despite the simplicity of jointing opportunistic routing with network coding, one challenge of much of the literature on this area is how many coded packets should be transmitted in each forwarder. An ideal number of coded packets can not only avoid duplicate transmissions but also guarantee the successful decoding. There are two ways to estimate the number of coded packets. The first type computes the offline expected number of transmissions based on the average link loss rates. In such case, end-to-end feedback is required to acknowledge the successful decoding. The second type requires intermediate nodes to detect the real-time link quality by hop-by-hop feedback, which will piggyback on data packets. We will discuss these two types of protocols in section 2.3.1.1 and section 2.3.1.2, shown in Figure 2.5.

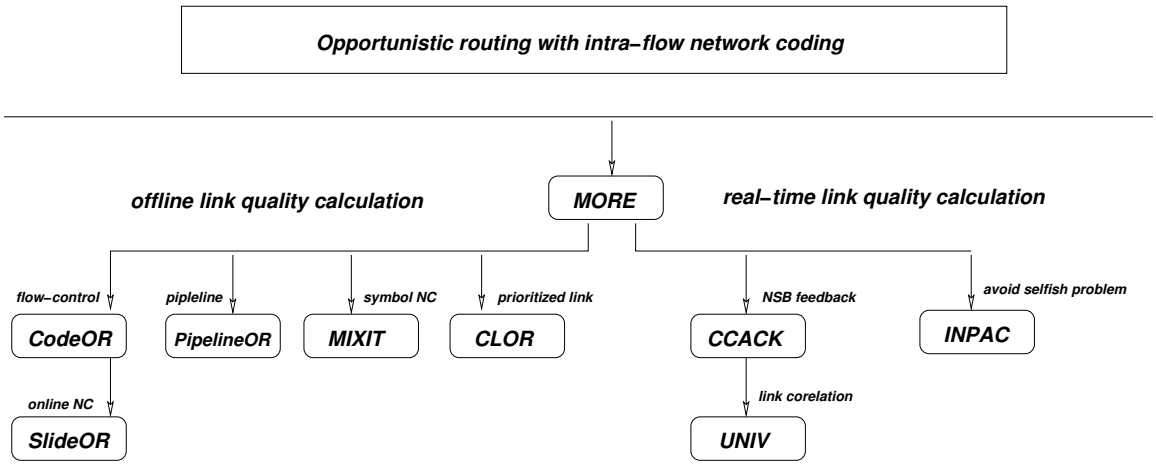

Figure 2.5: Opportunistic Routing with intra-flow network coding protocols.

2.3.1.1 End-to-end feedback based forwarding schedule

MORE (Mac-independent Opportunistic Routing and Encoding) is a practical opportunistic routing protocol with intra-flow network coding [18]. It is the first paper that works on opportunistic routing with network coding and has a seminal effect in this area. MORE selects potential candidates at the source before data transmissions, these candidates are 'closer' to the destination than the source. Then data packets from the upper layer will be grouped into batches and coded packets will be generated according to the packets in that batch. Coded packets will be forwarded with the order of batch index, in which the transmission of the next batch of coded packets begins after all coded packets from the previous batch reach the destination. The source node keeps transmitting the current batch via candidate nodes until the destination node can decode all packets. One challenge of this mechanism is how to avoid duplicate transmissions and also guarantee reliable transmissions. MORE proposes a transmission credit system to control the number of data transmissions, which is calculated based on how effective it would be in forwarding coded packets to downstream nodes. The transmission credit system has three

important aspects:

- For each node j, calculate the number of packets that j must forward:

$$L_i = \sum_{i>j} (z_i(1-\epsilon_{ij}) \prod_{k<j} \epsilon_{ik}) \tag{2.4}$$

  Note that $i < j$ denote that node i is closer to the destination than node j (i has a smaller ETX than j) and ϵ_{ik} denote the loss probability in sending a packet from i to j.

- The expected number of transmissions that j must make is:

$$z_j = \frac{L_j}{(1 - \prod_{k<j} \epsilon_{jk})} \tag{2.5}$$

- The TX_{credit} of node j:

$$TX_{credit_j} = \frac{z_j}{\sum_{i>j} z_i(1-\epsilon_{ij})} \tag{2.6}$$

  The intermediate node j keeps a credit counter and increases the counter by its TX_{credit} when receiving an innovative coded packet. The intermediate node sends a coded packet when the counter is positive. After the node sends a coded packet, it decreases the counter by 1.

MORE also supports the multicast transmission, which has three important modifications compared to the unicast transmission. First, the candidate nodes are the union of the candidates of different unicast traffic. Second, the transmission credit value is calculated from the maximum of transmission rates of different unicast traffic. Third, the source transmits the next batch until the packets in the previous batch reach all destinations.

The main drawback of MORE is that the source simply keeps transmitting coded packets belonging to one batch until the acknowledgment of this batch from the destination has been received. The transmission of the next batch will be suspended while awaiting the acknowledgment. It is a 'stop-and-wait' design, which is unable to be applied in large scale networks. CodeOR (Coding in Opportunistic Routing) [69] proposes a mathematical analysis, which demonstrates that this 'stop-and-wait' mechanism slows down the network traffic, especially in a topology with the large or long diameter. Thus, in CodeOR, it allows the source to transmit multiple batches in the way of a sliding window to alleviate this problem. The proper window size is estimated by applying the ideas of TCP flow control. But it performs the flow control on batches rather than on packets, where the sending window limits the number of outstanding batches that the source can transmit at any time in the network.

It studies the impact of the window size on throughput and estimates the proper window size to be approximately equal to the delay-bandwidth product between the source and the destination. Since multiple batches of coded packets can simultaneously transmit in the network, it is intuitive that CodeOR outperforms MORE on large-scale networks.

PipelineOR [72], inspired by MORE, uses the same coordination mechanism. It allows the intermediate nodes to be responsible for the packet transmission on behalf of the source when these intermediate nodes receive all packets in a batch. The source node can transmit the next batch of packets when more than two intermediate nodes receive all packets from the previous batch. Thus, the batches of packets can be transmitted in a pipelined manner.

SlideOR proposes the online network coding for opportunistic routing [71], in which

the most useful packets will be encoded and other redundant information can be easily eliminated. SlideOR, instead of coding packets from separate batch, creates a sliding window to encode most useful packets. Original packets from a sliding window are coded together; since sliding windows can overlap with each other, a coded packet from one sliding window may be useful towards decoding the packets from another sliding window. It uses a 'seen' mechanism to differentiate how a useful packet. If a packet is already 'seen' instead of being decoded, this packet will not be encoded by the source any more. In Figure 2.6, the sender transmits 6 packets, the second transmission is lost and the decoding matrix is shown on the left. The receiver can decode packets 1, 2 and 3. Even though packets 4 and 5 cannot be decoded right now, they can be decoded after other 'unseen' packets arrival. Therefore, the sender only needs to send the unseen packet 6.

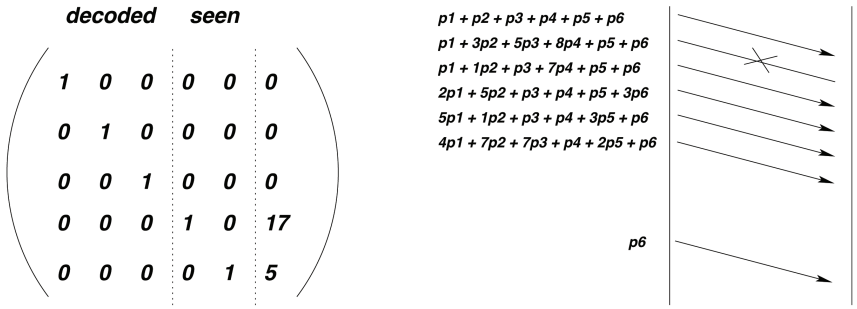

Figure 2.6: An example of online network coding in SlideOR.

MIXIT [54] operates network coding on small groups of bits, called symbols. A packet consists of multiple symbols and a symbol is the smallest transmission unit over the wireless link. MIXIT allows intermediate nodes to opportunistically transmit groups of coded symbols to their destination. With such a simple modification, it can utilize correct symbols in a corrupted packet, therefore achieve high throughput. In other research,

each intermediate node forwards a packet only if this packet has no errors. In contrast, MIXIT takes a much looser approach: the intermediate nodes do not attempt to recover from any errors by themselves. It incorporates an end-to-end error correction component that the destination will use to correct errors and decode original packets. Even none of intermediate nodes receives the whole packet correctly, as long as all the individual symbols in this packet are received correctly by intermediate nodes separately, these symbols can be use to decoded original packets at the destination. One example is shown in Figure 2.7. The source node sends two packets and two intermediate nodes receive these two packets with errors. A shaded cell represents an error symbol. Intermediate nodes can still recode the correct symbols and send them to the destination.

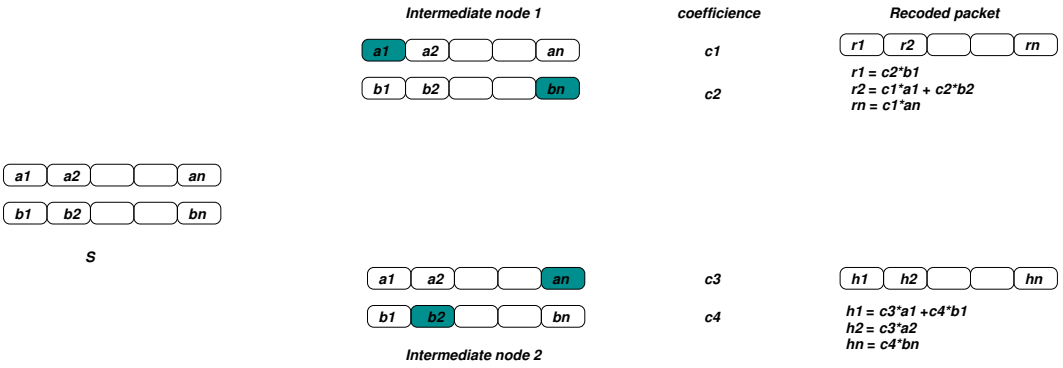

Figure 2.7: An example of symbol-level network coding.

Garrido et al. [31] propose a Cross-Layer Opportunistic Routing (CLOR) to balance the transmissions between intermediate nodes based on the quality of wireless links. The transmission opportunity of an intermediate node not only depends on the quality of the link from itself to a downstream node, but is also affected by the proportion of this link quality in the sum of the quality for all links in the network. The transmission opportunity

of a node i is calculated by

$$P_i = \frac{1}{FER_i^r} \times \frac{1}{\sum_{j=1}^{N} (\frac{1}{FER_j^r})^{-r}} \tag{2.7}$$

The FER is frame error rate for the link. The parameter r is the delivery ratio from a particular sender at any time. The r parameter in this equation is to give a priority to high quality links. The larger value it is, the higher transmission opportunity is given to this link.

2.3.1.2 Hop-by-hop feedback based forwarding schedule

MORE and similar protocols compute offline the expected number of transmissions before data transmissions by periodic measurements of average link loss rates. Although attractive due to the simply coordination mechanism and low overhead, these solutions may suffer performance degradation in dynamic wireless networks where the link quality changes frequently. The performance of these solutions heavily depends on the accuracy of the measurement. CCACK (Cumulative Coded ACKnowledgment) [58] provides real-time feedback to estimate the link loss rate between intermediate nodes. It can acknowledge received coded packets to upstream nodes with negligible overhead by utilizing a null space-based (NSB) coded feedback vector to represent the entire decoding matrix.

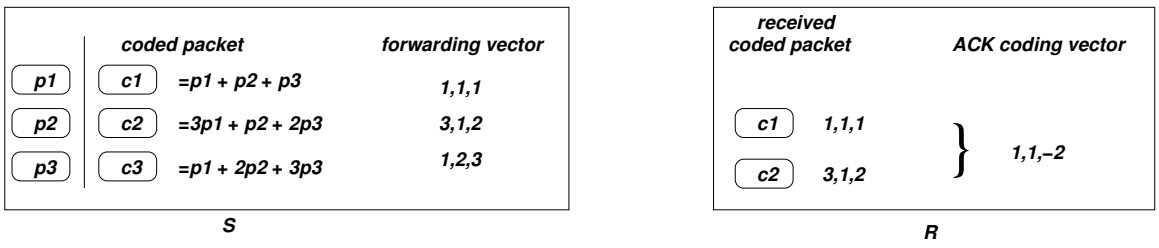

Figure 2.8: An example of null space-based (NSB) coded feedback.

In CCACK, each coded packet has two coding vectors: forwarding coding vector and ACK coding vector. The forwarding coding vector is used to recode or decode the payload of data packets by any downstream node or the destination respectively. The ACK coding vector is used by the upstream node to check whether data packets are received by downstream nodes. The inner product of the ACK coding vector with the encoding vector of each received coded packet is zero. i.e., if a vector x is from the null space of the decoding matrix A, then $Ax^T = 0$, where x^T is the transpose of x. In Figure 2.8, the left graph shows three coded packets with corresponding forwarding vectors. The right graph shows the ACK coding vector according to two coded packets. Sender s transmits three coded packets to Receiver r and two of them are successful received. Receiver r will compute the null space for received coded packets, pick one vector from the null space and send it back to s. Then sender s multiples this vector with the forwarding vector of each coded packet it has sent. If all results are zero, Sender s can infer that all packets have been received with a high probability. Otherwise, it keeps sending packets to the receivers. In this case, the third coded packet is lost and the inner product of the third forwarding vector with the ACK vector is not zero. With such a mechanism, CCACK can make real-time detection of the link quality and dynamically update the number of coded packets to be transmitted in each forwarder.

UNIV [55] considers the correlation among the wireless links and proves the number of coded packets to be sent not only depends on the link quality, but also on the correlation among the links. In a general network, the links will have different correlations and the correlations can change over time. UNIV studies three types of correlation. The first is that the links are independent, in which the reception process is independent among the links. The second one is that the links are positively correlated or correlated, which

means that if one link is inactive, the other one will be inactive. The third case is that the links are negatively correlated or uncorrelated, which means that if one link is active, the other one will be inactive. Their work estimates the link quality in these three correlation cases and considers the coded feedback approach in different channel conditions.

The hop-by-hop feedback from downstream nodes can help upstream nodes to detect the link condition and improve the utilization of limited network resources. However, any node could be selfish and let itself deviate from the communication protocol if this deviation is helpful to itself. Therefore, Chen et al. [21] propose an incentive-compatible communication protocol to make nodes follow the protocol faithfully. They model the packet forwarding procedure by a game-theoretic model. The players of this game are the nodes that are following the MORE protocol to forward the packets. The game is divided into stages, where each stage represents a very short period of network transmissions. The forwarding scheme consists of two parts: the data transmission as MORE and the report transmission after a short period of data transmissions. The report is used to make sure that each intermediate node faithfully sends the required number of data packets.

Radunovic et al. [82] propose an optimization framework for opportunistic routing based on network utility maximization. It uses network coding to simplify the routing problem and analyzes the flow control, routing, scheduling, and rate adaptation schemes in opportunistic routing. This framework considers a primal-dual problem. The primal formula analyzes the optimization problem as a function of the transmission rates of various flows in the network. The dual formula uses the queue lengths in each node as variables. The framework derives a credit mechanism, a TCP-like sliding window model, to decide how many packets should be forwarded by each node. This credit mechanism guarantees that the destination will receive as enough linear combinations of coded packets

as the number of original packets. It also considers the flow control algorithm and analyzes fairness to maximize the aggregate utility of end-to-end flows.

Soldo et al. [86] study the optimization problem of opportunistic routing and congestion control in wireless mesh networks with intra-session network coding. The optimization problem aims at maximizing the aggregate utility as a function of the number of linearly independent packets. The dual problems are decomposed to three parts: congestion control, wireless interference and the routing selection. To achieve the network utility maximization, this framework requires two types of feedback: the end-to-end feedback to adjust the sending rate of the source based on what is actually received by the destination. The hop-by-hop feedback to guarantee that the information transmitted from the upstream nodes are useful to downstream nodes.

MIC-NCOR [15] is a candidate selection algorithm to allocate traffic among candidate nodes and approach optimal performance. It creates a relationship tree to describe the child-parent relations along the path from the source to the destination. The cost of a node is the sum of the costs of its constituent hyperlinks for delivering one unit of information from the source to itself. The calculation of a given node is iteratively made from the destination and back to the source. These nodes that create the path with the minimum cost can be chosen as candidate nodes. For more details on opportunistic routing with intra-flow network coding, we refer the readers to [63, 92].

2.3.2 Inter-flow network coding

COPE [8] is the prominent study in inter-flow network coding. It is the first network coding architecture that supports unicast transmissions by performing coding of packets

from different sources. COPE incorporates three main techniques. First is opportunistic listening. Each node can overhear packets from the neighbours' transmissions and keep these packets for a while to decode following coded packets. Second is opportunistic coding. COPE aims to maximize the number of original packets delivered in a single transmission, the sender should ensure that the receiver has enough information to decode original packets. Packets from multiple unicast flows may be coded together at some intermediate nodes and decode at the next hop where the paths will diverage. Opportunistic coding will try to guarantee the coded packets can be decoded successfully. Otherwise, unneeded data will be forwarded to areas where there is no interested receivers. Third is about learning neighbour state, it is used by nodes to exchange information about what packets their neighbours have.

In COPE, the paths of flows are decided by the routing protocol. Hence, the coding opportunities depend on the current topologies formed on the paths. Coding opportunities in this case are limited by the fixed path routing. BEND [100], as an advancement of COPE, is the first attempt to combine opportunistic forwarding and practical network coding in wireless mesh networks. It can employ the diversity of different potential forwarders and unify the needs of traffic separation and concentration. In multi-hop wireless mesh networks, traffic flows should be separated to minimize the interference among them. Conversely, for network coding to function, the same traffic flows are expected to be jointed at some points. These two needs were difficult to balance before. Therefore, BEND proposes a second-next-hop coding mechanism where a node will combine two packets only if the next hop of the first packet is the previous hop of the second packet or one of its neighbours. To increase traffic separation, BEND defines helper nodes that overheard the packets but are not selected as the next hop in the route. These helper

nodes can receive original packets, encode these packets and forward them if the selected nodes do not receive packets. However, these helper nodes are only allowed to encode original packets. If they receive a coded packet, they need to discard it. In addition, a helper node is allowed to forward packets only if this helper is one-hop away from the selected node. This guarantees that the packet propagation is restricted within a band along the route without flooding in the network.

Inter-flow Network Coding based Opportunistic Routing (INCOR) [103] is designed to further reduce the expected number of transmissions for each packet. It presents a new metric called Coding-based Expected Transmission Count (CETX) to determine the priority of the forwarders. The CETX metric computes the expected transmission number required to deliver one packet to a destination when inter-flow network coding is incorporated. NC-MAC [6] improves the efficiency of coding decisions by verifying the decoding of packets before they are transmitted. The scheme focuses on improving correct decoding probability at the destination.

The coding-aware opportunistic routing mechanism will take into account available coding opportunities when they make routing decision. These routing protocols will select routes in a way that the flows intersect at some joint nodes to maximize the probability of the packets of multiple flows being combined together. However, if the flows are close to each other, the interference at the joint nodes may be a bottleneck in the network and lead to degrading the performance. Therefore, coding-aware opportunistic routing protocols need to consider the trade-off between coding opportunities and the interference. CAOR (Coding Aware Opportunistic Routing) [95] can improve the performance by selecting routes with joint intermediate nodes for different flows. It proposes a localized coding-aware opportunistic routing mechanism, which aims at increasing the coding

opportunities when multiple simultaneous unicast flows exist. The amount of coding opportunities is measured by how many original packets can be successfully delivered in a single transmission.

COPE has two fundamental limitations: the coding opportunity is crucially dependent on the established routes and the coding structure in COPE is limited within a two-hop region only. DCAR (Distributed Coding-Aware Routing) [61] is distributed coding-aware routing system for wireless networks. It incorporates potential coding opportunities into route selection using the Coding-aware Routing Metric (CRM). The metric takes into account the 'free-ride' benefit of the coding-possible paths: if a new flow can be encoded with some existing flows, it can free-ride on the bandwidth used by the existing flows. The source will collect route information about other flows and the topology before the data transmissions. Therefore, DCAR knows the coding opportunities on the entire path and eliminates the two-hop coding limitation in COPE. However, DCAR cannot guarantee successful decoding of coded packets when more than two flows are coded together or two flows intersect in more than one node. Therefore, Free-ride-Oriented Routing Metric (FORM) [34] is proposed to iteratively recalculate the route for all existing flows and attempts to reach the global optimality of the entire network. More interesting papers in this area, readers are referred to [36, 39, 42, 49, 57, 85, 96, 103] for comprehensive reviews and taxonomy.

2.3.3 Intra- and inter-flow network coding

I^2NC [84] and CORE [59] combine inter-session and intra-session network coding. Both of them borrow the idea of intra-flow network coding from MORE and combine

it with idea of COPE to improve the unreliable overhearing mechanism in opportunistic routing. I^2NC uses intra-flow network coding to combine packets within the same flow. It makes packet transmissions resilient to loss and intermediate nodes can operate coded packets without knowledge of the decoding buffers of their neighbours. Based only on the knowledge of the loss rates on each link, intermediate nodes can make a decision on how much coded packets to create in each flow and what percentage of flows to code together.

CORE (Coding-aware Opportunistic Routing mEchanism) allows nodes in the network to setup inter-flow coding regions where packets from different flows can be XORed. Packets from the same flow use random linear network coding for intra-flow coding. It identifies regions where two or more flows intersect and makes inter-flow network coding for each flow at the identified regions. Within the identified regions, CORE combines packets of multiple flows using inter-flow network coding, then separates them before leaving the region. Furthermore, CORE requires each node to signal to its neighbours when it receives all degrees of freedom, i.e., it has enough coded packets to recover the original data. This feedback can reduce energy consumption in the coding region and to allow next batches of coded packets to be sent into that region. FlexONC [52] joint cooperative forwarding and network coding with rrecise encoding conditions, it increases the coding opportunities and combines packets of different flows that are overly optimistic and would affect the network performance adversely.

CodePipe [62] is a reliable multicast protocol with both intra- and inter-flow network coding. It proposes a destination-cooperation mechanism for multicast transmissions. Once a destination has received enough coded packets for decoding, this destination will proactively participate in packet transmissions by cooperation with the source for other destinations.

2.4 Network coding in TCP

A number of recent papers have utilized network coding to improve TCP throughput. In particular, Huang et al. [40] introduce inter-flow network coding to TCP traffic, where data segments in one direction and ACK segments in the opposite direction can be coded at intermediate nodes. The simulation shows that making a small delay at each intermediate node can increase the coding opportunity and increase the TCP throughput. ComboCoding [20] uses both inter- and intra-flow networking to support TCP with deterministic routing. The inter-flow coding is done between the data flows of the two directions of the same TCP session. The intra-flow coding is based on random linear coding serving as a forward-error correction mechanism. It has an adaptive redundancy to overcome variable packet loss rates over wireless links.

TCP/NC [88] incorporates intra-flow network coding into the TCP-compatible sliding-window approach. It reuses the congestion control principle of TCP where the number of packets involved in transmissions cannot exceed the congestion window size. Whenever the source is allowed to transmit, it sends a random linear combination of all packets in the congestion window. Such a variant of TCP modifies the ACK feedback mechanism. The destination acknowledges the degree of freedom and not original packets. If a received packet increases the degree of freedom in the decoding matrix, this packet is an innovative packet and defines as being 'seen' by the destination. The destination node will generate an acknowledgment whenever a coded packet is seen instead of producing an original packet.

TCP-FNC [87] introduces online network coding to TCP/NC, where the coded packets is restricted to a small group of packets in the congestion window. It can efficiently control

the waiting time of decoding when the congestion window is large. Furthermore, the progressive decoding algorithm in a small group of coded packets can reduce computation delay. It can be applied to loss-based congestion control, but it does not take advantage of opportunistic routing. Since TCP-FNC is based on traditional IP forwarding, it is easily affected by link quality variation. TCP-VON [7] also incorporates online network coding into TCP. In TCP-VON, the sender transmits redundant coded packets when it detects packet losses from an acknowledgement. Otherwise, it transmits innovative coded packets. It can dynamically update the transmission rate of data packets based on the acknowledgement. However, these protocols are variants of RTT-based congestion control TCP protocols, e.g., Vegas, which limits their applications in practice since most TCP protocols employ loss-based congestion, e.g., Reno. Therefore, we will propose an adaptation layer functioning below TCP Reno in Chapter 3. With this adaptation layer, TCP Reno does not make any change to itself and it can take advantage of both network coding and opportunistic routing.

Table 2.2: A summary of opportunistic routing and network coding studies.

Protocol	TR/ OR[1]	NC	Intra/ Inter	UDP/ TCP	Coding-aware	Multicast/ Unicast[2]	Evaluation method[3]
ExOR	OR	No	-	UDP	-	Unicast	Experiment
SOAR	OR	No	-	UDP	-	Unicast	Experiment Simulation
CORMAN	OR	No	-	UDP	-	Unicast	Simulation
SDF	OR	No	-	UDP	-	Unicast	Simulation
MORE	OR	Yes	Intra	UDP	No	Both	Experiment Analytical Model
CCACK	OR	Yes	Intra	UDP	No	Both	Simulation Analytical Model
CodeOR	OR	Yes	Intra	UDP	No	Unicast	Simulation
SlideOR	OR	Yes	Intra	UDP	No	Unicast	Simulation
MIXIT	OR	Yes	Intra	UDP	No	Unicast	Experiment
ONCR	OR	Yes	Intra	UDP	No	Unicast	Experiment Analytical Model
NCOR	OR	Yes	Intra	UDP	No	Unicast	Simulation
[51]	TR	Yes	Intra	Both	No	Unicast	Experiment
COPE	TR	Yes	Inter	Both	No	Unicast	Experiment
BEND	OR	Yes	Inter	UDP	No	Unicast	Simulation
INCOR	OR	Yes	Inter	UDP	No	Unicast	Simulation
NC-MAC	OR	Yes	Inter	UDP	No	Unicast	Simulation Analytical Model
CAOR	OR	Yes	Inter	UDP	Yes	Unicast	Simulation
DCAR	TR	Yes	Inter	UDP	Yes	Unicast	Simulation Analytical Model
FORM	TR	Yes	Inter	UDP	Yes	Unicast	Simulation Analytical Model
CORE	TR	Yes	Both	UDP	No	Unicast	Experiment Simulation
I^2NC	TR	Yes	Both	Both	No	Unicast	Simulation Analytical Model
CodePipe	OR	Yes	Both	UDP	No	Multicast	Simulation
[40]	TR	Yes	Inter	TCP	No	Unicast	Experiment
TCP/NC	TR	Yes	Intra	TCP	No	Unicast	Simulation Analytical Model
TCP-FNC	TR	Yes	Intra	TCP	No	Unicast	Simulation Analytical Model
TCP-VON	TR	Yes	Intra	TCP	No	Unicast	Simulation
ComboCoding	TR	Yes	Both	TCP	No	Unicast	Simulation

[1]Traditional Routing/ Opportunistic Routing.

[2]Unicast: one source to one destination. Multicast: one source to multiple destinations

[3]Evaluation methods include: Experiment, Simulation and Analytical Model.

Chapter 3

Analysis model for batch-based opportunistic routing

3.1 Background

In opportunistic routing, the next-hop forwarder is decided after the sender broadcasts the data packets. Multiple downstream nodes in addition to that matching receiver will receive these packets and be potential forwarders. One of these potential forwarders, which is the 'closest' to the destination, will be selected to forward packets. The on-the-fly selection of the next-hop is the fundamental principle of the opportunistic routing [25]. Since multiple downstream nodes are potential next-hop forwarders, opportunistic routings can reduce the possibilities of re-constructing paths or re-transmitting packets due to link breakage on a pre-selected path. In opportunistic routing, a set of intermediate nodes between a source and a destination can be selected as potential forwarders. These selected nodes are called candidate nodes. Candidate selection algorithms choose the candidate

nodes and give the proper priority to each candidate. After the selection of candidate nodes, opportunistic routing allows any selected node to forward packets according to the coordination algorithm, which is used by nodes to avoid duplicate transmissions and also guarantee that at least one node will forward packets to downstream nodes.

With a coordination algorithm, all nodes that have received data packets from upstream nodes will decide whether to forward packets, discard packets, or wait for others transmissions. To coordinate transmissions, a node can send extra coordination messages or combine coordination message with data packets. The coordination message informs other nodes whether it has received packets or not; it is an accurate way to make a coordination between each node given that the coordination message is not lost. Extra coordination messages require network resources and may slow down the transmission of data packets because data packets are usually sent out only after a node receives the coordination message. In addition to sending extra coordination messages, most opportunistic routing protocols [10] [83] [97] join coordination messages with data packets instead. To increase the successfully receiving possibility of coordination information, they transmit a batch of data packets together with a same coordination message. The coordination messages can be received if only one of the data packets in that batch is received.

In batch-based opportunistic routing, the source node transmits a batch of packets to downstream nodes at the beginning of each round. After the source node finishes the transmission of packets in that batch, each candidate will prepare to forward its received fragment of that batch. A same coordination message will be attached in each packet of that fragment; this coordination message carries information about whether any packet in that batch has received by any higher priority candidate. The transmission of each round will begin from the highest-priority nodes who receive any packet in that batch back to the

source node. Since high-priority candidates forward packets first, low-priority nodes can avoid duplicate transmissions. The detailed description of the batch-based coordination can be found in [10]. When a candidate is transmitting a fragment of that batch, all neighbour nodes will delay their transmissions by setting a timer to avoid interference. Candidates, which are far away to interfere each other, are scheduled to forward received packets at the same time; this pipelined transmission is a general situation in opportunistic routing.

Previous analytical models assume that the packet transmission is independent and only analyze a single packet transmission in the network. Such studies focus on the end-to-end transmission cost of a packet and accumulate this cost for a batch of packets. They only analyze the performance of the network in terms of the expected number of transmissions but not time slots since accumulating the time slot of a packet for a batch of packets would yield a poor prediction where packets are transmitted in a pipelined manner. Furthermore, the batch-based coordination conflicts some of the inherent features of the 802.11 MAC, which may prevent spatial channel reuse and thus underutilize wireless media [18]. The analysis of a single packet transmission will ignore this problem and explicitly emphasise the advantage of opportunistic routing. A more accurate analysis model is needed to better quantify the benefits and the cost of opportunistic routing and investigate the effect of batch-based transmissions. Indeed, such analysis model can be considered as an important next step towards modelling general scenarios of opportunistic routing.

In this chapter, we will propose an analytical model based on the Markov chain to evaluate the performance of opportunistic routing. Compared to previous analytical models which only analyze a single packet transmission in a network. Our model has following

three contributions:

- In wireless mesh network, nodes can make transmissions simultaneously if there are few destructive interference. Our models simulate the pipelined transmission and analyze both the expected number of transmission and time slots for a batch of packets.

- In opportunistic routing, each sender can have multiple potential receivers instead of a pre-decided receiver in traditional routing. Therefore, if two senders decide to transmit packets at the same time, they should consider the interference for multiple receivers. In our model, we will introduce *pipeline set* to consider the interference for multiple potential receivers.

- Opportunistic routing transmits a batch of packets together which may prevent spatial channel reuse. We will discuss the situation where opportunistic routing sacrifices from simultaneously transmissions to deliver opportunistic gains.

3.2 Description of model

In this section, we describe how to model the progression of packet transmissions in the network with a Discrete Time Markov Chain. We map the network state of the packet advancement progression with a Markov chain because the conditional probability distribution of a future network state depends only on the present network state and has no dependence of the sequence of states before the present state. The memoryless characteristic of the Markov chain is similar to the packet transmission progression of opportunistic routing if there is a strong coordination scheme. Here, a strong coordination

in opportunistic routing means all nodes can receive the coordination message correctly. Therefore, if more than one candidate receives a packet, only the highest priority node forwards this packet while other candidates drop the same packet. It is also the same assumption used in previous analytical models [66] [25]. As we see, the properties of packet transmissions are very similar to state transitions of a Markov chain. Therefore, we use a Markov chain to analyze the performance of opportunistic routing.

Coordination and candidate selection are defined as fundamental elements of opportunistic routing. Since the ExOR coordination scheme is widely investigated, we validate the state transition of our Markov chain by it. Our Markov model is independent from the candidate selection algorithm. In the following description, we use the same candidate selection algorithm as ExOR. In fact, it supports different candidate selection algorithms. After choosing a candidate selection algorithm, the number of candidates and the packet success rate of a link between each candidate are only required input parameters in our model.

3.2.1 Markov model of packet advancement progress

Here, we illustrate how we can model the packet advancement progression using a Markov chain. In order to simplify the description of our model, we will give a general model, then illustrate the model with a simple example.

The general description of our model is defined as follows. For a network with N nodes, allowing B packets to be transmitted in one batch, and the maximum number of candidate forwarders of C. The network state can be represented as $\langle \vec{P}, \vec{t} \rangle$, with $\vec{P} = \langle v_1, v_2, v_3, ..., v_B \rangle$ and $\vec{t} = \langle t_1, t_2, ..., t_i, ..., t_B \rangle$. Specifically, $\vec{P}$ denotes the progression

of each packet and v_i represents that packet i is received by node v, which is the highest-priority node having received this packet. Notation $\vec{t}$ is a binary vector, and its dimension is also the batch size B. Each element represents whether the corresponding packet will be transmitted or not, i.e. $t_i = 1$ for packet i being transmitted next. To consider the expected number of transmissions needed to transport a batch of packets, each network state only has one packet to be transmitted next. That is, $t_i \in \{0, 1\}$ $(i = 1, 2, ..., B)$ and $\sum_{i=1}^{B} t_i = 1$. We use a special vector $\vec{t} = \langle 0, 0, 0, ..., 0 \rangle$ to denote the network state where all packets have reached the destination. Thus, the initial network state is $\langle\langle 1, 1, 1, ..., 1\rangle, \langle 1, 0, 0, ..., 0\rangle\rangle$ and the final network state is $\langle\langle n, n, n, ..., n\rangle, \langle 0, 0, 0, ..., 0\rangle\rangle$. The total number of valid states is $\mathbb{N} = N^B \times B - N^{B-1} \times B + 1$. Specifically, the number of all combinations of the values is $N^B \times B$. $N^{B-1} \times B$ of these combinations are invalid, i.e. those taking the form of $v_i = n$ and $t_i = 1$ for $i = 1, 2, \ldots, B$, where packet i has reached the destination and no longer needs to be transmitted. Note that a more compact representation of the network state would be $\langle \vec{P}, t \rangle$, where $t \in \{1, 2, \ldots, B\}$ is the index of the packet to be transmitted next. However, we chose the binary vector form so that we can use the same notation to analyze the time slot metric where each network state can have more than one packet to be transmitted as we will see shortly.

One example to illustrate the network state is shown in Figure 3.1. It is a linear topology with $N = 4$, $B = 2$ and $C = 2$. The transmission range is up to two hops. Assuming that the packet success rate in one-hop link is p_1 and in two-hop link is p_2. Therefore, each transmission can reach as far as two-hop down from the sender with the possibility of $P_2 = p_2$. The possibility of reaching only one-hop away is $P_1 = p_1 \times (1 - p_2)$ and the transmission fails to reach any node with possibility of $P_0 = (1 - p_1) \times (1 - p_2)$. In the figure, the current packet transmission progressing of the network can be represented

by $\langle\langle 2,3\rangle, \langle 0,1\rangle\rangle$. Specifically, $\langle 2,3\rangle$ represents the current network state where packet 1 reaches node 2 and packet 2 reaches node 3. The second element $\langle 0,1\rangle$ describes the schedule of opportunistic forwarding, where packet 2 is scheduled to be transmitted next.

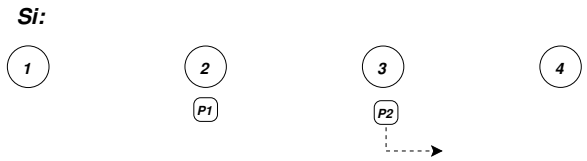

Figure 3.1: An example of network state.

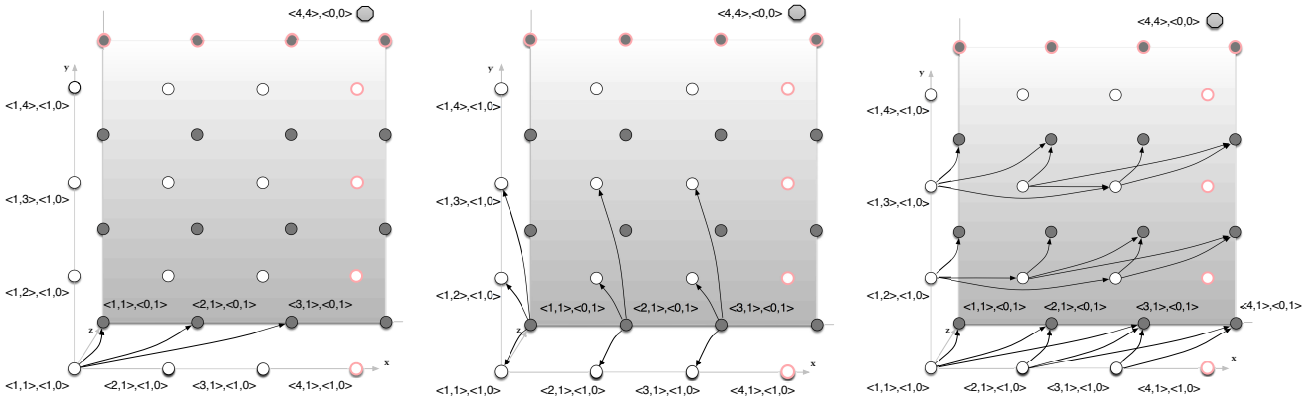

Figure 3.2: The first three steps of state transitions in a Markov chain.

Each state in our Markov chain represents a network state of packet transportation. The state transition is triggered by packet transmissions. We use the same coordination scheme in ExOR [10]. After a sender transmits a batch of packets, a subset of downstream nodes can receive some packets. The highest-priority node in these nodes would

be scheduled to transmit packets subsequently. Among all received packets by this node, the packet with the smallest index will be transmitted first. The remaining packets will be transmitted successively with increasing indices. After the highest priority node has transmitted all received packets, the next high-priority node on the sender's candidate list will transmit its received packets that are not received by higher-priority nodes. After it finishes, the next node on the list starts. This continues until all nodes on the list have attempted to transmit their packets.

In the linear network topology depicted in Figure 3.1, the packet progression of opportunistic routing can be mapped to the state transition of a Markov chain. The detail description of the first three state transitions is shown in Figure 3.2. There are two planes, the white plane represents packet 1 is to be transmitted and the dark plane represents packet 2 is to be transmitted. The x axis shows the progression of packet 1 from node 1 to node 4 and y axis shows the progression of packet 2 from node 1 to node 4. The bottom-left state is the initial state $\langle\langle 1,1\rangle,\langle 1,0\rangle\rangle$ and the top-right state is the final state $\langle\langle 4,4\rangle,\langle 0,0\rangle\rangle$. Because transitions in the x and y axes are monotonic in that they are always left to right or bottom to up, after a number of state transitions, the network state will eventually change from the initial state to the final state. The state does transit between the two planes back and forth. When a packet has reached the destination, it no longer needs to be transmitted. So there is no possibility to reach the states with the red color border.

Figure 3.2(a) shows the first step of state transitions from the initial state; Figures 3.2(b) and 3.2(c) show all possible state transitions for second and third steps. In Figure 3.2(a), as node 1 transmit packet 1, the initial state $\langle\langle 1,1\rangle,\langle 1,0\rangle\rangle$ can transmit to the three states of $\langle\langle 1,1\rangle,\langle 0,1\rangle\rangle$, $\langle\langle 2,1\rangle,\langle 0,1\rangle\rangle$ and $\langle\langle 3,1\rangle,\langle 0,1\rangle\rangle$ with probabilities P_0, P_1, P_2,

respectively. The state transition to $\langle\langle 3,1\rangle,\langle 0,1\rangle\rangle$ represents that packet 1 reaches node 3 with the probability of P_2 as defined previously. Alternatively, the state can transmit to $\langle\langle 2,1\rangle,\langle 0,1\rangle\rangle$ if the packet 1 reaches node 2 but not node 3. The probability of this state transition is P_1. If the packet 1 has no progression, the initial state will transit to $\langle\langle 1,1\rangle,\langle 0,1\rangle\rangle$ and the probability is P_0. After the sender transmits the first packet, it will send the second packet. Therefore, all three next states are in dark plane.

Figure 3.2(b) shows all possible state transitions after the first step. For example, if the state transmits to $\langle\langle 1,1\rangle,\langle 0,1\rangle\rangle$ in the first step, the second step can transit to $\langle\langle 1,1\rangle,\langle 1,0\rangle\rangle$, $\langle\langle 1,2\rangle,\langle 1,0\rangle\rangle$ and $\langle\langle 1,3\rangle,\langle 1,0\rangle\rangle$. Figure 3.2(c) shows all possible state transitions after the second step. The state transitions in the third step are similar to two previous steps, but it also shows two special cases. The first special case is where a small-index packet is transmitted and gets ahead of large-index packets, e.g., the state transition from $\langle\langle 1,2\rangle,\langle 1,0\rangle\rangle$ to $\langle\langle 2,2\rangle,\langle 1,0\rangle\rangle$ or $\langle\langle 3,2\rangle,\langle 1,0\rangle\rangle$. If packet 1 reaches node 2, it will be transmitted subsequently since it will be the smallest index in node 2. Therefore, the state transition mapping this transmission will be $\langle\langle 1,2\rangle,\langle 1,0\rangle\rangle$ to $\langle\langle 2,2\rangle,\langle 1,0\rangle\rangle$. Similarly, if packet 1 reaches node 3, it will be transmitted subsequently since node 3 has higher-priority than node 2. The state transition will be $\langle\langle 1,2\rangle,\langle 1,0\rangle\rangle$ to $\langle\langle 3,2\rangle,\langle 1,0\rangle\rangle$. The second special case is that a packet reaches a node that is one-hop away from the destination. In this case, there are no two-hop receptions any more, e.g., state $\langle\langle 1,3\rangle,\langle 0,1\rangle\rangle$ can only transit to $\langle\langle 1,3\rangle,\langle 1,0\rangle\rangle$ or $\langle\langle 1,4\rangle,\langle 1,0\rangle\rangle$.

Our model is in fact a general mathematical analysis for different types of forwarding protocols. Besides opportunistic routing protocols, it can also map the progression of packet transmissions of traditional routing protocols. The illustration of state transitions is shown Figure 3.3. In traditional routing, the routing path is pre-defined before the

source sends packets. The traditional routing protocols will first transmit the packet with the smallest index in a batch, then move to the next one. Note that there is no pipeline transmission in the Figure because this topology is too short to allow multiple transmissions without interferences. As we will see later, our model can be further generalized to incorporate pipeline transmissions for both traditional and opportunistic routing forwarding. Only seven states are valid in this example. Each node forwards packets to a dedicated next-hop forwarder; if the transmission fails, the sender will retransmit packets. Therefore, each state only has two out-going edges with nonzero probabilities. Note that in this case the packet reception probability can be either a single transmission attempt or a compound of retransmits.

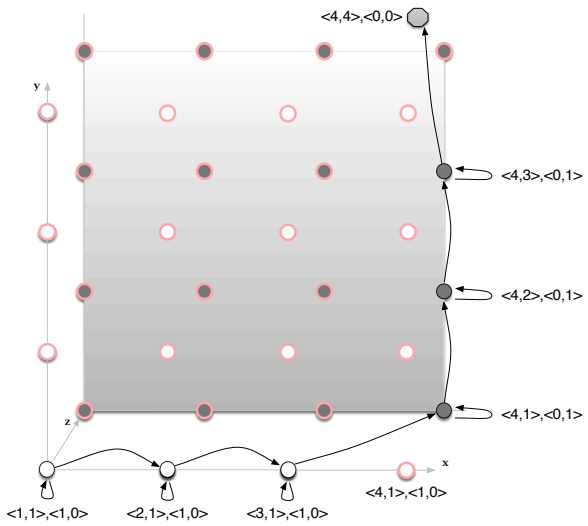

Figure 3.3: All state transitions of traditional routing in a Markov chain.

3.2.2 Metrics of opportunistic routing

3.2.2.1 Expected number of transmissions

The expected number of transmissions provides an understanding of the efforts required to successfully deliver a batch of packets from a source to a destination. This metric is the an important parameter to evaluate opportunistic routing protocols. Using this metric, opportunistic routing protocols can show great improvement compared to traditional routing protocols because they can reduce the number of transmissions significantly.

This metric can be derived from our model based on the theory of Markov chain. The initial state of the Markov chain is $\langle\langle 1, 1, 1, ..., 1\rangle, \langle 1, 0, 0, ..., 0\rangle\rangle$ and final state is $\langle\langle n, n, n, ..., n\rangle, \langle 0, 0, 0, ..., 0\rangle\rangle$. The expected number of state transitions from the initial state to the final state is the average number of transmissions for the source to transmit a batch of packets to the destination. For example, in the Markov chain in Figure 3.2, the initial state is $\langle\langle 1, 1\rangle, \langle 1, 0\rangle\rangle$ and the final state is $\langle\langle 4, 4\rangle, \langle 0, 0\rangle\rangle$. Two packets need to be transmitted from the source node 1 to the destination node 4. If the packet transmission fails, the state will move along the z axis (i.e. between planes) and not transit towards the final state. If the packet makes any progress, the state will transit towards the final state along either the x axis or y axis. Eventually, when all packets reach the destination, the state will transit to the final state. In each state transition, only one packet will be transmitted. Therefore, the expected number of packet transmissions is exactly same as the expected number of state transitions.

3.2.2.2 Expected number of time slots

In this section, we will consider the other metric to analyze opportunistic routing. The expected number of time slots will calculate how many time slots required to forward a batch of packets from the source to the destination. Without the pipelined transfer, the expected number of time slots is the same as the expected number of transmission. Due to spatial separation, several nodes can make wireless transmissions simultaneously if there are few destructive interferences. In our model, we consider the interference model proposed in Gupta and Kumar [35], called the *Protocol Model*. Suppose node i transmits to node k. This transmission is successfully received by node k if

$$|X_j - X_k| \geq (1 + \Delta)|X_i - X_k| \tag{3.1}$$

for other nodes j simultaneously transmitting over the same channel. X_i is location of node i and $|X_i - X_j|$ is the Euclidean distance between the two nodes. Δ is chosen by the protocol to prevent a neighbouring node from transmitting on the same time. In this model, any pair of nodes i and j can make simultaneous transmissions if they satisfy the Equation 3.1.

Our model considers multiple packet transmissions into one state transition. The network state can still be represented as $\langle \vec{P}, \vec{t} \rangle$, where $\vec{P} = \langle v_1, v_2, v_3, ..., v_B \rangle$ and $\vec{t} = \langle t_1, t_2, t_3, ..., t_B \rangle$. That is, $\vec{P}$ denotes the progression of each packet and $\vec{t}$ represents which packets will be transmitted next. Unlike the previous network formulation, more that one t_i can be equal to 1. It represents that more than one packet will be transmitted in one time slot. Each state can have more nonzero possibilities of going to other states since multiple packet transmissions are considered in one state transition. As a result, the number of state transitions represents the number of time slots in the network.

In traditional routing, node i can transmit packets to node k while node j transmitting packets if node j can not interfere the transmission from node i to k. In opportunistic routing, node i can have more than one potential receivers $k = k_1, k_2, k_3, ..., k_x$. If nodes i and j should satisfy the Equation 3.1 for all potential receivers to make simultaneous transmissions. The possibility of simultaneous transmissions is greatly limited. Especially, some of potential receivers are a little bit far away from the sender and have small possibilities to receive packets. Considering these potential receivers, opportunistic routing will extend the interference range and reduce the possibility of simultaneous transmissions. On the other hand, if only one potential receiver k_x is considered to decide whether nodes i and j can make simultaneous transmissions, the receiving process for other potential receivers will be interfered. In this case, only one receiver is utilized for packet transmissions. Opportunistic routing can not consider multiple potential receivers where the advantage of it will be limited. Therefore, there is tradeoff between opportunistic gains to reduce the number of transmissions and the utilization of simultaneous transmissions.

The *pipeline set* decides which transmissions can happen at the same time according to the interference model. We define $\vec{S_i^k} = \langle S_{i,1}^k, S_{i,2}^k, S_{i,3}^k, ..., S_{i,j}^k, ..., S_{i,n}^k \rangle^T$, which is a binary vector and represents whether the transmission from node i to node k can happen simultaneously with the transmission from j to k. We use $\mathbb{S}_i^k$ to denote the pipeline set of nodes that can transmit at the same time with the transmission from i to k. For instance, $\vec{S_i^k} = \langle 0, 0, 0, 0, 0, 0, 1, 1, 1, 1 \rangle^T$, the pipeline set $\mathbb{S}_i^k = \{7, 8, 9, 10\}$. In opportunistic routing, each transmitting node i can have multiple potential receivers $k_1, k_2, ..., k_x$; each receivers can have its own pipeline set $\mathbb{S}_i^k$. Note that there can be many ways for a node i to define its pipeline set. Here, we consider two extreme cases, $\mathbb{S}_i^{\cap} = \mathbb{S}_i^{k_1} \cap \mathbb{S}_i^{k_2} \cap \mathbb{S}_i^{k_3} \cap \ldots \mathbb{S}_i^{k_x}$ and $\mathbb{S}_i^{\cup} = \mathbb{S}_i^{k_1} \cup \mathbb{S}_i^{k_2} \cup \mathbb{S}_i^{k_3} \cup \ldots \mathbb{S}_i^{k_x}$. We will compare the performance of opportunistic routing

in these two extreme cases.

3.3 Solving matrix

3.3.1 Transition matrix

The transition matrix consists of the transition probabilities between network states. We have two sets of network states. The first set of network states considers the number of packet transmissions, each state transition represents one packet transmissions. The second set of network states considers the number of time slots, each state transition represents one time slot. Some network states of time slots will be same as the network states of packet transmissions if only one packet can be transmitted in these network states. In both sets of network states, the initial state and the final state are same and the final states $\langle\langle n, n, n, ..., n\rangle, \langle 0, 0, 0, ..., 0\rangle\rangle$ is an *absorbing state*, where the transition probabilities from an absorbing state to other states are always zero and the probability to itself is always 1. In this section, we defined two transition matrices M_{tx} and M_{time} to consider the transition probabilities in these two sets of network states.

First, we define the transition probability of M_{tx}. In essence, these transitions correspond to the events of a packet being transmitted by a node and received by another. Specifically, assume packet i is transmitted by node v_i and received by node v_i'. Recall that v_i has a fixed set of candidates $\{c_1, c_2, \ldots, c_{C_i}\}$, where their priorities increase from c_1 to c_{C_i}. Each v_i can have a different set of candidates and C_i is the number of candidates of v_i. Furthermore, we denote the link reliability between v_i and these candidates as $\{\pi_1, \pi_2, \ldots, \pi_{C_i}\}$. Therefore, the probability of packet i being received

by candidate v_i' but none of the higher-priority candidates of v_i is defined as T_i in Equation 3.2. Here, S_x is $\langle\langle v_1, v_2, \ldots, v_i, \ldots, v_B\rangle, \langle t_1, t_2, \ldots, t_i, \ldots, t_B\rangle\rangle$,$t_i = 1$ and S_y is $\langle\langle {v_1}', {v_2}', \ldots, {v_i}', \ldots, {v_B}'\rangle, \langle {t_1}', {t_2}', \ldots, {t_i}', \ldots, {t_B}'\rangle\rangle$. The first case means the packet is received by the n-th highest priority v_i', but none of higher-priority node from $n+1$ to C_i receives it. These nodes with the priority from 1 to $n-1$ may or may not receive this packet. With the assumption of strong coordination, all nodes can know the coordination message correctly. The n-th highest priority node will keep this packet and other priority nodes from 1 to $n-1$ will drop the same packet. Therefore, the probability of the state transition from S_x to S_y is the multiplication of the packet success rate from v_i to v_i' and the packet fail rates from v_i to other higher-priority nodes from $n+1$ to C_i. The second case is that the packet fails to reach any candidate of v_i. Apparently, this matrix is $\mathbb{N} \times \mathbb{N}$, where $\mathbb{N} = N^B \times B - N^{B-1} \times B + 1$. Each row has at most $C+1$ non-zero entries because a transmission can only involve that many nodes. Thus, it is a very sparse matrix. Equation 3.3 is an example of M_{tx} for the 4-node linear network topology in Figure 3.1. In particular, the transition probability between two states S_x and S_y has three general cases. Probability P_0 means packet i makes no progress in advancing towards the destination. P_1 means packet i is received by the next hop but no further. P_2 means the packet is received by the $2nd$ next hop. In this example, we consider the transmission range up to two hops. Our model supports the transmission range up to the diameter of the network.

$$P\{S_x \to S_y\} = T_i = \begin{cases} \pi_n \prod_{k=n+1}^{C_i} (1 - \pi_k) & \text{for} \quad {v_i}' = v_i + n \\ \prod_{k=1}^{C_i} (1 - \pi_k) & \text{for} \quad {v_i}' = v_i \\ 0 & \text{otherwise} \end{cases} \tag{3.2}$$

$$
P = \begin{array}{c}
\\
\langle 1,1\rangle,\langle 1,0\rangle \\
\langle 1,1\rangle,\langle 0,1\rangle \\
\langle 1,2\rangle,\langle 1,0\rangle \\
\langle 1,2\rangle,\langle 0,1\rangle \\
\langle 1,3\rangle,\langle 1,0\rangle \\
\langle 1,3\rangle,\langle 0,1\rangle \\
\langle 1,4\rangle,\langle 1,0\rangle \\
\langle 2,1\rangle,\langle 1,0\rangle \\
\langle 2,1\rangle,\langle 0,1\rangle \\
\langle 2,2\rangle,\langle 1,0\rangle \\
\langle 2,2\rangle,\langle 0,1\rangle \\
\langle 2,3\rangle,\langle 1,0\rangle \\
\langle 2,3\rangle,\langle 0,1\rangle \\
\langle 2,4\rangle,\langle 1,0\rangle \\
\langle 3,1\rangle,\langle 1,0\rangle \\
\langle 3,1\rangle,\langle 0,1\rangle \\
\langle 3,2\rangle,\langle 1,0\rangle \\
\langle 3,2\rangle,\langle 0,1\rangle \\
\langle 3,3\rangle,\langle 1,0\rangle \\
\langle 3,3\rangle,\langle 0,1\rangle \\
\langle 3,4\rangle,\langle 1,0\rangle \\
\langle 4,1\rangle,\langle 0,1\rangle \\
\langle 4,2\rangle,\langle 0,1\rangle \\
\langle 4,3\rangle,\langle 0,1\rangle \\
\langle 4,4\rangle,\langle 0,0\rangle
\end{array}
\begin{array}{c}
\begin{array}{cccccccccccccccccccccccc}
\langle 1,1\rangle,\langle 1,0\rangle & \langle 1,1\rangle,\langle 0,1\rangle & \langle 1,2\rangle,\langle 1,0\rangle & \langle 1,2\rangle,\langle 0,1\rangle & \langle 1,3\rangle,\langle 1,0\rangle & \langle 1,3\rangle,\langle 0,1\rangle & \langle 1,4\rangle,\langle 1,0\rangle & \langle 2,1\rangle,\langle 1,0\rangle & \langle 2,1\rangle,\langle 0,1\rangle & \langle 2,2\rangle,\langle 1,0\rangle & \langle 2,2\rangle,\langle 0,1\rangle & \langle 2,3\rangle,\langle 1,0\rangle & \langle 2,3\rangle,\langle 0,1\rangle & \langle 2,4\rangle,\langle 1,0\rangle & \langle 3,1\rangle,\langle 1,0\rangle & \langle 3,1\rangle,\langle 0,1\rangle & \langle 3,2\rangle,\langle 1,0\rangle & \langle 3,2\rangle,\langle 0,1\rangle & \langle 3,3\rangle,\langle 1,0\rangle & \langle 3,3\rangle,\langle 0,1\rangle & \langle 3,4\rangle,\langle 1,0\rangle & \langle 4,1\rangle,\langle 0,1\rangle & \langle 4,2\rangle,\langle 0,1\rangle & \langle 4,3\rangle,\langle 0,1\rangle & \langle 4,4\rangle,\langle 0,0\rangle
\end{array} \\
\left[\begin{array}{cccccccccccccccccccccccc}
0 & P_0 & 0 & 0 & 0 & 0 & 0 & 0 & P_1 & 0 & 0 & 0 & 0 & 0 & 0 & P_2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
P_0 & 0 & P_1 & 0 & P_2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & P_0 & 0 & 0 & 0 & 0 & 0 & P_1 & 0 & 0 & 0 & 0 & 0 & 0 & P_2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & P_0 & 0 & P_1 & 0 & P_2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & P_0 & 0 & 0 & 0 & 0 & 0 & 0 & P_1 & 0 & 0 & 0 & 0 & 0 & P_2 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & P_0 & 0 & P_1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & P_0 & 0 & 0 & 0 & 0 & 0 & 0 & P_1 & 0 & 0 & 0 & 0 & 0 & 0 & P_2 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & P_0 & 0 & 0 & 0 & 0 & 0 & 0 & P_1 & 0 & 0 & 0 & 0 & 0 & P_2 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & P_0 & 0 & P_1 & 0 & 0 & P_2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\
\hline
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1
\end{array}\right]
\end{array}
\tag{3.3}
$$

Next is the transition matrix of M_{time} used to calculate the expected number of time slots. In the state transition corresponding to one time slot, multiple packets may be transmitted. We describe the probabilities between this state transition as

$$P\{S_x \to S_y\} = \prod_{i=1}^{B} T_i^{t_i}. \tag{3.4}$$

Here, S_x is $\langle\langle v_1, v_2, \ldots, v_i, \ldots, v_B\rangle, \langle t_1, t_2, \ldots, t_i, \ldots, t_B\rangle\rangle$ and

S_y is $\langle\langle {v_1}', {v_2}', \ldots, {v_i}', \ldots, {v'}_B\rangle, \langle {t_1}', {t_2}', \ldots, {t_i}', \ldots, {t_B}'\rangle\rangle$. Recall that $t_i \in \{0, 1\}$ $(i = 1, 2, \ldots, B)$ represents whether packet i is being transmitted. In the state S_x, it can have more than one $t_i = 1$. Here T_i is from Equation 3.2 which is the probability of the state transition where only one packet is transmitted. Each factor of Equation 3.4 corresponds to the progression of a packet being transmitted. Thus, it can represent the probability of a state transition where multiple simultaneous transmissions occur.

3.3.2 Analysis of expected number of state transitions in the Markov chains

With these two transition matrices, we can calculate the expected number of packet transmissions and the expected number of time slots, respectively. The value in the first row and last column of the transition matrix represents the probability that the initial state goes to the final state by one step. In general, this value in the k-th power of the transition matrix represents the probability that the initial state transits to the final state in at most k steps, noticing that the final state is absorbing. Recall that one state transition corresponds to either one packet transmission or one time slot in these two matrices. Thus, we can obtain the expected number of packet transmissions and the expected number of time slots by calculating the expected number of state transitions

from the initial state to the final state. Because these matrices are the same in nature, we do not distinguish them in the rest of this section.

Observe that the matrix is absorbing as all states can go to the final state, such a matrix has a *canonical form* as

$$P = \begin{pmatrix} Q & R \\ 0 & I \end{pmatrix}. \tag{3.5}$$

The number of columns of R and I is equal to the number of absorbing states in P. I, an identity matrix, represents that the probability of an absorbing state to itself is always 1. The probability of reaching the final state from the initial state in k steps is the value in the first row and last column of k-th power of the transition matrix:

$$P^k = \begin{pmatrix} Q^k & (I + Q + ... + Q^{k-1}) \times R \\ 0 & I \end{pmatrix}. \tag{3.6}$$

When $k \to \infty$, P^k has a limit

$$P^{k\to\infty} = \begin{pmatrix} 0 & \hat{R} \\ 0 & I \end{pmatrix}. \tag{3.7}$$

Here, the number of columns of $\hat{R}$ is equal to the number of absorbing states. The sum of each row of $\hat{R}$ is equal to 1. Thus, all states will eventually transit to one of the absorbing states. In our case, we only have one absorbing state, the limit is

$$P^{k\to\infty} = \begin{pmatrix} 0_{(\mathbb{N}-1)\times(\mathbb{N}-1)} & 1_{(\mathbb{N}-1)\times 1} \\ 0_{1\times(\mathbb{N}-1)} & 1_{1\times 1} \end{pmatrix}. \tag{3.8}$$

That is, only the last column is 1, with the rest of the matrix being all 0s. Therefore, the probability of transiting from initial state to the final state in 'exactly k steps' can be

described as

$$\begin{aligned} P[X=k] &= \left[\left(I+Q+...+Q^{k-1}\right)\times R\right]_{1,1} \\ &\quad - \left[\left(I+Q+...+Q^{k-2}\right)\times R\right]_{1,1} \\ &= \left[Q^{k-1}\times R\right]_{1,1}. \end{aligned} \tag{3.9}$$

That is, since the final state is absorbing, if the network state goes to the final state in $k-1$ steps, it will still stay there. The difference is Equation refeq:probabilityK is the probability that the chain transits to the final state in k steps but not sooner.

We are interested in how long it takes the Markov Chain to go from $\langle\langle 1,1,1,...,1\rangle,\langle 1,0,0,...,0\rangle\rangle$ to $\langle\langle n,n,n,...,n\rangle,\langle 0,0,0,...,0\rangle\rangle$. This average number of transitions can be calculated [33] as

$$\begin{aligned} E[X] &= \sum_{k=1}^{\infty}(k\times p[X=k]) \\ &= \frac{[F^2\times R]_{1,1}}{[F\times R]_{1,1}} \\ &= F_{1,1}, \end{aligned} \tag{3.10}$$

where $F = I+Q+...+Q^{k-1}+....$ Note that $F\times(I-Q) = (I-Q^{\infty}) = I$. Therefore $F=(I-Q)^{-1}$. The number of state transitions from the initial state to the final state can be obtained by calculating the inverse matrix of the transition matrix, where the time complexity is $O(\mathbb{N}^3)$.

For lower computation cost, we choose iteratively multiply the transition matrix until the probability of transition from $\langle\langle 1,1,1,...,1\rangle,\langle 1,0,0,...,0\rangle\rangle$ to $\langle\langle n,n,n,...,n\rangle,\langle 0,0,0,...,0\rangle\rangle$

is sufficiently close to 1 (i.e. within $1-\epsilon$). The iterative calculation can be described as follows. The expected number of state transitions is

$$E[X] = \sum_{k=1}^{T} (k \times p[X = k]), \tag{3.11}$$

with $P[X = T] \geq 1 - \epsilon$. As a result of replacing inversion with multiplication, the time complexity reduces to $O(\mathbb{N}^2)$. For our problem, $O(\mathbb{N}^2)$ is acceptable for the estimation of transitions where $\mathbb{N}$ is of the order of 10^5 or less. For greater value of N, we further resort to a sampling technique using random walk.

The starting point of the random walk is $\langle\langle 1, 1, 1, ..., 1\rangle, \langle 1, 0, 0, ..., 0\rangle\rangle$, and the end point is $\langle\langle n, n, n, ..., n\rangle, \langle 0, 0, 0, ..., 0\rangle\rangle$. Each step is dictated by the probability of the state transition. For each walk, we record the number of steps taken with the given parameter values of N, B and C. We perform a large number of random walks, the mean of these samples will give us a good estimation of the expected number of state transitions. The time complexity further reduces to $O(\mathbb{N})$.

3.4 Simulation results

This section presents analytical results and simulation results from NS-2 [1] to evaluate the model of opportunistic routing. The analytical model will be calculated by two methods: the iteratively multiplication and the sampleing technique using random walk. We first compare the iteratively multiplication and sampleing technique in Section 3.4.1 and show the random walk can generate results close to the the iteratively multiplication. Since the time complexity of random walk is $O(\mathbb{N})$, we will use random walk to reflect analytical results in following works. In Sections 3.4.2, 3.4.3 and 3.4.4, we will compare

the analytical results with the simulation results in different scenarios to evaluate the precision of the models. Then, we will discuss the advantage and cost of opportunistic routing.

In linear topologies, we consider networks of N nodes ($N = 10, 15, 20$, and 25) with different batch sizes from 10 to 100. The packet success rates of each link are indicated in the Table 5.1. We consider five different groups of packet success rates. The comparision of analytical models and NS2 simulation in linear topology can not only prove the precision of the models but also show the pros and cons of opportunistic routing. In grid topologies, we consider two different pipeline sets and packet success rates are specified in Figure 5.5. We will extend the number of nodes from $N = 10 \times 10$ to $N = 20 \times 20$ for discussing the performance of opportunistic routing and the precision of our models in different network diameters with a same network density. Free space and two-ray ground propagation, consider the received power of the sender as a deterministic function of distance. These two models consider the communication range as a perfect distance. If a node is located within this distance, it will receive the transmitted packet, whereas the nodes outside this range will not receive the packet. In reality, due to multipath propagation effects, the received power at a certain distance is a random variable. Therefore, we have used the shadowing propagation model, which considers the received power as a random variable due to fading effects. Our model uses the same propagation model and parameters as in [25]. We will consider a square with sides equal to 1000m. The number of nodes in our random topology varied from $N = 30$ to 50 nodes. The source node is located at the bottom-left and the destination node is located at the top-right. We will show how the performance of opportunistic routing in the topology with a constant network diameter but different network densities.

Table 3.1: Packet success rate

One-hop	Two-hop
100%	50%
90%	45%
80%	40%
70%	35%
60%	30%

For each scenario, one FTP application sends long files from the source to the destination. The source node emits a batch of packets continuously until the end of the simulation, and each simulation lasts for 100 seconds. All the wireless links have a bandwidth 1Mbps and the buffer size on the interfaces is set to 100 packets. All nodes operate in the 802.11 broadcast mode. The results of each scenario is the average of 100 runs.

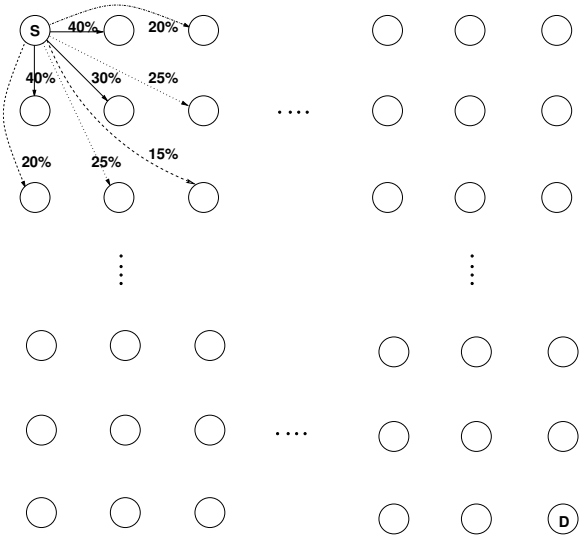

Figure 3.4: The grid topology.

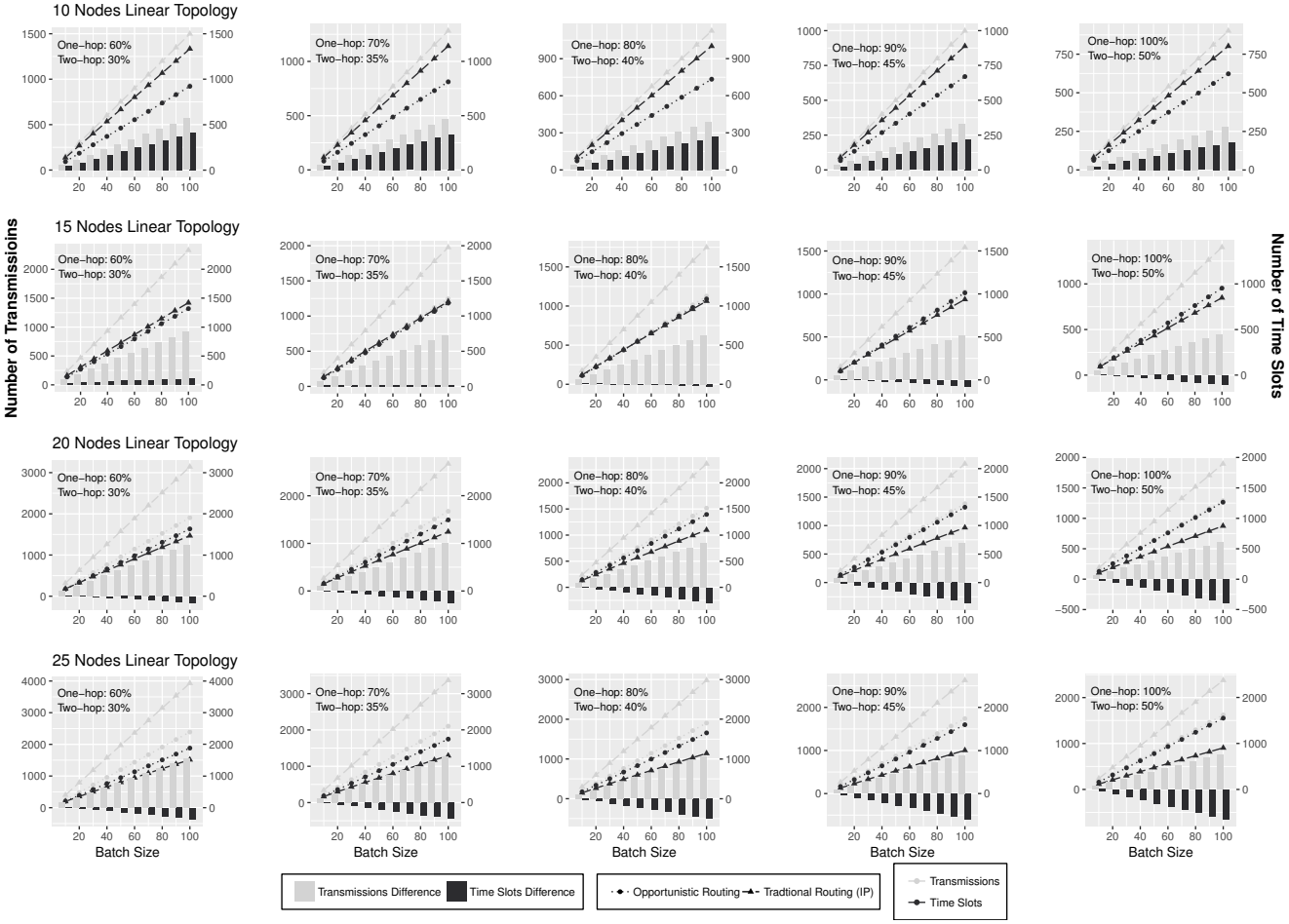

Figure 3.5: The expected number of transmissions and time slots for linear topologies.

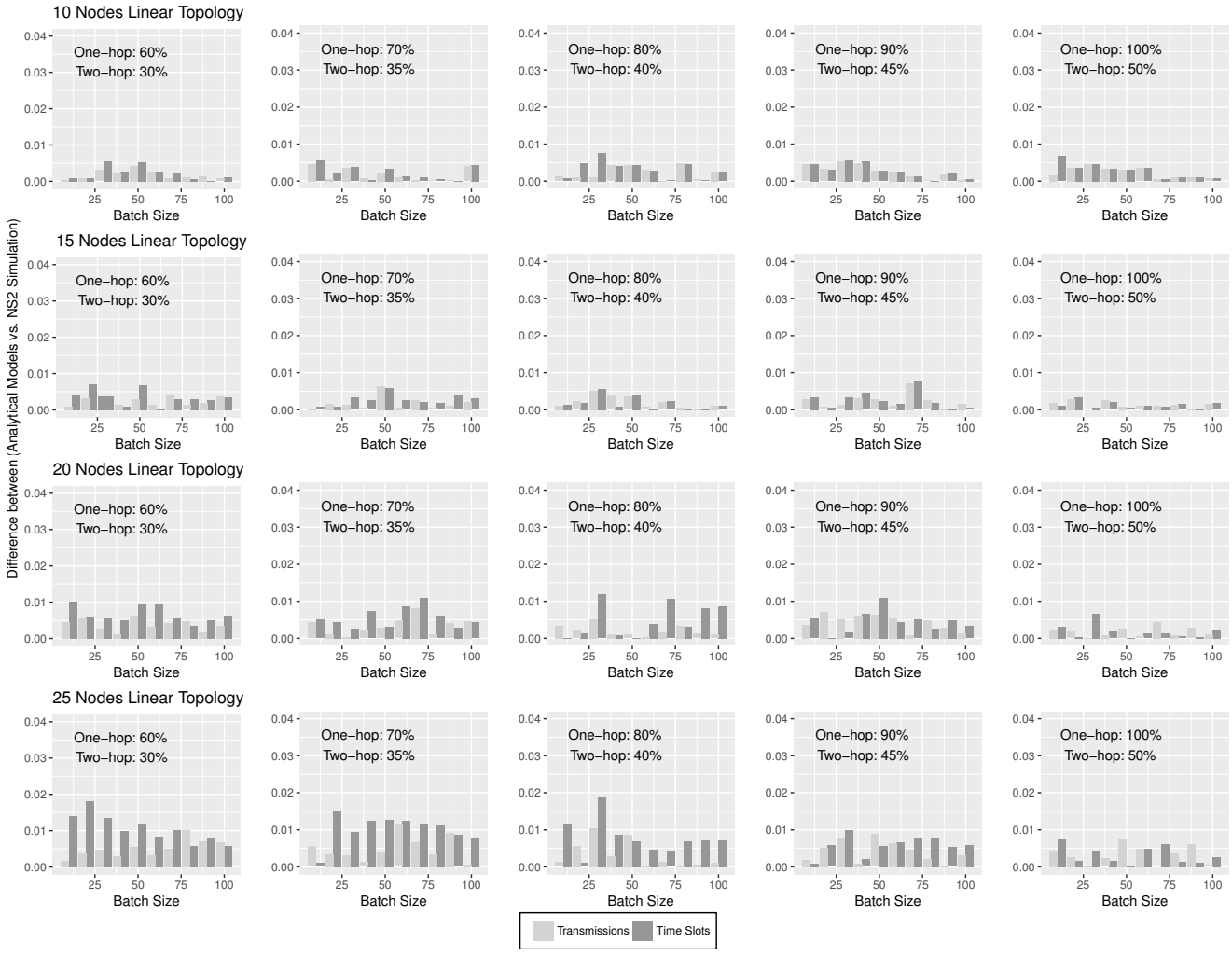

Figure 3.6: The difference between analytical model and NS2 simulation.

Table 3.2: Comparison of iterative multiplication and random walk

		$N=5$		$N=6$		$N=7$		$N=8$		$N=9$		$N=10$	
$C=1$	$B=2$	0.00020	8.88	0.00012	11.10	0.00018	13.33	0.00001	15.55	0.00005	17.77	0.00011	20.00
	$B=3$	0.00018	13.33	0.00033	16.66	0.00011	20.00	0.00007	23.33	0.00010	26.67	0.00006	30.02
	$B=4$	0.00005	17.77	0.00007	22.21	0.00010	26.67	0.00006	31.11	0.00006	35.56	0.00006	40.00
	$B=5$	0.00007	22.21	0.00005	27.77	0.00008	33.33	0.00012	38.89	0.00007	44.45	0.00011	50.21

		$N=5$		$N=6$		$N=7$		$N=8$		$N=9$		$N=10$	
$C=2$	$B=2$	0.00018	6.95	0.00049	8.59	0.00004	10.20	0.00011	11.83	0.00041	13.47	0.00004	15.08
	$B=3$	0.00009	10.42	0.00028	12.88	0.00001	15.30	0.00045	17.76	0.00005	20.19	0.00011	22.63
	$B=4$	0.00040	13.91	0.00012	17.16	0.00013	20.41	0.00037	23.66	0.00052	26.92	0.00016	30.17
	$B=5$	0.00002	17.37	0.00008	21.46	0.00021	25.51	0.00038	29.59	0.00041	33.65	0.00007	37.69

		$N=5$		$N=6$		$N=7$		$N=8$		$N=9$		$N=10$	
$C=3$	$B=2$	0.00028	6.17	0.00031	7.50	0.00036	8.81	0.00034	10.14	0.00041	11.46	0.00040	12.78
	$B=3$	0.00032	9.25	0.00032	11.25	0.00034	13.22	0.00036	15.21	0.00033	17.21	0.00038	19.18
	$B=4$	0.00023	12.36	0.00038	15.00	0.00041	17.60	0.00026	20.28	0.00021	22.92	0.00017	25.59
	$B=5$	0.00032	15.42	0.00029	18.76	0.00031	22.03	0.00045	25.35	0.00037	28.68	0.00029	31.96

		$N=5$		$N=6$		$N=7$		$N=8$		$N=9$		$N=10$	
$C=4$	$B=2$	0.00030	5.95	0.00038	7.19	0.00042	8.41	0.00039	9.64	0.00040	10.86	0.00044	12.08
	$B=3$	0.00034	8.91	0.00034	10.80	0.00038	12.63	0.00044	14.44	0.00041	16.29	0.00047	18.11
	$B=4$	0.00030	11.90	0.00028	14.39	0.00037	16.85	0.00045	19.26	0.00023	21.71	0.00037	24.15
	$B=5$	0.00036	14.86	0.00044	18.01	0.00029	21.05	0.00038	24.09	0.00044	27.15	0.00029	30.19

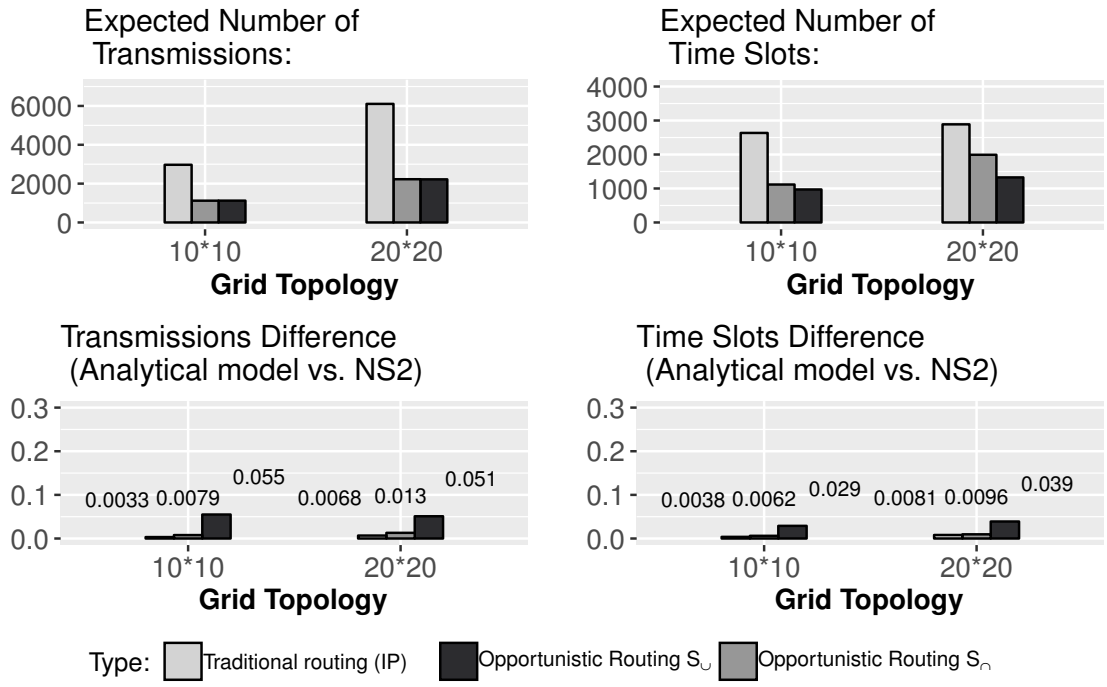

Figure 3.7: The expected number of transmissions and time slots for grid topologies.

3.4.1 Numerical comparison of iterative multiplication and random walk

In this section, we will evaluate our Markov models by iterative multiplication and random walk. Specifically, we are interested in the number of state transitions required to transfer a batch of data packets in a given network. Recall that the open form value of the expected steps of transitions has a computation complexity of $O(\mathbb{N}^3)$, which is prohibitive for large $\mathbb{N}$. Even for moderately large $\mathbb{N}$, the iterative multiplication estimation, which requires $O(\mathbb{N}^2)$, would need a great amount of CPU time. Resorting to random walk as a sampling technique, we can further reduce the time-complexity to $O(\mathbb{N})$. A random walk is a mathematical formalization of a path that consists of a succession of random steps. In our test, each step represents one state transition. For smaller $\mathbb{N}$ ($\mathbb{N} < 10^5$), we compare

the expected number of transitions from the iterative multiplication with that obtained from random walk. The results show that the normalized relative error is negligible. i.e. less than 0.1%. For the large $\mathbb{N}$ ($\mathbb{N} > 10^5$), we use the random walk to show the expected number of state transitions.

The numerical comparisons of iterative multiplication and random walk are conducted in linear networks, the packet success rates are 90% for one hop, 30% for two hops, 15% for three hops and 5% for four hops. To reduce the memory requirement and speed up the calculation of the transition matrix, we used Compressed Sparse Row to represent a sparse matrix by three one-dimensional arrays [13].

We set $1 - \epsilon = 0.999$ to control the termination of iterative multiplication and take 10,000 samples of random walk. Because two transition matrices M_{tx} and M_{time} are the same in nature, we only show the difference between iterative multiplication and random walk for M_{tx}. The difference of the expected number of state transitions between these two methods is shown in Table 5.1 with different parameters N, B and C. In each table cell, the value on the right is the expected number of state transitions from the random walk process and the value on the left is the normalized relative error between these two methods:

$$\varepsilon = \left| \frac{\tau_1 - \tau_2}{\tau_1 + \tau_2} \right|, \tag{3.12}$$

where τ_1 and τ_2 are mean number of transitions from the iterative multiplication and the random walk, respectively. From the results, we observe that the expected number of transitions increases with the increment of N and B and decrease with the increment of C. The normalized relative error is always less than 0.1%, it shows that random walk can generate results close to the iterative multiplication reliably.

3.4.2 Simulation results for linear topologies

In this section, we will analyze the performance of opportunistic routing in four linear topologies with different link qualities and batch sizes. Figure 3.5 shows the expected number of transmissions and time slots for opportunistic routing and traditional routing. The grey bar and black bar show the difference between them by the number from traditional routing minus that from opportunistic routing.

As a first observation, all grey bars are positive since opportunistic routing always has a smaller number of transmissions than traditional routing in all cases. In addition, increasing the number of nodes or increasing batch sizes causes a greater difference. Furthermore, when the link quality is good, the advantage of opportunistic routing is not as remarkable, specially when the number of nodes is 10. This is due to the fact that opportunistic routing would utilize pretty much the same nodes as traditional routing as potential forwarders when the link quality is high. When the link quality is not good, opportunistic routing shows more advantage than traditional routing.

The other metric that can be derived from the proposed model is the expected number of time slots. In linear topologies, we only consider the pipeline set $\mathbb{S}_{\cup}$. We can observe that some black bars are less than 0; it shows that opportunistic routing would have a larger number of time slots than traditional routing. Observing these figures from left to right, the value for each black bar decreases while the link quality becomes better. These phenomenon are remarkable when the number of nodes is 25 since traditional routing has more number of pipelined transmissions than that of opportunistic routing.

In the linear topologies with 10 nodes, we can observe that the number of transmission and time slots for traditional routing are close. For opportunistic routing, the lines for the

number of transmission and time slots are overlapped since few simultaneous transmissions occur. This is because that the network diameter is short and nodes will interfere with each other if multiple nodes make simultaneous transmissions. When the number of nodes increases to 25, the difference between the number of transmissions and the number of time slot for traditional routing shows that traditional routing has many simultaneous transmissions. In opportunistic routing, the difference between the number of transmission and the number of time slot is not remarkable. However, the difference increases from the right to left. It shows that opportunistic routing can have more pipelined transmissions when the link quality is not good. Considering both the number of transmissions and time slots, opportunistic routing greatly outperforms traditional routing under the short network diameter topology with low quality links.

Next, Figure 3.6 shows the precision of our models in linear topologies. The difference between the NS2 simulation and analytical models for the expected number of transmissions and time slots are plotted as bars. The difference is negligible and ranges from 1×10^{-3} to 1×10^{-2}. This small difference results from experimental scenarios in which a transmission may collide with another, consequently requiring retransmission of the packet, a situation not captured by the Markov model.

3.4.3 Simulation results for grid topologies

Figure 3.7 show the expected number of transmissions and time slots for both opportunistic routing and traditional routing in two grid topologies. Here, we consider two pipeline sets $\mathbb{S}_{\cap}$ and $\mathbb{S}_{\cup}$ for opportunistic routing. We can observe that opportunistic routing has better performance than traditional routing in both cases. We also plot the

difference between two pipeline sets $\mathbb{S}_{\cap}$ and $\mathbb{S}_{\cup}$. Opportunistic routing with these two pipeline sets has almost the same expected number of transmissions ($\mathbb{S}_{\cup}$ has a little bit more number of transmissions in NS-2 simulations). $\mathbb{S}_{\cup}$ has a smaller number of time slots. This is because $\mathbb{S}_{\cup}$ allows more nodes to transmit packets simultaneously, it will require less time slots to transmit a batch of packet. On the other hand, the pipeline set $\mathbb{S}_{\cup}$ involves more nodes and increases the probability that a transmission collides with another, consequently requiring more packet retransmissions. The difference between analytical results and simulation results are plotted at the bottom. The difference between them is also very small and ranges from 1×10^{-3} to 1×10^{-2}. The relative difference of the pipeline set $\mathbb{S}_{\cup}$ is higher than others because it involves more candidates and causes more collision of transmissions in NS2 simulation, which is not captured by the Markov model.

Considering the performance change where the number of nodes in grid topologies increase from 10×10 to 20×20, the network density is constant but the network diameter increases. The long topology is almost double the number of transmissions since the network diameter is double. However, the number of time slots sluggishly increase from the short topology to the long topology since the long topology increases the possibility of simultaneous transmissions and pipelined transmissions only consume one time slot. The increment of time slots in opportunistic routing is more obvious. This is because that few simultaneous transmissions occur in opportunistic routing. Since the density of this grid topology is constant, we can predict that the number of time slots of opportunistic routing will be larger than that of traditional routing if the network diameter keeps increasing.

3.4.4 Simulation results for random topologies

We analyze the performance of opportunistic routing in random topologies. In Figure 3.8, there are 30 nodes randomly located in a square. The top graph show the expected number of transmissions and time slots. As a first observation, opportunistic routing always has smaller number of transmissions than traditional routing. Two different pipeline sets of opportunistic routing has almost the same number of packet transmissions. In the analytical models, the total number of transmissions can not be directly affected by whether the pipelined transmission is enable or not. The following three graphs show how many pipelined transmissions occur for traditional routing and opportunistic routing. Opportunistic routing requires a batch of packets transmitting together which limits the simulation of pipelined transmissions. Therefore, in most of case, only one transmission happens in one time slot. Obviously, opportunistic routing with $\mathbb{S}_{\cap}$ is even worse than opportunistic routing with $\mathbb{S}_{\cup}$ since the pipeline set $\mathbb{S}_{\cap}$ further limits the opportunity of pipelined transmissions.

Figure 3.9 show the expected number of transmissions and time slots for other two random topologies. In a constant square area, the performance of opportunistic routing and traditional routing will be slightly improved by increasing the number of nodes in the topology. Since it potential increase the probability of finding better candidates. Figure 3.10 show the difference between analytical models and simulation results. The difference between them is also very small and proves the precision of our models. In NS-2 simulation, the $\mathbb{S}_{\cup}$ has more collision and requires some retransmissions which causes the difference is larger than another case.

Spatial channel reuse problem in opportunistic routing:

In all simulation results and analytical results, opportunistic routing has better performance than traditional routing in terms of the expected number of transmissions. Opportunistic routing outperforms traditional routing under study by delivering packets on low quality links, but when the link qualities are high, traditional routing can have a smaller number of time slots. The reason is that opportunistic routing will sacrifice from spatial channel reuse inherently. Recall that opportunistic routing imposes a coordination mechanism to obtain opportunistic gains (e.g., reduce the number of transmissions). Each node transmits packets in order. A node transmits a batch of packets to its downstream nodes, each downstream node will receive a 'fragment' of the batch due to instantaneous link quality. After the batch of packet is transmitted, the next high-priority node can transmit the received fragment of the batch. When the link qualities are high, the fragments are almost the entire batch, and the next highest-priority would forward its fragment/batch further down. The distribution of packets is limited and will be concentrated to a group of adjacent nodes. When the link qualities are low, packets in a batch have a high probability to be distributed in the long area where nodes many simultaneous transmissions can occur. Therefore, the coordination mechanism of opportunistic routing potentially prevents nodes from exploiting spatial channel in a network with stable links. Taking an extreme case as example, consider the scenario where link success rates are 100% for all links. Traditional routing and opportunistic routing will use the same nodes as forwarders. In opportunistic routing, after a node sends a batch of packets, only the highest downstream node will forward the whole batch of packets since the highest downstream node can receive all of them. Because the packets of a batch can not be separated to nodes whose distance are far away than interference ranges, opportunistic routing has no

simultaneous transmission. Therefore, it eventually requires more time slots to transmit a batch of packets from the source to the destination. We can observe that increasing the number of nodes or batch sizes will make the spatial channel reuse problem more serious.

3.5 Summary

Opportunistic routing has a great impact on network performance as it can increase reliability of wireless networks and reduce the amount of consumed resources. However, most existing research focuses on analyzing the transmission of one packet in opportunistic routing, which ignores pipelined data transfer.

In this chapter, we proposed Markov models that enable the evaluation of opportunistic routing for different wireless networks in terms of the expected number of transmissions and time slots. The network states of packet advancement are mapped by a Markov chain and the states in the Markov chain denote all valid progression of opportunistic routing. The state transition of a Markov chain represents packet transmissions in the wireless mesh networks. When only one packet is transmitted in each state, the total number of state transitions estimates the expected number of packet transmissions. When all possible simultaneous transmissions are considered in each state, the total number of state transitions estimates the expected number of time slots. Our models take into account the theory of the absorbing matrix and enable the estimation of the number of state transitions in a very effective way. Iterative multiplication replaces inversion of matrix and provides accurate estimation results. Resorting to random walk, the time complexity of estimation can be further reduced to $O(\mathbb{N})$. Given the link quality of wireless mesh networks and the coordination mechanism, our models are independent from network

topologies and candidate selection algorithms. The only inputs in our models are the number of nodes in the network, batch sizes and pipeline sets. Therefore, our analytical models are valid for different type of network scenarios and for different batch sizes of packets. Furthermore, our analytical models consider pipelining, and it does not only present the advantage of opportunistic routing but also explores the spatial reuse problem. Considering different coordination mechanisms, the proposed models can simulate different network transmission process in wireless mesh networks including both opportunistic routing and traditional IP forwarding.

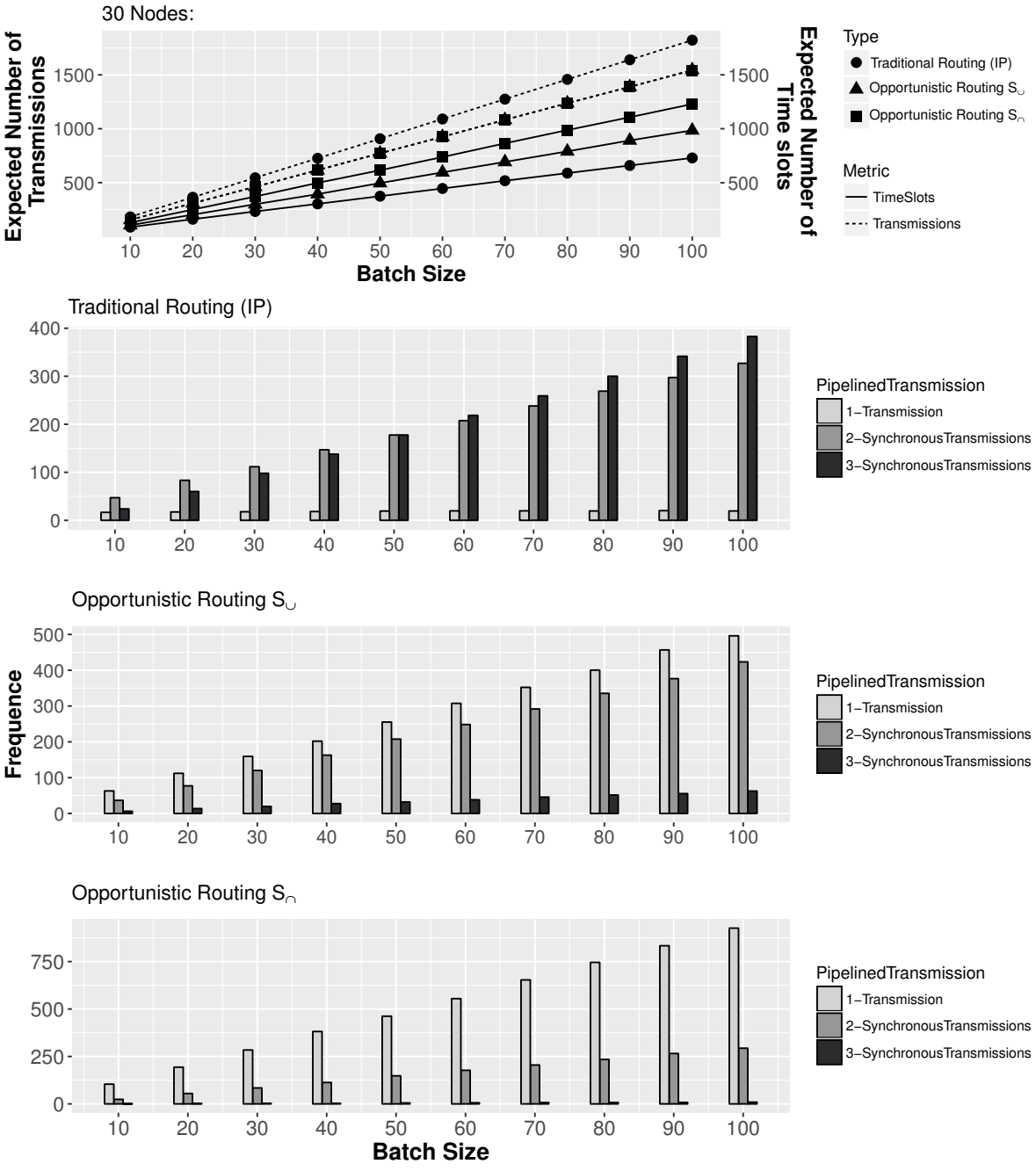

Figure 3.8: The expected number of transmissions and time slots for the random topology with 30 nodes.

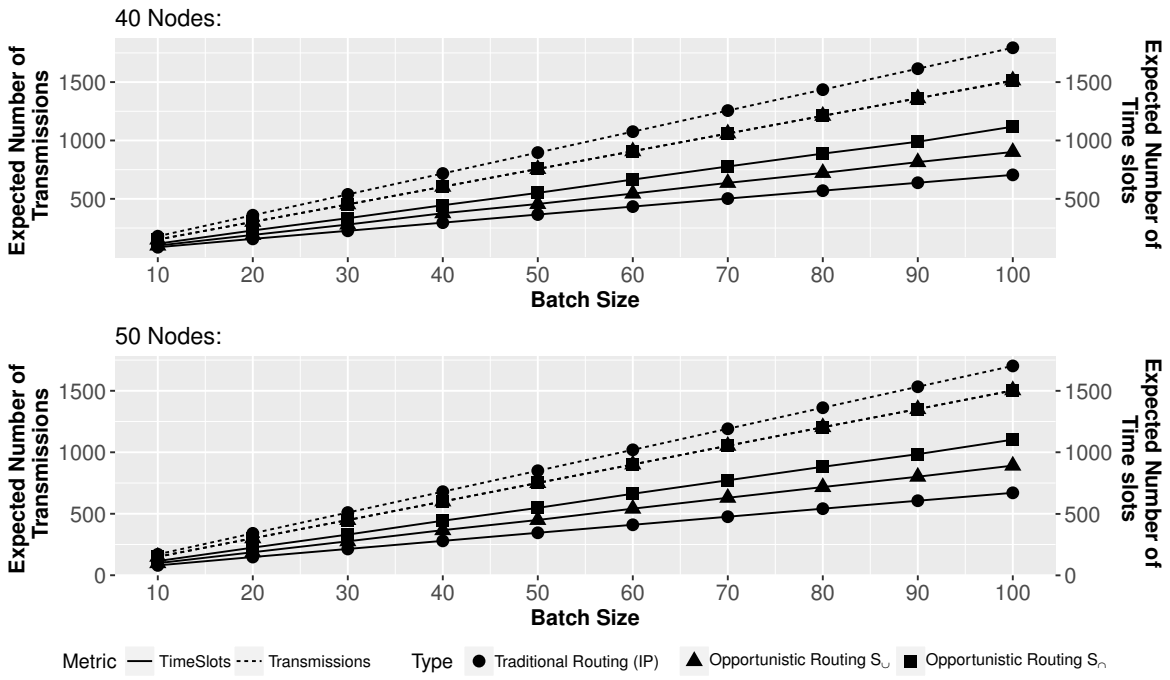

Figure 3.9: The expected number of transmissions and time slots for the random topologies with 40 and 50 nodes.

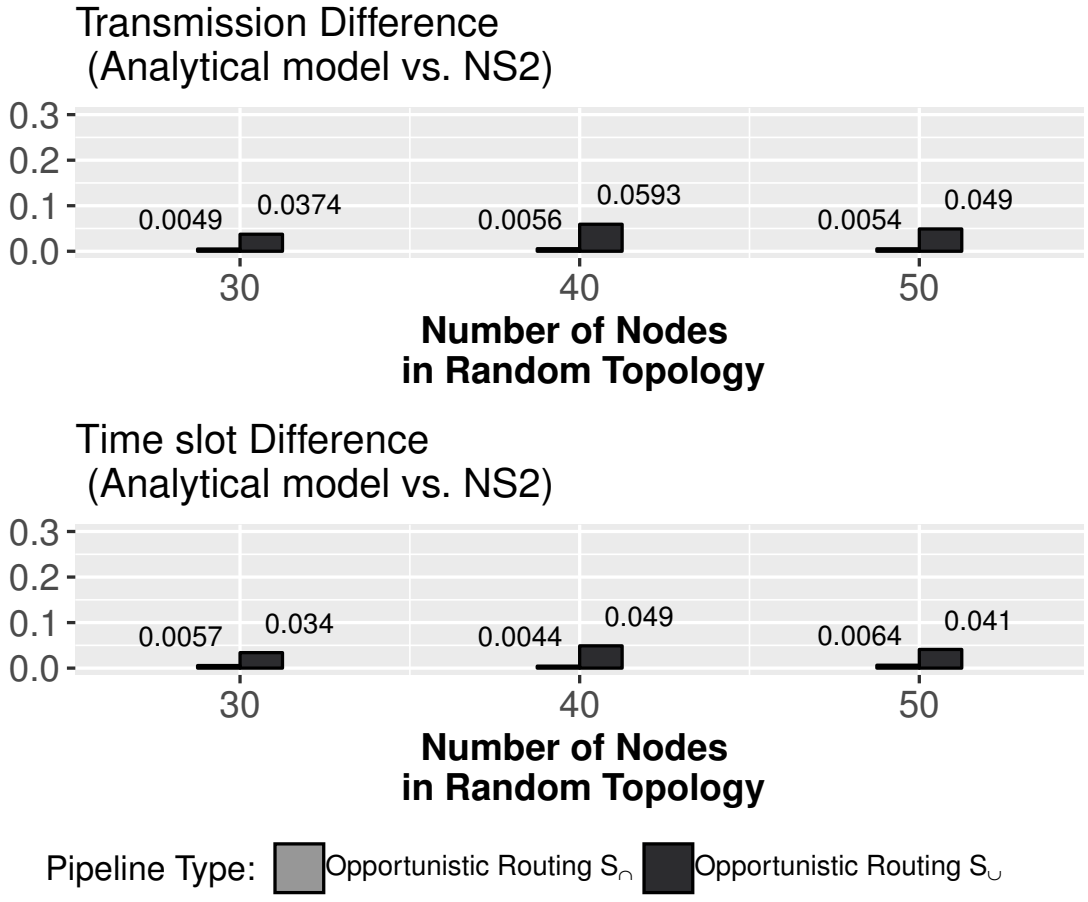

Figure 3.10: The difference between analytical models and NS2 in different topologies.

Chapter 4

ExOR Compact

4.1 Overview of ExOR Compact

Opportunistic routing can provide a significant throughput gain compared to traditional routing. The important reason is that each transmission may have more independent chances of being received and forwarded, it takes advantage of transmissions that reach unexpectedly far, or fall unexpectedly short. Multiple downstream nodes in addition to the matched receiver in the routing module will receive packets and be potential forwarders. After the sender forwards packets, One of these potential forwarders, which is closest to the destination, will be selected to be the next-hop forwarder. Such on-the-fly selection of the next-hop is the fundamental principle of opportunistic routing. Since multiple downstream nodes are potential next-hop forwarders, opportunistic routing can effectively reduce the possibilities of unnecessarily reconstruction path or retransmission due to link breakage on a pre-selected path. However, since more than one node are potential forwarders, an effective coordination mechanism is required to find this closest

node, avoid other nodes transmitting duplicate packets and also guarantee at least one node forwarding packets to downstream nodes. Such reliable coordination mechanism is often complex and different to carry out accurately, which limits the performance and the use of opportunistic routing protocols.

The current opportunistic routing protocol, ExOR, proposes a batch-based coordination. It creates a transmission schedule, which goes in rounds. In each round, each forwarder has one chance to forward the received packets. Forwarders transmit packets in order, and only one forwarder is allowed to transmit at any given time. The others listen to learn which packets were overheard by each node. The high-priority nodes forward packets first, and the low-priority nodes send packets that are not received by higher priority ones. This coordination among these multiple forwarders is a crucial issue in order for its overhead not to overwhelm its potential gains. The main overhead is that the transmission delay of forwarders are waitting to transmit packets. If two packets are far away and their corresponding forwarders can simultaneously transmit them without interference, the coordination of opportunistic routing in this situation will prevents the forwarders from exploiting spatial channel reuse. Therefore, opportunistic routing still has room for improvement about its coordination for better spatial channel reuse.

ExOR Compact, as with ExOR and MORE, uses the ETX metric [23] to reflect loss rate. Each node periodically measures link qualities in terms of ETX. Based on this information, the source node determines a set of forwarders for a route to the destination, and sends data packets with the forwarder list embedded in its packet header. As the most important difference from ExOR, our proposal lets each forwarder create its own *regional* transmission schedule based on the forwarder list received from the source node. That is, the schedule is confined to a subset of the forwarders that are within range of

the transmitting node so that the waiting time is only a function of this range rather than of the entire forwarder list. As such, it facilitates a data transfer pipeline for higher throughput than ExOR. To moderate the effect brought by the denser network activities as above, ExOR Compact uses random linear network coding to mix information among packets so that downstream nodes are less sensitive about receive particular packets. Furthermore, thanks to the deferring of packet forwarding at lower-priority nodes, ExOR Compact tends to capture more coding opportunities than MORE.

4.1.1 Design challenges

To realize the general design of ExOR Compact, we must address the following challenges.

- To increase the spatial channel reuse for a deeper pipeline, a forwarder should coordinate a priority scheduling only among the downstream forwarders within its transmission range. How exactly are downstream forwarders aware of this range and which other forwarders are part of this coordinated effort?
- When we have a busy opportunistic forwarding pipeline, we face a higher risk of lower-priority nodes ending up transmitting before higher-priority nodes by mistake. This can be especially problematic in that our design leaves the link layer independent in medium access control. If such reverse-ordered transmissions happen, how to we handle them?
- Recall that ExOR's scheduling is centered around the crucial feedback mechanism of batch map. It would, however, be difficult to map the reception of coded packets by downstream forwarders because representing the decoding matrix of a forwarder

takes up much packet header space. How do we devise an efficient way to summarize the received coded packets and feed it back to upstream forwarders?

These challenges are addressed in the rest of this chapter.

4.2 ExOR Compact algorithm

4.2.1 Regional forwarding schedule

ExOR Compact is inline with the prioritization design in the original ExOR. The forwarders for a source-destination pair coordinate among themselves to decide in what order they should forward packets and which packets. The network runs a link-state routing with ETX as the link metric. When a source R node has data to send to a destination D, it composes a forwarder list, which is a subset of nodes that have a lower ETX value towards D. This subset should contain nodes on the shortest path plus some redundant nodes near the path. Figure 4.1 shows an example, where S decides that its forwarding list has 14 nodes (including S and D). The idea of forwarding list is for a lower-priority forwarder to defer to higher-priority forwarders' transmission; and only transmits when it knows the higher-priority forwarders have missed some packets.

For the convenience of subsequent description, we provide some notations. Given a source node S that intends to send data to destination D, the routing module of S can decide a forwarder list $\langle F_1, F_2, \ldots, F_S \rangle$, where $F_1 = S$ and $F_{14} = D$. For a given forwarder F_i on this list, we define its *upstream neighbors* as the subset of the upstream forwarders that can transmit to F_i directly, denoted $F_i \uparrow$. For example in Figure 4.1, $F_5 \uparrow$ can be $\{F_1, F_2, F_3, F_4\}$, $F_6 \uparrow$ can be $\{F_2, F_3, F_4, F_5, \}$, $F_7 \uparrow$ can be $\{F_3, F_4, F_5, F_6\}$, $F_8 \uparrow$ can be

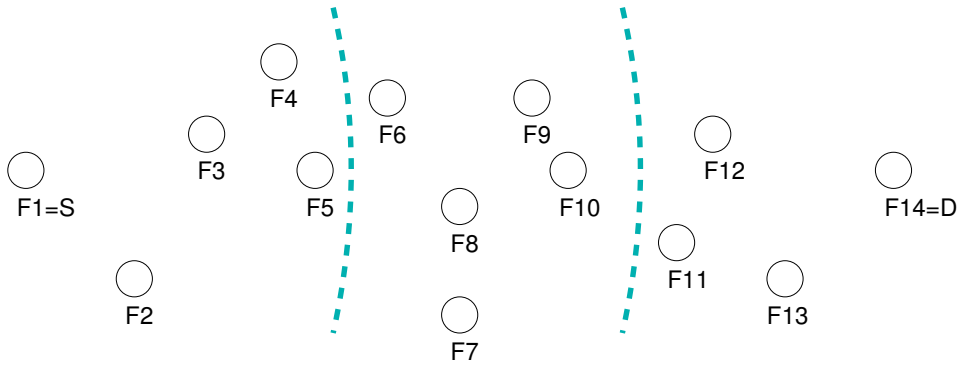

Figure 4.1: Example of regional forwarding schedule

$\{F_4, F_5, F_6, F_7\}$, $F_9 \uparrow$ can be $\{F_5, F_6, F_7, F_8\}$ and $F_{10} \uparrow = \{F_5, F_6, F_7, F_8, F_9\}$ Similarly, we define the *downstream neighbors* of F_i as the subset of the downstream forwarders that F_i can transmit to directly, denoted $F_i \downarrow$. In the figure, $F_1 \downarrow = \{F_2, F_3, F_4, F_5\}$,$F_2 \downarrow = \{F_3, F_4, F_5, F_6\}$,$F_4 \downarrow = F_5, F_6, F_7, F_8, F_9\}$ and $F_5 \downarrow = \{F_6, F_7, F_8, F_9, F_{10}\}$

All forwarders (include S and D) transition among three states, i.e. idle, receiving and sending in Figure 4.2. A forwarder F_i is initially idle until it receives a packet, starting processing its fragment of packets. At this point, it sets a timer τ using the following rules.

- If this packet is from a downstream forwarder, it does nothing.
- If this packet is from an upstream forwarder, say F_a $(1 \leq a < i)$, it needs to wait for nodes in a subset of nodes in its upstream and downstream neighborhoods to transmit before itself. Thus, it sets

$$\tau = (|F_a \downarrow - F_i \uparrow - F_i| + |F_i \uparrow \cap F_a \uparrow|) \times T_c \tag{4.1}$$

  where T_c is set to the time for F_i to transmit three data packets. The timer starts counting after it assumes F_a has completed its fragment. Furthermore, it maintains

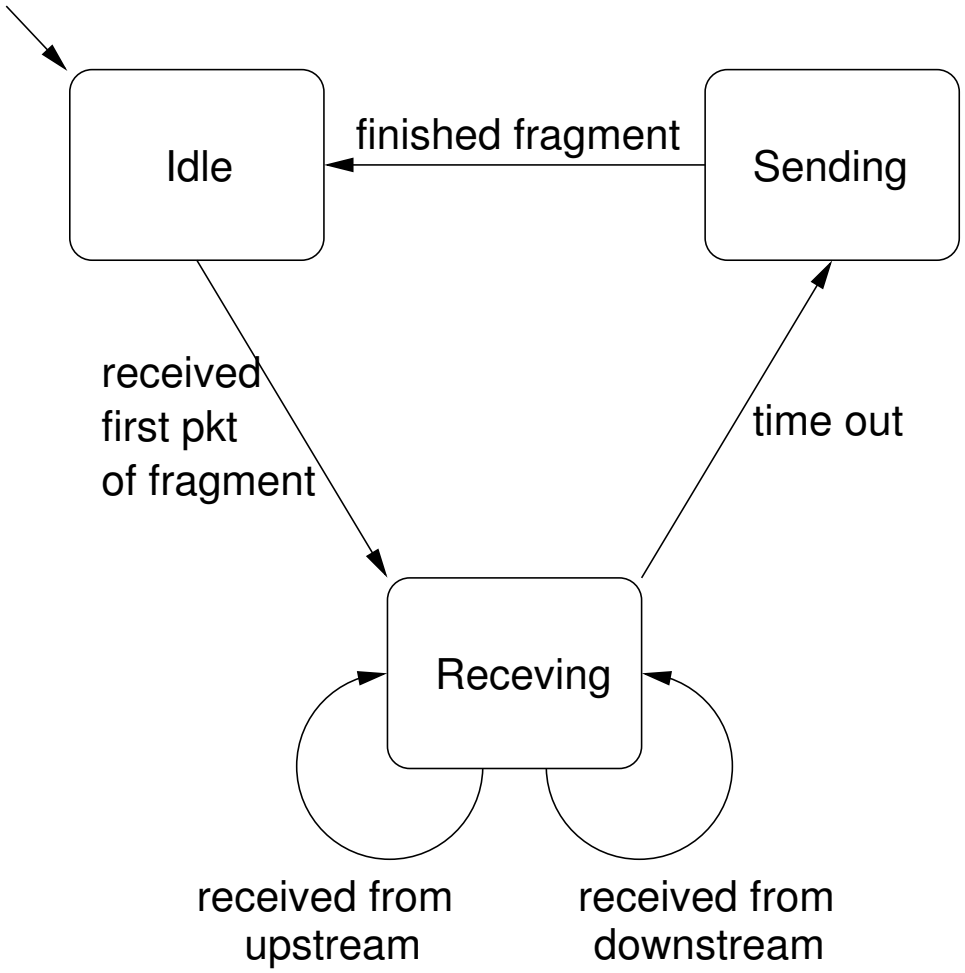

Figure 4.2: State transition of a forwarder

a set $\mathcal{F}_i$ of nodes from which it has received from. At this point, $\mathcal{F}_i = \{F_a\}$ Also, we call F_a the *fragment tail*, which the lowest-priority node F_i has received so far, and will be updated as it receives more packets.

Since this point, F_i may receive more packets from either direction. It updates τ using the following rules.

- If the packet is from a downstream node, denoted F_b, we update $\mathcal{F}_i$ by removing any forwarder with priority higher than or equal to F_b. Then, we update τ based

on 4.2.

- If the packet is from an upstream node, we denote it F_a. If F_a has a higher priority than the current fragment tail, it does nothing. Otherwise, we add F_a to $\mathcal{F}_i$. In addition, we update τ to

$$\left(\left| \bigcup_{F \in \mathcal{F}_i} (F\downarrow - F_i\uparrow - F_i) \right| + |F_i\uparrow \cap F_a\uparrow| \right) \times T_c, \tag{4.2}$$

  and set F_a as the new fragment tail. After F_i sends packets, the fragment tail will be reset. (Here, 4.1 is a special case of 4.2)

Consider an example in Figure 4.1. After the source node F_1 forwards packets, F_2, F_3, F_4 and F_5 could receive packets with different possibilities. If F_2 receives packets, it sets $\tau_2 = (0+3) \times T_c$ based on (1). Similarity, $\tau_3 = 2 \times T_c, \tau_4 = 1 \times T_c, \tau_5 = 0 \times T_c$. Then, F_5 could forward packets first. After F_5 forwards packets, nodes in $F_5\downarrow$ could receives packets opportunistically. If some of them receives packets, the timer is like: $\tau_{10} = 0 \times T_c, \tau_9 = (1+0) \times T_c, \tau_8 = (2+1) \times T_c, \tau_7 = (3+2) \times T_c, \tau_6 = (4+3) \times T_c$. At the same time, nodes in $F_5\uparrow$ could overhear packets from F_5, the timers for its upstream forwarders will be updated. Based on (2), the timers are updated to : $\tau_4 = 0, \tau_3 = 1, \tau_2 = 2, \tau_1 = 3, \tau_0 = 4$. Then, F_4 and F_{10} could forward packets synchronous since they are out of transmission range.

Our regional schedule not only improves the spatial reuse, but also potential increases the coding opportunities at intermediate nodes, because each node could forward packets after receiving packets from all upstream forwarders.

4.2.2 Schedule remedy mechanism

The regional forwarding schedule could allow nodes, which are out of transmission range, to forward packets at the same time. It also incurs many contentions in the MAC layer since more forwarders evolve in data transmissions at the same time. And our opportunistic forwarding protocol is the MAC-layer-independent. Once packets are pushed to the MAC layer, they are out of control by the opportunistic forwarding module. Unfortunately, those packets could be delayed by the MAC layer collisions and it causes lower-priority nodes ending up transmitting before higher-priority nodes by mistake. Furthermore, the delayed packets in the MAC layer of higher-priority nodes could not carry information of new arriving packets from lower-priority nodes, those delayed packets could activate lower-priority nodes to retransmit its previous fragment.

So we proposed the First Packet Detection mechanism. When a routing module is ready to send a fragment of packets, it only pushes the first packet in the fragment to the MAC layer, after the MAC layer forwards this packet, the MAC layer will give a feedback to the routing module, then the routing module will push the remaining packets to the MAC layer. Even though the lower priority nodes send packets first, the higher priority nodes could update the remaining packets in routing module and give lower priority nodes correct feedback.

4.2.3 Feedback on received coded packets

The random linear network coding could avoid duplicate transmissions, but it is difficult to summarize the received packets and feedback it to upstream forwarders. ExOR uses the batch map to describe which packets are received by higher-priority nodes. In

ExOR Compact, we use the number of innovative packets (the degree of freedom of decoding metric) as an indicator. Each forwarder would overhear a packet from both higher-priority nodes and lower-priority nodes, and keep it in its receiving queue if it is innovative for current receiving queue. When a forwarder is active to deliver packets, it only checks how many innovative packets are not received by high priority nodes, then re-coded all packets in its receiving queue and forwards corresponding number of unreceived innovative packets. To recover the packet loss by the unreliable link quality, we use the credit mechanism of MORE [18] to calculate how many coded packets a forwarder should transmit for delivering one innovative packet to one of the downstream nodes.

4.3 Performance evaluation

To evaluate ExOR Compact experimentally, we use the Network Simulator 2 [1] to compare the performance of ExOR Compact to traditional IP forwarding and ExOR.

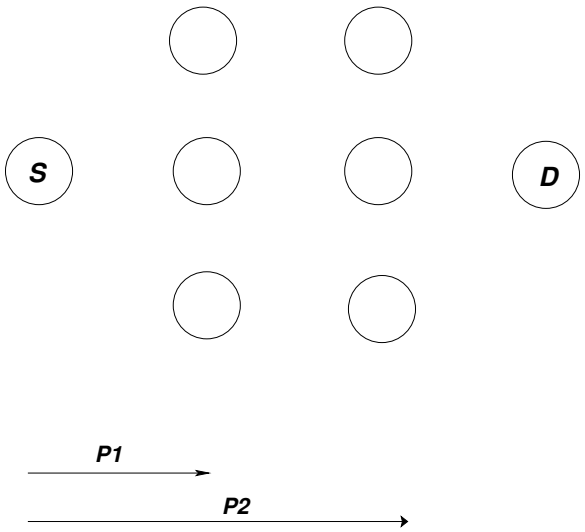

Figure 4.3: short-distance belt topology

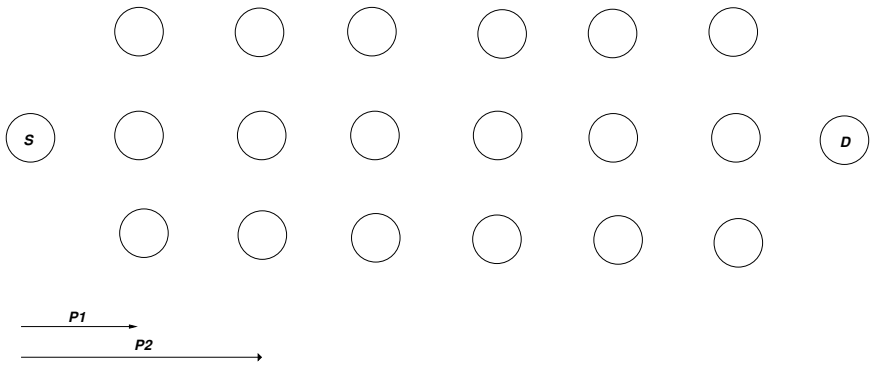

Figure 4.4: long-distance belt topology.

4.3.1 Experiment settings

The link layer of the simulated wireless nodes is the standard 802.11 DCF, where we use the reliable unicast (with up to 7 retransmissions) for IP forwarding and the unprotected broadcast for ExOR and ExOR Compact. The data rate is the base 1Mbps; the relative performance of all tested protocols will remain similar for higher data rates. The propagation model is the ns2 built-in two-ray ground model, which translates to a transmission range of approximately 250 meters.

We test the protocols in two topologies, depicted in Figures 4.3 and 4.4. The first scenario, called *short topology*, has 8 nodes deployed in 4 stages. The source and destination nodes are at the 1st and 4th stages, respectively, and three more nodes are at stages 2 and 3. The second scenario, called *long topology*, has 20 nodes deployed in 8 stages, where we have six intermediate stages of 3 nodes each. In either case, the inter-stage distance is 100 meters, and the intra-stage distance is 100 meters.

To simulate link quality variation, we added a link reliability probability p to the MAC receiver code, so that a packet would dropped with a probability of $1 - p$ even if it is

received by the PHY correctly. The simulation is conducted for various p values for the three protocols. Specifically, we have two such probabilities in each simulation scenario. When two nodes are in the same stage or 1 stage apart, it is p_1, and when they are two stages apart, it is p_2. Their values are summarized in Table 4.1.

p_1	100%	80%	60%	40%	20%
p_2	50%	40%	30%	20%	10%

Table 4.1: Link reliability

In each simulation, we run one of the tested protocols (IP forwarding, ExOR, or ExOR Compact) with one column of link parameters in Table 4.1. The source node needs to send 1MB of data to the destination. It divides the data into packets of 1024 bytes each. The data packets are injected as a CBR User Datagram Protocol flow at the rate of 20 packets per second.

- In IP forwarding, the route is selected with the minimum end-to-end ETX metric [23]. In order for the destination to receive 1000 packets in total, the source may need to inject more packets for scenarios with less reliable links.
- In ExOR's design, each batch only forwards 90% of its data packets opportunistically, with the remaining 10% forwarded by traditional IP forwarding. In our simulation, and as in the original ExOR [10], the source sends 1.1MB of data in 10 batches of 110 packets each, and a batch is completed as long as 100 packets are delivered to the destination opportunistically. (This change would give ExOR a performance advantage apparently because the 10% data transported by IP would

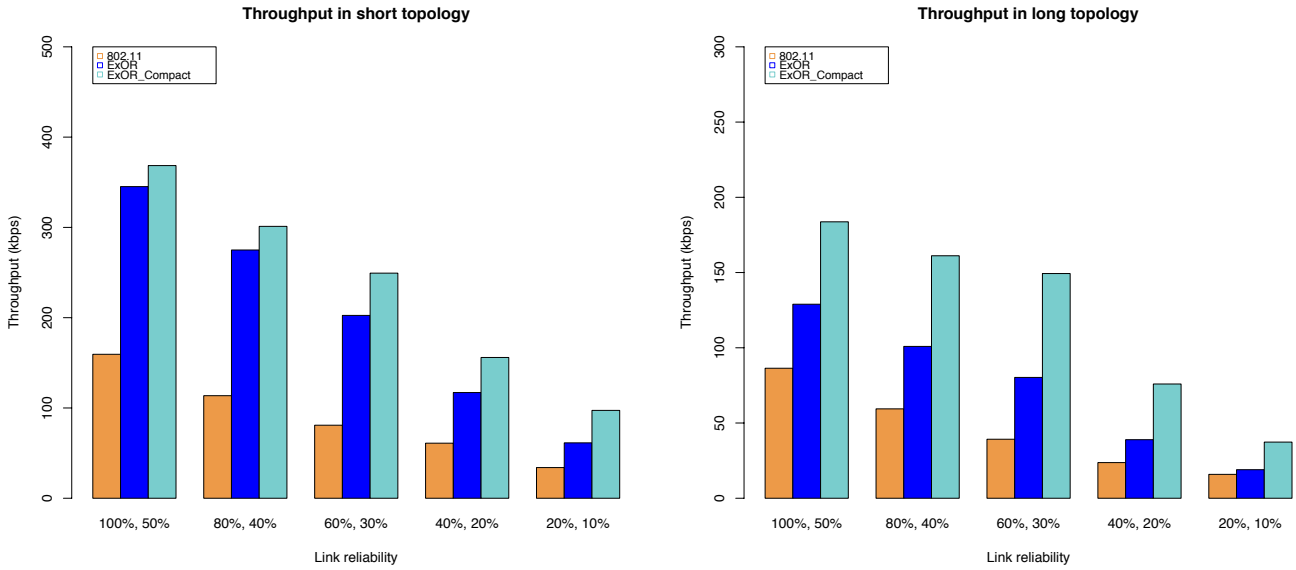

Figure 4.5: Data throughput

incur a fairly large overhead.)

- Because ExOR Compact provides a reliable data forwarding service, the source node's application layer agent only needs to push down the 1000 data packets. The network layer mixes the packets with network coding and forwards them opportunistically. The destination is ensured to receive and decode them for its application layer.

4.3.2 Results

To compare ExOR Compact to IP forwarding and the original ExOR, we measure their data transportation capacity, overhead in terms of transmissions at the link layer, and the end-to-end delay at the application layer.

The data transfer capacity is measured by the inverse of the amount time used to deliver 1MB of application-layer data. We tested the three protocols under the link reliability probabilities in Table 4.1 in both short and long topologies, where each scenario is repeated 10 times. The average throughput and standard deviation are plotted in Figure 4.5 for the short topology (left) and right topology (right). Generally, the short topology allows all protocols to have a higher throughput because of the lesser effect of intra-flow interference. For both throughput plots, the data transfer capacities decrease with an increasing link error. In each particular scenario of comparison among the three protocols, ExOR Compact offers the highest throughput, while IP forwarding has the lowest. Furthermore, in the long topology, ExOR Compact's explicit exploitation of spatial channel reuse gives it a greater margin over the original ExOR.

Next, we show the overhead for these protocols to achieve the data transfer throughput in terms of the total number of transmissions at the link layer as in Figure 4.6. Here again, each data point is 10 repeats of the same scenario. We can observe that the long topology entails a greater amount of efforts for each protocol to fulfill the 1MB of data transportation requirement. Unsurprisingly, the more reliable the links are, the smaller the total number of transmissions. In addition, the more efficient spatial channel reuse of ExOR Compact in the long topology allows it to further save the work at the link layer. Note that the distributions of the number of transmissions among nodes are different

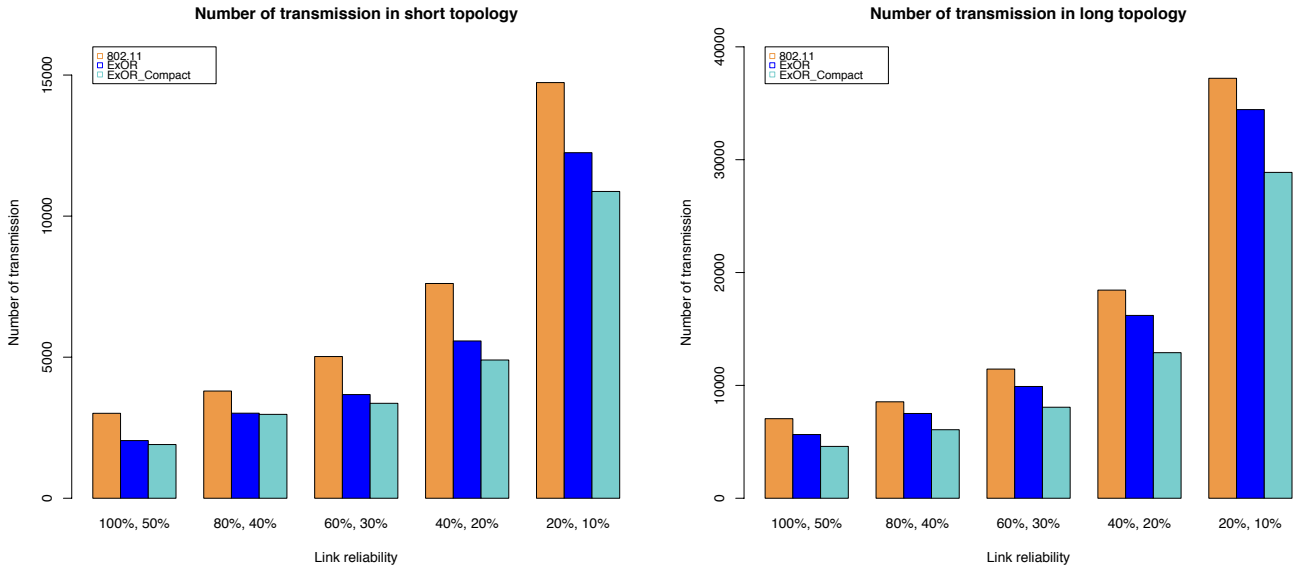

Figure 4.6: Number of transmissions

across the three protocols. That is, IP forwarding would concentrate much of its work around the source node as a fraction of packets are lost every hop along the path to the destination, especially for unreliable links. In contrast, both ExOR and ExOR Compact have an even division of labour. As we are curious about how many data packets are injected by the source nodes, we summarize the measurements in Table 4.2 for different link reliability settings in both topologies.

	100%, 50%	80%, 40%	60%, 30%	40%, 20%	20%, 10%
short	1028	1320	1820	2441	4160
long	1621	2140	3132	4640	10410

Table 4.2: Total number of packets injected

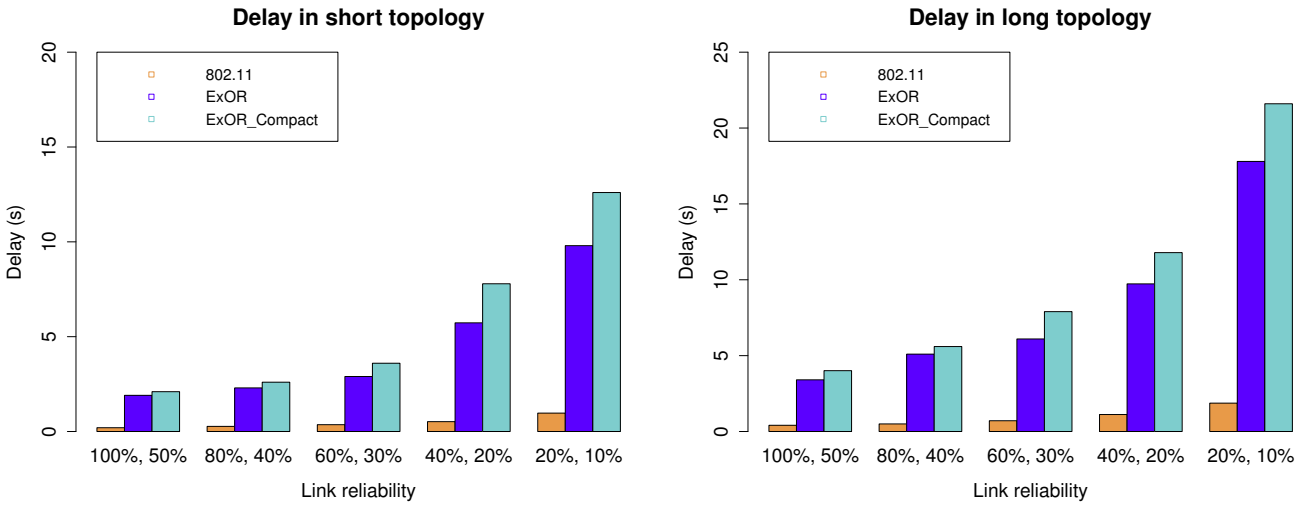

Figure 4.7: End-to-end delay

Using a tighter spatial channel reuse and network coding, ExOR Compact offers a high-throughput, reliable data transfer service with fewer transmissions. The cost is that the majority of the packets of a batch can only be decided when the entire batch has been received by the destination. We are interested in this inevitable extra delay, and plotted the measurement on all protocols. Figure 4.7 depicts the delays in the short and long topologies. Note that, although traditional IP forward may have a very low delay, it has a very large number of dropped packets when the links are unreliable. Such dropped packets effectively have an infinite delay which are not included in the figures.

4.4 Summary

Opportunistic routing has shown great improvement compared to traditional routing. It takes advantage of the broadcast nature of wireless media and utilizes multiple downstream nodes to forward data packets. It transmits each packet fewer times than traditional routing, which should cause less interference.

The key challenge in opportunistic routing is ensuring that only the best receiver of each packet forwards it. In order to avoid duplication, ExOR operates on batches in order to reduce the communication cost of agreement. However, it imposes a strict scheduler on forwarders access to the medium. Although the medium access scheduler delivers opportunistic throughput gains, it can not exploit the spatial channel reuse when multiple packets can be simultaneously received by their corresponding receivers.

Our work is motivated by the improving the spatial channel reuse of the timer-based opportunistic routing. We proposed an adaptive forwarding schedule that enables nodes in the forwarder list to transmit packets efficiently. The schedule is confined to a subset of the forwarders that are within range of the transmitting node so that the waiting time is only a function of this range rather than of the entire forwarder list. As such, it facilitates a data transfer pipeline for higher throughput than ExOR. Compared to IP forwarding and ExOR, we can also observe a greater amount of improvement in long topology.

Chapter 5

TCPFender

5.1 Overview of TCPFender

In this chapter, we introduce TCPFender as an adaptation layer above the network layer, which hides opportunistic routing and network coding from the transport layer. The process of TCPFender is shown in Figure 5.1. It confines the modification of the system only under the network layer. The goal of TCPFender is to improve TCP throughput in wireless mesh networks by opportunistic routing and network coding. However, opportunistic routing in wireless networks causes many dropped packets and out-of-order arrivals, and it is difficult for TCP sender to maintain a large congestion window. Especially the underlying link layer is the stock IEEE 802.11, which only provides standard unreliable broadcast or reliable unicast (best effort with a limited number of retransmissions). TCP has its own interpretation of the arrival (or absence) of the ACK segments and their timing. It opens up its congestion window based on continuous ACKs coming in from the destination. The dilemma is that when packets arrive out of order or are

dropped, the TCP receiver cannot signal the sender to proceed with the expected ACK segment. Unfortunately, opportunistic routing can introduce many out-of-order arrivals, which can significantly reduce the congestion window size of regular TCP since it increases the possibility of duplicated ACKs. Furthermore, the long decoding delay for batch-based network coding does not fare well with TCP, because it triggers excessive time-out events.

TCPFender adaptation layer at the receiving side functions over the network layer and provides positive feedback early on when innovative coded packets are received, i.e. suggesting that more information has come through the network despite not being decoded for the time being. This process helps the sender to open its congestion window and trigger fast recovery when the receiving side acknowledges the arrival of packets belonging to a later batch, in which case the sending side will resend dropped packets of the unfinished batch. On the sender side, the ACK signalling module is able to differentiate duplicated ACKs and filter useless ACKs (shown in Figure 5.1).

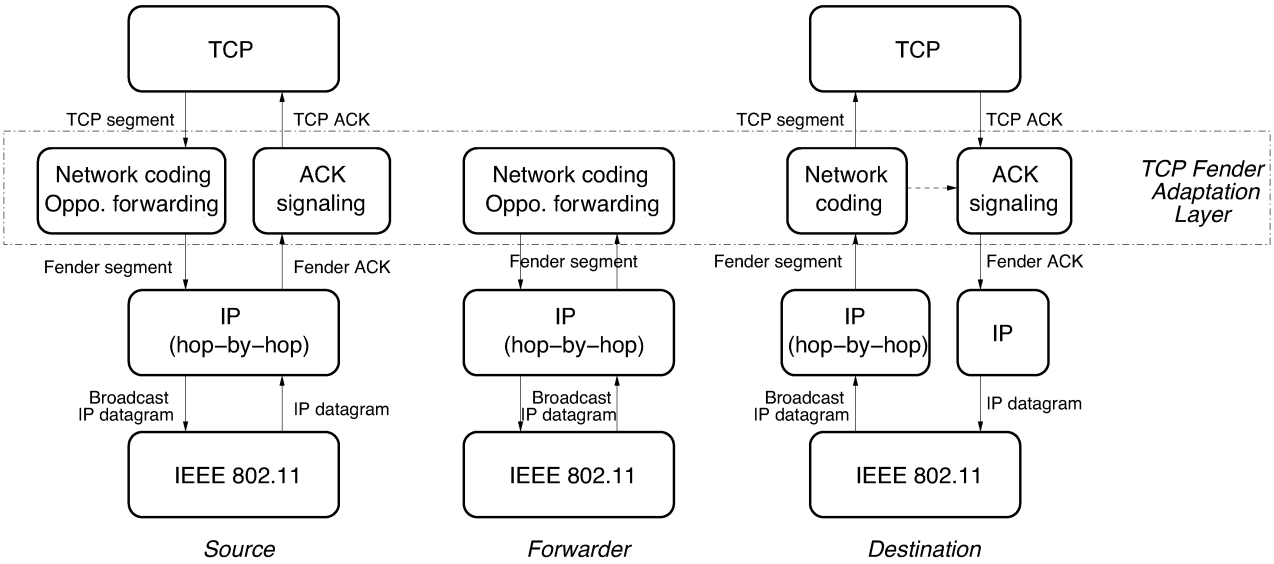

Figure 5.1: TCPFender design scheme.

5.2 TCPFender algorithm

To better support TCP with opportunistic routing and network coding, TCPFender inserts the TCP adaptation layer above the network work layer at the source, the forwarder, and the destination. The main work of the TCP adaptation layer is to interpret observations of the network layer phenomena in a way that is understandable by TCP. The network coding module in the adaptation layer is based on a batch-oriented network coding operation. The original TCP packets are grouped into batches, where all packets in the same batch carry encoding vectors on the same basis. At the intermediate nodes, packets will be recoded and forwarded following the schedule of opportunistic data forwarding proposed by MORE, which proposes a transmission credit system to describe the duplication of packets. This transmission credit system can compensate the packet loss, increase the reliability of the transmission, and represent the schedule of opportunistic data forwarding. The network coding module in the destination node will try to decode received coded packets to original packets when it receives any coded packet. The ACK signalling modules at the source and the destination are responsible for translation between TCP ACKs and TCPFender ACKs.

5.2.1 Network coding in TCPFender

We implement batch-oriented network coding operations at the sender and receiver to support TCP transmissions. All data pushed down by the transport layer in sender are grouped into batches, and each batch has a fixed number β ($\beta = 10$ in our implementation) of packets of equal length (with possible padding). When the source has accumulated packets in a batch, these packets are coded with random linear network

coding, tagged with the encoding vectors, and transmitted to downstream nodes. The downstream nodes are any nodes in the network closer to the destination. Any downstream node can recode and forward packets when it receives a sufficient number of them. We use transmission credit mechanism, as proposed in MORE, to balance the number of packets to be forwarded in intermediate nodes.

We make two important changes to improve the network coding process of MORE for TCP transmissions. For a given batch, the source does not need to wait until the last packet of a batch from the TCP before transmitting coded packets. We call this accumulative coding. That is, if k packets ($k < \beta$) have been sent down by TCP at a point of time, a random linear combination of these k packets is created and transmitted. Initially, the coded packets only include information for the first few TCP data segments of the batch, but will include more towards the end of the batch. The reason for this "early release" behaviour is for the TCP receiving side to be able to provide early feedback for the sender to open up the congestion window. On the other hand, we use a deeper pipelining than MORE where we allow multiple batches to flow in the network at the same time. To do that, the sending side does not need to wait for the batch acknowledgement before proceeding with the next batch. In this case, packets of a batch are labeled with a batch index for differentiation, in order for TCP to have a stable, large congestion window size rather than having to reset it to 1 for each new batch. The cost of such pipelining is that all nodes need to maintain packets for multiple batches.

5.2.2 Source adaptation layer

The source adaptation layer buffers all original packets of a batch that have not been acknowledged. The purpose is that when TCP pushes down a new data packet or previously sent data packet due to a loss event, the source adaptation layer can still mix it with other data packets of the same batch. The ACK signalling module can discern duplicated ACKs which are not in fact caused by the network congestion. Opportunistic data forwarding may cause many extra coded packets, specifically when some network links are of the high quality at a certain point. This causes the destination node to send multiple ACKs with same sequence number. In this case, such duplicated ACKs are not a signal for the network congestion, and should be treated differently by the ACK signalling module in the source. These two cases of duplicated ACKs can actually be differentiated by tagging the ACKs with the associated sequence numbers of the TCP data segment. These ACKs are used by the TCPFender adaptation layer at the source and the destination and should be converted to original TCP ACKs before being delivered to the upper layer.

The flow of data or ACKs transmissions is shown in the left of Figure 5.1. Original TCP data segments are generated and delivered to the module of network coding and opportunistic forwarding. Here, TCP data segments may be distributed to several batches based on their TCP segment sequences, so the retransmitted packets will be always in the same batch as their initial distribution. After the current TCP data segment mixes with packets in a batch, TCPFender data segments will be generated and injected to the network via hop-by-hop IP forwarding, which is essentially broadcasting of IP datagrams. On the ACK signalling module, when it receives TCPFender ACKs, if the ACK's sequence

number is greater than the maximum received ACK sequence number, this ACK will be translated into a TCP ACK and delivered to the TCP sender. Otherwise, the ACK signalling module will check whether this duplicated ACK is caused by opportunistic data forwarding or not. Then it will decide whether to forward a TCP ACK to the TCP or not. The reason for differentiating duplicated ACKs at the source instead of at the destination is to reduce the impact of ACK loss on TCP congestion control.

5.2.3 Destination adaptation layer

The main function of the destination adaptation layer is to generate ACKs and detect congestion in the network. It expects packets in the order of increasing batch index. For example, when it is expecting the bth batch, it implies that it has successfully received packets of the previous $b-1$ batches and delivered them up to the TCP layer. In this case, it is only interested in and buffers packets of the bth batch or later. However, the destination node may receive packets of any batch. Suppose that the destination node is expecting the bth batch, and that the rank of the decoding matrix of this batch is r. In this case, the destination node has "almost" received $\beta \times (b-1) + r$ packets of the TCP flow, where $\beta \times (b-1)$ packets have been decoded and pushed up the TCP receiver, and r packets are still in the decoding matrix. When it receives a coded packet of the b'th batch, if $b' < b$, the packet is discarded. Otherwise, this packet is inserted into the corresponding decoding matrix. Such an insertion can increase r by 1 if $b' = b$ and this received packet is an innovative packet. The received packet is defined as an innovative packet only if the received packet is linearly independent with all the buffered coded packets within the same batch. In either case, it generates an ACK of sequence number $\beta \times (b-1) + r$, which

is sent over IP back to the source node. One exception is that if $r = \beta$ (i.e. decoding matrix become full rank), the ACK sequence number is $\beta \times (\hat{b} - 1) + \hat{r}$, where $\hat{b}$ is the next batch that is not full and $\hat{r}$ is its rank. At this point, the receiver moves on to the $\hat{b}$th batch. This mechanism ensures that the receiver can send multiple duplicate ACKs for the sender to detect congestion and start fast recovery. It also supports multiple-batch transmissions in the network and guarantees the reliable transmission at the end of the transmission of each batch.

The design of the destination adaptation layer is shown on the right of Figure 5.1. The network coding module has two functions. First, it will check whether the received TCPFender data segment is innovative or not. In either case, it will notify the ACK signalling to generate a TCPFender ACK. Second, it will deliver original TCP data segments to TCP layer if one or more original TCP data segment are decoded after receiving an innovative coded data packet. This mechanism can significantly reduce the decoding delay of the batch-based network coding. On the other hand, TCPFender has its own congestion control mechanism, so TCP ACK that is generated by the TCP layer will be dropped by the ACK signalling module at the destination.

5.2.4 Forwarder adaptation layer

The flow of data at forwarders is shown in the middle of Figure 5.1. The ACK is unicast from the destination to the source by IP forwarding, which is standard forwarding mechanism and is not shown in the diagram. The intermediate node receives TCPFender data segment from below and this segment will be distributed into corresponding batches and regenerates a new coded TCPFender data segment. This new TCPFender data

segment will be sent to downstream forwarders via hop-by-hop IP broadcasting based on the credit transmission system proposed by MORE.

5.3 Performance evaluation

In this section, we investigate the performance of TCPFender through computer simulations using NS-2. The topologies of the simulations are made up of three exemplar network topologies and one specific mesh. These topologies are depicted in Figure 5.2 diamond topology, Figure 5.3 string topology, Figure 5.4 grid topology, and Figure 5.5 mesh topology. The packet delivery rates at the physical layer for the mesh topology are marked in Figure 5.5, and the packet delivery rates for other topologies are described in Table. 5.1. The source node and the destination node are at the opposite ends of the network. One FTP application sends long files from the source to the destination. The source node emits packets continuously until the end of the simulation, and each simulation lasts for 100 seconds. All the wireless links have a bandwidth of 1Mbps and the buffer size on the interfaces is set to 100 packets. The source node keeps sending packets for 100 seconds and we calculate the throughput by how many packets received by the destination in 100 seconds. Here, the packet size is 1000 bytes. To compensate for the link loss, we used the hop-to-hop redundancy factor for TCPFender on a lossy link. Recall that the redundancy factor is calculated based on the packet loss rate, which was proposed in MORE [18]. This packet loss rate should incorporate the loss effect at both the Physical and Link layers, which is higher than the marked physical layer loss rates. The redundancy factors of the links are thus set according to these revised rates. We compared our protocol against TCP and TCP+NC in four network topologies. In our simulations,

TCP ran on top of IP, and TCP+NC has batch-based network coding enabled but still over IP. The version of TCP is TCP Reno for TCPFender and both baselines. The ACK packet for the three protocols are routed to the source by the shortest-path routing.

In this section, we examined whether TCPFender can effectively utilize opportunistic forwarding and network coding. TCPFender can provide reliable transmissions in these four topologies and the analysis metrics we took are the network throughput and the end-to-end packet delay at the application layer. We repeated each scenario 10 times with different random seeds for TCPFender, TCP+NC, and TCP/IP, respectively. In TCPFender, every intermediate node has the opportunity to forward coded packets and all nodes operate in the 802.11 broadcast mode. By contrast, for TCP/IP and TCP+NC, we use the unicast model of 802.11 with ARQ and the routing module is the shortest-path routing of ETX [23].

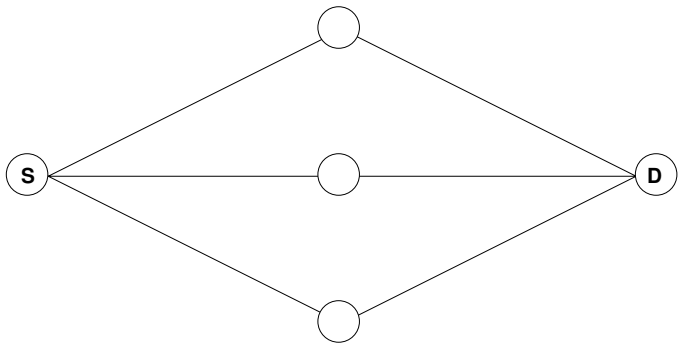

Figure 5.2: Diamond topology

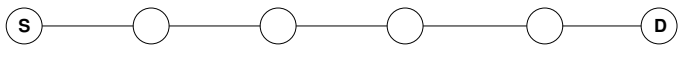

Figure 5.3: String topology

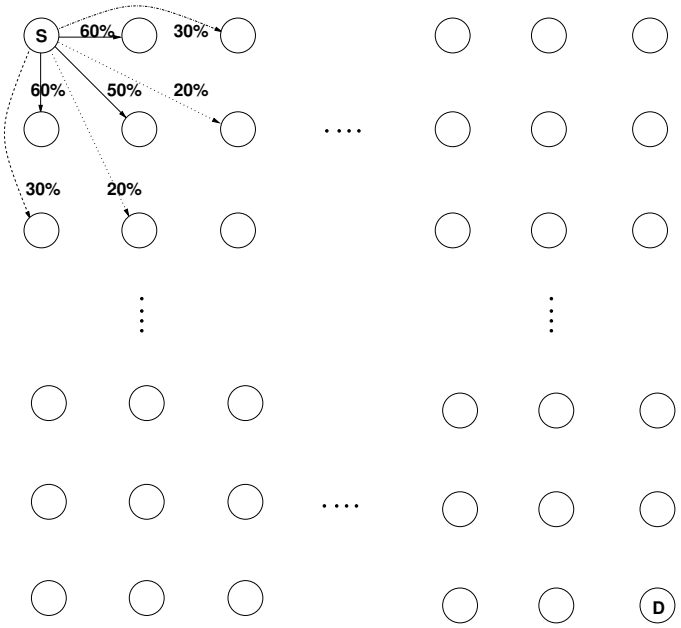

Figure 5.4: Grid topology

In the diamond topology (Figure 5.2), the source node has three different paths to the destination. TCP and TCP+NC only use one path to the destination, but TCPFender could utilize more intermediate forwarders thanks to opportunistic routing. The packet delivery rates for each link are varied between 20%, 40%, 60% and 80%. We plotted the throughput of these three protocols in Figure 5.6. In all cases, the TCPFender has the highest throughput, and the performance gain is more visible for poor link qualities.

Next, we tested these protocols in the string topology (Figure 5.3) with 6 nodes. The distance between the two nodes is 100 meters, and the transmission range is the default 250 meters. Different combinations of packet delivery rates for 100-meter and 200-meter

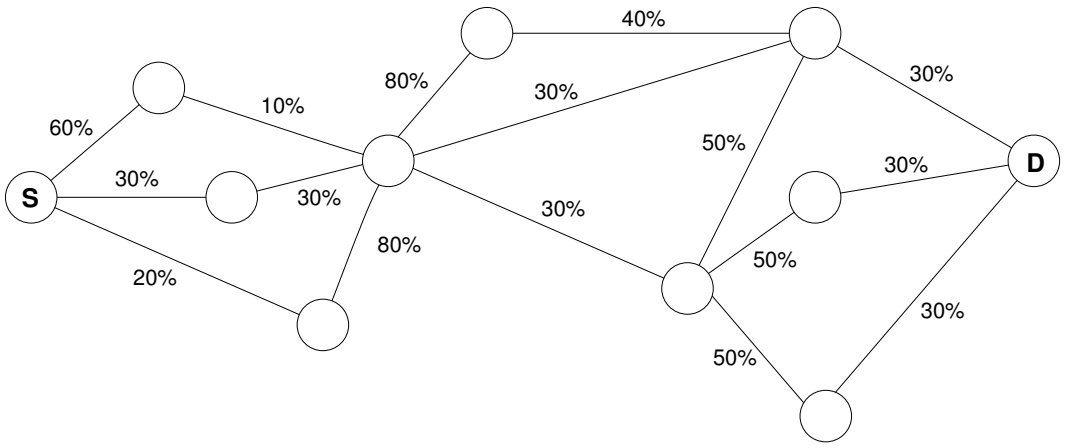

Figure 5.5: Mesh topology

Table 5.1: Packet delivery rate

100m	200 m			
100%	80%	60%	40%	20%
80%		60%	40%	20%
60%			40%	20%
40%				20%

distances are described in Table 5.1. As a result, the shortest path routing used by TCP and TCP+NC can decide to use the 100m or 200m links depending on their relative reliability. The throughputs of the three protocols are plotted in Figure 5.7, where we observed how they perform under different link qualities. Except for the one case where both the 100m and 200m links are very stable (i.e 100% and 80%, respectively), the gains of having network coding and opportunistic forwarding are fairly significant in maintaining TCP's capacity to the application layer. When the links are very stable, the cost of the opportunistic forwarding schedule and the network coding delay will slightly reduce the

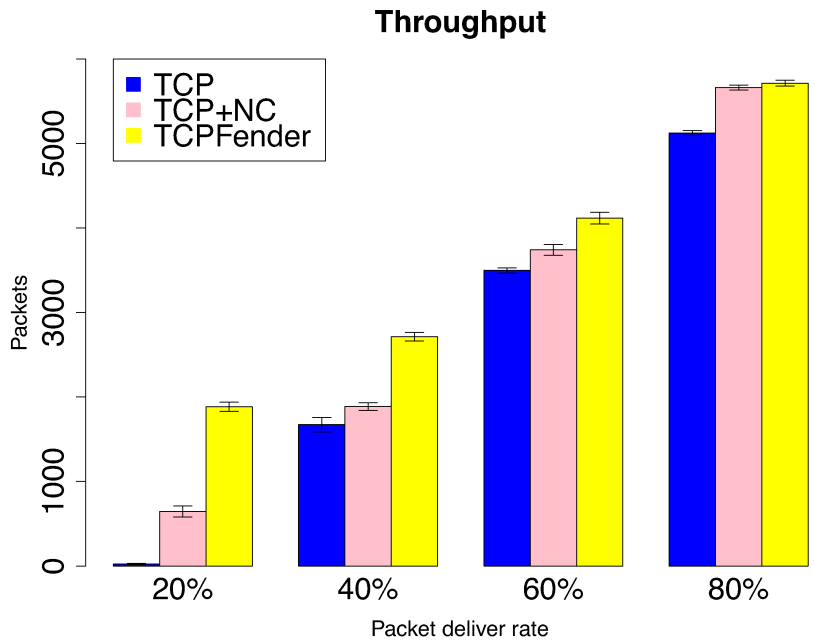

Figure 5.6: Throughput for diamond topology

network throughput.

We also plotted these three protocols' throughputs in a grid topology (Figure 5.4) and a mesh topology (Figure 5.5). Each node has more neighbours in these two topologies, compared to string topology (Figure 5.3), which increases the chance of opportunistic data forwarding. The packet delivery rates are indicated in these two Figures (Figure 5.4 and Figure 5.5). In general, the packet delivery rates drop when the distance between a sender and a receiver increases. In our experiment, the source and destination nodes deploy at the opposite ends of the network. The throughput of TCPFender is depicted in Figure 5.8 and it is much higher than TCP/IP because opportunistic data forwarding and network coding increase the utilization of network capacity. The gain is about 100% in our experiment. The end-to-end delays of the grid topology and the mesh topology are plotted in Figure 5.8. In general, TCP+NC has long end-to-end delays because packets

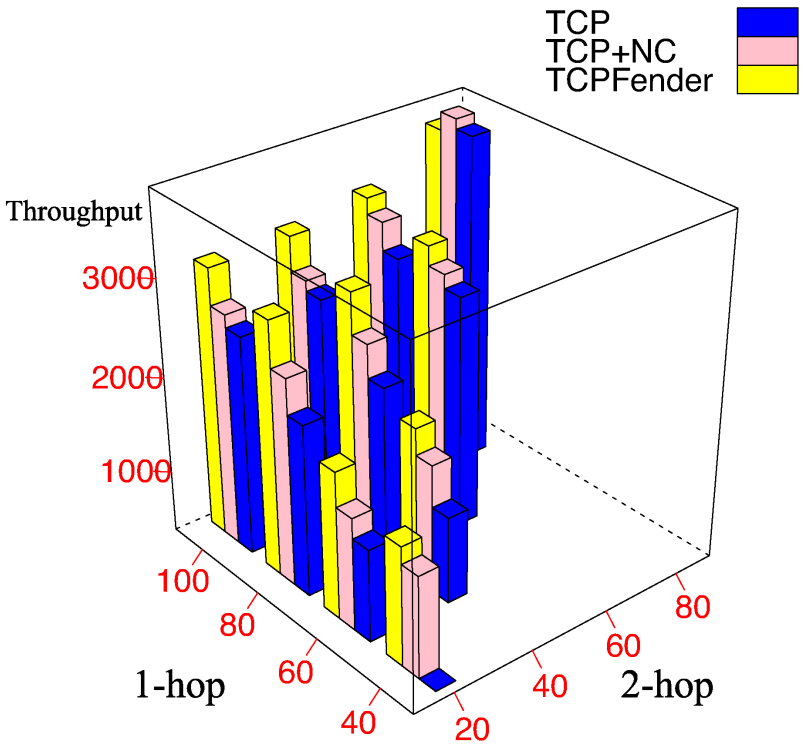

Figure 5.7: Throughput for string topology

need be decoded before being delivered to the application layer; this is an inherent feature of batch-based network coding. TCPFender can benefit from backup paths and receive packets early, so it reduces the time-consumption of waiting for decoding and its end-to-end delay is shorter than TCP+NC.

Next, we are interested in the impact of batch sizes on the throughput and the end-to-end delay. Figure 5.9 shows the throughput of TCPFender in the mesh topology for batch sizes of 10, 20, 30, ..., 100 packets. In general, batch sizes will have an impact on the TCP throughput (as exemplified in Figure 5.11). When the batch size is small ($\leq$ 40), the increment of the batch size can increase the throughput, since it expands

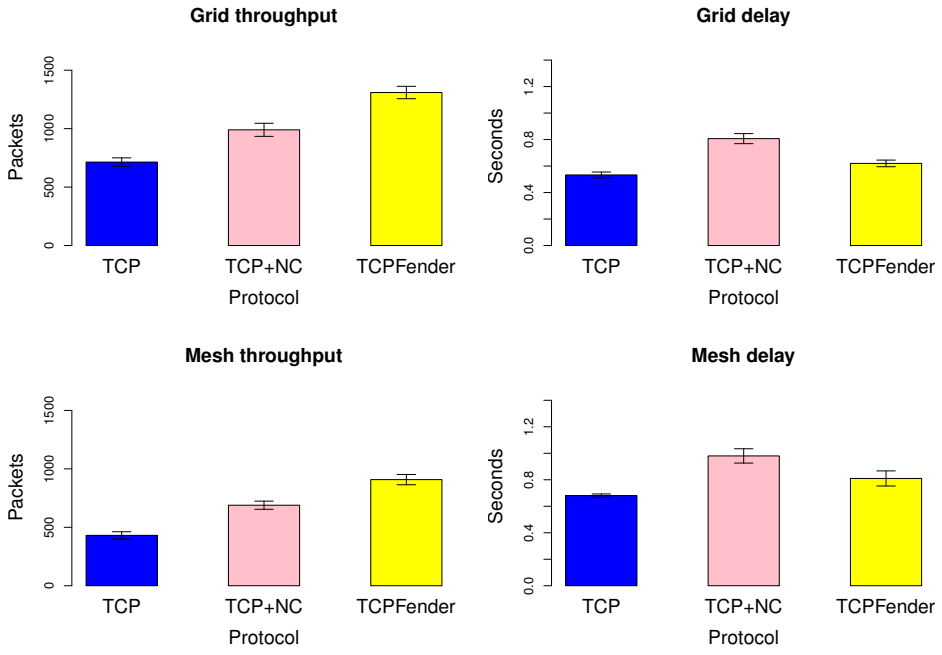

Figure 5.8: Throughput and delay for grid topology and mesh topology.

the congestion window. However, if the batch size is too large (> 40), the increment of the batch size will decrease the throughput because the increase of batch size will amplify the fluctuation of the congestion window and also increase packet overhead by long encoding vectors. Figure 5.11 also describes how many packets are transmitted in the network. Each intermediate node will keep all unfinished batches. From Figure 5.11, since the number of packets transmitted in the network is smaller than two batch sizes, intermediate nodes only need to keep two batches of packets and the memories required to store the packets are acceptable. The nature of batch-based network coding will also introduce decoding delays, so the batch size has a direct impact on the end-to-end delay, as summaries in Figure 5.9. In Figure 5.10, we plotted the end-to-end delays of all packets over time in two sample simulations. Note that these tests were done for files that need many batches to carry. On the other hand, when the file size is comparable to the batch

size, the file-wise delay will be comparable to the decoding delay of an entire batch, which may seem relatively large. However, because the file size is small, this delay is not overly significant as the delay is at the order of its transmission time. Nevertheless, network coding does add a considerable amount of delay in comparison to pure TCP/IP.

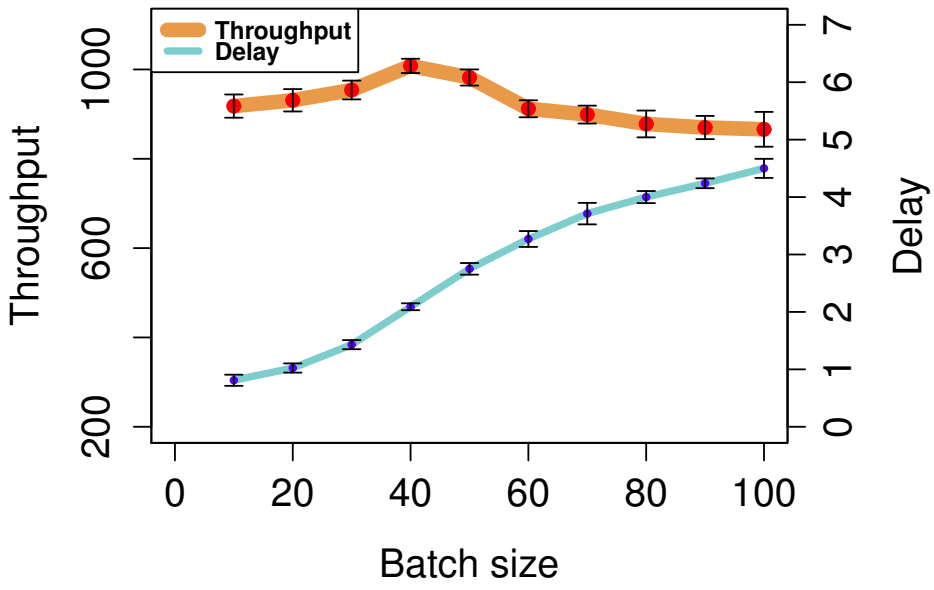

Figure 5.9: Throughput and delay for different batch sizes

5.4 Summary

In this chapter, we proposed TCPFender, which is a novel mechanism to support TCP with opportunistic routing network coding. TCPFender completes the control feedback loop of TCP by creating a bridge between the adaptation modules of the sender and the receiver. The sender adaptation layer in TCPFender differentiates duplicate ACKs caused by network congestion from these caused by opportunistic routing, and the receiver side releases ACK segments whenever receiving an innovative packet. In current work, we implemented our algorithm to support TCP Reno. In fact, TCPFender can also support other TCP protocols with loss-based congestion control (e.g., TCP-NewReno, TCP-Tahoe). The adaptive modules are designed generally enough to not only support opportunistic routing and network coding, but also any packet forwarding techniques that can cause many dropping packets or out-of-order arrivals. One example will be multi-path routing, where IP packets of the same data flow can follow different paths from the source to the destination. By simulating how TCP receiver will signal the TCP sender, we are able to adapt TCPFender to functioning over such the multi-path routing without having to modify TCP itself.

In the simulation results, we compared TCPFender and TCP/IP in four different network topologies. The result shows that TCPFender has a sizeable throughput gain over TCP/IP, and the gain will be very distinct from each other when the link quality is not that good. We also discussed the influence of batch size on the network throughput and end-to-end packet delay. In general, the bath size has a small impact on the network throughput, but it has direct impact on end-to-end packet delay.

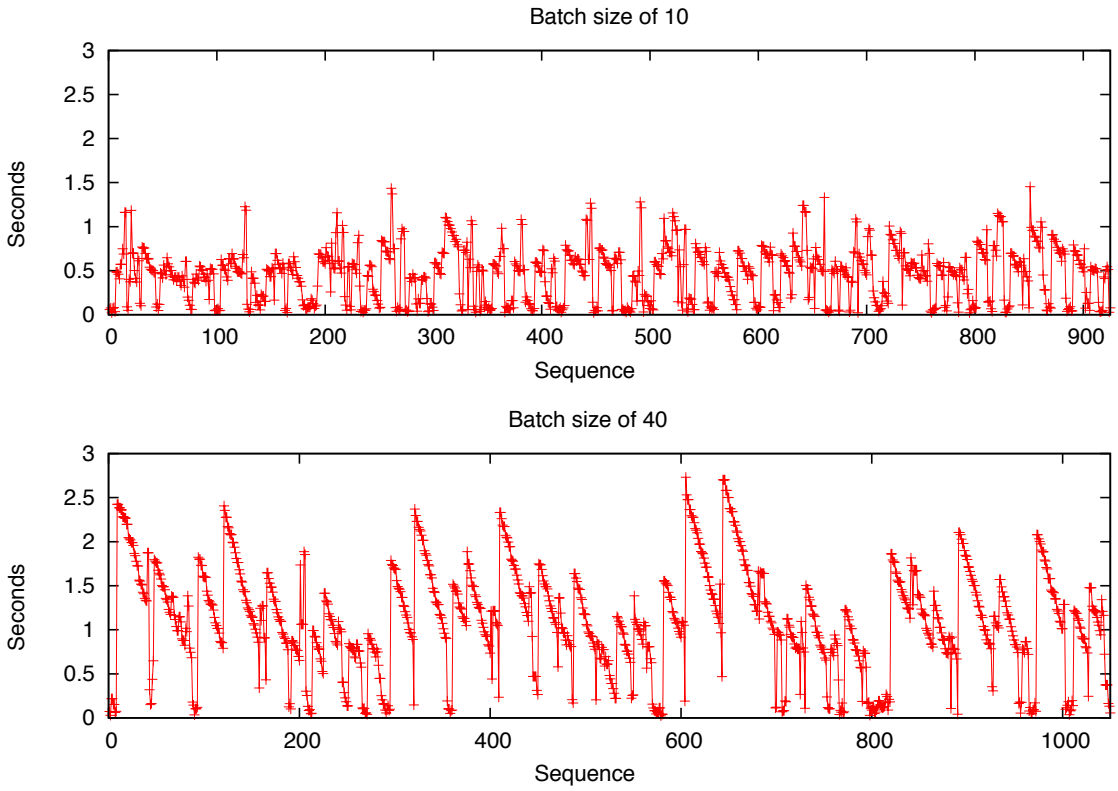

Figure 5.10: Delay for two specific cases with batch sizes of 10 and 40

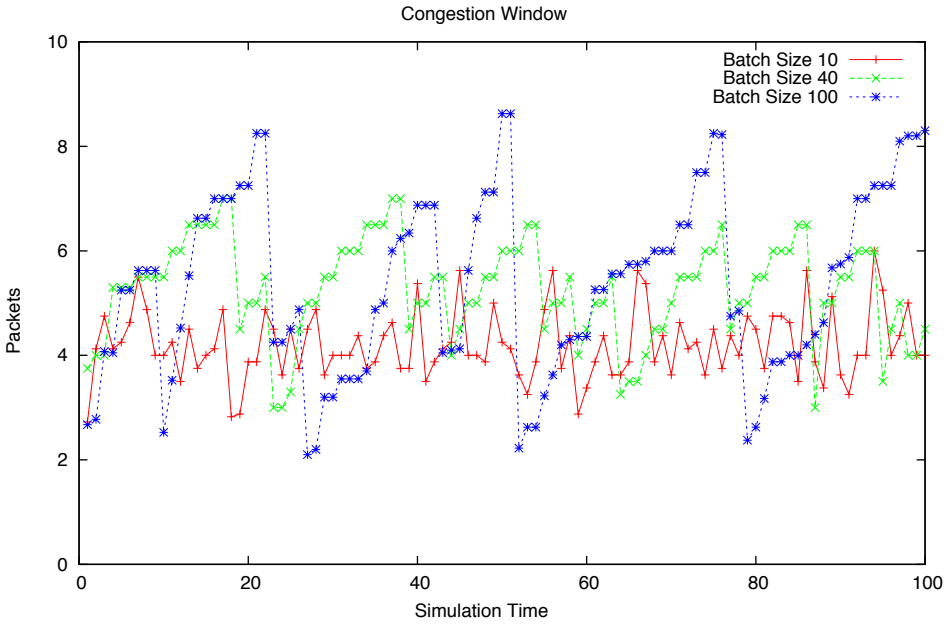

Figure 5.11: Evolution of congestion window for three different batch sizes simulated in the mesh topology.

Chapter 6

Conclusions and Future Work

In this final chapter, we will summarize the contributions presented in this dissertation and discuss future work.

6.1 Conclusions

Multi-hop wireless mesh networks aim to enable an unlimited number of people freely and easily communicate with one another in an unrestricted way and eventually realize the dream of a seamlessly connected world. However, the distinct features and critical design factors of multi-hop wireless networks also pose several challenges. Since the last decade, opportunistic routing and network coding are proposed to improve the performance of wireless networks by proactively exploiting the broadcast nature and space diversity provided by wireless media. Opportunistic routing considers multiple potential paths to transmit packets to the destination, the successful rate of each progression of forwarding can be much improved. Network coding improves throughput performance and reliability in wireless media. Intra-flow network coding mixes packets from the same

flow and provides higher resilience to packet losses.

In this dissertation, we described the background of opportunistic routing and network coding in Chapter 1. Then, we surveyed the main research of this topic in Chapter 2. In Chapter 3, we described an analytical model to quantify the benefits of opportunistic routing over traditional routing. After studying the performance of opportunistic routing, Chapter 4 presented ExOR Compact, which created a regional transmission schedule to improve the UDP throughput. In Chapter 5, we added an adaptation layer to mask the packet loss caused by wireless link errors and provide early positive feedbacks to trigger a larger congestion window for TCP. This adaptation layer functions over the network layer without any modification of current TCP communication systems.

The following conclusions can be drawn from this dissertation:

- We explain the fundamental idea of opportunistic routing and propose a discrete-time Markov chain as a general model to map the transmission process of opportunistic routing. Our study demonstrates how to map packet transmissions in the network with state transitions in a Markov chain. In our model, we consider the pipelined data transfer and evaluate opportunistic routing for different wireless networks. Our modeling techniques proposed in this thesis provide a comprehensive framework to study the performance of packet transmissions, and also can be considered as an important fundamental toward modeling more complicated scenarios leading to the communication networks with better performance.

- Batch-based coordination is the most frequently investigated topic in opportunistic routing. The interference between packets in a batch can also affect the performance of opportunistic routing. Our work is the first study investigating the effect of batch-

based transmissions of opportunistic routing. We take into account the interference model, call Protocol Model, to analyze the pipelined data transfer. Our Markov model is independent from network topology and candidate selection algoritym. The input parameters are batch sizes, packet success rates, number of nodes, and pipeline sets. We will analyze how these parameters affect the performance of opportunistic routing in terms of the expected number of packet transmissions and time slots.

- Our analytical models consider pipelining, and it does not only present the advantage of opportunistic routing but also explores the spatial reuse problem of it. It takes into account the theory of the absorbing matrix and enables the estimation of the number of state transitions in a very effective way. Considering different coordination mechanisms, the proposed models can simulate different network transmission process in wireless mesh networks including both opportunistic routing and traditional IP forwarding.

- We use computer simulation to verify the validity of the analytical model by both NS-2 and Java project in different scenarios. The opportunity gain of packet transmissions are increasing function of the diversity of transmission paths. In addition, the difference between the simulation and the numerical results showes that how collision and retransmissions affect the performance of opportunistic routing.

- We propose ExOR Compact as a joint opportunistic routing and network coding technique to improve network performance. Coordination of opportunistic packet forwarding is limited to the greatest possible range of the transmitting node rather than among the entire set of forwarders. As such, a node's mandatory waiting time is reduced. With such a higher degree of spatial channel reuse, ExOR Compact has

a data pipeline for higher data transfer capacity.

- ExOR Compact is a network-layer-only solution, and it relies on the standard 802.11 MAC to accommodate multiple forwarding activities scheduled at the same time. Once a packet is given to the MAC, it is completely up to the control of the MAC. Thus, ExOR Compact keeps packets at the network layer as long as possible so that a node can make a last-minute change or cancellation.

- A lower-priority forwarder in ExOR Compact will monitor the channel activities to gauge the number of innovative packets that a higher-priority forwarder has collected so that it can determine a proper number of coded transmissions to attempt at its turn.

- We propose TCPFender to incorporated TCP with opportunistic routing and network coding. TCPFender adds an adaptation layer above the network layer to cooperate with TCP's control feedback loop. This adaptation layer masks the packet loss caused by wireless link errors and provides early positive feedbacks to trigger a larger congestion window for TCP. It can distinguish duplicated ACKs caused by out-of-order arrivals in opportunistic routing from those caused by network congestion and make the TCP's congestion control work well with the frequent occurrences of out-of-order packets in opportunistic routing and long decoding delay in network coding.

- Opportunistic routing and network coding do not inherently support TCP, so many previous research on opportunistic data forwarding and network coding were not designed for TCP. Other studies modified TCP protocols by cooperating network

coding into TCP protocols; these works created different variants of TCP protocols to improve the throughput. However, TCP protocols (especially, TCP Reno) are widely deployed in current communication systems, it is not easy work to modify all TCP protocols of the communication systems. Therefore, we propose an adaptation layer (TCPFender) functioning below TCP Reno. With the help of TCPFender, TCP Reno does not make any change to itself and it can take advantage of both network coding and opportunistic data forwarding, it is easy to deploy in wireless mesh networks.

- We compare the throughput of TCPFender and TCP/IP in different topologies of wireless mesh networks, and analyze the influence of batch sizes on the TCP throughput and the end-to-end delay. Comparing TCPFender with other baselines in aspects such as throughput, end-to-end delay and evolution of TCP congestion window, TCPFender can significantly outperform TCP/IP.

6.2 Future Work

There are various directions to extend our work, which can be briefly outlined as follows:

- *Analytical model of OR with NC* – As shown in this dissertation and other related works, the combination of opportunistic routing and network coding in a single joint protocol outperforms each of them individually. To the best of our knowledge, there is no comprehensive analytical model studying the performance under this joint approach. In future work, our models will be extended to analyze opportunis-

tic routing with network coding. Both opportunistic routing and intra-flow network coding take advantage of the broadcast nature of the wireless medium and transmit data packets as a batch in each round. Network coding randomly mixes a batch of packets before forwarding them, which can be exploited to avoid or minimize replicate transmissions. Each coded packet randomly combines partial information of data packets in a batch, which further extends the distribution of data packets in the network. Network coding can increase the throughput and the stability of transmissions. To map opportunistic routing with network coding by a Markov chain, two important aspects should be considered. First, as network coding fuses information in the original packets, the Markov model can no longer track individual original packets without a significant enhancement. Second, the destination is considered to have received all original packets only if it has accumulated sufficient evidence to have a full-rank decoding matrix. How quickly this happens is determined by the way intermediate nodes recode packets and how aggressively they do it. Thus, coded packets transiting in the network is an important factor of the protocol overhead, and estimating how it interacts with the packet delivery performance is crucial. Further studies will take into account the amount of consumed resources for opportunistic routing in mobile ad hoc networks and vehicular networks, where the link quality frequently changes. The dynamic changes of link qualities require a time-variation matrix to represent the state transition process. We will consider how to map these dynamic network states into a Markov chain.

- *Analytical model of OR with both Intra- and Inter-flow NC* – A number of research have been proposed to combine intra- and inter-flow network coding, however, they

only explored the coding opportunities along fixed routing path because opportunistic routing works with these two network coding techniques in different ways. Opportunistic routing with intra-flow network coding transmits packets in the same direction, the coordination focused on avoiding duplicate transmissions. The transmission probability of one packet to multiple downstream nodes are considered to calculate opportunity gains. Opportunistic routing with inter-flow network coding codes packets from different flows where the directions of them are opposing. Opportunity gains depend on who are the senders of two packets. When an intermediate node receives two packets from two flows and the sender of one packet is the recevier of another packet, vice versa, the intermediate node will code these two packets and transmit them by one transmission. We believe that the analytical model of opportunistic routing with intra- and inter-flow network coding can be an interesting area of future research investigation, which helps better understand both approachs and provides more reliable and efficient communication networks.

- *Adding Inter-flow Network Coding to ExOR Compact* – ExOR Compact proposes a regional transmission schedule to limit the waiting time being only a function of the transmission range rather than of the entire forwarder list. The regional transmission schedule involved only a subset of the forwarders that are within range of the sender. Whenever the transmission range of two senders has overlapped each other, packets from different flows can be coded together to further exploit the coding opportunities in entire network. Thus, it will be interesting to extend ExOR Compact by including a combination of intra- and inter-flow network coding.

- *ExOR Compact works with TCP* – Our adaptive forwarding schedule is designed

generally enough to not just support UDP transmission, but also TCP transmission. It can adapt TCP congestion window by acknowledgementing the number of innovative packets as the sequence number of TCP. We will discuss TCP performance in our future work. On the other hand, ExOR Compact can be extended to verify how multiple flows interact with each other or over more generally any error-prone wireless networks.

- *TCPFender supports RTT-based congestion control* – We will consider TCP protocols with RTT-based congestion control and also analyze how multiple TCP flows interact with each other in a network coded, opportunistic forwarding network layer, or a more generally error-prone network layer. We will refine the redundancy factor and the bandwidth estimation to optimize the congestion control feedback of TCP. Finally, we will propose a theoretical model of TCP with opportunistic routing and network coding, which will enable us to study TCPFender as a function in various communication systems.

References

[1] The Network Simulator (NS-2).

[2] R. Ahlswede, N. Cai, S.-Y. R. Li, and R. W. Yeung. Network Information Flow. *IEEE Transcations on Information Theory*, 46(4):1204–1216, July 2000.

[3] K. Akkaya and M. Younis. A Survey on Routing Protocols for Wireless Sensor Networks. *Ad Hoc Networks*, 3(3):325–349, 2005.

[4] I. F. Akyildiz, X. Wang, and W. Wang. Wireless Mesh Networks: A Survey. *Journal of Computer and Telecommunications Networking*, 47(4):445–487, March 2005.

[5] M. Amerimehr and F. Ashtiani. Delay and Throughput Analysis of a Two-Way Opportunistic Network Coding-Based Relay Network. *IEEE Transactions on Wireless Communications*, 13(5):2863–2873, May 2014.

[6] A. Argyriou. Wireless Network Coding with Improved Opportunistic Listening. *IEEE Transactions on Wireless Communications*, 8(4):2014–2023, April 2009.

[7] W. Bao, V. Shah-Mansouri, V. W. Wong, and V. C. Leung. TCP-VON: Joint Congestion Control and Online Network Coding for Wireless Networks. In *Proceed-*

ings of IEEE Global Communications Conference, pages 125–130, Anaheim, USA, December 2012.

[8] S. Basagni, M. Conti, S. Giordano, and I. Stojmenovic. XORs in the Air: Practical Wireless Network Coding. *IEEE/ACM Transactions on Networking*, 16(3):497–510, June 2008.

[9] R. Bassoli, H. Marques, J. Rodriguez, K. W. Shum, and R. Tafazolli. Network Coding Theory: A Survey. *IEEE Communications Surveys & Tutorials*, 15(4):1950–1978, February 2013.

[10] S. Biswas and R. Morris. ExOR: Opportunistic Multi-Hop Routing for Wireless Networks. *ACM SIGCOMM Computer Communication Review*, 35(4):133–144, October 2005.

[11] A. Boukerche and A. Darehshoorzadeh. Opportunistic Routing in Wireless Networks: Models, Algorithms, and Classifications. *ACM Computing Surveys*, 47(22):22–36, November 2015.

[12] R. Bruno and M. Nurchis. Survey on Diversity-based Routing in Wireless Mesh Networks: Challenges and Solutions. *Computer Communications*, 33(3):269–282, February 2010.

[13] A. Buluc, J. T. Fineman, M. Frigo, J. R. Gilbert, and C. E. Leiserson. Parallel Sparse Matrix-vector and Matrix-transpose-vector Multiplication using Compressed Sparse Blocks. In *Proceedings of Twenty-first Annual Symposium on Parallelism in Algorithms and Architectures*, pages 233–244, Calgary, Canada, August 2009.

[14] N. Cai and R. W. Yeung. Secure Network Coding on a Wiretap Network. *IEEE Transmission on Information Theory*, 57(1):424–435, January 2011.

[15] S. Cai, S. Zhang, G. Wu, Y. Dong, and T. Znati. Minimum Cost Opportunistic Routing with Intra-session Network Coding. In *Proceedings of IEEE International Conference on Communications*, pages 502–507, Sydney, Australia, June 2014.

[16] L. Cerda-Alabern, A. Darehshoorzadeh, and V. Pla. On the Maximum Performance in Opportunistic Routing. In *Proceedings of IEEE International Symposium on World of Wireless Mobile and Multimedia Networks*, pages 1–8, Montreal, Canada, June 2010.

[17] L. Cerdà-Alabern, A. Darehshoorzadeh, and V. Pla. Optimum Node Placement in Wireless Opportunistic Routing Networks. *Ad Hoc Networks*, 11(8):2273–2287, November 2013.

[18] S. Chachulski, M. Jennings, S. Katt, and D. Katabi. Trading Structure for Randomness in Wireless Opportunistic Routing. *ACM SIGCOMM Computer Communication Review*, 37(12):169–180, October 2007.

[19] N. Chakchouk. A Survey on Opportunistic Routing in Wireless Communication Networks. *IEEE Communications Surveys & Tutorials*, 17(4):2214 – 2241, March 2015.

[20] C.-C. Chen, C. Chen, S. Y. Oh, J.-S. Park, M. Gerla, and M. Sanadidi. ComboCoding: Combined Intra-/Inter-flow Network Coding for TCP over Disruptive MANETs. *Journal of Advanced Research*, 2(3):241–252, July 2011.

[21] T. Chen and S. Zhong. An Enforceable Scheme for Packet Forwarding Cooperation in Network-Coding Wireless Networks With Opportunistic Routing. *IEEE Transactions on Vehicular Technology*, 63(9):4476–4491, November 2014.

[22] P. A. Chou, Y. Wu, K. Jain, and P. A. Chou. Practical network coding. In *Proceedings of Allerton Conference on Communication, Control, and Computing*, October 2003.

[23] D. D. Couto, D. Aguayo, J. Bicket, and R. Morris. A High-throughput Path Metric for Multi-hop Wireless Routing. In *Proceedings of ACM International Conference on Mobile Computing and Networking*, pages 134–146, New York, USA, 2003.

[24] A. Darehshoorzadeh, M. Almulla, A. Boukerche, and S. Chaiwala. On the Number of Candidates in Opportunistic Routing for Multi-hop Wireless Networks. In *Proceedings of ACM International Symposium on Mobility Management and Wireless Access*, pages 9–16, New York, USA, November 2013.

[25] A. Darehshoorzadeh, R. E. D. Grande, and A. Boukerche. Towards a Comprehensive Model for Performance Analysis of Opportunistic Routing in Wireless Mesh Networks. *IEEE Transactions on Vehicular Technology*, 65(7):5424–5438, July 2015.

[26] H. Dubois-Ferrière, M. Grossglauser, and M. Vetterli. Valuable Detours: Least-cost Anypath Routing. *IEEE/ACM Transactions on Networking*, 19(2):333–346, April 2011.

[27] M. Z. Farooqi, S. M. Tabassum, M. H. Rehmani, and Y. Saleem. A Survey on Network Coding: From Traditional Wireless Networks to Emerging Cognitive Radio

Networks. *Journal of Network and Computer Applications*, 46:166–181, November 2014.

[28] C. Fragouli, J.-Y. L. Boudec, and J. Widmer. Network Coding: An Instant Primer. *ACM SIGCOMM Computer Communication Review*, 36(1):63–68, January 2006.

[29] C. Fragouli, J. Widmer, and J.-Y. L. Boudec. A Network Coding Approach to Energy Efficient Broadcasting: From Theory to Practice. In *Proceedings of IEEE International Conference on Computer Communications*, pages 1–11, Barcelona, Spain, April 2006.

[30] J. J. Garcia-Luna-Aceves and J. Behrens. Distributed, Scalable Routing Based on Vectors of Link States. *IEEE Journal on Selected Areas in Communications*, 13(8):1383–1395, October 1995.

[31] P. Garrido, D. Gómez, R. Agüero, and J. Serrat. Combination of Random Linear Coding and Cross-layer Opportunistic Routing: Performance over Bursty Wireless Channels. In *Proceedings of IEEE International Symposium on Personal, Indoor, and Mobile Radio Communications*, pages 1692–1696, Hong Kong, China, August 2015.

[32] C. Gkantsidis. Network Coding for Large Scale Content Distribution. In *Proceedings of IEEE International Conference on Computer Communications*, pages 1–11, Miami, USA, March 2005.

[33] C. M. Grinstead and J. L. Snell. *Introduction to Probability*. American Mathmatical Society, 1997.

[34] B. Guo, H. Li, C. Zhou, and Y. Cheng. Analysis of General Network Coding Conditions and Design of a Free-Ride-Oriented Routing Metric. *IEEE Transactions on Vehicular Technology*, 60(4):1714–1727, May 2011.

[35] P. Gupta and d P. R. Kumar. The Capacity of Wireless Networks. *IEEE Transcations on Information Theory*, 46(2):388–404, March 2000.

[36] L. Hai, H. Wang, J. Wang, and Z. Tang. HCOR: A High-throughput Coding-aware Opportunistic Routing for Inter-flow Network Coding in Wireless Mesh Networks. *EURASIP Journal on Wireless Communications and Networking*, 2014(1):148–160, December 2014.

[37] T. Ho, R. Koetter, M. Medard, D. R. Karger, and M. Effros. The Benefits of Coding over Routing in a Randomized Setting. In *Proceedings of IEEE International Symposium on Information Theory*, pages 442–442, Yokohama, Japan, June 2003.

[38] T. Ho, M. Médard, R. Koetter, D. R. Karger, M. Effros, J. Shi, and B. Leong. A Random Linear Network Coding Approach to Multicast. *IEEE Transmission on Information Theory*, 52(10):4413–4430, October 2006.

[39] Q. Hu and J. Zheng. CoAOR: An Efficient Network Coding Aware Opportunistic Routing Mechanism for Wireless Mesh Networks. In *Proceedings of IEEE Global Communications Conference*, pages 4578–4583, Atlanta, USA, December 2013.

[40] Y. Huang, M. Ghaderi, D. Towsley, and W. Gong. TCP Performance in Coded Wireless Mesh Networks. In *Proceedings of IEEE Communications Society Conference on Sensor and Ad Hoc Communications and Networks*, pages 179–187, San Francisco, USA, June 2008.

[41] M. A. Iqbal, B. Dai, B. Huang, A. Hassan, and S. Yu. Review: Survey of Network Coding-aware Routing Protocols in Wireless Networks. *Journal of Network and Computer Applications*, 34(6):1956–1970, November 2011.

[42] J. Islam and D. P. K. Singh. CORMEN: Coding-aware Opportunistic Routing in Wireless Mesh Network. *Journal of Computing*, 2(6):71–77, June 2010.

[43] P. Jadhav and R. Satao. A Survey on Opportunistic Routing Protocols for Wireless Sensor Networks. *Procedia Computer Science*, 79(2016):603–609, April 2016.

[44] M. Jafarisiavoshani, C. Fragouli, and S. Diggavi. Subspace Properties of Randomized Network Coding. In *Proceedings of IEEE Information Theory Workshop on Information Theory for Wireless Networks*, pages 1–5, Solstrand, Norway, September 2007.

[45] S. Jaggi, M. Langberg, S. Katti, T. Ho, D. Katabi, M. Médard, and M. Effros. Resilient Network Coding in the Presence of Byzantine Adversaries. *IEEE Transmission on Information Theory*, 54(6):2596–2603, June 2008.

[46] S. Jaggi, P. Sanders, P. A. Chou, M. Effros, S. Egner, K. Jain, and L. M. G. M. Tolhuizen. Polynomial Time Algorithms for Multicast Network Code Construction. *IEEE Transmission on Information Theory*, 51(6):1973–1982, June 2005.

[47] V. Jamali, N. Zlatanov, and R. Schober. Bidirectional Buffer-aided Relay Networks With Fixed Rate Transmission-Part II: Delay-Constrained Case. *IEEE Transactions on Wireless Communications*, 14(3):1339–1355, March 2015.

[48] A. Jiang. Network Coding for Joint Storage and Transmission with Minimum Cost. In *Proceedings of IEEE International Symposium on Information Theory*, pages 1359–1363, Seattle, USA, July 2006.

[49] X. Jiao, X. Wang, and X. Zhou. Active Network Coding Based High-Throughput Optimizing Routing for Wireless Ad Hoc Networks. In *Proceedings of 4th International Conference on Wireless Communications, Networking and Mobile Computing*, pages 1–5, Dalian, China, October 2008.

[50] D. B. Johnson, D. A. Maltz, and J. Broch. DSR: the Dynamic Source Routing Protocol for Multihop Wireless Ad hoc Networks. *Ad hoc networking*, 1(4):139–172, May 2001.

[51] J. Joy, Y.-T. Yu, M. Gerla, S. Wood, J. Mathewson, and M.-O. Stehr. Network Coding for Content-based Intermittently Connected Emergency Networks. In *Proceedings of the 19th Annual International Conference on Mobile Computing and Networking*, pages 123–126, Miami, USA, October 2013.

[52] S. Kafaie, Y. Chen, M. H. Ahmed, and O. A. Dobre. Flexonc: Joint cooperative forwarding and network coding with precise encoding conditions. *IEEE Transactions on Vehicular Technology*, 66(8):7262 – 7277, August 2017.

[53] S. Katti, S. Gollakota, and D. Katabi. Embracing Wireless Interference: Analog Network Coding. *ACM SIGCOMM Computer Communication Review*, 37(4):397–408, October 2007.

[54] S. Katti, D. Katabi, H. Balakrishnan, and M. Medard. Symbol-level Network Coding for Wireless Mesh Networks. *ACM SIGCOMM Computer Communication Review*, 38(4):401–412, October 2008.

[55] A. Khreishah, I. M. Khalil, and J. Wu. Universal Opportunistic Routing Scheme using Network Coding. In *Proceedings of IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks*, pages 353–361, Seattle, USA, June 2015.

[56] R. Koetter and M. Médard. An Algebraic Approach to Network Coding. *IEEE/ACM Transactions on Networking*, 11(5):782–795, October 2003.

[57] D. Koutsonikolas, Y. C. Hu, and C. C. Wang. XCOR: Synergistic Interflow Network Coding and Opportunistic Routing. In *Proceedings of the 14th Annual International Conference on Mobile Computing and Networking*, pages 1–7, San Francisco, USA, September 2008.

[58] D. Koutsonikolas, C.-C. Wang, and Y. C. Hu. CCACK: Efficient Network Coding Based Opportunistic Routing Through Cumulative Coded Acknowledgments. *IEEE/ACM Transactions on Networking*, 19(5):1368–1381, October 2011.

[59] J. Krigslund, J. Hansen, M. Hundeboll, D. E. Lucani, and F. H. P. Fitzek. CORE: COPE with MORE in Wireless Meshed Networks. In *Proceedings of IEEE Vehicular Technology Conference*, pages 1–6, Dresden, Germany, June 2013.

[60] P. Larsson. Selection Diversity Forwarding in a Multihop Packet Radio Network with Fading Channel and Capture. *ACM SIGMOBILE Mobile Computing and Communications Review*, 5(4):47–54, October 2001.

[61] J. Le, J. C. Lu, and D. M. Chiu. DCAR: Distributed Coding-aware Routing in Wireless Networks. *IEEE Transcations on Mobile Computing*, 9(4):596–608, April 2008.

[62] P. Li, S. Guo, S. Yu, and A. V. Vasilakos. CodePipe: An Opportunistic Feeding and Routing Protocol for Reliable Multicast with Pipelined Network Coding. In *Proceedings of IEEE International Conference on Computer Communications*, pages 100–109, Orlando, USA, March 2012.

[63] S.-Y. R. Li, R. W. Yeung, and N. Cai. Linear Network Coding. *IEEE Transmission on Information Theory*, 49(2):371–381, February 2003.

[64] Y. Li, Y. an Liu, and P. Luo. Link Probability Based Opportunistic Routing Metric in Wireless Network. In *Proceedings of International Conference on Communications and Mobile Computing*, pages 308–312, Yunnan, China, Jane 2009.

[65] Y. Li, W. Chen, and Z.-L. Zhang. Optimal Forwarder List Selection in Opportunistic Routing. In *Proceedings of IEEE International Conference on Mobile Ad hoc and Sensor Systems*, pages 670–675, Macau, China, October 2009.

[66] Y. Li and Z.-L. Zhang. Random Walks and Green's Function on Digraphs: A Framework for Estimating Wireless Transmission Costs. *IEEE/ACM Transactions on Networking*, 21(1):135–148, February 2013.

[67] R. Lidl and H. Niederreiter. *Finite Fields*, volume 20. Cambridge University Press, 1997.

[68] S. Lin, L. Fu, J. Xie, and X. Wang. Hybrid Network Coding for Unbalanced Slotted ALOHA Relay Networks. *IEEE Transactions on Wireless Communications*, 15(1):298–313, January 2016.

[69] Y. Lin, B. Li, and B. Liang. CodeOR: Opportunistic Routing in Wireless Mesh Networks with Segmented Network Coding. In *Proceedings of IEEE International Conference on Network Protocols*, pages 13–22, Orlando, USA, October 2008.

[70] Y. Lin, B. Li, and B. Liang. Efficient Network Coded Data Transmissions in Disruption Tolerant Networks. In *Proceedings of IEEE International Conference on Computer Communications*, pages 2180–2188, Phoenix, USA, April 2008.

[71] Y. Lin, B. Liang, and B. Li. SlideOR: Online Opportunistic Network Coding in Wireless Mesh Networks. In *Proceedings of IEEE International Conference on Computer Communications*, pages 1–5, San Diego, USA, 2010.

[72] Y.-J. Lin, C.-C. Huang, and J.-L. Huang. PipelineOR: A Pipelined Opportunistic Routing Protocol with Network Coding in Wireless Mesh Networks. In *Proceedings of IEEE Vehicular Technology Conference*, Ottawa, Canada, May 2010.

[73] H. Liu, B. Zhang, H. T. Mouftah, X. Shen, and J. Ma. Opportunistic Routing for Wireless Ad Hoc and Sensor Networks: Present and Future Directions. *IEEE Communications Magazine*, 47(12):103–109, December 2009.

[74] C.-P. Luk, W.-C. Lau, and O.-C. Yue. An Analysis of Opportunistic Routing in Wireless Mesh Network. In *Proceedings of IEEE International Conference on Communications*, pages 2877–2883, Beijing, China, May 2008.

[75] T. Mehta and Z. Narmawala. Survey on Multimedia Transmission Using Network Coding over Wireless Networks. In *Proceedings of Nirma University International Conference on Engineering*, pages 1–6, Ahmedabad, India, December 2011.

[76] S. Murthy. *Routing in Packet-Switched Networks Using Path-Finding Algorithms*. PhD thesis, University of California, CA, United States, 1996.

[77] P. Ostovari, J. Wu, and A. Khreishah. *Network Coding Techniques for Wireless and Sensor Networks*. The Art of Wireless Sensor Networks, December 2014.

[78] G. Paschos, C. Fragiadakis, L. Georgiadis, and L. Tassiulas. Wireless Network Coding with Partial Overhearing Information. In *Proceedings of IEEE International Conference on Computer Communications*, pages 2337–2345, Turin, Italy, April 2013.

[79] C. E. Perkins and P. Bhagwat. Highly Dynamic Destination-Sequenced Distance-Vector routing (DSDV) for Mobile Computers. *ACM SIGCOMM Computer Communication Review*, 24(4):234–244, September 1994.

[80] C. E. Perkins and E. M. Royer. Ad-hoc On-Demand Distance Vector Routing. In *Proceedings of IEEE Workshop on Mobile Computing Systems and Applications*, pages 90–96, Washington, USA, February 1999.

[81] B. Poonguzharselvi and V. Vetriselvi. Survey on Routing Algorithms in Opportunistic Networks. In *Proceedings of International Conference on Computer Communication and Informatics*, pages 1–4, Coimbatore, India, February 2013.

[82] B. Radunovic, C. Gkantsidis, P. Key, and P. Rodriguez. Toward Practical Opportunistic Routing with Intra-session Network Coding for Mesh Networks. *IEEE/ACM Transactions on Networking*, 18(2):420–433, January 2010.

[83] E. Rozner, J. Seshadri, Y. A. Mehta, and L. Qiu. SOAR: Simple Opportunistic Adaptive Routing Protocol for Wireless Mesh Networks. *IEEE Transcations on Mobile Computing*, 8(12):1622–1635, December 2009.

[84] H. Seferoglu, A. Markopoulou, and K. K. Ramakrishnan. Intra- and Inter-Session Network Coding in Wireless Networks. In *Proceedings of IEEE International Conference on Computer Communications*, pages 1035–1043, Shanghai, China, April 2011.

[85] H. Shen, G. Bai, L. Zhao, and Z. Tang. An Adaptive Opportunistic Network Coding Mechanism in Wireless Multimedia Sensor Networks. *International Journal of Distributed Sensor Networks*, 8(12):1–13, December 2012.

[86] F. Soldo, A. Markopoulou, and A. L. Toledo. A Simple Optimization Model for Wireless Opportunistic Routing with Intra-Session Network Coding. In *Proceedings of IEEE International Symposium on Network Coding*, pages 1–6, Washington, USA, July 2010.

[87] J. Sun, Y. Zhang, D. Tang, S. Zhang, Z. Zhao, and S. Ci. TCP-FNC: A Novel TCP with Network Coding for Wireless Networks. In *Proceedings of IEEE International Conference on Communications*, pages 2078–2084, London, UK, June 2015.

[88] J. K. Sundararajan, D. Shah, M. Medard, S. Jakubczak, M. Mitzenmacher, and J. Barros. Network Coding Meets TCP: Theory and Implementation. *Proceedings of the IEEE*, 99(3):490–512, March 2011.

[89] N. R. Wagner. *The Laws of Cryptography with Java Code*. Available online at Neal Wagner's home page, 2003.

[90] Z. Wang, Y. Chen, and C. Li. CORMAN: A Novel Cooperative Opportunistic Routing Scheme in Mobile Ad Hoc Networks. *IEEE Journal on Selected Areas in Communications*, 30(2):289–296, February 2012.

[91] Z. Wang, Y. Chen, and C. Li. PSR: A Lightweight Proactive Source Routing Protocol For Mobile Ad Hoc Networks. *IEEE Transactions on Vehicular Technology*, 63(2):859–868, February 2014.

[92] J. Widmer and J.-Y. Le Boudec. Network Coding for Efficient Communication in Extreme Networks. In *Proceedings of the 2005 ACM SIGCOMM Workshop on Delay-tolerant Networking*, pages 284–291, Pennsylvania, USA, August 2005.

[93] Y. Wu, P. A. Chou, and K. Jain. A Comparison of Network Coding and Tree Packing. In *Proceedings of IEEE International Symposium on Information Theory*, pages 145–146, Chicago, USA, June 2004.

[94] Q. Xiang, H. Zhang, J. Wang, G. Xing, S. Lin, and X. Liu. On Optimal Diversity in Network-Coding-Based Routing in Wireless Networks. In *Proceedings of IEEE International Conference on Computer Communications*, pages 765–773, Hong Kong, China, April 2015.

[95] Y. Yan, B. Zhang, H. T. Mouftah, and J. Ma. Practical Coding-aware Mechanism for Opportunistic Routing in Wireless Mesh Networks. In *Proceedings of IEEE International Conference on Communications*, pages 2871–2876, Beijing, China, May 2008.

[96] Y. Yan, B. Zhang, J. Zheng, and J. Ma. CORE: A Coding-aware Opportunistic Routing Mechanism for Wireless Mesh Networks. *IEEE Wireless Communications*, 17(3):96–103, June 2010.

[97] Y. Yuan, H. Yang, S. H. Wong, S. Lu, and W. Arbaugh. ROMER: Resilient Opportunistic Mesh Routing for Wireless Mesh Networks. In *IEEE Workshop on Wireless Mesh Networks*, pages 1–9, Santa Clara, USA, September 2005.

[98] D. Zeng, S. Guo, Y. Xiang, and H. Jin. On the Throughput of Two-Way Relay Networks Using Network Coding. *IEEE Transactions on Parallel and Distributed Systems*, 25(1):191–199, January 2014.

[99] K. Zeng, Z. Yang, and W. Lou. Location-Aided Opportunistic Forwarding in Multirate and Multihop Wireless Networks. *IEEE Transactions on Vehicular Technology*, 58(6):3032–3040, 2009.

[100] J. Zhang, Y. P. Chen, and I. Marsic. MAC Scheduling Using Channel State Diversity for High-Throughput IEEE 802.11 Mesh Networks. *IEEE Communications Magazine*, 45(11):94–99, November 2007.

[101] S. Zhang, S.-C. Liew, and P. P. K. Lam. Physical Layer Network Coding. In *Proceedings of ACM International Conference on Mobile Computing and Networking*, pages 358–365, New York, USA, March 2006.

[102] Z. Zhong, J. Wang, and S. Nelakuditi. On Selection of Candidates for Opportunistic Any Path Forwarding. *ACM SIGMOBILE Mobile Computing and Communications Review*, 10(4):1–2, October 2006.

[103] D. Zhu, X. Yang, W. Yu, C. Lu, and X. Fu. INCOR: Inter-flow Network Coding based Opportunistic Routing in Wireless Mesh Networks. In *Proceedings of IEEE International Conference on Communications*, pages 3666–3671, London, UK, June 2015.