Network Coding: Performance Analysis and Robust Design in Multi-hop Wireless Mesh Networks

by

© Somayeh Kafaie

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

Doctor of Philosophy

Faculty of Engineering and Applied Science
Memorial University of Newfoundland

October 2017

St. John's, Newfoundland

Abstract

Network coding is an innovative idea to boost the capacity of wireless networks. However, there are not enough analytical studies on throughput and end-to-end delay of network coding in multi-hop wireless mesh network that incorporates the specifications of IEEE 802.11 Distributed Coordination Function. In this dissertation, we utilize queuing theory to propose an analytical framework for bidirectional unicast flows in multi-hop wireless mesh networks. We study the throughput and end-to-end delay of inter-flow network coding under the IEEE 802.11 standard with CSMA/CA random access and exponential back-off time considering clock freezing and virtual carrier sensing, and formulate several parameters such as the probability of successful transmission in terms of bit error rate and collision probability, waiting time of packets at nodes, and retransmission mechanism. Our model uses a multi-class queuing network with stable queues, where coded packets have a non-preemptive higher priority over native packets, and forwarding of native packets is not delayed if no coding opportunities are available. The accuracy of our analytical model is verified using computer simulations.

Furthermore, while inter-flow network coding is proposed to help wireless networks approach the maximum capacity, the majority of research conducted in this area is yet to fully utilize the broadcast nature of wireless networks, and to perform effectively under poor channel quality. This vulnerability is mostly caused by assuming fixed route between the source and destination that every packet should travel through. This assumption

not only limits coding opportunities, but can also cause buffer overflow at some specific intermediate nodes. Although some studies considered scattering of the flows dynamically in the network, they still face some limitations. This dissertation explains pros and cons of some prominent research in network coding and proposes a Flexible and Opportunistic Network Coding scheme (FlexONC) as a solution to such issues. Moreover, this research discovers that the conditions used in previous studies to combine packets of different flows are overly optimistic and would affect the network performance adversarially. Therefore, we provide a more accurate set of rules for packet encoding. The experimental results show that FlexONC outperforms previous methods especially in networks with high bit error rates, by better utilizing redundant packets permeating the network, and benefiting from precise coding conditions.

To my family ...

Acknowledgements

As this chapter of my life comes to an end, it is a great pleasure to acknowledge several individuals who have contributed to who I am today.

I am deeply indebted to Dr. Yuanzhu Chen, the best supervisor I have ever known in my entire life, whose expertise, understanding, willingness to help and tremendous patience made this difficult journey easier, productive and a lot of fun. His joy and enthusiasm for the research, and his support and belief in me guided me during tough times and added considerably to my graduate experience.

I am also massively indebted to my other supervisors, Dr. Mohamed H. Ahmed and Dr. Octavia A. Dobre, for their excellent support, caring and patience. Their valuable guidance, dedication, and encouragement have pushed me far beyond my expectations.

A significant part of my education was in Iran, where my foundations were laid. I wish to offer my sincere thanks to all my teachers at "13 Aban" elementary school, "Nemooneh" middle school and "Daneshgah" high school in Zabol, and Amirkabir University of Technology and Iran University of Science and Technology in Tehran.

I would like to acknowledge the financial support provided by my supervisors, Natural Science and Engineering Research Council (NSERC), the School of Graduate Studies, the Faculty of Engineering and Applied Science, the Faculty of Science, Emera Inc. (The

Emera Graduate Scholarship for Engineering and Applied Science) and IEEE ComSoc (Student Travel Grant to attend IEEE GlobeCom 2016).

I would like to thank the staff of Memorial University, especially at the Department of Electrical and Computer Engineering and the Department of Computer Science, and all my friends and colleagues, especially in Wireless Networking and Mobile Computing Laboratory (WineMocol), for the help I have received during more than four years and for a pleasant working atmosphere.

I wish to express the deepest appreciation to my parents, my first and best teachers, and to my two brothers, the most amazing brothers of the world; without your unconditional love and support, and your unwavering belief in me I would never have accomplished all I have in my life.

Finally, I wish to convey my warmest thanks to my husband, who was always there cheering me up and standing by me through the good times and bad; without you nothing is complete.

Co-Authorship Statements

I, Somayeh Kafaie, hold a principle author status for all the manuscript chapters (Chapter 2 - 5) in this dissertation. However, each manuscript is co-authored by my supervisors, Dr. Yuanzhu Chen, Dr. Mohamed Hossam Ahmed, and Dr. Octavia A. Dobre, 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 followings:

- Chapter 2:

  "Joint Inter-flow Network Coding and Opportunistic Routing in Multi-hop Wireless Mesh Networks: A Comprehensive Survey," Submitted to IEEE Communication Surveys and Tutorials, 2017.

- Chapter 3:

  "Performance Analysis of Network Coding with IEEE 802.11 DCF in Multi-Hop Wireless Networks," accepted in IEEE Transactions on Mobile Computing, 2017.

  "Throughput Analysis of Network Coding in Multi-Hop Wireless Mesh Networks using Queuing Theory," in proceedings of IEEE Global Communications Conference (GlobeCom), pp. 1-6, 2016.

- Chapter 4:

  "FlexONC: Joint Cooperative Forwarding and Network Coding with Precise Encoding Conditions," IEEE Transactions on Vehicular Technology, vol. 66, no., 8, pp. 7262 - 7277, Aug. 2017.

  "Network Coding with Link Layer Cooperation in Wireless Mesh Networks", in proceedings of IEEE International Conference on Communications (ICC), pp. 3672-3677, 2015.

- Chapter 5:

  "FlexONC: Joint Cooperative Forwarding and Network Coding with Precise Encoding Conditions," IEEE Transactions on Vehicular Technology, vol. 66, no., 8, pp. 7262 - 7277, Aug. 2017.

Somayeh Kafaie	11-09-2017 Date

Contents

List of Figures

List of Tables

List of Abbreviations

ACK ..... Acknowledgement

ARQ ..... Automatic Repeat Request

BER ..... Bit Error Rate

CBR ..... Constant Bit Rate

CSMA/CA ..... Carrier Sense Multiple Access with Collision Avoidance

CTS ..... Clear to Send

DCF ..... Distributed Coordination Function

DIFS ..... Distributed Inter-Frame Space

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

ETX ..... Expected Transmission Count

ExNT ..... Expected Number of Transmissions

ExOR ..... Extremely Opportunistic Routing

FlexONC ..... Flexible Opportunistic Network Coding

IANC ..... Intra-flow Network Coding

IXNC ..... Inter-flow Network Coding

MAC ..... Media Access Control

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

NACK ..... Negative Acknowledgement

NAV ..... Network Allocation Vector

NC ..... Network Coding

OR ..... Opportunistic Routing

PHY ..... Physical

PNC ..... Physical-layer Network Coding

RLNC ..... Random Linear Network Coding

RTS ..... Request to Send

SIFS ..... Short Inter-Frame Space

UDP ..... User Datagram Protocol

WMN ..... Wireless Mesh Network

WMSN ..... Wireless Multimedia Sensor Network

Chapter 1

Introduction

1.1 Background

Can you imagine living without your smartphone or other wireless communication devices? Most of us find it hard. Wireless technology has a significant impact on modern society, and now has become an essential part of daily life of many people around the world. It has impacted the world in many important ways from health care and education to business, news reporting and entertainment, mostly by mobility, introducing smartphones with advanced capabilities, Internet connectivity and improving the distribution of information.

Wireless mesh network is a type of wireless communication networks aiming to realize the dream of a seamlessly connected world. In mesh infrastructure, radio nodes are connected via wireless links creating a multi-hop wireless network in which nodes can talk to each other and pass data over long distances by dividing the path to smaller segments and handing off data over mulitple hops. This cooperative data delivery is the

key idea of mesh networks to share connectivity across a large area with inexpensive Wi-Fi technology.

Despite these advancements, users' expectations rise fast, and new applications require higher throughput. In addition, the performance of wireless networks is significantly restricted by the contention among different data flows and devices in sharing bandwidth and other network resources, interference, and the unreliability of the wireless channel. However, since the last decade two promising approaches of "Opportunistic Routing" and "Network Coding" are proved to improve the performance of wireless networks significantly by proactively utilizing the broadcast nature of the wireless medium.

Network coding, or more specifically Inter-flow Network Coding (IXNC), increases the throughput by forwarding more than one packet in each transmission, and thus increasing the "effective" capacity of the network [54]. In recent years, a significant amount of research has been conducted to explore the effect of network coding in different scenarios and improve the network performance. A survey on unicast, multicast and broadcast applications of network coding especially for wireless sensor networks can be found in [74].

Opportunistic routing [1] (OR) also benefits from the broadcast nature of wireless networks via path diversity. In OR, in contrast to traditional forwarding, there is no fixed route, and a packet forwarded by a node can be possibly received by any of its neighbors. In fact, a node first broadcasts the packet and then the next-hop is selected among all neighbors that have received the packet successfully. In addition, OR can reduce the total number of transmissions by exploiting long but low-quality links. Doing so, OR can largely increase the packet delivery probability and network throughput.

[1] Also called "opportunistic forwarding" in some research.

1.1.1 Network coding and its benefits

Network coding represents an effective idea introduced by Ahlswede *et al.* [2] in 2000 to increase the transmission capacity of a data communication network as well as its robustness. In general, two different types of network coding can be applied, namely intra-flow network coding (IANC) and inter-flow network coding (IXNC). Despite carrying similar names, the goals and challenges in these two types of network coding are quite different as IANC is used to improve the robustness and reliability of wireless networks, while IXNC is utilized to boost the capacity of the network.

IANC increases the robustness by generating and forwarding a random linear (RLNC) combination of the packets of the same flow. It is an efficient alternative to the hop-by-hop feedback mechanism used in traditional forwarding in order to achieve reliability in the network. By encoding packets originated from the same source, IANC makes all packets equally beneficial. Hence, it eliminates hop-by-hop feedback, and saves bandwidth.

This idea has received considerable attention from the research community and a significant amount of research has been conducted on IANC from both theoretical and practical points of view [21, 43, 46, 55, 67, 68, 100]. MORE (MAC-independent Opportunistic Routing and Encoding) [13] is one of the first methods that realizes this idea in practical wireless scenarios. For more details on IANC and encoding and decoding using linear network coding, we refer the readers to [18, 19, 45, 62, 99].

On the other hand, IXNC, which is the focus of this dissertation, is a coding scheme in which a node combines the packets of multiple flows together and sends them at the same time over the channel. The XOR operation is used to combine packets in IXNC. Therefore, by reducing the number of transmissions, it improves the throughput and

decreases the interference in wireless channel. In fact, in IXNC by utilizing the broadcast nature of wireless networks, the network can reach its maximum capacity and improve the performance [48]. Xie *et al.* provide a survey on IXNC for both reliable links and lossy links [101].

The research on IXNC in wireless networks was originally inspired by COPE [48]. One of the most understood examples showing the gain behind IXNC is the X-topology in Figure 1.1, where S_1 sends packet a to D_1, and S_2 sends packet b to D_2. The destinations are not in the transmission range of their corresponding source, and packets are delivered through an intermediate node N. Since D_1 and D_2 are able to overhear the packets of the other flow from its source, the relay node N mixes packets of two flows and sends their combination to the network. Doing so, network coding decreases the number of required transmissions to deliver packets to their final destinations and improves the performance.

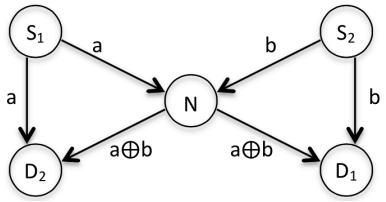


Figure 1.1: X-topology showing how IXNC improves throughput.

1.1.2 Opportunistic routing

OR is an effective idea to improve the performance of wireless networks, especially in lossy networks, by providing more chances for a packet to make progress toward the destination. In contrast to traditional forwarding in which the packets are forwarded along a fixed path, OR picks the next-hop of each packet only after that the packet has

been forwarded.

The idea behind OR, mostly recognized by ExOR (Extremely Opportunistic Routing) [8], is that the route which packets traverse is not predetermined and can be different for each packet of the flow. In fact, the source selects a set of nodes (i.e., called forwarder set) which are closer to the destination than itself, and for each packet the node closest to the destination that receives the packet will forward it toward the destination. The nodes in the forwarder set are ordered based on a metric such as hop-count, geo-distance or ETX (Expected Transmission Count) [17], which is the expected number of transmissions for a packet.

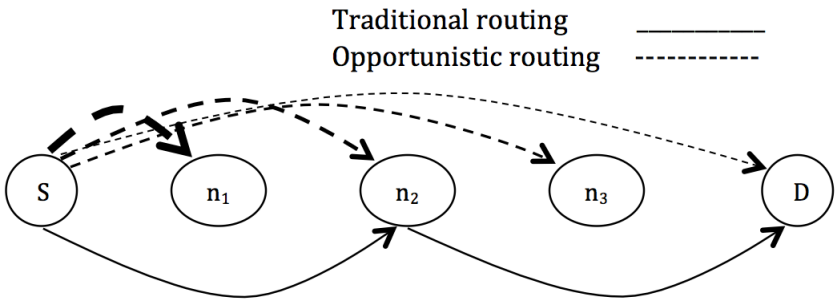


Figure 1.2: Opportunistic routing.

We use Figure 1.2 to elaborate on how OR works. Let us assume that node S wants to deliver its packets to node D in multi-hop wireless networks. In traditional scheme, the routing protocol will select the shortest path from the source to the destination. Let us further assume node n_2 is chosen as the only intermediate node to forward S's packets. Therefore, if n_2 does not receive a packet, S has to retransmit it. Note that not only n_2 but also n_1, n_3 and D might have received the packet, but with different probabilities which is usually related to their distance from the source. Thus, in OR, D, n_3, n_2 and n_1

are selected in the forwarder set, and as long as one of them receives the packet, S does not need to retransmit it.

Since multiple nodes are candidates of receiving and forwarding a packet, one needs to coordinate the intermediate nodes and prevent forwarding multiple copies of the same packet. Therefore, the most important steps in OR are routing metric selection, forwarder set determination, and forwarder set coordination [10]. Survey of OR protocols can be found in [10, 11, 14, 70].

1.2 Research Motivation and Challenges

1.2.1 Performance analysis of network coding

A variety of studies have explored the effectiveness of network coding, and more specifically IXNC, in different scenarios. However, most of them show the advantage of network coding experimentally, or analytically but without considering physical (PHY) layer or Media Access Control (MAC) layer specifications. Hence, there are few analytical studies on throughput and end-to-end delay of network coding in multi-hop wireless networks that incorporate the specifications of IEEE 802.11 Distributed Coordination Function (DCF).

In many previous mathematical studies, a simple topology is considered, where the source and destination are only one or two hops apart. Some studies are designed for saturated queues, where each node always has a packet to transmit that would cause an infinite delay. In addition, the theoretical research on multi-hop networks with unsaturated queues usually considers simplifying assumptions, such as conflict-free scheduled access, no interference, no collision, or no back-off. Furthermore, in most studies on this

subject only the throughput of the network is investigated, and also they postpone the transmission of native packets for the sake of providing more coding opportunities (i.e., opportunistic coding is not taken into account).

Therefore, more theoretical studies are needed to better quantify the benefits of network coding over traditional forwarding for actual protocols considering PHY/MAC layer specifications. Indeed, such theoretical analysis can be an important building block toward modeling more general scenarios.

In this dissertation, we utilize queuing theory to propose an analytical framework for bidirectional unicast flows in multi-hop wireless mesh networks, and study the throughput and end-to-end delay of IXNC with opportunistic coding (i.e., if there is no coding opportunity, native packets are sent without any artificial delay). Our model considers the IEEE 802.11 standard with Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA) random access and exponential back-off time with clock freezing and virtual carrier sensing. We formulate several parameters such as the probability of successful transmission in terms of bit error rate and collision probability, waiting time of packets at nodes, and retransmission mechanism.

1.2.2 Joint network coding and opportunistic forwarding

As stated earlier, COPE is one of the prominent examples of IXNC. However, coding opportunities in COPE are restricted only to joint nodes that receive packets from multiple flows. On one hand, to provide more coding opportunities, COPE needs more packets to arrive at the same node. On the other hand, this traffic concentration may overload coding nodes (i.e., the node that encodes packets of multiple flows together), and cause

faster energy drainage, longer delay, buffer overflow, and channel contention.

Furthermore, IXNC is not applicable in lossy environments since the accuracy of the coding node's estimate of next hops' decoding ability decreases as the loss rate in the network increases. Due to this problem, COPE turns off network coding if the loss rate in the network is higher than a threshold (i.e., the default value is 20% in COPE's implementation) [48].

To illustrate the issue, let us assume D_2, in Figure 1.1, cannot overhear a considerable number of sent packets from S_1 due to the loss of the link between S_1 and D_2. As explained earlier, N encodes the received packets of S_1 and S_2 together. However, due to the loss of overhearing link, D_2 cannot decode some received coded packets. In fact, D_2 cannot decode some packets like b because it was not able to overhear corresponding packet a. In addition, in a poor-quality channel, the reception reports (i.e., control packets sent by each node to advertise its packet repository to its neighbors) are lost easily, which makes encoding decisions more difficult. Although a few studies have been conducted to make IXNC efficient in lossy environments [81], usually they are not as practical due to their computational complexity.

On the other hand, OR scatters the packets of a flow over different paths, and enlists multiple paths from the source to the destination. In fact, by selecting more than one next-hop, OR provides more chances for a packet to make progress toward the destination, and can largely reduce the number of required transmissions and increase throughput, especially in lossy environments. In addition, in OR the packet is first broadcast and then the next-hop will be decided based on a metric. This metric, which prioritizes the possible forwarders of a packet, can also be chosen so that the forwarder with more coding opportunities is selected in each transmission. Doing so, we can provide more

coding opportunities in the network without forcing flows to meet at a joint node and causing channel contention, as described before.

Now, given that the benefits of network coding and opportunistic routing in wireless networks are clear, how to boost the performance even more by combining this two great ideas? How to select a routing metric and a forwarder set prioritization mechanism for OR such that network coding recognizes more coding opportunities in the network leading to an even higher throughput?

We believe that this combination, if realized carefully, would enable further improvement in the performance. Therefore, in this dissertation, to better utilize the broadcast nature of wireless networks, we introduce Flexible and Opportunistic Network Coding (FlexONC), which provides more flexibility to previous IXNC methods like COPE by adding OR. FlexONC, as a MAC layer solution, not only better captures the coding opportunities in the network, but also allows to control effectively how far packets stray away from a designated shortest path.

1.2.3 Coding conditions

To ensure that encoded packets are decodable at the next-hop, IXNC applies a set of coding conditions. In general, if packets P_1 and P_2 are to be encoded at a node, the node needs to verify that the next-hop of P_1 has already received P_2, and vice versa. To let the neighbor nodes know about the received packets, each node sends reception reports to its neighbors periodically or piggy-backed on data packets. In addition, if the link quality between nodes is higher than a threshold, a node can combine two packets if the next-hop of each packet is the previous-hop of the other packet or one of the neighbors

of the previous-hop. For example, in Figure 1.1, N can combine packets a and b because the next-hop of a (i.e. D_1) is a neighbor of the previous-hop of b (i.e., S_2), and vice versa. Therefore, N knows that with a high probability D_1 and D_2 have already overheard b and a, respectively.

However, as explained in this dissertation, we noticed that these coding conditions may decide erroneously to mix some packets that cannot be decoded at the next-hops. Overly-optimistic encoding of the packets, which leads to failure in decoding, reduces network performance because a larger number of retransmissions are required to deliver packets to their final destination. Therefore, we believe that a correct set of coding conditions are required to avoid incorrect packet encoding. To do so, in this dissertation, we propose an additional coding condition to be added to the current coding conditions. Also, we design a method to ensure that the new set of conditions works perfectly in different scenarios and finds coding opportunities accurately.

1.3 Research Contributions

This dissertation presents the following novel contributions to the inter-flow network coding area

- Applying the multi-class queuing network to study the performance of IXNC with opportunistic coding in multi-hop wireless mesh networks with bidirectional unicast flows

- Providing an analytical framework to study not only the throughput but also the end-to-end delay of both traditional forwarding and network coding

- Taking into account PHY/MAC layer specifications, and applying IEEE 802.11 Distributed Coordination Function (DCF) with random medium access CSMA/CA, while considering retransmission and the binary exponential back-off mechanism with clock freezing and virtual carrier sensing
- Verifying the validity of the analytical model by computer simulation in NS-2
- Proposing FlexONC as a joint IXNC and OR method to improve network performance
- Discovering the coding condition problem and proposing more intelligent and comprehensive encoding decisions to avoid transmitting undecodable packets in the network
- Evaluating the performance of FlexONC using simulation in NS-2
- Comparing FlexONC with other baselines in aspects such as throughput, end-to-end delay, the number of duplicate packets, the number of coding opportunities, overall overhead and complexity.

1.4 Thesis Outline

The rest of this dissertation is organized as follows. Chapter 2 provides a review of the analytical studies on IXNC as well as a comprehensive survey of research combining IXNC and OR, highlighting the fundamental components, challenges and the performance of each method. By comparing existing studies in the subject, we lay the groundwork for further research in next chapters.

In Chapter 3, an analytical framework is provided to study the performance of network coding (IXNC) in multi-hop wireless mesh networks with bidirectional unicast flows considering the specifications of IEEE 802.11 DCF with random medium access CSMA/CA, binary exponential back-off mechanism and opportunistic coding. The multi-class queuing network with stable queues separating native and coded packets is applied to calculate the throughput and an upper-bound of average end-to-end delay of the network.

After studying the performance of network coding, Chapter 4 presents FlexONC, a joint network coding and OR approach. In FlexONC, while packets travel around the shortest path, OR helps consider a union of the packets of the neighborhood to create coding opportunities, and improve the throughput of the network.

In Chapter 5, we describe an issue regarding coding conditions in IXNC methods, and show that it can cause a large number of packet drops in some scenarios. We address this problem by proposing an additional coding condition and a method to merge it with the current coding conditions.

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

In recent years, many studies have been conducted to explore the effect of network coding in different scenarios and improve the network performance by mixing packets in intermediate nodes before forwarding. There have been many experimental studies on network coding but much fewer mathematical analyses. In this chapter, we review research conducted on performance analysis of inter-flow network coding (IXNC) in wireless networks from a theoretical point of view, and discuss challenges of former analytical models of network coding. Furthermore, we describe related work aiming to capture more coding opportunities in the network and improve the performance by integrating IXNC and opportunistic routing (OR) in multi-hop wireless mesh networks, and discuss their limitation and drawbacks.

2.1 Analytical Model of Network Coding

Network coding represents an innovative idea introduced by Ahlswede *et al.* [2] in 2000 to increase the transmission capacity of the network, as well as its robustness. Prior

mathematical studies on network coding usually consider a simple topology. Most of them study the performance for a two-way relay [3, 40, 76], or derive some analytical bounds for a single relay in a two-hop region, where multiple sources initiate unicast sessions to multiple destinations [5, 59, 66, 104]. In particular, Amerimehr and Ashtiani [3] study the throughput and delay of a two-way relay by adopting frequency division duplexing (FDD). Without focusing on PHY/MAC layer constraints, they compare the throughput and delay in the relay for two cases, where 1) the relay postpones transmission of native packets, and 2) native packets are sent immediately.

Sagduyu *et al.* study the stable throughput when one or two sources broadcast their packets to two destinations [85] or more [86] via independent channels. Paschos *et al.* [76] study a two-way relay in IXNC taking into account overhearing, where coding decisions at the relay are either stochastic or deterministic via receiving overhearing reports. Moghadam and Li [72, 73] study the maximum stable throughput in single-hop wireless networks, where a source multicasts data packets to several destinations directly, and network coding is applied to retransmit the packets not received by a subset of the destinations.

In addition, Jamali *et al.* propose a dynamic scheduling based on a threshold on the amount of information at nodes' transmission buffers in bidirectional relay networks. This scheduling is used to maximize throughput both without any constraint on the delay [39], and with constraint to guarantee a certain average delay [40]. Furthermore, Umehara et al. [94] analyze the throughput and delay of network coding in two-hop networks with two unbalanced traffic cases (i.e., one-to-one and one-to-many bidirectional relay) employing slotted ALOHA. They also extend the model to single-relay multi-user wireless networks and provide the achievable region in throughput [93].

In another single-relay research, Lin et al. [64] study the throughput of network-layer and physical-layer network coding under IEEE 802.11 DCF with two groups of nodes communicating with each other via a relay node. In a similar work, where again all nodes are in carrier sensing range of each other, they not only study the throughput under slotted ALOHA but also propose a hybrid network coding scheme (i.e., a combination of physical-layer and network-layer network coding) to improve performance [66].

Regarding multi-hop wireless networks, Sagduyu *et al.* [87] consider a collision-free scheduled access to formulate throughput for both saturated and non-saturated queues. However, in the case of a random access scheme, their analytical model is limited to saturated queues. In a similar theoretical-based approach, for multicast sessions, Amerimehr *et al.* [4] derive throughput for multi-hop wireless networks. They also define a new metric, *network unbalance ratio*, which identifies the amount of imbalance in stability among nodes. However, their estimate of service time does not take into account some important features of IEEE 802.11 DCF like binary exponential random back-off. Furthermore, they postpone transmission of the native packet at a node until receiving a packet from another flow to be combined with it, and thus, causing a long delay.

In another work considering IEEE 802.11 DCF, Lin and Fu [65] investigate the throughput capacity of physical-layer network coding in which a common center node exchanges packets with others in multi-hop wireless networks. They analyze such canonical networks both with equal and variable link-length, and find the optimal number of hops to maximize the throughput. In addition, Ko and Kim [53] study the throughput and end-to-end delay of multi-hop wireless networks utilizing IEEE 802.11 DCF only for traditional forwarding, when every node initiates a flow with the same packet generation rate to a random destination. They derive a delay-constrained capacity in terms of carrier

Table 2.1: Overview of the analytical research in the literature.

Reference	Network coding	Number of hops	Throughput	Delay	Stable queues	Random access	Unicast /multidcast	Opportunistic coding	Exponential back-off
[85, 86]	√	1	√	-	√	√	multicast	-	-
[72, 73]	√	1	√	-	√	√	multicast	√	-
[76]	√	2	√	-	√	-	unicast	-	-
[3]	√	2	√	relay	√	-	unicast	both	-
[40]	√	2	√	√	√	√	unicast	-	-
[5, 66, 104]	√	2	√	-	√	√	unicast	-	-
[59]	√	2	√	-	√	priority/ equal access	unicast	both	-
[64]	√	2	√	-	√	√	unicast	-	√
[93, 94]	√	2	√	√	√	√	unicast	√	-
[53]	-	≥ 3	√	√	√	√	multicast	not applicable	√
[87]	√	≥ 3	√	-	-	√	multicast	-	-
[4]	√	≥ 3	√	-	√	√	multicast	-	-
[65]	√	≥ 3	√	-	-	√	unicast	-	-
[33]	√	≥ 3	√	-	-	√	unicast	-	√

sensing range and packet generation rate.

Furthermore, Hwang *et al.* [33] propose an analytical framework for bidirectional unicast flows in multi-hop wireless networks. Their work considers collision and different interference levels in CSMA/CA by varying the carrier-sensing range and signal-to-interference ratio to maximize the throughput in different retransmission schemes.

Table 2.1 presents an overview of all studies discussed in this section.

2.2 Joint Inter-Flow Network Coding and Opportunistic Routing

2.2.1 Motivation and benefits

Both IXNC and OR improve the performance of wireless networks by exploiting the broadcast nature of the wireless medium. In IXNC packets of different flows are XORed at intermediate nodes so that each transmission piggybacks multiple packets. Therefore, by reducing the number of transmissions, IXNC boosts the capacity of the network and improves the network performance. OR is another innovative idea to increase the robustness and reliability of wireless networks. It chooses more than one potential forwarder for each packet forwarding, which reduces the number of retransmissions as well as the number of required hops to deliver a packet to the final destination, leading to a smaller total packet transmission count and higher throughput.

Despite sharing the similar goal of improving the performance by utilizing the broadcast nature of wireless networks, IXNC and OR have different applications and address separate challenges. By leveraging multiple potential next-hops, OR is mostly suitable for lossy environments with medium- to low-quality links between nodes, where selecting a single next-hop causes several packet losses and retransmissions. On the other hand, IXNC is mostly effective in reliable networks with high-quality links, where nodes can rely on packet overhearing to decode received coded packets.

Furthermore, in OR protocols like MORE as the number of flows increases, the throughput gain of the protocol decreases. On the other hand, in IXNC methods like COPE the more flows crossing at the same node, the more coding opportunities exist.

For example, let us assume that in the cross topology depicted in Figure 2.1, for each node all other nodes are in its transmission range except for the diametrically opposite one, and that n_1, n_3, n_4 and n_5 are the sources of 4 flows intersecting at n_2. Then, n_2 can mix 4 packets received from all sources because each next-hop contains all other coding partners except for its intended packet.

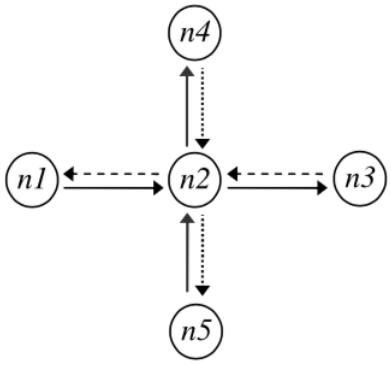


Figure 2.1: Cross topology with 4 flows intersecting at n_2 [48].

However, this traffic concentration can turn intermediate nodes to bottlenecks causing issues such as faster energy drainage, longer end-to-end delay and higher channel contention. As a matter of fact, the improvement of throughput in protocols like COPE depends on the traffic pattern. They limit coding opportunities because coding can be performed only at joint nodes. As an example, if in Figure 2.1 the sources choose a different intermediate node than n_2, all flows cannot intersect at the same node and fewer coding opportunities are provided by COPE.

Looking at pros and cons of OR and IXNC, one can think of them as two complementing techniques. OR can improve the performance of IXNC in lossy networks and increase the number of coding opportunities by considering the packets of the neighbor-

hood collectively instead of forcing flows to cross at focal nodes. In addition, by providing free-ride for packets of several flows, IXNC can improve the performance of OR in the presence of multiple flows.

Therefore, the research question is how we can combine OR and IXNC such that their integration outperforms each of them individually in different scenarios. To realize such a powerful joint approach, the following challenges should be addressed.

- Choosing an appropriate routing metric to determine the set of forwarders, considering the specifications of both OR and IXNC.
- Recognizing coding opportunities and selecting the right packets to be coded together.
- Prioritizing the candidates in the forwarder set and selecting the best one.
- Coordinating the forwarder set and suppressing duplicate packets in the network.

2.2.2 Taxonomy of joint protocols

As discussed before, to develop an effective joint OR and IXNC approach some issues need to be addressed. In this section, some important components of both IXNC and OR and their realization in different protocols are discussed. A summary of classification of joint IXNC and OR protocols is provided in Table 2.2.

2.2.2.1 Routing metric

A routing metric is used in OR protocols to determine and rank the nodes in the forwarder set. The main purpose of OR is to reduce the expected number of transmissions (ExNT)

required to deliver packets to their final destination, leading to a shorter end-to-end delay and higher throughput [10]. Therefore, it is very important to choose the right nodes for the forwarder set, and prioritize them in an efficient way. This necessitates having an appropriate routing metric, which is even more critical in joint IXNC and OR approaches as not only ExNT but also the number of coding opportunities must be taken into account.

2.2.2.2 Forwarder set coordination method

By selecting more than one potential forwarder, OR provides more chances for a packet to progress toward the destination. However, in each transmission, only one of the nodes in the forwarder set (i.e., the node with the highest priority that has received the packet) should forward the packet, and other nodes should discard it. Otherwise, there would be many duplicate packets in the network degrading its performance. Therefore, it is crucial to have an effective method to coordinate the nodes in the forwarder set such that they can agree on the next forwarder among themselves and avoid duplicate transmissions.

To deal with duplicate packets, most joint IXNC and OR protocols apply a strict scheduling to coordinate forwarders. Each node sets a forwarding timer according to its priority in the forwarder set, and transmits the packet after timer expiration unless it receives a signal from a higher priority node indicating the transmission of the packet. The signaling solutions are either data-based or control-based [10]. In data-based methods, the nodes in the forwarder set cancel their transmission after overhearing the transmission of the same data packet by a higher-priority node, while in control-based approach the higher priority node sends a control packet (e.g., ACK or probe) to notify others about receiving the packet. Furthermore, in some studies, intra-flow network coding (IANC) is incorporated with OR to tackle the forwarder coordination problem by making packets

equally beneficial to the destination through RLNC.

2.2.2.3 Forwarder set selection strategy

The selection of the nodes in the forwarder set can be either end-to-end or hop-by-hop [34,70]. In end-to-end forwarder set selection, the set of potential forwarders is determined by the source once for the whole path toward the destination. On the other hand, in hop-by-hop forwarder set selection, each node determines the forwarder set toward the destination independently.

While an end-to-end approach covers a broader area and provides more chances for a packet to progress, its overhead is higher and its implementation is harder than a hop-by-hop strategy. Also, the coordination among forwarders is more difficult in an end-to-end strategy, and can cause duplicate transmissions as some nodes may not overhear each other. However, end-to-end approach usually outperforms hop-by-hop approach capitalizing on more network state information [70].

2.2.2.4 Coding region

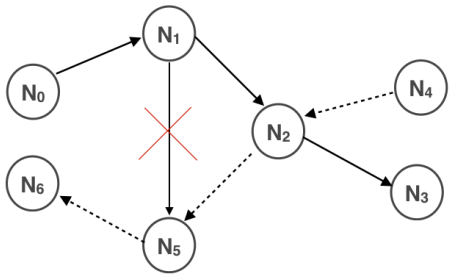


Figure 2.2: Coding opportunities beyond a two-hop region

The majority of research on IXNC is limited to a two-hop region. This means a node

will encode a packet with other packets if the next hop of the packet is known to be able to decode it. In fact, if the next hop cannot decode the packet, it will drop the packet. However, in some topologies, even if the next hop can not decode the packet immediately, it could still forward the packet as coded, and another node in down stream will be able to perform decoding successfully. We use the scenario depicted in Figure 2.2 to demonstrate the idea. In this figure, N_0 and N_4 send their packets to N_3 and N_6, respectively. In a two-hop region, IXNC cannot find any coding opportunity in such a topology because if N_2 encodes packets of these two flows, N_5, which is the next hop of the packets of the second flow (i.e., that from N_4 to N_6), cannot overhear any packet from the first flow and cannot decode the packet. However, if nodes have access to information about the network topology and the route of the flows, IXNC protocols will be able to capture such coding opportunities and benefit from them.

2.2.2.5 Coding strategy

In IXNC methods, coded packets are either generated when a transmission opportunity is available (i.e., on-demand) or beforehand (i.e., prepared). In an on-demand approach, all packets are stored as native packets in the forwarding queue, and when there is a transmission opportunity, the node chooses the native packet at the head of queue, encodes and transmits it. This approach may capture more coding opportunities (i.e., coded packets with more coding partners) as the coding decisions are based on the latest information before transmission. In a prepared approach, to avoid delaying a transmission for searching queues and finding coding opportunities, as soon as a new packet arrives at the forwarding queue, the node mixes it with other packets if there is any coding opportunity.

Table 2.2: Taxonomy of joint OR and IXNC protocols.

Protocol	Routing metric	Coordination method	Coding region	Max flows mixed	Opportunistic coding	Coding strategy	ACK strategy	Forwarder set selection
XCOR [54]	ETX	Timer, data-based	Two-hop	Multi	Yes	On-demand, max utility gain by checking flows in desc. order of Q's length	Reception reports	End-to-end
CAOR [102]	ETX, coding partners	Timer, data-based	Two-hop	Multi	-	On-demand, comb. of first k packets	Reception reports	Hop-by-hop
ANCHOR [41]	No. of transmissions	Notification, data-based	Two-hop	Multi	No	Prepared, greedy based on fewest trans.	-	-
CORMEN [36]	ETX, coding partners	Timer, control-based	Two-hop	Multi	Yes	Prepared	End-to end	Hop-by-hop
CORE [103]	No. of receivers, geo. dist.	Timer, data-based	Two-hop	Multi	No	On-demand, comb. of first k packets	-	Hop-by-hop
BEND [106]	Coding partners	Timer, data-based	Two-hop	Multi	Yes	Prepared, greedy	Hop-by hop	Hop-by-hop
O3 [27]	ETX	IANC	Two-hop	Two	No	On-demand, greedy	End-to end	End-to-end
AONC [90]	Maximum space utilization	Timer	Tow-hop	Multi	Yes	Prepared, maximizing space utilization	Hop-by hop	Hop-by-hop
CAOR [16]	ETX	IANC	Two-hop	Multi	-	-	-	-
CoAOR [31]	ETX, coding partners	Timer, data-based	Tow-hop	Multi	Yes	On-demand, comb. of first k packets	Reception reports	Hop-by-hop
HCOR [26]	Anypath cost	-	Two-hop + destination	Two	Yes	Prepared, greedy	-	Hop-by-hop + destination
INCOR [109]	CETX	Timer, control-based	Two-hop	Two	Yes	On-demand, checking first k packets	Hop-by hop	Hop-by-hop
CAR [69]	coding partners, geo. dist.	Timer, data-based	Two-hop	Multi	No	Prepared, greedy grouping flows	-	Hop-by-hop

2.2.3 Comparison of proposed joint protocols

The possibility of combining OR and IXNC was first discussed in [47], where a preliminary version of COPE was introduced as well. However, the results suggested that the benefit of combining these two techniques is not notable, and even duplicate packets can degrade the network performance in some scenarios. In that early research, forwarders are prioritized based on their distance from the destination, and coding opportunities are not taken into

account. Also, the coordination among forwarders is not discussed in details.

From one perspective, the research that reflects the advantages of combining IXNC and OR can be classified as those with IANC [16, 27] and those without IANC [26, 31, 36, 41, 54, 69, 90, 102, 103, 106, 109]. CAOR (Coding-Aware Opportunistic Routing) [16] is one of those few studies that utilize IANC as the coordination method of opportunistic routing in realizing the joint approach. In each transmission, CAOR combines the packets of flows that maximize a metric, which is defined in terms of the progress of the packet in each transmission (based on ETX) and the probability that the next-hops will receive the coded packet and decode it. However, the throughput gain of CAOR is relatively smaller than that of the other joint methods [69] especially because combining IANC and IXNC reduces the number of coding opportunities in the network.

O3 (Optimized Overlay-based Opportunistic routing) [27] is another approach that exploits IANC in integration of OR and IXNC, where packets of two flows can be mixed. In O3, an overlay network performs overlay routing, IANC and IXNC, while in the underlay network OR is applied, and an optimization problem is solved to find the desirable sending rates for IANC and IXNC packets. Using Qualnet simulation, the results show that O3 outperforms shortest path routing, COPE and MORE. Note that while in regular IANC, only the final destination needs to decode RLNC packets, joint approaches discussed here [16,27] impose more overhead because all intermediate nodes need to apply Gaussian elimination and decode RLNC packets (to decode IXNC packets).

One of the first studies on joint OR and IXNC is XCOR (Interflow NC with Opportunistic Routing) [54], whose OR component has been inspired by SOAR (Simple Opportunistic Adaptive Routing) [83]. In XCOR, the forwarder set, which forms a "thin belt" along the shortest path, is calculated recursively for each next-hop by the source

and stored in the packet header. Also, the forwarders are prioritized based on their closeness to the destination in terms of ETX. Before forwarding a received packet, the forwarders start a timer according to their priority, and cancel the packet transmission if they overhear it from a higher-priority node. To find the best coded packet at each node, XCOR defines a utility function as the sum of the utility gain of the next-hops, which is calculated in terms of the progress toward the destination, the probability of successful transmission to the next-hop, and the probability of successful decoding at the next-hops. Applying a heuristic algorithm, they rank flows in terms of the length of their queues, and mix the packet at the head of the longest one with the packet at the head of other flow's queues if this combination increases the utility gain. In the evaluation of XCOR in Qualnet, two simple topologies (i.e., a hexagon topology and a chain topology with 4 nodes) are considered, and its performance degrades considerably in lightly loaded or lossy environments [16].

In another method called CAOR [102], the nodes in the forwarder set are neighbors of the sender closer to the destination than the sender (in terms of ETX) that can mutually overhear each other. To find the higher-priority forwarder with most coding opportunities, nodes exchange reception reports advertising not only their own stored packets but also their neighbors' packets. Doing so, all nodes in the forwarder set can compute available coding opportunities in each other and will know which one of them is the best forwarder for this particular transmission. Also to compensate for lost or delayed reception reports, each node guesses about packets it would receive; if a node has received M consecutive packets of a flow, it can report the next two packets of that flow in its current reception report.

In CORMEN (Coding-aware Opportunistic Routing in wireless Mesh Network) [36],

as an IXNC scheme enhanced with OR, the nodes in the forwarder set should have a good quality link with the sender (in terms of ETX), and the ETX between any pair of them is within a threshold. Also, to avoid diverging the path and unnecessary duplicate packets, the nodes in the forwarder set are neighbors of the nodes in the shortest path. In CORMEN, end-to-end acknowledgments are sent instead of hop-by-hop ones, and each forwarder starts a forwarding timer in terms of ETX and the maximum number of flows that can be mixed in a coded packet. Similar to source routing protocols, the packet header should contain not only the forwarder set but also the nodes on the shortest path. In addition, since the packet may not follow the shortest path, the forwarders need to keep updating the path.

ANCHOR (Active Network Coding High-throughput Optimizing Routing) [41] is another method in which packets carry the shortest path information. By exploiting coding opportunities, ANCHOR actively updates the route, which has been embedded in the packet header. Based on reception reports, if a node other than the next-hop of the packet can provide more coding opportunities, it notifies the other nodes to update the route. Simulation results in Glomosim show that ANCHOR performs better than COPE and DSR [42] in a number of scenarios.

In another work, considering geographic distance as the routing metric instead of ETX, CORE (Coding-aware Opportunistic Routing) [103] selects the forwarder set from the neighbors of the sender which are geographically closer to the destination than itself. The main components of CORE are forwarder set selection, coding opportunity calculation, primary forwarder selection (i.e., calculating local coding opportunities by each node), and priority-based forwarding (i.e., using timers to coordinate nodes). In each transmission, among all nodes in the forwarder set, CORE selects the node with the most

coding gain as the next forwarder. To prioritize nodes with different coding opportunities, forwarding timers are used so that the node with more coding opportunities forwards its packet earlier. In addition, in CORE each packet carries the location of the sender and the destination, and the packets are broadcasted without any acknowledgment or retransmission mechanism. To forward a packet at the head of queue, CORE picks the next k packets as seeds for possible encoding, and chooses the one that maximizes the coding gain.

While CORE defines the coding gain function at each node in terms of the number of candidates in the forwarder set that are able to decode a coded packet, CoAOR [31] takes into account the number of flows coded in a packet, the link quality and the number of nodes that are able to encode and decode packets as well. Analytic Hierarchy Process (AHP) [84] is applied to find the weight of these parameters. Three main components of CoAOR are coding-aware forwarder set selection, node coding gain calculation, and priority-based packet forwarding. The candidates in the forwarder set are selected from the neighbors of the sender closer to the destination than the sender itself (in terms of ETX), which are able to overhear each other. They coordinate among themselves using a forwarding timer inversely proportional to their coding gain.

In another study focusing on wireless multimedia sensor networks (WMSN), AONC (Adaptive Opportunistic Network Coding) [90] improves the transmission quality of video stream. Given that video packets have variable lengths, AONC might send more than one packet of a flow in each transmission. In fact, to maximize the forwarded length, it splices packets of the same flow as long as the spliced packet's length is less than the space length limit. Then, the spliced packets of different flows could be mixed using IXNC. Their optimization algorithm is repeated for different coding groups and different

possible space length limits, and finally the one with maximum space utilization is selected. Although this method reduces the number of required transmissions, it intensifies the packet reordering problem of OR.

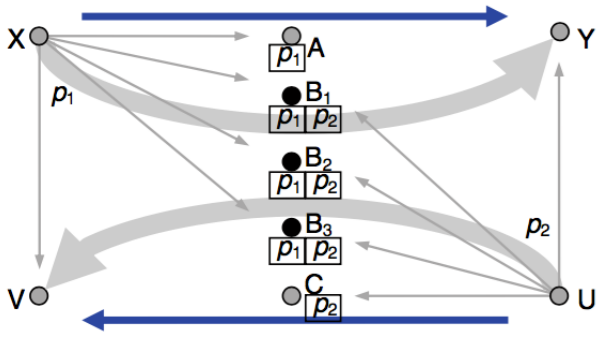


Figure 2.3: Diffusion gain in BEND [106].

BEND [106], as another advancement of COPE, introduces a type of gain, referred to as the *diffusion gain*, which is the benefit of being able to scatter flows through multiple forwarders dynamically. To avoid traffic concentration in BEND, a non-intended forwarder (i.e., the receivers of the packet which are not specified as the next-hop on the route defined by the routing protocol, and can help in forwarding) may receive a native packet and mix and forward it on behalf of the intended forwarder (i.e., the next-hop designated by the routing protocol). For example in Figure 2.3, where nodes A and C are the intended forwarders of the flows from X to Y and from U to V, respectively, COPE cannot find any coding opportunity. On the other hand, BEND allows non-intended forwarders which can overhear packets of both flows (e.g., B_1, B_2 and B_3) to combine and forward the packets on behalf of the intended forwarders. To do so, a *second-next-hop* field is included in native packets. As such, when a non-intended forwarder receives a native packet, it can find the address of the next-hop in the *second-next-hop* field. How-

ever in BEND, OR cannot be applied to two consecutive hops, and for coded packets, the *second-next-hop* field does not present the correct address on the shortest path such that the packets can travel around the shortest route. Therefore, non-intended forwarders drop coded packets since they do not know the address of the next-hop from the intended forwarder to the destination.

CAR (Coding-Aware opportunistic Routing) [69] is another joint scheme that aims to maximize the number of native packets coded together in a single transmission by dynamically selecting the route based on real-time coding opportunities. Regarding encoding, CAR keeps a set of coding groups representing the flows that can be potentially coded together. In CAR, each node knows the geographic position of all other nodes in the network, and the nodes in the forwarder set are neighbors of the sender that 1) their hop-count to the destination is less than or equal to the sender, and 2) are closer to the destination than the sender (in terms of geographic distance). Each node sets the forwarding timer inversely proportional to the number of native packets in a coded packet, and nodes cancel their transmission after overhearing the same packet from another node in the forwarder set. Also, native packets are only sent by the next-hop designated by the shortest path routing. For TCP flows, ACK packets are sent along the shortest path and are coded only with themselves.

HCOR (High-throughput Coding-aware Opportunistic Routing) [26] is a distributed system based on anypath routing [57] claiming that maximizing coding opportunities does not necessarily improve the network performance. Since the forwarder set of a coded packet is a subset of the original forwarder set (i.e., the subset that can decode the coded packet), HCOR argues that sending coded packets is not always beneficial, and one may need to decide if network coding decreases the cost by free-riding or increases it by

shrinking the forwarder set. Therefore, they consider network coding gain as well as link qualities to find the path with minimal anypath cost, while at most two packets can be mixed in HCOR. They compare HCOR with anypath routing as well as a joint OR and IXNC approach that always encodes packets if there is any coding opportunity, referred to as COOR in their paper. The results show that HCOR outperforms other baselines in different scenarios by 10% to 30%.

INCOR (Inter-flow Network Coding-based Opportunistic Routing) [109] introduces a metric called Coding-based Expected Transmission Count (CETX) that computes the expected transmission count required to deliver a packet to a destination using IXNC. This metric is used to prioritize the nodes in the forwarder set when lower CETX means higher priority. To calculate CETX for all nodes, they run a centralized algorithm similar to the Dijkstra's algorithm based on the idea that adding the nearer neighbors of node i and their forwarder set (in terms of CETX) to the forwarder set of node i can reduce its CETX. Each packet carries its prioritized forwarder set, and the forwarders start a forwarder timer proportional to their priority, which will be canceled upon hearing an ACK from a higher-priority node.

Table 2.3 summarizes the comparison of mentioned joint protocols.

2.3 Summary

As explained in Section 2.1, there have been a number of studies on performance analysis of IXNC in multi-hop wireless mesh networks, and the focus of some of them is on saturated queues. Most research on this subject investigates only the throughput of the network. Also, they do not take into account the network performance under opportunistic coding.

Table 2.3: Comparison of joint OR and IXNC protocols.

Protocol	Principle idea	Simulation tool	Evaluation topology	TCP /UDP	Compared protocols	Throughput gain
XCOR [54]	One of the first methods showing benefits of the joint approach	Qualnet	4-node chain, hexagon	UDP	SOAR [83], COPE, Srcr [7]	By 115%, 34% and 13% in chain, and 75%, 22% and 70% in hexagon, it outperforms Srcr, SOAR and COPE
CAOR [102]	Coding gain by forwarding based on coding opportunity awareness	NS-2	200 nodes in $1000 \times 1000\ m^2$	UDP	COPE	On average 15% improvement
ANCHOR [41]	Optimizing route based on no. of transmissions	Glomosim	$100 - 700$ nodes in $800 \times 800\ m^2$	UDP	COPE, DSR [42]	outperforms COPE by up to 38%
CORMEN [36]	Finding the shortest coding possible path	NS-2	3×3, 5×3 and 5×5 grids	UDP	COPE	Outperforms COPE slightly
CORE [103]	Forwarding based on maximizing coding opportunities	NS-2	200 nodes in $800 \times 800\ m^2$	UDP	COPE, OR	On average 22% improvement
BEND [106]	Neighborhood coding repository around the shortest route	NS-2	Cross, 3-tier 5×5 grid	UDP	COPE, traditional routing	On average about 25% improvement over COPE
O3 [27]	Sending rate optimization in joint IANC, OR and IXNC	Qualnet	3-node chain, diamond, 5×5 grid, 25-node random, MIT Roofnet [71], UW testbed [82]	UDP	COPE, MORE, traditional routing	Significantly outperforms other baselines
AONC [90]	Better video transmission quality in WMSNs	NS-2	Cross, 3-tier, 5×5 grid	UDP	COPE, BEND, traditional routing	Higher quality video trans.
CAOR [16]	Joint OR, IANC and IXNC to increase coding opportunities	-	2-tier with 8 nodes, random	UDP	MORE	$20\% - 27\%$ improvement
CoAOR [31]	Coding gain based on link quality, coding partners and decoding ability	MATLAB	200 nodes in $1000 \times 1000\ m^2$	UDP	CAOR [102]	$2\% - 30\%$ improvement
HCOR [26]	Deciding between coding and native transmissions	NS-2	3-node chain, cross, hexagon, diamond, 50 nodes in $1000 \times 1000\ m^2$	UDP	anypath routing, COOR (HCOR without calculating IXNC cost)	$10\% - 30\%$ improvement
INCOR [109]	Forwarder set selection and prioritization based on CETX	-	5×5 grid, MIT Roofnet [71]	UDP	COPE, EAX [108]	Outperforms COPE and EAX , on average by 12% and 17%
CAR [69]	Maximizing the no. of native packets coded in each transmission	NS-2	Cross, 5×5 grid	Both	COPE, BEND	Cross, TCP: 43%(COPE), 36%(BEND). Cross, UDP: 34%(COPE), 15%(BEND)

Moreover in many cases, the source and destination are only one or two hops apart from each other, and even those considering multi-hop networks usually model network coding

with simplifying assumptions, such as conflict-free scheduled access, no interference, no collision, or no exponential back-off. Therefore, more studies on this subject are needed to analyze the performance of IXNC in multi-hop wireless networks for actual protocols considering PHY/MAC layer specifications and more realistic case of stable queues, and to take into account other performance metrics such as end-to-end delay in addition to throughput.

Furthermore, to improve the performance of IXNC especially under poor quality channels, its integration with OR seems promising as discussed in Section 2.2. In recent years, some methods have been proposed to combine IXNC and OR; however, most of them are yet to fully utilize the broadcast nature of wireless networks. In some described works, the closeness to the destination (i.e., to find the forwarder set) is calculated in terms of the geographic distance [69, 103], which does not necessarily represent the quality of the path. In addition, in most of the research in this area, the path traveled by the node can be excessively longer than the shortest path [31, 69, 102, 103, 109], which can increase the end-to-end delay and degrade the performance. Even those studies that take into account the length of the route and select the forwarder set from nodes around the shortest path cannot combine packets of more than two flows [26] or require the source to know the shortest path and embed it in the packet header [36,41,54]. More so, the majority of them either broadcast packets without any feedback or retransmission mechanism, or end-to-end acknowledgments are sent instead of hop-by-hop ones. Therefore, further research on the idea of integrating IXNC and OR is imperative to not only better capture the coding opportunities in the network, but also control effectively how far packets stray away from a designated shortest path, and finally further improve the network performance.

Chapter 3

Performance Analysis of Network Coding with IEEE 802.11 DCF in Multi-Hop Wireless Networks

Capacity is a crucial resource in multi-hop wireless networks because it is shared not only between the source and destination of data packets but also among relay nodes forwarding the packets. To increase the transmission capacity of wirless networks, the powerful concept of network coding [2] has been introduced, which can improve performance significantly in theory, without considering PHY/MAC layer constraints such as contention, collision and interference. However, network protocols inevitably deal with such physical phenomena and constraints. Therefore, more theoretical studies are needed to better quantify the benefits of network coding over traditional forwarding for actual protocols considering PHY/MAC layer specifications.

There have been many experimental studies on this subject, but much fewer math-

ematical analyses. Some previous theoretical studies are designed for saturated queues, where each node always has a packet to transmit that would cause an infinite delay. In many cases, researchers consider a simple topology, where source is one [72, 73, 85, 86] or two [3, 5, 40, 59, 66, 76, 104] hops away from the destination. Furthermore, the theoretical research on multi-hop networks [4, 33, 35, 44, 65, 87] usually models network coding with simplifying assumptions, such as conflict-free scheduled access, no interference, no collision, or no exponential back-off. Moreover, most research on this subject investigates only the throughput of the network, and postpones the transmission of native packets in favor of providing more coding opportunities (i.e., not applying opportunistic coding).

In this chapter, we provide an analytical framework based on multi-class queuing network to study the throughput and end-to-end delay of multi-hop wireless mesh networks applying IXNC [48], where packets of different sources are mixed by bitwise *XOR* operation. We apply random medium access CSMA/CA as in IEEE 802.11 Distributed Coordination Function (DCF) with binary exponential back-off considering clock freezing and virtual carrier sensing as explained in Sections 3.1.2 and 3.2.1.2. We formulate collision probability, successful transmission probability of links, service time at different nodes, feedback and retransmission mechanism, and coding probabilities.

We model a multi-hop chain topology with bidirectional unicast flows in opposite directions, where intermediate nodes can combine packets of two flows. In our model for packet forwarding process, opportunistic coding is used, which means a packet is sent natively if there is not any coding opportunity. In fact, in contrast to other analytical works, we do not postpone transmission of native packets artificially to generate coded packets. Also, we consider separate classes of queues for native and coded packets, while the coded queue is a higher-priority queue. We develop our analytical framework for

both non-coding and coding schemes considering packet retransmission, and formulate the throughput and an upper-bound of end-to-end delay in a stable network. Also, we verify our analytical model using simulations in NS-2.

The main contributions of our proposed analytical model are as follows:

1. We apply the multi-class queuing network to study the performance of network coding in multi-hop wireless mesh networks with bidirectional unicast flows, where in contrast to other studies no artificial delay is injected in forwarding native packets even if there is no coding opportunity.

2. This model provides a framework to study not only the throughput but also the end-to-end delay of traditional forwarding and network coding schemes in multi-hop wireless mesh networks.

3. The proposed model takes into account PHY/MAC layer specifications, and applies IEEE 802.11 DCF with random medium access CSMA/CA. We consider retransmission and the binary exponential back-off mechanism with clock freezing and virtual carrier sensing as well as collision and link qualities in calculating the throughput and an upper-bound of average end-to-end delay of the network.

4. The validity of the analytical model, which is constructed based on queuing theory, is shown by simulations in NS-2.

3.1 System Overview

Before further explanation of the model, let us summarize the symbols used in this chapter in Table 3.1.

Table 3.1: Notations.

Symbol	Description	Symbol	Description
δ	maximum propagation delay	γ_i	packet generation rate at the source N_i
λ_i	arrival rate at node N_i	μ	service rate of the queue
L_p	packet length	$\mathrm{CW}_{\min}$	minimum contention window
T_{data}	transmission time of a packet	T_{ack}	transmission time of an acknowledgement
θ	throughput	T_{trans}	$T_{\mathrm{data}} + T_{\mathrm{ack}} + \mathrm{SIFS}$
$T_{\mathrm{backoff}}(m)$	mean of back-off distribution in m^{th} transmission	$T_{\mathrm{counter}}(m)$	$\mathrm{DIFS} + T_{\mathrm{backoff}}(m)$
W	upper-bound of the average end-to-end delay	$T_s(m)$	service time at the m^{th} transmission of a packet
$\lambda_{in,i}^{n(j)}$	arrival rate of native packets of the j^{th} flow at N_i	$\lambda_{out,i}^{n(j)}$	output rate of native packets of the j^{th} flow at N_i
$\lambda_{in,i}^{c(j)}$	arrival rate of coded packets of the j^{th} flow at N_i	$\lambda_{out,i}^{c}$	output rate of coded packets at N_i
$\lambda_i^{n(j)}$	arrival rate of the j^{th} flow in Q^n of N_i	λ_i^c	arrival rate in the coded queue of N_i
$W(Q)$	average waiting time in queue Q	W_{system}	average waiting time in the queuing system
$\mu_i^{n,\mathrm{seen}}$	service time seen by lower priority queue, Q^n	$\pi_0(Q^{n(r)})$	probability of having no packets from flow r in Q^n
$p_{i,j}$	probability of successful transmission from N_i to N_j		
p_p	packet error rate calculated in terms of bit error rate (p_e) as $p_p \approx 1 - p_e \times L_p$		
$W^{(j)}$	upper-bound of the average end-to-end delay of the j^{th} flow		
β	maximum number of transmissions of a packet at each node		
$\bar{R}_i$	mean residual service time in priority queues at N_i		
$P_{\mathrm{mtc}}(r)$	probability that a packet from flow r in Q^n moves to Q^c		
$N(r,w)$	average number of the packets of flow r arrived in Q^n during w time window		
$N(r)$	average number of the packets of flow r ahead of the currently arrived packet in Q^n		
I_i	set of all nodes in interference range of N_i including N_i		
h_x	probability that node x transmits a packet during transmission between two other nodes		
$P_{i,j}^d$	probability that N_i drops a packet with next-hop N_j after failure in β transmissions		
$P_{i,j}^{\mathrm{decode}}$	decoding probability of a coded packet arrived at N_j from N_i		
$T(m)$	time spent on DIFS and back-off in m^{th} transmission by taking into account the "clock freezing" behavior		
$P_{i,j}^{\mathrm{decode}}$	decoding probability of a coded packet arrived at N_j from N_i		

3.1.1 Network model and assumptions

We propose an analytical model of IXNC for bidirectional unicast flows in multi-hop wireless mesh networks to study the throughput and end-to-end delay. Our analytical

results are provided for a chain topology with k nodes as depicted in Figure 3.1, with two flows in opposite directions. As shown in this figure, N_1 and N_k transmit their packets to each other via intermediate nodes N_2 to N_{k-1}, while we assume that only N_{i-1} and N_{i+1} are in the transmission range of N_i.

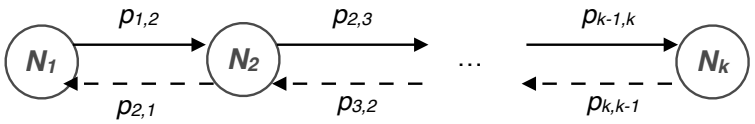


Figure 3.1: Chain topology used for the analytical model.

In this model, we assume that nodes usually do not move, and packets of UDP flows from a source traverse multiple wireless hops to be received by the destination. The model that we consider for interference assumes that a node cannot transmit and receive at the same time, and all transmissions in the carrier sensing range of the receiver are considered as interference. Furthermore, we assume that the feedback channel is reliable; thus if a node does not hear an acknowledgement (ACK) on time, it assumes that the data packet is lost.

In this network, there exist k nodes, namely N_1 to N_k with unlimited queue capacity. When a node finds the channel idle, it sends a packet from the head of its queue. We consider each node as a queuing system, where the packets in the sending buffer are customers of the queue and the node acts as the server. Hence, each node provides services with one server to the packets in its queue, and *Queuing Theory* can be used to model this network.

We assume that the queues are in a stable state, i.e., the arrival rate is less than the service rate. The packet generation rate at source N_i $(i = 1, k)$ follows the Poisson

model with a mean rate γ_i, while the service time at each N_i $(i = 1, ..., k)$ is exponentially distributed with a mean $1/\mu_i$. We also assume that the probability that node N_i delivers a packet to its neighbor N_j is $p_{i,j} > 0$, and for other nodes equals zero.

One may notice that our described network has all the properties of *open Jackson networks* [37, 38]; including 1) each node is considered a queuing system; 2) the packet generation rate at the source is assumed to be a Poisson process; 3) service time at node N_i is assumed independent from that of other nodes, and it is exponentially distributed with parameter μ_i; and 4) a packet that has completed service at node N_i (i.e., the packet has been transmitted) will go next to node N_j with probability $r_{i,j}$. This probability, presented in (3.1), for the next-hop equals successful transmission probability $p_{i,j}$, and for other nodes equals zero.

$$r_{i,j} = \begin{cases} p_{i,j} & \text{if } N_j \text{ is a neighbor of } N_i \\ 0 & \text{elsewhere} \end{cases} \tag{3.1}$$

To formulate this network, we employ concepts from the probability theory, queuing theory and Jackson networks [22, 51, 52]. Based on the *Burke's* Theorem [12], in the case of a stable stationary queuing system, the departure process of an exponential server is Poisson if the arrival rate follows a Poisson process. Furthermore, the *Jackson's Theorem* states that in the Jackson network each node behaves as if its input were Poisson. Therefore, the arrival rate at other nodes, in addition to the sources, can be considered a Poisson process. We assume that λ_i denotes the arrival rate at N_i.

3.1.2 Data link layer description

In this chapter, the same data link layer signaling as IEEE 802.11 DCF [1] is applied, with CSMA/CA random access. At the beginning of each time slot, a node, with a new packet to transmit, senses the channel. If the node finds the channel idle for a DIFS (Distributed Inter-Frame Space) period of time, it waits for a random back-off interval to minimize the probability of collision with packets transmitted by other nodes, and then transmits the packet. We consider that the transmission time of each packet is a fixed number of time slots.

The random back-off interval in DCF is discrete with binary exponential growth. To transmit a new packet, random back-off is uniformly chosen from $[0, \mathrm{CW}_{\min} - 1]$, where $\mathrm{CW}_{\min}$ is the minimum contention window. When a packet is retransmitted for the m^{th} time ($m > 0$), the contention window range will be extended to $[0, 2^m \mathrm{CW}_{\min} - 1]$, while $2^m \mathrm{CW}_{\min}$ is upper-bounded by $\mathrm{CW}_{\max}$.

Based on the specifications of the IEEE 802.11 standard, the back-off time and the DIFS counters are decremented as long as the channel is sensed idle. As soon as it is sensed busy, the node freezes the state of the clock and stops counting down until sensing the idle channel again. Therefore, although the value of DIFS and selected back-off (i.e., the number of ticks in the counter) are specified, the counter may pause due to another transmission which makes the channel busy. This "clock freezing" behaviour needs to be taken into account in calculating the back-off time.

The default feedback mechanism in the DCF is automatic repeat request (ARQ), where an ACK is transmitted by the receiver of the data packet, after a period of time called short inter-frame space (SIFS). Since the SIFS is shorter than the DIFS, no other

node will sense the channel idle for a DIFS before the end of the ACK transmission. If the sender of a data packet does not receive an ACK before time-out, it will increase the back-off interval and retransmit the packet.

3.1.3 The probability of successful transmission

We calculate the probability of successful transmission at each link in terms of the bit error rate (p_e) and collision. In general, a packet transmission, at the link between N_i and N_j, may fail due to packet error rate (p_p) or collision ($C_{i,j}$). Thus, the probability of successful transmission of a packet, with length L_p, from N_i to N_j can be calculated as:

$$p_{i,j} = (1 - C_{i,j})(1 - p_p) \approx (1 - C_{i,j})(1 - p_e \times L_p)\,. \tag{3.2}$$

We assume that the probability of collision between a data packet and an ACK is negligible; this is a valid assumption because: 1) the length of ACKs is significantly shorter than the length of data packets, and 2) ACKs are given higher priority and are sent earlier than any data packet. A transmission from N_i to N_j will fail if at the same time, N_j or any other node in its interference range transmits a packet. Let us denote I_j as the set of all nodes in the interference range of N_j, including N_j itself. Then the probability of successful transmission from N_i to its neighbor, N_j, can be computed as

$$p_{i,j} = (1 - p_e \times L_p) \prod_{N_x \in I_j - \{N_i\}} (1 - h_x)\,, \tag{3.3}$$

where h_x represents the probability that node N_x transmits a packet during packet transmission between two other nodes.

If N_i transmits a packet at time t, any node in its interference range will sense that the channel is busy after the propagation delay (δ), and avoid any transmission. Therefore,

during a propagation delay window before and after N_i's transmission (i.e., $(t-\delta, t+\delta)$), other nodes may transmit their packet which will collide with N_i's transmission. Although the propagation delay depends on the distance, we assume a fixed propagation delay as the maximum propagation delay. The probability that N_x transmits a packet during this time window can be estimated as $h_x = 2\delta\lambda_x$. The proof is given in Appendix A.

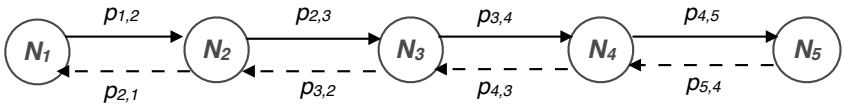


Figure 3.2: Chain topology with 5 nodes.

As an example, in the chain topology with 5 nodes depicted in Figure 3.2, a transmission from N_2 to N_3 will fail if at the same time slot that N_2 is transmitting, N_3 or N_4 transmits as well. Note that in this topology, where successive nodes are equally far apart, assuming a two-ray ground reflection propagation model with the default capture threshold of 10 dB, a transmission from N_1 or N_5 will not collide with the reception at N_3 due to capture effect [80] (i.e., transmissions from nodes two hops or farther away cannot cause any collision). Therefore, the probability of a successful transmission from N_2 to N_3 equals $p_{2,3} = (1-2\delta\lambda_3)(1-2\delta\lambda_4)(1-p_e L_p)$. In fact, the following equation can be used to compute the probability of successful transmission from N_i to N_j, when N_i and N_j are neighbors:

$$p_{i,j} = (1 - p_e \times L_p) \prod_{N_x \in I_j - \{N_i\}} (1 - 2\delta\lambda_x)\,. \tag{3.4}$$

3.2 Problem Formulation

In this section, we first provide the analytical model for traditional forwarding (i.e., non-coding scheme); then we extend it to the case that intermediate nodes can utilize network coding and combine packets of the two flows (i.e., coding scheme).

3.2.1 Non-coding scheme

In the non-coding scheme, the intermediate nodes forward only native packets, while the packets may enter the network (i.e., the queue network) either at node N_1 with a generation rate γ_1 or at node N_k with a generation rate γ_k, and depart from the other end of the chain. Therefore, the intermediate nodes receive packets from both directions. Let $\lambda_i^{(1)}$ and $\lambda_i^{(2)}$ denote the arrival rate of the first flow (i.e., from N_1 to N_k) and the second flow (i.e., from N_k to N_1) arriving at node N_i, respectively. Therefore, at each node $\lambda_i = \lambda_i^{(1)} + \lambda_i^{(2)}$.

We consider each node as a single $M/M/1/\infty$ queuing model. As explained earlier, the inter-departure time distribution in an $M/M/1$ queue with arrival rate λ, in a stable state, is an exponential distribution with mean $1/\lambda$. One of the key rules of probability used in this model states that "the sum of t independent Poisson processes with arrival rates λ_1, ..., λ_t is also a Poisson process with an arrival rate $\lambda = \sum_{i=1}^{t} \lambda_i$" [22]. Hence, the assumption of having Poisson arrivals at intermediate nodes holds, and each node can be considered as an independent $M/M/1$ queuing system.

To model the retransmission of packets in the network, feedback queues are required. As shown in Figure 3.3, we consider that node N_i delivers its packets to the next-hop N_j successfully with probability $p_{i,j}$, and retransmits the packets with probability $1 - p_{i,j}$, at

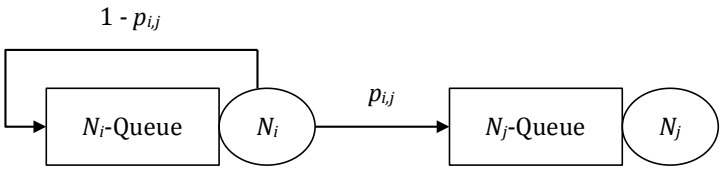


Figure 3.3: Feedback queue to model retransmission.

most $\beta - 1$ times (i.e., the packet is retransmitted if the last transmission fails). Hence, a packet is dropped if it cannot be delivered to the next-hop after β transmissions. We consider that the feedback channel is reliable, and the ACK messages are received successfully. Therefore, the probability that N_i drops a packet, after failure in β transmissions to the next-hop N_j, can be calculated as

$$P_{i,j}^{d} = (1 - p_{i,j})^{\beta} . \tag{3.5}$$

Taking retransmissions into account, (3.6a) and (3.6b) represent the arrival rate of the first flow (i.e., from N_1 to N_k) and the second flow (i.e., from N_k to N_1) at all nodes, respectively.

$$\begin{cases} \lambda_i^{(1)} = \gamma_i + \lambda_i^{(1)}(1 - p_{i,i+1})(1 - P_{i,i+1}^{d}) & \text{if } i = 1 \\ \lambda_i^{(1)} = \lambda_{i-1}^{(1)} p_{i-1,i} + \lambda_i^{(1)}(1 - p_{i,i+1})(1 - P_{i,i+1}^{d}) & \text{if } 1 < i < k \\ \lambda_i^{(1)} = \lambda_{i-1}^{(1)} p_{i-1,i} & \text{if } i = k \end{cases} \tag{3.6a}$$

$$\begin{cases} \lambda_i^{(2)} = \gamma_i + \lambda_i^{(2)}(1 - p_{i,i-1})(1 - P_{i,i-1}^{d}) & \text{if } i = k \\ \lambda_i^{(2)} = \lambda_{i+1}^{(2)} p_{i+1,i} + \lambda_i^{(2)}(1 - p_{i,i-1})(1 - P_{i,i-1}^{d}) & \text{if } 1 < i < k \\ \lambda_i^{(2)} = \lambda_{i+1}^{(2)} p_{i+1,i} & \text{if } i = 1 \end{cases} \tag{3.6b}$$

3.2.1.1 Successful transmission probabilities

As explained in Section 3.1.3, the probability of transmitting a packet successfully can be calculated in terms of the packet arrival rates and propagation delay by solving the following system of non-linear equations:

$$\begin{cases} p_{1,2} = (1-p_p)(1-2\delta\lambda_2)(1-2\delta\lambda_3) \\ ... \\ p_{i-1,i} = (1-p_p)(1-2\delta\lambda_i)(1-2\delta\lambda_{i+1}) \\ ... \\ p_{k-2,k-1} = (1-p_p)(1-2\delta\lambda_{k-1})(1-2\delta\lambda_k^{(2)}) \\ p_{k-1,k} = (1-p_p)(1-2\delta\lambda_k^{(2)}) \\ p_{k,k-1} = (1-p_p)(1-2\delta\lambda_{k-1})(1-2\delta\lambda_{k-2}) \\ ... \\ p_{i+1,i} = (1-p_p)(1-2\delta\lambda_i)(1-2\delta\lambda_{i-1}) \\ ... \\ p_{3,2} = (1-p_p)(1-2\delta\lambda_2)(1-2\delta\lambda_1^{(1)}) \\ p_{2,1} = (1-p_p)(1-2\delta\lambda_1^{(1)}) \,, \end{cases} \tag{3.7}$$

where all λ_is are functions of γ_1, γ_k, and successful transmission probabilities as described in (3.6).

3.2.1.2 Service time

The average service time (i.e., $1/\mu$), which is the time until a packet at the head of the transmission queue of N_i is received by the next-hop N_j, can be computed as:

$$\frac{1}{\mu} = \sum_{m=1}^{\beta} p_{i,j}(1-p_{i,j})^{m-1} \sum_{n=1}^{m} T_s(n) \,, \tag{3.8}$$

where $T_s(m)$ denotes the service time at the m^{th} transmission of a packet ($1 \leq m \leq \beta$), which is $T_s(m) = T(m) + T_{\text{data}} + \delta + \text{SIFS} + T_{\text{ack}} + \delta$, $1 \leq m \leq \beta$. T_{data} is the

transmission time of a packet (we assume the length of the packets is fixed.), T_{ack} denotes the transmission time of an ACK message, and $T(m)$ is calculated for the m^{th} transmission of a packet in terms of DIFS and back-off time, considering the "clock freezing" feature.

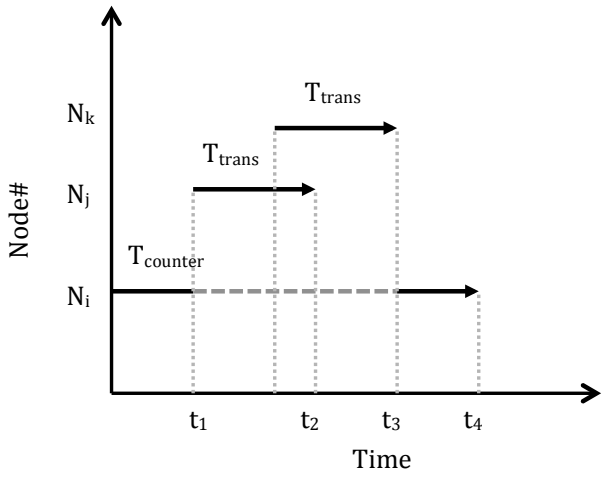


Figure 3.4: Clock freezing behavior of the back-off timer.

To explain "clock freezing", let us use the scenario depicted in Figure 3.4. As shown in this figure, node N_i sets the timer for T_{counter} to back-off before transmitting its packet. However at t_1, before the back-off timer reaches zero, N_i senses a packet transmission from N_j, stops counting down, and freezes the state for $T_{\text{trans}} = T_{\text{data}} + T_{\text{ack}} + \text{SIFS}$. Then, since another transmission by N_k occurs, N_i needs to wait until t_3. After that N_i senses the idle channel, resumes the timer, and it is ready to transmit the packet at t_4. Note that to consider *network allocation vector (NAV) virtual carrier-sensing* mechanism, we take into account T_{ack} and SIFS in calculating T_{trans}.

To take into account this "clock freezing" behavior, we compute the waiting time due

to DIFS and back-off as follows

$$T(m) = T_{\text{counter}}(m) \times e^{-\lambda T_{\text{counter}}(m)} + \sum_{i=1}^{\infty} (T_{\text{counter}}(m) + i \times T_{\text{trans}}) \times (1 - e^{-\lambda T_{\text{trans}}})^i \,, \tag{3.9}$$

which means that a node waits T_{counter} with the probability that during this period of time, it does not sense any other transmission. In addition, a node waits for $T_{\text{counter}} + i \times T_{\text{trans}}$ with the probability that during each T_{trans} time period, the node senses at least one packet transmission, and it happens i times. This equation provides an upper-bound for the expected back-off time. Its closed form is calculated as follows, and the proof is given in Appendix B.

$$T(m) = T_{\text{counter}}(m) \times e^{-\lambda T_{\text{counter}}(m)} + (1 - e^{-\lambda T_{\text{trans}}}) \left(\frac{T_{\text{counter}}(m)}{e^{-\lambda T_{\text{trans}}}} + \frac{T_{\text{trans}}}{e^{-2\lambda T_{\text{trans}}}} \right) . \tag{3.10}$$

In (3.9) and (3.10), λ represents the sum of arrival rates of the group of nodes which are in carrier sensing range of this node, and the term in the second line of both equations presents the probability of i transmissions from the nodes of this group during the waiting time $T_{\text{counter}}(m)$. In addition, $T_{\text{counter}}(m)$ is calculated in terms of m, the number of transmissions of a packet, considering the *binary exponential random back-off* interval, as:

$$\begin{aligned} T_{\text{counter}}(m) &= \text{DIFS} + T_{\text{backoff}}(m) \\ &= \text{DIFS} + \frac{2^{m-1}\text{CW}_{\min} - 1}{2}, 1 \le m \le \beta \,. \end{aligned} \tag{3.11}$$

Note that since the random back-off has a uniform distribution, its mean for the m^{th} transmission equals $\dfrac{2^{m-1}\text{CW}_{\min} - 1}{2}$.

3.2.1.3 Throughput

It is clear that the throughput, denoted by θ, is identical to the arrival rate at the destinations. Thus, it can be calculated by adding the arrival rate of the second flow at N_1 and the arrival rate of the first flow at N_k as follows

$$\theta = \lambda_1^{(2)} + \lambda_k^{(1)} . \tag{3.12}$$

3.2.1.4 End-to-end delay

The average end-to-end delay equals the summation of the time that each packet spends at the source and intermediate nodes. Also, the time spent at each node consists of the waiting time in the queue, and the time which takes a packet at the head of the queue to be received by the next-hop (i.e., service time). Based on queuing theory, the average time a packet spends at node N_i until it is received by the next-hop, defined as W_i, can be expressed as

$$W_i = \frac{1}{\mu_i - \lambda_i} . \tag{3.13}$$

Since in (3.8) we calculate an upper-bound of the service time (i.e., an upper-bound of $T(m)$), W_i presents an upper-bound of the waiting time at node N_i. There are two flows in the network; hence, we calculate the end-to-end delay for the packets of each flow separately, and then the average end-to-end delay is computed by applying the weighted average over the end-to-end delay of the two flows. It is clear that the end-to-end delay for each flow equals the sum of waiting time of the packets of the flow in different nodes, except for the destination. Therefore, an upper-bound of the end-to-end delay for the first and second flows can be computed by (3.14a) and (3.14b), respectively.

$$W^{(1)} = W_1^{(1)} + \sum_{i=2}^{k-1} W_i \tag{3.14a}$$

$$W^{(2)} = W_k^{(2)} + \sum_{i=2}^{k-1} W_i \,, \tag{3.14b}$$

where $W_i = \dfrac{1}{\mu_i - \left(\lambda_i^{(1)} + \lambda_i^{(2)}\right)}$ for intermediate nodes.

Note that while at intermediate nodes the packets of both flows arrive at the queue, in the queue at either of the sources the only packets arrived are those of the flow initiated from that source. Due to this reason, the waiting times at the sources are $W_1^{(1)} = \dfrac{1}{\mu_1 - \lambda_1^{(1)}}$ and $W_k^{(2)} = \dfrac{1}{\mu_k - \lambda_k^{(2)}}$. Then, an upper-bound of the average end-to-end delay can be computed as

$$W = \frac{\gamma_1}{\gamma_1 + \gamma_k} \times W^{(1)} + \frac{\gamma_k}{\gamma_1 + \gamma_k} \times W^{(2)} \,. \tag{3.15}$$

3.2.2 Coding scheme

To model network coding, we use multi-class queuing networks, and consider that native and coded packets enter separate queues. Furthermore, coded packets in Q^c have a non-preemptive higher priority over the native packets in Q^n. This means that a coded packet will be forwarded earlier than all the packets waiting in Q^n, but a native packet in service (i.e., the native packet which is being transmitted) is not interrupted by coded packets.

As in the previous case, we assume that the rate of generating packets at N_1 and N_k equals γ_1 and γ_k, respectively, and $\lambda_i^{(1)}$ and $\lambda_i^{(2)}$ represent the rate of the first and the second flow at N_i, respectively. Also, we define λ_i^n as the arrival rate of native packets, and λ_i^c as the arrival rate of coded packets at N_i.

3.2.2.1 Coding module

As shown in Figure 3.5, N_i receives native and coded packets of both flows from the previous hops. Although a coded packet is the combination of both flows, the receiver N_i is the next-hop of either the first flow or the second flow (i.e., intended flow). Due to this reason, we distinguish coded packets of different flows arriving at N_i.

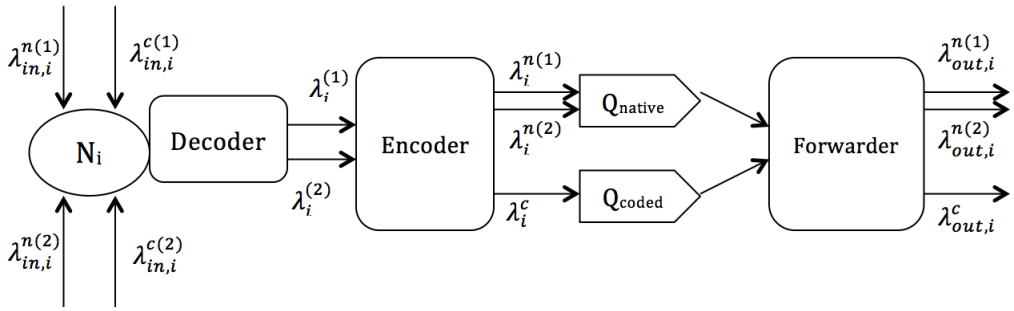


Figure 3.5: A packet from arrival until departure.

The *decoder*, in Figure 3.5, decodes the received coded packets and finds the next-hop of the packets. The outputs of this module are native packets of the first and the second flows with rates $\lambda_i^{(1)}$ and $\lambda_i^{(2)}$, respectively. In fact $\lambda_i^{(1)}$ ($\lambda_i^{(2)}$) represents the sum of the rate of arrived native packets of the first (second) flow, denoted by $\lambda_{in,i}^{n(1)}$ ($\lambda_{in,i}^{n(2)}$), the rate of successfully decoded packets of the first (second) flow, and the rate of retransmitted packets. Therefore, the arrival rates at the *encoder* for both flows (i.e., $\lambda_i^{(1)}$ and $\lambda_i^{(2)}$ in Figure 3.5) are calculated as

$$\begin{cases} \lambda_i^{(1)} = \lambda_{in,i}^{n(1)} + \lambda_{in,i}^{c(1)} P_{i-1,i}^{\text{decode}} + \lambda_i^{(1)}(1 - p_{i,i+1})(1 - P_{i,i+1}^{d}) & \text{if } 1 \leq i < k \\ \lambda_i^{(1)} = \lambda_{in,i}^{n(1)} + \lambda_{in,i}^{c(1)} & \text{if } i = k \end{cases} \tag{3.16a}$$

$$\begin{cases} \lambda_i^{(2)} = \lambda_{in,i}^{n(2)} + \lambda_{in,i}^{c(2)} P_{i+1,i}^{\text{decode}} + \lambda_i^{(2)}(1 - p_{i,i-1})(1 - P_{i,i-1}^{d}) & \text{if } 1 < i \leq k \\ \lambda_i^{(2)} = \lambda_{in,i}^{n(2)} + \lambda_{in,i}^{c(2)} & \text{if } i = 1 \end{cases} \tag{3.16b}$$

Note that in a general topology, to decode a coded packet with two coding partners, the node should have already received one of them from the opposite direction. Therefore, the decoding probability of a coded packet arrived at N_i from N_{i-1} (or N_{i+1}) is $P_{i-1,i}^{\text{decode}} = (1 - P_{i+1,i}^{d})$ (or $P_{i+1,i}^{\text{decode}} = (1 - P_{i-1,i}^{d})$). However, in the chain topology discussed in this chapter, the decoding probability is always one (i.e., $P_{i-1,i}^{\text{decode}} = P_{i+1,i}^{\text{decode}} = 1$). In fact, if N_i receives coded packet $P_1 \oplus P_2$ from N_{i-1}, and P_1 is its intended packet (i.e., the packet that this node was its next-hop), it must have already received P_2 from N_{i+1}; otherwise, N_{i-1} could not have received P_2 to combine it with P_1.

Previous analytical studies on network coding usually do not consider opportunistic coding, and assume that the transmission of a native packet at the head of the queue, ready to be forwarded, is postponed until receiving packets from other flows, to mix them with the native packet, and send coded packets instead of native ones as much as possible. This assumption provides more coding opportunities, and simplifies estimating the rate of coding opportunities (i.e., forwarding coded packets) at each node. For example, in the chain topology explained here, the rate would be calculated as the minimum of the arrival rates of the flows.

However, this postponing will increase the end-to-end delay tremendously, especially when the flows are asymmetric as the transmission of native packets should be delayed, waiting for coding partners to arrive. In addition, many practical and well-known network coding protocols are designed based on opportunistic coding, and do not impose such an artificial delay [48, 60, 106]. To limit the delay in the network, and also to analyze the

behavior of network coding in practical scenarios, we do not hold transmission of native packets. This means that the arrival rate in Q^c is not the minimum of the arrival rates of the two flows any more, and can be calculated as will be explained here.

In our model, a packet may be transmitted natively if it is at the head of Q^n, and there is no packet in Q^c. Therefore, the encoder receives the arrived native packet P from flow r ($r = 1, 2$), and looks for a packet from flow $\bar{r}$ (i.e., the flow from the opposite direction that can be mixed with flow r, $\bar{r} = 3 - r$) in Q^n. If the node finds such a packet $\bar{P}$, it removes $\bar{P}$ from Q^n, mixes it with P and inserts the coded packet into Q^c; otherwise, P arrives at Q^n. Therefore, a packet will be inserted into Q^c if a native packet from flow r arrives at the encoder, and if Q^n contains at least one packet from the flow $\bar{r}$.

On the other hand, packet P, from flow r, will be sent natively if before it is forwarded, it cannot be mixed with any packet from the other flow. This happens if 1) when it arrives, the queue of the other flow is empty, and 2) during the time that P is waiting in Q^n to be forwarded, the number of packets of the other flow which arrive at Q^n is less than the number of packets of flow r in Q^n ahead of P. Note that although all native packets arrive at the same queue, we send them to two separate virtual queues, one for each flow, to be able to calculate the number of packets of each flow in the queue.

If we denote $W(Q^n)$ as the waiting time of an arrived native packet in Q^n, then the number of packets of flow r that arrive at Q^n during this time equals $N(r, W(Q^n)) = \lambda_i^{(r)} \times W(Q^n)$. When P from flow r arrives, if the number of packets of its flow in Q^n (i.e., packets of flow r ahead of P) is less than $N(\bar{r}, W(Q^n))$, P is moved from Q^n to Q^c before it is forwarded; otherwise, it stays in Q^n. Thus, the probability that a packet from flow r moves to the coded queue of node N_i, Q_i^c, even if it first arrives at the native queue, Q_i^n, can be calculated as

$$
\begin{aligned}
P_{\text{mtc}}(r) &= \Pr[N(\bar{r}, W(Q^n)) > N(r)] \\
&= \sum_{k=0}^{\infty} \Pr[(N(\bar{r}, W(Q^n)) > N(r))|(N(r) = k)] \times \Pr[N(r) = k] \\
&= \sum_{k=0}^{\infty} \Pr[(N(\bar{r}, W(Q^n)) > k)] \times \Pr[N(r) = k] \\
&= \sum_{k=0}^{\infty} \left(1 - \sum_{j=0}^{k} \frac{e^{\left(-\lambda_i^{n(\bar{r})} W(Q_i^n)\right)} \left(\lambda_i^{n(\bar{r})} W(Q_i^n)\right)^j}{j!} \right) \left(\frac{\lambda_i^{n(r)}}{\mu_i^{n,\text{seen}}} \right)^k \left(1 - \frac{\lambda_i^{n(r)}}{\mu_i^{n,\text{seen}}} \right) ,
\end{aligned}
\tag{3.17}
$$

where $\bar{r}$ is the flow from the opposite direction that can be combined with flow r, $N(r, w)$ denotes the number of packets of flow r arrived in Q^n during time window w, and $N(r)$ is the number of the packets of flow r ahead of the currently arrived packet in Q^n. Also, $\mu_i^{n,\text{seen}}$ denotes the service time seen by Q_i^n as is discussed later. The closed form of (3.17) can be computed as

$$
P_{\text{mtc}}(r) = 1 - e^{\left(W(Q_i^n) \lambda_i^{n(\bar{r})} \left(\frac{\lambda_i^{n(r)}}{\mu_i^{n,\text{seen}}} - 1 \right) \right)} . \tag{3.18}
$$

We provide the proof in Appendix C.

Next the arrival rate of the packets of the first and second flows in the native queue of N_i is calculated as

$$
\lambda_i^{n(1)} = \lambda_i^{(1)} \times \pi_0(Q_i^{n(2)}) \times (1 - P_{\text{mtc}}(1)) , \tag{3.19a}
$$

$$
\lambda_i^{n(2)} = \lambda_i^{(2)} \times \pi_0(Q^{n(1)}) \times (1 - P_{\text{mtc}}(2)) , \tag{3.19b}
$$

where $\pi_0(Q)$ is the probability that queue Q is empty. This equation means that the arrival rate of native packets of the first flow in Q_i^n (i.e., $\lambda_i^{n(1)}$) equals the arrival rate

of the packets of the first flow at the encoder (i.e., $\lambda_i^{(1)}$) for which, in their arrival time, 1) there is no packet from the second flow in Q_i^n (i.e., $\pi_0(Q_i^{n(2)})$), and 2) the packet will stay in Q_i^n during its waiting time in the queue (i.e., $1 - P_{\text{mtc}}(1)$). $\lambda_i^{n(2)}$ is calculated in a similar way. Also, the arrival rate in the coded queue of N_i, can be calculated as

$$\lambda_i^c = \frac{\lambda_i^{(1)} + \lambda_i^{(2)} - \lambda_i^{n(1)} - \lambda_i^{n(2)}}{2}. \tag{3.20}$$

The division by two is because each coded packet is a combination of two native packets.

3.2.2.2 Native and coded queues

The arrival rates in Q_i^n and Q_i^c equal $\lambda_i^{n(1)} + \lambda_i^{n(2)}$ and λ_i^c, respectively. The *forwarder* module, in Figure 3.5, is responsible for forwarding packets. If Q_i^c is not empty, it will select the packet from the head of Q_i^c; otherwise, the packet is chosen from the head of Q_i^n if it is not empty.

As stated earlier, priority queues are used to model this case, where the arrival rate in Q_i^n is the sum of the arrival rates of both flows (i.e., $\lambda_i^n = \lambda_i^{n(1)} + \lambda_i^{n(2)}$), and the total arrival rate in the queuing system of N_i is presented by $\lambda_i = \lambda_i^n + \lambda_i^c$. Knowing the input rate of native and coded packets at all nodes, one can calculate the output rate at different nodes. Note that since we assume that the queuing system is in a stable state, the departure rates equal the arrival rates ($\lambda_{out,i}^{n(1)} = \lambda_i^{n(1)}, \lambda_{out,i}^{n(2)} = \lambda_i^{n(2)}, \lambda_{out,i}^c = \lambda_i^c$). Finally, the throughput can be computed using (3.12).

Tables 3.2 and 3.3 provide the input rates of native and coded packets at all nodes. Moreover, it is clear that the output rate of the first flow at N_1 and that of the second flow at N_k are γ_1 and γ_k, respectively. In addition, the output rate of the second flow and coded packets at N_1 and the output rate of the first flow and coded packets at N_k

Table 3.2: Input rates of native packets at all nodes.

$\boldsymbol{i}$	$\boldsymbol{\lambda_{in,i}^{n(1)}}$	$\boldsymbol{\lambda_{in,i}^{n(2)}}$
$i = 1$	γ_1	$\lambda_{out,i+1}^{n(2)} \times p_{i+1,i}$
$1 < i < k$	$\lambda_{out,i-1}^{n(1)} \times p_{i-1,i}$	$\lambda_{out,i+1}^{n(2)} \times p_{i+1,i}$
$i = k$	$\lambda_{out,i-1}^{n(1)} \times p_{i-1,i}$	γ_k

Table 3.3: Input rates of coded packets at all nodes.

$\boldsymbol{i}$	$\boldsymbol{\lambda_{in,i}^{c(1)}}$	$\boldsymbol{\lambda_{in,i}^{c(2)}}$
$i = 1, i = 2$	0	$\lambda_{out,i+1}^{c} \times p_{i+1,i}$
$2 < i < k-1$	$\lambda_{out,i-1}^{c} \times p_{i-1,i}$	$\lambda_{out,i+1}^{c} \times p_{i+1,i}$
$i = k, i = k-1$	$\lambda_{out,i-1}^{c} \times p_{i-1,i}$	0

are equal to zero, as in (3.21).

$$\begin{cases} \lambda_{out,1}^{n(1)} = \gamma_1 \\ \lambda_{out,k}^{n(2)} = \gamma_k \\ \lambda_{out,1}^{n(2)} = 0 \\ \lambda_{out,k}^{n(1)} = 0 \\ \lambda_{out,i}^{c} = 0 \quad \text{if } i = 1, k \end{cases} \tag{3.21}$$

3.2.2.3 Service time and end-to-end delay

As stated earlier, we use two different types of queues for native and coded packets, while the coded packets in Q^c have a non-preemptive higher priority over native packets in Q^n. In such a scenario, the service time seen by the native packets is different from the service

time of a regular $M/M/1$ queue. The reason is that a native packet at the head of Q^n should wait for all packets in Q^c to be transmitted before its turn for transmission. To estimate the service time *seen* by the native packets (i.e., the packets in lower priority queue) at N_i, denoted by $\mu_i^{n,\text{seen}}$, we start from the formula in queuing theory, which calculates the waiting time of a packet in a $M/M/1$ queuing system as

$$W_{\text{system}} = \frac{1}{\mu - \lambda}\,. \tag{3.22}$$

Therefore, the service time can be calculated as $\mu = \lambda + 1/W_{\text{system}}$. Since for the native queue at N_i, $\lambda = \lambda_i^{n(1)} + \lambda_i^{n(2)}$, the waiting time of the packets in the lower priority queue (i.e., Q_i^n) can be computed as $W(Q_i^n) = \bar{R}_i/(1-\rho_i^c)(1-\rho_i^c-\rho_i^n)$, and the waiting time of native packets before delivery to the next-hop equals $W_{\text{system}} = W(Q_i^n) + \dfrac{1}{\mu_i^n}$, we can calculate the service time *seen* by the packets in Q_i^n as

$$\mu_i^{n,\text{seen}} = \lambda_i^{n(1)} + \lambda_i^{n(2)} + \frac{1}{\dfrac{\bar{R}_i}{(1-\rho_i^c)(1-\rho_i^c-\rho_i^n)} + \dfrac{1}{\mu_i^n}}\,, \tag{3.23}$$

where μ_i^n is the service time of native packets in a regular queuing system that has been calculated earlier in (3.8). As presented in (3.17) and (3.18), $\mu_i^{n,\text{seen}}$ is used to calculate P_{mtc}. Table 3.4 shows the required equations to compute variables described in this subsection.

Furthermore, when a node sends a coded packet, it needs to wait for more than one ACK. In our model with two flows, the service time for coded packets, μ_i^c, is calculated using (3.8) again, where

$$T_s(m) = T(m) + T_{\text{data}} + \delta + 2 \times (\text{SIFS} + T_{\text{ack}} + \delta)\,. \tag{3.24}$$

Table 3.4: Calculation of some variables' values.

Variable	**Equation**
ρ_i	$\frac{\lambda_i}{\mu_i}$
$\bar{R}_i$	$\frac{\rho_i^n}{\mu_i^n} + \frac{\rho_i^c}{\mu_i^c}$
$W(Q_i^c)$	$\frac{\bar{R}_i}{(1-\rho_i^c)}$
$W(Q_i^n)$	$\frac{\bar{R}_i}{(1-\rho_i^c)(1-\rho_i^c-\rho_i^n)}$
$\mu_i^{n,\text{seen}}$	$\lambda_i^{n(1)} + \lambda_i^{n(2)} + \frac{1}{W(Q_i^n) + \frac{1}{\mu_i^n}}$

Since packets in Q^c and Q^n have different average waiting times, we calculate the waiting time of native and coded packets separately at each node, and then apply the weighted average to compute the average waiting time at each node, as

$$W_i = \frac{\lambda_i^c}{\lambda_i^c + \lambda_i^n} \times \left(W(Q_i^c) + \frac{1}{\mu_i^c} \right) + \frac{\lambda_i^n}{\lambda_i^c + \lambda_i^n} \times \left(W(Q_i^n) + \frac{1}{\mu_i^n} \right) . \tag{3.25}$$

Finally, the average end-to-end delay can be computed using (3.14)-(3.15). Note that we assume that the encoding and decoding delays are negligible, and the coding overhead is small enough that we can consider similar length for coded and native packets. In addition, since we calculate an upper-bound of $T_s(m)$, our analytical model provides an upper-bound of the end-to-end delay for both non-coding and coding schemes.

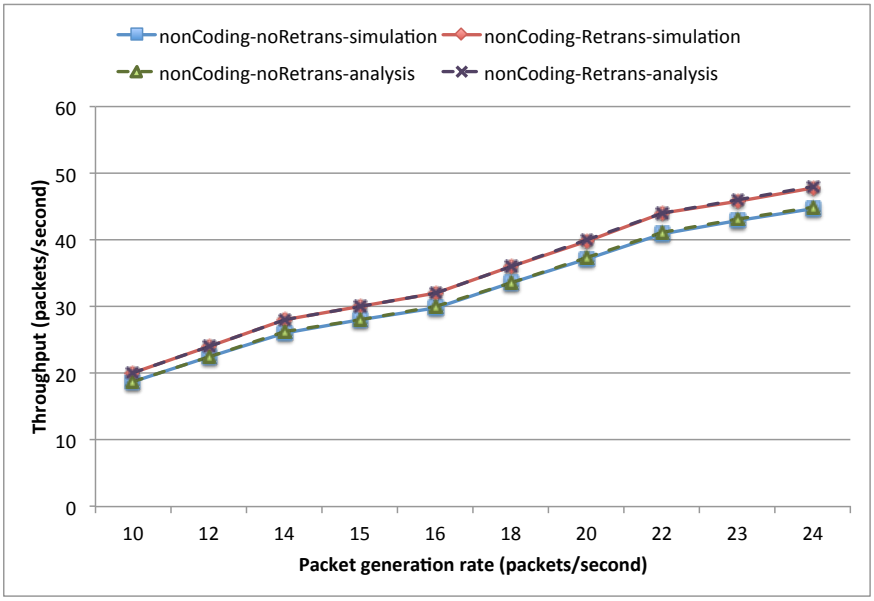
3.3 Performance Evaluation

3.3.1 Network description

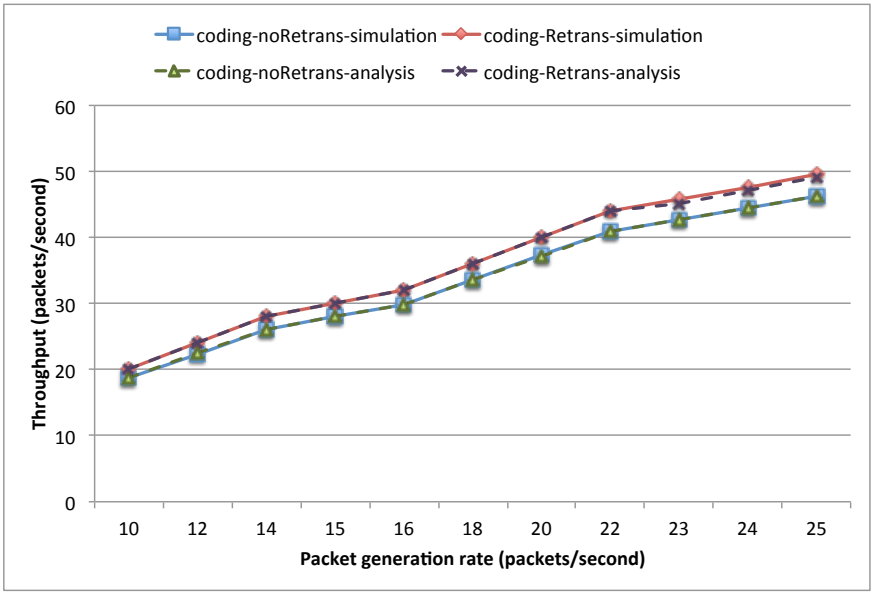
To verify the accuracy of our proposed analytical model, we run simulations in NS-2 for the chain topology depicted in Figure 3.1, where the distance between successive nodes is 200 m, and N_1 and N_k transmit packets to each other via intermediate nodes N_2, ..., N_{k-1}. The channel propagation used in NS-2 is a two-ray ground reflection model [80], the transmission range is 250 m, and the carrier sensing range is 550 m. Hence, in our chain topology, the nodes within two-hop distance of each node are in its carrier sensing range. However, due to the capture effect, the interference range is limited to the nodes one hop away.

In our simulation, we use the IEEE 802.11 standard as the MAC layer protocol, and our physical layer introduces random packet loss by adopting bit error rates (p_e). Therefore, the receiver will drop the packet with a probability which is calculated in terms of p_e. In addition, a node may drop a packet due to collision. Based on the specifications, a node transmits a packet at most 7 times (i.e., $\beta = 7$).

The link rate is set to 2 Mbps. The sources, in our simulation scenarios, send Poisson data flows with a datagram size of 1000 bytes. We compare the analytical results with the simulation results in different scenarios in terms of throughput and end-to-end delay by varying the packet generation rate and BER. Also, to compare coding and non-coding schemes, we calculate the maximum stable throughput for both cases.



(a) Non-coding scheme.



(b) Coding scheme.

Figure 3.6: Throughput comparison for different packet generation rates in a chain topology with 5 nodes and $p_e = 2 \times 10^{-6}$.

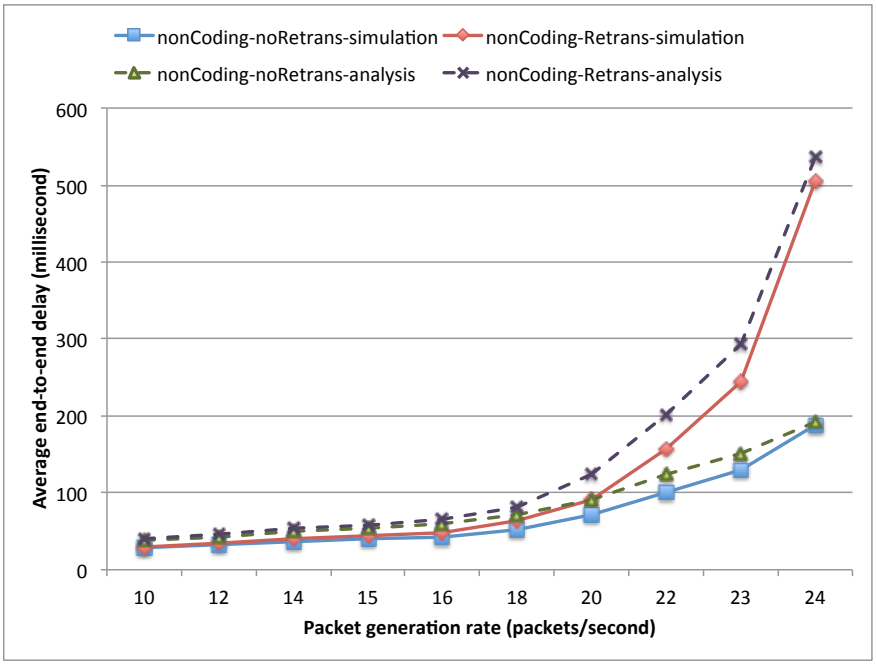
3.3.2 Effect of packet generation rate

In this section, we compare the simulation and analytical results for several packet generation rates in the topology depicted in Figure 3.2 with 5 nodes. In our simulations, Poisson flows between N_1 and N_5 last for 170 seconds, and throughput is calculated as the arrival rate of packets at the destinations by the end of simulation. We change the generation rate of packets at sources while the BER is fixed to 2×10^{-6}, and calculate the total throughput and an upper-bound of the average end-to-end delay by assuming an equal packet generation rate at sources (i.e., $\gamma = \gamma_1 = \gamma_k$). We compare the simulation and analytical results for the cases that 1) nodes do not retransmit a packet even if its transmission fails, and 2) nodes transmit a packet at most 7 times (i.e., $\beta = 7$).

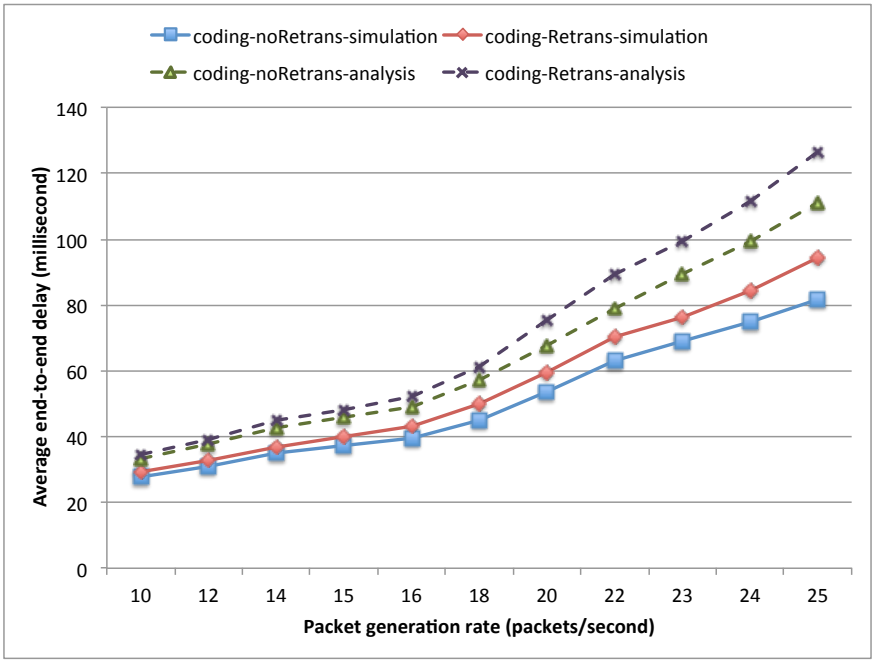
Figure 3.6a presents the analytical and simulation results of throughput for non-coding scheme, both with and without retransmission. Also, Figure 3.6b shows the corresponding results for coding scheme. The consistency of the simulation and analytical results corroborates the validity of our analytical model. In addition, one may notice that the throughput at each given packet generation rate is higher when retransmission is enabled. In fact, by disabling the reransmission mechanism, all the efforts to deliver a packet are wasted even if it has made all the way but the very last hop.

Comparing Figure 3.6a and Figure 3.6b, no considerable throughput gain can be seen for coding scheme in comparison with non-coding scheme especially at lower arrival rates. This is due to the fact that without holding native packets, network coding usually shows its gain over the traditional forwarding approach, where arrival rates are high enough to provide frequent coding opportunities. We will discuss the gain further in Section 3.3.4.

Regarding the average end-to-end delay, the results in Figure 3.7 show that our ana-



(a) Non-coding scheme.



(b) Coding scheme.

Figure 3.7: Average end-to-end delay comparison for different packet generation rates in a chain topology with 5 nodes and $p_e = 2 \times 10^{-6}$.

lytical model provides an upper-bound for the average end-to-end delay in different packet generation rates for both non-coding and coding schemes. In addition, in both scenarios (i.e., with and without retransmission), the average end-to-end delay increases with the packet generation rate; the reason is that at higher generation rates more packets are queued at nodes, which increases the waiting time and consequently the end-to-end delay of the network. However, the end-to-end delay is shorter when retransmission is disabled because each packet has only one transmission chance to be delivered to the next-hop, and lost packets do not contribute to delay calculation.

As a matter of fact, without retransmission a packet is either dropped or delivered to the next hop with only one transmission. On the other hand, with enabling retransmission, the packet is provided with up to β chances to repeat, which improves throughput at the cost of a longer delay. Furthermore, as shown in Figure 3.7, without utilizing network coding the delay grows faster. This is due to the fact that network coding allows more than one packet to be delivered to the next-hop in one transmission, which accelerates packet delivery, and reduces contention.

3.3.2.1 Throughput-delay trade-off

As presented in Figure 3.6, if the end-to-end delay of the network is finite (i.e., the queues are in stable state), the throughput is an increasing function of packet generation rate [53]. On the other hand, Figure 3.7 shows that the end-to-end delay is also an increasing function of the packet generation rate. As explained earlier, this is due to the fact that generating new packets faster increases the number of packets queued to be transmitted, which means longer waiting time in the queues, as confirmed by (3.13).

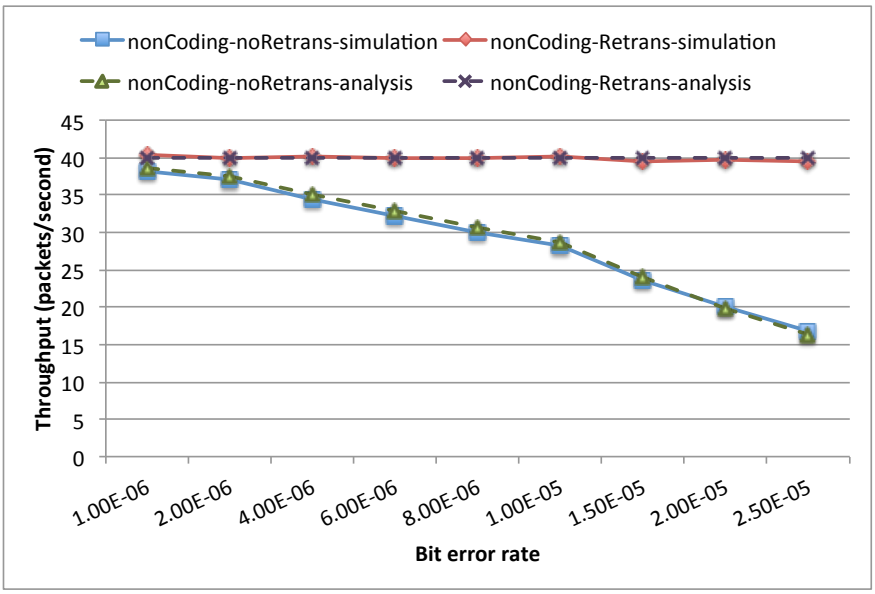
This verifies a trade-off between throughput and end-to-end delay that has been dis-

cussed in the literature [20, 40, 53]. To find the delay-constraint capacity, one needs to calculate the optimal packet generation rate for a given end-to-end delay. In our model, for both traditional forwarding and network coding, it can be calculated by increasing the packet generation rate as long as the end-to-end delay is less than a specified maximum value. Doing so, one can obtain the packet generation rate in which the network achieves the maximum throughput satisfying the end-to-end delay requirement.

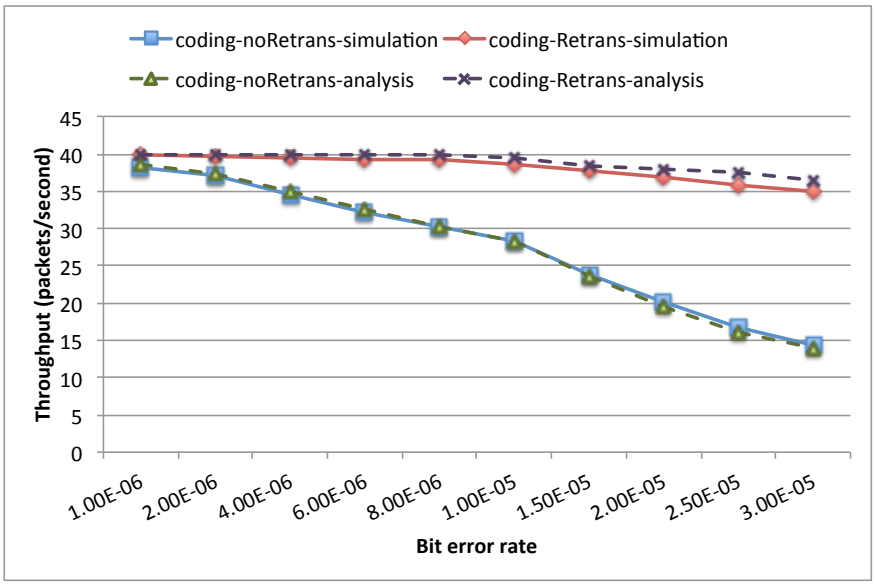
3.3.3 Effect of bit error rate

To study the validity of our model under different link qualities and packet loss probabilities, we change the BER, and provide simulation and analytical results for coding and non-coding schemes for the same topology depicted in Figure 3.2, both with and without retransmission. In these experiments, the packet generation rate at both sources (i.e., N_1 and N_5) is set to 20 packets/second. In general, as shown in Figure 3.8 and Figure 3.9, at lower BERs, the network performance with retransmission is very close to the case that retransmission is disabled. This is because at higher link qualities most of the packets are delivered to the next hop with one transmission without any need for retransmission.

As shown in Figure 3.8, the throughput calculated based on the proposed model perfectly matches the simulation results for different BERs. In addition, when retransmission is disabled, the throughput drops with increase in the BER. On the other hand by enabling retransmission, the throughput remains almost constant especially for the non-coding scheme. The reason is that retransmission provides each packet with up to β chances to be delivered to the next-hop, which is usually sufficient for most packets in these scenarios even at higher BERs. One may notice that the coding scheme does not

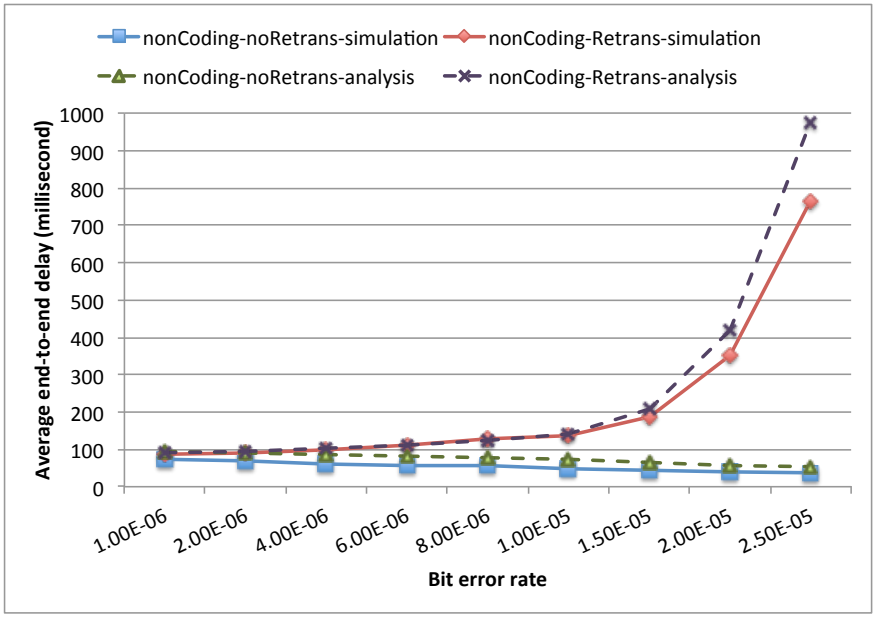


(a) Non-coding scheme.

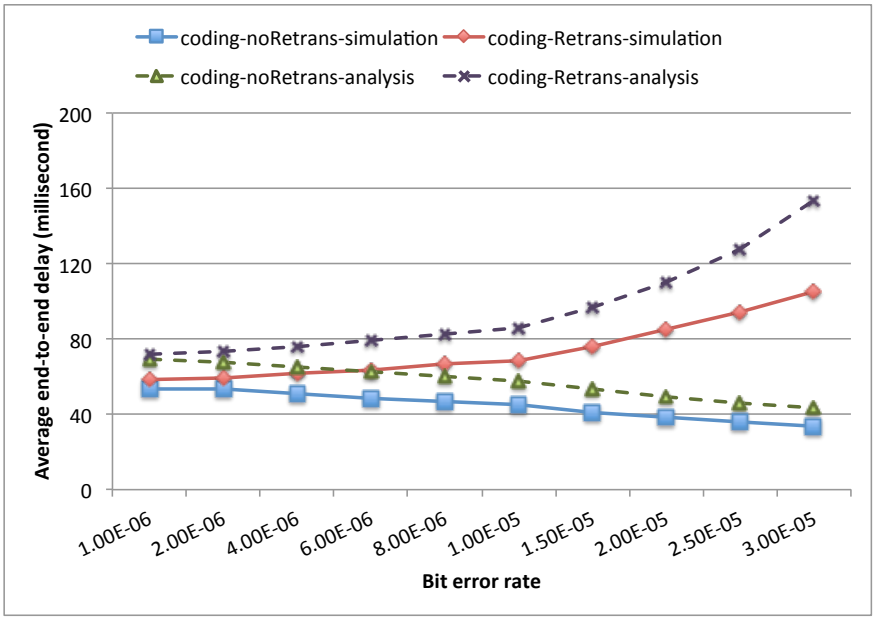


(b) Coding scheme.

Figure 3.8: Throughput comparison for different BERs in a chain topology with 5 nodes and $\gamma = 20$.



(a) Non-coding scheme.



(b) Coding scheme.

Figure 3.9: Average end-to-end delay comparison for different BERs in a chain topology with 5 nodes and $\gamma = 20$.

seem as resilient as the non-coding scheme when retransmission is enabled; the reason is that, to decode each coded packet, two packets should be delivered successfully rather than one, which reduces the chance of successful delivery of coded packets even when retransmission is enabled.

Regarding the average end-to-end delay, as shown in Figure 3.9, when the retransmission is disabled, the delay decreases for higher BERs. The reason is that more packets are dropped, and dropped packets do not contribute to the delay calculations. In addition, by increasing the packet loss rate, the number of packets waiting in the transmission queue of nodes decreases, which again causes a shorter end-to-end delay for delivered packets. On the other hand, the delay increases with the BER when the retransmission mechanism is utilized, as packets require more retransmissions to get to the next-hop; this adds to both service time and waiting time.

Comparing the coding and non-coding schemes, the effect of the BER is less on coding scheme than on non-coding scheme because in the coding scheme more packets can be forwarded in each transmission. Moreover, as shown in Figure 3.9, the average end-to-end delay calculated based on our analytical model provides an upper bound for the simulation results in all scenarios.

3.3.4 Maximum stable throughput

In this subsection, we compare the maximum stable throughput of the coding and non-coding schemes using both analytical and simulation results. The maximum stable throughput, as the name suggests, presents the maximum throughput of the network while the nodes' queues are still in a stable state (i.e., the arrival rate is less than the ser-

vice rate). In these experiments, the BER is set to 2×10^{-6}, and the results are provided for the chain topology depicted in Figure 3.1 with variant number of nodes.

```
Initialization:
γ1 = γk = γini   //arrival rate at the sources
θ = 0           //throughput
for each node i
    ρi < 1

Main Procedure:
While (∀1 < i < k, ρi < 1)
    θmax = θ
    solve the system of non-linear equations and
    calculate θ, λi, 1 < i < k
    for each node i
        calculate μi
        ρi = λi/μi
    γ1 = γk = γ1 + 1

return θmax
```

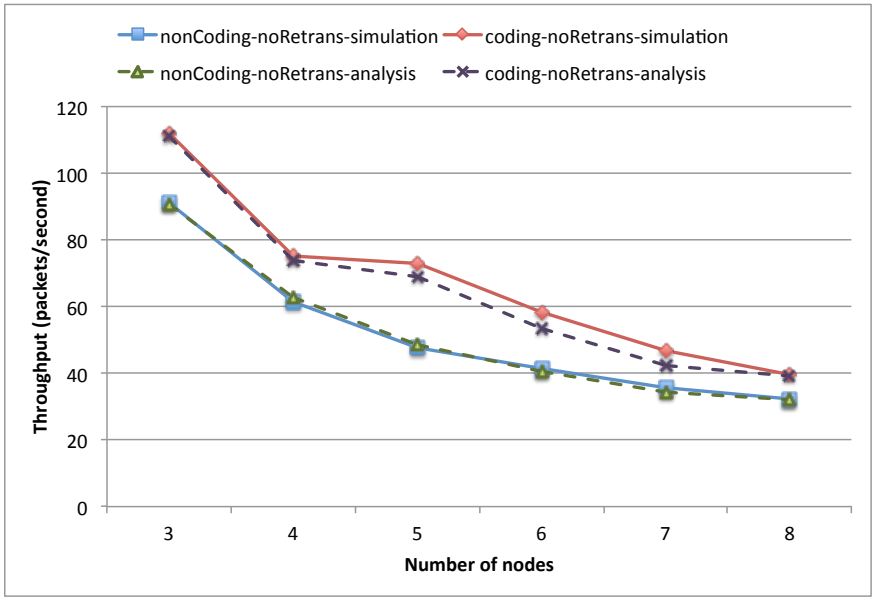
Figure 3.10: Pseudo-code of calculating the maximum stable throughput. γ_1 and γ_k represent the packet generation rates at the sources, initialized with a small value γ_{ini}. θ denotes the calculated throughput for the given generation rate.

To find the maximum stable throughput in simulations for each network size, we increase the packet generation rate at the sources as the throughput increases, and the queues are in stable state. In our analytical model, we follow the same idea since the maximum stable throughput is an increasing function of the packet generation rate. We gradually increase the packet generation rate at the sources. For each given generation rate, the system of non-linear equations provided in Section 3.2 is solved, providing us with the arrival rates at all nodes as well as other required parameters. Then by calculating the service rates, we can verify whether all nodes are still in a stable state. As soon as the condition of stability is not valid in at least one node, we conclude that the packet

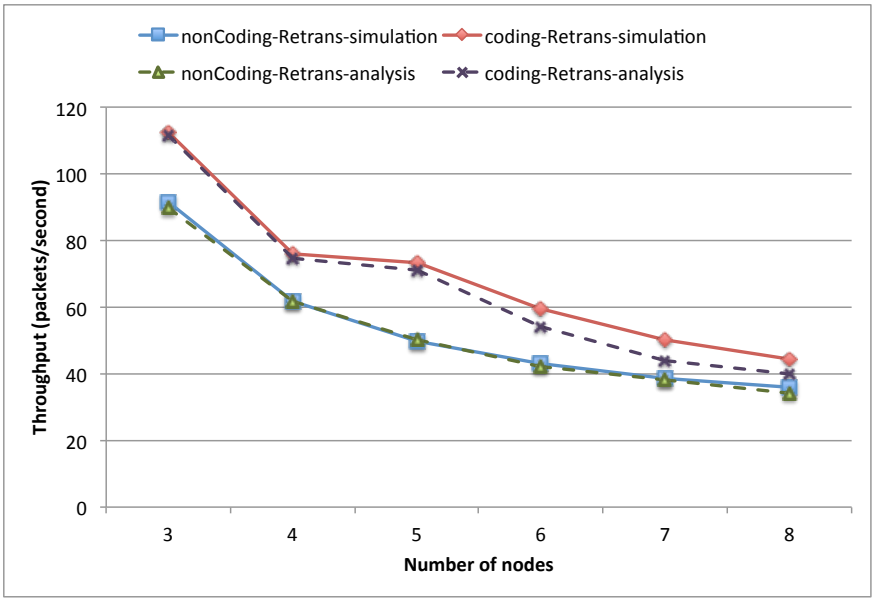
generation rate is not acceptable at the sources anymore, and we calculate the throughput for the greatest acceptable generation rate as the maximum stable throughput. Figure 3.10 presents the pseudo-code for finding the maximum stable throughput in our analytical model.

As shown in Figure 3.11, our analytical model provides a good estimate of the maximum stable throughput of the network for both coding and non-coding schemes in chain topologies with different sizes. Figure 3.11a presents the results when the retransmission mechanism is disabled, while in Figure 3.11b, nodes are allowed to transmit each packet at most β times. In both cases, by increasing the number of nodes in the chain topology, the maximum stable throughput decreases, especially in smaller topologies. In our chain topology, the number of nodes in the carrier sensing range of a transmitter is between 2 and 4, depending on the transmitter's location. As the chain length increases, a larger fraction of the nodes will have 4 nodes in their carrier sensing range, which leads to more waiting due to CSMA random access. This causes a longer back-off time and consequently a longer service time, reducing the maximum stable throughput.

Furthermore, as also stated in [4], when the number of intermediate nodes increases, network coding's advantage over traditional routing fades out, and the maximum stable throughput of the coding scheme approaches that of the non-coding scheme. One reason is that most coding opportunities are provided by the middle node, where the arrival rate of packets from both directions are similar and balanced. As the chain topology grows (i.e., the number of hops increases), fewer packets from both directions can be received by the middle nodes, which reduces the coding opportunities. In addition, in longer chains, the ratio of unbalanced flows increases in other nodes, which further causes less coding opportunities.



(a) Without retransmission.



(b) With retransmission.

Figure 3.11: The maximum stable throughput comparison for different chain topology sizes, $p_e = 2 \times 10^{-6}$.

3.4 Summary

In this chapter, we utilized queuing theory to study the throughput and end-to-end delay of both traditional forwarding (i.e., non-coding scheme) and IXNC in multi-hop wireless mesh networks, where two unicast sessions in opposite directions traverse the network. We proposed an analytical framework considering the specifications of the IEEE 802.11 DCF, such as the binary exponential back-off time with clock freezing and virtual carrier sensing, to formulate the links quality, waiting time of the packets and retransmissions. Our analytical model assumes M/M/1 queues, which are in a stable state, while coded and native packets arrive at separate queues and coded packets have a non-preemptive higher priority over native packets. Furthermore, in our model as opposed to previous studies, the transmission of native packets is not artificially delayed for generating more coded packets (i.e., opportunistic coding); this makes it significantly more challenging to estimate coding opportunities at nodes, as described in Section 3.2.2.1.

We verified the accuracy of the proposed analytical model by computer simulation in NS-2, and the consistency of the results corroborates the validity of the model. Also, the results show that at any given packet generation rate, both throughput and end-to-end delay are higher when retransmission is enabled. However, when the BER increases, the trend is totally different with and without retransmission. By enabling retransmission, throughput stays almost constant across different BERs while the end-to-end delay increases significantly. On the other hand, when retransmission is disabled, both throughput and end-to-end delay are decreasing functions of the BER. In addition, while network coding in theory promises a greater capacity for wireless networks, the results for the maximum stable throughput show that when PHY/MAC layer constraints are taken into

account, this promise can be fulfilled better for smaller topologies. In fact, when the number of intermediate nodes increases, the maximum stable throughput of network coding becomes comparable to traditional forwarding. However, wireless mesh networks are meant as an extended access technology, and it is unlikely to have very long paths; thus network coding can still offer a competitive edge.

Chapter 4

FlexONC: Joint Opportunistic Routing and Network Coding in Wireless Mesh Networks

In Chapter 3, we proposed a comprehensive analytical model for IXNC to quantify its benefits over traditional non-coding scheme and study its performance for actual protocols considering PHY/MAC layer specifications. As discussed in Chapter 3, network coding and more specifically inter-flow network coding (IXNC) can improve the throughput of the network by intelligent mixing of packets of different flows. COPE [48] is one of the first methods that realize this idea in practical scenarios. Whenever an intermediate node receives packets from different flows, it encodes them if it is likely that the next-hops of the native packets combined in the coded packet are able to decode this packet and retrieve the original content.

However, coding opportunities in COPE are restricted only to joint nodes that receive

packets from multiple flows. Therefore, to provide more coding opportunities, COPE needs more packets to arrive at the same node. This traffic concentration may overload intermediate nodes, and cause longer delay, buffer overflow, and channel contention. In addition, in lossy networks the performance gain of IXNC significantly drops as the probability of overhearing and decoding of coded packets decreases, and intermediate nodes will not likely have many coding opportunities. For example, in COPE, IXNC is turned off if the loss rate exceeds a threshold with default value 20%.

On the other hand, OR is suitable and effective in highly lossy networks because it provides more chances for a packet to make progress toward the destination. A variety of studies address the discussed issues of IXNC by adding OR to it [31, 36, 69, 103]. As explained in Chapter 2, in some works, the closeness to the destination (i.e., to find the forwarder set) is calculated in terms of the geographical distance, which does not necessarily represent the quality of the path. In addition, in most of the research in this area, the maximum coding opportunities is the only factor taken into account to select the next forwarder, even if the path traveled by the packets is excessively longer than the shortest path. Furthermore, they either broadcast packets without any acknowledgment and retransmission mechanism [103], or end-to-end acknowledgments are sent instead of hop-by-hop ones [36].

BEND [106] is also an advancement over COPE that applies the combination of IXNC and OR to avoid traffic concentration. By taking advantage of the broadcast nature of wireless networks, BEND allows all receivers of the packet, in addition to the intended next-hop specified by the routing protocol, to help in mixing and forwarding the packet if they believe they can be helpful. However, this OR cannot be applied to two consecutive hops, and these *non-intended forwarders* (i.e., the receivers of the packet which are not

Table 4.1: Definition of some terms used in this dissertation.

Term	**Definition**
native packet	a packet that is not combined with any other packet
coded packet	XORed of more than one native packet
intended forwarder	the designated next-hop by the routing protocol
non-intended forwarder	the neighbors of the next-hop which can help in forwarding
coding node	a node in which coded packets are generated
eligible forwarder	a node which is the neighbor of both the next-hop and the second next-hop of a packet
coding partner	a native packet encoded with other packets

specified as the next-hop on the route defined by the routing protocol, and can help in forwarding) are allowed to assist the *intended forwarder* only in forwarding received native packets. In fact, if they receive a coded packet, they just discard it, even if they were able to decode the received packet. This restriction not only limits the number of coding opportunities in the network but also increases the number of retransmissions. The terms *intended* and *non-intended forwarders* as well as some other terms used in this dissertation are summarized in Table 4.1.

To better utilize the broadcast nature of wireless networks, we introduce FlexONC (Flexible and Opportunistic Network Coding), which provides more flexibility to previous methods like COPE and BEND by adding OR, and allowing non-intended forwarders to help in decoding in addition to encoding and forwarding.

The main contributions of FlexONC are as follows:

1. More diffusion gain since more packets (i.e., coded and native packets) can be forwarded by a node other than their intended forwarder

2. Faster packet delivery to the final destination because even if the intended forwarder does not receive the packet or cannot decode the received coded packet, some non-

intended forwarders can still help

3. More coding opportunities as non-intended forwarders are eligible to receive and probably decode coded packets and consider them as candidates to be mixed with other packets.

4.1 Overview of FlexONC

4.1.1 Motivating example

Figure 4.1 presents an 8-node topology where there exist two flows from N_0 to N_4, and vice versa. In all topologies used in this dissertation, we assume each node can receive packets only from nodes immediately next to it horizontally, vertically, or diagonally. As shown in this figure, N_1's queue contains 2 native packets P_0 and P_2 with different next-hops N_0 and N_2, respectively. Let us assume P_0's next-hop is P_2's previous forwarder or one of its neighbors, and vice versa. So, N_1 decides to mix these packets together, hoping that N_2 (N_0) has already received P_0 (P_2) and it can decode P_2 (P_0). Therefore, N_1 sends a coded packet $P = P_0 \oplus P_2$ to N_0 and N_2 (i.e., next-hop list in the packet header contains N_0 and N_2) while we assume N_6 overhears the packet.

In the previous methods like COPE and BEND, N_6 discards the packet immediately because either it is not the next-hop (as in COPE) or the packet is not a native packet (as in BEND). Here, we assume that N_2 does not receive the coded packet or P_0, so it cannot decode P_2, and that N_6 receives it successfully, and also can decode the packet. In such a scenario, in previous methods, after a time-out, N_1, which has not heard any ACK from N_2, retransmits the packet. However, FlexONC avoids such unnecessary retransmissions,

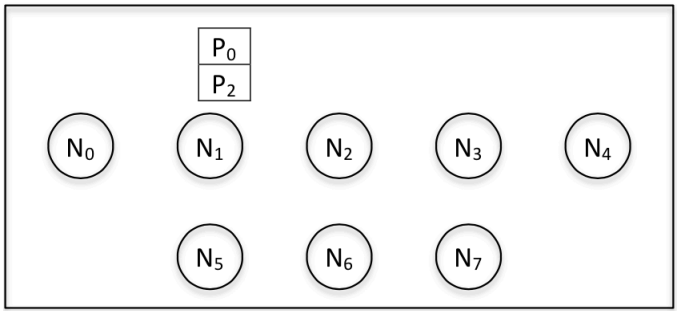


Figure 4.1: Non-intended forwarders can help decoding.

and N_6 forwards the packet to its next-hop on behalf of N_2.

In fact, FlexONC allows non-intended forwarders like N_6 to decode a received coded packet if they can, and forward it toward the final destination as long as the intended forwarder fails to do so. By doing so, since N_2 is not the only node in charge of forwarding packets, the traffic is spread in the network. That is if N_2 fails to receive or decode a packet, its role is immediately covered by N_6. This idea not only can accelerate packet delivery by removing some retransmissions but also can provide more coding opportunities. For example, let us further assume N_6 is going to forward P_2 on behalf of N_2. If P_2 is eligible to be mixed with some packets queued at N_6, by allowing N_6 to decode and forward it, we capture more coding opportunities at N_6. However as will be described later, we provide some strategies to ensure that the nodes do not stray far away from the original route, and also to limit the number of duplicate packets in the network.

4.1.2 Objectives and challenges

FlexONC should avoid unnecessary changes to the standard MAC protocols, and be as simple as possible to be feasible in real scenarios. Moreover, it should be compatible with different routing protocols despite few modifications. To realize such compatibility, while

having more flexibility in forwarding and coding, FlexONC should address the following questions.

- How to select the forwarder set, the nodes that can help the intended forwarder to forward packets: In other words, how should we decide which nodes are eligible? For example, in Figure 4.1, when N_1 sends the packet, N_5, N_2 and N_6 may receive it, but are they good candidates to forward the packet? Which one has the first priority?

- Duplicate packets: Since more nodes cooperate to move packets toward the destination, their imperfect collaboration may cause a significant number of duplicate packets travelling in the network leading to unnecessary contention and collision. Some mechanisms are required to control duplicate packets in the network.

- Flexible forwarding but not too far from the specified route: Although in FlexONC, like BEND, packets may not follow the exact route specified by the routing protocol, we need to keep them around the determined route. To do so, BEND uses the *second-next-hop* field in native packets. However, as we described earlier, it is not applicable to coded packets at non-intended forwarders. For example, in Figure 4.1 when N_6 receives the coded packet, even if it can decode P_2, it does not know the address of next hop from N_2 toward the destination. Thus in FlexONC, to enable N_6 to forward this packet, a new approach is required so that non-intended forwarders can find the correct address of the next hop.

We address all these questions in the next section.

4.2 Implementation Details

As described earlier, the idea behind FlexONC is not only to apply OR but also to have backup nodes which can decode and forward a packet in case that the intended forwarder fails to do so, either due to unsuccessful reception of the packet or lack of required packets in the buffer to decode the original packet. This section describes in detail the responsibility of the sender and the receiver of a coded packet to realize this idea, and answers the questions stated in the previous section.

4.2.1 Decoding and forwarding strategy

In FlexONC, nodes in the network are in promiscuous mode, and store all received and overheard packets in a buffer, called *coding buffer*. Each packet is kept there for a period of time, long enough that the node can use these packets to decode the received coded packets. In case of successful decoding, the receiver sends an ACK while a NACK (i.e., negative acknowledgement) signals failure in decoding. In addition, each node has three transmission queues for intended native packets, overheard native packets, and coded packets. Similar to BEND, coded packets are sent with a higher probability than native packets.

A non-intended forwarder may forward a packet on behalf of an intended forwarder if the intended forwarder fails to do so or the non-intended forwarder can provide more coding opportunities. As a matter of fact, by adjusting the back-off time before packet transmission, FlexONC coordinates the forwarders, and gives a higher priority to the forwarders with more coding gains. While a packet with four or more coding partners waits for the shortest back-off time before transmission, a non-intended forwarder forwarding a

native packet has the lowest priority.

In FlexONC, although packets may not follow the exact route specified by the routing protocol, they travel near it and do not stray too far away. Thus, when a non-intended forwarder forwards the packet on behalf of the intended forwarder, it should send it to the next-hop toward the destination from the intended forwarder's point of view. For example in Figure 4.1, when N_1 sends the coded packet $P = P_0 \oplus P_2$, N_0, N_5, N_2, and N_6 may receive the packet. If N_2, which is the intended next-hop for P_2, fails to receive the packet successfully, and if one of the non-intended forwarders (e.g., N_5, N_0, N_6) wants to forward it, they need to know the address of the next-hop from N_2 toward the destination (not from themselves), which is N_3 in this example.

Since the *second-next-hop* field in BEND cannot solve this problem, instead of adding this field to the packet header, in FlexONC, the routing protocol is enhanced such that each node also maintains forwarding tables of all its neighbors. As such, when for example N_6 forwards P_2 on behalf of N_2, it knows the address of the next-hop from N_2 toward the destination, and simply sends the packet to it.

4.2.2 Receivers in FlexONC

Since every node in the vicinity of the sender can receive the packet, we classify the receivers of a packet in two groups, intended forwarders and non-intended forwarders. As summarized in Table 4.1, an intended forwarder is a node whose address has been specified in the packet header as the next-hop of the packet by the routing protocol. On the other hand, non-intended forwarders are the nodes that are in the neighborhood of the next-hop and can help it in forwarding packets.

When a sender transmits a coded packet, all of its neighbors may receive it. However, every node that receives the packet is not necessarily eligible to forward it. In addition, if all eligible nodes were to forward the same packet, that would be a waste of the network bandwidth as well as a source of collision. We need a method to choose and prioritize eligible forwarders.

A node is an *eligible* non-intended forwarder and is added to the forwarder set if it is not only the neighbor of the sender but also a neighbor of both next-hop and the second next-hop of a coding partner. Following this rule ensures that a packet would travel correctly toward its final destination, even if it is forwarded by a different node than its next-hop. In the rest of the chapter, we use the term "non-intended forwarder" to refer to "eligible non-intended forwarders".

If an intended forwarder (e.g., N_2 in Figure 4.1) receives a coded packet and can decode the packet, it simply replies with an ACK. However, if it cannot decode the packet, it sends a NACK instead. In FlexONC, ACKs and NACKs contain the address of their sender (i.e., the transmitter of ACK/NACK) instead of the receiver, the same as in BEND. If non-intended forwarders (e.g., N_6) hear the ACK, they realize that the intended forwarder has decoded the packet successfully and does not need their help.

In FlexONC, when a node like N_6 in Figure 4.1 receives a coded packet, it first looks for its address in the next-hop list. If it cannot find its address, clearly it is not the intended forwarder for any coding partner in the coded packet. Therefore, N_6 searches for a native packet in the coded packet that 1) its intended forwarder (e.g., N_2 for P_2 in Figure 4.1) is N_6's neighbor, 2) its next-hop from the intended forwarder (e.g., N_3 for P_2 in Figure 4.1) is N_6's neighbor, and 3) it is decodable by N_6. Based on these criteria, in Figure 4.1, although when N_1 sends the coded packet P, N_0, N_5 and N_6 as well as N_2 may

receive the packet, N_0 is not eligible to forward P_2 due to the first criterion. Furthermore, N_5 is not qualified for the second criterion, and therefore N_6 is the only non-intended forwarder which can send P_2 on behalf of N_2 if it can decode it.

However, a non-intended forwarder should not forward a packet immediately after decoding it because the intended forwarder may forward the packet itself and would not need the non-intended forwarders' help. In addition, if there are more than one eligible non-intended forwarder, an ordering among them is required to avoid the transmission of more than one ACK to the packet sender. Due to this reason, in FlexONC the sender adds the index of all eligible non-intended forwarders to the packet header.[1] Specifically, when a non-intended forwarder receives a coded packet, it sorts the list of indexes (i.e., all non-intended forwarders), gives the first priority to the intended forwarder of the decoded packet, and considers its index in the sorted list as its rank. Then, it sets a timer and waits for an ACK from any node with a higher rank. If it does not hear any ACK after time-out, it is likely that none of the nodes with a higher rank has received and can forward the packet, so it is its turn to send the ACK back to the sender, mixes possibly the decoded packet with other packets in the queue, and forwards it. Figure 4.2 presents the flowchart for receivers of a coded packet in FlexONC.

4.2.3 Senders in FlexONC

When a node sends a coded packet, it adds the list of the next-hops of all coding partners to the packet header. Thus when each next-hop receives the packet, it does not send the acknowledgement (either ACK or NACK) immediately but after some time proportional

[1]We assume that all nodes in the network agree on the same numbering system which represents each of them with a unique index known by all other nodes.

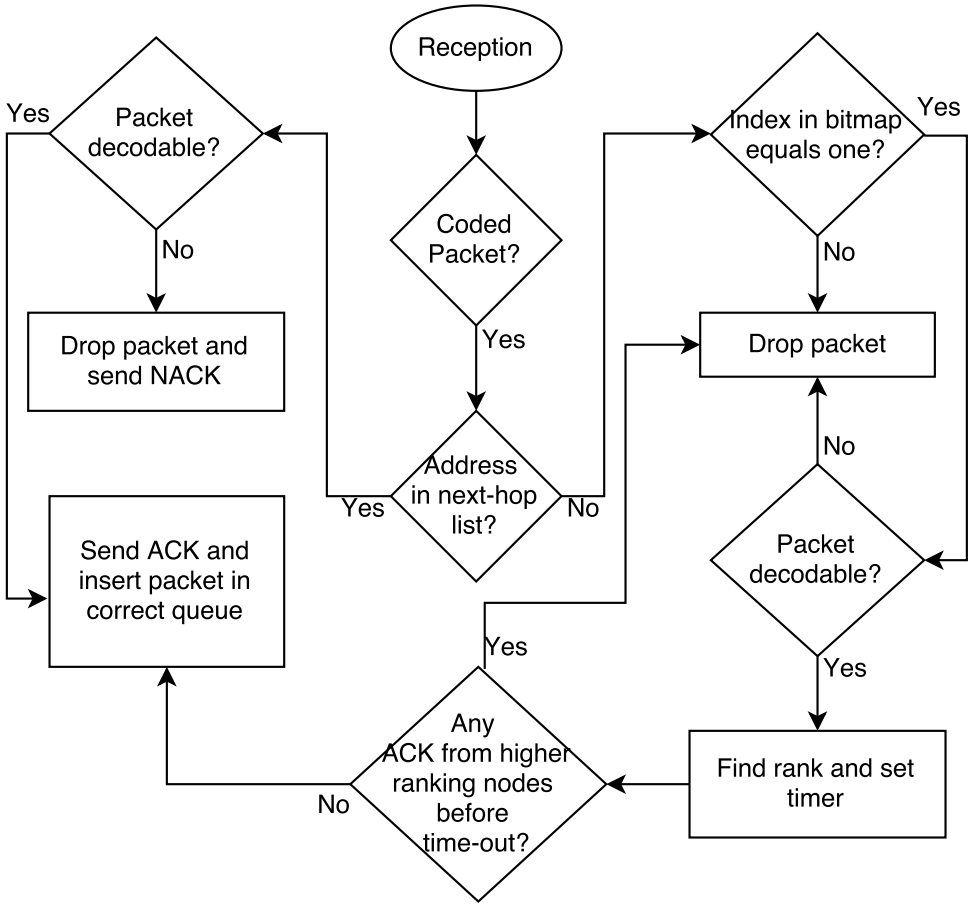


Figure 4.2: Flowchart for receivers of coded packets in FlexONC.

to its position in the next-hop list plus the transmission and propagation time of the acknowledgement. For example, if a node transmits the combination of 3 packets with the next-hops N_1, N_2 and N_3, after receiving the coded packet, N_3 waits for a certain amount of time to ensure that N_1 and N_2 have sent their packet acknowledgements, and then N_3 sends back ACK/NACK.

Furthermore, the sender detects all eligible non-intended forwarders of a coded packet, and adds a bitmap to the packet header, where each bit represents one of the nodes in the network (as discussed in Subsection 4.4.7, the overhead introduced by adding this bitmap

is less than a few bytes). If the node is an eligible forwarder, the corresponding bit is set to 1, otherwise the bit keeps the default value, which is 0. We assume that each node is represented with a unique index known by all other nodes, and each node ranks eligible non-intended forwarders based on their indexes.

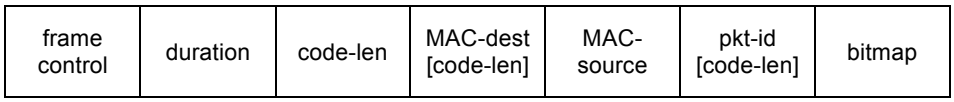


Figure 4.3: MAC header for coded packets.

In FlexONC, the bitmap, representing the forwarder set, is added to the packet header packets. In addition, the MAC-layer header of coded packets includes some additional information, such as the number of coding partners, and the address of the next hop and the packet-id of all coding partners as presented in Figure 4.3. Similar to BEND and COPE, the packet-id is generated by creating a 4-byte hash value out of the source's IP address and the sequence number carried by the packet. Note that we keep the original format of the upper layers' headers, and the *XOR* of the coding partners is added to the MAC data-frame as payload.

Since the sender stores the forwarding table of its neighbors, it can check which neighbors are eligible non-intended forwarders, as shown in Figure 4.4. Doing so, the sender can calculate its maximum waiting time for receiving an ACK, which is proportional to the number of the next-hops (i.e., intended forwarders) and eligible non-intended forwarders of coding partners. It is obvious that when a sender sends a combination of n packets, it should wait to receive n ACKs. Thus, its waiting time before time-out is more than when it transmits a native packet. In FlexONC, because more nodes can help in decoding and

```
FindFwds(N_s,p):
//Node N_s sends packet p

  N_s looks at its forwarding table and finds the next-hop NH(P)

  N_s looks at the forwarding table of its neighbour NH(p) and
  finds the second-next-hop NH2(p)

  For all x ∈ ng(N_s)
     If x ∈ ng(NH(p))
          If NH2(p) ∈ ng(x)
              N_s Adds x to the list of eligible non-intended
              forwarders
```

Figure 4.4: Pseudo-code of finding eligible non-intended forwarders when node N_s sends packet p. NH(p) and NH2(p) denote the next-hop and the second-next-hop of packet p, respectively. Also, ng(N) represents the set of neighbors of node N.

forwarding a packet, if the sender does not hear an ACK from the intended forwarder, there is still a chance that it receives the ACK from a non-intended forwarder. Therefore, the sender should wait a little longer before it retransmits the packet. As such, in FlexONC the waiting time of the sender for coded packets is calculated in terms of the number of both coding partners and eligible non-intended forwarders.

To illustrate the idea in more details, let us assume that in Figure 4.1, N_2 mixes two native packets and forwards the coded packet to the next-hops N_1 and N_3 (i.e., N_1 and N_3 are the intended forwarders of these two packets), while N_5 and N_7 are eligible non-intended forwarders specified in the bitmap. Figure 4.5 shows the maximum waiting time at the sender, N_2, after transmitting the data packet and the time-window dedicated to the intended and non-intended forwarders to reply if they need. Note that the intended forwarders reply by an ACK after successful decoding and send a NACK after decoding failure. In addition, a non-intended forwarder replies by an ACK only if decoding is successful and no ACK was heard from neither the corresponding intended forwarder nor

higher-ranking non-intended forwarders.

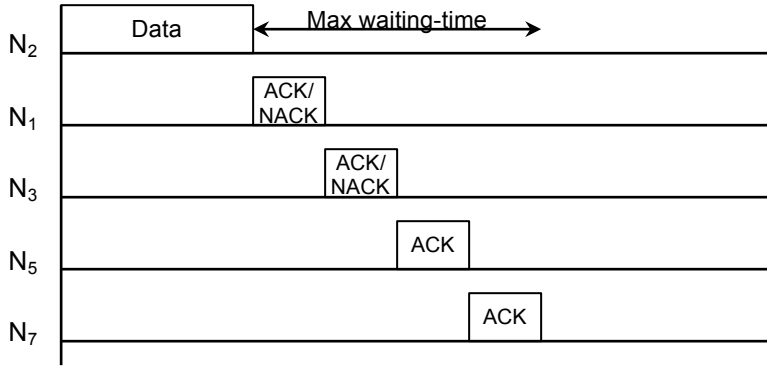


Figure 4.5: The time-window dedicated to different nodes to send back the acknowledgment, where in the topology depicted in Figure 4.1 N_2 transmits a coded packet to the next-hops N_1 and N_3, and N_5 and N_7 are non-intended forwarders.

When the sender receives an ACK for a packet, it removes the packet from its transmission queue; it may still keep it in the coding buffer for decoding purposes. On the other hand, when the sender receives a NACK for the sent packet, it keeps waiting until either time-out or receiving an ACK for the same packet. In the case of time-out for native packets, the sender retransmits the same packet if the number of transmissions does not exceed the maximum retransmission count. However, for coded packets, if the node receives ACKs or NACKs for none of the coding partners, it retransmits the same coded packet. Otherwise, it inserts the coding partners which are not ACKed in the transmission queue.

4.2.4 How to limit the number of duplicate packets?

Although FlexONC aims to eliminate duplicate packets by prioritizing non-intended forwarders and making the sender wait for their ACKs, duplicate packets may still exist in the network, due to various reasons such as lack of perfect synchronization. For example, a non-intended forwarder may not hear the ACK sent by the intended forwarder or higher-ranking non-intended forwarders, and transmit the packet unnecessarily. Therefore, FlexONC relies on more strategies to control the number of duplicate packets in the network.

First, after receiving an ACK for a given packet-id, if the node finds a packet with the same packet-id in its transmission queue that the sender of the ACK is the next-hop of the packet or one of corresponding eligible non-intended forwarders, the node drops the packet (i.e., the packet has already been received by down stream nodes). Second, in FlexONC each node stores a limited number of received ACKs, and if it receives a packet, it searches this ACK list. If it finds an ACK for the same packet sent by its next-hop or one of its eligible non-intended forwarders, it also drops the packet. Third, if the node overhears a packet, which is already in its transmission queue, from the next-hop or one of corresponding eligible non-intended forwarders of the packet, the node cancels the transmission of the packet.

4.3 Performance Evaluation

We use the Network Simulator (NS-2) to compare the performance of FlexONC against the non-coding scheme, a simulation version of COPE as a prominent research on network

Table 4.2: Information available at nodes in different schemes.

Information	**Non-coding**	**COPE**	**CORE**	**BEND**	**FlexONC**
next-hop	√	√		√	√
second next-hop				√	
neighbors' forwarding info					√
forwarder set			√		√
node's geo-position			√		

coding, and two OR schemes in IXNC (i.e., BEND and CORE).[2] Table 4.2 summarizes the type of information provided at nodes in different schemes. The rest of this section describes the experiment scenarios as well as the performance results in two different topologies.

4.3.1 Settings

To study the performance under different link qualities and packet loss probabilities in our simulations, bit error rate (BER) is added to the physical layer. In fact, even if the signal strength of a received packet is higher than reception threshold, the packet may still be dropped with a probability calculated in terms of BER. BEND and CORE also use a similar physical layer model. The channel propagation used in NS-2 is a two-ray ground reflection model [80], and the maximum transmission range is 250 m. The data rate is fixed to 1 Mbps. The sources, in our simulation scenarios, send CBR (constant bit rate) data flows with a datagram size of 1000 bytes. Also, we use DSDV (Destination-Sequenced Distance Vector) [77] as the routing protocol and apply a few minor changes

[2]Note that in all simulations, IEEE 802.11 [1] is selected as the data link layer signaling method.

so that each node can obtain forwarding tables from its neighbors.

To investigate the performance of FlexONC in comparison to BEND, CORE, COPE and the non-coding scheme, we test them in different scenarios and compare their throughput as well as the throughput gain of FlexONC over the baselines for different BERs in two topologies. First, we compare them using a simple 8-node topology shown in Figure 4.1, and then we use a 5×5 grid topology as a more general case.

4.3.2 8-Node topology

In the 8-node topology presented in Figure 4.1, two flows in opposite directions transmit packets from N_0 to N_4 and vice versa. Since the distance between adjacent nodes in both X and Y axes is 150 m, each node can receive packets only from nodes immediately next to it horizontally, vertically, or diagonally (e.g., N_1 can hear from N_0, N_5, N_2, and N_6). The inter-arrival time of CBR flows in these scenarios is 0.07 s and its duration is 150 s.

In this topology, for each intended forwarder except for the destination, there exists at least one non-intended forwarder that can help the intended forwarder and forward packets when the intended forwarder fails to do so. Regarding CORE, it means that at least two nodes can be chosen in the forwarder set of each packet. Figure 4.6 presents the throughput of BEND, CORE, COPE, non-coding and FlexONC for three lowest BERs in our experiments.

We observe that when BER $= 2 \times 10^{-6}$ (i.e., the network condition is almost perfect), most transmitted packets are received by the intended forwarders successfully. Therefore, there hardly exists an opportunity for non-intended forwarders to decode and forward a packet on behalf of the intended forwarder. It is obvious that in such a situation,

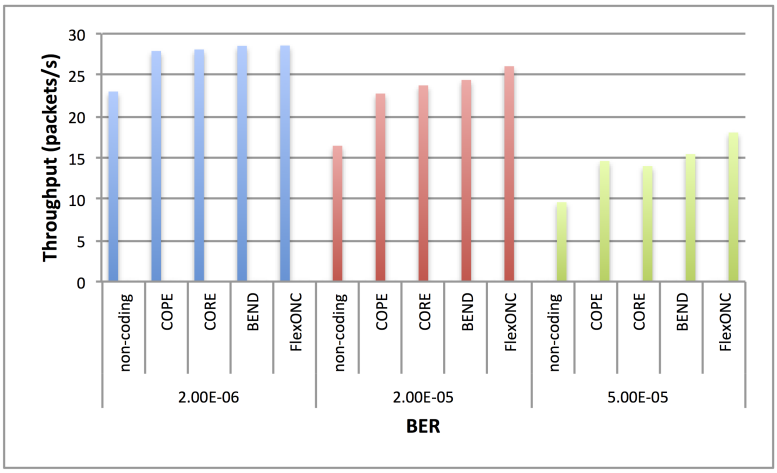


Figure 4.6: Throughput of different methods in 8-node topology for different BERs.

FlexONC does not show its real power and its throughput is close to BEND. However, as the BER increases, more opportunities for non-intended forwarders are provided and FlexONC's gain over other methods increases significantly.

Furthermore, Figure 4.7 presents the performance gain of FlexONC over BEND, CORE, COPE and non-coding for 6 different BER levels, which corroborates our observation. In particular, by increasing the BER, FlexONC becomes more powerful in comparison to the baselines, and its throughput gain increases. The throughput gain of FlexONC over each baseline is calculated as:

$$\text{throughput gain} = \frac{Tr(\text{FlexONC}) - Tr(\text{baseline})}{Tr(\text{baseline})} \times 100 \tag{4.1}$$

where $Tr(x)$ denotes the calculated throughput for scheme x.

As shown in these figures, although at lower BER, CORE's performance is very close to FlexONC's, in lossy networks FlexONC outperforms CORE due to the following reasons. First, in this topology with a small forwarder set, at high BERs many packets are lost without being received by any forwarder. Second, in CORE packets are broadcasted

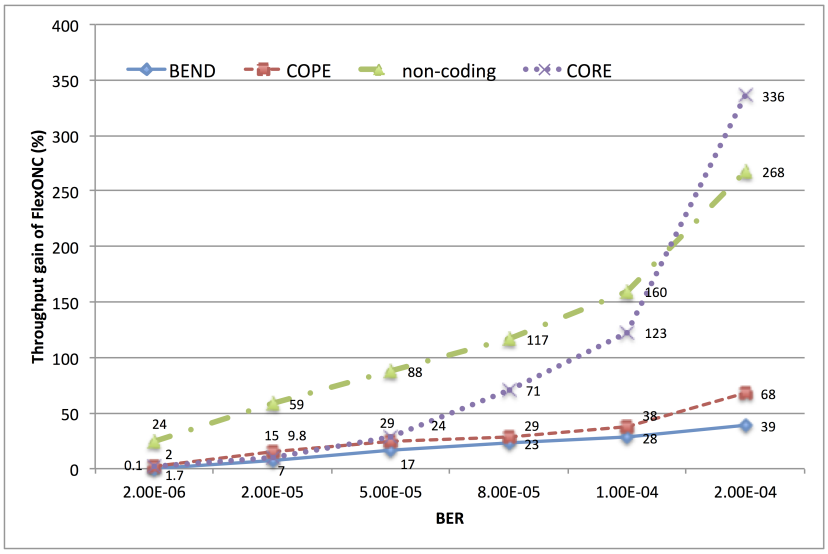


Figure 4.7: FlexONC's gain over other methods in 8-node topology.

without any retransmission mechanism to compensate for packet loss.

4.3.3 Grid topology

To investigate the performance of FlexONC in a general topology, we test it in a 5×5 grid, where again the distance between two adjacent nodes is 150 m. 8 different flows with an inter-arrival time of 0.1 s and duration of 150 s transmit packets between Row 2 and Row 4, and also Column 2 and Column 4 of the grid, as shown in Figure 4.8.

The performance results depicted in Figure 4.9 and Figure 4.10 again show that at non-trivial BER levels, FlexONC almost always outperforms other methods. In perfect network conditions (BER $= 2 \times 10^{-6}$), CORE performs slightly better than FlexONC because there is no intended forwarder in CORE, and it distributes packet transmissions more evenly than FlexONC among possible forwarders. However, as explained earlier, in lossy environments CORE cannot benefit from OR and network coding as much as

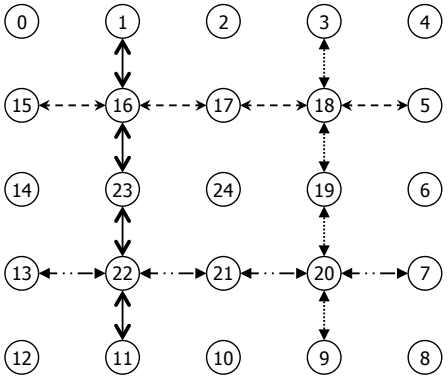


Figure 4.8: 5×5 grid topology with 8 flow.

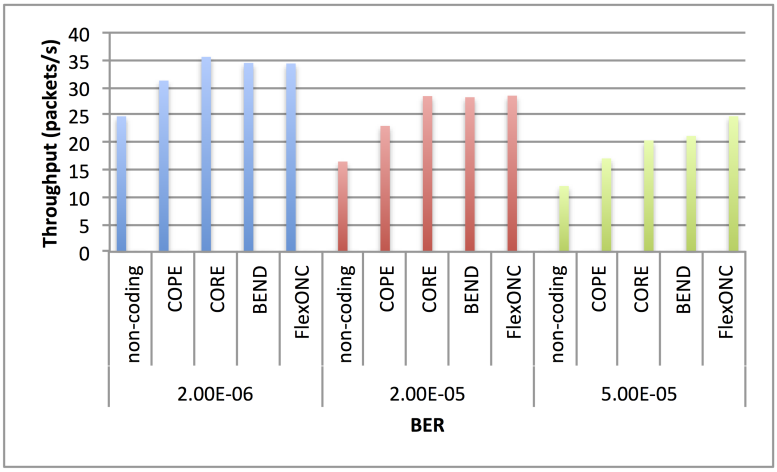


Figure 4.9: Throughput of different methods in the grid topology for different BERs.

FlexONC due to the lack of any retransmission mechanism, especially in such multi-hop routes (i.e., each node should pass at least 4 hops to be delivered to the destination).

In addition, one may notice that by increasing the BER, the throughput gain of FlexONC over CORE increases faster in the 8-node topology in comparison to the grid topology. In fact, the larger forwarder set in the grid topology decreases the probability of packet loss in each transmission.

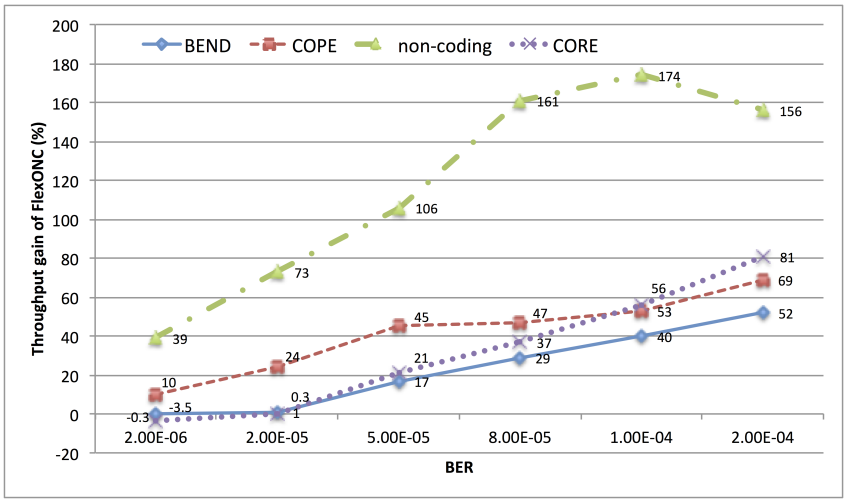


Figure 4.10: FlexONC's gain over other methods in the grid topology.

4.4 Discussion

4.4.1 Routing protocol

In our experiments, we selected DSDV as the routing protocol for its well-known behavior. Moreover, it is a distance-vector approach that makes fewer assumptions about the routing information in comparison to source routing protocols. Therefore, if FlexONC works well with DSDV, it will work with source routing protocols as well. As a matter of fact, choosing DSDV as the routing module does not lose generality of our scheme in a stationary mesh network. We believe choosing any other routing protocol would not make a big difference in FlexONC's performance gain, as long as the routing protocol can be modified in a way that each node contains forwarding information for its neighbors.

4.4.2 End-to-end delay

On one hand, FlexONC decreases the delay in forwarding packets and increases the throughput by avoiding packet retransmission when an intended forwarder fails to decode the coded packet, and a non-intended forwarder alternatively passes the packet toward the destination. On the other hand, when more nodes have the responsibility of passing the packet further to the destination, in case of retransmissions, the sender should wait longer for an ACK before it retransmits the packet, and this longer waiting time means longer delay which may lead to a lower throughput.

Therefore, we face a trade-off here. While the maximum waiting time of the sender is proportional to the number of eligible forwarders, the gain of FlexONC is also related to the number of neighbors of the sender (i.e., more precisely, eligible non-intended forwarders), as well as the probability of intended forwarder's failure in receiving or decoding a coded packet, which is in turn affected by the packet loss probability and BER in the network. The performance results showed that even for a very low BER, when the intended forwarder itself can decode and forward the majority of received coded packets and FlexONC does not have much chance to be applied, its performance is comparable to BEND's performance or even better.

Figure 4.11 shows the average end-to-end delay of delivered packets in different methods, for the scenario described in Subsection 4.3.2. While the non-coding scheme has the highest average end-to-end delay, the delay in FlexONC is slightly longer than BEND. As explained earlier, the most important reason of this longer delay is that the sender of coded packets in FlexONC waits longer to receive an ACK than in BEND. Therefore, if the packet transmission fails and no ACK is received, BEND's timer, for anticipated ACKs,

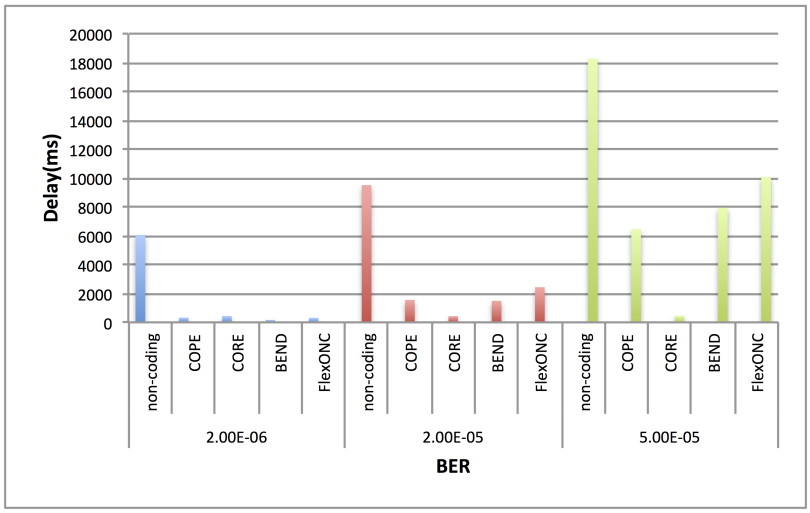


Figure 4.11: End-to-end delay of different methods in 8-node topology for different BERs.

usually expires earlier than FlexONC's, leading to a faster retransmission in BEND, which can reduce its average end-to-end delay in comparison to FlexONC.

In addition, one may notice that in CORE the end-to-end delay does not vary much over different BERs. While at lower BERs, CORE's delay is longer than that of other coding schemes, at higher BERs its delay is significantly shorter than that of other protocols. The main reason of this shorter and almost constant delay in delivery is the lack of any retransmission mechanism; any packet either is delivered by one transmission or is dropped.

As shown in Figure 4.11, the delay in the non-coding scheme is significantly higher than other methods. The main reason is that coding enables free-riding. In other methods, more than one packet can be combined and sent simultaneously, which means that packets can free-ride on other packets. Therefore, the packets are forwarded faster. In addition, this decreases the queue length at nodes, causing shorter waiting time and consequently shorter delay.

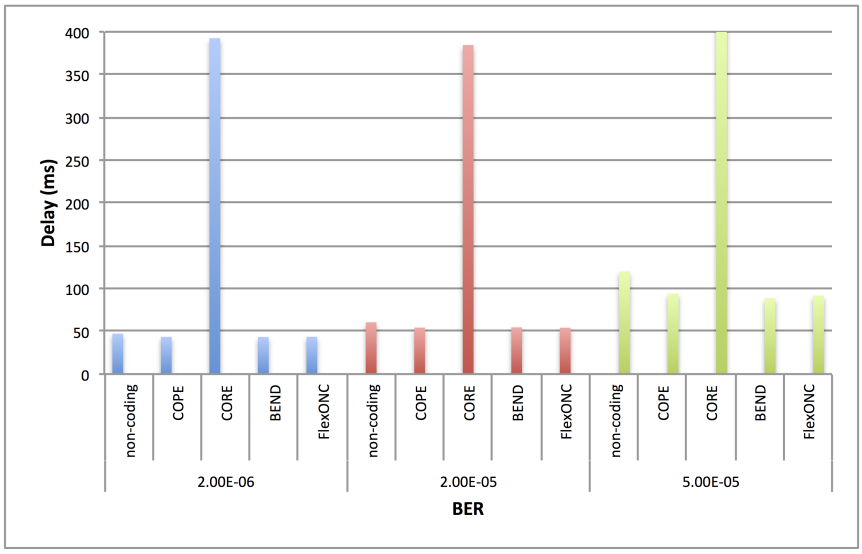


Figure 4.12: End-to-end delay of different methods in 8-node topology for different BERs with less CBR traffic.

To verify this explanation we repeat simulations with less CBR traffic with the inter-arrival time of 0.15 s (instead of 0.07 s). By increasing the inter-arrival time, fewer packets are injected to the network per second, which reduces the probability of having more than one packet in the queues, and in turn, creates less coding opportunities at nodes. The results are shown in Figure 4.12, where the delay in non-coding is comparable to the other methods, as the coding schemes provide less free-riding opportunities for the packets.

Furthermore, while this figure justifies the almost constant end-to-end delay in CORE over different BERs, it also shows that the delay in CORE is significantly longer than that of other methods. As mentioned earlier, in this scenario with a small packet arrival rate, the coding opportunities are rare in the network, and most packets are sent natively. To provide higher priority for coded transmissions in CORE, the native packets are delayed before transmission; therefore, forwarding a large number of native packets in this scenario increases the end-to-end delay significantly.

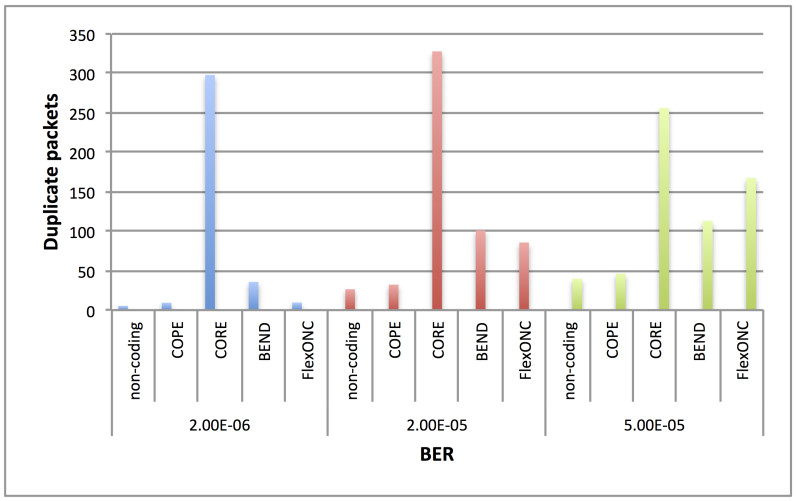


Figure 4.13: Duplicate packets of different methods in 8-node topology for different BERs.

4.4.3 Duplicate packets

As explained in [106], since in BEND more nodes cooperate in forwarding packets toward the final destination, it is prone to generating more duplicate packets in case of imperfect collaboration among nodes. The situation in FlexONC could seem even more severe, as it allows non-intended forwarders to cooperate in more ways (i.e., forwarding of not only received native packets, but also received coded packets). To control duplicate packets in FlexONC, we introduced some mechanisms in Subsection 4.2.4.

Figure 4.13 shows the number of duplicate packets generated by different methods. As shown in this figure, the largest number of duplicate packets are generated at CORE, as nodes should only rely on overhearing other transmissions to avoid duplicate packets. In addition, while the number of duplicate packets in BEND is higher than non-coding and COPE, FlexONC is able to control the number of duplicate packets, especially at lower BERs. The reason could be related to the additional mechanisms introduced in FlexONC to control the number of duplicate packets. However at higher BER=5×10^{-5}, there are

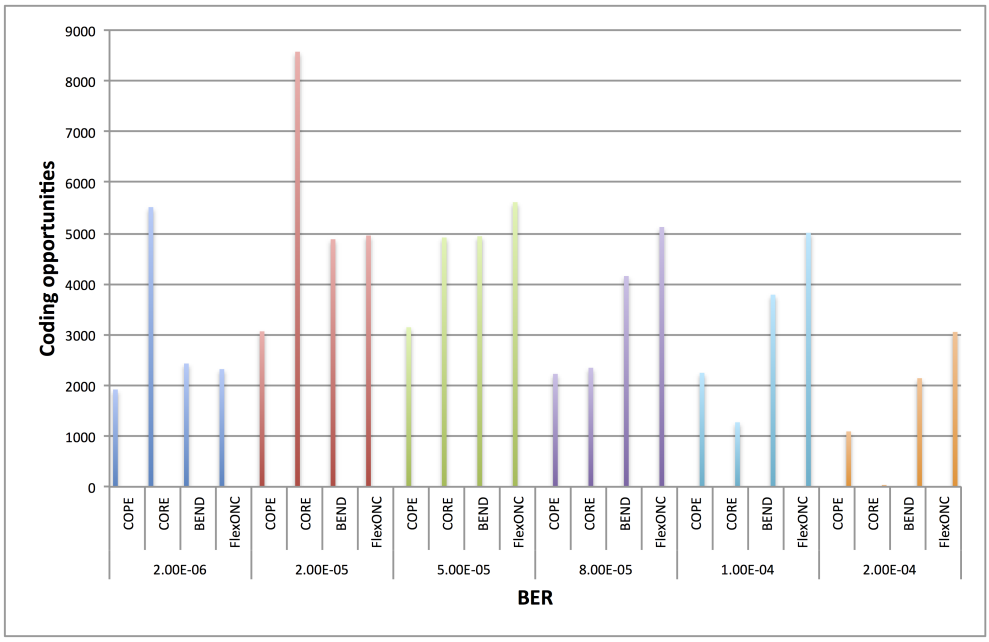


Figure 4.14: Coding opportunities in different methods in 8-node topology for different BERs.

more duplicate packets in FlexONC than in BEND because these mechanisms are highly susceptible to the reception of ACKs and at higher BERs the probability of losing ACKs increases.

4.4.4 Coding opportunities

As shown in Figure 4.14, at lower BERs the code opportunities at CORE are more than that of FlexONC. However, at higher BERs, FlexONC provides more coding opportunities than other schemes. One may notice that, by increasing BER, first coding opportunities in all methods increases. The reason is that, due to a greater need for retransmission, packets stay longer in the queue and the chance of combining them with the packets of other nodes increases, leading to more coding opportunities. On the other hand, when

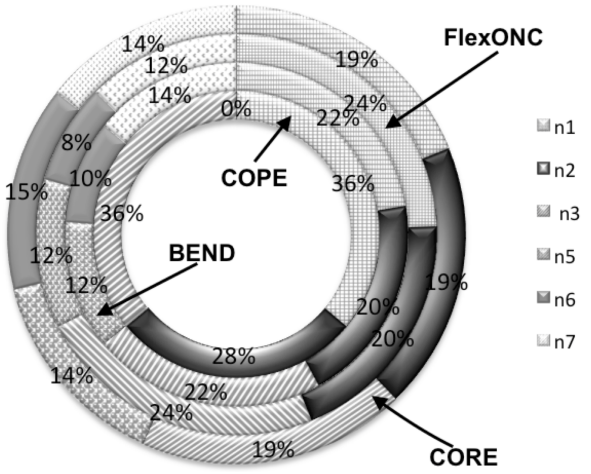


Figure 4.15: Distribution of coding opportunities at different nodes in different methods in 8-node topology.

BER further increases, the number of retransmissions increases significantly; therefore the probability of generating new coding opportunities decreases. That is why for BERs higher than 5×10^{-5} the coding opportunities in the networks drops. In CORE, although there is no retransmission, at higher BERs and in this topology many packets can not go further than one or two hops, which decreases the number of packets in nodes' queues as well as the number of coding opportunities.

To show the distribution of coding opportunities at different nodes, we run simulations using the topology depicted in Figure 4.1 and the scenario explained in Subsection 4.3.2, but the route between N_0 and N_4 is fixed through N_1, N_2 and N_3 for COPE, BEND and FlexONC (i.e., the intended forwarders are N_1, N_2 and N_3). As shown in Figure 4.15, coding opportunities in COPE are restricted to the intended forwarders; however, other coding schemes use non-intended forwarders (i.e., N_5, N_6 and N_7) to accelerate packet forwarding and provide more coding opportunities. In addition, since in CORE there is no intended forwarder, and possible forwarders are prioritized only based on coding

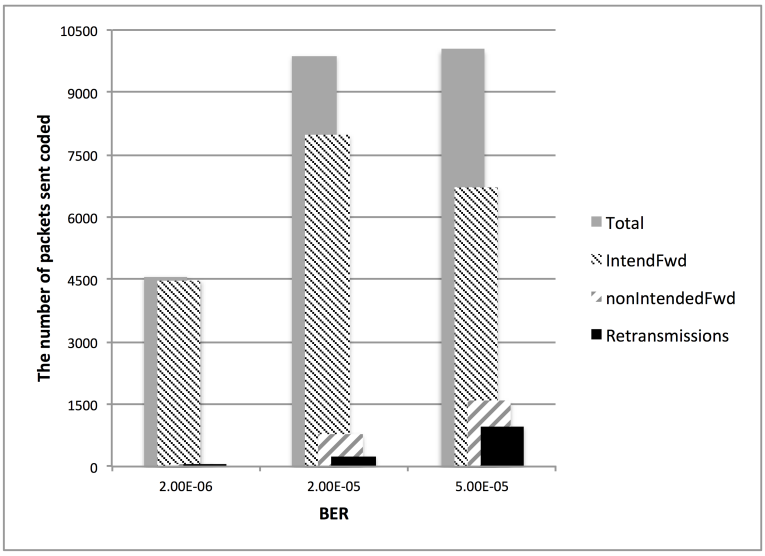


Figure 4.16: What happens to coded packets when BER changes.

opportunities, the coding opportunities are distributed more evenly in CORE than in other coding schemes.

4.4.5 What happens to coded packets in FlexONC?

To show why by increasing BER FlexONC outperforms other schemes in throughput, we run simulations using the scenario depicted in Subsection 4.3.2, and calculate: 1) the total number of coded packets sent, 2) the number of coded packets received and forwarded by the intended forwarder, 3) the number of coded packets only received and forwarded by one of the non-intended forwarders (i.e., on behalf of the intended forwarder), and 4) the number of coded packets for which the sender does not receive any ACK (or NACK) and retransmits.

As shown in Figure 4.16, by increasing BER, intended forwarders receive a smaller percentage of total coded packets sent, and the portion of coded packets which are received

only by non-intended forwarders increases. This means that non-intended forwarders can cooperate more effectively in forwarding and be more beneficial. This collaboration among nodes, which increases at higher BER, is the key idea of FlexONC, which leads to increased robustness and higher packet delivery rate in comparison to the baselines.

4.4.6 Packet delivery rate

OR is utilized to increase the probability of successful delivery of a packet as more nodes can help in forwarding packets. In this subsection, we investigate the effect of the number of nodes in the forwarder set, and the link quality on the performance of OR protocols, especially BEND and FlexONC, for both native and coded packets. We focus on the case with no retransmission first, and the case with retransmission is a natural extension, as we see later. Also, we assume that the nodes in the forwarder set have a perfect coordination mechanism, which means that all nodes in the forwarder set know which one of them forwards the packet.

Let us denote p as the probability of successful transmission at each link, and N as the average number of nodes in the forwarder set. Then, the probability of successful transmission of a native packet to at least one of the nodes in the forwarder set equals: $p_f^n = 1-(1-p)^N$. If a packet traverses H hops in average to be delivered to the destination, in each transmission $N-1$ non-intended forwarders help the intended forwarder except for the transmission to the destination. Then, the probability of successful delivery to the destination can be calculated as: $p_d^n = (1-(1-p)^N)^{H-1} \times p$. It is worth noticing that for $N = 1$ (i.e., only one node in the forwarder set of each transmission), $p_d^n = p^H$, which is basically the probability of successful delivery of a packet in traditional forwarding

with H hops. Furthermore, when N increases, $p_d^n > p^H$, which shows that by increasing the number of non-intended forwarders (i.e., the nodes in the forwarder set) the packet delivery rate increases.

Regarding coded packets, a received coded packet with m coding partners is decoded successfully if $m-1$ coded partners have already been received. Therefore, the probability of delivery of a coded packet to the next-hop equals p^m. As discussed earlier, in BEND coded packets are only forwarded by the intended forwarder (i.e., no OR). Therefore, the probability of delivery of a coded packet with m coding partners to the destinations in BEND equals $p_d^c(\text{BEND}) = (1-(1-p)^N)(p^m)^{H-1}$, given that the source always sends native packets. On the other hand, since FlexONC extends OR to coded packets as well, the probability of delivery of coded packets to the destination in FlexONC equals: $p_d^c(\text{FlexONC}) = (1-(1-p)^N)(1-(1-p^m)^N)^{H-2}p^m$.

To compare the delivery rate in BEND and FlexONC, we focus on the delivery of coded packets, which is different in these two approaches. Assuming that the coding opportunities in both protocols are similar, when the number of non-intended forwarders (i.e., N) increases, $p_d^c(\text{FlexONC})$ increases faster than $p_d^c(\text{BEND})$, which shows that the gain obtained by OR is greater in FlexONC than in BEND. Furthermore, when the link quality is perfect (i.e., $p = 1$), the packet delivery rate for both protocols is the same and independent of N, justifying the observation that in perfect network conditions OR is not beneficial. However, as shown below, in imperfect link qualities (i.e., $p < 1$), FlexONC

outperforms BEND.

$$
\begin{aligned}
& 0 < p < 1 \\
\Rightarrow \quad & 0 < p^m < 1 \\
\xRightarrow{N>1} \quad & (1 - p^m) > (1 - p^m)^N \\
\Rightarrow \quad & (1 - (1 - p^m)) < (1 - (1 - p^m)^N) \\
\Rightarrow \quad & (p^m)^{H-2} < (1 - (1 - p^m)^N)^{H-2} \\
\Rightarrow \quad & p_d^c(\text{BEND}) < p_d^c(\text{FlexONC}).
\end{aligned}
$$

In addition, we can prove in a similar fashion that the performance gain of FlexONC over BEND, in terms of packet delivery rate, increases as the link quality decreases. Furthermore, when retransmission is enabled, since $p_d^c(\text{FlexONC}) > p_d^c(\text{BEND})$, each coded packet in FlexONC needs fewer retransmissions to be delivered to the destination, which increases the capacity of the network, and consequently improves the performance.

4.4.7 Overall comparison

In this subsection, we provide an overall comparison of FlexONC with other methods, especially BEND, in terms of required storage, packet overhead, computational complexity, delay and throughput. FlexONC provides more coding opportunities, and outperforms other schemes in terms of throughput, especially at higher BERs. Even though having a more powerful protocol may imply increased complexity and overhead, this is not the case of FlexONC, and it is able to keep other metrics such as the end-to-end delay and the number of duplicate packets comparable to other methods, particularly BEND.

Regarding the packet header overhead, while BEND adds the *second-next-hop* field to

the packet header of native packets (i.e., four bytes), FlexONC does not need this field. Instead, it adds a bitmap to the packet header to specify eligible forwarders, which is the case in CORE as well. Given the total number of nodes N in the network, the array needs N bits in the packet header, which does not exceed a few bytes in average. Furthermore, to find the forwarder set in each node, CORE adds the geographical-position of the sender and the final destination of each packet to its header, which is not required by FlexONC.

On the other hand, COPE needs neither the second-next-hop field nor the bitmap since it does not benefit from OR. Moreover, in FlexONC as well as all other OR protocols with network coding (e.g., CORE and BEND), all nodes are in promiscuous mode, and store overheard (in addition to intended) packets. Therefore, this overhead is common in all mentioned baselines except for COPE. In fact, in all experiments over different methods, nodes have the same buffer size.

As explained earlier, in FlexONC, in contrast to COPE, CORE and BEND, each node stores the forwarding information of its neighbors. This information is used to control the route followed by packets and prevent them from straying too away from the designated shortest path. If K denotes the maximum number of neighbors of a node in the network, and each entry of the forwarding table needs at most 10 bytes, the total memory required to store the forwarding information of the neighbors equals $10 \times K \times N$ bytes. Thus, in a network with about 30 nodes, even if we assume all nodes are connected to each other, the total required storage is less than 9 KB. On the other hand, while in BEND each node only stores its own forwarding table, the size of this forwarding table is greater than a regular forwarding table, as it stores the IP addresses of the *second-next-hops* in addition to the next-hops themselves.

All mentioned schemes need to utilize a routing protocol except for CORE as it broad-

casts the packets. However, this broadcasting mechanism and lack of retransmission affects the performance of CORE significantly in lossy networks, as shown in the last section. Having routing information of the neighbors in FlexONC only requires adding one extra field to the route advertisement messages of a proactive routing protocol to include the next hop leading to each destination. However, this very small additional routing overhead is not limited to FlexONC; BEND also adds the same field to the route control packets to update *second-next-hop* field in the forwarding table of each node.

Regarding the computational complexity, the most important processes are encoding and decoding which are almost the same in all coding schemes except for CORE. While in FlexONC and other mentioned coding schemes nodes encode the packets in advance immediately after reception, in CORE a packet is encoded when it is about to be transmitted. In addition, to increase the coding gain in lossy environments, CORE introduces a more complicated encoding algorithm in which each node checks all possible coding patterns of the first K packets in its queue.

In terms of the average end-to-end delay, as explained in Subsection 4.4.2, the delay in FlexONC is slightly longer than that in BEND because of the longer maximum waiting time before triggering retransmission of coded packets. Compared to CORE, at lower arrival rates the delay in CORE is significantly longer than that of FlexONC, since CORE delays native transmissions. On the other hand, at higher arrival rates the delay in FlexONC is longer.

4.5 Summary

This chapter presented FlexONC (Flexible and Opportunistic Network Coding), a joint OR and IXNC approach, which provides more flexibility and coding opportunities in the network, especially in poor channel quality and lossy networks. By utilizing the broadcast nature of wireless networks, FlexONC is able to spread different flows better than former studies such as BEND, and enable a higher level of cooperation between intended and non-intended forwarders at the link layer in a multi-hop wireless network.

The performance results show that at higher BERs, when an intended forwarder will more likely fail to receive packets or decode coded packets, and needs its neighbor's help, FlexONC significantly outperforms previous methods like BEND, CORE, COPE, and non-coding scheme. Even under an ideal network condition, when intended forwarders usually do not need any help and can decode and forward received coded packets, FlexONC can perform as good as other protocols. In fact, the results show that the combination of IXNC and OR if realized carefully can boost the performance of the network significantly, and address the challenges of applying network coding in lossy networks.

Chapter 5

Finding the Precise Coding Conditions for Network Coding

Almost all IXNC methods, which mix packets within a two-hop region, follow a similar set of coding conditions to encode packets. We call this set "common coding conditions". Based on these coding conditions, given a high delivery probability between nodes, two packets are combined if the next-hop of each packet is the previous hop of the other packet or one of the neighbors of the previous hop.

However, in some scenarios as shown in this chapter, the common coding conditions may decide incorrectly to mix some packets that cannot be decoded at the next-hops. This wrong encoding causes failures in decoding, increases the number of required retransmissions to deliver the packets, and consequently decreases the network throughput. Therefore, we enhance FlexONC, proposed in Chapter 4, with an additional coding condition to find coding opportunities more accurately, and design a mechanism to apply these coding conditions appropriately and limit decoding failures. Table 5.1 presents

Table 5.1: Definition of some terms and symbols used in this chapter.

Term/Symbol	**Definition**
decoded-native packet	a native packet which was received coded and has been decoded
coding partner	each native packet encoded with other packets
common coding conditions	the conditions used by previous methods (e.g., COPE and BEND) to combine packets
coding node	a node in which coded packets are generated
$p_{i,j}$	the probability of successful transmission from N_i to N_j
$P_c^{(i)}$	the probability that node N_i sends a coded packet
$P_n^{(i)}$	the probability that node N_i sends a native packet
$P_f^{(i)}$	the probability that N_i drops a packet because of the coding condition problem
$P_{\text{common}}^{(i)}$	the probability of successful packet reception at N_i under common coding conditions
$P_{\text{recoding}}^{(i)}$	the probability of successful packet reception at N_i after adding our *RecodingRule*

some terms and symbols used in this chapter.

5.1 Basic Idea

5.1.1 Encoding decisions

In IXNC, an intermediate node combines two packets if the next-hop of each packet has already received the other coding partner. To keep track of the packets received by each node, two types of information can be used: deterministic information and probabilistic information. Deterministic information are provided by exchanging "reception reports" among nodes, where each node's reception report contains the packets that have recently been received or overheard by the node [48]. These reception reports are usually piggy-backed on data packets or broadcasted periodically.

In the absence of deterministic information (e.g., when a node does not transmit any data packet and only relies on periodic updates), probabilistic information is used to

decide on encoding. In this case, when the delivery probability between nodes is greater than a threshold, two packets are combined if the next-hop of each packet is the previous-hop of the other coding partner or one of the neighbors of the previous-hop. In this section, we present scenarios where encoding decisions made based on the probabilistic information through the common coding conditions are incorrect at times, and cause a significant number of decoding failures.

5.1.2 Motivating example

Let us consider the grid topology provided in Figure 5.1, and focus on three specific flows: 1) F_1 with packets like P_1 from N_0 to N_7, 2) F_2 with packets like P_2 from N_7 to N_9, and 3) F_3 with packets like P_3 from N_2 to N_0. Let us further assume that N_5 transmits a coded packet from flows F_1 and F_3, $P_1 \oplus P_3$. We assume N_6, as the intended forwarder of P_1 can decode the packet successfully, but N_9 cannot decode it as N_9 cannot overhear P_3. Let us call a packet like P_1, which has been received coded by the node and then it is decoded, a *decoded-native* packet.

The question, now, is under what conditions a node (e.g., N_6) can combine a decoded-native packet (e.g., P_1) with other packets? For example, can N_6 combine packets received from N_5 and N_7? Are the common coding conditions enough to decide on encoding such packets?

Based on the common coding conditions, the combination of P_1 and P_2 at N_6 seems a valid encoding strategy because the next-hop of P_1 (i.e., N_7) is the previous hop of P_2, and the next-hop of P_2 (i.e., N_9) is one of the neighbors of the previous hop of P_1 (i.e., N_5). However, one may notice that if N_9 receives the coded packet $P_1 \oplus P_2$, it cannot

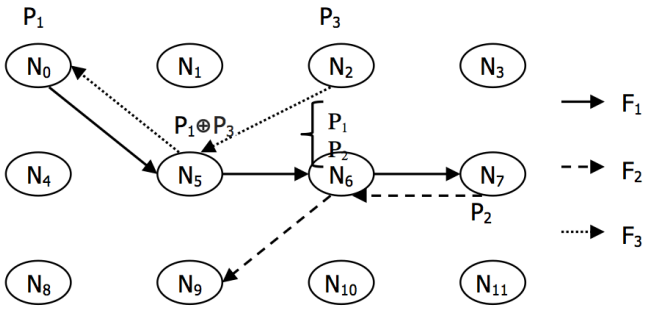


Figure 5.1: Common coding conditions are not sufficient.

decode P_2 correctly as it has only overheard $P_1 \oplus P_3$ but neither P_1 nor P_3. In fact, the problem happens because the previous hop of P_1 (i.e., N_5) sends it as a coded packet; therefore its neighbors (e.g., N_9) do not receive P_1 natively. As a result, if N_6 encodes this decoded-native packet, N_9 cannot decode the received coded packet $P_1 \oplus P_2$.

Note that although COPE uses reception reports, in such a scenario COPE could not rely on them for encoding. Since N_9 does not send any packet, it has to send the reception reports periodically, which reduces the probability that its neighbors receive a fresh report on time. Therefore, most of the time the neighbors do not have deterministic information required for encoding and would need to guess based on the delivery probability between nodes. Hence, if the delivery probability between different nodes is high, in COPE, N_6 will encode P_1 and P_2. To show the severity of the issue, we ran simulations, using a simulation version of COPE in NS-2, to decide on encoding of the packets in the topology depicted in Figure 5.1.

Figure 5.2 presents the number of coded packets received by N_6 (i.e., coded@6), the number of coded packets received by N_9 (i.e., coded@9), and also the number of coded packets that N_9 cannot decode (i.e., failure@9) because of the explained issue. As shown in this figure, by decreasing the inter-arrival time (i.e., increasing the arrival rate), the

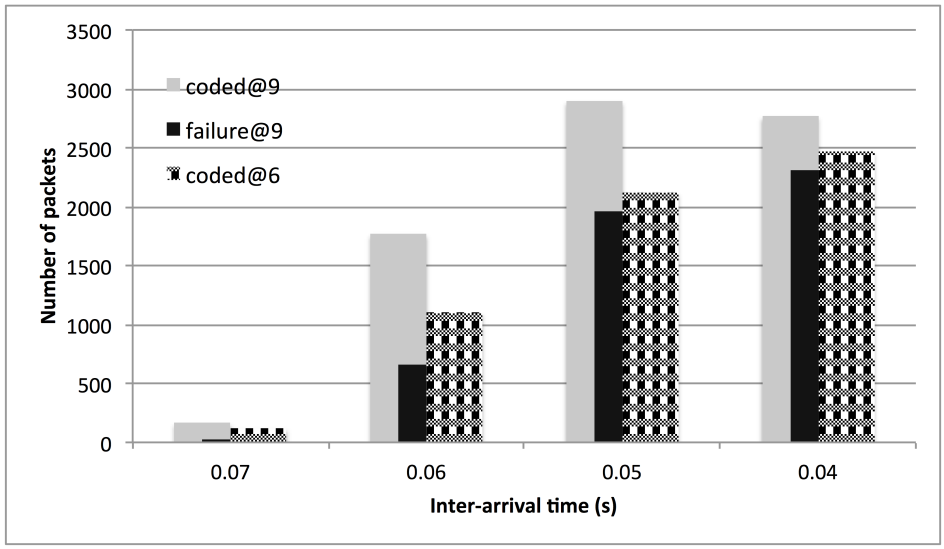


Figure 5.2: Decoding failure when applying the common coding conditions.

length of the transmission queue as well as the coding opportunities at nodes increase. Therefore, the probability that an encoded packet received and decoded by N_6 (i.e., a decoded-native packet) can be encoded again increases, which in this scenario causes the explained issue and consequently increases decoding failures at N_9. As shown here, the fraction of coded packets failed in decoding increases with the packet arrival rate when at the fourth group of Figure 5.2 (i.e., inter-arrival time=0.04 s) this fraction can be as high as 83%.

This example and simulation results show that the common coding conditions are not enough, and more restrictive coding conditions are required to address the issue stated here. Therefore, we address it by proposing an additional rule to restrict the common coding conditions.

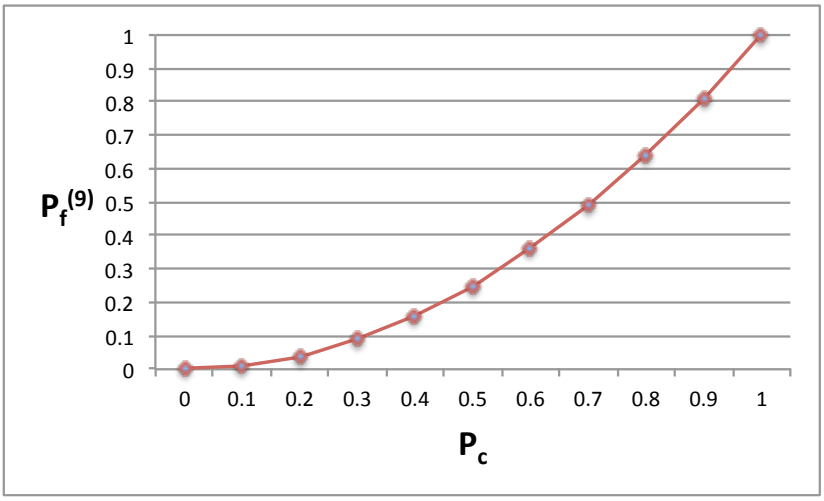


Figure 5.3: Severity of the coding condition problem in different decoding probabilities.

5.1.3 Severity of the problem

Based on what we discussed in Chapter 3, the probability that a coding node sends packets natively or coded can be calculated in terms of the arrival rate of the flows. Therefore, in Figure 5.1 the probability that nodes N_5 and N_6 send coded packets can be calculated in terms of the arrival rates of the corresponding flows at these nodes. Let $P_c^{(i)}$ and $P_n^{(i)}$ denote the probabilities that node N_i sends the packet coded and natively, respectively. Then, the probability that this decoding issue happens at N_9, denoted by $P_f^{(9)}$, equals the probability that both N_5 and N_6 forward a packet coded, which can be calculated as

$$P_f^{(9)} = P_c^{(5)} \times P_c^{(6)}. \tag{5.1}$$

Without loss of generality and for the sake of simpler explanation, we assume the probability of sending coded packets at all coding nodes is the same (i.e., $P_c^{(5)} = P_c^{(6)} = P_c$); hence $P_f^{(9)} = P_c^2$. Therefore, with probability P_c^2 the coding condition problem happens, and N_9 drops the packets due to undecodability. As shown in Figure 5.3, by increasing the arrival rate of flows, which means higher encoding probability and more

coding opportunities, this issue becomes more severe, and a larger portion of coded packets will be dropped at N_9.

5.2 Design Details

5.2.1 An additional rule

As explained earlier to decide on encoding packets, the majority of encoding methods, within a two-hop region, use a similar coding structure called *two-hop coding structure* [101] with the same coding conditions [25, 32, 48, 88, 98, 106]. Based on such *common coding conditions*, node N can combine two packets P_1 and P_2 if:

1. The next-hop of P_1 is the previous hop of P_2 or one of its neighbors, and

2. The next-hop of P_2 is the previous hop of P_1 or one of its neighbors.

However, as illustrated in Section 5.1.2 in some scenarios such as Figure 5.1, these coding conditions are not sufficient. The issue happens because in the common coding conditions, it is assumed that all the neighbors of the previous hop (e.g., N_5) are able to decode the coded packet sent by it (e.g., $P_1 \oplus P_3$). In fact, this is not necessarily a valid assumption as some of these neighbors (e.g., N_9) may not be able to do so. The question is how to establish a complete set of rules to correctly decide on mixing the packets of flows which are decodable at the next-hop? To address this issue, we add an additional condition to the common coding conditions as follows.

RecodingRule - To combine a decoded-native packet (i.e., a packet received as a coded packet from its previous hop and has been decoded) with other packets (i.e., recode the packet), the node does not check the neighborhood of the previous hop of the packet. In

fact, if P_1 is a decoded-native packet the common coding conditions should be modified as follows:

1. The next-hop of P_1 is the previous hop of P_2 or one of its neighbors, and
2. The next-hop of P_2 is the previous hop of P_1.

That is, we remove the neighbor clause from Case 2.

5.2.2 SwitchRule method

RecodingRule is sufficient but may not always be necessary. That is, although it avoids misleading coding opportunities and decoding failures in the scenario depicted in Figure 5.1, in some other scenarios it may limit the number of right coding opportunities in the network. As an example, let us describe the effect of our *RecodingRule* on the scenario presented in Figure 5.4. In this figure, the route of flow F_3, in comparison to Figure 5.1, has changed so that N_9 can overhear the packets of this flow. Now, N_9 overhears P_3 from N_{10}, and $P_1 \oplus P_3$ from N_5. As a result, we do not need to apply *RecodingRule*, and N_9 can decode $P_1 \oplus P_2$ received from N_6 successfully.

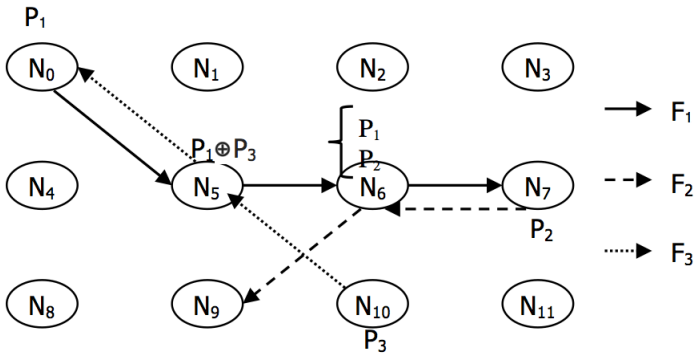


Figure 5.4: RecodingRule, sufficient but not necessary.

Therefore, *RecodingRule* should be intelligently used only in cases that the interaction between flows is so that the common coding conditions may provide misleading coding opportunities. This type of encoded packets cannot be decoded in the next-hop, and the sender will receive a NACK for it, as explained in Section 4.2.1. Thus, we propose a solution called *SwitchRule* to decide properly on applying *RecodingRule* on different flows at different nodes. In fact, *SwitchRule*, based on the received NACKs for each flow at each node, decides to switch on or off *RecodingRule*. Note that *SwitchRule* only needs to be applied at the flow-granularity, not the packet-granularity; thus it only introduces a trivial overhead.

At the beginning, every node uses the common coding conditions to encode packets. However, when each node combines a decoded-native packet, P_1, with another packet, P_2, if the next-hop of P_2 is not the previous hop of P_1 but one of its neighbors, P_1 is tagged as a *suspect* packet. This means we are suspicious that decoding failure may happen because the next-hop of P_1's partner (i.e., P_2) may have not overheard the suspect packet, P_1. Each node keeps track of the number of NACKs received for the partners of suspect packets of each flow. If the number of NACKS for a flow is greater than a threshold, the node switches on the *RecodingRule* for the rest of the packets of that particular flow. This means the node will not combine a decoded-native packet of that flow with any other partner if the next-hop of the partner is not the previous-hop of the decoded-native packet.

Furthermore, a node will switch off the *RecodingRule* whenever it hears packets of a new flow or it does not hear any packet from a flow anymore. To implement the latter case in *SwitchRule*, each node set a timer for each flow. If the timer of a flow times-out before receiving a new packet of that flow, the node switches back to the common coding

```
Initialization:
for each flow F_i
    NACK[F_i]=0
    ExtraRule[F_i]=false
To encode a packet:
if P is decoded-native
    if ExtraRule[F(P)]
        apply ExtraRule
    else
        apply current coding conditions
        if P is combined with P'
            if NH(P') ∈ ng(PH(P))
                tag P as suspect
After receiving an ACK/NACK:
if a NACK is received for P'
    if its partner P was tagged as suspect
        NACK[F(P)]= NACK[F(P)]+1
        if NACK[F(P)] > NACK_th
            ExtraRule[F(P)]=true

if packet P of flow F is sensed
    if the node is a neighbour of NH(P)
        MIAT[F]=0.5×MIAT[F] + 0.5×IAT[F]
        Set flow's timer to α×MIAT[F]
if a flow's timer times-out or a new flow is sensed
    for each flow F_i
        NACK[F_i]=0
        ExtraRule[F_i]=false
```

Figure 5.5: Pseudo-code of *SwitchRule*. The number of NACKs received for flow F is stored in NACK[F]. NH(P), PH(P) and F(P) denote the next-hop, the previous-hop and the flow of P, respectively. The set of neighbors of node N is represented by ng(N). Also, IAT[F] and MIAT[F] denote the inter-arrival time and the mean inter-arrival time of flow F. The timer for flow F is set to α times of MIAT[F], where $\alpha > 1$.

conditions for all flows. The waiting time before the time-out is several times of the estimated inter-arrival time of the packets of the flow. The inter-arrival time of each flow is estimated using a weighted-average over the previous average and the latest measured inter-arrival time, the same idea as the RTT estimation of TCP retransmission timer [95]. Figure 5.5 presents the pseudo-code of the *SwitchRule*'s mechanism.

5.2.3 How RecodingRule improves the performance

To show how much adding *RecodingRule* improves the network performance, we compare the packet delivery rate at N_9, in the topology depicted in Figure 5.1, with and without this additional rule. As explained earlier, because common coding conditions are not accurate enough, some coded packets arrived at N_9 are undecodable. Thus, N_9 sends a NACK, and these packets will be retransmitted again possibly as a new coded packet.

To assure that the only reason that N_9 cannot decode the packets is the described coding condition issue, here we assume that the overhearing link between N_5 and N_9 is perfectly reliable. Applying common coding conditions, a coded packet arrived at N_9 is decodable if it has been received successfully by N_9, and it was not sent coded by N_5. Therefore, the probability that N_9 can successfully receive a packet sent by N_6, under common coding conditions, is calculated as follows

$$P^{(9)}_{\text{common}} = P^{(6)}_{n} \times p_{6,9} + P^{(6)}_{c} \times p_{6,9} \times (1 - P^{(5)}_{c}) \,, \tag{5.2}$$

where $p_{i,j}$ denotes the probability of successful transmission of a packet from N_i to N_j, and $P^{(i)}_{n} = 1 - P^{(i)}_{c}$.

Now, if we add our *RecodingRule*, after receiving a few NACKs, N_6 stops recoding the packets of the suspect flow (i.e., flow1). Therefore, the packets which were sent natively

by $N5$ can be coded, and the rest are sent natively. In this case, ignoring those few suspect packets which were sent coded by N_6 wrongly at the beginning (and the NACKs received for them helped us to diagnose the problem), the probability of successful packet transmission to N_9 is

$$P^{(9)}_{\text{recoding}} = P^{(6)}_n \times p_{6,9} + (P^{(6)}_c \times P^{(5)}_c) \times p_{6,9} + (P^{(6)}_c \times (1 - P^{(5)}_c)) \times p_{6,9}\,. \tag{5.3}$$

By comparing (5.2) and (5.3), one may notice that *RecodingRule* can increase the probability of successful transmission of packets to N_9 by $(P^{(6)}_c \times P^{(5)}_c) \times p_{6,9}$ through intelligently avoiding destructive encoding of the packets that are undecodable and will be dropped at the destination, and instead sending them as native packets.

5.3 Performance Evaluation

5.3.1 Settings

We use the Network Simulator (NS-2) to compare the performance of FlexONC with *RecodingRule*, against the non-coding scheme, a simulation version of COPE, BEND, CORE, and the original FlexONC as proposed in Chapter 4.

The same as Chapter 4, the channel propagation used in NS-2 is a two-ray ground reflection model [80], and the maximum transmission range is 250 m. The data rate is fixed to 1 Mbps. The sources, in our simulation scenarios, send CBR (constant bit rate) data flows with a datagram size of 1000 bytes. Also, we use DSDV (Destination-Sequenced Distance-Vector) [77] as the routing protocol.

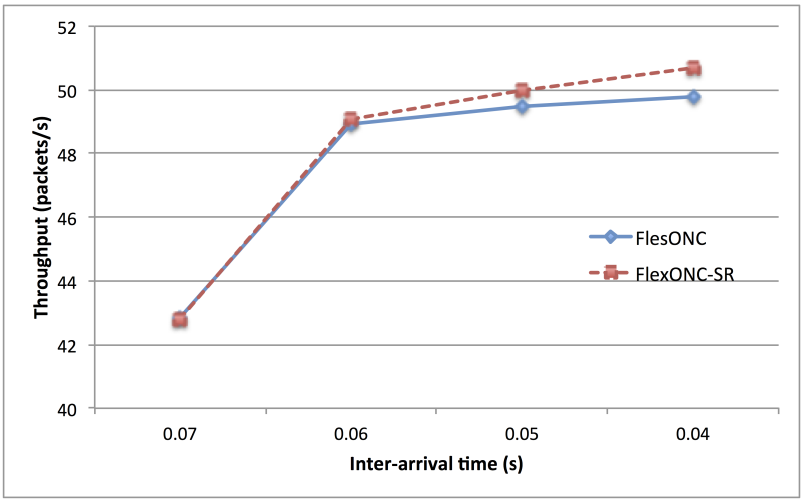


Figure 5.6: Effect of SwitchRule on the throughput of FlexONC in the topology depicted in Figure 5.1.

5.3.2 Performance under SwitchRule

We investigate the effect of *SwitchRule* on the performance of FlexONC in two different scenarios, where at some nodes the common coding conditions may be insufficient to combine the right packets. First, we compare the throughput of FlexONC in the topology with sources and destinations selected as in Figure 5.1 for different inter-arrival times considering both cases that the *SwitchRule* functionality is off (i.e., only common coding conditions are used) and is on. We call the latter version of FlexONC, which uses *SwitchRule*, *FlexONC-SR*. In this scenario, 3 flows transmit their packets for 150 s, BER equals 2×10^{-6}, and in FlexONC-SR, the NACK threshold to start applying *RecodingRule* is equal to 5.

As shown in Figure 5.6, although at lower packet arrival rates (i.e., longer inter-arrival time) the performance of FlexONC and FlexONC-SR is close, at higher arrival rates FlexONC-SR can benefit from *SwitchRule* to avoid decoding failures and more re-

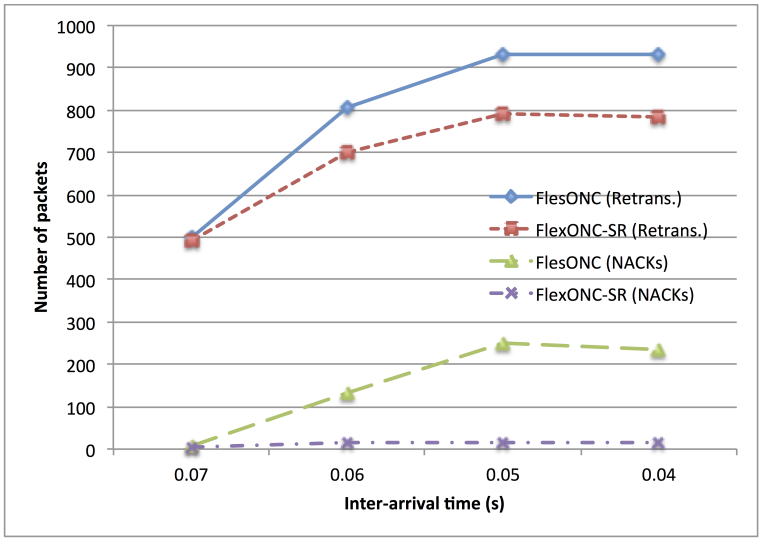


Figure 5.7: Number of retransmissions and received NACKs with and without applying SwitchRule in FlexONC.

trasnmissions to deliver packets to the destination. As an evidence, Figure 5.7 presents the number of retransmitted packets and the number of received NACKs in both FlexONC and FlexONC-SR. As explained in Subsection 5.1.2, the common coding conditions may erroneously decide to combine the decoded-native packets with other packets, and obviously at higher arrival rates, more decoded-native packets are generated (i.e., the probability that the same packet is encoded at different nodes increases).

We also compare the performance of FlexONC-SR with other baselines in a more general 5×5 mesh network with 8 different CBR flows (Figure 5.8), with duration of 150 s. As shown in Figure 5.9, although BER is very small ($BER = 2 \times 10^{-6}$), FlexONC outperforms other schemes. Moreover, when the functionality of *SwitchRule* is added to FlexONC (i.e., FlexONC-SR), its throughput is even further boosted.

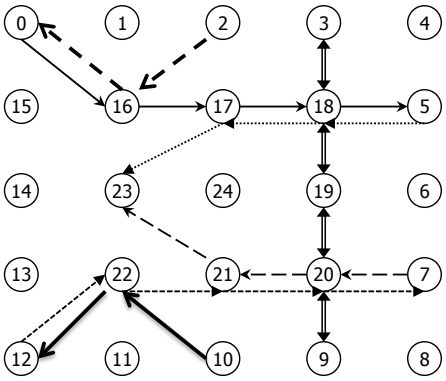


Figure 5.8: 5×5 mesh network used to investigate the performance of SwitchRule.

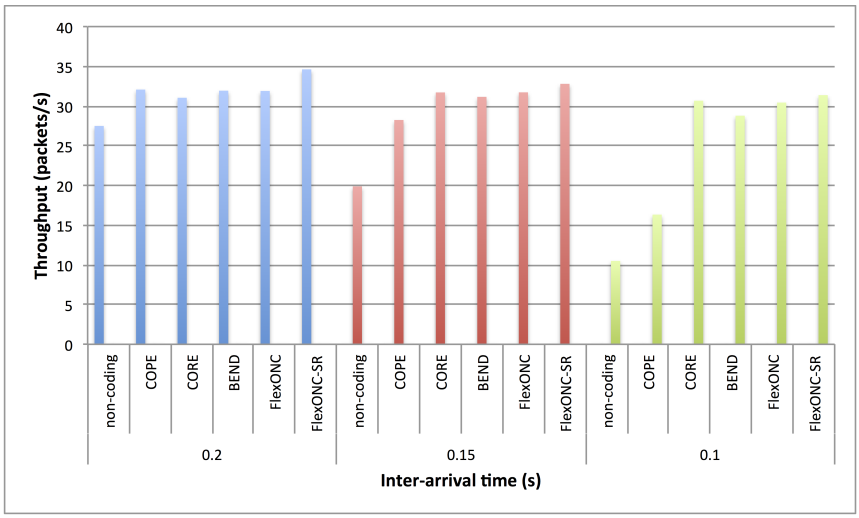


Figure 5.9: Throughput of different methods in the 5×5 grid topology.

5.4 Summary

As illustrated in this chapter, the common coding conditions used in other IXNC methods are not accurate enough to recognize right coding opportunities in some scenarios, and may lead to decoding failures. We discovered this issue, and addressed it by adding an additional rule to the current conditions used to encode the packets in different methods. FlexONC was enhanced with these more accurate coding conditions, and *SwitchRule* was

utilized to apply these coding conditions appropriately and limit decoding failures. The simulation results show that by applying *SwitchRule*, FlexONC is able to adapt coding conditions in different scenarios, and uses a more complete set of rules for encoding when *common coding conditions* are not sufficient.

Chapter 6

Conclusions and Future Work

In this final chapter, we summarize the contributions presented in this dissertation and discuss several potential extensions to our work.

6.1 Conclusions

The performance of wireless mesh networks, aiming to realize the dream of a seamlessly connected world, is restricted with several challenges. However, since the last decade "Network Coding" is proved to improve the performance of wireless networks significantly by proactively utilizing the broadcast nature of the wireless medium. To better quantify the benefits of network coding over traditional forwarding, we studied the throughput and end-to-end delay of IXNC for actual protocols considering PHY/MAC layer specifications. In addition, to further boost the performance of IXNC especially in lossy environments, we integrated it with opportunistic routing, another promising technique in wireless networks, and also proposed a set of coding conditions, which avoids incorrect encoding decisions. The following conclusions can be drawn from this dissertation:

- We utilized queuing theory to derive the throughput and an upper bound of average end-to-end delay of both traditional forwarding (i.e., non-coding scheme) and IXNC in multi-hop wireless mesh networks, where two unicast sessions in opposite directions traverse the network. The modeling techniques proposed here provide a comprehensive framework to study the performance of similar cases, and also can be considered as an important building block toward modeling more complicated scenarios leading to the communication networks with better performance.

- We proposed an analytical framework considering the specifications of the IEEE 802.11 DCF with CSMA/CA random access, such as the binary exponential back-off time with clock freezing and virtual carrier sensing, to formulate the links quality, waiting time of the packets and retransmissions.

- We used a multi-class queuing network with stable queues, where coded packets have a non-preemptive higher priority over native packets, and forwarding of native packets is not delayed if no coding opportunities are available (i.e., opportunistic coding). As described in Section 3.2.2.1, applying opportunistic coding makes it significantly more challenging to estimate coding opportunities at nodes.

- We used computer simulations to verify the accuracy of our analytical model in different scenarios, and the consistency of the analytical and simulation results corroborates the validity of the model. Also, the results showed that when retransmission is enabled, both throughput and end-to-end delay of the network increase. In addition, while without retransmission both throughput and end-to-end delay are decreasing functions of the BER, with retransmission, throughput stays constant across different BERs and end-to-end delay is an increasing function of BER.

- We compared the maximum stable throughput of traditional forwarding and network coding, and the results showed that when PHY/MAC layer constraints are taken into account, the benefits of network coding are more notable in smaller topologies, and it becomes comparable to traditional forwarding as the number of intermediate nodes increases. However, this finding does not contradict the fact that network coding can offer a competitive edge in wireless mesh networks since these networks are meant as an extended access technology, and it is unlikely to have very long paths.

- We presented FlexONC, a joint IXNC and OR approach that provides more flexibility and coding opportunities in the network. It spreads different flows better than former related studies like BEND and enables a higher level of cooperation between intended and non-intended forwarders at the link layer of multi-hop wireless networks.

- We ran simulations in NS-2 to compare the performance of FlexONC against the non-coding scheme, a simulation version of COPE as a prominent research on network coding, and two OR schemes in network coding (i.e., BEND and CORE) from different aspects such as throughput, end-to-end delay, the number of duplicate packets, the number of coding opportunities, and overall overhead and complexity. The results show that FlexONC benefits from OR and provides a higher throughput than other baselines especially at higher BERs.

- We discovered that the conditions used in previous IXNC studies to combine packets of different flows are overly optimistic and would adversely affect the network performance.

- We added an additional rule to the current conditions used to encode the packets in different IXNC methods, and provided a more accurate set of rules for packet encoding when *common coding conditions* are not sufficient. We also proposed a mechanism to apply these coding conditions appropriately and limit decoding failures. The simulation results showed that by augmenting the description of FlexONC to incorporate our solution, FlexONC-SR is able to make more intelligent and comprehensive encoding decisions to avoid transmitting undecodable packets in the network.

- Although the experiments on FlexONC are conducted in grid topologies, the benefit of having more *diffusion gain* as well as an additional rule in the coding conditions and having a mechanism to turn it on/off dynamically is still present in general scenarios with random node distribution and flow assignments. Hence, we expect the relative performance among FlexONC and other baselines to be similar to what we have shown here.

6.2 Future Work

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

- *Analytical model of a general topology* – although our analytical model was formulated in a chain topology, it is applicable to any topology as long as the two opposite flows follow the same path. A future extension of our work could be developing an analytical framework for a general topology, where more than two flows are traveling and possibly mixing together.

- *Analytical model of joint IXNC and OR* – as shown in this dissertation and other related works, adding OR to IXNC can improve network performance significantly. To the best of our knowledge, there is no comprehensive analytical model studying the performance under this joint approach. Hence, incorporating OR to our model can be an interesting area of future investigation, which helps better understand the pros and cons of this approach and provide more reliable and efficient communication networks.

- *Adding IANC to FlexONC* – in recent years, a number of publications have been presented that apply both inter- and intra-flow network coding, but in some limited scenarios [28, 49, 50, 56, 61, 75, 79, 89, 96, 97, 110]. However, most of these studies use traditional forwarding with a fixed route and cannot benefit from OR. Also, they can only explore and capture the coding opportunities along fixed routes. Furthermore, most of them are not able to efficiently merge these two great techniques; instead they just use them in different layers. We believe that this combination, if realized carefully, could introduce further improvement in the performance, and represents another way to extend FlexONC.

- *Exploring the coding conditions problems* – as explained in Chapter 5, we detected the problem with common coding conditions and addressed it by proposing *RecodingRule* and *SwitchRule*. It would be worth conducting more research to explore the similar issues related to the coding conditions in IXNC, and propose a scheme that provides nodes with more timely deterministic information and also more accurate probabilistic decisions on encoding.

- *Coding beyond a two-hop region* – there has been some research on extending

the coding region in the network by proposing a new coding-aware routing protocol [23, 24, 58]. However, these studies have limitations. First their approach cannot work with well-known and popular routing protocols. In fact, they can exploit coding opportunities in entire network only if they use their coding-aware routing protocol. Second since they use coding-aware metrics, a large amount of communication overhead is imposed to the network. Thus, that would be of interest to extend FlexONC by including a combination of OR and more powerful detection of coding opportunities beyond a two-hop region.

- *Working properly with TCP* – in general, network coding supports UDP flows well but not so much for TCP because it may achieve a gain much lower than expected due to the congestion control mechanism in TCP windows. However, in recent years, a few studies have been conducted to control sent and received packets and ACKs to the transport layer, so that network coding can be applied without much effect on TCP windows [6, 15, 29, 91, 92, 105]. Hence, a future extension of FlexONC could be its exploration and modification for TCP flows.

- *Physical-layer network coding* – Physical-layer network coding (PNC) [30,63,78,107] is another type of network coding, in which nodes simultaneously transmit packets to a relay node that exploits mixed wireless signals to extract a coded packet. In recent years, a number of analytical studies have investigated the throughput capacity of PNC in multi-hop networks [64–66], and we believe the model proposed here to study the throughput and delay of chain topologies can be extended into PNC, where some two-hop nodes can transmit simultaneously to a relay node without causing collision but these concurrent transmissions increase the carrier sensing

range of the network [65].

Appendices

A Estimating h_x in (3.3)

Given that the arrival rate of packets is Poisson, the probability that node N_x does not transmit a packet during the time window of 2δ, can be calculated as follows: $P(t \geq 2\delta) = 1 - P(t < 2\delta) = 1 - (1 - e^{-2\delta\lambda_x}) = e^{-2\delta\lambda_x}$.

Since $e^{-a} = 1 - a + \frac{a^2}{2!} - \frac{a^3}{3!} + ... \overset{a<1}{\approx} 1 - a$, this probability for N_j can be estimated as $P(t \geq 2\delta) \approx 1 - 2\delta\lambda_x$. Therefore, $h_x = 2\delta\lambda_x$.

B Back-off Time Considering "Clock Freezing" Behavior

Based on (3.9), if during back-off time (i.e., T_{counter}), one of the neighbors of the node sends a packet, the waiting time will be extended to $T_{\text{counter}} + T_{\text{trans}}$. In the same way if i packets are transmitted by the neighbors, this waiting time (until the counter reaches zero) equals $T_{\text{counter}} + i \times T_{\text{trans}}$. To explain why this equation provides an upper-bound for back-off time estimation, let us consider the scenario depicted in Figure 3.4. In this example, since a part of transmission by N_k overlaps the transmission by N_j (i.e., assuming

N_j and N_k are not in interference range of each other), the waiting time at N_i is less than $T_{\text{counter}} + 2T_{\text{trans}}$. However, the equation does not take into account the overlapped transmissions, and assumes the transmission by the next node begins exactly after the end of the transmission by the last node.

Furthermore for a given i, $(1 - e^{-\lambda T_{\text{trans}}})^i$ represents the probability of having one or more packet transmissions during iT_{trans}, which is greater than the real probability that needs to be calculated; the probability of having exactly one transmission during each $[T_{\text{counter}} + (j-1)T_{\text{trans}}, T_{\text{counter}} + jT_{\text{trans}}], (j = 1, ..., i)$. Therefore, this equation provides an upper-bound of the expected waiting time due to the "clock freezing" behavior of back-off timer.

Given that for $0 < q < 1$, $\sum_{i=1}^{\infty} q^n = \frac{q}{1-q}$, and $\sum_{i=1}^{\infty} nq^n = \frac{q}{(1-q)^2}$, and since $0 < (1 - e^{-\lambda T_{\text{trans}}}) < 1$, the closed form of (3.9) can be calculated as:

$$
\begin{aligned}
T(m) &= T_{\text{counter}}(m) \times e^{-\lambda T_{\text{counter}}(m)} + \sum_{i=1}^{\infty} (T_{\text{counter}}(m) + i \times T_{\text{trans}}) \times (1 - e^{-\lambda T_{\text{trans}}})^i \\
&= T_{\text{counter}}(m) \times e^{-\lambda T_{\text{counter}}(m)} + T_{\text{counter}}(m) \sum_{i=1}^{\infty} (1 - e^{-\lambda T_{\text{trans}}})^i + T_{\text{trans}} \sum_{i=1}^{\infty} i(1 - e^{-\lambda T_{\text{trans}}})^i \\
&= T_{\text{counter}}(m) \times e^{-\lambda T_{\text{counter}}(m)} + (1 - e^{-\lambda T_{\text{trans}}}) \left(\frac{T_{\text{counter}}(m)}{e^{-\lambda T_{\text{trans}}}} + \frac{T_{\text{trans}}}{e^{-2\lambda T_{\text{trans}}}} \right) .
\end{aligned}
$$

C Closed form of $P_{\mathbf{mtc}}(r)$

Based on (3.17),

$$P_{\text{mtc}}(r) = \sum_{k=0}^{\infty} \left(\frac{\lambda_i^{n(r)}}{\mu_i^{n,\text{seen}}}\right)^k \left(1 - \frac{\lambda_i^{n(r)}}{\mu_i^{n,\text{seen}}}\right) \left(1 - \sum_{j=0}^{k} \frac{e^{\left(-\lambda_i^{n(\bar{r})} W(Q^n)\right)} \left(\lambda_i^{n(\bar{r})} W(Q^n)\right)^j}{j!}\right)$$

$$= \left(1 - \frac{\lambda_i^{n(r)}}{\mu_i^{n,\text{seen}}}\right) \sum_{k=0}^{\infty} \left[\left(\frac{\lambda_i^{n(r)}}{\mu_i^{n,\text{seen}}}\right)^k - \left(\frac{\lambda_i^{n(r)}}{\mu_i^{n,\text{seen}}}\right)^k \sum_{j=0}^{k} \frac{e^{\left(-\lambda_i^{n(\bar{r})} W(Q^n)\right)} \left(\lambda_i^{n(\bar{r})} W(Q^n)\right)^j}{j!}\right]$$

$$= \left(1 - \frac{\lambda_i^{n(r)}}{\mu_i^{n,\text{seen}}}\right) \left[\left(\sum_{k=0}^{\infty} \left(\frac{\lambda_i^{n(r)}}{\mu_i^{n,\text{seen}}}\right)^k\right)\right.$$

$$\left. - \left(\sum_{k=0}^{\infty} \left(\frac{\lambda_i^{n(r)}}{\mu_i^{n,\text{seen}}}\right)^k \sum_{j=0}^{k} \frac{e^{\left(-\lambda_i^{n(\bar{r})} W(Q^n)\right)} \left(\lambda_i^{n(\bar{r})} W(Q^n)\right)^j}{j!}\right)\right].$$

Since $\sum_{k=0}^{\infty} q^k = \frac{1}{1-q}$, where $|q| < 1$,

$$P_{\text{mtc}}(r) = \left(1 - \frac{\lambda_i^{n(r)}}{\mu_i^{n,\text{seen}}}\right) \left[\frac{1}{1 - \frac{\lambda_i^{n(r)}}{\mu_i^{n,\text{seen}}}} - e^{\left(-\lambda_i^{n(\bar{r})} W(Q^n)\right)} \sum_{k=0}^{\infty} \left(\frac{\lambda_i^{n(r)}}{\mu_i^{n,\text{seen}}}\right)^k \sum_{j=0}^{k} \frac{\left(\lambda_i^{n(\bar{r})} W(Q^n)\right)^j}{j!}\right].$$

Now, we need to find the closed form of $\sum_{k=0}^{\infty} q^k \sum_{j=0}^{k} \frac{a^j}{j!}$, where $|q| < 1$. Based on Fubini's theorem [9],

$\sum_{k\geq 0} \sum_{j=0}^{k} f(k,j) = \sum_{j\geq 0} \sum_{k=j}^{\infty} f(k,j)$.

Therefore,

$\sum_{k=0}^{\infty} q^k \sum_{j=0}^{k} \frac{a^j}{j!} = \sum_{j=0}^{\infty} \frac{a^j}{j!} \sum_{k=j}^{\infty} q^k = \frac{1}{1-q} \sum_{j=0}^{\infty} \frac{(aq)^j}{j!} = \frac{e^{aq}}{1-q}$,

as, $\sum_{j=0}^{\infty} \frac{a^j}{j!} = e^a$, $\sum_{k=j}^{\infty} q^k = \frac{q^j}{1-q}$.

Finally, $P_{\text{mtc}}(r)$ for flow r can be calculated as presented in (3.18).

References

[1] IEEE 802.11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications. *IEEE Std 802.11-2007 (Revision of Std. 802.11-1999)*, 2007.

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

[3] 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.

[4] M. Amerimehr, F. Ashtiani, and S. Valaee. Maximum Stable Throughput of Network-Coded Multiple Broadcast Sessions for Wireless Tandem Random Access Networks. *IEEE Transactions on Mobile Computing*, 13(6):1256–1267, June 2014.

[5] M. H. Amerimehr, F. Ashtiani, and M. B. Iraji. An Analytical Approach for Throughput Evaluation of Wireless Network Coding. In *Proceedings of the IEEE International Conference on Communications (ICC)*, pages 1696–1700, June 2009.

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

ings of IEEE Global Communications Conference (GLOBECOM), pages 125–130, Dec. 2012.

[7] J. Bicket, D. Aguayo, S. Biswas, and R. Morris. Architecture and Evaluation of an Unplanned 802.11B Mesh Network. In *Proceedings of the 11th Annual International Conference on Mobile Computing and Networking (MobiCom)*, pages 31–42, Sept. 2005.

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

[9] J. M. Borwein, D. H. Bailey, and R. Girgensohn. *Experimentation in Mathematics: Computational Paths to Discovery*. Wellesley, MA: A K Peters, 2004.

[10] A. Boukerche and A. Darehshoorzadeh. Opportunistic Routing in Wireless Networks: Models, Algorithms, and Classifications. *ACM Computing Surveys*, 47(2):22:1–22:36, Nov. 2014.

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

[12] P. J. Burke. The Output of a Queueing System. *Operations Research*, 4:699–714, 1956.

[13] S. Chachulski, M. Jennings, S. Katti, and D. Katabi. Trading Structure for Randomness in Wireless Opportunistic Routing. *SIGCOMM Computer Communication Review*, 37(4):169–180, Aug. 2007.

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

[15] 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.

[16] K. Chung, Y. C. Chou, and W. Liao. CAOR: Coding-aware Opportunistic Routing in Wireless Ad Hoc Networks. In *Proceedings of IEEE International Conference on Communications (ICC)*, pages 136–140, June 2012.

[17] D. S. J. De Couto, D. Aguayo, J. Bicket, and R. Morris. A High-throughput Path Metric for Multi-hop Wireless Routing. In *Proceedings of the 9th Annual International Conference on Mobile Computing and Networking (MobiCom)*, pages 134–146, Sept. 2003.

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

[19] C. Fragouli, J. Widmer, and J.-Y. Le Boudec. A Network Coding Approach to Energy Efficient Broadcasting: From Theory to Practice. In *Proceedings of IEEE INFOCOM*, pages 1–11, Apr. 2006.

[20] A. E. Gamal, J. Mammen, B. Prabhakar, and D. Shah. Optimal Throughput-Delay Scaling in Wireless Networks - Part I: the Fluid Model. *IEEE Transactions on Information Theory*, 52(6):2568–2592, June 2006.

[21] C. Gkantsidis and P. Rodriguez. Network Coding for Large Scale Content Distribution. In *Proceedings of IEEE INFOCOM*, volume 4, pages 2235–2245, Mar. 2005.

[22] D. Gross and C. M. Harris. *Fundamentals of Queueing Theory*. John Wiley & Sons, Inc., third edition, 1998.

[23] B. Guo, H. Li, C. Zhou, and Y. Cheng. General Network Coding Conditions in Multi-Hop Wireless Networks. In *Proceedings of IEEE International Conference on Communications (ICC)*, pages 1–5, May 2010.

[24] 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.

[25] H. Guo, X. Liu, Z. Shi, and X. Bai. Image Relevance Feedback Retrieval Based on Selective Cluster Ensembles. In *Proceedings of the Chinese Conference on Pattern Recognition (CCPR'10)*, pages 1–5, Oct. 2010.

[26] 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, Dec. 2014.

[27] M. K. Han, A. Bhartia, L. Qiu, and E. Rozner. O3: Optimized Overlay-based Opportunistic Routing. In *Proceedings of the 12th ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc)*, pages 1–11, May 2011.

[28] J. Hansen, J. Krigslund, D. Lucani, and F. Fitzek. Bridging Inter-flow and Intra-flow Network Coding for Video Applications: Testbed Description and Performance Evaluation. In *Proceedings of the 18th International Workshop on Computer Aided Modeling and Design of Communication Links and Networks (CAMAD)*, pages 7–12, Sept. 2013.

[29] S. Hassayoun, P. Maille, and D. Ros. On the Impact of Random Losses on TCP Performance in Coded Wireless Mesh Networks. In *Proceedings of IEEE INFOCOM*, pages 1–9, Mar. 2010.

[30] P. Hu and M. Ibnkahla. A Survey of Physical-Layer Network Coding in Wireless Networks. In *Proceeding of 25th Biennial Symposium on Communications*, pages 311–314, May 2010.

[31] 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 (GLOBECOM)*, pages 4578–4583, Dec. 2013.

[32] L. Huang and C. W. Sung. An Iterative Routing Algorithm for Energy Minimization in Coded Wireless Networks. In *Proceedings of the 22nd IEEE International Symposium on Personal Indoor and Mobile Radio Communications (PIMRC)*, pages 1124–1128, Sept. 2011.

[33] J. Hwang and S.-L. Kim. Cross-Layer Optimization and Network Coding in CSMA/CA-Based Wireless Multihop Networks. *IEEE/ACM Transactions on Networking*, 19(4):1028–1042, Aug. 2011.

[34] 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, Nov. 2011.

[35] M. Iraji, M. Amerimehr, and F. Ashtiani. A Queueing Model for Wireless Tandem Network Coding. In *Proceedings of IEEE Wireless Communications and Networking Conference (WCNC)*, pages 1–6, Apr. 2009.

[36] 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.

[37] J. R. Jackson. Networks of Waiting Lines. *Operations Research*, 5:518–521, 1957.

[38] J. R. Jackson. Jobshop-Like Queueing Systems. *Management Science*, 10:131–142, 1963.

[39] V. Jamali, N. Zlatanov, and R. Schober. Bidirectional Buffer-Aided Relay Networks With Fixed Rate Transmission-Part I: Delay-Unconstrained Case. *IEEE Transactions on Wireless Communications*, 14(3):1323–1338, Mar. 2015.

[40] 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, Mar. 2015.

[41] 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, Oct. 2008.

[42] D. Johnson, Y. Hu, and D. Maltz. The Dynamic Source Routing Protocol. RFC 4728, Network Working Group, Feb. 2007.

[43] 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 (MobiCom)*, pages 123–126, Oct. 2013.

[44] S. Kafaie, M. H. Ahmed, Y. Chen, and O. A. Dobre. Throughput Analysis of Network Coding in Multi-Hop Wireless Mesh Networks Using Queueing Theory. In *Proceedings of IEEE Global Communications Conference (GLOBECOM)*, pages 1–6, Dec. 2016.

[45] S. Kafaie, Y. Chen, O. Dobre, and M. Ahmed. Network Coding Implementation Details: A Guidance Document. In *22th Annual Newfoundland Electrical and Computer Engineering Conference (NECEC)*, St. John's, Canada, Nov. 2013.

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

[47] S. Katti, D. Katabi, W. Hu, H. Rahul, and M. Medard. The Importance of Being Opportunistic: Practical Network Coding for Wireless Environments. In *Proceedings of Allerton Annual Conference on Communication, Control and Computing*, Sept. 2005.

[48] S. Katti, H. Rahul, W. Hu, D. Katabi, M. Medard, and J. Crowcroft. XORs in the Air: Practical Wireless Network Coding. *IEEE/ACM Transactions on Networking*, 16(3):497–510, June 2008.

[49] A. Khreishah, I. Khalil, and J. Wu. Polynomial Time and Provably Efficient Network Coding Scheme for Lossy Wireless Networks. In *Proceedings of IEEE 8th International Conference on Mobile Adhoc and Sensor Systems (MASS)*, pages 391–400, Oct. 2011.

[50] A. Khreishah, I. Khalil, and J. Wu. Low Complexity and Provably Efficient Algorithm for Joint Inter and Intrasession Network Coding in Wireless Networks. *IEEE Transactions on Parallel and Distributed Systems*, 24(10):2015–2024, Oct. 2013.

[51] L. Kleinrock. *Queueing Systems*, volume 1: Theory. John Wiley & Sons, Inc., 1975.

[52] L. Kleinrock. *Queueing Systems*, volume 2: Computer Applications. John Wiley & Sons, Inc., 1976.

[53] S. W. Ko and S. L. Kim. Delay-Constrained Capacity of the IEEE 802.11 DCF in Wireless Multihop Networks. *IEEE Transactions on Mobile Computing*, 15(5):1105–1115, May 2016.

[54] 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 (MobiCom)*, pages 1–3, Sept. 2008.

[55] D. Koutsonikolas, C.-C. Wang, and Y. Hu. CCACK: Efficient Network Coding Based Opportunistic Routing Through Cumulative Coded Acknowledgments. In *Proceedings of IEEE INFOCOM*, pages 1–9, Mar. 2010.

[56] J. Krigslund, J. Hansen, M. Hundeboll, D. Lucani, and F. Fitzek. CORE: COPE with MORE in Wireless Meshed Networks. In *Proceedings of the IEEE Vehicular Technology Conference (VTC Spring)*, pages 1–6, June 2013.

[57] R. Laufer, H. Dubois-Ferriere, and L. Kleinrock. Polynomial-Time Algorithms for Multirate Anypath Routing in Wireless Multihop Networks. *IEEE/ACM Transactions on Networking*, 20(3):742–755, June 2012.

[58] J. Le, J. Lui, and D. M. Chiu. DCAR: Distributed Coding-Aware Routing in Wireless Networks. In *Proceedings of the 28th International Conference on Distributed Computing Systems (ICDCS)*, pages 462–469, June 2008.

[59] J. Le, J. Lui, and D.-M. Chiu. On the Performance Bounds of Practical Wireless Network Coding. *IEEE Transactions on Mobile Computing*, 9(8):1134–1146, Aug. 2010.

[60] J. Le, J. C. S. Lui, and D. M. Chiu. DCAR: Distributed Coding-Aware Routing in Wireless Networks. *IEEE Transactions on Mobile Computing*, 9(4):596–608, Apr. 2010.

[61] P. Li and S. Guo. On the Multicast Capacity in Energy-Constrained Lossy Wireless Networks by Exploiting Intrabatch and Interbatch Network Coding. *IEEE Transactions on Parallel and Distributed Systems*, 24(11):2251–2260, Nov. 2013.

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

[63] S. C. Liew, S. Zhang, and L. Lu. Physical-layer Network Coding: Tutorial, Survey, and Beyond. *Physical Communication*, 6:4–42, 2013. Network Coding and its Applications to Wireless Communications.

[64] S. Lin and L. Fu. Unsaturated Throughput Analysis of Physical-Layer Network Coding Based on IEEE 802.11 Distributed Coordination Function. *IEEE Transactions on Wireless Communications*, 12(11):5544–5556, Nov. 2013.

[65] S. Lin and L. Fu. Throughput Capacity of IEEE 802.11 Many-to/From-One Bidirectional Networks With Physical-Layer Network Coding. *IEEE Transactions on Wireless Communications*, 15(1):217–231, Jan. 2016.

[66] 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, Jan. 2016.

[67] 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 (ICNP)*, pages 13–22, Oct. 2008.

[68] Y. Lin, B. Li, and B. Liang. Efficient Network Coded Data Transmissions in Disruption Tolerant Networks. In *Proceedings of IEEE INFOCOM*, pages 2180–2188, Apr. 2008.

[69] H. Liu, H. Yang, Y. Wang, B. Wang, and Y. Gu. CAR: Coding-Aware Opportunistic Routing for Unicast Traffic in Wireless Mesh Networks. *Journal of Network and Systems Management*, 23(4):1104–1124, Oct. 2015.

[70] 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, Dec. 2009.

[71] MIT Roofnet. http://pdos.csail.mit.edu/roofnet/doku.php.

[72] N. Moghadam and H. Li. A New Wireless Multicast Queuing Design Using Network Coding and Data-Flow Model. *IEEE Communications Letters*, 20(8):1603–1606, Aug. 2016.

[73] N. Moghadam and H. Li. Queue Stability Analysis in Network Coded Wireless Multicast Network. *IEEE Communications Letters*, 20(5):950–953, May 2016.

[74] P. Ostovari, J. Wu, and A. Khreishah. Network Coding Techniques for Wireless and Sensor Networks. In *The Art of Wireless Sensor Networks*, Signals and Communication Technology, pages 129–162. Dec. 2014.

[75] K. Ou, Y. Xu, and S. Fu. Rate Selection for Wireless Networks with Intra- and Inter-Session Network Coding. In *Proceedings of IEEE Global Communications Conference (GLOBECOM)*, pages 4707–4712, Dec. 2012.

[76] G. Paschos, C. Fragiadakis, L. Georgiadis, and L. Tassiulas. Wireless Network Coding with Partial Overhearing Information. In *Proceedings of IEEE INFOCOM*, pages 2337–2345, Apr. 2013.

[77] 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, Oct. 1994.

[78] P. Popovski and H. Yomo. The Anti-Packets Can Increase the Achievable Throughput of a Wireless Multi-Hop Network. In *Proceedings of IEEE International Conference on Communications (ICC)*, volume 9, pages 3885–3890, June 2006.

[79] C. Qin, Y. Xian, C. Gray, N. Santhapuri, and S. Nelakuditi. I^{2}MIX: Integration of Intra-Flow and Inter-Flow Wireless Network Coding. In *Proceedings of 5th IEEE Annual Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks Workshops (SECON)*, pages 1–6, June 2008.

[80] T. S. Rappaport. *Wireless Communications: Principles and Practice*. Prentice Hall, second edition, Dec. 2001.

[81] S. Rayanchu, S. Sen, J. Wu, S. Banerjee, and S. Sengupta. Loss-aware Network Coding for Unicast Wireless Sessions: Design, Implementation, and Performance Evaluation. In *Proceedings of SIGMETRICS*, volume 36, pages 85–96, June 2008.

[82] C. Reis, R. Mahajan, M. Rodrig, D. Wetherall, and J. Zahorjan. Measurement-based Models of Delivery and Interference in Static Wireless Networks. *SIGCOMM Computer Communication Review*, 36(4):51–62, Aug. 2006.

[83] E. Rozner, J. Seshadri, Y. Mebta, and L. Qiu. Simple Opportunistic Routing Protocol for Wireless Mesh Networks. In *Proceedings of IEEE Workshop on Wireless Mesh Networks*, pages 48–54, Sept. 2006.

[84] T. L. Saaty. Axiomatic Foundation of the Analytic Hierarchy Process. *Management Science*, 32(7):841–855, July 1986.

[85] Y. Sagduyu and A. Ephremides. On Broadcast Stability Region in Random Access through Network Coding. In *Proceedings of the 44th Annual Allerton Conference on Communication, Control, and Computing*, pages 143–150, Sept. 2006.

[86] Y. Sagduyu and A. Ephremides. On Network Coding for Stable Multicast Communication. In *Proceedings of IEEE Military Communications Conference (MILCOM)*, pages 1–7, Oct. 2007.

[87] Y. Sagduyu and A. Ephremides. Cross-Layer Optimization of MAC and Network Coding in Wireless Queueing Tandem Networks. *IEEE Transactions on Information Theory*, 54(2):554–571, Feb. 2008.

[88] H. Seferoglu and A. Markopoulou. Network Coding-Aware Queue Management for Unicast Flows over Coded Wireless Networks. In *Proceedings of the IEEE International Symposium on Network Coding (NetCod)*, pages 1–6, June 2010.

[89] H. Seferoglu, A. Markopoulou, and K. Ramakrishnan. I^2NC: Intra- and Inter-session Network Coding for Unicast Flows in Wireless Networks. In *Proceedings of IEEE INFOCOM*, pages 1035–1043, Apr. 2011.

[90] 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, Dec. 2012.

[91] 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 (ICC)*, pages 2078–2084, June 2015.

[92] J. 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, Mar. 2011.

[93] D. Umehara, S. Denno, M. Morikura, and T. Sugiyama. Performance Analysis of Slotted ALOHA and Network Coding for Single-Relay Multi-User Wireless Networks. *Ad Hoc Networks*, 9(2):164–179, Mar. 2011.

[94] D. Umehara, T. Hirano, S. Denno, M. Morikura, and T. Sugiyama. Wireless Network Coding in Slotted ALOHA with Two-hop Unbalanced Traffic. *IEEE Journal on Selected Areas in Communications*, 27(5):647–661, June 2009.

[95] V.Paxson and M.Allman. Computing TCP's Retransmission Timer. RFC 2988, Network Working Group, Nov. 2000.

[96] C.-C. Wang, A. Khreishah, and N. Shroff. Cross-layer Optimizations for Intersession Network Coding on Practical 2-hop Relay Networks. In *Proceedings of Asilomar Conference on Signals, Systems and Computers*, pages 771–775, Nov. 2009.

[97] C.-C. Wang, D. Koutsonikolas, Y. C. Hu, and N. Shroff. FEC-based AP Downlink Transmission Schemes for Multiple Flows: Combining the Reliability and Throughput Enhancement of Intra- and Inter-flow Coding. *Performance Evaluation*, 68(11):1118–1135, Nov. 2011.

[98] S. Wang, G. Tan, Y. Liu, H. Jiang, and T. He. Coding Opportunity Aware Backbone Metrics for Broadcast in Wireless Networks. In *Proceedings of IEEE INFOCOM*, pages 275–279, Apr. 2013.

[99] 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 (WDTN)*, pages 284–291, Aug. 2005.

[100] 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 INFOCOM*, pages 765–773, Apr. 2015.

[101] L. Xie, P. H. Chong, I. W. Ho, and Y. Guan. A Survey of Inter-flow Network Coding in Wireless Mesh Networks with Unicast Traffic. *Computer Networks*, 91(C):738–751, Nov. 2015.

[102] 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 (ICC)*, pages 2871–2876, May 2008.

[103] 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.

[104] 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, Jan. 2014.

[105] C. Zhang, Y. Chen, and C. Li. TCP Adaptation with Network Coding and Opportunistic Data Forwarding in Multi-hop Wireless Networks. *PeerJ Computer Science*, 2:e89, Oct. 2016.

[106] J. Zhang, Y. P. Chen, and I. Marsic. MAC-layer Proactive Mixing for Network Coding in Multi-hop Wireless Networks. *Computer Networks*, 54(2):196–207, Feb. 2010.

[107] S. Zhang, S. C. Liew, and P. P. Lam. Hot Topic: Physical-layer Network Coding. In *Proceedings of the 12th Annual International Conference on Mobile Computing and Networking (MobiCom)*, pages 358–365. ACM, Sept. 2006.

[108] Z. Zhong and S. Nelakuditi. On the Efficacy of Opportunistic Routing. In *Proceedings of IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks*, pages 441–450, June 2007.

[109] 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 (ICC)*, pages 5276–5281, June 2015.

[110] X. Zhu, H. Yue, and Y. Wang. C&M: A New Network Coding Scheme for Wireless Networks. In *Proceedings of 5th International Conference on Information Assurance and Security (IAS)*, volume 2, pages 432–436, Aug. 2009.