Network Coding for High-Performance Communications

Network Coding for High-Performance Communications

by

Kaikai Chi

Submitted to
Tohoku University
in partial fulfillment of the requirements
for the degree of
Doctor of Philosophy

Supervisor: Professor Susumu Horiguchi

Graduate School of Information Sciences
Tohoku University

March, 2009

Abstract

In existing packet-switched networks, each network node functions as a switch in the sense that it either relays information from an input link to an output link (in unicast sessions), or replicates information received from an input link and sends it to a certain set of output links (in multicast sessions). From the information-theoretic point of view, however, there is no reason to restrict the function of a node to that of a switch. Rather, a node can function as an encoder in the sense that when it receives information from the input links, it can encode the information and then send the encoded information to the output link(s). From this point of view, a switch is a special case of an encoder. In communication networks coding at network nodes is called *network coding*.

The network coding technique was originally proposed to increase the throughput of multicast connections in wired networks, and later was shown to be able to offer benefits (like transmission efficiency, computational efficiency, robustness, etc.) for other communication cases also (like multiple unicast, two-source multicast, broadcast, etc.), in both wired and wireless networks. In this thesis, we study the application of network coding in both wired networks and wireless networks. Specifically, Chapter 2 in this thesis studies the application of network coding in wired networks, and Chapters 3, 4 and 5 study the application of network coding in wireless networks. In summary, this thesis presents the following contributions to the theory and application of network coding.

Topology design of coding-based multicast networks In Chapter 2, for the first time, we study the topology design problem of multicast networks when network coding is applied to efficiently support multicast. We first formally formulate the optimal topology design of network coding-based multicast networks as a mixed-integer programming problem, which is proved to be NP-hard. The mathematical formulation can help us assess the essence and understand the hardness of this problem well. Then we propose efficient algorithms to design the low-cost topology of network coding-based multicast networks, which not only have low computational complexity but also can take full advantage of the characteristics of network coding-based multicast to save bandwidth in the design process.

Efficient coding in multihop wireless networks In Chapter 3, we significantly en-

hance the promising coding-based packet forwarding architecture COPE, which is designed for multihop wireless networks to let network nodes intelligently encode multiple packets of different unicast flows together and forward these packets via one transmission. Specifically, we first propose a flow-oriented virtual queue structure that can dramatically increase the packet coding opportunities and also can completely eliminate the packet reordering. We then formulate the corresponding optimal packet coding problem as an optimization problem and prove its NP-completeness. Finally, we present an efficient coding algorithm for finding good coding solutions, such that the transmission efficiency of network nodes can be greatly improved.

QoS-guaranteed coding in multihop wireless networks In Chapter 4, we extend the work in Chapter 3 by taking the QoS issue into account and propose a QoS-guaranteed COPE-type packet forwarding architecture. Specifically, we first present a queueing structure for COPE, which can provide more potential coding opportunities, and then propose a new packet scheduling algorithm for this queueing structure to guarantee different priorities for different types of packets. Finally, we propose an efficient coding algorithm for finding good coding solutions.

Reliable multicast in wireless networks In Chapter 5, we further extend the work in Chapters 3 and 4 by applying network coding to efficiently support reliable multicast in wireless networks. We first prove that in the current coding-based reliable multicast schemes for wireless networks, the search problem for the optimal set of lost packets to encode is NP-complete. We then propose two improved schemes, which can not only achieve low time-complexity but also obtain more coding opportunities to effectively improve the transmission efficiency.

Acknowledgments

I am truly and deeply indebted to so many people that there is no way to acknowledge them all, or even any of them properly. Without their help and encouragement, this work would not and could not have been done. I extend my deepest gratitude to all.

First and foremost, I am deeply indebted to my supervisor, Professor Susumu Horiguchi, for all that he has taught me and all that he has done to aid my development–both professionally and personally. I wish to thank him for his kindness, his constant support in my scientific research and my life, and his invaluable advice during the course of my Ph.D. study. In addition, he creates such a place as the Horiguchi Lab where the creative environment, people associated, and resources available combine to make the place stimulating, exciting and productive.

Also, I am deeply grateful to Associate Professor Xiaohong Jiang, for his insightful guidance, positive encouragement and continuous support in both my research and my life. I learned very much from many discussions with him and now I cannot think about problem without his influence. Without him, this thesis could not have been finished.

I would like to give my sincere thanks to everyone of the Horiguchi Lab in Tohoku University. Their kindness and help let me live a happy life in Sendai. In particular, I want to thank Dr. Masaru Fukushi, who has helped me a lot in the past three years. Additionally, I have no words to express my thanks for my PhD colleagues. I has benefited a lot from the discussion with Xu Zhang, Ahmed Shawish, Yusuke Fukushima, Xiaoliang Wang, Jianming Liu, Dalia Mohamed Nashat and many other people.

My thanks also go to my dear friends: Xinxin Fan, Zhuo Lv, Youxi Zhou, Xiaohong Di, Jianguo Zhan, Jue Lou, Yuan Yang, Yi Liu, Lin Li, Junfeng Li, Chen Yang, Guangyao Lin and so on. Please forgive me for not mentioning all of your names individually. I wish no one of my friends to feel that I am not grateful for the favor they have shown me. It is my big pleasure to be a friend of yours. With your friendship, the time of my life became joyful and memorable.

This thesis is dedicated to my parents and my older sister, whose love and unconditional support provide a constant inspiration in my life.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

In today's practical communication networks, each network node functions as a switch in the sense that it either relays information from an input link to an output link (in unicast sessions), or replicates information received from an input link and sends it to a certain set of output links (in multicast sessions). Due to the nodes' switching operation, the end-to-end data delivery is performed by routing (i.e., by having intermediate nodes store and forward packets). However, from the information-theoretic point of view, there is no reason to restrict the function of a node to that of a switch. Rather, a node can function as an encoder in the sense that when it receives data from the input links, it can encode (i.e., computing certain functions of) the received date and then send the encoded data to the output link(s). From this point of view, a switch is a special case of an encoder. In communication networks coding at network nodes is called *Network coding* [1].

Network coding was introduced by Ahlswede *et al.* in their pioneering work [1], where

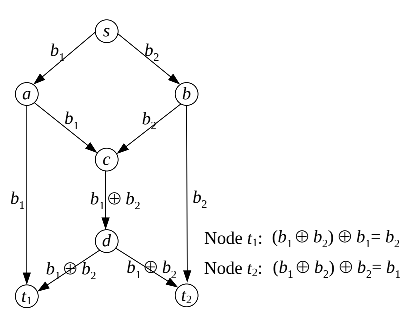


Figure 1.1: A one-source two-sink network with coding.

it was shown that by using network coding the maximum multicast throughput can be achieved. The principle of network coding can be easily explained by considering a simple multicast example (from [1]) shown in Figure 1.1. All links there are error-free and have a capacity of one bit per unit time. Source node s has to transmit data to sink nodes t_1 and t_2 at the rate of two bits per unit time. We can see that this network problem can be satisfied if node c can perform network coding as shown in Figure 1.1, but cannot be satisfied by only forwarding bits at intermediate nodes.

Since the propose of network coding, this topic has been undergoing an active development in the research community. Various studies reported in the past years have resulted in a significant advance in our understanding of network coding. So far, it has been shown that this generality of network coding over routing can, in both wired networks and wireless networks, provide many potential advantages, such as throughput improvement, resource efficiency, computational efficiency, and robustness to network dynamics, etc.

1.1 Coding in Wired Networks

So far, research on the application of network coding in wired networks mainly focus on the multicast and the peer-to-peer (P2P) systems. Some work also studies the application of network coding to network management [2, 3], network tomography [4, 5], security [6, 7, 8, 9, 10], distributed storage [11], etc. Below, we review the application of network coding in multicast and P2P systems, respectively.

1.1.1 Multicast

Efficient (single-source) multicast is one of the central problems in communication networks.

Ahlswede *et al.* showed in [1] that network coding can help to achieve the maximum multicast throughput (i.e. the multicast capacity) which the traditional non-coding multicast usually cannot achieve. After this seminal work on network coding, Li *et al.* [12] soon showed that linear network coding is enough to achieve the multicast capacity. [13, 14] showed that there exist directed graphs where the throughput gains of using network coding for multicast can be very significant. However, in undirected graphs (e.g., a wired network where all links are half-duplex) the throughput gain is at most a factor of two [15]. In [16], Wu *et al.* performed the comparison of achievable throughput of network coding solutions and non-coding solutions in the network topologies of six commercial Internet service providers, showing a small throughput gain in this case.

Given the multicast rate to be supported, network coding can help to establish multicast connections with significantly lower bandwidth consumption than that consumed by Steiner tree-based multicast transmissions [17]. The establishment of minimum-cost multicast connection can be decomposed into two phases: *routing* (determining which links and how much bandwidth resource will be used) and *coding solution construction* (defining the operation function of each node on the selected multicast route). In [17], Lun *et al.* showed that the minimum-cost routing of single multicast connection with network coding can be posed as a linear optimisation problem and proposed a distributed minimum-cost routing algorithm. As for the coding solution construction, much work has been done on this problem. In [18, 19], Koetter and Medard presented an algebraic condition for checking the validity of a linear coding solution to a given multicast connecting problem. Sanders *et al.* [20] and Jaggi *et al.* [21, 22, 23] presented centralized deterministic polynomial-time algorithms, and Fragouli *et al.* [24] proposed decentralized deterministic algorithms allowing to locally specify the coding operations at network nodes without knowledge of the overall network topology. Ho *et al.* [25, 26, 27] studied the decentralized random network coding problem for a feasible multicast connection problem. They gave an upper bound on failure probability, which is on the order of the inverse of the size of the finite field. Thus, the failure probability can be made arbitrarily small by coding in a sufficiently large finite field. Chou *et al.* presented a practical distributed scheme for random network coding [28], that obviates the need for centralized knowledge of the graph topology, the encoding functions, and the decoding functions, and furthermore obviates the need for information to be communicated synchronously through the network.

1.1.2 P2P Systems

Gkantsidis *et al.* [29, 30] presented the implementation of a P2P content distribution system (Avalanche) that uses network coding. Both simulation studies and realistic experiments demonstrated that network coding may improve the overall performance of peer-to-peer content distribution.

Rather than P2P content distribution, Wang *et al.* [31] applied network coding to P2P live multimedia streaming. The most critical requirement of P2P live multimedia streaming applications is that the streaming rate has to be maintained for smooth playback. Through a realistic testbed called Lava, they showed that network coding makes it possible to perform streaming with a finer granularity, which reduces the redundancy of bandwidth usage, improves resilience to network dynamics, and is most instrumental when the bandwidth supply barely meets the streaming demand.

1.2 Coding in Wireless Networks

Recently, network coding has gained much popularity in wireless networks. So far, considerable efforts have been devoted to demonstrate the benefits of using network coding for different communication paradigms in wireless networks (such as the unicast, multicast and broadcast).

1.2.1 Unicast

For the unicast scenario, Wu *et al.* [32] showed that the exchange of independent information between two nodes in a multihop wireless network can be efficiently performed by exploiting both the network coding and physical-layer broadcast. Li *et al.* [33, 34] studied the cases of multiple unicast sessions, where network coding can only provide marginal benefits. Recently, Katti *et al.* [35] proposed a practical network coding-based packet forwarding architecture (called COPE) to essentially improve the network throughput of multihop wireless networks. Based on the COPE-type XOR coding scheme, the coding-aware routing was proposed in [36, 37]. Some efforts (e.g. [37, 38, 39, 40]) have also been made to theoretically evaluate the performance (like throughput) of COPE-type wireless networks. More recently, the physical-layer network coding was proposed to utilize wireless interference for network coding [41, 42].

1.2.2 Multicast

As for multicast case, Wu *et al.* [43] showed that in a mobile ad hoc network, adopting network coding for minimum-cost multicast can be formulated as a linear optimization problem and solved in polynomial time. The corresponding decentralized algorithms were further proposed in [17] to establish the minimum-cost multicast tree. The theoretical throughput analysis of multicast with network coding has also been conducted in [44] for unreliable ad hoc networks.

For tree-based reliable multicast in multihop wireless networks, Ghaderi *et al.* [45, 46] analytically quantified the reliability gain (expressed as the expected number of retransmissions) of using network coding, compared to ARQ and end-to-end FEC. For link-layer (i.e. one-hop) reliable multicast in wireless networks, Nguyen *et al.* [47] studied the use of network coding and showed through analysis and simulations that it is more bandwidth-efficient, compared to ARQ. The analysis is overly simplified with many unrealistic assumptions, e.g., they consider that ACKs/NACKs are never lost and can reach the source with zero delay.

For unreliable multicast, Park *et al.* [48] proposed CodeCast, a network coding-based protocol for increasing reliability in multimedia multicast applications in MANETs. They showed that network coding offers high reliability, however this protocol cannot guarantee 100% packet delivery ratio.

1.2.3 Broadcast

Concerning the application of network coding for broadcast in wireless ad hoc networks, some work also has been done recently. For group communication, where each node of the network is a source that wants to transmit information to all other nodes, distributed probabilistic broadcast algorithms and deterministic broadcast algorithms have been proposed by Fragouli *et al.* [49, 50] and Li *et al.* [51], respectively, resulting in a significant energy saving. For the reliable broadcast with only one source node, Hou *et al.* [52] proposed a coding-based protocol (called AdapCode) which adaptively changes the coding scheme according to the link quality to reduce broadcast traffic.

1.3 Thesis Outline

In this thesis, we study the application of network coding in both wired networks and wireless networks. Chapter 2 studies the application of network coding in wired networks, and Chapters 3, 4 and 5 study the application of network coding in wireless networks. In more detail, the thesis is organized as follows.

Chapter 2 studies the challenging topology design problem of network coding-based multicast networks. Based on the characteristics of multicast and network coding, we formulate this problem as an NP-hard mixed-integer nonlinear programming problem, which is much more complicated than the conventional unicast-oriented topology design problems. Then we propose two heuristic algorithms for this topology design problem. Finally, simulation results in this chapter show that in comparison with the conventional unicast-oriented design for multicast networks, the Steiner tree-based design has moderate improvement in term of topology cost, but the network coding-based design can make this improvement very significant.

Chapter 3 enhances the current COPE architecture (a promising coding-based packet forwarding architecture) by first proposing a flow-oriented virtual queue structure for it and then introducing an efficient algorithm for searching good coding solutions under the new queue structure. This queue structure can not only completely eliminate the packet reordering but also offer the maximum number of coding opportunities under the condition that no packet reordering is allowed. Extensive simulation results demonstrated that the available COPE can improve the node transmission efficiency, but this improvement can be more significant when the proposed virtual queue structure and new coding algorithm are jointly adopted.

Chapter 4 extends the work in Chapter 3 by taking the QoS issue into account. Specifically, we present for the COPE architecture a new QoS queueing structure which can increase the potential coding opportunities and are convenient for the allocation of priorities to packets, and also proposes a new efficient packet coding algorithm. Rather than adopting FIFO scheduler, allocating priorities to different flows can satisfy the QoS requirement of multihop wireless networks for supporting real-time services such as voice applications. To our knowledge, this is the first time to take the QoS issue into account in the literature of wireless network coding. Simulation results demonstrate that by adopting the new queueing structure and new coding algorithm, COPE can further greatly improve the node transmission efficiency.

Chapter 5 further extends the work in Chapters 3 and 4 by applying network coding to efficiently support reliable multicast in wireless networks (including multihop wireless networks). Specifically, we present two efficient network coding-based schemes for the reliable link-layer multicast: a static one with low complexity and a dynamic one with relatively higher complexity but a better performance. Unlike the available network coding-based schemes which have exponential computational complexity, the proposed schemes run in polynomial time. We evaluate, by both theoretical analysis and computer simulation, the performance of our schemes. Compared with the available coding-based schemes, the proposed schemes can more effectively reduce the bandwidth consumption, especially in the case of high packet loss probabilities and many receivers.

In Chapter 6, we give a final perspective on our work and outline some future work in this area.

1.4 Thesis Contributions

The thesis contributions are summarized below.

- For the first time, we formally formulate the optimal topology design of network coding-based networks as a mixed-integer programming problem, which is NP-hard. The mathematical formulation can help us assess the essence and understand the hardness of this problem well. We also propose efficient algorithms to design the low-cost topology of network-coding-based multicast networks, which not only take full advantage of the characteristics of network coding-based multicast to save bandwidth in the design process and but also have low computational complexity.

- We significantly improve a promising coding-based packet forwarding architecture COPE. Specifically, we first propose a flow-oriented virtual queue structure that can dramatically increase the coding opportunities and also can completely eliminate the packet reordering. We then formulate the corresponding optimal packet coding problem as an optimization problem and prove its NP-completeness. Finally, we present an efficient coding algorithm for finding good coding solutions.

- We propose a QoS-guaranteed COPE-type packet forwarding architecture. Specifically, we first present a new queueing structure for COPE, which can provide more potential coding opportunities, and then propose a new packet scheduling algorithm for this queueing structure to guarantee different priorities for different types of packets. Finally, we propose an efficient coding algorithm to find appropriate packets for coding.

- We prove that in the current coding-based reliable multicast schemes for wireless networks, the search of the optimal set of lost packets for encodingS is NP-complete. We then propose two improved schemes which not only can effectively improve the transmission efficiency but also have low time-complexity such that they are scalable to large number of multicast receivers.

Chapter 2

Topology Design of Network Coding-Based Multicast Networks

With the advance of communication networks, a great number of multicast applications such as video conferencing have emerged and it is foreseeable that more multicast applications will emerge in the near future. As many multicast services require the transmission of video streaming traffic, future networks will need to support a considerable amount of multicast traffic.

Owing to the high capability to efficiently support multicast transmissions, the network coding technique is promising to be applied in future multicast networks. Consequently, network coding-based multicast (NCM) network design with the consideration of efficiently supporting multicast by network coding technique becomes an important issue now. Complete network design involves a lot of aspects, such as traffic matrix estimation, topology design, node function specification and management[53]. Topology design is one of the most important aspects of network design.

Network topology design has long been a challenging problem. Given the number of nodes, physical locations of these nodes, knowledge of communication lines available and traffic requirements, topology design is to assign communication links, capacity of each link and flow of each traffic requirement. These assignments should keep the resulting topology cost as low as possible while satisfying a set of requirements, such as delay requirement and reliability requirement. The topology optimization problem is generally an NP-hard combinatorial optimization problem [54, 55], and quickly becomes intractable as the number of nodes increases. Conventional topology design problems only considered unicast requirements due to the fact that at that time there were no or few multicast applications. So far, a number of unicast-oriented heuristic algorithms have been proposed to deal with the specific topology design problems, including some classic ones such as Branch Exchange, Cut Saturation and MENTOR Algorithm[55, 56, 57], and some modern ones such as Tabu Search, Simulated Annealing and Genetic Algorithm [58, 59, 60].

The topology design problem of NCM networks is based on the assumption that network nodes have the capability of performing encoding, and is more difficult than traditional ones. Two aspects distinguish this problem from conventional ones. First, multicast requirements are considered in this problem. Second, network coding technique is applied to support multicast transmissions. The consideration of network coding-based multicast increases the complexity of optimal routing subproblem and the corresponding topology design problem, because the NCM routing complexity is much higher than that of unicast case[17] and routing procedure must be embedded in topology design algorithms. Therefore, effective topology design heuristics should be developed for NCM networks. How to take full advantage of the characteristics of network coding-based multicast to save bandwidth in the design process and at the same time keep the algorithm complexity as low as possible is the challenge the topology designers have to face.

In this chapter we consider the topology design problem of NCM networks. The main contributions of this chapter are summarized as follows:

1. For the first time we formally formulate the optimal topology design of NCM networks as a mixed-integer programming problem, which is NP-hard. The mathematical formulation can help us assess the essence and understand the hardness of this problem well.

2. Two heuristic algorithms, *link deletion and exchange algorithm* and *link addition and exchange algorithm*, are proposed for the efficient topology design of NCM networks.

3. We demonstrate that through adopting network coding technique to support multicast transmissions, we can design a multicast network topology with significantly lower network cost than that of the conventional unicast-oriented and Steiner tree-based designs.

This chapter is organized as follows. Section 2.1 defines the topology design problem and formally formulates it as a mixed-integer nonlinear programming problem. Two heuristic algorithms for our NCM topology design problem are introduced in Section 2.2. Section 2.3 presents simulation results to evaluate their performance, and demonstrates the benefit offered by network coding technique in network topology design as well. Finally, Section 2.4 concludes the chapter.

2.1 Problem Statement and Formulation

Some important notions in the network topology design are listed as follows.

- **Traffic requirement:** the average number of bits per second sent from a source to a destination or a set of destinations.

- **Network reliability:** the reliability of the overall network to provide communication in the event of failure of a component or components in the network.

- **Topological configuration** (for simplicity, called configuration): the set of links connecting network nodes together.

- **Capacity assignment:** the determination of the maximum number of bits per second that can be transmitted by each communication link of a given configuration.

- **Flow assignment:** the selection of the route for each traffic requirement.

- **The average packet delay:** the mean time taken by a packet travel from a source node to a destination node.

In available literature, almost all network topology design problems are unicast-oriented. Two significant aspects distinguish the topology design problem of NCM networks from old ones. First, multicast requirements are considered specially. Second, network coding technique is applied to support multicast transmissions. The specific topology design problem of NCM networks we consider can be stated as follows.

Given:

1. number of nodes N and their corresponding locations
2. unicast traffic requirement between each ordered pair of distinct nodes
3. source node, destination nodes and multicast rate of each multicast traffic requirement
4. capacity, fixed cost and cost per unit length of each type of communication line (i.e. cable)
5. reliability requirement
6. delay requirement

Minimize: the overall topology cost

Over:

1. all possible configurations
2. all possible link capacity assignments
3. all possible flow assignments

Subject to:

1. capacity assignment constraint
2. reliability requirement
3. flow conservation constraint
4. link utilization (ratio of the used capacity to the total capacity) constraint
5. delay requirement

Table 2.1: Notation used in Chapter 2

Notation	Meaning
$\mathcal{G}(\mathcal{N}, \mathcal{A})$	Directed graph consisting of a node set $\mathcal{N}$ and an arc set $\mathcal{A}$.
$\{i, j\}$	An unordered node pair called a link. It is the same as $\{j, i\}$.
(i, j)	An ordered node pair called an arc.
N	Number of network nodes.
$d_{i,j}$	Distance between node i and node j. It is the same as $d_{j,i}$.
K	Number of available communication line types.
C_i	Capacity of i-type line ($C_1 < C_2 < \cdots < C_K$).
f_i	Fixed cost of i-type line.
p_i	Cost per unit length of i-type line.
k	Node-connectivity.
e_{max}	Maximum link utilization constraint.
$r_{i,j}$	Traffic requirement rate from node i to node j.
$f_{i,j}^{(i_1,i_2)}$	Amount of unicast flow from node i_1 to i_2 on arc (i, j).
M	Number of multicast traffic requirements.
R_i	Traffic rate of ith multicast requirement.
S_i	Node set of ith multicast requirement. Denote by $n_{i,0}$ the source node, and denote by $n_{i,1}, \cdots, n_{i,\|S_i\|-1}$ destination nodes in this node set.
$g_{i,j}^{(n_{i,0},n_{i,j})}$	Amount of flow from $n_{i,0}$ to $n_{i,j}$ on arc (i, j).
$f_{i,j}$	Total amount of flow on arc (i, j).
$C_{i,j}$	Assigned capacity of link $\{i, j\}$. It is the same as $C_{j,i}$.
$D_{i,j}$	Cost of link $\{i, j\}$. It is the same as $D_{j,i}$.

The notation used in this chapter is shown in Table 2.1. In the remainder of this section, we will deal with different aspects of this problem in detail and finally formulate this problem mathematically.

2.1.1 Capacity Assignment

Only those types of communication lines which are available in the market can be assigned on network links. Thus, the capacity which can be allocated to a link is the combination of available line capacities. Assume that there are K types of communication lines available, with each type of line having a discrete capacity. Then the capacities which can be allocated to each network link $\{i, j\}$ are

$$C_{i,j} = u_{i,j}^1 C_1 + \cdots + u_{i,j}^K C_K,$$

where $u_{i,j}^1, \cdots, u_{i,j}^K \in \{0, 1, 2, \cdots\}$.

2.1.2 Network Cost Model

Topology cost consists of material cost of communication lines, installation cost, network node (such as switch) cost, etc. For simplicity, it is often reasonable to approximately model the cost of nodes as fixed line costs and assume the network cost consists of line costs merely. We assume that the cost of placing a line between two nodes comprises two components: a *fixed cost* related to the capacity of this line and a *variable cost* related to the physical length of this line. The fixed cost of a t-type line, f_t, includes installation cost, the overhead incurred by the endpoints and so on. The variable cost related to length is linear with line length d and its cost per unit length p_t, that is, it equals $p_t \cdot d$. In addition, the total fixed cost of a network topology usually accounts for a significant percentage of the total cost, and the cost per unit capacity per unit length decreases with the increase of line capacity due to the economy of scale.

For a link $\{i, j\}$, one or more communication lines can be placed on it. Thus the cost of link $\{i, j\}$ can be expressed as $D_{i,j} = \sum_{t=1}^{K} u_{i,j}^t (f_t + d_{i,j} \cdot p_t)$ where $u_{i,j}^t$ is the number of t-type lines assigned to link $\{i, j\}$. Index N nodes from 1 to N. Then the overall topology cost is

$$\sum_{i=1}^{N-1} \sum_{j=i+1}^{N} D_{i,j} = \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} \sum_{t=1}^{K} u_{i,j}^t (f_t + d_{i,j} \cdot p_t). \tag{2.1}$$

2.1.3 Network Reliability Requirement

Network links and nodes can fail because of different causes. It is necessary to consider the network reliability at the topology design stage. There are different measures to scale the reliability of a network. Here the concept of k-connectivity is used as the reliability measure. k-connectivity indicates that there are at least k node-disjoint paths available between each pair of nodes. The network is said to be k-node-connected if it satisfies k-connectivity condition.

Define function $F(x)$ as follows: If $x > 0$, $F(x) = 1$; otherwise $F(x) = 0$.

For each node i, the number of links incident to it is $\sum_{j=1,j\neq i}^{N} F(C_{i,j})$. Then the k-connectivity requirement for networks can be formulated as follows [61].

$$\sum_{i\in S}\ \sum_{j\in\{1,2,\cdots,N\}\backslash(Z\bigcup S)} F(C_{i,j}) \geq 1,$$

$$\begin{aligned}
&\forall 1 \leq s, d \leq N (s \neq d),\\
&\forall Z \subseteq \mathcal{N}\backslash\{s,d\} \text{ with } |Z| = k-1,\\
&\forall S \subseteq \mathcal{N}\backslash Z \text{ with } s \in S \text{ and } d \notin S.
\end{aligned} \tag{2.2}$$

2.1.4 Conservation of Flow

The flow conservation law states that, at each node in a communication network, the total incoming flow, plus the flow originating at this node, minus the demand at this node, equals the total outgoing flow. It is easy to understand that unicast flows comply with the flow conservation principle. However, the case of multicast flows is different. At an intermediate node, one ingoing packet of a multicast flow may induce one or several outgoing packets. Thus multicast flows violate the flow conservation principle. Next, we will consider this issue of unicast, Steiner tree-based multicast and network coding-based multicast, respectively.

1) Unicast transmission

For a unicast transmission with rate $r_{s,d}$ from source node s to destination node d, the amount of this unicast traffic into a node must be equal to the amount of this unicast traffic out of this node, unless this node is the source or the destination of this unicast.

The flow conservation constraint can be expressed as:

$$\sum_{\{j:(i,j)\in\mathcal{A}\}} f_{i,j}^{(s,d)} - \sum_{\{j:(j,i)\in\mathcal{A}\}} f_{j,i}^{(s,d)} = \begin{cases} -r_{s,d} & \text{if } i = d, \\ r_{s,d} & \text{if } i = s, \\ 0 & \text{otherwise,} \end{cases}$$

$$\forall i \in \mathcal{N}. \tag{2.3}$$

2) Steiner tree-based multicast transmission

Steiner tree-based multicast transmission with node set $S_t = \{n_{t,0}, n_{t,1}, \cdots, n_{t,|S_t|-1}\}$, is a special combination of $|S_t| - 1$ unicast transmissions. Each unicast flow of them should satisfy flow conservation constraint. Moreover, there is only one path to route message for each unicast from source $n_{t,0}$ to one destination $n_{t,i}(1 \le i \le |S_t| - 1)$. The difference between Steiner tree-based multicast with node set $S_t = \{n_{t,0}, n_{t,1}, \cdots, n_{t,|S_t|-1}\}$ and $|S_t|-1$ unicasts from node $n_{t,0}$ to each node in $\{n_{t,1}, \cdots, n_{t,|S_t|-1}\}$ is that, in the former the consumed resource of each arc (i,j) is the maximum one of $g_{i,j}^{(n_{t,0},n_{t,1})}, \cdots, g_{i,j}^{(n_{t,0},n_{t,|S_t|-1})}$, whereas in the latter the consumed resource of each arc (i,j) is the sum of $f_{i,j}^{(n_{t,0},n_{t,1})}, \cdots,$ $f_{i,j}^{(n_{t,0},n_{t,|S_t|-1})}$. It is this difference that induces the effectiveness of Steiner tree-based multicast in utilizing the available communication resource. The flow constraint of Steiner tree-based multicast transmissions can be expressed as:

$$\sum_{\{j:(i,j)\in\mathcal{A}\}} g_{i,j}^{(n_{t,0},n_{t,l})} - \sum_{\{j:(j,i)\in\mathcal{A}\}} g_{j,i}^{(n_{t,0},n_{t,l})} = \begin{cases} -R_t & \text{if } i = n_{t,l}, \\ R_t & \text{if } i = n_{t,0}, \\ 0 & \text{otherwise,} \end{cases}$$

$$\forall i \in \mathcal{N}, l \in \{1, \cdots, |S_t| - 1\}; \tag{2.4a}$$

$$\sum_{\{j:(i,j)\in\mathcal{A}\}} F(g_{i,j}^{(n_{t,0},n_{t,l})}) \le 1, \ \forall i \in \mathcal{N}; \tag{2.4b}$$

$$\sum_{\{j:(j,i)\in\mathcal{A}\}} F(g_{j,i}^{(n_{t,0},n_{t,l})}) \leq 1,\ \forall i \in \mathcal{N}. \tag{2.4c}$$

3) Network coding-based multicast transmission

When network coding is used, the problem of establishing a multicast connection with node set $S_t = \{n_{t,0}, n_{t,1}, \cdots, n_{t,|S_t|-1}\}$ and traffic rate R_t, equates to two essentially decoupled problems: one is determining the subgraph in current network (i.e., determining how much flow to put on each link), and the other is determining the code to use over that subgraph (i.e. specifying how to encode packets together at each related node.) [62]. The necessary and sufficient condition for the feasibility of a subgraph is shown in Equation (2.5) [62]. Different feasible subgraphs may have different resource consumptions. Once we select a feasible subgraph, any feasible code can be used to implement this multicast connection.

$$\sum_{\{j:(i,j)\in\mathcal{A}\}} g_{i,j}^{(n_{t,0},n_{t,l})} - \sum_{\{j:(j,i)\in\mathcal{A}\}} g_{j,i}^{(n_{t,0},n_{t,l})} = \begin{cases} -R_t & \text{if } i = n_{t,l}, \\ R_t & \text{if } i = n_{t,0}, \\ 0 & \text{otherwise,} \end{cases}$$

$$\forall i \in \mathcal{N}, l \in \{1, \cdots, |S_t| - 1\}. \tag{2.5}$$

Such multicast is another special combination of $|S_t| - 1$ unicasts. Each unicast flow of them satisfies flow conservation constraint as shown in Equation (2.5). However, different from the case in Steiner tree-based multicast, there can be multiple paths to route message simultaneously for each unicast from source $n_{t,0}$ to one destination (that is, no constraints 2.4b and 2.4c). For example, in Figure 1.1, paths $s \to a \to t_1$ and $s \to b \to c \to d \to t_1$ are from s to t_1, and paths $s \to b \to t_2$ and $s \to a \to c \to d \to t_2$ are from s to t_2. Obviously, like the multi-path routing in [63], network coding-based multicast routing can also balance the network load. The optimal routing in [63] applies multi-path routing technique for each unicast connection to achieve system-optimal objective, but it brings no benefit in terms of resource consumption from the perspective of each unicast. For a multicast connection, the purpose of applying network coding-based routing instead of Steiner-tree based routing is to achieve the user-optimal routing, which can significantly reduce the bandwidth consumption of each connection[17] and thus reduce the overall resource consumption in a network. The same as the case in Steiner tree-based multicast, the consumed bandwidth of each arc (i, j) is the maximum one of $g_{i,j}^{(n_{t,0},n_{t,1})}, \cdots, g_{i,j}^{(n_{t,0},n_{t,|S_t|-2})}$and $g_{i,j}^{(n_{t,0},n_{t,|S_t|-1})}$, instead of the sum of them. Therefore, Steiner tree-based multicast is a special case of network coding-base multicast. Network coding-based minimum-cost multicast is at least as effective as Steiner tree-based multicast, and generally more effective than Steiner tree-based multicast[17].

2.1.5 Network Coding-Based Minimum-Cost Multicast

Denote by $a_{i,j}$ the cost per unit flow on arc (i, j). In a network coding-based network represented by $\mathcal{G}(\mathcal{N}, \mathcal{A})$, the problem of constructing a single minimum-cost multicast connection with node set $S_t = \{n_{t,0}, n_{t,1}, \cdots, n_{t,|S_t|-1}\}$ can be formulated as follows [17, 43, 62, 64]:

Minimize: $\sum_{(i,j)\in\mathcal{A}} a_{i,j} \cdot z_{i,j}$

Subject to:

$$z_{i,j} \geq g_{i,j}^{(n_{t,0},n_{t,l})}, \forall (i,j) \in \mathcal{A}, l \in \{1, \cdots, |S_t| - 1\}; \tag{2.6}$$

$$\sum_{\{j:(i,j)\in\mathcal{A}\}} g_{i,j}^{(n_{t,0},n_{t,l})} - \sum_{\{j:(j,i)\in\mathcal{A}\}} g_{j,i}^{(n_{t,0},n_{t,l})} = \begin{cases} -R_t & \text{if } i = n_{t,l}, \\ R_t & \text{if } i = n_{t,0}, \\ 0 & \text{otherwise}, \end{cases}$$

$$\forall i \in \mathcal{N}, l \in \{1, \cdots, |S_t| - 1\}; \tag{2.7}$$

$$C_{i,j} \geq g_{i,j}^{(n_{t,0},n_{t,l})} \geq 0,\ \forall (i,j) \in \mathcal{A}, l \in \{1, \cdots, |S_t| - 1\}. \tag{2.8}$$

This is a linear programming problem with polynomial-time algorithms to obtain the optimal solution. In our topology design algorithms, we regard distance $d_{i,j}$ as $a_{i,j}$ and construct the minimum-cost multicast connection for each multicast requirement.

2.1.6 Link Utilization Constraint

We assume that communication lines are bi-directional (i.e. signals can be carried in both directions simultaneously). This assumption is true in most practical cases. In a network $\mathcal{G}(\mathcal{N}, \mathcal{A})$, the total amount of unicast flows and multicast flows on an arc (i, j) should be less than or equal to $C_{i,j}$, the capacity assigned to link $\{i, j\}$. This constraint can be expressed as:

$$\sum_{i_1=1}^{N} \sum_{\substack{i_2=1 \\ i_2 \neq i_1}}^{N} f_{i,j}^{(i_1,i_2)} + \sum_{t=1}^{M} \max_{l \in \{1,\cdots,|S_t|-1\}} g_{i,j}^{(n_{t,0},n_{t,l})} \leq C_{i,j},$$

$$\forall (i, j) \in \mathcal{A}. \tag{2.9}$$

The first term on the left-hand side of (2.9) is the total amount of unicast traffic on arc (i, j) and the second term is the total amount of network coding-based multicast traffic on arc (i, j). Note that, as mentioned previously, for tth multicast the amount of traffic on arc (i, j) is the maximum one of $|S_t| - 1$ unicast flows, i.e. $\max_{l \in \{1,\ldots,|S_t|-1\}} g_{i,j}^{(n_{t,0},n_{t,l})}$, instead of the sum of $|S_t| - 1$ unicast flows on arc (i, j).

2.1.7 Delay Requirement

It is necessary to keep the average end-to-end packet (AEEP) delay (a network-wide metric) within an admissible value. In most available literature, M/M/1 queueing model based on Kleinrock's independence assumptions is adopted to calculate the average packet delay on each network link. Based on this model, the AEEP delay can be expressed as

$$T = \frac{1}{\gamma} \sum_{(i,j) \in \mathcal{A}} \frac{f_{i,j}}{C_{i,j} - f_{i,j}} \tag{2.10}$$

where γ is the total arrival rate into the network in packets per second; $f_{i,j}$ and $C_{i,j}$ are the total traffic rate on arc (i, j) and the capacity of arc (i, j) in bits per second [56, 63].

However, it is inappropriate to still apply this model to current high-speed multi-service networks. One reason is that Equation (2.10) considers neither propagation delay nor nodal processing delay, both of which are very important in high-speed networks where it is unrealistic to neglect them. Another important reason is that, high-speed networks are capable of carrying many types of services such as voice, data and video, whose corresponding packets are probably separated in different queues with different priorities, rather than one queue.

The appropriate delay model for current and future networks is related with the specific packet scheduling scheme adopted, and it is far more complex than the traditional one. It is not desirable to embed a burdensome analysis of delay in the complex topology design. In addition, it is possible that in a network meeting the AEEP delay constraint, most requirements have small average end-to-end packet delays and some requirements have large average end-to-end packet delays. It is preferable to create a more balanced design. The more balanced design is also better able to withstand variations in requirement level and distribution.

A delay-balanced design can be obtained by limiting the utilization of each arc separately [57]. In our topology design problem, a limit (or threshold) is imposed on the utilization of each arc to control packet delay. Denote the maximum permitted utilization of each arc by $e_{\max}$. Regretfully, we can not get an explicit relationship between parameter e_{max} and the AEEP delay. Nevertheless, some literatures have studied the effect of link utilization on the delay performance [65, 66, 67], and obtained some results. For example, for a link loaded with TCP traffic composed by many TCP connections, when the global offered load increases above 80%, the performance of each single connection decreases very quickly [65]. The results of these papers can provide us some general guidelines about value specification of parameter e_{max}.

This constraint on arc utilization is more stringent than previous link utilization constraint.

2.1.8 Formulation

Now the topology design problem we consider can be formulated as follows.

Given:

1. node number N and distance matrix $(d_{i,j})_{N\times N}$
2. unicast requirement matrix $(r_{i,j})_{N\times N}$
3. the node set $\{n_{i,0}, n_{i,1}, \cdots, n_{i,|S_i|-1}\}$ and the traffic rate R_i of ith multicast requirement $(i = 1, 2, \cdots, M)$
4. capacities $C_1, \cdots, C_K$, fixed costs $f_1, \cdots, f_K$ and costs per unit length $p_1, \cdots, p_K$ of different types of lines
5. connectivity k
6. maximum arc utilization $e_{\max}$

Minimize:

$$\sum_{i=1}^{N-1}\sum_{j=i+1}^{N} D_{i,j} = \sum_{i=1}^{N-1}\sum_{j=i+1}^{N}\sum_{t=1}^{K} u_{i,j}^{t}(f_t + d_{i,j}\cdot p_t)$$

Over the design variables:

$u_{i,j}^1, \cdots, u_{i,j}^K \in \mathbb{N} : 1 \le i \le N-1,\ i+1 \le j \le N$

$f_{i,j}^{(s,d)} \ge 0 : 1 \le i, j, s, d \le N\ (i \ne j, s \ne d)$

$g_{i,j}^{(n_{t,0},n_{t,l})} \ge 0 :\ t \in \{1, \cdots, M\}, l \in \{1, \cdots, |S_t| - 1\}, 1 \le i, j \le N(i \ne j)$

Subject to:

1) $C_{i,j} = u_{i,j}^1 C_1 + \cdots + u_{i,j}^K C_K$ where $u_{i,j}^1, \cdots, u_{i,j}^K \in \{0, 1, 2, \cdots\}$, $C_{j,i} = C_{i,j}$, $\forall 1 \le i \le N-1,\ i+1 \le j \le N$.

2) network reliability requirement:

$$\sum_{i\in S}\ \sum_{j\in\{1,2,\cdots,N\}\setminus(Z\bigcup S)} F(C_{i,j}) \ge 1,$$

$$\forall 1 \le s, d \le N (s \ne d),$$

$$\forall Z \subseteq \mathcal{N} \backslash \{s, d\} \text{ with } |Z| = k-1,$$

$$\forall S \subseteq \mathcal{N} \backslash Z \text{ with } s \in S \text{ and } d \notin S.$$

3.1) unicast flow conservation constraint:

$$\sum_{\substack{1 \le j \le N \\ j \ne i}} f_{i,j}^{(s,d)} - \sum_{\substack{1 \le j \le N \\ j \ne i}} f_{j,i}^{(s,d)} = \begin{cases} -r_{s,d} & \text{if } i = d, \\ r_{s,d} & \text{if } i = s, \\ 0 & \text{otherwise,} \end{cases}$$

$$\forall 1 \le i, s, d \le N (s \ne d).$$

3.2) multicast flow conservation constraint:

$$\sum_{\substack{1 \le j \le N \\ j \ne i}} g_{i,j}^{(n_{t,0}, n_{t,l})} - \sum_{\substack{1 \le j \le N \\ j \ne i}} g_{j,i}^{(n_{t,0}, n_{t,l})} = \begin{cases} -R_t & \text{if } i = n_{t,l}, \\ R_t & \text{if } i = n_{t,0}, \\ 0 & \text{otherwise,} \end{cases}$$

$$\forall t \in \{1, \cdots, M\}, l \in \{1, \cdots, |S_t| - 1\}, 1 \le i \le N.$$

4) link utilization constraint & delay requirement :

$$f_{i,j} = \sum_{i_1=1}^{N} \sum_{\substack{i_2=1 \\ i_2 \ne i_1}}^{N} f_{i,j}^{(i_1, i_2)} + \sum_{t=1}^{M} \max_{l \in \{1, \cdots, |S_t|-1\}} g_{i,j}^{(n_{t,0}, n_{t,l})}$$

$$\le e_{max} \cdot C_{i,j}, \ \forall 1 \le i, j \le N (i \ne j).$$

Compared with traditional topology design problems, this problem has an additional constraint, i.e., flow conservation constraint of network coding-based multicast transmissions. In addition, because there are multicast transmissions, when compared with conventional problems constraint (4) has an additional term reflecting the characteristic of network coding.

Lemma 1. *The topology design problem of survivable (i.e. k-node connected) unicast networks is NP-hard.*

Proof. This topology design problem is NP-hard even when the traffic requirement $r_{i,j}$ ($i, j \in V$ and $i \neq j$) is very small such that the smallest capacity C_1 is enough for each link to be assigned, because it contains some known NP-hard problems, such as the traveling salesman problem and connectivity augmentation problem, as special cases [54, 55]. □

Theorem 1. *The topology design problem of survivable network coding-based multicast networks is NP-hard.*

Proof. This new topology design problem of survivable network coding-based multicast networks contains the traditional unicast-oriented design problem as a special case and thus is also NP-hard. □

No polynomial-time algorithms are available to obtain the optimal solution of an NP-hard optimization problem. It is necessary to develop heuristic algorithms to deal with it.

2.2 Heuristic Algorithms for Topology Design

In this section, we will introduce two heuristic algorithms, link deletion and exchange (LDE) algorithm and link addition and exchange (LAE) algorithm, for this topology design problem.

These two proposed algorithms are both composed of two phases, starting topology generation and local optimization process. In the first phase of LDE algorithm, through deleting links one by one from the fully connected topology until no one link can be deleted any more, a k-node-connected starting topology with relatively low cost is generated. In the first phase of LAE algorithm, through adding links one by one from the original topology with no link until no one more link is needed any more, a k-node-connected starting topology with relatively low cost is generated. In the second phase of both

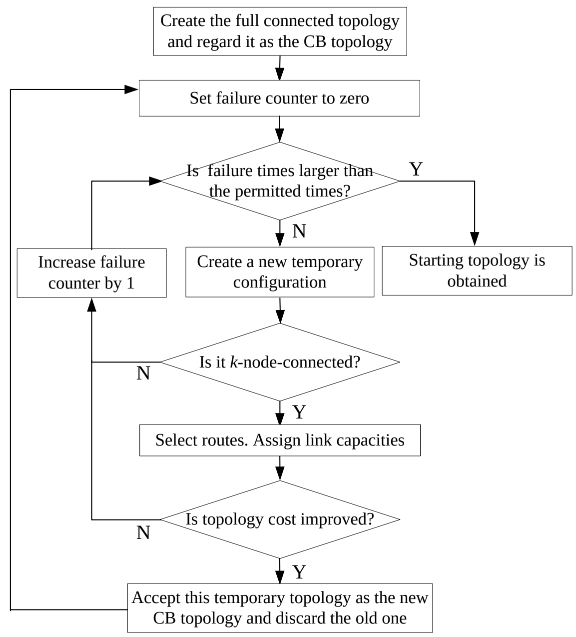


Figure 2.1: Flow chart of starting topology generation in LDE algorithm.

algorithms, link exchange is iteratively performed to locally improve the starting topology step by step.

For simplicity, we first consider the case that only a line can be assigned to each link $\{i, j\}$, that is, $C_{i,j} \in \{0, C_1, \cdots, C_K\}$. Then these two algorithms will be extended to the general case that several communication lines can be assigned on each link.

2.2.1 Link Deletion and Exchange (LDE) Algorithm

1) Starting topology generation

The objective of this phase is to generate a k-node-connected topology whose cost is relatively low. The flow chart of this phase is shown in Figure 2.1.

First, create the fully connected topology and regard it as the current best (CB) topology. Then obtain a temporary configuration by deleting a particular link in current configuration. If this temporary configuration satisfies some particular conditions, it means that based on this temporary configuration a new feasible topology with lower cost can be obtained. Accept this new feasible topology as the new CB topology, discard the old one, and set parameter t, which is a counter parameter used to count the continuous failure times, back to zero. If this temporary configuration does not satisfy all those conditions, discard it and increase t by one. If the value of t exceeds a given value $t_{\max}$, terminate the algorithm and the CB topology is the final topology of this phase. Otherwise, obtain another temporary configuration and test it. In this way, link deletion operation is conducted repeatedly until no appropriate link can be deleted any more.

Define an efficiency metric $m_{i,j}$ on each link $\{i,j\}$ by $m_{i,j} = D_{i,j}/(f_{i,j} + f_{j,i})$.

This process consists of following detailed steps.

1. Index N nodes from node 1 to N randomly, and create the fully connected configuration. Then select the route for each requirement and allocate link capacities. Regard the resulting topology as the CB topology.

   Routing and capacity allocation procedure:

   For each unicast requirement select the shortest distance path between source node and determination node as its route, and for each multicast requirement select the route obtained by network coding-base minimum-cost multicast algorithm as its route [1].

   For each link $\{i,j\}$ assign to it the smallest capacity in the set $\{0, C_1, \cdots, C_K\}$ which is greater than or equal to

$$\frac{1}{e_{max}}\left(\sum_{i_1=1}^{N} \sum_{\substack{i_2=1 \\ i_2 \neq i_1}}^{N} f_{i,j}^{(i_1,i_2)} + \sum_{t=1}^{M} \max_{l \in \{1,\cdots,|S_t|-1\}} g_{i,j}^{(n_{t,0},n_{t,l})}\right) \tag{2.11}$$

   and

$$\frac{1}{e_{max}}\left(\sum_{i_1=1}^{N} \sum_{\substack{i_2=1 \\ i_2 \neq i_1}}^{N} f_{j,i}^{(i_1,i_2)} + \sum_{t=1}^{M} \max_{l \in \{1,\cdots,|S_t|-1\}} g_{j,i}^{(n_{t,0},n_{t,l})}\right). \tag{2.12}$$

2. Set counter parameter t to zero and initialize E, which consists of the candidate links to delete, to the set consisting of all links in the CB topology.

3. Check whether the value of t is larger than $t_{\max} = \lceil N \cdot k/2 \rceil$. If it is, go to Step 7.

4. From E, select the link l whose efficiency metric value is largest. Obtain a temporary

[1]The minimum-cost multicast route here is obtained by relaxing (discarding) the constraints (2.8), that is, each link capacity is considered as infinite.

configuration by removing link l from current configuration.

Test whether this temporary configuration is k-node-connected. If it is not, discard it, increase t by one and remove link l from candidate link set E. Then go back to Step 3.

5. Assign routes again only for those unicast requirements and multicast requirements whose routes pass through link l in the CB topology.

6. Calculate the total cost of all links. If the topology cost is improved (i.e. lower), accept this temporary topology as the CB topology. Then go back to Step 2. If it is not, discard the temporary configuration, increase t by one and remove link l from the candidate link set E. Then go back to Step 3.

7. Exit and return the CB topology.

In Step 3, the reason why we let $t_{\max}$ equal $\lceil N \cdot k/2 \rceil$ is that each CB topology which is k-node-connected has at least $\lceil N \cdot k/2 \rceil$ links.

2) Local optimization process

In this phase, the starting topology obtained in the first phase will be improved by exchanging two links iteratively.

Given two links, there are several possible cases of link exchange. If these two links are adjacent, that is, they have a common node, after exchanging these two links the configuration remains unchanged. If these two links are not adjacent, there are two possible exchange schemes. In more detail, given links $\{A,B\}$ and $\{C,D\}$ where node A, B, C and D are different from each other, we can exchange them to new links $\{A,C\}$ and $\{B,D\}$, or to new links $\{A,D\}$ and $\{B,C\}$. If one old link and one new link are same, we regard them as one link. Maybe one or both of these two exchange schemes will cause a new feasible topology with lower cost, or maybe neither of them will cause a new feasible topology with lower cost.

The main idea of this process is as follows. For the CB topology, select two candidate links to exchange. If a feasible topology with lower cost can be obtained by link exchange, accept this topology as the CB topology and continue to improve this new CB topology by link exchange. If no feasible topology with lower cost can be obtained by link exchange, continue to select another two candidate links to test. If finally all possible link pairs have been tried and no better topology can be obtained, terminate the algorithm and the CB topology is the final topology.

The order of link pairs for test in the CB topology is determined by the following rule. Assume there are l links in the CB topology. First, index these l links from 1 to l such that if $i < j$, the efficiency metric value of link i is larger than that of link j. For each link pair (link i, link j), define a metric $S = i + j$. Then sort all link pairs according to their values of metric S in ascending order. As for the order of those link pairs with the same metric value, sort them according to the smaller index in each link pair. For example, for link pairs (link 1, link 4) and (link 2, link 3), their values of metric S are both 5. The smaller index in (link 1, link 4) is 1 and the smaller index in (link 2, link 3) is 2. Thus (link 1, link 4) ranks ahead of (link 2, link 3). The order of link pairs is shown as follows: (link 1, link 2), (link 1, link 3), (link 1, link 4), (link 2, link 3), (link 1, link 5), (link 2, link 4), $\cdots$.

The flow chart of local optimization process is shown in Figure 2.2. It consists of following steps.

1. Set counter parameter t, which is used to count the continuous failure times, to zero. For the CB topology, obtain the link pair order according to the rule described above.

2. Check whether the value of t is larger than $t_{\max} = \binom{l}{2}$. If it is, go to Step 5.

3. Select the link pair (link i, link j) which has not been tested, according to the link pair order.

4. If link i and link j are adjacent, increase t by one and go back to Step 2. If link i and link j are not adjacent, there are two possible exchange schemes. Pick an arbitrary one and conduct following test first. If this exchange scheme cannot prompt a better topology, then select the other exchange scheme and also conduct following test.

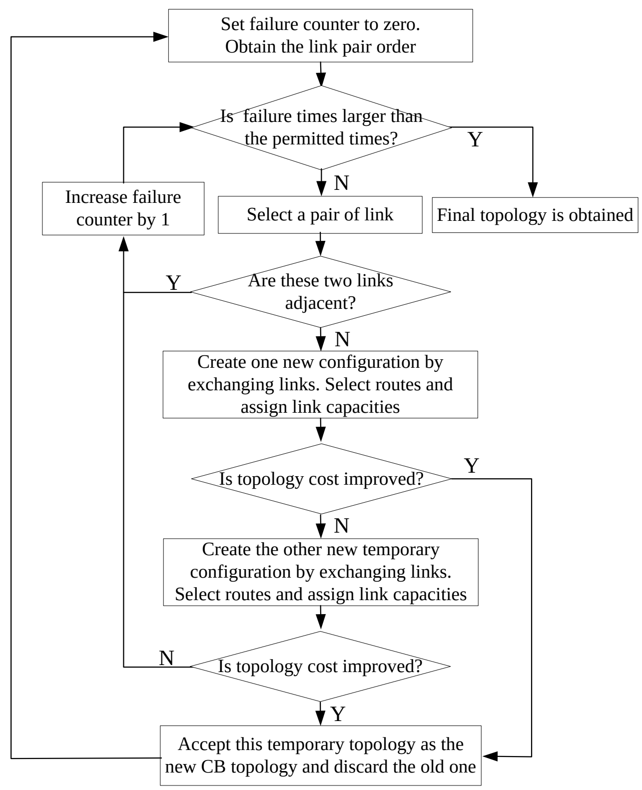


Figure 2.2: Flow chart of local optimization process.

Feasibility test: After link exchange, we get a new configuration. Determine if it is k-node-connected. If it is not, then this link exchange cannot induce a feasible topology. Otherwise, select the route for each requirement, allocate link capacities, and then calculate the total cost of this new topology. If this total cost is lower than that of the CB topology, discard the CB topology, regard this new topology as new CB topology and go back to Step 1.

If both two link exchange schemes cannot prompt a better topology, increase t by one and go back to Step 2.

5. Exit and return the CB topology.

Table 2.2: Running time of different operations

	Connectivity testing	**Unicast routing**	**Multicast routing**	**Capacity allocation**	**Cost calculation**
Complexity	$O(k^2N\|E\|)$	$O(N^3)$	$O(M\|E\|^3\|S\|^3)$	$O(N^2\|E\|)$	$O(K \cdot N^2)$

2.2.2 Link Addition and Exchange (LAE) Algorithm

This algorithm also consists of two phases, starting topology generation and local optimization process, and the second phase is the same as that of LDE algorithm. Hence here we only describe the first phase.

1) Starting topology generation

The main idea of this phase is that we first generate a k-node-connected configuration which has the potential to be a low-cost topology, and then build a topology based on this configuration.

This phase consists of following detailed steps.

1. Index N nodes from 1 to N randomly.

2. Determine the node with the smallest degree. Call this node X. If there are several candidate nodes, select the one with the smallest index. Determine the node with the smallest degree that is not already connected to X. Call this node Y. If there are several candidate nodes, select the one that is nearest to X. Add the link $\{X,Y\}$.

3. Repeat Step 2 until each node's degree is at least k.

4. Check whether current configuration is k-node-connected. If it is, go to Step 6.

5. Check whether the connectivity of current configuration can be increased (by one) by only adding one link. If it can be, add the shortest link whose addition can increase the connectivity. Otherwise discard current configuration and go back to Step 1.

   Repeat above operation until current configuration is k-node-connected or until the connectivity of current configuration cannot be increased by one by only adding one link.

6. Then select the route for each requirement and allocate link capacities.

7. Exit and return the CB topology.

In Step 5, if more than one link must be added to increase the connectivity, the rule is quite complex to determine which links are appropriate to add to guarantee the resulting topology has low cost [68].

2.2.3 Complexity Analysis

The running time for testing k-node connectivity is $O(k^2 N|E|)$ where E is the link set [53].

The complexity of routing for all unicast requirements is $O(N^3)$ [53]. There are M multicast requirements. For each one of them, the simplex method [2] is adopted to obtain the minimum-cost route. The expected complexity of the simplex method is $O(m^2 n)$ where m is the number of constraint equations and n is the number of variables in the linear programming problem [69]. Then the expected complexity to build a multicast route is $O(|E|^3|S|^3)$ where S is the multicast node set. Routing for M multicast requirements takes time $O(M|E|^3|S|^3)$.

According to Equation (2.11) and (2.12), it is easy to know that allocating capacities for $|E|$ links takes time $O(N^2|E|)$. According to Equation (2.1), the cost calculation of a topology takes time $O(K \cdot N^2)$.

Computational complexity of LDE algorithm

During the first phase, for each new temporary configuration, either only connectivity testing is done, or all operations listed in Table 2.2 are done. Among these operations, multicast routing is the most time-consuming one. In the worst case, for each CB topology with $|E|$ links, $|E|$ temporary configurations are all tested and until the $|E|$'th test a better topology is obtained. However, our simulation shows that at almost all iterations (other than the last several iterations) only after testing several temporary configurations a better topology can be obtained, far better than the worst case. Thus, it is more useful

[2] There exist polynomial algorithms for linear programming. Whereas the simplex method takes exponential time in the worst case, we adopt it because of its remarkable efficiency in practice.

to analyze the average-case complexity. The running time of the first phase is:

$$\begin{aligned} T_1 &= O(M(\frac{N^2-N}{2})^3|S|^3 + M(\frac{N^2-N}{2}-1)^3|S|^3 + \\ &\quad \cdots + M(\frac{kN}{2})^3|S|^3) \\ &= O(M|S|^3N^8). \end{aligned}$$

The topology obtained from the first phase has around $k \cdot N/2$ links, and thus has around $O(k^2N^2)$ different link pairs. During the second phase, the topology will be improved repeatedly. According to our simulation, the times of improving the CB topology is $O(N)$. The running time of the second phase is:

$$T_2 = O(M|E|^3|S|^3) \cdot O(k^2N^2) \cdot O(N) = O(k^5M|S|^3N^6).$$

Overall, the running time of LDE algorithm is $O(M|S|^3N^6(N^2 + k^5))$.

Computational complexity of LAE algorithm

In the first phase, it takes time $O(kN^3)$ to construct a configuration in which each node's degree is at least k, and according to our simulation experience we run Step 1 to Step 5 $O(N)$ times to get a k-node-connected configuration. In Step 6, routing and capacity allocation take time $O(Mk^3N^3|S|^3)$. Hence the overall running time of the first phase is $O(kN^4 + Mk^3N^3|S|^3)$, which is far lower than the running time of the second phase.

The overall running time of LAE algorithm is $O(k^5M|S|^3N^6)$.

One potential way for reducing the complexity is adopting a sub-optimal routing having low complexity, instead of the minimum-cost routing, to build routes for multicast requirements.

2.2.4 General Case of Link Capacity Assignment

If more than one line can be assigned to one link, the only difference between new algorithms and above algorithms is capacity assignment. New capacity assignment is to

Table 2.3: Available capacity options & costs

Capacity (Mbps)	**Variable cost** (unit cost/unit length)	**Fixed cost** (unit cost)
100	1	80
300	2	100
600	3.5	120
1000	5	160
1500	7	200

determine the quantity of each link type. Here we explore this problem in brief.

The capacity assignment problem of link $\{i, j\}$ can be formulated as follows:

$$\min \sum_{t=1}^{K} u_{i,j}^{t}(p_t + f_t/d_{i,j}) \text{ where } u_{i,j}^{t} = 0, 1, 2, \cdots ,$$

subject to:

$$\sum_{t=1}^{K} u_{i,j}^{t} C_t \geq f_{i,j}/e_{\max}.$$

This problem can be iteratively solved by dynamic programming methods [70].

2.3 Simulation Results

Table 2.4: Node locations of the network to be designed

Node	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
X	344	168	154	10	168	158	195	310	315	393	277	292	173	474	190	468
Y	224	139	262	41	287	130	127	196	42	104	193	173	228	239	199	179

In this section, first we will compare LDE algorithm with LAE algorithm and determine which one is better according to simulation results. Then we will evaluate the effectiveness of the better algorithm through comparing it with the exhaustive search method in small-size networks. Finally, the benefit brought by network coding technique in topology design is shown through comparing the coding-based design with the unicast-oriented design.

2.3.1 Simulation Parameter Settings

Information of available types of communication lines is shown in Table 2.3. The fixed costs are set to appropriate values so that in resulting topologies the total fixed cost accounts for around 25 percent of the total cost. Unit length costs of different types of lines follow the principle of scale economy. In our simulations $e_{\max}$ is set to 0.85, and unless otherwise mentioned, we consider designing 3-node-connected topologies, that is, k equals 3.

In practice, the amount of traffic from node i to j is different but generally not far different from the amount of traffic from node j to i [71]. Hence in our tests we set unicast rate in the following way. Unicast requirement rate $r_{i,j}(i < j)$ is selected uniformly in the interval $[r_{\min}, r_{\max}]$ (Mbps), and unicast requirement rate $r_{j,i}$ is selected uniformly in the interval $[0.6r_{i,j}, 1.4r_{i,j}]$ (Mbps).

In a network with N nodes, there are totally $N(C_{N-1}^{2} + \cdots + C_{N-1}^{N-1})$ possible multicast requirements. However, it is not difficult to imagine that in practice most of them are with low rates. It is unpractical and not quite necessary to consider all multicast requirements specially. It is practical that at the stage of traffic requirement estimation only those multicast requirements with moderate or high rates are considered separately and the traffic of low-rate multicast requirements is considered as unicast traffic. In our tests, there are $3N$ multicast requirements and the number of sinks of each multicast is selected uniformly in the integer interval $[2, N-1]$. Each multicast requirement rate is selected uniformly in the interval $[R_{\min}, R_{\max}]$ (Mbps). The parameters $r_{\min}$, $r_{\max}$, $R_{\min}$ and $R_{\max}$ are used to adjust unicast traffic amount and multicast traffic amount.

2.3.2 Comparison of Two Heuristic Algorithms

Topology cost resulted from an algorithm depends on input parameter values and the performance of this algorithm. *Workload* (i.e. the total amount of traffic originating from all nodes) and the ratio of multicast traffic amount[3] to the total network traffic amount,

[3]The traffic amount of a multicast transmission with transmission rate R and t receivers is considered as $R \cdot t$.

somewhat vaguely called *traffic ratio*, are two important input parameters closely related to topology cost. The larger the workload is, the higher the resulting topology cost is. Given a workload, the larger the traffic ratio is, the lower the resulting topology cost is, if the topology design algorithm takes advantage of multicast characteristic.

To evaluate the performance of our algorithms, we consider a set of 16 nodes whose positions are randomly selected in scale 500×300 (unit distance). Table 2.4 shows the node locations represented by the set of Cartesian coordinates X and Y. Based on these 16 nodes, we investigate the performance of two proposed algorithms under different workloads and different traffic ratios.

If for any two nodes i and j, the amount of traffic from i to j equals that from j to i, we say the traffic is symmetric; otherwise, the traffic is unsymmetric. Let us illustrate the effect of the symmetry of traffic on the topology cost through an example about the traffic in a communication line. In one case, 70 Mbps traffic is transmitted in one direction and 70 Mbps traffic is in the other direction. In another case, 20 Mbps traffic is transmitted in one direction and 120 Mbps traffic is in the other direction. Although in both cases the total loads in this line are equal, capacity 100 Mbps is enough for it in the first case and capacity 300 Mbps is needed for it in the second case. Thus, if the traffic in the network is highly asymmetric, the cost of the resulting topology is higher than that resulted from the same amount of relatively symmetric traffic.

First we investigate the performances of two algorithms under different workloads with traffic ratio 40%. For each workload, we obtain the average topology cost of a number of cases with different spatial distribution of traffic among 16 nodes. Figure 2.3 shows the average topology costs under different workloads of LDE and LAE algorithms. For each algorithm, the average topology cost increases approximately linearly with increasing workload. This is very explicit, since more traffic will consume more capacity in the resulting topology. In addition, the principle of scale economy about line cost is demonstrated here. Take LAE algorithm as an example. When the workload increase from 3000 to 7000 Mbps, the average topology cost only increases to around 1.5 times.

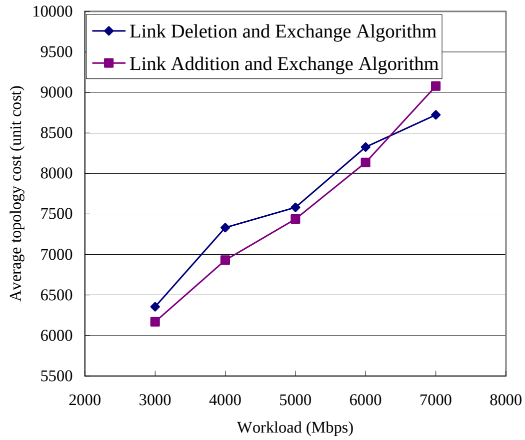


Figure 2.3: Average topology costs versus workloads with traffic ratio=40%.

Comparison under different workloads

It is interesting to note from the Figure 2.3 that when the workload is not very high (e.g. below 6500 Mbps), LAE algorithm always perform better than LDE algorithm. When the workload is high (e.g., above 7000 Mbps), however, LDE algorithm actually outperforms LAE algorithm. This topology cost crossover observation in Figure 2.3 is actually due to a similar crossover in the network-wide average link (NAL) utilization of two algorithms. As the workload increases from 3000 to 7000 Mbps, the NAL utilization of LDE algorithm increases from 50.6% to 61.5%, while the NAL utilization of LAE algorithm grows from 56.3% to 60.7%[4]. It is notable that the topology cost is heavily related to the NAL utilization, since a low NAL utilization usually results in a high cost topology. The NAL utilization crossover of two algorithms can be explained by their difference in the number of links of the final topology designs. The number of network links resulted from LDE algorithm are mainly distributed in the interval [26, 30], while the number of network links resulted from LAE algorithm are usually 24 or 25 [5]. When the workload is low (e.g. 3000 Mbps), links in LDE-based topology designs usually carry less amount of traffic and thus

[4]The reason the NAL utilization increases as the workload increases is that, when the workload is low many arcs carry a small amount of traffic and are underutilized (note that the smallest capacity can be allocated is 100 Mbps), but as the workload increases the traffic amount over each arc will grow and consequently the NAL utilization will increase for both LDE and LAE algorithm. In addition, the NAL utilization is not very high here is due to the unsymmetric traffic distribution and the imposed constraint on link utilization.

[5]The link number of the resulted topology depends on the link deletion process in LDE algorithm, and on the the link addition process in LAE algorithm.

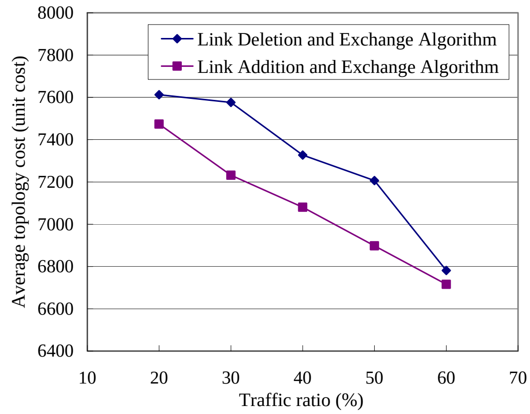


Figure 2.4: Average topology costs versus traffic ratios with a moderate workload.

have lower link utilizations than those in the LAE-based topology designs (note that the smallest capacity can be allocated is 100 Mbps), because the topologies obtained from the LDE algorithm usually have more links than those from the LAE algorithm to support the same workload. When the workload is high, however, we can actually benefit from the topologies that have more links. For a given multicast connection, the coding-based minimum-cost route generally consumes lower bandwidth and has a better load-balance capability in topologies with more links. A more uniform distribution of multicast traffic can actually relieve the negative effect caused by the traffic unsymmetry and thus increase the NAL utilization.

Comparison under different traffic ratios

Now we investigate the performances of two algorithms under the same workload and different traffic ratios. Note that it is often not the practical case that the workload of a network to design is very high, thus the evaluation is performed under a moderate workload. For each traffic ratio, we obtain the average topology cost of a number of cases with different spatial distribution of traffic among 16 nodes. Figure 2.4 shows the average topology costs of different ratios. For each algorithm, the average cost approximately linearly decreases with the increase of traffic ratio. It is easy to understand such tendency, since a certain amount of multicast traffic will consume less resource than that consumed by the same amount of unicast traffic. So for a given workload, the higher percentage multicast traffic accounts for, the less total capacity the resulting topology needs.

From Figure 2.4 we can see that the average cost does not decrease fast with the increase of traffic ratio, partially because of the unsymmetric traffic pattern we used for test. Because the overall computational burden of all simulations is heavy, as mentioned previously, there are only $3N$ multicast requirements in the topology design simulations we conducted. However, we conjecture that in practice there are at least $O(N^2)$ multicast requirements with moderate or high rates, and multicast traffic is relatively uniformly distributed among N nodes. If this is true, the average cost will decrease with increasing traffic ratio at a faster rate than that shown in Figure 2.4.

Compared with LAE algorithm, the average cost of LDE algorithm increases with 1.8%, 4.8%, 3.5%, 4.5% and 1.0% corresponding to traffic ratio 20%, 30%, 40%, 50% and 60%, respectively.

According to above comparison results, we conclude that on the whole LAE algorithm performs slightly better than LDE algorithm. Only the performance of LAE algorithm will be evaluated below.

2.3.3 Performance Evaluation

The performance of a topology design algorithm can be evaluated through comparison with available good algorithms on the same problem, or by gauging the gap between the topology cost gotten by this algorithm and lower bound on the cost of the optimal topology [56]. Regretfully, there is no available heuristic algorithms used to design NCM network topologies, and lower bounds are only known for simple cases even for the unicast-oriented topology design problem[70], not to mention NCM network topology design problem. The approach we take is comparing LAE algorithm with the exhaustive search method.

However, it is impossible to obtain the optimal topology by the exhaustive search method even for 5-node cases. Here we briefly deal with the complexity of the exhaustive search method for 5-node cases. For 5-node cases if there are 5 types of lines available, there are $6^{N(N-1)/2} \approx 6.0 \times 10^7$ possible topologies to be test, and for each topology

Table 2.5: Comparison between LAE algorithm and the exhaustive search method

	Case 1	Case 2	Case 3	Case 4	Case 5
Topology cost (LAE algorithm)	2035.53	2034.46	1788.07	1614.37	1888.22
Topology cost (ES method)	1899.28	1819.78	1727.67	1442.91	1635.29
Cost gap	7.17%	11.80%	3.50%	11.88%	15.47%

we should confirm whether it is k-node-connected and whether it is with lower cost. If these two conditions are both satisfied, then we should try all possible flow assignments to confirm whether all requirements can be accommodated simultaneously in this topology. Since the flow from the source to the determination can be split (or divided) and transmitted over multiple paths simultaneously, there are a large number of possible flow assignments.

Hence we use five 4-node cases with different parameter values for test and the objective is to obtain 2-node-connected low-cost topologies. We make a reasonable assumption that when the traffic from one node to another is split and transmitted over L paths, the traffic amount on each path should be the times of a basic traffic amount, not arbitrary amount.

As is shown in Table 2.5, the LAE algorithm performs almost as good as the exhaustive search method in 4-node cases. In each case the difference between the solution cost of link addition algorithm and the optimal solution cost is typically less than 16 percent. This degree of accuracy is deemed adequate for most topology designs, especially considering that traffic requirements cannot be predicted with much accuracy before network implementation, or tend to change during the life of the network. Therefore, we conclude that link addition algorithm is very effective in designing network coding-base multicast networks.

2.3.4 Benefit of Network Coding

When design the topology of a NCM network, how much can we gain in terms of topology cost by separating multicast requirements from unicast requirements and taking advantage of the characteristic of multicast in topology design algorithms? Furthermore, how much can we gain further if network coding technique is used to support multicast transmissions?

To answer the above question, we investigate the topology cost difference between the following three cases. In the first case, each multicast requirement is treat as multiple unicast requirements. In the second case, multicast requirements are considered separately from unicast requirements and Steiner tree algorithm are used to build multicast routes. In the third case, multicast requirements are considered separately and network coding-based minimum-cost multicast algorithm is used to build multicast routes.

For the first case conventional unicast-oriented algorithms can be used to design topologies. Unfortunately, no well-known conventional algorithm available deals with the exactly same design problem as ours[6]. One good algorithm used for almost the same design problem with ours is well-known MENTOR algorithm. The difference is that the problem this algorithm deals with does not include reliability requirement, whereas the problem we consider includes it. In addition, as far as we know no well-known algorithm is available for the second case.

LAE algorithm can be used to design topologies for the first case, like conventional algorithms, by removing the routing procedure for multicast requirements and transforming multicast requirements to unicast requirements. For simplicity, call this revised algorithm unicast-oriented link addition and exchange (ULAE) algorithm. LAE algorithm can also be used to design topologies for the second case, by using Steiner tree algorithms to obtain multicast routes, instead of using network coding-based minimum-cost multicast algorithm. Call this revised algorithm Steiner tree-based link addition and exchange (SLAE) algorithm. In our test, we use the DST (Directed Steiner Tree) approximation algorithm described in [72] to build Steiner trees in SLAE algorithm. In addition, temporarily call the original LAE algorithm, i.e. the network coding-based one, network coding-based link addition and exchange (CLAE) algorithm.

[6]Topology design problems include a lot of assumptions and requirements. Few well-known algorithms were proposed for an exactly same design problem. For example, some consider the case that there is only one type of line, and others consider the case that several types of lines are available. Some consider reliability requirement, and others not.

Table 2.6: Comparison between MENTOR and ULAE algorithms

	Case 1	Case 2	Case 3	Case 4	Case 5	Case 6	Case 7	Case 8	Case 9
Topology cost(MENTOR)	10359.9	9203.7	10030.3	8480.2	9312.9	8064.6	8275.8	9785.7	6957.7
Topology cost(ULAE)	10505.4	9677.3	9925.9	8468.9	9469.2	7952.4	8283.8	9589.6	6821.8
Cost ratio	0.986	0.951	1.011	1.001	0.983	1.014	0.999	1.020	1.020

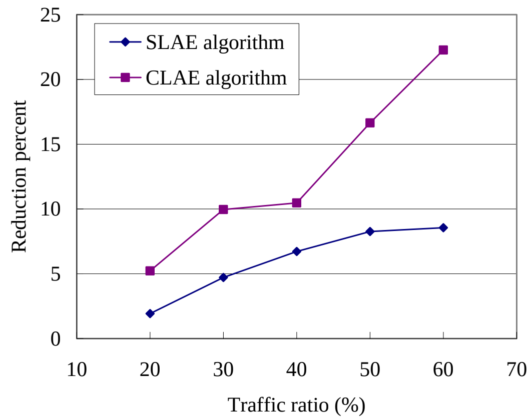


Figure 2.5: Percent reduction in terms of the average topology cost of SLAE algorithm and CLAE algorithm, using the average topology cost of ULAE algorithm as the base.

Comparison between MENTOR and ULAE algorithms

Extensive simulations show that, for those cases that 3-node-connected topologies are obtained by MENTOR algorithm, the average cost of topologies gotten by ULAE algorithm is only 0.28% higher than that of topologies gotten by MENTOR algorithm[7]. Table 2.6 shows some comparison results between MENTOR and ULAE algorithms.

Based on this observation, we can use ULAE algorithm, SLAE algorithm and CLAE algorithm to investigate the rough gain in terms of topology cost obtained by considering multicast traffic specially, and the gain obtained further by using network coding technique to support multicast.

Comparison between ULAE, SLAE and CLAE algorithms

Figure 2.5 shows the percent reduction in terms of the average topology cost of SLAE algorithm and CLAE algorithm, using the average topology cost of ULAE algorithm as the base. For SLAE algorithm, the percent reduction increases slowly with the increase of

[7]MENTOR algorithm has lower complexity, compared to ULAE algorithm which is not specially proposed for unicast-oriented topology design.

traffic ratio. Nevertheless, for CLAE algorithm, the percent reduction increases rapidly with the increase of traffic ratio. Take traffic ratio 50% as an example. If network coding technique is used to support multicast transmissions, the average topology cost can reduce with 16.6%, far higher than 8.3% corresponding to the Steiner tree-based algorithm. It can be seen from Figure 2.5 that network coding can offer much benefit in designing topologies, especially when the amount of multicast traffic accounts for a large percentage of the total traffic. We conclude that, when we design multicast network topologies, it is necessary and beneficial to consider multicast traffic specially rather than treat each multicast as multiple unicasts, and if technique is adopted topology cost can be greatly reduced.

2.4 Summary

In this chapter we studied the challenging topology design problem of network coding-based multicast networks. Based on the characteristics of multicast and network coding, we formulated this problem as an NP-hard mixed-integer nonlinear programming problem, which is much more complicated than the conventional unicast-oriented topology design problems. Two heuristic algorithms, link deletion and exchange (LDE) algorithm and link addition and exchange (LAE) algorithm, are proposed for our design problem. Extensive comparisons indicated that overall the LAE algorithm performs better than LDE algorithm, and LAE algorithm is effective to design the topologies of network coding-based multicast networks.

Our results in this chapter show that in comparison with the conventional unicast-oriented design for multicast networks, the Steiner tree-based design has moderate improvement in term of topology cost, but the network coding-based design can make this improvement very significant. For example, for the 16-node topology design problem examined in this chapter, the Steiner-tree based design can reduce the topology cost by about 8.3 percent than the conventional unicast-oriented design when the multicast traffic accounts for 50% of the total traffic, but our network coding-based design can make this reduction in topology cost as high as 16.6 percent.

Chapter 3

Packet Coding in Multihop Wireless Networks

Multihop wireless networks have been an active area of research for many years. In such type of networks, there is no network infrastructure or centralized administration, and each mobile node operates not only as a host but also as a router, forwarding packets for other mobile nodes. Promising applications of such type of networks include wireless sensor networks, wireless mesh networks, etc. One of the most significant problems of multihop wireless networks is that their current implementations suffer from a severe throughput limitation and do not scale well as the number of network nodes increases [73, 74, 75].

Network coding is a promising technique to improve the throughput of wireless networks. The basic idea of network coding in wireless networks is quite simple and can be illustrated using the scenario in Figure 3.1 (from Wu *et al.* [43]), where node A wants to send packet P_1 to node B and node B wants to send packet P_2 to node A with the

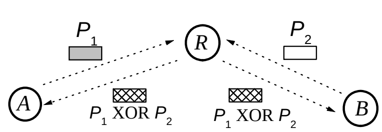


Figure 3.1: A simple scenario of wirelesses network coding.

help of intermediate node R. Assume node R has received P_1 and P_2. In traditional transmission way, node R transmits P_1 and P_2 separately. However, node R can XOR P_1 and P_2 together and broadcast $P_1 \oplus P_2$. Upon receiving $P_1 \oplus P_2$, node A can decode P_2 by $P_2 = P_1 \oplus (P_1 \oplus P_2)$. Similarly, node B can decode P_1 by $P_1 = P_2 \oplus (P_1 \oplus P_2)$. Therefore, with the network coding function, node R can forward two packets in a single packet transmission and its transmission efficiency is improved by 100% when P_1 and P_2 have the same size.

Following the study of the above basic scenario by Wu *et al.* [43], recently, Katti *et al.* proposed the first practical network coding-based packet forwarding architecture (called COPE) to essentially improve the network throughput of multihop wireless networks [35]. In COPE, each node can opportunistically overhear and store those native packets transmitted by its neighbors, which are not addressed to itself. Each node can intelligently encode (XOR) multiple packets destined to different nexthops such that multiple packets can be forwarded in a single transmission, resulting in a significant bandwidth saving. Since the proposal of promising COPE architecture, some efforts have been made to theoretically evaluate the performance of COPE-type wireless network coding [37, 38, 40]. Liu *et al.* [38] presented upper bounds on the throughput benefit ratio[1]. Sengupta *et al.* [37] presented a theoretical framework for computing the maximal throughput of a COPE-type network with fixed network topology and static traffic demands. In addition, it has been shown that the optimal coding problem in COPE is NP-complete [76].

In the current COPE architecture, a network node maintains one dedicated FIFO queue for packets to be forwarded. In addition to the FIFO queue, the node also maintains one large-size virtual queue and one small-size virtual queue for all packets destined to the same neighbor. When making coding decision, COPE first dequeues the head packet of the FIFO queue, and then check only the head packet of each virtual queue one by one to determine if the packet can be encoded with the head packet of FIFO queue. The above virtual queue structure is quite simple and introduces very limited packet reordering.

[1] The ratio of the throughput when using the optimal COPE-type network coding scheme to the throughput when using the optimal non-coding scheme

It is notable that theoretically all the packets of distinct flows have the potential to be encoded with the head packet of FIFO queue for throughput improvement. However, the above packet size-oriented virtual queue structure significantly limits this potential coding opportunity[2], since among packets destined to the same neighbor at most two packets (the head packets of small-size and large-size queues) will be examined in the coding process, regardless of the number of flows. To address the above problem, in this chapter we propose a flow-oriented virtual queue structure for the COPE architecture, where a dedicated virtual queue is maintained for each flow and all head packets of virtual queues are regarded as candidates for encoding with the head packet of FIFO queue. Since this new virtual queue structure ensures that one packet from each flow is considered in the coding process, the potential coding opportunities will be dramatically increased. For the proposed virtual queue structure, we further study the optimal packet coding problem (i.e., finding the optimal coding solution) and also present a very efficient coding algorithm for it.

In summary, the main contributions of this chapter are as follows:

1. We propose a flow-oriented virtual queue structure that can dramatically increase the coding opportunities and also can completely eliminate the packet reordering.

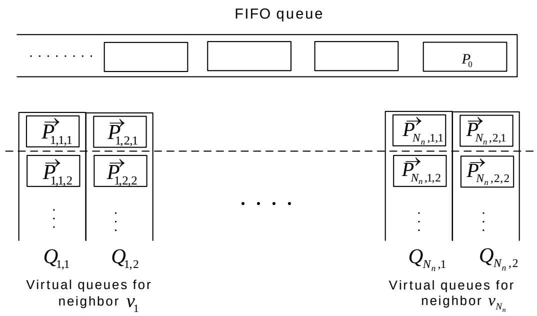
2. We formulate the corresponding optimal packet coding problem as an optimization problem and prove its NP-completeness.

3. We present an efficient coding algorithm for finding good coding solutions.

4. We demonstrate that although the available simple COPE architecture can essentially improve the node throughput, this improvement can be much more significant if our proposed new virtual queue structure and coding algorithm are jointly adopted.

The rest of this chapter is organized as follows. In Section 3.1, we briefly review the available COPE architecture. Section 3.2 presents the new virtual queue structure.

[2] By coding opportunity we mean that two or more packets can be encoded together and each nexthop of this encoded packet can decode its native packet with probability larger than a given value.



(a) Queues inside a network node.

	P_0	P_1	P_2	...
v_1	$\theta_{1,0}$	$\theta_{1,1}$	$\theta_{1,2}$	...
v_2	$\theta_{2,0}$	$\theta_{2,1}$	$\theta_{2,2}$	...
v_3	$\theta_{3,0}$	$\theta_{3,1}$	$\theta_{3,2}$	...
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
v_{N_n}	$\theta_{N_n,0}$	$\theta_{N_n,1}$	$\theta_{N_n,2}$	...

(b) Table of packet possession indicators.

Figure 3.2: The data inside a COPE-based network node with N_n neighbors.

In Section 3.3, we provide the formulation of the optimal packet coding problem, prove its NP-completeness, and then propose an efficient packet coding algorithm. Numerical results are presented in Section 3.4. Finally, section 3.5 concludes this chapter.

3.1 Overview of COPE

3.1.1 COPE Architecture

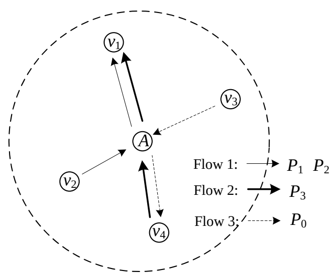
The COPE architecture virtually inserts a coding layer between the IP and MAC layers, which identifies coding opportunities to XOR multiple packets together and forwards them in a single packet transmission [35]. In a COPE-based network, a node maintains one FIFO queue (called output queue) and also maintains for each neighbor v_i two virtual queues $Q_{i,1}$ and $Q_{i,2}$ (one for small packets whose sizes are smaller than 100 bytes and another for large packets), as illustrated in Figure 3.2(a). In addition to these queues, each node also maintains a table, whose entry $\theta_{m,n}$ indicates the probability that neighbor v_m possesses packet P_n at the current time, as illustrated in Figure 3.2(b). We refer to the probability $\theta_{m,n}$ as *packet possession indicator* in this chapter.

The COPE works as follows. Each node always snoops on all communications over the wireless medium. On one hand, when a node overhears a packet being delivered to another node, it will store the overheard packet in its memory for a limited period (say 0.5s). On the other hand, when a node successfully receives a native packet or retrieves a native packet from an encoded packet delivered to it, if it is the ultimate destination of this native packet, it delivers the packet to the higher layers of the network stack; otherwise, it first adds this native packet to the output queue, then adds a pointer (pointing to this packet in the output queue) to the appropriate virtual queue based on the packet's nexthop and size, and finally updates the hash table by including the probabilities that its neighbors possess this native packet. In addition to overhearing and receiving packets, each node also broadcasts *reception reports* to inform its neighbors the packets it possesses by annotating the data packets or by special control packets. Due to different reasons like packet loss and severe congestion, a node cannot solely rely on reception reports to decide which packets its neighbors possess and thus it may need to estimate the probability that a neighbor possesses a particular packet. If a node learns from reception reports that neighbor v_m possesses packet P_n, then $\theta_{m,n} = 1$. Otherwise, it will estimate the value of $\theta_{m,n}$.

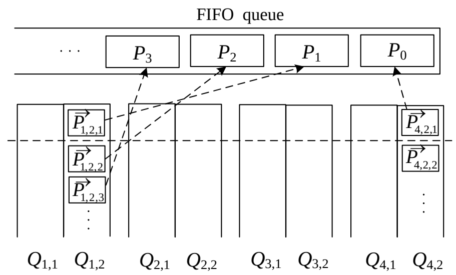
The *packet coding algorithm* inside COPE makes coding decision in the following way. The COPE first dequeues the packet P_0 at the head of the output queue, and then checks one by one the head packets of virtual queues with the same packet size as P_0 to find appropriate packets to encode with P_0. After exhausting the head packets of the same size as P_0, COPE then checks one by one the head packets of virtual queues of another size. The following rule is adopted to determine if a packet P_{i_n} is feasible to further encode with the currently encoded packet. Suppose we have already decided to XOR n packets $P_0 \oplus P_{i_1} \oplus \cdots \oplus P_{i_{n-1}}$ together, and are considering XOR-ing the $(n+1)$'th packet P_{i_n} with them. The packet coding $P_0 \oplus P_{i_1} \oplus \cdots \oplus P_{i_n}$ is *feasible* only if the following constraint, namely *probability threshold* (PT) constraint, is satisfied: each nexthop to whom a packet $P_i \in \{P_0, P_{i_1}, \ldots, P_{i_n}\}$ is headed can decode its packet P_i with the probability greater than a threshold G (the default value of G is set as 0.8 in COPE [35]).

3.1.2 Limitations of Available Virtual Queue Structure

The current virtual queue structure of COPE is quite easy to maintain. However, it has the following two limitations. First, although COPE introduces very limited packet reordering



(a) Flows passing through node A.



(b) FIFO queue and virtual queues inside node A.

Figure 3.3: Limitation illustration of the current virtual queue structure.

by checking only the head packets of virtual queues, it cannot completely eliminate packet reordering[35]. Second, and more importantly, we should notice that theoretically all the oldest packets of distinct flows[3] have the potential to be encoded with the head packet of FIFO queue for throughput improvement. However, the current structure cannot fully explore this potential, because among those packets to the same neighbor at most two packets (the heads of two virtual queues) can be the candidate packets for encoding with P_0. More specifically, when more than one flow with small packets or large packets are routed to the same nexthop, only one oldest packet can locate at the head of the virtual queue (i.e., serves as the candidate packet). Therefore, this structure will significantly limit the potential coding opportunities.

To illustrate the limitation of current packet size-based queue structure, we consider a simple example shown in Figure 3.3. In this example, Flows 1 and 2 with large-size packets are passing through node A and going to neighbor v_1, while Flow 3 also with large-size packets is passing through node A and going to neighbor v_4. Then, all packets of Flow 1 and 2 will be queued in the same virtual queue $Q_{1,2}$, as shown in Figure 3.3(b). Suppose the coding $P_0 \oplus P_1$ is infeasible and the coding $P_0 \oplus P_3$ is feasible. During the search for a feasible coding solution, however, the node A will only check the feasibility of coding $P_0 \oplus P_1$, without the consideration of $P_0 \oplus P_3$. Finally, P_0 will be transmitted alone, resulting in the loss of coding opportunity $P_0 \oplus P_3$.

To address the above problem, we propose here a flow-oriented virtual queue structure,

[3]Inside a node, the oldest packet of a flow is the firstly arrived packet among all the stored packets of this flow.

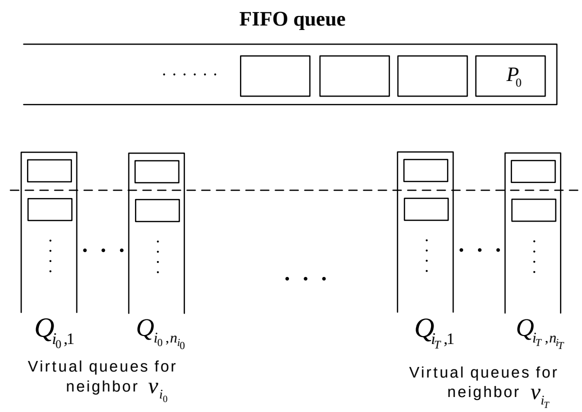


Figure 3.4: Flow-oriented virtual queue structure.

as discussed in the next section.

3.2 Flow-Oriented Virtual Queue Structure

In this section, we first introduce the new queue structure and then examine the candidate packets increment from using it.

3.2.1 Flow-Oriented Virtual Queue Structure

The basic idea of our virtual queue structure is to maintain a dedicated virtual queue for each flow such that packets of distinct flows have the chance to encode with the head packet of FIFO queue. We call this new virtual queue structure *flow-oriented virtual queue structure*. The maintenance of virtual queues is now flow-oriented rather than packet size-oriented. As shown in Figure 3.4, instead of maintaining a fixed number of virtual queues for each neighbor, a network node now dynamically allocate virtual queues to each neighbor v_i, depending on the number of active flows from this node to neighbor v_i. If there does not exist any flow passing from node A to a neighbor at the current time, node A does not maintain any virtual queues for it. When a new flow whose route includes link (A, v_j) is initiated in the network, the node A then allocates a new virtual queue for neighbor v_j to store the packets of this new flow. On the contrary, when a flow passing through node A is terminated, node A will release the allocated virtual queue for this flow.

Like the available coding scheme, with the consideration of packet reordering, only the packets at the head of virtual queues are regarded as the candidates for coding with P_0. In this way, this new queue structure does not introduce any packet reordering[4]. With this virtual queue structure, the oldest packet of each flow now has the chance to encode with P_0, so it significantly increases the candidate packets for coding. Actually, this queue structure provides the maximum number of candidate packets under the condition that no packet reordering is allowed.

Let us still consider the example in Figure 3.3. With the proposed queue structure, P_3 will be at the head of the virtual queue maintained for Flow 2 and thus the coding algorithm can find the feasible coding solution $P_0 \oplus P_3$. This example indicates that the new queue structure can increase candidate packets and consequently has the potential to increase the coding opportunities.

3.2.2 Candidate Packets Increment

To have a solid understanding on how the number of candidate packets can be increased by using the flow-oriented queue structure, for a node with i neighbors and k active flows of all large (or all small) packets[5], we investigate here the ratio $\delta_{i,k}$ of the number of virtual queues of the flow-oriented queueing structure (i.e., k) to the expected number of virtual queues of the available packet size-oriented structure. The $\delta_{i,k}$ reflects the increment of virtual queues (and thus the increment of coding opportunities).

For a given node, we call one of its neighbors as its *downstream neighbor* if there exists at least one flow from this node to this neighbor. Then, the problem of calculating the expected number of virtual queues of the available structure is reduced to the calculation of its conditional expected number of downstream neighbors $E(N_d|N_n = i, N_f = k)$, where

[4] Packet reordering happens only the arrival order of packets of a flow is different from their departure (transmission) order.

[5] Since in the available structure large packets and small packets are separately queued in the virtual queues, we need to separately consider the number of virtual queues maintained for large packets and the number of virtual queues maintained for small packets.

N_d, N_n and N_f are the numbers of downstream neighbors, neighbors and active flows of a node, respectively. Thus, $\delta_{i,k}$ is evaluated as

$$\delta_{i,k} = k/E(N_d|N_n = i, N_f = k). \tag{3.1}$$

To evaluate $E(N_d|N_n = i, N_f = k)$, we first establish the following lemma.

Lemma 2. *Let $f(m,n)$ be the number of ways of distributing m distinct objects among n $(1 \leq n \leq m)$ distinct boxes such that each box has at least one object. Then $f(m,n)$ is given by:*

$$f(m,n) = \sum_{l=0}^{n-1} (-1)^l \binom{n}{l} (n-l)^m, \tag{3.2}$$

where $\binom{n}{l}$ is the binomial coefficient $\frac{n!}{l!(n-l)!}$.

Proof. Clearly, there are n^m different distribution ways of distributing m distinct objects into n distinct boxes.

Number n boxes from 1 to n. Let $A_i (i = 1, 2, \ldots, n)$ be the set of all ways for distributing m distinct objects into boxes $\{1, 2, \ldots, n\}/\{i\}$. Then $f(m,n)$ can be expressed as $f(m,n) = n^m - |A_1 \cup A_2 \cup \cdots \cup A_n|$. According to the inclusion-exclusion principle, we have

$$\begin{aligned}
&|A_1 \cup A_2 \cup \cdots \cup A_n| \\
&= (|A_1| + |A_2| + \cdots + |A_n|) \\
&\quad + (-1)^1 (|A_1 \cap A_2| + |A_1 \cap A_3| + \cdots + |A_{n-1} \cap A_n|) \\
&\quad + (-1)^2 (|A_1 \cap A_2 \cap A_3| + |A_1 \cap A_2 \cap A_4| + \cdots \\
&\quad + |A_{n-2} \cap A_{n-1} \cap A_n|) + \cdots \\
&\quad + (-1)^{n-1} |A_1 \cap A_2 \cap \cdots \cap A_n| \\
&= (-1)^0 \binom{n}{1} (n-1)^m + (-1)^1 \binom{n}{2} (n-2)^m \\
&\quad + \cdots + (-1)^{n-1} \binom{n}{n} (0)^m
\end{aligned}$$

Then $f(m,n) = \sum_{l=0}^{n-1}(-1)^l\binom{n}{l}(n-l)^m$. $\square$

Based on the above lemma, we can establish the following theorem for $E(N_d|N_n = i, N_f = k)$.

Theorem 2. *For a node with i ($i \geq 1$) neighbors and k ($k \geq 1$) active flows, the expected number of downstream neighbors is given by*

$$\begin{aligned} &E(N_d|N_n = i, N_f = k) \\ &= \sum_{j=1}^{\min\{i,k\}} \left(j \frac{\binom{i}{j}}{i^k} \left[\sum_{l=0}^{j-1} (-1)^l \binom{j}{l} (j-l)^k \right] \right). \end{aligned} \tag{3.3}$$

Proof. Totally, there are i^k different ways to distribute k flows among i neighbors. Since each flow has the same probability of passing from the current node to each neighbor, all the ways of flow distribution will happen with the same possibility.

Based on Lemma (2), we know that the number of ways for distributing k flows among j ($1 \leq j \leq \min\{i,k\}$) downstream neighbors will be

$$n_k = \binom{i}{j} f(k,j) = \binom{i}{j} \left[\sum_{l=0}^{j-1} (-1)^l \binom{j}{l} (j-l)^k \right].$$

Thus, the probability of having j downstream neighbors is n_k/i^k, given by

$$\begin{aligned} &P(N_d = j|N_n = i, N_f = k) \\ &= \frac{\binom{i}{j}}{i^k} \left[\sum_{l=0}^{j-1} (-1)^l \binom{j}{l} (j-l)^k \right], 1 \leq j \leq \min\{i,k\}. \end{aligned} \tag{3.4}$$

Then, according to

$$\begin{aligned} &E(N_d|N_n = i, N_f = k) \\ &= \sum_{j=1}^{\min\{i,k\}} j \cdot P(N_d = j|N_n = i, N_f = k), \end{aligned} \tag{3.5}$$

we get the result. $\square$

Table 3.1: Ratio of the number of virtual queues in the proposed structure to the expected number of virtual queues in the available structure

	$N_n = 2$	3	4	5	6	7
$N_f = 2$	**1.33**	1.20	1.14	1.11	1.09	1.08
3	1.71	**1.42**	1.30	1.23	1.19	1.16
4	2.13	1.66	**1.46**	1.36	1.29	1.24
5	2.58	1.92	1.64	**1.49**	1.39	1.33
6	3.05	2.19	1.82	1.63	**1.50**	1.42
7	3.53	2.48	2.02	1.77	1.62	**1.51**

Table 3.2: Notations employed to describe the optimal packet coding problem

Notation	**Meaning**
v_i	downstream neighbor i.
P_0	head packet of the output queue. Its nexthop is v_0.
R_0	packet set $\{P_0\}$.
T	number of downstream neighbors except v_0.
p_i^r	probability that a packet transmitted by the current node can be successfully received by v_i.
p_i^d	the probability that the encoded packet can be decoded by v_i.
n_i	number of non-empty virtual queues maintained for flows from the current node to downstream neighbor v_i. $n_i \geq 1$.
$P_{i,j}^r$	head packet of the j-th virtual queue maintained for v_i. $(1 \leq i \leq T,\ 1 \leq j \leq n_i)$
R_i	packet set $\{P_{i,1}^r, \cdots, P_{i,n_i}^r\}$. $(1 \leq i \leq T)$
R	candidate packet set $\bigcup_{i=0}^{T} R_i = \{P_0, P_1, \cdots, P_K\}$. $K = n_1 + \cdots + n_T$.
l_i	size of packet P_i.
S_i	set of packets that v_i possesses with probability greater than G. $(0 \leq i \leq T)$
N_i	number of packets in S_i.
$P_{i,j}^s$	j-th packet in S_i. $(0 \leq j \leq N_i)$
$g(P_{i,j}^r) = n$	the mapping function from packet $P_{i,j}^r$ to its ID n in R.
$\mathbf{e}_i$	the i-th unit vector of dimension $K+1$. $(1 \leq i \leq K+1)$

Based on the above theorem and Equation (3.1), we have

$$\delta_{i,k} = \frac{k}{\sum_{j=1}^{\min\{i,k\}} \left(j \frac{\binom{i}{j}}{i^k} \left[\sum_{l=0}^{j-1} (-1)^l \binom{j}{l} (j-l)^k \right] \right)}. \tag{3.6}$$

Numerical results from the above equation are shown in Table 3.1. We can observe that when the number of flows is larger than the number of neighbors, $\delta_{i,k}$ will be large, i.e., the virtual queues in the proposed queue structure is, on average, much more than those in the available structure. Only at those nodes where the number of flows is far smaller than the number of neighbors, $\delta_{i,k}$ is approximately equal to one. Therefore, the proposed virtual queue structure can greatly increase the number of candidate packets and thus the coding opportunities.

3.3 Optimal Packet Coding

In this section, we proceed to study the optimal packet coding (OPC) problem for the flow-oriented virtual queueing structure, i.e., to find the coding solution with the largest coding gain. We first define the packet coding gain for quantitatively measuring the "goodness" of a coding solution. We then formulate the optimal packet coding problem and then prove its NP-completeness. Finally, we present an efficient coding algorithm for finding coding solutions.

3.3.1 Packet Coding Gain

In the original literature of COPE [35], there is no metric available for quantitatively measuring the "goodness" of a coding solution. Here, we introduce a metric for such purpose.

Definition 1. *Define the* packet coding gain *(PCG) of a coding solution* $P_0 \oplus \cdots \oplus P_L$ ($L \geq 0$) *as the ratio* γ *of the expected number of successfully decoded bytes (after this encoded packet is transmitted) to the encoded packet size in byte, i.e.,*

$$\gamma = \frac{p_0^r \cdot p_0^d \cdot l_0 + p_1^r \cdot p_1^d \cdot l_1 + \cdots + p_L^r \cdot p_L^d \cdot l_L}{\max\{l_0, l_1, \cdots, l_L\}}, \tag{3.7}$$

where l_k is the size of packet P_k in byte, p_k^r is the probability that the encoded packet can be successfully received by the intended nexthop of P_k, and p_k^d is the probability that the encoded packet can be decoded by the intended nexthop of P_k, $k = 0, \ldots, L$.

When packets $P_0, \ldots, P_L$ are encoded together, the size of encoded packet is approximately equal to the size of the largest packet[6]. If this encoded packet is transmitted, it is expected that in total $p_0^r \cdot p_0^d \cdot l_0 + \cdots + p_L^r \cdot p_L^d \cdot l_L$ bytes will be successfully forwarded to nexthops. Thus, this metric accurately reflects the transmission efficiency improvement that can be achieved during the transmission period of encoded packet. By definition, we can know that $0 \leq \gamma < L+1$, and the larger the γ, the higher the transmission efficiency. According to this metric, we can classify different coding solutions into the following three categories:

a) $\gamma < p_0^r$: Node's transmission efficiency is lower than that of the non-coding transmission, that is, transmitting such an encoded packet has a lower transmission efficiency than transmitting P_0 alone.

b) $\gamma = p_0^r$: Node's transmission efficiency keeps unchanged, and it is same as that of transmitting P_0 alone.

c) $\gamma > p_0^r$: Node's transmission efficiency is improved in comparison with the non-coding transmission.

3.3.2 Problem Formulation of Optimal Packet Coding

With the help of PCG, we can now study the OPC problem for achieving the maximum coding benefit. The main idea of the OPC formulation is: given the related information of the head packet of FIFO queue and the head packets of virtual queues, to maximize

[6] In each encoded packet's header, several symbols are used for recording the number of native packets XOR-ed together, IDs of native packets, etc.

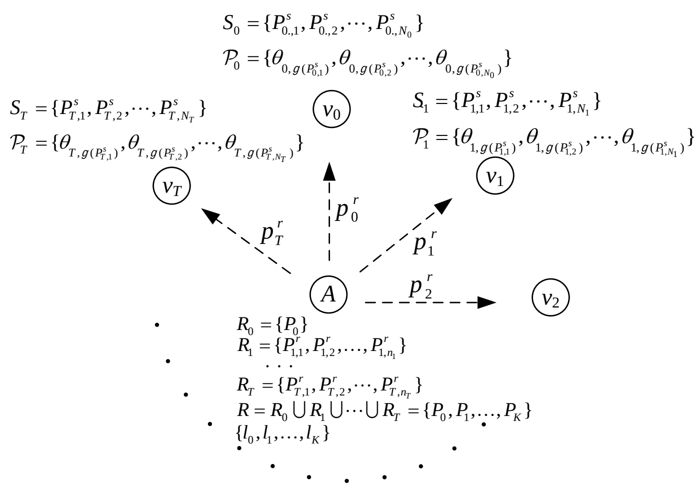


Figure 3.5: Parameters of the OPC problem.

the PCG of coding solution while satisfying the PT constraint introduced in Section 3.1 (i.e., each nexthop of the encoded packet can decode the encoded packet with probability not less than G).

Notations employed in the optimal packet coding problem are summarized in Table 3.2. Without loss of generality, we suppose that the nexthop of packet P_0 is downstream neighbor v_0. We call R_i in Table 3.2 the downstream neighbor v_i's *requirement set*. Since the current COPE architecture only supports unicast traffic (i.e. each flow only goes to one neighbor), we have $R_i \cap R_j = \emptyset$, $\forall 0 \leq i \neq j \leq T$. We call S_i in Table 3.2 downstream neighbor v_i's *storage set*. Since for any $P_k \in R_i$ $\theta_{i,k} = 0$, we have $S_i \subseteq R \backslash R_i$. Related parameters of the optimal coding problem are illustrated in Figure 3.5.

Based on the notations in Table 3.2, the OPC problem can be formally formulated as shown in Figure 3.6.

Obviously, the above OPC problem is an integer programming problem. The constraint set (3.10) enforces that at most one packet in R_1 can be encoded with P_0, and if such one packet exists, all other native packets participating in the coding must belong to S_1. Moreover, the additional constraint set (3.11) enforces that downstream neighbor v_1 can decode the encoded packet with a probability greater than G. Similar constraints are also applied to other downstream neighbors. It should be noted that, when G in the above formulation is set to zero, the objective of this optimization is exactly to find

Given: The threshold G, R_i, S_i, etc. (refer to Figure 3.5)
Encoded packet: $P = P_0 \oplus x_1 P_1 \oplus \cdots \oplus x_K P_K$
Maximize:
$\gamma = (p_0^r p_0^d l_0 + x_1 p_1^r p_1^d l_1 + \cdots + x_K p_K^r p_K^d l_K) / \max\{l_0, x_1 l_1, \cdots, x_K l_K\}$,
where $p_i^d = \prod_{j \in \{0,1,\cdots,K\} \setminus i} (\theta_{n,j})^{x_j}$, and n is the ID of packet P_i's intended nexthop (i.e. $P_i \in R_n$).
Over variables:
$x_i \in \{0,1\} : 1 \leq i \leq K$
$k_i^t \in \{0,1\} : 1 \leq t \leq T,\ 1 \leq i \leq n_t$
$k_{i,j}^t \in \{0,1\} : 0 \leq t \leq T,\ 1 \leq i \leq n_t,\ 1 \leq j \leq N_t\ (n_0 = 1)$
Subject to:
0) Constraints that ensure nexthop v_0 can decode its intended packet P_0 with probability greater than G:

$$X \oplus k_{1,1}^0 \mathbf{e}_{g(P_{0,1}^s)+1} \oplus \cdots \oplus k_{1,N_0}^0 \mathbf{e}_{g(P_{0,N_0}^s)+1} = \mathbf{e}_1, \tag{3.8}$$

$$\prod_{j \in \{1,\cdots,K\}} (\theta_{0,j})^{x_j} > G, \tag{3.9}$$

where $X = [1, x_1, \cdots, x_K]$.
1) Constraints that ensure nexthop v_1 can decode its intended packet with probability greater than G:

$$k_i^1 X \oplus k_{i,1}^1 \mathbf{e}_{g(P_{1,1}^s)+1} \oplus \cdots \oplus k_{i,N_1}^1 \mathbf{e}_{g(P_{1,N_1}^s)+1} = x_{g(P_{1,i}^r)} \mathbf{e}_{g(P_{1,i}^r)+1},\ \forall 1 \leq i \leq n_1, \tag{3.10}$$

$$\prod_{j \in \{0,1,\cdots,K\} \setminus \{g(P_{1,i}^r)\}} (\theta_{1,j})^{x_j} > x_{g(P_{1,i}^r)} \cdot G,\ \forall 1 \leq i \leq n_1. \tag{3.11}$$

2) $\cdots$

T) Constraints that ensure nexthop v_T can decode its intended packet with probability greater than G:

$$k_i^T X \oplus k_{i,1}^T \mathbf{e}_{g(P_{T,1}^s)+1} \oplus \cdots \oplus k_{i,N_T}^T \mathbf{e}_{g(P_{T,N_T}^s)+1} = x_{g(P_{T,i}^r)} \mathbf{e}_{g(P_{T,i}^r)+1},\ \forall 1 \leq i \leq n_T, \tag{3.12}$$

$$\prod_{j \in \{0,1,\cdots,K\} \setminus \{g(P_{T,i}^r)\}} (\theta_{T,j})^{x_j} > x_{g(P_{T,i}^r)} \cdot G,\ \forall 1 \leq i \leq n_T. \tag{3.13}$$

Figure 3.6: Mathematical formulation of the OPC problem.

the global optimal coding solution which has the largest PCG, under either the available virtual queue structure or the proposed virtual queue structure.

The following theorem demonstrates that the OPC problem is actually NP-complete no matter $G = 0$ or $G > 0$.

Theorem 3. *The OPC problem is NP-complete.*

Proof. It is easy to know that the OPC problem belongs to NP. Therefore, it is enough to show a polynomial-time reduction from the maximum clique (MC) problem described below (one of the typical NP-complete problems[77]) to the OPC problem.

Instance: An undirected graph $G = (V, E)$ and a positive integer $k \leq |V|$.

Question: Is there a set of k mutually adjacent nodes?

Here is the reduction. Given an arbitrary instance $G = (V, E)$ of the MC problem, where $V = \{u_1, u_2, \ldots, u_n\}$ and $E \subseteq \{\{u_i, u_j\} | u_i \in V, u_j \in V \text{and } u_i \neq u_j\}$, we construct a corresponding instance of the OPC problem as follows. Let the number of downstream neighbors in the OPC problem be $n+1$ and denote them by $v_0, v_1, \ldots, v_n$. Let $R_0 = \{P_0\}$ and $S_0 = \{P_1, P_2, \cdots, P_n\}$, where P_0 is the packet at the head of the output queue. For each $i \in \{1, 2, \cdots, n\}$, let $R_i = \{P_i\}$ and $S_i = \{P_0\} \cup \{P_j : (i, j) \in E\}$. Each $\theta_{i,j} \in \mathcal{P}_i$ is equal to 1 and all packets have the same size. Each p_i^r is approximately equal to one.

Based on the above construction of the OPC problem, we can know that the answer to the instance of the MC problem is "YES" if and only if there is a feasible packet coding of $k + 1$ packets for the corresponding instance of the OPC problem. □

3.3.3 Packet Coding Algorithm

Since the OPC problem is NP-complete, it is impossible for us to design a polynomial-time algorithm to find the optimal coding solution. In this section, we first show that although the available coding algorithm of COPE can still sever as a heuristic for finding feasible coding solutions in our new queue architecture, it cannot really take full advantage of coding opportunities offered by the new queue structure because of its several limitations.

We then show that due to some unique properties of the OPC problem, it is possible for us to design an efficient coding algorithm for finding good coding solutions.

Available coding algorithm of COPE

For the new flow-oriented virtual queue architecture, the available coding algorithm can still be used to find feasible coding solutions. By this algorithm, if P_0 is a small (large) packet, we first check one by one the small-size (large-size) head packets of virtual queues, and then check one by one the head packets of another size. Although such a simple search strategy has the advantage of very low time-complexity, it has the following limitations: (a) Many potentially good coding solutions will not be checked (refer to the example of Figure 3.3); (b) An infeasible coding solution is not always bad (in other words, a coding solution not satisfying the PT constraint may have a large packet coding gain); (c) A feasible coding solution sometimes may result in a small PCG. Due to the above severe limitations, there usually exist much better coding solutions than the one obtained by this algorithm.

A new coding algorithm

Although the OPC problem itself is NP-complete, it is possible for us to have a very efficient search for good coding solutions, due to the following good properties of the OPC problem:

- P1: Among all possible coding solutions, only the solutions that encode native packets destined to different nexthops need to be considered, because if two or more packets destined to the same nexthop are encoded together, their nexthop cannot decode the encoded packet.

- P2: Good coding solutions usually have high decoding probabilities, so they are very likely to satisfy the PT constraint with moderate threshold G. Therefore, we are able to greatly shrink the search space by searching for a good coding solution only among the solutions satisfying the PT constraint.

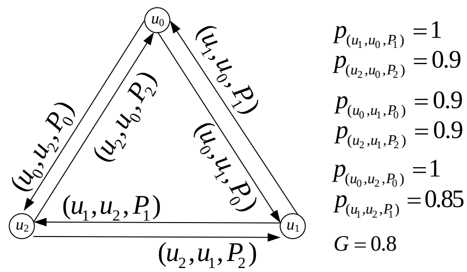


Figure 3.7: An example of a feasible coding graph.

P3: In most cases, encoding too many packets together will result in small decoding probabilities. We can achieve good performance by encoding at most four native packets in all cases, as indicated in [35].

Based on the above important properties of the original OPC problem, we propose here a new coding algorithm. The main idea is to first construct a directed graph corresponding to the original OPC problem, such that the search of coding solutions is reduced to the search of special subgraphs in this graph. Then, we repeatedly apply the above properties P1 and P2 to quickly remove arcs and nodes in the graph such that the search space of coding solutions can be quickly shrunk. The simplification of the graph and searching for subgraphs will be repeated at most three times by applying the above property P3.

Several main procedures of our coding algorithm are as follows.

Procedure 1 (Graph construction):

Given the OPC problem introduced in Section 3.3.2, construct a directed graph $\mathcal{G}(\mathcal{N},\mathcal{A})$ (referred to as a *coding graph* henceforth). The node set $\mathcal{N}$ of $\mathcal{G}$ is defined as $\mathcal{N} = \{u_0, u_1, \ldots, u_T\}$, where node u_i corresponds to downstream neighbor v_i in the OPC problem and has a weight $z_{(u_i)} = p_i^r$. The arc set $\mathcal{A}$ of $\mathcal{G}$ is defined as: for each packet $P_{i,k}^r$ in each R_i, there are T corresponding directed arcs $(u_i, u_j, P_{i,k}^r)$, $j \in \{0, 1, \ldots, T\}\backslash\{i\}$. Each arc $(u_i, u_j, P_{i,k}^r)$ has two different weights: an existence probability $p_{(u_i,u_j,P_{i,k}^r)}$ equaling to $\theta_{j,g(P_{i,k}^r)}$ and an integer length $s_{(u_i,u_j,P_{i,k}^r)}$ equaling to $l_{g(P_{i,k}^r)}$.

For a subgraph of $\mathcal{G}$, call it a feasible coding subgraph if:

(a) it contains v_0;

(b) for any two different nodes u_i and u_j in it, these is exactly one arc from u_i to u_j

and also one arc from u_j to u_i;

(c) all arcs departing from the same node have the same label P;

(d) for each node, the product of the existence probabilities of all arcs entering this node is larger than G.

Note that a feasible coding subgraph in $\mathcal{G}$ corresponds to a feasible coding solution. Figure 3.7 shows a simple feasible coding subgraph, whose corresponding feasible coding solution is $P_0 \oplus P_1 \oplus P_2$. We further call a feasible coding subgraph with k nodes a k-node feasible coding subgraph.

Procedure 2 (PT constraint-based graph simplification):

In this step, we apply the property of PT constraint to quickly remove some arcs and nodes in the coding graph $\mathcal{G}$. First, for any node u_i ($1 \leq i \leq T$), if $p_{(u_0,u_i,P_0)} < G$, remove this node and all its adjacent arcs. Then remove any arc (u_i, u_j, P) with $p_{(u_i,u_j,P)} < G$. Finally, remove any u_i and all its adjacent arcs if there is no arc from it to u_0.

Procedure 3 (Graph simplification before searching for k-node feasible subgraphs): Here the properties of k-node feasible subgraphs are applied to further simplify the coding graph $\mathcal{G}$.

(1) Remove each arc (u_i, u_j, P) with capacity $c_{(u_i,u_j,P)} < k-1$, where $c_{(u_i,u_j,P)}$ is the total number of arcs that leave from u_i and have the same label P as arc (u_i, u_j, P).

(2) Remove a node and all its adjacent arcs if its in-degree is less than $k-1$. Here, the *in-degree* of u_i is the number of nodes from which there exist one or more arcs to node u_i.

(3) Remove a node and all its adjacent arcs if there is no arc from it to u_0.

Repeat these steps until case (a): $\mathcal{G}$ does not contain u_0 or has less than k nodes, then return FALSE; or case (b): no node and arc in $\mathcal{G}$ can be removed any more, then return TRUE.

Procedure 4 (Searching for k-node feasible subgraphs):

In the current simplified coding graph $\mathcal{G}$, pick a coding subgraph $\mathcal{G}'$ which includes node u_0 and also $k-1$ other nodes. Determine if there are feasible subgraphs $\mathcal{G}_f = (\mathcal{N}_f, \mathcal{A}_f)$ in this coding subgraph $\mathcal{G}'$. If so, calculate the weight $W(\mathcal{G}_f)$ of each feasible subgraph

$\mathcal{G}_f$, where

$$W(\mathcal{G}_f) = \frac{\sum_{u_i \in \mathcal{N}_f} \left(z_{(u_i)} \cdot s_{(u_i,u_j,P)} \cdot \prod_{(u_j,u_i,P)\in\mathcal{A}_f} p_{(u_j,u_i,P)} \right)}{\max_{(u_i,u_j,P)\in\mathcal{A}_f} s_{(u_i,u_j,P)}},$$

and store this feasible subgraph if its weight is currently largest (The weight of a feasible coding subgraph is just the PCG of its corresponding feasible coding solution). Conduct this operation for each subgraph $\mathcal{G}'$. If current coding graph $\mathcal{G}$ does not contain any feasible subgraph, return FALSE; otherwise, return TRUE.

Formally, the new coding algorithm is as follows.

Packet Coding Algorithm

Input:

R_i $(0 \leq i \leq T)$ and size of each packet in R_i
Values of all $\theta_{m,n}$'s $(0 \leq m \leq T, 0 \leq n \leq K)$
Packet delivery ratio p_k^r $(0 \leq k \leq T)$
Value of probability threshold G

Main procedure

(1) Based on the input, construct the directed graph $\mathcal{G}$ by Procedure 1.
(2) Execute Procedure 2 to simplify the graph $\mathcal{G}$.
(3) **for** $k = 2$ to $min\{4, \text{node number of } \mathcal{G}\}$ **do**
  Execute Procedure 3 for $\mathcal{G}$.
  if (Procedure 3 return FALSE) **then**
    go to Exit.
  else
    Execute Procedure 4 for $\mathcal{G}$.
  end if
  if (Procedure 4 return FALSE) **then**
    go to Exit.
end for

Exit: Return the feasible subgraph which includes node u_0 and has the largest weight.

It is easy to know that if there does not exist any i-node feasible subgraph, definitely there does not exist any $(i+1)$-node feasible subgraph. To take advantage of this property, it is in ascending order of k that we search for k-node feasible subgraphs and calculate their PCGs, as shown in Step (3) of coding algorithm. Furthermore, according to property P3, the search will be conducted up to $k = 4$.

About the computational complexity of the coding algorithm, we can easily see that Procedures 1, 2 and 3 take time $O(T^3n)$, $O(T^3n)$ and $O(T^3n)$, respectively, where T is the number of downstream neighbors and n is the number of flows passing through a link. Procedure 4 has the highest computational complexity $O(T^3n^3)$ when $k = 4$. Thus, the computational complexity of this algorithm is $O(T^3n^3)$.

3.4 Simulation Results

In this section, we investigate how much the node transmission efficiency can be further improved by adopting the proposed virtual queue structure and packet coding algorithm, as compared with the original COPE. Since the PCG defined previously is used for measuring the short-term (one packet transmission period) transmission efficiency improvement, we define here a new metric to measure the long-term performance in terms of the node transmission efficiency. Let E_c and E_{nc} represent the average number of bytes delivered to neighbors per transmitted byte when using coding-based transmission and using non-coding (traditional) transmission, respectively. Then we define the *node transmission efficiency improvement* (NTEI) ρ as

$$\rho = E_c/E_{nc}.$$

This metric clearly reflects the improvement in the node transmission efficiency, independent of the adopted physical layer protocol (i.e. the bit-rate) and the MAC layer protocol. Using non-coding transmission, network nodes transmit native packets and suffer packet loss. Thus $E_{nc} < 1$. However, in the COPE-based networks, network nodes can forward multiple packets in a single packet transmission, so E_c can be larger than one there.

3.4.1 Simulation Setting

To evaluate the effectiveness of our proposed virtual queue structure and packet coding algorithm, we conduct simulations based on network configurations randomly generated as follows.

(a) Random topology generation: First, place node A at coordinate (0, 0). Then randomly and independently distribute N_n neighbors within the transmission range of unit one. The number of neighbors N_n follows a Poisson distribution with mean ζ [78]: $P(N_n = i) = e^{-\zeta}\frac{\zeta^i}{i!},\ i = 0, 1, 2, \ldots$.

(b) Random traffic generation: The total number of flows N_f passing through node A follows a Poisson distribution with mean λ: $P(N_f = k) = e^{-\lambda}\frac{\lambda^k}{k!},\ k = 0, 1, 2, \ldots$.

(c) For each flow, randomly select two neighbor nodes, say nodes X and Y. If the distance between nodes X and Y is less than or equal to unit one, randomly select two neighbor nodes again, until the distance between nodes X and Y is greater than unit one[7]. Then this flow will be routed through $X \to A \to Y$.

Each flow can be a UDP flow or a TCP flow, which comprises of a forward flow of data packets and a reverse flow of ACK packets with size 40 bytes. The data packet size of a flow remains unchanged and follows the packet size distribution presented in [79]. Since TCP is the dominant transport protocol for network applications [80], each generated flow is set to TCP flow with a probability of 80% and set to UDP flow with a probability of 20%. In addition, we consider the case that the flows are infinite and steady, and each flow always has packets in the output queue.

For wireless channels we adopt the Rayleigh block fading model and approximate the packet error rate of a channel with the probability that the instantaneous received SNR is smaller than a fixed threshold γ_T [81]. Then packet possession indicator $\theta_{m,n}$ is estimated based on the following model proposed in [82]:

$$\theta_{m,n} = \exp(-\frac{\gamma_T}{K} d^\alpha),$$

where d is the link distance, α is the path loss exponent and K is a constant depending on the transmitting power, the antenna gain, etc. The path loss exponent α is set to 4,

[7]Traffic between in-range nodes does not need to be forwarded by the relay.

and γ_T/K is set to 0.2, achieving a delivery ratio about 0.82 between two nodes with unit distance.

For each setting of triple (ζ, λ, η), we generate 10000 random configurations. For each configuration, we simulate the packet transmissions by using the non-coding transmission, the original COPE-based transmission and the improved COPE-based transmission, respectively. The observed NTEIs of the original COPE and the improved COPE are finally averaged over 10000 configurations.

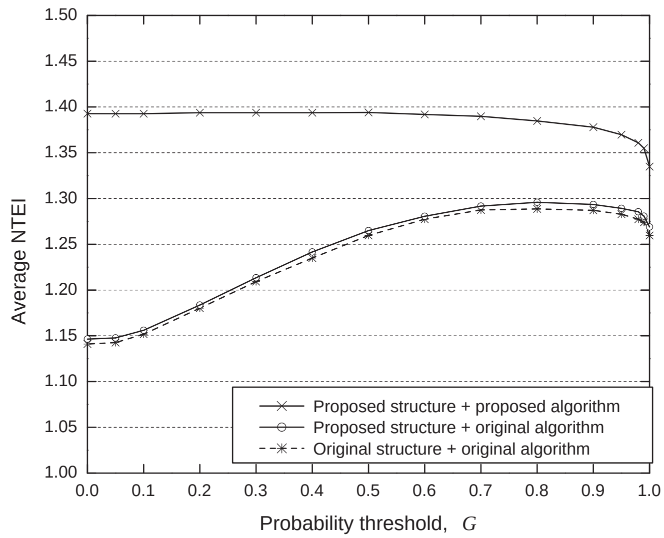
3.4.2 Average NTEI versus Threshold G

We first investigate the influence of threshold G on the NTEI, since it is a key parameter for the performance guarantee of both the original coding algorithm and the proposed coding algorithm. The corresponding results are summarized in Figure 3.8 for both the moderate traffic load case ($\zeta = 3$, $\lambda = 3$) and heavy traffic load case ($\zeta = 5$, $\lambda = 5$)[8].

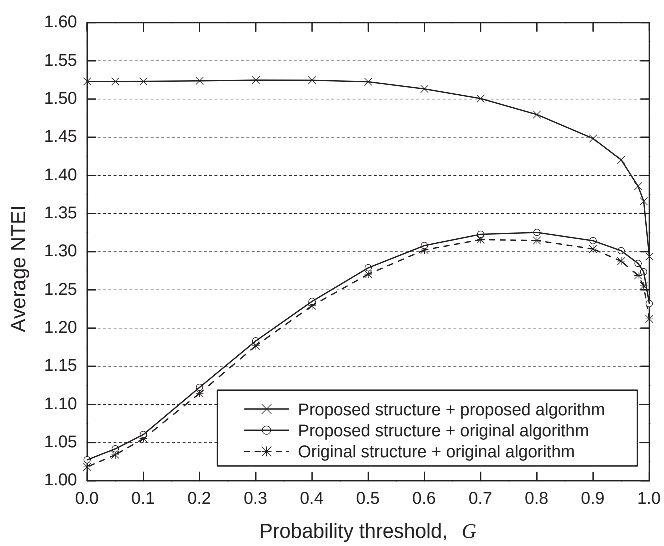
From Figure 3.8, we can clearly see that by applying the original coding algorithm to the new flow-oriented queue structure, we can only have a very slight improvement in NTEI as compared with the original COPE. However, when the flow-oriented queue structure and the new coding algorithm are jointly adopted, the average NTEI can be dramatically improved. For example, in networks with $G = 0.8$, $\zeta = 5$ and $\lambda = 5$, the NTEI can be slightly improved from 1.31 to 1.32 when the new queue structure and the original coding algorithm are applied, but this improvement can be as high as 16.5% when the new queue structure and the proposed coding algorithm are jointly applied.

Figure 3.8 also indicates clearly that for both the original and the proposed coding algorithms there exist optimal setting of G to maximize the NTEI, and this optimal value of G is sensitive to which coding algorithm is adopted but insensitive to the queue architecture and network characteristics (as also confirmed under other network settings). For the original coding algorithm, we can observe that it does not work well when G is too small or too large. The best value of G is around 0.8 (the default value for G

[8] Percentage of timely received reception reports is small when the network traffic is light or heavy, and is large when the network traffic is moderate[35].



(a) Average NTEI in networks with $\zeta = 3$, $\lambda = 3$ and $\eta = 0.7$.



(b) Average NTEI in networks with $\zeta = 5$, $\lambda = 5$ and $\eta = 0.3$.

Figure 3.8: Average NTEI under different different probability thresholds G.

used in [35]). This can be explained as follows. If G is set to a very small value (say 0.1), network nodes have a very high probability of encoding multiple packets together. However, according to the definition of PCG (Equation 3.7) we know that such a low probability threshold may induce a very small PCG. For instance, suppose that G is set to 0.1 and node A has two downstream neighbors. Let $R_1 = \{P_1\}$, $R_2 = \{P_2\}$, $l_1 = l_2$, $p_1^r = 0.8$, $p_2^r = 0.8$, $\theta_{1,2} = 0.2$ and $\theta_{2,1} = 0.2$. Then this node will encode P_1 and P_2 together and the resulting PCG is 0.32. Transmitting this encoded packet is even much worse than transmitting a native packet. On the contrary, if G is too large, although we accomplish the purpose that each encoded packet usually has a large PCG, network nodes have very few coding opportunities and consequently the coding gain is small. For the proposed coding algorithm, it is clear that the smaller the value of G, the better the performance. This is clearly demonstrated in both Figures 3.8(a) and 3.8(b). One important conclusion we can draw is that the proposed coding algorithm with $G = 0.5$ almost achieves the same performance as that with $G = 0$. However, setting G to 0.5 can greatly shrink the search space of coding solutions and significantly reduce the search time. Similar conclusions on the setting of G can be drawn under other settings of ζ and λ.

To have a fair comparison between the original COPE and the COPE adopting the new queue architecture and new coding algorithm, in the following we set G to 0.8 and 0.5 in the original and proposed coding algorithms, respectively.

3.4.3 Average NTEI versus Percentage of Timely Received Reception Report

We denote by the parameter η the percentage of reception reports which are timely received by node A. When η is large, a network node has much information about what packets its neighbors have and thus can obtain better coding solutions as compared with the case of a small η. Here, we investigate the impact of η on the performance of COPE. Figure 3.9 shows the average NTEI of the original COPE and the COPE with the flow-

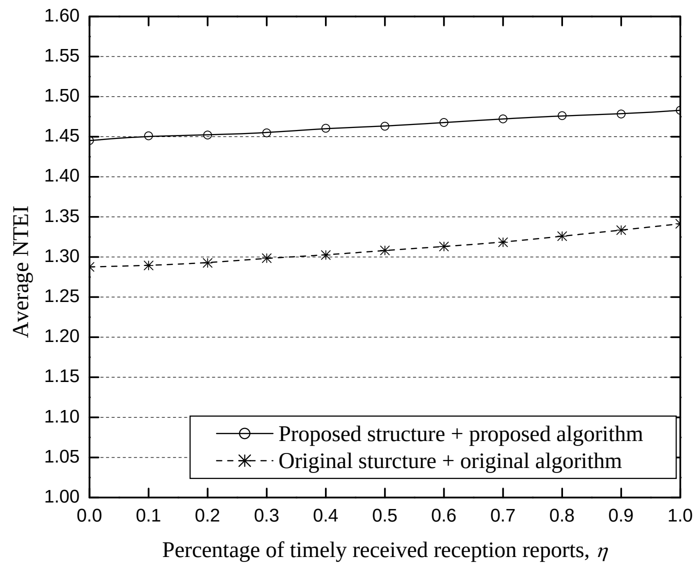


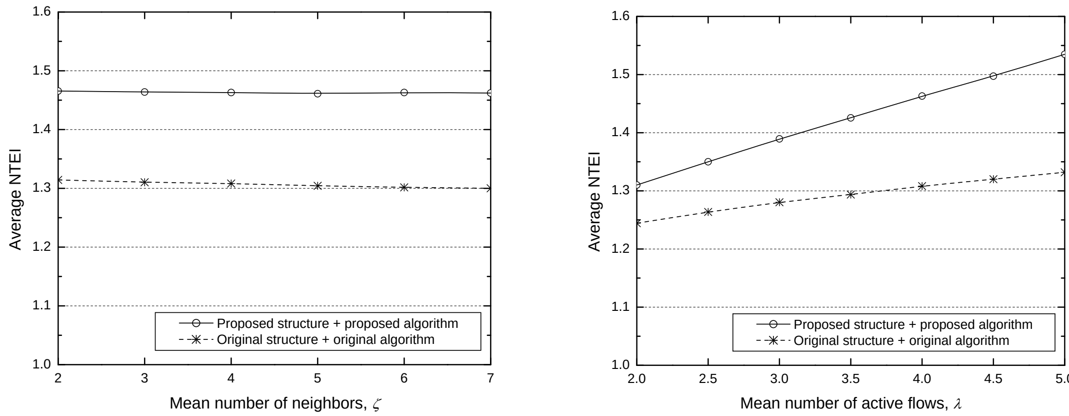
Figure 3.9: Average NTEI under different η. ($\zeta = 4$ and $\lambda = 4$)

oriented queue structure and the proposed coding algorithm, under different values of η. From this figure, we can see that under different η, the COPE with the proposed structure and proposed coding algorithm can always further significantly improve the average NTEI, as compared with the original COPE. We can also see that for both two schemes, although the average NTEI increases as η increases, this increase is not so significant. Take the improved COPE as an example. When the η varies from 0 to 1, the corresponding NTEI only slightly increases from 1.44 to 1.48. The results in Figure 3.9 indicate that the NTEI is actually not very sensitive to the variation of η. Thus, the late arrival or loss of reception reports will not severely degrade the performance.

3.4.4 Average NTEI versus Node Density and Traffic Load

Multihop wireless networks can be characterized by the average number of neighbor nodes ζ (node density) and average number of flows λ passing through a network node (traffic load). Figure 3.10(a) and 3.10(b) show the average NTEI under different ζ and λ, respectively.

First, both two figures clearly demonstrate that the proposed coding scheme significantly outperforms the original coding scheme under different ζ and λ. Additionally, in



(a) Node throughput improvement under different mean numbers of neighbors. ($\lambda = 4$, $\eta = 0.5$)

(b) Node throughput improvement under different mean numbers of active flows. ($\zeta = 4$, $\eta = 0.5$)

Figure 3.10: Average NTEI under different mean numbers of neighbors and different mean numbers of active flows.

Figure 3.10(a), we can see that the average NTEI almost keeps unchanged as ζ increases. This indicates that given the number of flows passing through a node, the NTEI is not sensitive to the variation of node density. In Figure 3.10(b), we can see that the average NTEI of original COPE increases slowly as λ increases, whereas the average NTEI of improved COPE increases rapidly as λ increases. In the original COPE, because the virtual queue is packet size-oriented, at most two packets destined to the same neighbor can be the candidate packets for coding with the head packet of output queue. Thus, once there are already at least two flows going to a neighbor, increasing more flows going to this neighbor will not increase the number of candidate packets. For the new flow-oriented virtual queue structure, however, the oldest packet of each flow is a candidate packet. Therefore, under such a structure, the node transmission efficiency increase greatly as λ increases. Such a performance characteristic is crucial for the practical application of the COPE architecture. Networks with heavy traffic load have a higher demand of improving the node throughput as compared with networks with light traffic load, so COPE neatly meets such a demand.

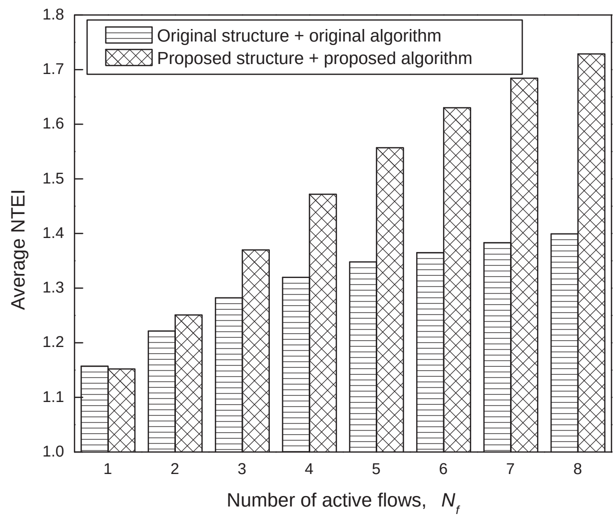


Figure 3.11: Average NTEI of network nodes with the same number of active flows N_f. ($\zeta = 4$)

3.4.5 Average NTEI versus Number of Active Flows

In multihop wireless networks, a flow may traverse several nodes and its end-to-end throughput is upper bounded by the bottleneck node with the heaviest traffic. Therefore, in comparison with those nodes with light traffic, it is more significant to improve the throughput of those bottleneck nodes for throughput improvement. Here, we investigate the average NTEI of those network nodes with the same number active flows N_f. Simulation results from the network configurations with $\zeta = 4$ are shown in Figure 3.11. From this figure, first we can observe that the larger the number of active flows passing through a node, the larger NTEI the COPE can provide (for both the original and new COPE). Second, and more importantly, compared to the network nodes with light traffic load, the transmission efficiency improvement from using the new COPE architecture is more significant for heavy load nodes. For example, on average, the transmission efficiency of network nodes with only two active flows can be further improved by only 2.94%, whereas the transmission efficiency improvement of network nodes with five active flows can be as high as 20.9%. Since the network throughput is limited by the bottleneck nodes, the results in this figure explicitly indicate that the COPE architecture can effectively improve

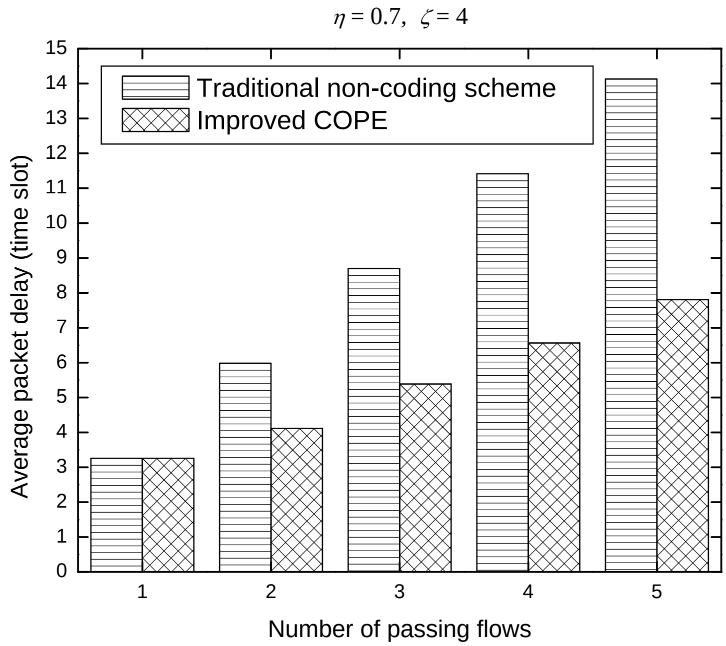


Figure 3.12: Average queueing delays of different schemes.

the network throughput, as confirmed in [35].

3.4.6 Packet Queueing Delay

Here we compare the queuing delay performance between the non-coding transmission with FIFO buffer and our improved COPE-based transmission in the following way: each flow passing through node A has 5 packets in the buffer queue and we simulate the average packet queueing delay during the transmission of these buffered packets. Here we assume that node A continuously transmits packets and each transmission takes a time slot of fixed duration. Figure 3.12 shows the average delay of all packets. We can observe that the COPE-based transmission greatly outperforms the traditional non-coding transmission. This is because the coding-based transmission can encode multiple packets together and deliver these packets via single transmission, and thus much faster deliver buffered packets to the node's neighbors.

3.5 Summary

In this chapter, we extended the current COPE architecture by first proposing a flow-oriented virtual queue structure for it and then introducing an efficient algorithm for searching good coding solutions under the new queue structure. This queue structure can not only completely eliminate the packet reordering but also offer the maximum number of coding opportunities under the condition that no packet reordering is allowed.

Extensive simulation results demonstrated that the available COPE can improve the node transmission efficiency, but this improvement can be more significant when the proposed virtual queue structure and new coding algorithm are jointly adopted. For example, in a network where each node on average has four neighbors and four active flows, the available COPE can improve the node transmission efficiency by around 30%, while this improvement can be as high as 45% with the help of the new queue architecture and coding algorithm. The results in this chapter also indicate clearly that compared to the network nodes with light traffic load, the transmission efficiency improvement from using the new COPE architecture is more significant for heavy load nodes(bottleneck nodes). For example, on average, the transmission efficiency of network nodes with only two active flows can be further improved by only 2.94%, whereas the transmission efficiency improvement of network nodes with five active flows can be as high as 20.9%.

Chapter 4

QoS-guaranteed Queueing and Packet Coding in Multihop Wireless Networks

In multihop wireless networks, it is necessary to provide suitable quality of service (QoS) support for the delivery of real-time audio, video and data. In order to support QoS on multi-hop paths, QoS must be designed for the end-to-end path as well as for each hop. The physical and MAC layers are responsible for QoS properties on a single-hop. In this chapter, we focus on the design of coding-based packet forwarding scheme (which works at the MAC layer) with the consideration of QoS issue.

The COPE architecture proposed by Katti *et al.* has demonstrated its capability of improving the network throughput by intelligently using network coding technique at the MAC layer [35]. However, it is still in its infancy and has the following limitations including the QoS problem: (1) COPE adopts the FIFO packet scheduler and thus does not enforce different priorities to different types of packets, like routing control packets, voice packets, best-effort packets, etc. (2) COPE simply classifies all packets destined to the same nexthop into small-size or large-size virtual queues and examines only the head packet of each virtual queue to find coding solutions. Such a queueing structure will lose some potential coding opportunities, because among packets destined to the same nexthop at most two packets (the head packets of small-size and large-size queues) will be examined in the coding process, regardless of the number of flows. (3) The coding algorithm adopted in COPE, which finds appropriate packets for coding, is fast but cannot always find good solutions. In order to address the above limitations, especially the incapability of providing QoS guarantees, in this chapter we first present a new queueing structure for COPE, which can provide more potential coding opportunities, and then propose a new packet scheduling algorithm for this queueing structure to guarantee different priorities for different types of packets. Finally, we propose an efficient coding algorithm to find appropriate packets for coding.

The rest of this chapter is organized as follows. In Section 4.1, we briefly review the COPE architecture and describe its limitations. Section 4.2 presents a new packet queueing structure and a new packet scheduling algorithm. In Section 4.3, we propose an efficient packet coding algorithm. Simulation results are presented in Section 4.4. Finally, Section 4.5 concludes this chapter.

4.1 Limitations of COPE

4.1.1 Limitations of the Available Queueing Structure

The current queueing structure of COPE is quite easy to maintain. However, it has the following two limitations: (1) In multihop wireless networks, it is quite necessary to give priority to some special types of packets (like routing-used control packets) over data packets [83]. Additionally, it is also necessary to set priorities among data packets. Although the FIFO scheduler is trivial to implement, it cannot satisfy this QoS requirement and it also allows rogue flows to capture an arbitrary fraction of the output bandwidth. (2) We should notice that under the condition that no packet reordering is allowed, theoretically all the oldest packets of distinct flows[1] have the potential to code together for throughput improvement. However, the current structure cannot fully explore this poten-

[1]Inside a node, the oldest packet of a flow is the firstly arrived packet among all the stored packets of this flow.

tial, because among those packets to the same neighbor at most two packets (the heads of two virtual queues) can be the candidate packets for encoding with P_0. More specifically, when more than one flow with small packets (or large packets) are routed to the same nexthop, only one oldest packet can locate at the head of the virtual queue (i.e., serves as the candidate packet). Therefore, this structure will significantly limit the potential coding opportunities.

To address the above two limitations of the available queueing structure in COPE, we propose a new queueing structure and a corresponding packet scheduling algorithm in Section 4.2.

4.1.2 Limitations of the Available Coding Algorithm

In the original literature of COPE [35], there is no metric available for quantitatively measuring the "goodness" of a coding solution. In Equation 3.7 of Chapter 3, we have introduced the following metric for such purpose.

$$\gamma = \frac{p_0^r \cdot p_0^d \cdot l_0 + p_1^r \cdot p_1^d \cdot l_1 + \cdots + p_L^r \cdot p_L^d \cdot l_L}{\max_{0 \le i \le L} l_i}, \tag{4.1}$$

The available coding algorithm does not take packet size and delivery ratio into account when searching for the coding solutions, and thus has the following limitations:

1) It skips all infeasible coding solutions, which may have large γ;

2) Many potentially good coding solutions will not be checked (After finding a feasible solution encoding k native packets, the algorithm will stop checking those unchecked solutions which encode k native packets, and attempt to find another native packet to code with the current k native packets.).

3) It may obtain a feasible coding solution which has a small γ. For example, the γ will be small when p_i^r's are small.

Due to the above severe limitations, there usually exist much better coding solutions

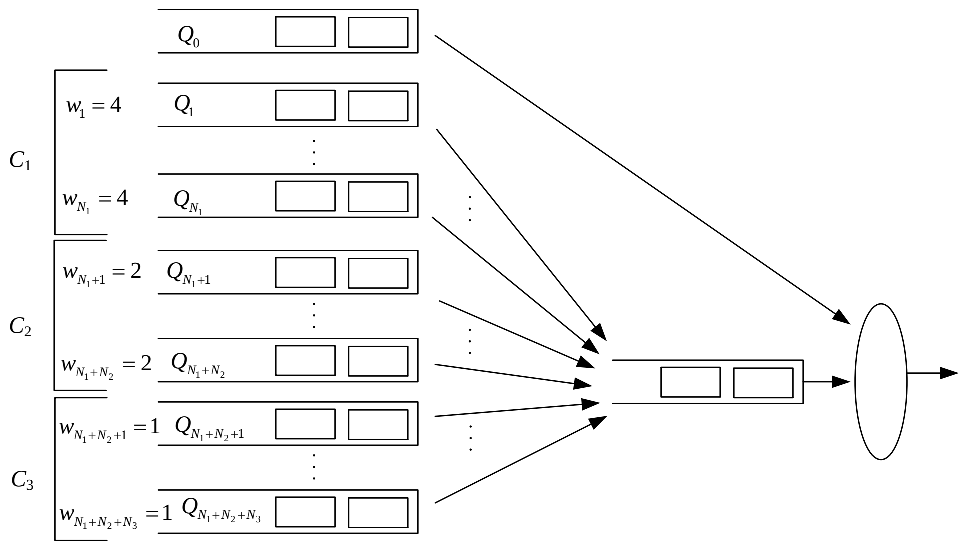


Figure 4.1: Flow-oriented queueing structure.

than the one obtained by this algorithm. We will present an efficient coding algorithm in Section 4.3.

4.2 Packet Queueing and Scheduling

In this section, we present a new packet queueing structure and also a packet scheduling algorithm.

4.2.1 Packet Queueing

Rather than queues all packets in a single queue, the new queueing structure is to maintain a dedicated FIFO queue Q_0, called *control queue*, for some special packets (like routing control packets) and maintain a FIFO queue Q_i for each active flow f_i passing through the current node, $i \geq 1$, as shown in Figure 4.1.

Such a queueing structure can provide more potential coding opportunities. Let us still consider the example in Figure 3.3. With the proposed queueing structure, P_3 will be at the head of the queue maintained for Flow 2 and thus the coding algorithm can find the feasible coding solution $P_0 \oplus P_3$. This example indicates that the new queueing structure increases candidate packets and consequently increases the coding opportunities.

Furthermore, this new queueing structure enables us to not only easily give higher priority to those special packets than to data packets, but also easily assign weights or priorities among data packets.

4.2.2 Packet Scheduling

With the above queueing structure, we proceed to specify how to assign transmission chances to data flows.

When a network node obtains a transmission chance from the MAC layer, its packet scheduler first checks whether the control queue is nonempty. If so, it will dequeue the packet at the head of control queue and transmit it alone (encoding it with other packets will decrease the probability of its successful delivery). In other words, data packets have no chance for transmission until there is no any packet in the control queue. Note that since packets in control queue only account for a small percentage of all buffered packets, giving priority to these packets almost does not affect the end-to-end delay of data packets [83].

In the following, we introduce how to schedule data packets. Similar to IEEE 802.11e, In our scheduling algorithm, traffic flows are also separated into the following three classes: flows of voice packets, flows of video packets and flows of best-effort packets, denoted by F_1, F_2 and F_3, respectively, as shown in Figure 4.1. Let N_i denote the number of flows belonging to class F_i for $i = 1, 2$ and 3, and let N denote the total number of active flows (i.e. $N = \sum_{i=1}^{3} N_i$). To achieve the target of giving higher priority to voice and video packets than to best-effort packets, we allocate larger weights to voice and video flows than to best-effort traffic flows. Denote by w_i the weight of flow f_i, and let W be $W = \sum_{i=1}^{N} w_i$. We expect that the percentage of transmission times assigned to flow f_i is approximately equal to w_i/W. Now an obvious problem arising is the appropriate value setting of w_i. In the IEEE 802.11e standard which supports multimedia applications such as voice and video over the IEEE 802.11 WLANs, by default, the contention window (CW) of best effort traffic is four times as large as voice packets' CW and two times as large as video packets' CW. Thus, a reasonable setting of w_i is as follows: $w_i = 4$ for each voice flow f_i, $w_j = 2$ for each video flow f_j and $w_k = 1$ for each best-effort traffic flow f_k. Note that, upon the specific requirement we can also separate packets into classes in other ways and set their respective weights. For example, among best-effort flows, web surfing can have larger weight than FTP and email applications.

With the above specifications, we schedule packets in a similar manner as round robin scheduling [84]. In order to explain the scheduling algorithm, we first clarify two concepts: *small-round exploring* and *large-round exploring*. A round of exploring N flows one by one is called a small-round exploring, and the conduction of $\max_{1 \le i \le N} w_i$ times of the small-round exploring is called a large-round exploring. For the above setting, $\max_{1 \le i \le N} w_i = 4$. Let I denote the ID of the flow which will be serviced at the current transmission time. In this scheduler, parameters R_i $(1 \le i \le N)$ are adopted to determine whether the scheduler will service a flow or skip over it. When starting a new large-round exploring, for each i initialize R_i as $R_i = w_i$ and set I to 1. R_i represents the number of times flow f_i needs to be serviced during the remaining services of the current large-round exploring. When a node obtains a transmission chance and the control queue is empty, the scheduler dequeues the packet at the head of Q_I and select by the coding algorithm appropriate packets to code with P_I. One important point we should notice is that by using network coding, multiple native packets can be forwarded by the transmission of an encoded packet. To achieve the target that the percentage of transmission times assigned to flow f_i is approximately equal to w_i/W, for each successfully decoded native packet P_i let $R_i = R_i - 1$. If P_I is successfully forwarded in the current transmission, conduct $I = I + 1$ until $R_I > 0$. Otherwise, keep I unchanged.

Now the scheduling algorithm is summarized in Figure 4.2.

Similar to the round robin scheduler, compared to the FIFO scheduler in COPE, such a scheduler has two important advantages: first, it prevents a rogue source from arbitrarily increasing its share of the bandwidth; second, it satisfies the QoS requirement of multihop wireless networks.

Scheduler

```
Dequeue the packet P_I at the head of Q_I.
Find appropriate packets to code with P_I by the coding algorithm and transmit the encoded packet.
for each successfully forwarded P_i do
    R_i = R_i − 1
end for
if P_I is successfully forwarded then
    I = I + 1
    if I > N then
        I = 1
    end if
end if
if all R_i are equal to zero do
    for i = 1 to N do
        R_i = w_i
    end for
    I = 1
else if R_I = 0 do
    while R_I = 0 do
        I = I + 1
    end while
end if
```

Figure 4.2: Packet scheduling algorithm.

4.3 Efficient Packet Coding Algorithm

To take full advantage of the coding opportunities provided by the new queueing structure, in this section, we present a more efficient coding algorithm than the available one in COPE.

As discussed previously, the available coding algorithm has several limitations which may lead to the obtaining of a not-so-good coding solution in the case there exist good coding solutions. However, it is possible for us to have a very efficient search for good coding solutions, due to the following good properties:

P1: Good coding solutions usually have high decoding probabilities, so they are very likely to satisfy the PT constraint (i.e. be feasible coding solutions). Therefore, we are able to greatly shrink the search space by searching for a good coding solution only among the feasible solutions.

P2: In most cases, encoding too many packets together will result in small decoding probabilities. We can achieve good performance by encoding not more than a given number of native packets (say 4) in all cases, as indicated in [35].

Based on the above properties, the goal of our coding algorithm is to find the best coding solution (with the largest γ) only among feasible coding solutions which encode at most N_{max} native packets. The appropriate value of N_{max} will be determined by virtue of simulation results. To describe this new coding algorithm, we first introduce a special type of directed graph, called *coding graph*.

Definition 2: (Coding Graph) Given knowledge (like packet size) of packet P_I being served by the packet scheduler and knowledge of all packets P_i's at the heads of queues, construct a corresponding coding graph $\mathcal{G}(\mathcal{V}, \mathcal{A})$ as follows:

- The vertex set $\mathcal{V}$ of $\mathcal{G}$ is defined as $\mathcal{V} = \{v_1, v_2, \ldots, v_N\}$, where node v_i corresponds to packet P_i. Assign two weights $s_i = l_i$ and $z_i = p_i^r$ to each node v_i.

- The arc set $\mathcal{A}$ of $\mathcal{G}$ is defined as: for each v_i $(i \neq I)$ satisfying $\theta_{N(P_i),I} > G$ and $\theta_{N(P_I),i} > G$, where $N(P_i)$ represents the nexthop ID of P_i, there are an arc (v_I, v_i) with weight $p_{(v_I,v_i)} = \theta_{N(P_i),I}$ and an arc (v_i, v_I) with weight $p_{(v_i,v_I)} = \theta_{N(P_I),i}$; between any two vertexes v_i $(i \neq I)$ and v_j $(j \neq I)$ from which there are arcs to v_I, if $\theta_{N(P_j),i} > G$ and $\theta_{N(P_i),j} > G$, there are an arc (v_i, v_j) with weight $p_{(v_i,v_j)} = \theta_{N(P_j),i}$ and an arc (v_j, v_i) with weight $p_{(v_j,v_i)} = \theta_{N(P_i),j}$.

For a subgraph of $\mathcal{G}$, call it a feasible coding subgraph if:

(a) it contains v_I;

(b) between any two different nodes u_i and u_j in it, these are arcs (u_i, u_j) and (u_j, u_i);

(c) for each node of this subgraph, the product of weights of all arcs entering this node is larger than G.

Note that a feasible coding subgraph $\mathcal{G}_f(\mathcal{V}_f, \mathcal{A}_f)$ in $\mathcal{G}$ corresponds to a feasible coding solution. Figure 4.3 shows a simple feasible coding subgraph, whose corresponding feasible coding solution is $P_I \oplus P_1 \oplus P_3$.

Let the weight $W(\mathcal{G}_f)$ of a feasible coding subgraph $\mathcal{G}_f$ be

$$W(\mathcal{G}_f) = \frac{\sum_{v_i \in V_f} \left(s_i \cdot z_i \cdot \prod_{(v_j,v_i) \in \mathcal{A}_f} p_{(v_j,v_i)} \right)}{\max_{v_i \in \mathcal{V}_f} s_i}. \tag{4.2}$$

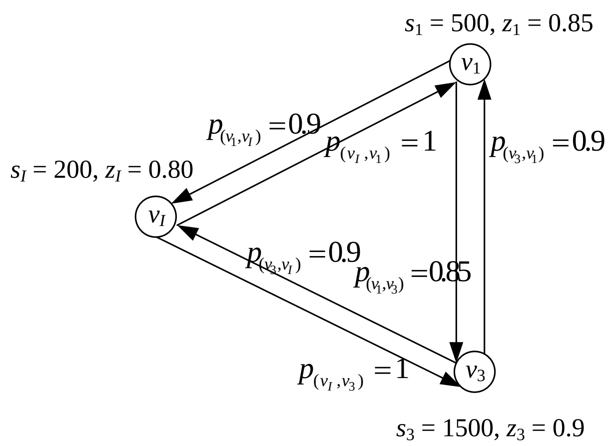


Figure 4.3: An example of a feasible coding graph for $G = 0.8$.

Clearly, the weight of a feasible coding subgraph is equal to the γ of the corresponding feasible coding solution. Formally, the coding algorithm is as follows.

Packet Coding Algorithm

Input:
  Value of I and size of each head packet P_i $(1 \leq i \leq N)$
  Values of all $\theta_{m,n}$'s $(0 \leq n \leq N)$
  Packet delivery ratio p_k^r $(0 \leq k \leq N)$
Procedure
Based on the input, construct the corresponding coding graph $\mathcal{G}$.
$W_{max} = 0$.
for $k = 2$ to $min\{N_{max}$, node number of $\mathcal{G}\}$ **do**
  for each subgraph $\mathcal{G}'$ containing v_I and also $k - 1$
  other vertexes **do**
    if $\mathcal{G}'$ is feasible **do**
      if $W(\mathcal{G}') > W_{max}$ **do**
        $W_{max} = W(\mathcal{G}')$
      end if
    end if
  end for
end for
Exit: Return the feasible subgraph which includes node v_I and has the largest weight.

This new coding algorithm takes $O(N^{N_{max}-1})$ time, which is quite fast when N_{max} is small. Simulation results in the next section will demonstrate that setting N_{max} to 3 can achieve good enough performance. Thus, this algorithm only takes $O(N^2)$ time when $N_{max} = 3$.

4.4 Simulation Results

In this section, we investigate how much the node transmission efficiency can be further improved by adopting the proposed queueing structure and coding algorithm in COPE, as compared with the original COPE.

Since the γ defined previously is used for measuring the short-term (one packet transmission period) transmission efficiency improvement, we define here a new metric to measure the long-term performance in terms of the node transmission efficiency. Let E_c and E_{nc} represent the average number of bytes delivered to neighbors per transmitted byte when using coding-based transmission and using non-coding (traditional) transmission, respectively. Using non-coding transmission, network nodes transmit native packets and suffer packet loss. Thus $E_{nc} < 1$. However, in the COPE-based networks, network nodes can forward multiple packets in a single packet transmission, so E_c can be larger than one there. Then we define the *node transmission efficiency improvement* (NTEI) ρ as

$$\rho = E_c / E_{nc}. \tag{4.3}$$

This metric clearly reflects the improvement in the node transmission efficiency, independent of the adopted physical layer protocol (i.e. the bit-rate) and the MAC layer protocol.

4.4.1 Simulation Setting

The performance evaluation is conducted on network configurations randomly generated as follows. (a) Random topology generation: First, place relay node A at coordinate (0, 0). Then N_n neighbors are randomly and independently distributed within the transmission range of unit one. Each generated topology consists of one transmission node and several neighbors. The transmission node continuously transmits packets (which are native packets when using the non-coding transmission way and are encoded packets when adopting network coding) and the neighbors receive the packets. The E_c and E_{nc} are the ratios of the total number of successfully delivered bytes to the total number of transmitted bytes when using the coding-based transmission way and the non-coding transmission way, respectively. (b) Due to the small percentage of special packets link control packets, only data packets are considered in the simulation. For each data flow, randomly select two neighbors X and Y. If their distance $d(X, Y) \leq 1$, randomly select two neighbors again until $d(X, Y) > 1$[2]. Then this flow will be routed through $X \to A \to Y$. Each best-effort TCP flow comprises of a forward flow of data packets and a reverse flow of ACK packets with size 40 bytes. The data packet size of a flow remains unchanged and follows the packet size distribution presented in [79]. In addition, we consider the case that the flows are infinite and steady, and each flow always has packets in the output queue.

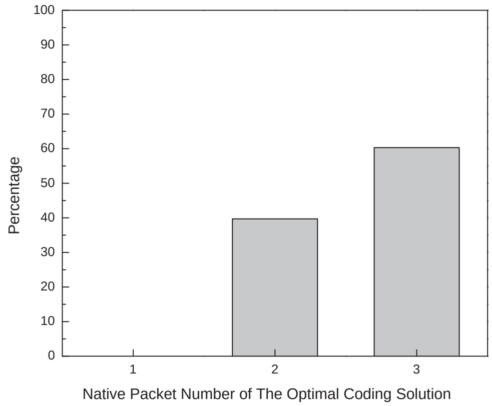
For wireless channels we adopt the Rayleigh block fading model and approximate the packet error rate of a channel with the probability that the instantaneous received SNR is smaller than a fixed threshold γ_T [81]. Then packet possession indicator $\theta_{m,n}$ is estimated based on the following model proposed in [82]: $\theta_{m,n} = \exp(-\frac{\gamma_T}{K} d^\alpha)$, where d is the link distance, α is the path loss exponent and K is a constant depending on the transmitting power, the antenna gain, etc. The path loss exponent α is set to 4, and γ_T/K is set to 0.2, achieving a delivery ratio about 0.82 between two nodes with unit distance.

For each setting of the numbers of flows and neighbors, we generate 5000 random configurations. For each configuration, we simulate the packet transmissions by using the non-coding transmission, the original COPE-based transmission and the improved COPE-based transmission, respectively. The observed NTEIs of the original COPE and the improved COPE are finally averaged over 5000 configurations.

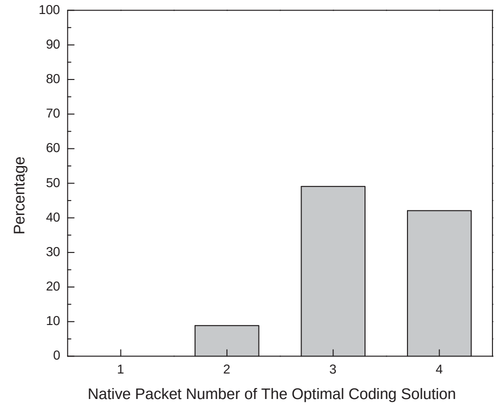
4.4.2 Shortcoming of Probability Threshold Constraint

The available coding algorithm in COPE aims to encode as many as possible native packets together while satisfying the PT constraint. However, the PT constraint only considers

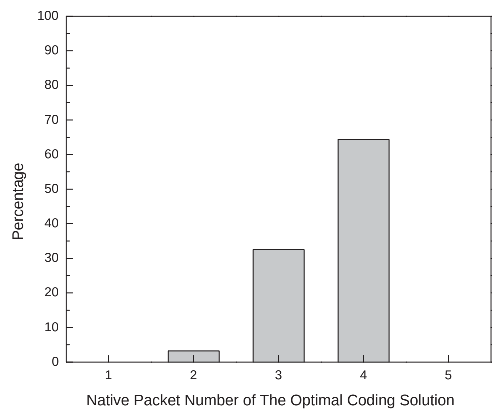
[2] Traffic between in-range nodes does not need to be forwarded by the relay.



(a) Distribution among those cases with $N_m = 3$.



(b) Distribution among those cases with $N_m = 4$.

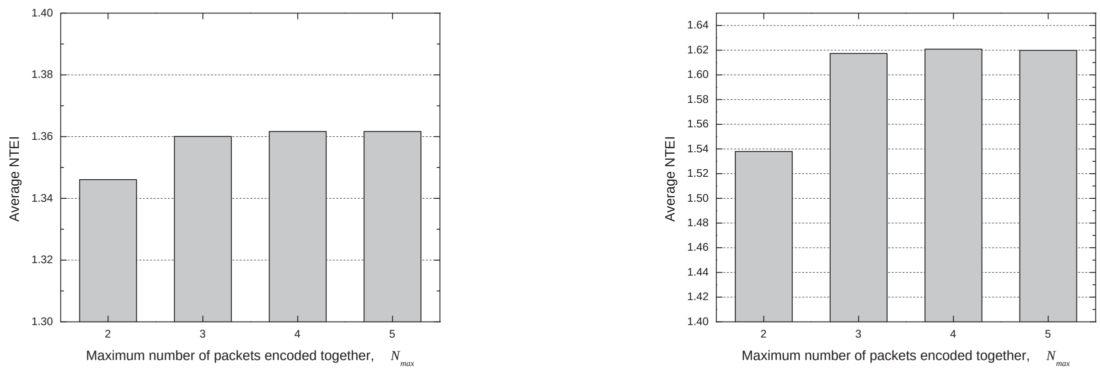


(c) Distribution among those cases with $N_m = 5$.

Figure 4.4: Distribution of the number of native packets in the optimal coding solutions.

the probabilities of successful decoding at nexthops, but does not take into account sizes of native packets and the link delivery ratios of those links from the delay node to nexthops. Thus, a feasible coding solution encoding many packets does not necessarily have a large γ.

For a coding problem, let N_m be the maximum number native packets which can be encoded together while satisfying the PT constraint. Figure 4.4 shows the distribution of native packet number of the optimal coding solution, among the cases with the same N_m. We can see that although at most N_m packets can be encoded together while satisfying the PT constraint, the optimal coding solutions are often some solutions which encodes less than N_m. For example, for those cases with $N_m = 5$, all the optimal coding solutions



(a) Average NTEI of nodes with normal traffic load and a normal number of neighbors. ($N_1 = 1, N_2 = 1, N_3 = 2$ and $N_n = 4$)

(b) Average NTEI of nodes with heavy traffic load and a large number of neighbors. ($N_1 = 1, N_2 = 1, N_3 = 5$ and $N_n = 7$)

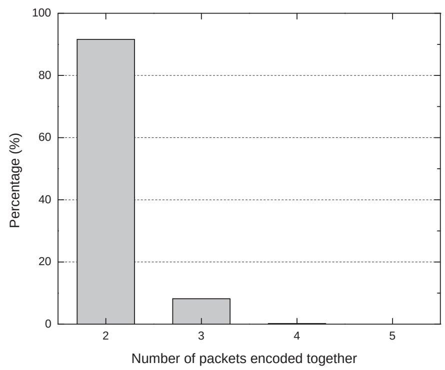
Figure 4.5: NTEI versus the maximum number of packets allowed to encode together.

encode less than five native packets. Therefore, the PT constraint is not good enough as a metric for measuring the "goodness" of a coding solution. It is quite necessary to take into account sizes of native packets and the link delivery ratios of those links from the delay to nexthops, as shown in Equation (4.1).

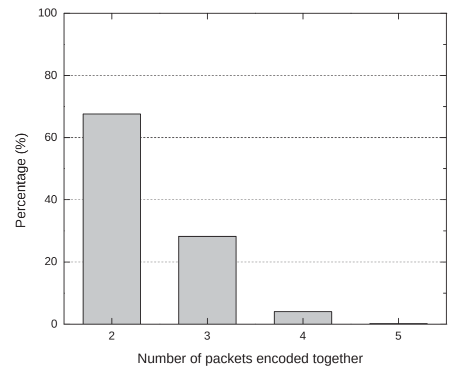
4.4.3 NTEI versus Maximum Number of Packets Allowed to Encode Together

Now we will investigate the appropriate setting of the maximum number N_{max} of packets allowed to encode together in the proposed coding algorithm. Figure 4.5 shows the average NTEI under different values of N_{max}. We can observe that compared to the setting $N_{max} = 2$, the setting $N_{max} = 3$ leads to a much larger average NTEI. However, setting N_{max} to 4 or a larger value only very slightly increases the average NTEI. Therefore, we can conclude that setting N_{max} to 3 can achieve good enough performance.

In order to clearly understand this performance characteristic, we further examine in great detail the distribution of the number of native packets encoded together in Figure 4.6. We can see that it is very rarely happen to encoded four or more packets together. This is easy to understand. Let us take the case of encoding four native packets as an example. To encode four packets together, each one of four nexthops of the encoded packet

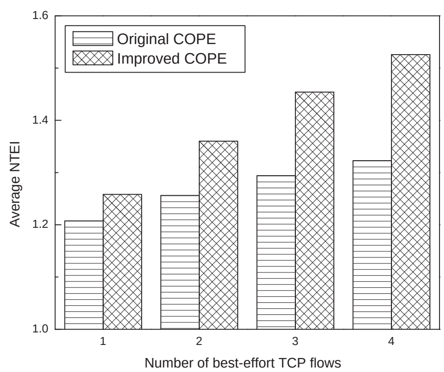


(a) At nodes with normal traffic load and a normal number of neighbors. ($N_1 = 1, N_2 = 1, N_3 = 2$ and $N_n = 4$)

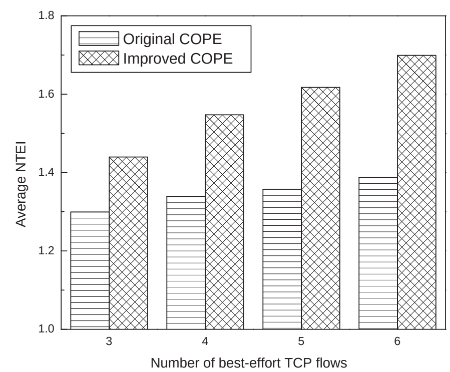


(b) At nodes with heavy traffic load and a large number of neighbors. ($N_1 = 1, N_2 = 1, N_3 = 5$ and $N_n = 7$)

Figure 4.6: Distribution of number of packets coded together.



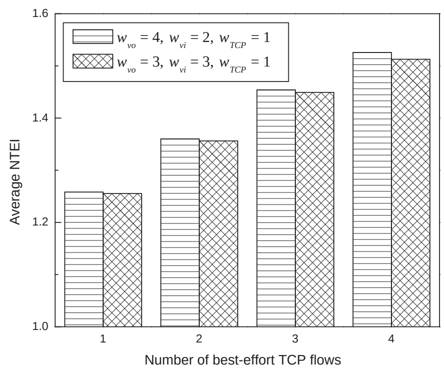
(a) At nodes with four neighbors. ($N_1 = 1, N_2 = 1$)



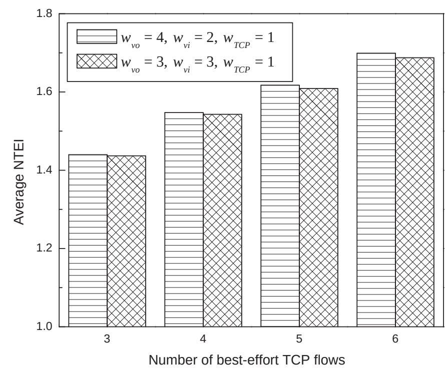
(b) At nodes with seven neighbors. ($N_1 = 1, N_2 = 1$)

Figure 4.7: Comparison between the original COPE and improved COPE.

needs to possess other three packets except the packet destined to it. This condition is so strict that it can be rarely satisfied. Due to the low probability of encoding four or more packets, setting N_{max} to 3 can achieve good enough performance and also lead to a low computational complexity of the coding algorithm.



(a) At nodes with four neighbors. $(N_1 = 1, N_2 = 1)$



(b) At nodes with seven neighbors. $(N_1 = 1, N_2 = 1)$

Figure 4.8: NTEI versus different settings of flow weight.

4.4.4 Comparison between the Original COPE and Improved COPE

In this subsection, we investigate the improvement achieved by adopting the new queueing structure and new coding algorithm. Figure 4.7 shows the average NTEI achieved by the original COPE and the improved COPE, respectively. We can see that the improved COPE always significantly outperforms the original COPE. For example, the average NTEI of nodes with 7 neighbors, one voice flow, one video flow and four TCP flows, is improved by 15.6%. In addition, the improvement increases as the number of active flows increases. This is because compared to nodes with few active flows, nodes with a lot of active flows have more potential coding opportunities and thus remain larger scope for improvement.

4.4.5 NTEI under Different Settings of Flow Weight

In the scheduling algorithm, different types of flows are assigned with different weights. Now we will investigate whether the algorithm performance is sensitive to the assignment of flow weight. Figure 4.8 shows the average NTEI under two different settings of flow weight. We can see that the average NTEI almost keep unchanged under these two settings. The same conclusion can be drawn when other settings are used. Therefore, we

Table 4.1: The average solution search time under different numbers of passing flows

N_3	1	2	3	4	5
Average search time (μs)	2.81	3.18	4.48	5.02	5.80

can expect that the NTEI will only slightly change when other schedulers like the one in [85] are adopted for the proposed queueing structure.

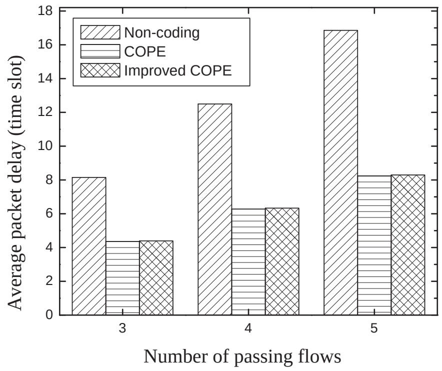
4.4.6 Packet Delay

Here we investigate the delay performance of the improved COPE.

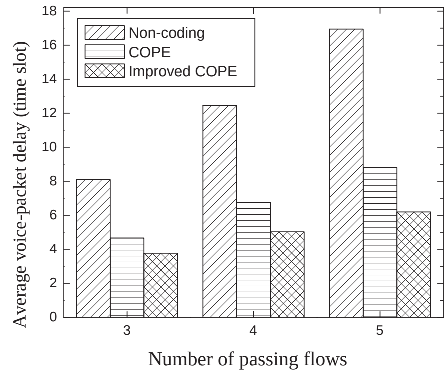
First, the storing function will not increase the packet delay. The COPE's storing function at one node is used only to store the overheard packets (not the forwarded packets) for a period in a particular buffer, which is not the buffer for queueing the packets to be forwarded. The packets needing to be forwarded wait in their own queue for getting their transmission chances, just like the current packet forwarding architecture.

Then we investigate the average running time for the search of coding solution. Table 4.1 shows the average running time for finding a set of packets for coding under different numbers of passing flows, among the cases with $N_1 = 1$, $N_2 = 1$ and $N_n = 7$. From this table we can see that the solution search time is at the microsecond level and is very small as compared with other parts like queueing delay. As for packet coding (i.e. XOR-ing) and packet decoding, they are linear operations and consume almost neglectable time.

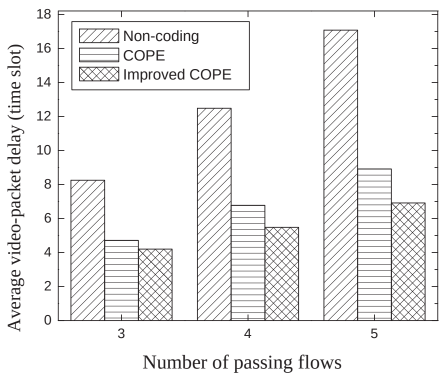
Finally, we compare the queuing delay performance between the non-coding transmission with FIFO buffer, the current COPE-based transmission and our improved COPE-based transmission in the following way: each flow passing through node A has 4 packets in the buffer queue and we simulate the average packet queueing delay during the transmission of these buffered packets. Here we assume that node A continuously transmits packets and each transmission take a time slot of fixed duration. Figure 4.9 shows the average delay of all packets, the average delay of voice packets, the average delay of video packets and the average delay of TCP packets, respectively. First, we can observe that



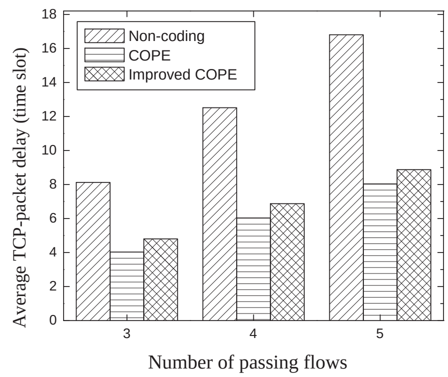
(a) Average delay of all packets.



(b) Average delay of voice packets.



(c) Average delay of video packets.



(d) Average delay of TCP packets.

Figure 4.9: Average queueing delays of different schemes.($N_1 = 1$, $N_2 = 1$, $N_n = 4$.)

both the COPE-based transmission and the improved COPE-based transmission greatly outperform the traditional non-coding transmission, because they can much faster deliver buffered packets of a node to this node's neighbors. In addition, from Figures 4.9(b) and 4.9(c) we can see that, compared to the COPE-based transmission, the improved COPE-based transmission leads to a smaller average delay of voice packets and a smaller average delay of video packets. This is because the improved COPE-based transmission gives high priority to the voice packets and video packets, at the cost of slightly increasing the average delay of TCP packets (as shown in Figure 4.9(d)).

4.4.7 The End-to-end Throughput

The node-level transmission efficiency improvement and delay performance improvement shown above can also suggest that the end-to-end throughput will be improved. We can understand this in the following way. Using the network coding technique to forward multiple packets via one packet transmission, is just like using a larger transmission bandwidth to improve the node transmission rate. Our improved COPE can forward more packets per packet transmission than the current COPE by more effectively utilizing the network coding technique. Therefore, the improve COPE can further improve the node-level performance (the transmission rate and packet delay) and consequently will improve the network-level performance.

4.5 Summary

In this chapter, we presented for the COPE architecture a new flow-oriented queueing structure which can increase the potential coding opportunities and are convenient for the allocation of priorities to packets, and also proposed a new efficient packet coding algorithm. Rather than adopting FIFO scheduler, allocating priorities to different flows can satisfy the QoS requirement of multihop wireless networks for supporting real-time services such as voice applications. To our knowledge, this is the first time to take the QoS issue into account in the literature of wireless network coding. Simulation results demonstrate that by adopting the new queueing structure and new coding algorithm, COPE can further greatly improve the node transmission efficiency.

Chapter 5

Network Coding-Based Reliable Multicast in Wireless Networks

Reliable multicast [86, 87], the lossless delivery of bulk data from one sender to a group of receivers, is widely used in many important applications such as the file distribution to a number of receivers and the dissemination of market data from a financial institution to its subscribers.

The reliable multicast generally does not allow data loss, but can tolerate delay due to retransmissions. Traditionally, to ensure the reliable link-layer multicast the source simply retransmits one by one the *lost packets* (i.e. the packets that are not received yet by one or more receivers). Recently, Nguyen *et al.* [47, 88] applied network coding to the reliable link-layer multicast in wireless networks and proposed two network coding-based schemes (a static one and a dynamic one) for it. The main idea of these coding-based reliable multicast schemes is to first buffer the lost packets for some time, then, instead of transmit these lost packets one by one, the source XORs an optimal set of lost packets with distinct intended receivers together into one packet and transmits this XOR-ed packet in one retransmission[1]. The main difference between the static and dynamic schemes in [47, 88] is that the static one will repeatedly retransmit the same XOR-ed packet until all its intended receivers successfully receive it, while the dynamic one can

[1]The intended receivers of a packet are the receivers which have not received this packet.

dynamically update the XOR-ed packet in each retransmission for a further improvement in transmission efficiency.

By intelligently XORing multiple lost packets together, the available coding-based multicast schemes can result in a significant improvement on the transmission efficiency of reliable link-layer multicast. However, these schemes suffer from two main limitations. First, its coding principle that only the lost packets with distinct intended receivers can be XORed together, is too strict to fully explore the potential coding opportunities, since the lost packets with common intended receivers also have the potential to be encoded together for transmission efficiency improvement. Second, in the current schemes the search algorithm for the optimal set of lost packets to XOR is very complex (actually, NP-complete), which significantly limits the scalability of these schemes.

In this chapter, we propose two improved schemes for reliable link-layer multicast such that the above limitations of the available coding-based schemes can be significantly alleviated. In summary, the main contributions of this work are as follows:

1. We first prove that in the current reliable multicast schemes, the search problem for the optimal set of lost packets to encode is NP-complete.

2. We then propose two improved schemes (also a static one and a dynamic one) to significantly reduce the search complexity (to polynomial time) and also to fully exploit the potential coding opportunities.

3. We provide analytical analysis to evaluate the performance in terms of both transmission efficiency and packet delay for two proposed reliable multicast schemes.

4. We demonstrate that although two available coding-based schemes have lower bandwidth requirement than the traditional non-coding scheme, the proposed schemes can further greatly reduce the bandwidth requirement, especially in the case of high packet loss probabilities and large number of receivers.

The rest of this chapter is organized as follows. Section 5.1 briefly reviews two available coding-based multicast schemes and section 5.2 presents two improved coding-based

	P_1	P_2	$\cdots$	P_N
R_1	$e_{1,1}$	$e_{1,2}$	$\cdots$	$e_{1,N}$
R_2	$e_{2,1}$	$e_{2,2}$	$\cdots$	$e_{2,N}$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
R_M	$e_{M,1}$	$e_{M,2}$	$\cdots$	$e_{M,N}$

Figure 5.1: Packet-loss table inside the source node.

multicast schemes. In Section 5.3, we analytically evaluate the transmission bandwidth and delay performance for two proposed schemes. Numerical results obtained from the analytical model and simulation are presented in Section 5.4. Finally, Section 5.5 concludes this chapter.

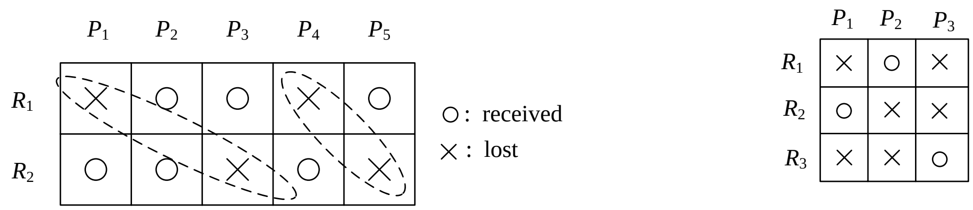
5.1 Available Coding-Based Multicast Schemes

In this section, we briefly review two available coding-based schemes proposed in [47, 88] for the reliable link-layer multicast and also their limitations.

5.1.1 Available Static Scheme and Dynamic Scheme

To achieve the reliable link-layer multicast, traditionally the source simply retransmits the lost packets one by one. Rather than one by one retransmission of lost packets, the basic idea of the coding-based schemes is to first buffer the lost packets for some time and then encode multiple lost packets together into one new packet for retransmission, such that multiple lost packets can be delivered via one retransmission. In detail, two available coding-based schemes are as follows.

Static scheme: This coding-based scheme consists of a *transmission phase* and a *retransmission phase*. In the transmission phase, the source R_0 transmits a fixed number of N packets one by one to M receivers, and stores the lost packets to a buffer of size N (called *lost-packet buffer* in this chapter). The R_0 also maintains a table whose entry



(a) An example XORing lost packets together for retransmission.

(b) An example of no coding chance when using the available coding-based schemes.

Figure 5.2: Examples of network coding-based reliable multicast.

$e_{i,j}$ is used to indicate whether the receiver R_i has correctly received P_j or not, as shown in Figure 5.1. Here, $e_{i,j} = 0$ means that R_i correctly received P_j and $e_{i,j} = 1$ means that R_i did not correctly receive P_j yet. In the retransmission phase, the R_0 first finds the optimal set of lost packets (in terms of the number of lost packets) without common intended receivers to XOR and then repeatedly transmits this XOR-ed packet until all its intended receivers receive it. After finishing the transmission of the current set of lost packets, the R_0 continues to find a new optimal set of lost packets and repeats the above operation. In this way, the source keeps sending out the encoded packets until no lost packet is on the list, and then starts the transmission of next N packets.

Dynamic scheme: Different from the static scheme, in this scheme the source R_0 will update (i.e. to find) the optimal set of lost packets for XORing once the last XOR-ed packet is received by one or more intended receivers (i.e. once the packet-loss table is changed), such that lost packets can be delivered to their intended receivers in a more efficient way.

Let us take the example in Figure 5.2(a) to illustrate how these two schemes work. In this example, both lost packets P_1 and P_4 have one intended receiver R_1 and both lost packets P_3 and P_5 have one intended receiver R_2. Traditionally, each one of P_1, P_3, P_4 and P_5 is retransmitted alone and each one of them has only one intended receiver. When using the above static scheme, the source can XOR P_1 and P_3 together to $P_C = P_1 \oplus P_3$, which has two intended receivers R_1 and R_2 (i.e. $P_1 \oplus P_3$ is useful for R_1 and R_2). Once R_1 receives P_C, it can recover P_1 by $P_1 = P_C \oplus P_3$. Similarly, R_2 can recover P_3 by $P_3 = P_C \oplus P_1$. The source repeatedly transmits P_C until both R_1 and R_2 receive it, and then starts the transmission of next group of lost packets $\{P_4, P_5\}$. When using the available dynamic scheme, however, the source dynamically changes the lost packets for coding. Suppose $P_1 \oplus P_3$ is transmitted and only received by R_1, then the source will XOR P_3 and P_4 together for next transmission. From this example we can know that, by XORing lost packets together to increase the average number of intended receivers per packet, the number of retransmissions can be effectively reduced.

5.1.2 Limitations

Despite the lower bandwidth requirements than the traditional non-coding scheme, both two available coding-based schemes actually suffer from the following two limitations. First, the coding principle that only the lost packets with distinct intended receivers can be XORed together, is too strict to fully explore the potential coding opportunities, since the lost packets with same intended receivers also have the potential to be encoded together for transmission efficiency improvement. For example, for the pattern of lost packets in Figure 5.2(b), there does not exist any coding chance when using available coding-based scheme, because any two lost packets have a common intended receiver. For the static scheme, P_1, P_2 and P_3 will be retransmitted one by one, same as the non-coding scheme. For the dynamic scheme, P_1 will be retransmitted first. If P_1 is received by R_1 and R_3 simultaneously, then the source continues to retransmit P_2. Only if P_1 is received by one of R_1 and R_3, the source can XOR P_1 with another packet. No matter using the static one or the dynamic one, the source needs to retransmit at least three times in this example. However, by adopting a new coding principle to be discussed in Section 5.2.1, these lost packets can actually be transmitted within a fewer retransmissions.

Second, in both two available schemes, finding the maximum set of lost packets with distinct intended receivers is actually a very complex problem, which will significantly limit its scalability. Let L be the number of lost packets. Without loss of generality, assume that $P_1, P_2, \ldots, P_L$ are lost packets. Then, this optimization problem can be mathematically formulated as follows.

Given: values of $e_{i,j}$'s: $i \in \{1, \ldots, M\},\ j \in \{1, \ldots, L\}$.

Encoded packet: $P = a_1 P_1 \oplus \cdots \oplus a_L P_L$

Maximize: $\sum_{i=1}^{L} a_i$

Over variables: $a_i \in \{0, 1\} : 1 \leq i \leq L$

Subject to: $\sum_{i=1}^{L} a_i e_{1,i} \leq 1,$

$\sum_{i=1}^{L} a_i e_{2,i} \leq 1,$

$\ldots$

$\sum_{i=1}^{L} a_i e_{M,i} \leq 1.$

Below, we show that this maximum lost-packet coding (MLPC) problem is NP-complete based on the reduction from the NP-complete maximum independent set (MIS) problem [54].

Theorem 4. *The MLPC problem is NP-complete.*

Proof. It is easy to know that the MLPC problem belongs to NP. Therefore, it is enough to show a polynomial-time reduction from the MIS problem described below to the MLPC problem.

Maximum Independent Set Problem:

Instance: A graph $G(V, E)$ and a positive integer $K \leq |V|$.

Question: Does G contain a subset of vertices with cardinality K such that no two vertices in this subset are adjacent in G?

Here is the reduction. Given an instance $G = (V, E)$ of the MIS problem, construct an instance of the MLPC problem as follows. Label the nodes in G by $v_1, v_2, \cdots, v_{|V|}$. Then the lost packet set is defined as $\{P_1, P_2, \cdots, P_{|V|}\}$, where P_i corresponds to the vertex v_i in the MIS problem. Let $e_{i,j} = 1$ mean that R_i did not correctly receive P_j and $e_{i,j} = 0$ mean that R_i correctly received P_j. At the beginning, set each $e_{i,j}$ to zero and set parameter k to zero. Now, in the order from $i = 1$ to $i = |V|$, we define the receivers which do not correctly receive P_i in the following way: corresponding to each v_i's neighbor v_j with

Table 5.1: Main notations employed in Chapter 5

Notation	Meaning
	common notations
R_0	source node.
R_i	receiver i ($i \geq 1$).
M	number of receivers.
p_i	packet delivery ratio of wireless link (R_0, R_i).
N	number of packets of each generation.
N_l	number of lost packets in a generation.
N_r	total number of retransmissions for a generation.
$e_{i,j}$	indicator about whether R_i correctly receives P_j or not. It equals zero if R_i correctly receives P_j; otherwise, it equals one.
$\mathbf{b}(P_C,A)$	coding vector of the encoded packet P_C over packet set A.
	special notations for static scheme
S_p	a set p of lost packets to be encoded together for retransmission.
$N_l^{p,i}$	number of lost packets in the set S_p which are not received at R_i.
$N_r^{p,i}$	number of retransmissions until R_i receive exactly $N_l^{p,i}$ packets, during the retransmission of lost packets in S_p.
N_r^p	total number of retransmissions for a set S_p of lost packets.
	special notations for dynamic scheme
S	set of lost packets in a generation and $N_l = \|S\|$.
S_d	set of lost packets to be encoded for the current retransmission.
V_i	set of coding vectors for the packets that have already been received by R_i.
N_t	total number of transmissions (including retransmissions) for a generation.

$j > i$, let $k = k + 1$, $e_{k,i} = 1$ and $e_{k,j} = 1$.

Based on the above construction, we can know that the answer to the instance of the MIS problem is "YES" iff there is a set of K lost packets from different receivers in the MLPC problem. □

5.2 New Network Coding-Based Multicast Schemes

In this section, we present improved schemes for the reliable link-layer multicast. The main idea of the improved schemes is to first relax the coding constraint that only lost packets with distinct intended receivers can be encoded together, such that the potential coding opportunities can be fully exploited. Then adopt a simple and polynomial-time algorithm to select the set of lost packets for encoding. By applying these improvements to the available schemes, we can get the corresponding improved static and dynamic ones.

Main notations employed in the proposed schemes and the performance analysis in

Section 5.3 are summarized in Table 5.1.

5.2.1 Improved Static Scheme

Same as the available coding-based schemes, this scheme also consists of the *transmission phase* and *retransmission phase*. The transmission phase of this scheme is the same as the old one, in which the source just simply transmits a fixed number of packets one by one. All these packets are called a *generation* in this chapter.

During the retransmission phase, rather than using a complex (NP-complete) algorithm to find the optimal set of lost packets for coding as the old scheme does, here we first adopt a simple approach to group all lost packets into different sets. Then, during the retransmission of each set of lost packets, we use a novel approach to determine the proper combination of these lost packets for an efficient retransmission of them. Basically, the retransmission phase of our new scheme involves the following several main procedures.

At the beginning of retransmission phase, the source first conducts the following operation.

Procedure 1 (Lost packets grouping): Suppose N_l packets are lost in the current generation. We group these N_l lost packets into $\lfloor \frac{N_l}{M} \rfloor + 1$ sets,[2] such that $\lfloor \frac{N_l}{M} \rfloor$ sets have the same cardinality M and the last set has cardinality (N_l mod M). For the last set with cardinality (N_l mod M), add additional $M - (N_l \bmod M)$ packets with only bits zero into this set, and also set all indicators $e_{i,j}$ of these additional packets as zero.

Unlike the available static schemes where only lost packets with distinct intended receivers will be encoded together, here all lost packets in the same set are encoded together for retransmission, no matter whether these packets have common intended receivers or not. In this way, the potential coding opportunities can be exploited more efficiently. Let us still consider the example in Figure 5.2(b). When using the improved static scheme, the source groups lost packets P_1, P_2 and P_3 into a set, so it now can transmit the encoded packets $P_1 + P_2 + P_3$ and also $P_1 + \alpha P_2 + \alpha^2 P_3$ built over the finite field $\mathbb{F}_{2^2} = \{0, 1, \alpha, \alpha^2\}$.

[2]Without loss of generality, we suppose that (N_l mod M) is not equal to zero.

In this way, it is possible to finish the transmission within two times rather than at least three times as in the old scheme.

For a set of native packets $A = \{P_1, \dots, P_k\}$ (i.e. the packets without encoding) and one of its encoded packet $P_C = \sum_{i=1}^{k} g_i P_i$ over a finite field $\mathbb{F}_q$ with the base q (i.e., $g_i \in \mathbb{F}_q$), we call $(g_1, \dots, g_k)$ as P_C's *coding vector* over A, and denote it by $\mathbf{b}(P_C, A)$. Thus, the main problem now is the selection of coding vector $(g_1, \dots, g_k)$ for each retransmission. Before retransmitting each set of lost packets, the source needs to first conduct the following parameters initialization.

Procedure 2 (Parameters initialization): For a given set S_p of lost packets, let $N_l^{p,i}$ be the number of packets in S_p which R_i has not received yet. Initialize the value of $N_l^{p,i}$ by $N_l^{p,i} = \sum_{P_j \in S_p} e_{i,j}, \forall i \in \{1, \dots, M\}$. Also, initialize the set V of coding vectors as $V = V_{M,q} \backslash \{(0^{i-1}, 1, 0^{M-i}) : i \in \{1, \dots, M\}$ and i-th lost packet in S_p has been received by at least one receiver$\}$, where $V_{M,q}$ is the maximum set of M-dimensional vectors over finite field $\mathbb{F}_q$, which contains M distinct unit vectors $(1, 0, \dots, 0), \dots (0, 0, \dots, 1)$ and any M vectors of it are linearly independent. The construction of $V_{M,q}$ has been widely studied in the field of the systematic maximum-distance separable (MDS) codes[89, 90].

After the above parameters initialization, now the source can select the coding vector for each retransmission.

Procedure 3 (Coding vector selection): Randomly selects an vector $\mathbf{v}$ in V and let $V \leftarrow V \backslash \{\mathbf{v}\}$. Then let vector $\mathbf{v}$ be a coding vector over S_p to obtain an encoded packet.

A native or encoded packet received by a network node is said to be *non-innovative* (*innovative*) for this node if this packet is available or can be (not available and cannot be) generated by linear combination of its previously received packets. Thus, the receiver R_i needs to receive at least $N_l^{p,i}$ innovative packets to recover all its lost packets in S_p. During the retransmission for lost packets in S_p, a receiver R_i that has not received $N_l^{p,i}$ innovative packets is said to be *unsaturated*. Note that for each unsaturated receiver, the coding vector selected in Procedure 3 is independent of the coding vectors of its previously received packets, i.e., the resulting encoded packet is innovative to it. Clearly, this coding approach minimizes the expected number of retransmissions required for the delivery of

Procedure of the improved static scheme

Steps:

Transmit N native packets one by one and build the packet-loss table.
Conduct Procedure 1 to group N_l lost packets into k sets.
for $i = 1$ to k **do**
  Let S_p be the i'th set of lost packets.
  Conduct Procedure 2 to initialize parameters $N_l^{p,i}$ and V.
  while exist one or more unsaturated receivers (i.e. $\exists i$, $N_l^{p,i} > 0$) **do**
    Conduct Procedure 3 to select a coding vector and obtain an encoded packet P_C.
    Repeatedly transmit packet P_C until at least one unsaturated receiver receives it.
    Conduct Procedure 4 to update parameters $N_l^{p,i}$ and V.
  end while
end for

Figure 5.3: Improved static multicast scheme.

lost packets in S_p.

After retransmitting an encoded packet, the source needs to update the parameters $N_l^{p,i}$ and V as follows according to the feedback from the receivers.

Procedure 4 (Parameters update): For each unsaturated receiver R_i (with $N_l^{p,i} \geq 1$), if it correctly receives P_C, $N_l^{p,i} \leftarrow N_l^{p,i} - 1$. For each packet P_j of $S_p \bigcup \{$ transmitted encoded packets from $S_p\}$, if $\sum_{i:N_l^{p,i} \geq 1} e_{i,j} = 0$, then the coding vector of P_j can be reused and thus $V \leftarrow V \bigcup \{\mathbf{b}(P_j, S_p)\}$.

Summarizing the above procedures, the new static scheme is formally illustrated in Figure 5.3.

Next, we discuss the necessary size of field $\mathbb{F}_q$ and also the complexity of this scheme.

Field size

As the number of receivers M increases, the necessary cardinality of the adopted $V_{M,q}$ (and thus the necessary size of $\mathbb{F}_q$) also increases. The following theorem shows the sufficient and necessary condition on the required field size.

Theorem 5. *For a given number of receivers M, the proposed static scheme can always*

	P_1	P_2	P_3	P_{C_1}	P_{C_2}	P_{C_3}
R_1	○	○	×	×	×	×
R_2	×	×	×	○	○	×
R_3	×	×	○	×	×	○

Figure 5.4: A packet-loss example needing the maximum number of innovative packets.

guarantee a innovative packet for all unsaturated receivers if and only if q satisfies $|V_{M,q}| > M(M-1)$.

Proof. The maximum $V_{M,q}$ is needed when the following worst case happens: each transmitted packet is received by exactly one receiver and each receiver has received $M-1$ packets, as illustrated in Figure 5.4. In this worst case, $M(M-1)$ innovative packets have already been transmitted out. If we have one more packet innovative to all receivers to transmit, then once a receiver R_i receives this innovative packet, R_i can recover all M packets and does not need to be considered any more. Then any packet previously received by R_i can be used for retransmission, which is innovative to all remaining unsaturated receivers. □

Computational complexity

Here, we briefly analyze the computational complexity of obtaining an encoded packet for transmission when using the new static scheme. During the transmission phase, the source just transmits a native packet, which takes only constant time. During the retransmission phase, the source first needs time $O(M^2)$ to get S_p and calculate $N_l^{p,i}$. Then, for each retransmission, the source takes time $O(M^2)$ to linearly combine M lost packets, takes time $O(M)$ to update parameter $N_l^{p,i}$ and takes time $O(M^3)$ to update parameter V. Thus, the overall computational complexity of obtaining an encoded packet for retransmission is $O(M^3)$.

5.2.2 Improved Dynamic Scheme

The new dynamic scheme also consists of the transmission phase and retransmission phase. Similar to the improved static scheme, the improved dynamic scheme also relaxes the restrict coding principle and uses a simple algorithm to find the set of lost packets for encoding. The main difference between them is that in the improved dynamic scheme, the encoded packet is dynamically updated for each retransmission such that the potential coding opportunities can be exploited more effectively. It is notable, however, that due to the new requirement of the dynamic update of encoded packet, now the main grouping process and also the selection process of coding vector in the retransmission phase become very different, as summarized in the follows.

At the beginning of retransmission phase, the source first conduct the following operation.

Procedure 1 (Parameters initialization): Let S be the set of lost packets in the current generation, S_d be the set of packets to be encoded for the current retransmission, and V_i be the set of coding vectors of the encoded packets that are already received by R_i. Initialize S_d and V_i $(i = 1, \ldots, M)$ as the empty set.

At each retransmission, we need to determine the set S_d and also the coding vector over S_d to get the encoded packet.

Procedure 2 (Determination of S_d): For each receiver R_i, check whether its $|V_i|$ is equal to $|S_d|$. If we cannot find a R_i with $|V_i|$ equaling $|S_d|$, the S_d for the current transmission keeps unchanged, just same as last transmission. Otherwise, the source updates S_d for the current transmission by removing some packets from and adding some packets into it as follows.

- **Updating** $e_{i,j}$: For each receiver R_i with $|V_i|$ equaling $|S_d|$ and each $P_k \in S_d$, set $e_{i,k}$ as 0 since R_i has already recovered all lost packets in S_d.

- **Packet-removing**: For any packet $P_j \in S_d$ satisfying $\sum_{k=1}^{M} e_{k,j} = 0$, first conduct the following coding vector update: for each R_i and each vector $\mathbf{v} = (v_1, \ldots, v_{|S_d|}) \in V_i$, remove from $\mathbf{v}$ the entry corresponding to packet P_j and if the resulting $\mathbf{v} = \mathbf{0}$, let $V_i \leftarrow V_i \backslash \{\mathbf{v}\}$. Second, remove this packet from S_d.

- **Packet-adding**: For each packet P_n in S, conduct the following operations: check whether there exists at least one receiver R_i satisfying $\sum_{k:P_k \in S_d} e_{i,k} = 0$ and $e_{i,n} = 1$. If so, first add packet P_n into S_d and remove P_n from S; then for each R_j and each $\mathbf{v} = (v_1, \ldots, v_{|S_d|-1}) \in V_j$, add a new entry of zero at the end of $\mathbf{v}$ and if $e_{j,n} = 0$ add the $|S_d|$-dimensional unit vector $(0, \ldots, 0, 1)$ into V_i.

With the set S_d, the determination of coding vector over S_d is done as follows.

Procedure 3 (Determination of coding vector): First, for each receiver R_i with $|V_i| < |S_d|$, obtain a vector $\mathbf{b}_i$ which is independent of V_i by using the Gaussian elimination method and generate an orthogonal set V_i' through orthogonalizing the vectors of V_i. Then, for each obtained vector $\mathbf{b}_i$, $\mathbf{y}_i \leftarrow \mathbf{b}_i - \sum_{\mathbf{v} \in V_i'} \frac{<\mathbf{b}_i, \mathbf{v}>}{||\mathbf{v}||} \mathbf{v}$. Finally, with the obtained $\mathbf{y}_i$, we can use the approach introduced in Lemma 7 of [20] to obtain a coding vector $\mathbf{y}'$ which satisfies that $\mathbf{y}' \cdot \mathbf{y} \neq 0$ for each $\mathbf{y}$.

The following lemma shows that the obtained $\mathbf{y}'$ is linearly independent of each V_i.

Lemma 3. *Let B denote a set of n-dimensional vectors. Vector $\boldsymbol{a}$ is orthogonal to B, i.e., $\boldsymbol{a} \cdot \boldsymbol{b} = 0$ for any vector $\boldsymbol{b} \in B$. Then if $\boldsymbol{x} \cdot \boldsymbol{a} \neq 0$, $\boldsymbol{x}$ is linearly independent of B.*

Procedure 3 guarantees that the selected coding vector is independent of the coding vectors for the received packets of this receiver. Clearly, this dynamic coding way can minimize the average number of retransmissions per generation.

Formally, the new dynamic scheme is shown in Figure 5.5. Next, we briefly discuss the necessary field size and the computational complexity of this scheme.

Field size

The following lemma (from Lemma 6 in [20]) and corollary show a sufficient condition on the necessary size of the field $\mathbb{F}_q$.

Lemma 4. *Let $\mathbb{F}^h$ be the space of h-dimensional vectors over $\mathbb{F}$. If $|\mathbb{F}| \geq n$ and vector pairs $(\boldsymbol{x}_i, \boldsymbol{y}_i) \in \mathbb{F}^h \times \mathbb{F}^h$ satisfy $\boldsymbol{x}_i \cdot \boldsymbol{y}_i \neq 0$ for each $i \in \{1, \ldots, n\}$, then there is a linear combination $\boldsymbol{u}$ of $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n$ such that $\boldsymbol{u} \cdot \boldsymbol{y}_i \neq 0$ for each i.*

Procedure of the improved dynamic scheme
Steps:

Transmit N native packets one by one and build the packet-loss table.
Conduct Procedure 1 to initialize parameters S, S_d and V_i ($i = 1, \ldots, M$).
while $S \neq \phi$ and $S_d \neq \phi$ **do**
  Conduct Procedure 2 to update S_d.
  Conduct Procedure 3 to obtain $\mathbf{y}'$ which is independent of each V_i satisfying $|V_i| < |S_d|$, and obtain the
    encoded packet P_C.
  Repeatedly transmit packet P_C until one or more receivers receive it.
  For any R_i which correctly receives P_C, $V_i \leftarrow V_i \bigcup \{\mathbf{y}'\}$.
end while

Figure 5.5: Improved dynamic multicast scheme.

Corollary 1. *If $|\mathbb{F}| \geq M$ and $n \leq M$, then for vectors $(\boldsymbol{y}_1, \boldsymbol{y}_1), \ldots, (\boldsymbol{y}_n, \boldsymbol{y}_n) \in \mathbb{F}^h \times \mathbb{F}^h$ there is a linear combination $\boldsymbol{y}'$ of $\boldsymbol{y}_1, \ldots, \boldsymbol{y}_n$ such that $\boldsymbol{y}' \cdot \boldsymbol{y}_i \neq 0$ for each i.*

Similarly to the new static scheme, this new dynamic scheme conducts coding operation over a general finite field $\mathbb{F}_q$ rather than over $\mathbb{F}_2$. The following theorem shows the sufficient condition on the size of $\mathbb{F}_q$ for this scheme.

Theorem 6. *Given the value of M, if $q \geq M$, then in the new dynamic scheme we always have a packet innovative to all unsaturated receivers for retransmission.*

Proof. Based on Corollary 1, we can easily arrive at the result. □

Computational complexity

Here, we analyze the computational complexity of obtaining a packet for transmission when using the new dynamic scheme. During the transmission phase, the source just transmits a native packet, which takes only constant time. During Procedure 2 of the retransmission phase, updating $e_{i,j}$ takes time $O(MN)$, removing packets from S_d and updating related parameters take time $O(MN^3)$, and adding packets into S_d and updating related parameters take time $O(M^2N^3)$. In Procedure 3, Gaussian elimination, Gram-Schmidt orthonormalization process and the calculation of $\mathbf{y}'$ take time $O(MN^3)$, $O(MN^3)$ and $O(MN^2)$, respectively. Thus, the overall computational complexity is $O(M^2N^3)$.

5.2.3 Decoding at The Receivers

When a receiver has received M encoded packets, denoted by $\alpha_{1,1}P_1 + \alpha_{1,2}P_2 + \ldots + \alpha_{1,M}P_M$, ..., $\alpha_{M,1}P_1 + \alpha_{M,2}P_2 + \ldots + \alpha_{M,M}P_M$ where vectors $(\alpha_{1,1}, \alpha_{1,2}, \ldots, \alpha_{1,M})$, ..., $(\alpha_{M,1}, \alpha_{M,2}, \ldots, \alpha_{M,M})$ are independent, this receiver can retrieve native packets $P_1, P_2, \ldots, P_M$ by the following decoding:

$$\begin{bmatrix} P_1 \\ P_2 \\ \vdots \\ P_M \end{bmatrix} = \begin{bmatrix} \alpha_{1,1} & \alpha_{1,2} & \ldots & \alpha_{1,M} \\ \alpha_{2,1} & \alpha_{2,2} & \ldots & \alpha_{2,M} \\ & & \vdots & \\ \alpha_{M,1} & \alpha_{M,2} & \ldots & \alpha_{M,M} \end{bmatrix}^{-1} \begin{bmatrix} \alpha_{1,1}P_1 + \alpha_{1,2}P_2 + \ldots + \alpha_{1,M}P_M \\ \alpha_{2,1}P_1 + \alpha_{2,2}P_2 + \ldots + \alpha_{2,M}P_M \\ \vdots \\ \alpha_{M,1}P_1 + \alpha_{M,2}P_2 + \ldots + \alpha_{M,M}P_M \end{bmatrix} \tag{5.1}$$

Let us take an example to see how the native packets are retrieved. Suppose receiver R_1 has received three encoded packets: P_2, $P_1 + P_2 + P_3$ and $P_1 + \alpha P_2 + \alpha^2 P_3$. Then R_1 conducts the following linear operations:

$$\begin{aligned} \begin{bmatrix} 0 & 1 & 0 \\ 1 & 1 & 1 \\ 1 & \alpha & \alpha^2 \end{bmatrix}^{-1} \begin{bmatrix} P_2 \\ P_1 + P_2 + P_3 \\ P_1 + \alpha P_2 + \alpha^2 P_3 \end{bmatrix} &= \begin{bmatrix} 0 & 1 & 0 \\ 1 & 1 & 1 \\ 1 & \alpha & \alpha^2 \end{bmatrix}^{-1} \begin{bmatrix} 0 & 1 & 0 \\ 1 & 1 & 1 \\ 1 & \alpha & \alpha^2 \end{bmatrix} \begin{bmatrix} P_2 \\ P_2 \\ P_3 \end{bmatrix} \\ &= \begin{bmatrix} P_2 \\ P_2 \\ P_3 \end{bmatrix} \end{aligned} \tag{5.2}$$

Thus, R_1 retrieves P_1,P_2 and P_3 throughput the above operations.

5.2.4 Discussion

In both the original literature and our above work on the coding-based reliable multicast, it is assumed that the multicast group keeps unchanged during the whole transmission period. In many practical cases, however, individual clients may join and leave multicasting sessions dynamically. For these dynamic cases, it is necessary for the multicast scheme to deal with dynamic memberships in multicast groups. How to design the adaptive coding-based scheme which can cope with the group membership dynamics and also obtain coding opportunities as many as possible, can be an important extension of the current work.

In addition, this work only considers the one-hop multicast (i.e. the link-layer multicast), which can be applied in the WLAN (from the access point to a set of users), the cellular system (from the base station to a set of users), etc. In the multihop wireless networks, however, a multicast tree has a number of hops. Thus, it is also worth much effort to design the coding-based multihop multicast scheme. Because the receivers $R_1, R_2, \ldots, R_M$ of a link-layer multicast (which is one intermediate hop of the multicast tree) are the source nodes of those subsequent hops in the multicast tree, these receivers will transmit packets to their respective children nodes. The packets transmitted by $R_1, R_2, \ldots, R_M$ will be overheard by one another among $R_1, R_2, \ldots, R_M$. Therefore, this characteristic of the multihop multicast can be take used in the coding-based multihop multicast scheme to further reduce the transmissions, as compared with the way of just applying the above proposed one-hop multicast scheme at each intermediate hop of the multicast tree.

5.3 Performance Analysis

In this section, we conduct the theoretical analysis of improved schemes in terms of the transmission efficiency and the delay performance. By transmission efficiency, the same metric called *transmission bandwidth* as in [47, 88] is adopted, which is defined as the average number of transmissions required to successfully transmit a packet to all receivers. By delay performance, we will evaluate the average number of transmissions a packet needs to wait from when this packet is transmitted for the first time until it is successfully received by all receivers (referred as *retransmission delay* in this chapter).

5.3.1 Analysis of Improved Static Scheme

We first provide the analysis for the improved static scheme.

Transmission Bandwidth

Denote by η_g the transmission bandwidth when using the proposed improved static scheme and by N_r the number of retransmission packets for a generation of lost packets. Then η_g is given by

$$\begin{aligned}\eta_g &= E[(N+N_r)/N] \\ &= 1+\frac{1}{N}E[N_r]. \\ &= 1+\frac{1}{N}\sum_{L=0}^{N} P[N_l=L]E[N_r|N_l=L],\end{aligned} \tag{5.3}$$

where N_l the total number of lost packets among a generation of packets.

In the above equation, under the assumption that the packet loss probabilities of different links are independent from one another, the $P[N_l = L]$ can be easily evaluated by

$$P[N_l=L] = \binom{N}{L}\left(1-\prod_{n=1}^{M} p_n\right)^{L}\left(\prod_{n=1}^{M} p_n\right)^{N-L}, \tag{5.4}$$

where p_n is the packet delivery ratio of wireless link (R_0, R_n) and $\prod_{n=1}^{M} p_n$ is the probability that a packet is successfully received by all receivers.

We now analyze the conditional expected number of retransmissions $E[N_r|N_l = L]$ in Equation (5.3). In the static scheme, L lost packets are grouped into $\left\lfloor \frac{L}{M} \right\rfloor + 1$ sets of lost packets, $\left\lfloor \frac{L}{M} \right\rfloor$ sets with cardinality M and one set with cardinality $L \bmod M$. Since the sets with the same cardinality M have the same expected number of retransmissions, so $E[N_r|N_l = L]$ is given by:

$$E[N_r|N_l=L] = \left\lfloor \frac{L}{M} \right\rfloor E[N_r^p \big| |S_p| = M] + E[N_r^p \big| |S_p| = L \bmod M], \tag{5.5}$$

where S_p denotes the set of lost packets which are encoded together for retransmission and N_r^p is the number of retransmission packets for S_p.

So far, the work left for evaluating η_g is the calculation of $E[N_r^p \big| |S_p|]$. It is given by the following formula:

$$\begin{aligned} E[N_r^p \big| |S_p|] &= \sum_{i=1}^{\infty} i \cdot P(N_r^p = i \big| |S_p|) \\ &= \sum_{i=1}^{\infty} i \cdot (P(N_r^p \le i \big| |S_p|) - P(N_r^p \le i-1 \big| |S_p|)). \end{aligned} \tag{5.6}$$

Denote by $N_l^{p,i}$ the number of unreceived packets at R_i in a set S_p of lost packets. For a set of lost packets, the number of retransmission packets N_r^p is

$$N_r^p = \max_{j \in \{1,\ldots,M\}} N_r^{p,j},$$

where $N_r^{p,j}$ is a random variable denoting the number of transmissions required for R_j to receive $N_l^{p,j}$ packets. Then we have

$$\begin{aligned} & P[N_r^p \le i \big| |S_p|] \\ &= P(N_r^{p,1} \le i, \ldots, N_r^{p,M} \le i \big| |S_p|) \\ &= \sum_{\substack{0 \le i_1,\ldots,i_M \le \min\{i,k\} \\ i_1+\cdots+i_M \ge k}} P(N_l^{p,1} = i_1, \ldots, N_l^{p,M} = i_M, N_r^{p,1} \le i, \ldots, N_r^{p,M} \le i \big| |S_p|) \\ &= \sum_{\substack{0 \le i_1,\ldots,i_M \le \min\{i,k\} \\ i_1+\cdots+i_M \ge k}} P(N_l^{p,1} = i_1, \ldots, N_l^{p,M} = i_M \big| |S_p|) \\ & \quad \cdot P(N_r^{p,1} \le i, \ldots, N_r^{p,M} \le i \big| |S_p|, N_l^{p,1} = i_1, \ldots, N_l^{p,M} = i_M), \ i = 1, 2, \ldots. \end{aligned} \tag{5.7}$$

The second term in the above equation can be evaluated as follows:

$$\begin{aligned} & P(N_r^{p,1} \le i, \ldots, N_r^{p,M} \le i \big| |S_p|, N_l^{p,1} = i_1, \ldots, N_l^{p,M} = i_M) \\ &= P(N_r^{p,1} \le i | N_l^{p,1} = i_1) P(N_r^{p,2} \le i | N_l^{p,2} = i_2) \cdots P(N_r^{p,M} \le i | N_l^{p,M} = i_M) \\ &= \prod_{j=1}^{M} \sum_{k=i_j}^{i} \binom{k-1}{k-i_j} p_j^{i_j} (1-p_j)^{k-i_j} \end{aligned} \tag{5.8}$$

About the evaluation of $P(N_l^{p,1} = i_1, \ldots, N_l^{p,M} = i_M \big| |S_p|)$ in Equation (5.7), we have the following lemma.

Lemma 5. *For k packets, given that each of them is not correctly received by at least one receiver, the probability that R_n $(n = 1, \ldots, M)$ did not correctly receiver i_n packets among these k packets is given by*

$$
\begin{aligned}
& P(N_l^{p,1} = i_1, \ldots, N_l^{p,M} = i_M \big| |S_p|) \\
&= \left(\prod_{n=1}^{M} p_n^{k-i_n} (1-p_n)^{i_n} \right) \cdot \left(\sum_{l=0}^{k-\max_{n\in\{1,\ldots,M\}} i_n} (-1)^l \binom{k}{l} \prod_{n=1}^{M} \binom{k-l}{i_n} \right) / (1 - \prod_{n=1}^{M} p_n)^k
\end{aligned} \tag{5.9}
$$

Proof. $P(N_l^{p,1} = i_1, \ldots, N_l^{p,M} = i_M \big| |S_p| = k)$ will be evaluated by

$$
\begin{aligned}
& P(N_l^{p,1} = i_1, \ldots, N_l^{p,M} = i_M \big| |S_p| = k) \\
&= P(N_l^{p,1} = i_1, \ldots, N_l^{p,M} = i_M) P(|S_p| = k \big| N_l^{p,1} = i_1, \ldots, N_l^{p,M} = i_M) / P(|S_p| = k)
\end{aligned} \tag{5.10}
$$

Among k packets, the probability that $R_1, \ldots R_M$ fail to receive $i_1, \ldots, i_M$ packets, respectively, is given by

$$
P[N_l^{p,1} = i_1, \ldots, N_l^{p,M} = i_M] = \prod_{n=1}^{M} \binom{k}{i_n} p_n^{k-i_n} (1-p_n)^{i_n}. \tag{5.11}
$$

For k packets, the probability that each packet is not correctly received by at least one receiver is given by

$$
P(|S_p| = k) = \left(1 - \prod_{n=1}^{M} p_n \right)^k. \tag{5.12}
$$

Now, we will evaluate $P(|S_p| = k \big| N_l^{p,1} = i_1, \ldots, N_l^{p,M} = i_M)$. Clearly, the total number of patterns of lost packets, which satisfy that $N_l^{p,n} = i_n$, $n = 1, \ldots, M$, is $\prod_{n=1}^{M} \binom{k}{i_n}$ (all these patterns happen with the same probability). We proceed to calculate the number N_{epl} of patterns satisfying that $N_l^{p,n} = i_n$ $(n = 1, \ldots, M)$ and each packet is lost at one or more receivers (i.e. $|S_p| = k$). Then

$$N_{epl} = \prod_{n=1}^{M} \binom{k}{i_n} - \Big| A_1 \bigcup A_2 \bigcup \cdots \bigcup A_N \Big|, \tag{5.13}$$

where A_i ($i = 1, \ldots, N$) denotes the set of patterns satisfying that $N_l^{p,n} = i_n$ for $n = 1, \ldots, M$ and packet P_i is received by all receivers.

According to the inclusion-exclusion principle, we have

$$\begin{aligned}
&|A_1 \cup A_2 \cup \cdots \cup A_k| \\
=&(|A_1|+|A_2|+\cdots+|A_k|)-(|A_1 \cap A_2|+|A_1 \cap A_3|+\cdots+|A_{k-1} \cap A_k|)+(|A_1 \cap A_2 \cap A_3| \\
&+|A_1 \cap A_2 \cap A_4| + \cdots + |A_{k-2} \cap A_{k-1} \cap A_k|) + \cdots + (-1)^{k-1}|A_1 \cap A_2 \cap \cdots \cap A_k| \\
=&(-1)^0 \binom{k}{1} \prod_{n=1}^{M} \binom{k-1}{i_n} + (-1)^1 \binom{k}{2} \prod_{n=1}^{M} \binom{k-2}{i_n} + \cdots + \\
&(-1)^{k-\max_{n\in\{1,\ldots,M\}} i_n} \binom{k}{k-\max_{n\in\{1,\ldots,M\}} i_n} \prod_{n=1}^{M} \binom{\max_{n\in\{1,\ldots,M\}} i_n}{i_n} + 0 + \cdots + 0.
\end{aligned} \tag{5.14}$$

Then we have

$$\begin{aligned}
&P(|S_p| = k \big| N_l^{p,1} = i_1, \ldots, N_l^{p,M} = i_M) \\
=& \left(\prod_{n=1}^{M} \binom{k}{i_n} - |A_1 \cup A_2 \cup \cdots \cup A_k| \right) / \prod_{n=1}^{M} \binom{k}{i_n} \\
=& \left(\sum_{l=0}^{k-\max_{n\in\{1,\ldots,M\}} i_n} (-1)^l \binom{k}{l} \prod_{n=1}^{M} \binom{k-l}{i_n} \right) / \prod_{n=1}^{M} \binom{k}{i_n}.
\end{aligned} \tag{5.15}$$

Finally, by substituting Equations (5.11), (5.12) and (5.15) into Equation (5.10), we get the result. □

Now, by substituting Equations (5.9) and (5.8) into (5.7) and substituting Equation (5.7) into (5.6), we have

$$E[N_r^p \big| |S_p|] = \sum_{i=1}^{\infty} i(q(i,k) - q(i-1,k)), \tag{5.16}$$

where

$$q(i,k) = \sum_{\substack{0 \le i_1,\dots,i_M \le \min\{i,k\} \\ i_1+\dots+i_M \ge k}} \left(\prod_{j=1}^{M} \sum_{k=i_j}^{i} \binom{k-1}{k-i_j} p_j^{i_j} (1-p_j)^{k-i_j} \right) \cdot \left(\prod_{n=1}^{M} p_n^{k-i_n} (1-p_n)^{i_n} \right)$$
$$\cdot \left(\sum_{l=0}^{k-\max_{n\in\{1,\dots,M\}} i_n} (-1)^l \binom{k}{l} \prod_{n=1}^{M} \binom{k-l}{i_n} \right) / (1 - \prod_{n=1}^{M} p_n)^k. \tag{5.17}$$

Finally, we summarize the evaluation of η_g as the following theorem.

Theorem 7. *The transmission bandwidth η_g of proposed static scheme with M receivers and lost-packet buffer size N is:*

$$\eta_g = 1 + \frac{1}{N} \sum_{L=0}^{N} \Bigg\{ f(\prod_{n=1}^{M} p_n, L, N) \cdot \sum_{i=1}^{\infty} \Bigg(i \cdot \left\lfloor \frac{L}{M} \right\rfloor \Big(q(i,M) - q(i-1,M) \Big)$$
$$+ i \cdot \Big(q(i, L \ mod \ M) - q(i-1, L \ mod \ M) \Big) \Bigg) \Bigg\}, \tag{5.18}$$

where $q(i,k)$ is given by Equation (5.17) and

$$f(p,i,j) = \binom{j}{i} p^{j-i} (1-p)^i. \tag{5.19}$$

Proof. Combining Equations (5.3), (5.4), (5.5) and (5.16), we easily get the result. □

Retransmission Delay

Denote by γ_g the retransmission delay when using the proposed improved static scheme. It is easy to know that the larger the lost-packet buffer size N, the larger the retransmission delay γ_g.

To decode the received encoded packets, every receiver will perform Gaussian elimination after every received innovative packet to ensure the earliest possible decoding. Because different selections of innovative packets for transmission will lead to different results of Gaussian elimination (i.e. different packet delay) at receivers, so the exact analysis of the retransmission delay is quite difficult. Here we present an upper bound on the retransmission delay in the following theorem.

Theorem 8. *The retransmission delay γ_g of the proposed static scheme is upper bounded by*

$$\gamma_g < \frac{1}{N}\sum_{L=0}^{N}\Bigg\{ f(\prod_{n=1}^{M} p_n, L, N)\cdot\left(\frac{(N-1)L}{2} + 0.5\left\lfloor\frac{L}{M}\right\rfloor (M+L+L\%M)\cdot\right.$$
$$\left.\sum_{i=1}^{\infty} i\cdot\Big(q(i,M)-q(i-1,M)\Big) + (L\%M)\sum_{i=1}^{\infty} i\cdot\Big(q(i,L\%M)-q(i-1,L\%M)\Big)\right)\Bigg\}, \quad (5.20)$$

where $q(i,k)$ and $f(p,i,j)$ are shown in Equations (5.17) and (5.19), respectively, and the symbol % represents the integer modulo operation.

Proof. The overall retransmission delay D of a generation of packets are induced only by those lost packets, including the waiting time in the transmission phase and the waiting time in the retransmission phase. Denote by $P_N, P_{N-1}, \ldots, P_1$ the N transmitted packets in turn. If P_n is lost, the waiting time of P_n in the transmission phase is $n-1$. Thus D is given by

$$D = \sum_{P_i\in S} i + \sum_{P_i\in S} D_i. \quad (5.21)$$

where S is the set of lost packets among a generation of packets and D_i is the number of packets transmitted in the retransmission phase until P_i is received by all receivers. Then we have

$$\gamma_g = \frac{1}{N}E[D] = \frac{1}{N}\left(E\left[\sum_{P_i\in S} i\right] + E\left[\sum_{P_i\in S} D_i\right]\right). \quad (5.22)$$

The term $E[\sum_{P_i\in S} i]$ in Equation (5.22) is evaluated by

$$\begin{aligned} E\left[\sum_{P_i\in S} i\right] &= \sum_{L=0}^{N} P[|S|=L]\,E\left[\sum_{P_i\in S} i \,\middle|\, |S|=L\right] \\ &= \sum_{L=0}^{N}\binom{N}{L}\left(1-\prod_{n=1}^{M}p_n\right)^{L}\left(\prod_{n=1}^{M}p_n\right)^{N-L}\frac{(N-1)L}{2}, \end{aligned} \quad (5.23)$$

and $E[\sum_{P_i \in S} D_i]$ in Equation (5.22) is evaluated by

$$E\left[\sum_{P_i \in S} D_i\right] = \sum_{L=0}^{N} \left(P[|S| = L] E\left[\sum_{P_i \in S} D_i ||S| = L\right]\right). \tag{5.24}$$

During the retransmission of a set S_p of lost packets, in the worst case, each receiver R_i receives $N_l^{p,i}$ retransmission packets exactly after the N_r^p'th retransmission, and recovers each one of $N_l^{p,i}$ lost packets exactly when receiving $N_l^{p,i}$ retransmission packets. Thus, the expected overall delay of lost packets of the n'th set is less than $n \cdot E[N_r^p \big| |S_p| = M] \cdot M$. Then

$$\begin{aligned}
&E\left[\sum_{P_i \in S} D_i \Big| |S| = L\right] \\
&< 1 \cdot E[N_r^p \big| |S_p| = M] \cdot M + \cdots + \left\lfloor \frac{L}{M} \right\rfloor \cdot E[N_r^p \big| |S_p| = M] \cdot M \\
&\quad + \left(\left\lfloor \frac{L}{M} \right\rfloor E[N_r^p \big| |S_p| = M] + E[N_r^p \big| |S_p| = L\%M] \right) (L\%M). \\
&= 0.5 E[N_r^p \big| |S_p| = M] \left\lfloor \frac{L}{M} \right\rfloor (M + L + L\%M) + E[N_r^p \big| |S_p| = L\%M](L\%M). && (5.25)
\end{aligned}$$

Finally, combining Equations (5.22)-(5.25) we obtain the result. □

5.3.2 Analysis of Improved Dynamic Scheme

Here we evaluate the transmission bandwidth and retransmission delay of the improved dynamic scheme.

Transmission Bandwidth

Denote by η_d the transmission bandwidth when using the proposed improved dynamic scheme. The transmission efficiency η_d of the proposed dynamic scheme is given in the following theorem.

Theorem 9. *The transmission bandwidth η_d of dynamic scheme with M receivers and* *lost-packet buffer size N is*

$$\eta_d = \frac{1}{N} \sum_{i=N}^{\infty} i \left(\prod_{j=1}^{M} \sum_{k=N}^{i} P_{j,k} - \prod_{j=1}^{M} \sum_{k=N}^{i-1} P_{j,k} \right), \tag{5.26}$$

where $P_{j,k} = \binom{k-1}{N-1} p_j^N (1-p_j)^{k-N}$.

Proof. Let N_i be a random variable denoting the number of transmissions for receiver R_i to successfully receive N packets. Clearly, $N_i \geq N$. Then the total number of transmissions to guarantee that all receivers successfully receive N packets is

$$N_t = \max_{j \in \{1,\ldots,M\}} N_j.$$

The average number of transmissions required to successfully transmit a packet to all receivers is given by

$$\begin{aligned}
\eta_d &= \frac{1}{N} E[N_t] \\
&= \frac{1}{N} \sum_{i=N}^{\infty} i P[N_t = i] \\
&= \frac{1}{N} \sum_{i=N}^{\infty} i (P[N_t \leq i] - P[N_t \leq i-1]) \\
&= \frac{1}{N} \sum_{i=N}^{\infty} i (P[N_1 \leq i, \ldots, N_M \leq i] - P[N_1 \leq i-1, \ldots, N_M \leq i-1]) \\
&= \frac{1}{N} \sum_{i=N}^{\infty} i \left(\prod_{j=1}^{M} P[N_j \leq i] - \prod_{j=1}^{M} P[N_j \leq i-1] \right).
\end{aligned} \tag{5.27}$$

In the above equation, $P[N_j \leq i]$ is given by

$$P[N_j \leq i] = \sum_{k=N}^{i} P[N_j = k] = \sum_{k=N}^{i} \binom{k-1}{N-1} p_j^N (1-p_j)^{k-N}. \tag{5.28}$$

Finally, substituting Equation (5.28) into (5.27), we arrive at the result. □

Retransmission Delay

Denote by γ_d the retransmission delay when using the proposed improved dynamic scheme, respectively. The following theorem shows the delay performance of the proposed dynamic scheme.

Theorem 10. *The retransmission delay γ_d of the proposed dynamic scheme is upper bounded by*

$$\gamma_d < \frac{1}{N}\sum_{L=0}^{N}\left\{ f(\prod_{n=1}^{M} p_n, L, N)\cdot\left(\frac{(N-1)L}{2} + L\sum_{i=1}^{\infty} i\cdot(q(i,M)-q(i-1,M))\right)\right\}, \tag{5.29}$$

where $q(i,k)$ and $f(p,i,j)$ are shown in Equations (5.17) and (5.19), respectively.

Proof. Same as the proposed static scheme, the retransmission delay of the proposed dynamic scheme is given by (see the proof of Theorem 8)

$$\gamma_d = E(D) = \frac{1}{N}\left(E[\sum_{P_i\in S} i] + E[\sum_{P_i\in S} D_i]\right). \tag{5.30}$$

In the above equation, $E[\sum_{P_i\in S} i]$ is already given in Equation (5.23), and $E[\sum_{P_i\in S} D_i]$ is upper bounded by

$$\begin{aligned} E[\sum_{P_i\in S} D_i] &= \sum_{L=0}^{N}\left(P[|S|=L]E[\sum_{P_i\in S} D_i||S|=L]\right) \\ &\leq \sum_{L=0}^{N}\left[\binom{N}{L}\left(1-\prod_{n=1}^{M} p_n\right)^{L}\left(\prod_{n=1}^{M} p_n\right)^{N-L}(L\cdot E[N_r^p||S_p|=L])\right]. \end{aligned} \tag{5.31}$$

Combining Equations (5.16), (5.23), (5.30) and (5.31), we obtain the result. □

5.4 Numerical Results

In this section, we demonstrate the performance of the proposed schemes in terms of transmission efficiency and also the delay. The numerical results provided are obtained from both the analysis and simulation. For comparison, the corresponding results for the

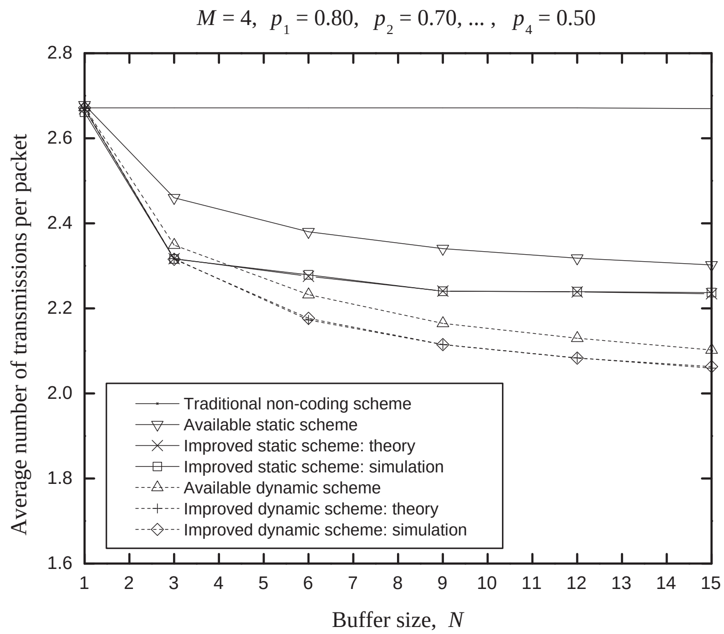


Figure 5.6: Transmission bandwidth versus lost-packet buffer size.

available schemes are also provided.

In the simulation, for each scenario of parameter setting (number M of receivers, size N of the lost-packet buffer and link packet loss probabilities), we simulate the multicast transmission of $10000 * N$ packets based on the non-coding scheme, the available coding-based schemes and the improved schemes, respectively.

5.4.1 Transmission Bandwidth

The transmission bandwidth of all network coding-based schemes greatly depend on the lost-packet buffer size, so we first investigate the transmission bandwidth of different schemes under different sizes of the lost-packet buffer. Figure 5.6 shows the numerical results of different schemes on the transmission bandwidth, where $N = 4$, $p_1 = 0.80$, $p_1 = 0.70$, $p_1 = 0.60$ and $p_4 = 0.50$. We can see that the analytical results on transmission bandwidth nicely match the simulation results, so the proposed models can be used to efficiently investigate the transmission bandwidth of the proposed schemes. From this figure, we can also observe that in general the transmission bandwidth of each network

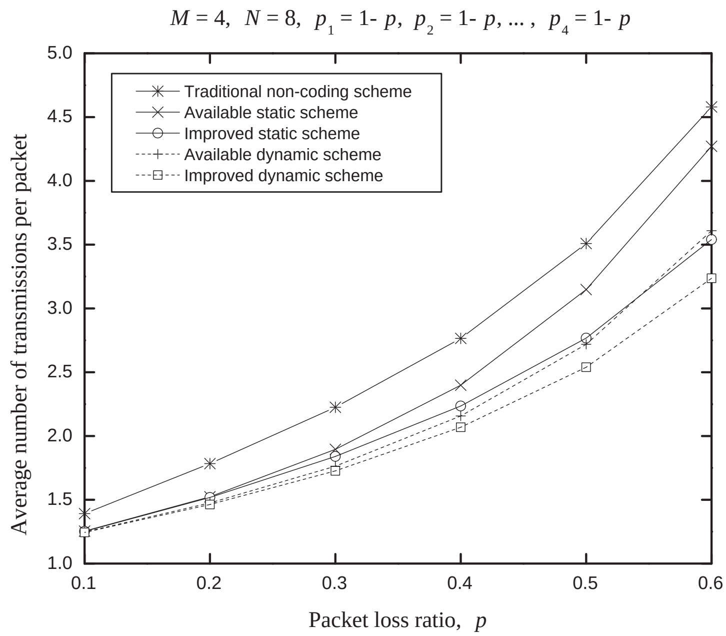


Figure 5.7: Transmission bandwidth versus packet loss probability.

coding-based scheme decreases as the lost-packet buffer size increases, and when the lost-packet buffer size is not very small, the coding-based multicast schemes can substantially outperform the non-coding multicast scheme. For example, for buffer size $N = 9$, compared to the traditional multicast scheme, the average number of transmissions per packet can be reduced by over 16% when using the proposed static scheme.

From Figure 5.6 we can also observed that, compared to the available static scheme, the proposed static scheme can more effectively reduce the transmission bandwidth, especially when the lost-packet buffer size is small. For example, when the buffer size is three, the available static scheme only reduces the bandwidth consumption by 7.9% percent, while this reduction can be 13.3% when using the improved static scheme. Similarly, the proposed dynamic scheme always outperforms the available dynamic scheme. For example, as compared with the available dynamic scheme, the proposed dynamic scheme can further improve the transmission efficiency by 2.2% when $N = 6$. Additionally, results in Figure 5.6 show that the dynamic schemes greatly outperform the static schemes, at the cost of increased computational complexity.

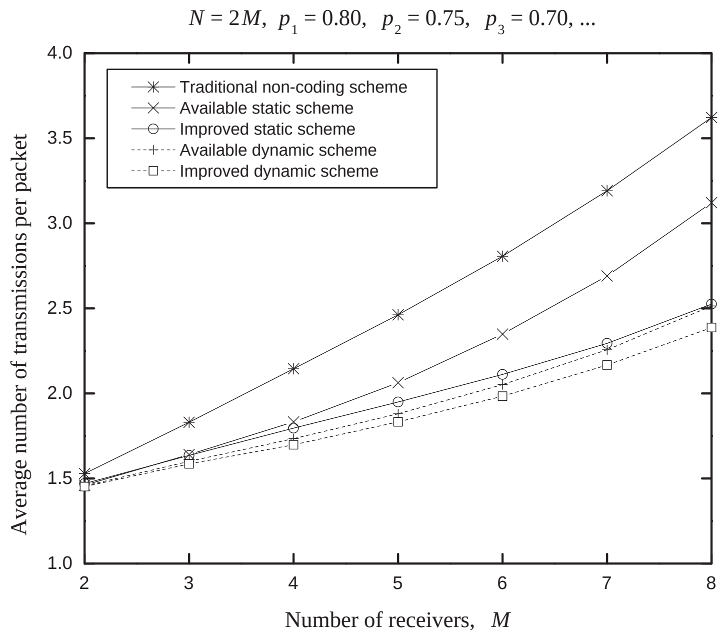
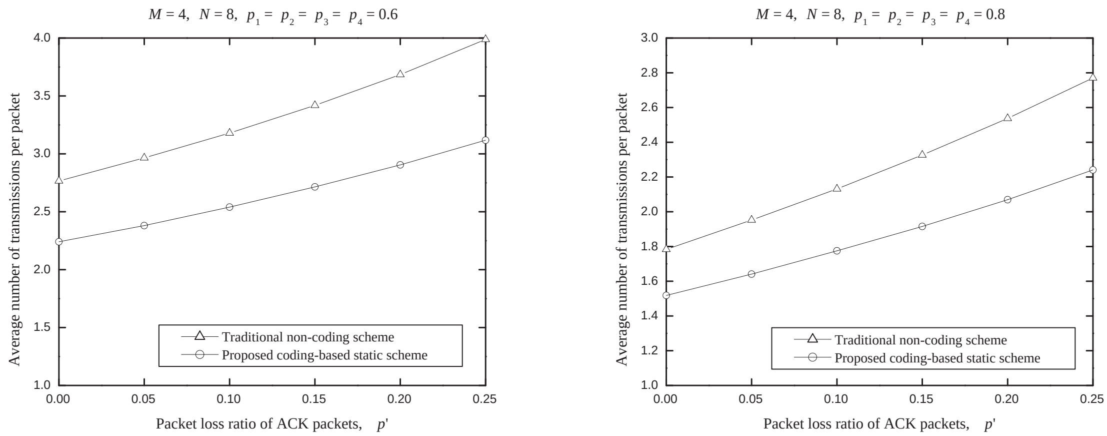


Figure 5.8: Transmission bandwidth versus number of receivers, in the medium packet loss scenario.

We further investigate the transmission bandwidth under different link packet loss probabilities and different numbers of receivers, as summarized in Figure 5.7 and Figure 5.8, respectively. The results in Figure 5.7 show that as the packet loss probabilities increase, the advantage of the improved schemes over the available schemes becomes more significant. For example, when the packet loss probability of each link is 0.5, compared with the non-coding scheme the available static scheme reduces the bandwidth consumption by 10.3%, but the bandwidth consumption achieved by the improved static scheme can be as high as 21.1%. The results in Figure 5.8 show that as compared with the available schemes, the transmission bandwidth reduction achieved by using the proposed schemes increases as the number of receivers increases. For example, for the case of three receivers, both the available and the proposed static schemes reduce the transmission bandwidth by about 10.4%. For the case of six receivers, however, the proposed static scheme can reduce the transmission bandwidth by as high as 24.8%, much higher than the 16.3% achieved by the available static scheme.



(a) In the cases with large packet loss probabilities. (b) In the cases with small packet loss probabilities.

Figure 5.9: Transmission efficiency under different loss ratios of ACK packets.

Effect of the acknowledgment (ACK) packet loss

In the above evaluation of transmission efficiency, it is assumed that the source node can timely know which packets are lost at each receiver. In practical, however, the ACK (or NACK) packets may be lost on the link from a receiver to the source node. In this subsection, we assume the ACK packets are used at receivers to notice the source node which packets are received and evaluate the impact of the ACK packet loss on the transmission efficiency of the proposed coding-based static scheme.

Figure 5.9 shows the transmission efficiency under different loss ratios of ACK packets. It can be observed that, for both the non-coding scheme and the coding-based scheme, as the loss probability of ACK packets increase the average number of transmission per packet increases. This is easy to understand. The loss of ACK packets will cause the unnecessary retransmissions at the source node. Another important conclusion drawn from this figure is that the transmission efficiency improvement achieved by using network coding almost keeps unchanged under different loss ratios of ACK packets.

5.4.2 Retransmission Delay

Since the analytical model for the exact delay analysis is not available, the proposed upper bound model is adopted here to roughly demonstrate the delay behavior of the proposed

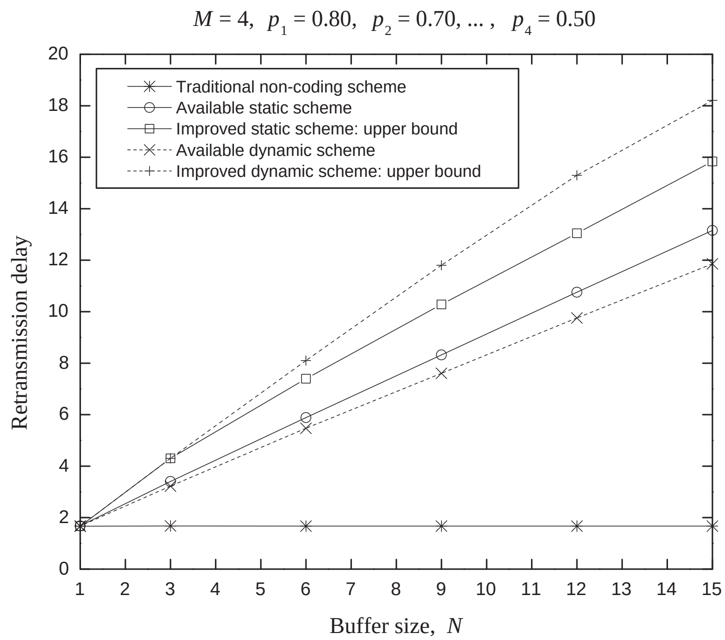


Figure 5.10: Delay versus lost-packet buffer size.

schemes.

Figure 5.10 shows the retransmission delay as a function of the lost-packet buffer size, where we can see that the retransmission delay of coding-based schemes approximately linearly increases as the lost-packet buffer size increases. The reason of this behavior is that during the transmission phase the source buffers the lost packets for future packet coding rather than retransmits them immediately. This delay increment is the cost one needs to pay for acquiring coding opportunities. As discussed previously, the transmission efficiency improvement also steadily increases as the lost-packet buffer size increases. Thus, there is a trade-off between the transmission efficiency and the packet delay when determining the lost-packet buffer size.

From Figure 5.10 we can also see that although the upper bounds of the improved schemes are adopted to compare with the available coding-based schemes, the gap between the improved schemes and their corresponding old ones are not big. For example, when the buffer size is 15, the upper bound of retransmission delay of the improved static scheme is only 20.4% larger than the retransmission delay of the old static one.

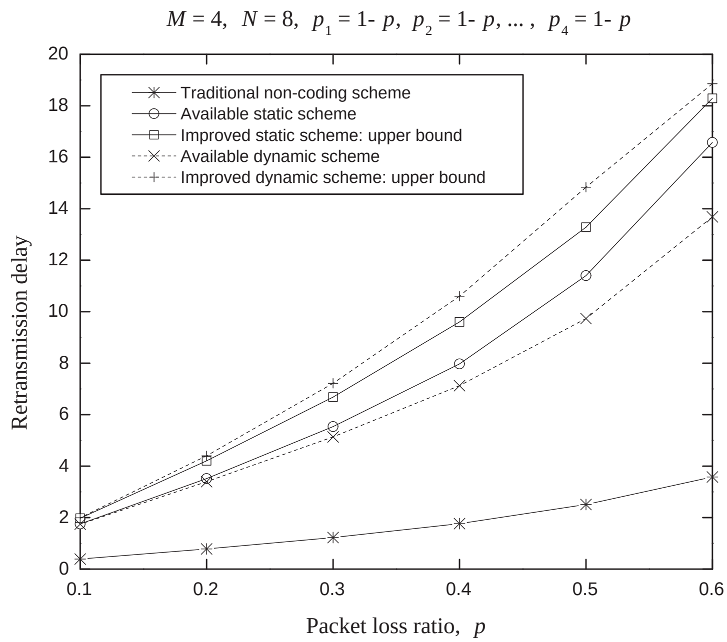


Figure 5.11: Delay versus packet loss probability.

We further show the retransmission delay under different packet loss probabilities in Figure 5.11 and the retransmission delay under different number of receivers in Figure 5.12. A similar conclusion can be draw from these two figures is that the transmission delay of the improved coding-based schemes is actually close to that of the old coding-based schemes.

5.5 Summary

In this chapter, we have proposed two improved network coding-based schemes for the reliable link-layer multicast: a static one with low complexity and a dynamic one with relatively higher complexity but a better performance. Unlike the available network coding-based schemes which have exponential computational complexity, the proposed schemes run in polynomial time. Moreover, the analytical and simulation results demonstrate that, compared with the available coding-based schemes, the improved schemes can more effectively reduce the bandwidth consumption, especially in the case of high packet loss

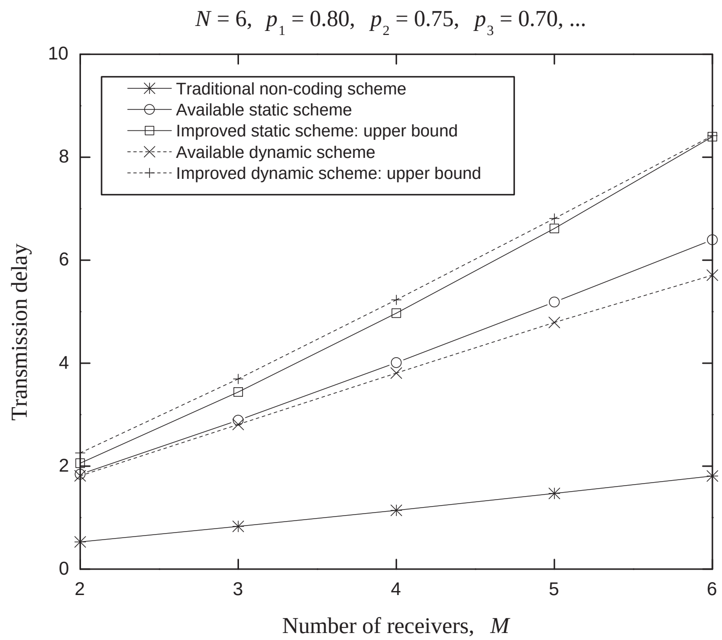


Figure 5.12: Delay versus number of receivers.

probabilities and many receivers.

It was also shown that the transmission efficiency improvement from using network coding increases with both the size of lost-packet buffer and also the number of multicast receivers. This improvement can be very significant when the lost-packet buffer size and number of receivers are large enough. E.g., for the case that the number of receivers is six and the buffer size is twelve packets, the transmission efficiency can be improved by as far as 24.8% when the proposed dynamic scheme is adopted. Thus, the network coding provides us a new dimension for a more efficient transmission of reliable link-layer multicast.

Chapter 6

Conclusion

6.1 Summary

Since network coding, a promising generalization of routing, was introduced by Ahlswede *et al.* in their pioneering work [1], this technique has been shown to be able to provide benefits for different connection cases (like multicast and broadcast), in both wired networks and wireless networks. Based on the available inspiring results of network coding, researchers in this fields believe that coding is a promising practical technique for packet networks. In this thesis, we have studied the application of network coding in several important communication cases. The main contributions are listed as follows.

Chapter 2 studied the challenging topology design problem of network coding-based multicast networks. Based on the characteristics of multicast and network coding, we formulated this problem as an NP-hard mixed-integer nonlinear programming problem, which is much more complicated than the conventional unicast-oriented topology design problems. Then we proposed two heuristic algorithms for this topology design problem. Finally, simulation results in this chapter showed that in comparison with the conventional unicast-oriented design for multicast networks, the Steiner tree-based design has moderate improvement in term of topology cost, but the network coding-based design can make this improvement very significant.

Chapter 3 extended the current COPE architecture by first proposing a flow-oriented virtual queue structure for it and then introducing an efficient algorithm for searching good coding solutions under the new queue structure. This queue structure can not only completely eliminate the packet reordering but also offer the maximum number of coding opportunities under the condition that no packet reordering is allowed. Extensive simulation results demonstrated that the available COPE can improve the node transmission efficiency, but this improvement can be more significant when the proposed virtual queue structure and new coding algorithm are jointly adopted.

Chapter 4 presented for the COPE architecture a new QoS queueing structure which can increase the potential coding opportunities and are convenient for the allocation of priorities to packets, and also proposed a new efficient packet coding algorithm. Rather than adopting FIFO scheduler, allocating priorities to different flows can satisfy the QoS requirement of multihop wireless networks for supporting real-time services such as voice applications. To our knowledge, this is the first time to take the QoS issue into account in the literature of wireless network coding. Simulation results demonstrated that by adopting the new queueing structure and new coding algorithm, COPE can further greatly improve the node transmission efficiency.

Chapter 5 presented two efficient network coding-based schemes for the reliable link-layer multicast: a static one with low complexity and a dynamic one with relatively higher complexity but a better performance. Unlike the available network coding-based schemes which have exponential computational complexity, the proposed schemes run in polynomial time. We evaluated, by both theoretical analysis and computer simulation, the performance of our schemes. Compared with the available coding-based schemes, the proposed schemes can more effectively reduce the bandwidth consumption, especially in the case of high packet loss probabilities and many receivers. It was also shown that the transmission efficiency improvement from using network coding increases with both the size of lost-packet buffer and also the number of multicast receivers. This improvement can be very significant when the lost-packet buffer size and number of receivers are large enough.

6.2 Future Work

Since the propose of network coding, this topic has been undergoing an active development in the research community. Realizing coded packet networks is a worthwhile goal. So far, there are still many interesting topics to be investigated about network coding. In the following, we list several future topics.

- As the first step, in this work we investigate the node-level performance improvement by using the new COPE architecture in Chapters 3 and 4. In the future, it is quite interesting to extend this work by investigating how much the network-level performance (like the end-to-end throughput) can be improved under different workloads, routing protocols, etc.

- The problem of coding-based reliable multicast in wireless networks deserves further research. Current work ([47], Chapter 5) is based on the assumption that the source node can timely receive the ACK (or NAK) packets to know which receivers do not successfully receive which packets. In practical networks, however, some ACK (or NCK) packets can not be correctly or timely received. Therefore, it is necessary to take this into account when design the coding-based reliable multicast scheme.

- Beside the address of the ACK problem, we can further take the group membership dynamics into account to get more practical coding-based multicast schemes which can adaptively cope with the joining and leaving of multicast members. Additionally, it is also necessary to design coding-based schemes for the multihop multicast (rather than one-hop multicast) in the multihop wireless networks.

- It is also worth the effort to study how to reduce the overhead in the packet header used for recording the linear mixture coefficients.

- It would be interesting to investigate the practicability of physical-layer network coding. The physical-layer network coding has been roughly explored in [41, 42]. However, a lot of practical issues need to be taken into account before demonstrating its practicability.

Bibliography

[1] R. Ahlswede, N. Cai, S.-Y. R. Li, and R. W. Yeung, "Network information flow," *IEEE Trans. Inform. Theory*, vol. 46, no. 4, pp. 1204–1216, July 2000.

[2] T. Ho, M. Medard, and R. Koetter, "An information-theoretic view of network management," *IEEE Trans. Inform. Theory*, vol. 51, no. 4, pp. 1295–1312, Apr. 2005.

[3] C. Fragouli, J. Boudec, , and J. Widmer, "Network coding: An instant primer," *ACM SIGCOMM Computer Communication Review*, vol. 36, no. 1, pp. 63–68, Jan. 2006.

[4] C. Fragouli and A. Markopoulou, "A network coding approach to overlay network monitoring," in *Proc. 43rd Annual Allerton Conference on Communication, Control, and Computing*, 2005.

[5] T. Ho, B. Leong, Y.-H. Chang, Y. Wen, and R. Koetter, "Network monitoring in multicast networks using network coding," in *Proc. IEEE ISIT*, 2005.

[6] K. Bhattad and K. R. Narayanan, "Weakly secure network coding," in *Proc. WINMEE, RAWNET and NETCOD 2005 Workshops*, Apr. 2005.

[7] N. Cai and R. W. Yeung, "Secure network coding," in *Proc. IEEE ISIT*, June 2002.

[8] J. Feldman, T. Malkin, R. A. Servedio, and C. Stein, "On the capacity of secure network coding," in *Proc. 42nd Annual Allerton Conference on Communication Control, and Computing*, Sept. 2004.

[9] T. Ho, B. Leong, R. Koetter, M. Medard, M. Effros, and D. R. Karger, "Byzantine modification detection in multicast networks using randomized network coding," in *Proc. IEEE ISIT*, June 2004.

[10] S. Jaggi, M. Langberg, T. Ho, and M. Effros, "Correction of adversarial errors in networks," in *Proc. IEEE ISIT*, Sept. 2005.

[11] M. M. S. Acedanski, S. Deb and R. Koetter, "How good is random linear coding based distributed networked storage," in *Proc. Netcod*, 2005.

[12] S.-Y. R. Li, R. W. Yeung, and N. Cai, "Linear network coding," *IEEE Trans. on Inform. Theory*, vol. 49, pp. 371–381, 2003.

[13] C. Chekuri, C. Fragouli, and E. Soljanin, "On average throughput and alphabet size in network coding," *IEEE Trans. Inform. Theory*, vol. 52, no. 6, pp. 2410–2424, June 2006.

[14] M. Charikar and A. Agarwal, "On the advantage of network coding for improving network throughput," in *Proc. IEEE Information Theory Workshop*, San Antonio, Oct. 2004.

[15] Z. Li and B. Li, "Network coding in undirected networks," in *Proc. CISS*, 2004.

[16] Y. Wu, P. Chou, and K. Jain, "A comparison of network coding and tree packing," in *Proc. IEEE ISIT*, Chicago, USA, 2004.

[17] D. S. Lun, N. Ratnakar, M. Médard, R. Koetter, D. R. Karger, T. HO, E. Ahmed, and F. Zhao, "Minimum-cost multicast over coded packet networks," *IEEE Trans. Inform. Theory*, vol. 52, no. 6, pp. 2608–2623, June 2006.

[18] R. Koetter and M. Medard, "Beyond routing: an algebraic approach to network coding," in *Proc. IEEE INFOCOM*, June 2002.

[19] ——, "An algebraic approach to network coding," *IEEE/ACM Trans. Networking*, vol. 11, no. 5, pp. 782–795, Oct. 2003.

[20] P. Sanders, S. Egner, and L. Tolhuizen, "Polynomial time algorithms for network information flow," in *Proc. 15th ACM Symp. on Parallel Algorithms and Architectures*, San Diego, June 2003.

[21] S. Jaggi, P. Chou, and K. Jain, "Low complexity algebraic network codes," in *Proc. IEEE ISIT*, Yokohama, Japan, June 2003.

[22] S. Jaggi, P. Sanders, P. A. Chou, M. Effros, S. Egner, K. Jain, and L. Tolhuizen, "Polynomial time algorithms for network code construction," *IEEE Trans. Information Theory*, vol. 51, pp. 1973–1982, June 2005.

[23] S. Jaggi, *Design and Analysis of Network Codes*. PhD thesis, California Institute of Technology, Pasadena, 2005.

[24] C. Fragouli and E. Soljanin, "Decentralized network coding," in *Proc. IEEE Information Theory Workshop*, Oct. 2004.

[25] T. Ho, D. R. Karger, M. Medard, and R. Koetter, "Network coding from a network flow perspective," in *Proc. IEEE ISIT*, Yokohama, Japan, June 2003.

[26] T. Ho, R. Koetter, M. Medard, D. R. Karger, and M. Effros, "The benefits of coding over routing in a randomized setting," in *Proc. IEEE ISIT*, Yokohama, Japan, June 2003.

[27] T. Ho, *Networking from a Network Coding Perspective*. PhD thesis, Massachusetts Institute of Technology, May 2004.

[28] P. A. Chou, Y. Wu, and K. Jain, "Practical network coding," in *Proc. 41st Allerton Conf. Communication, Control and Computing*, Monticello, IL, Oct. 2003.

[29] C. Gkantsidis and P. Rodriguez, "Network coding for large scale content distribution," in *Proc. IEEE INFOCOM*, Miami, Mar. 2005, pp. 2235–2245.

[30] C. Gkantsidis, J. Miller, and P. Rodriguez, "Comprehensive view of a live network coding P2P system," in *Proc. ACM SIGCOMM/USENIX IMC'06*, Oct. 2006.

[31] M. Wang and B. Li, "Lava: A reality check of network coding in Peer-to-Peer live streaming," in *Proc. IEEE INFOCOM*, May 2007.

[32] Y. Wu, P. A. Chou, and S. Y. Kung, "Information exchange in wireless networks with network coding and physical-layer broadcast," in *CISS'05*, Baltimore USA, Mar. 2005.

[33] Z. Li and B. Li, "Network coding: The case for multiple unicast sessions," in *Proc. Annual Allerton Conference on Communication, Control, and Computing*, Sept. 2004.

[34] ——, "On increasing end-to-end throughput in wireless ad hoc networks," in *Proc. The Second International Conference on Quality of Service in Heterogeneous Wired/Wireless Networks*, 2005.

[35] S. Katti, H. Rahul, W. Hu, D. Katabi, M. Medard, and J. Crowcroft, "XOR in the air: Practical wireless network coding," in *Proc. ACM SIGCOMM*, Pisa, 2006.

[36] B. Ni, N. Santhapuri, Z. Zhong, and S. Nelakuditi, "Routing with opportunistically coded exchanges in wireless mesh networks," in *Proc. WiMesh*, 2006.

[37] S. Sengupta, S. Rayanchu, and S. Banerjee, "An analysis of wireless network coding for unicast sessions: The case for coding-aware routing," in *Proc. IEEE INFOCOM*, Anchorage, Alaska, May 2007.

[38] J. Liu, D. Goeckel, and D. Towsley, "Bounds on the gain of network coding and broadcasting in wireless networks," in *Proc. IEEE INFOCOM*, Anchorage, Alaska, May 2007.

[39] J. Le, J. C. S. Lui, and D. M. Chiu, "How many packets can we encode? - An analysis of practical wireless network coding," in *Proc. IEEE INFOCOM*, Phoenix, Apr. 2008.

[40] K. Chi, X. Jiang, and S. Horiguchi, "Network coding opportunity analysis of COPE in multihop wireless networks," in *Proc. IEEE WCNC*, Las Vegas, 2008.

[41] S. Katti, S. Gollakota, and D. Katabi, "Embracing wireless interference: Analog network coding," in *Proc. ACM SIGCOMM*, Kyoto, Aug. 2007.

[42] S. Zhang, S. Liew, and P. Lam, "Hot topic: Physical-layer network coding," in *Proc. ACM MOBICOM*, Los Angeles, Sept. 2006.

[43] Y. Wu, P. A. Chou, and S.-Y. Kung, "Minimum-energy multicast in mobile Ad Hoc networks using network coding," *IEEE Trans. Commun.*, vol. 53, no. 11, pp. 1906–1918, Nov. 2005.

[44] J. Park, D. Lun, F. Soldo, M. Gerla, and M. Medard, "Performance of network coding in ad hoc networks," in *Proc. IEEE MILCOM*, Los Angeles, Sept. 2006.

[45] M. Ghaderi, D. Towsley, and J. Kurose, “Network coding performance for reliable multicast,” in *Proc. IEEE MILCOM*, 2007.

[46] ——, “Reliability gain of network coding in lossy wireless networks,” in *Proc. IEEE INFOCOM*, 2008.

[47] D. Nguyen, T. Nguyen, and B. Bose, “Wireless broadcasting using network coding,” in *Proc. Third Workshop on Network Coding, Theory, and Applications*, Jan. 2007.

[48] J. Park, M. Gerla, D. Lun, Y. Yi, and M. Medard, “Codecast: a network-coding-based ad hoc multicast protocol,” *IEEE Wireless Communications*, vol. 13, no. 5, 2006.

[49] C. Fragouli, J. Widmer, and J.-Y. L. Boudec, “A network coding approach to energy efficient broadcasting: From theory to practice,” in *Proc. IEEE INFOCOM*, Barcelona, Apr. 2006.

[50] ——, “Efficient broadcasting using network coding,” *IEEE/ACM Transactions on Networking*, vol. 16, no. 2, pp. 450–463, Apr. 2008.

[51] L. Li, R. Ramjee, M. Buddhikot, and S. Miller, “Network coding-based broadcast in mobile ad hoc networks,” in *Proc. IEEE INFOCOM*, Anchorage, Alaska, May 2007.

[52] I. Hou, Y. Tsai, T. Abdelzaher, and I. Gupta, “AdapCode: Adaptive Network Coding for Code Updates in Wireless Sensor Networks,” in *Proc. IEEE INFOCOM*, 2008.

[53] A. Kershenbaum, *Telecommunications Network Design Algorithms.* New York: McGraw-Hill, 1993.

[54] M. R. Garey and D. S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness.* New York: W.H. Freeman and Company, 1979.

[55] K. Steiglitz, P. Weiner, and D. Kleitman, “The design of minimum cost survivable networks,” *IEEE Trans. Circuit Theory*, vol. 16, no. 4, pp. 455–460, Nov. 1969.

[56] M. Gerla and L. Kleinrock, “On the topological design of distributed computer networks,” *IEEE Trans. Commun.*, vol. 25, no. 1, pp. 48–60, Jan. 1977.

[57] A. Kershenbaum, P. Kermani, and G. Grover, “MENTOR: An algorithm for mesh network topological optimization and routing,” *IEEE Trans. Commun.*, vol. 39, no. 4, pp. 503–513, Apr. 1991.

[58] F. Glover, M. Lee, and J. Ryan, “Least-cost network topology design for a new service: An application of a tabu search,” *Annals of Operations Research*, vol. 33, pp. 351–362, 1991.

[59] S. Pierre, M.-A. Hyppolite, J.-M. Bourjolly, and O. Dioume, “Topological design of computer communication networks using simulated annealing,” *Engineering Applications of Artificial Intelligence*, vol. 8, no. 1, pp. 61–69, 1995.

[60] S. Pierre and G. Legault, “An evolutionary approach for configuring economical packet switched computer networks,” *Artificial Intelligence in Engineering*, vol. 10, no. 2, pp. 127–134, 1996.

[61] C. L. M. M. Grotschel and M. Stoer, "Design of survivable networks," *Handbook in Operations Research and Management Science*, vol. 7, pp. 617–671, 1995.

[62] D. S. Lun, M. Médard, T. Ho, and R. Koetter, "Network coding with a cost criterion," in *Proc. ISITA'04*, Parma, Italy, Oct. 2004.

[63] D. Bertsekas and R. Gallager, *Data Networks*. Prentice Hall, 1992.

[64] K. Bhattad, N. Ratnakar, R. Koetter, and K. R. Narayanan, "Minimal network coding for multicast," in *Proc. IEEE ISIT*, Adelaide, Australia, Sept. 2005, pp. 1730–1734.

[65] R. Bolla, R. Bruschi, F. Davoli, and M. Repetto, "Analytical/simulation optimization system for access control and bandwidth allocation in IP networks with QoS," in *Proc. SPECTS'05*, Philadelphia, July 2005, pp. 339–348.

[66] K. Papagiannaki, S. Moon, C. Fraleigh, P. Thiran, and C. Diot, "Measurement and analysis of single-hop delay on an IP backbone network," *IEEE Journal on Selected Areas in Communications*, vol. 21, no. 6, pp. 908–921, Aug. 2003.

[67] K. Ishibashi, R. Kawahara, T. Asaka, M. Aida, S. Ono, and S. Asano, "Detection of TCP performance degradation using link utilization statistics," *IEICE transactions on communications*, no. 1, pp. 47–56, Jan. 2006.

[68] B. Jackson and T. Jordan, "A near optimal algorithm for vertex connectivity augmentation," *Lecture Notes In Computer Science*, vol. 1969, pp. 312–325, 2000.

[69] S.-C. Fang and S. Puthenpura, *Linear Optimization and Extensions: Theory and Algorithms*. New Jersey: Prentice Hall, 1993.

[70] D. W. Corne, M. J. Oates, and G. D. Smith, *Telecommunications Optimization: Heuristic and Adaptive Techniques*. New York: John Wiley & Sons Ltd, 2000.

[71] M. Listanti, V. Eramo, and R. Sabella, "Architectural and technological issues for future optical Internet networks," *IEEE Comm. Mag.*, vol. 38, no. 9, pp. 82–92, Sept. 2000.

[72] M. Charikar, C. Chekuri, T. y. Cheung, Z. Dai, A. Goel, S. Guha, and M. Li, "Approximation algorithms for directed Steiner problems," *Journal of Algorithms*, vol. 33, no. 1, pp. 73–91, 1999.

[73] P. Gupta and P. R. Kumar, "The capacity of wireless networks," *IEEE Transactions on Information Theory*, vol. 46, no. 2, pp. 388–404, Mar. 2000.

[74] J. Li, C. Blake, D. S. J. D. Couto, H. I. Lee, and R. Morris, "Capacity of Ad Hoc wireless networks," in *Proc. ACM MOBICOM*, Rome, 2001.

[75] J. Jun and M. L. Sichitiu, "The nominal capacity of wireless mesh networks," *IEEE Wireless Communications*, vol. 10, no. 5, pp. 8–14, Oct. 2003.

[76] K. Chi, X. Jiang, and S. Horiguchi, "A general packet coding scheme for multi-hop wireless networks," in *Proc. IEEE GLOBECOM*, Washington, 2007.

[77] R. M. Karp, *Complexity of Computer Computations*. New York: Plenum Press, 1972.

[78] C. Bettstetter, "On the minimum node degree and connectivity of a wireless multihop network," in *Proc. ACM MobiHoc*, Lausanne, Switzerland, 2002.

[79] Rishi Sinha and Christos Papadopoulos and John Heidemann, "Internet packet size distributions: Some observations," 2005. [Online]. Available: http://www.isi.edu/~johnh/PAPERS/Sinha07a.pdf

[80] Q. Dong, S. Banerjee, J. Wang, and D. Agrawal, "Wire speed packet classification without TCAMs: A few more registers (and a bit of logic) are enough," in *Proc. ACM SIGMETRICS*, 2007.

[81] M. Zorzi and S. Pupolin, "Optimum transmission ranges in multihop packet radio networks in the presence of fading," *IEEE Transactions on Communications*, vol. 43, no. 7, pp. 2201–2205, July 1995.

[82] H. Yang, K. Wu, and W. Lu, "Cross-layer path configuration for energy-efficient communication over wireless Ad Hoc networks," *Advances in Multimedia*, vol. 2007, 2007.

[83] B. G. Chun and M. Baker, "Evaluation of packet scheduling algorithms in mobile Ad Hoc networks," *ACM Mobile Computing and Communications Review*, July 2002.

[84] J. Nagle, "On packet switches with infinite storage," *IEEE Trans. on Communications*, vol. 35, pp. 435–438, 1987.

[85] S. Ramabhadran and J. Pasquale, "Stratified round robin: A low complexity packet scheduler with bandwidth fairness and bounded delay," in *Proc. ACM SIGCOMM*, Karlsruhe, Germany, Aug. 2003.

[86] J. Nonnenmacher, M. Lacher, M. Jung, G. Carl, and E. Biersack, "How bad is reliable multicast without local recovery?" in *Proc. IEEE INFOCOM*, San Francisco, Mar. 1998.

[87] S. Paul, K. Sabnani, J. Lin, and S. Bhattacharya, "Reliable multicast transport protocol (RMTP)," *IEEE Journal on Selected Areas in Communications*, Apr. 1997.

[88] D. Nguyen, T. Tran, T. Nguyen, and B. Bose, "Wireless broadcast using network coding," *accepted to the IEEE Transactions of Vehicular Technology*.

[89] F. J. MacWilliams and N. J. A. Sloane, *The Theory of Error Correcting Codes*. Amsterdam: North Holland, 1977.

[90] R. Roth and G. Seroussi, "On generator matrices of MDS codes," *IEEE Trans. Inform. Theory*, vol. 31, no. 6, 1985.

Publications

Journals

[1] K. Chi, X. Jiang, S. Horiguchi and M. Guo, "Topology Design of Network Coding-Based Multicast Networks," *IEEE Trans. on Parallel and Distributed System*, Vol. 19, No. 5, pp 627-640, 2008.

[2] K. Chi, X. Jiang and S. Horiguchi, "A More Efficient COPE Architecture for Network Coding in Multihop Wireless Networks," *IEICE Trans. on Communications*, to be published.

[3] K. Chi, C. Yang and X. Wang, "Performance of Network Coding Based Multicast," *IEE Proceedings on Communications*, Vol. 153, No. 3, pp.399-404, 2006.

[4] K. Chi and X. Wang, "Analysis of Network Error Correction Based on Network Coding," *IEE Proceedings on Communications*, Vol. 152, No. 4, pp.393-396, 2005.

Submitted papers

[5] K. Chi, X. Jiang and S. Horiguchi, "Flow-Oriented Queueing for Throughput Improvement in Network Coding-Based Multihop Wireless Networks," Submitted to *IEEE/ACM Trans. on Networking*.

[6] K. Chi, X. Jiang and S. Horiguchi, "Network Coding-Based Reliable Multicast in Wireless Networks," Submitted to *IEEE Journal on Selected Areas in Communications*.

[7] K. Chi, X. Jiang and S. Horiguchi, "Efficient Network Coding-Based Loss Recovery for Reliable Multicast in Wireless Networks," Submitted to *IEICE Trans. on Communications*.

[8] K. Chi, X. Jiang and S. Horiguchi, "Analysis of Node Transmission Rate Improvement for COPE-based Multihop Wireless Networks," Submitted to *International Journal of High Performance Computing and Networking*.

Conferences

[9] K. Chi, X. Jiang and S. Horiguchi, "Network Coding Opportunity Analysis of COPE in Multihop Wireless Networks ," in *IEEE WCNC 2008*, Las Vegas, 2008. **(Best paper award)**

[10] K. Chi, X. Jiang and S. Horiguchi, "A General Packet Coding Scheme for Multi-hop Wireless Networks," in *IEEE GLOBECOM 2007*, Washington D.C., 2007.

[11] K. Chi, X. Jiang and S. Horiguchi, "An Improved Topology Design Algorithm for Network Coding-Based Multicast Networks," in *IEEE ICC 2007*, Glasgow, 2007.

[12] K. Chi, X. Jiang and S. Horiguchi, "Network-coding Based Topology Design for Multicast Networks," in *IEEE BROADNETS 2006*, San Jose, 2006.