11-2-2009

Resource Management and Optimization in Wireless Mesh Networks

Xiaowen Zhang
Florida International University, xzhan001@fiu.edu

DOI: 10.25148/etd.FI10041636

FLORIDA INTERNATIONAL UNIVERSITY

Miami, Florida

RESOURCE MANAGEMENT AND OPTIMIZATION IN WIRELESS MESH NETWORKS

A dissertation submitted in partial fulfillment of the

requirements for the degree of

DOCTOR OF PHILOSOPHY

in

ELECTRICAL ENGINEERING

by

Xiaowen Zhang

2010

To: Dean Amir Mirmiran
College of Engineering and Computing

This dissertation, written by Xiaowen Zhang, and entitled Resource Management and Optimization in Wireless Mesh Networks, having been approved in respect to style and intellectual content, is referred to you for judgment.

We have read this dissertation and recommend that it be approved.

Niki Pissinou

Kia Makki

Jean Andrian

Shih-Ming Lee

Hao Zhu, Major Professor

Date of Defense: November 2, 2009

The dissertation of Xiaowen Zhang is approved.

Dean Amir Mirmiran
College of Engineering and Computing

Interim Dean Kevin O’Shea
University Graduate School

Florida International University, 2010

DEDICATION

To My Parents and My Husband.

ACKNOWLEDGMENTS

I want to thank the members of my dissertation committee for their consistent support.

I would like to give my deepest thanks to my Major Professor Dr. Hao Zhu for his guidance and support. He gave me a lot of helpful academic advices through my academic program at FIU and made a lot of technical contribution to my dissertation work. Without his constant support and encouragement, the completion of this work would not have been possible.

Besides my advisor, I would like to thank my committee members - Dr. Niki Pissinou, Dr. Kia Makki, Dr. Jean H. Andrian and Dr. Shih-Ming Lee for their helpful suggestion and instruction. I would also like to express my gratitude to Dr. Kang K. Yen, Dr. Kia Makki, Dr. Niki Pissinou, Dr. Zesheng Chen and Dr. Jean H. Andrian for the financial support without which the completion of the dissertation would be impossible. Also, many thanks to CS professor Geoffrey Smith for the latex template of dissertation.

I also want to thank all the lab mates in Telecommunication and Information Technology Insititute (IT2) for their inspiration and friendship. In particular, I want to thank Yuan Tang, Masoumeh Karimi, Qian Wang, Kai Chen, Zhiwen Wan, Yong Wang, Qutub Bakhtiar and Changan Han for their help. My appreciation also goes to Pat Brammer, Maria Benincasa, Daniela C. Madriz and Dr. Vinayak Ganapathy for their kind administrative support.

Lastly, and most importantly, I would like to thank my family - my parents and my husband Jingxue Lu. And I want to thank Lord Jesus for his endless love. They make me live a wonderful life.

ABSTRACT OF THE DISSERTATION

RESOURCE MANAGEMENT AND OPTIMIZATION IN WIRELESS MESH NETWORKS

by

Xiaowen Zhang

Florida International University, 2010

Miami, Florida

Professor Hao Zhu, Major Professor

A wireless mesh network is a mesh network implemented over a wireless network system such as wireless LANs. Wireless Mesh Networks(WMNs) are promising for numerous applications such as broadband home networking, enterprise networking, transportation systems, health and medical systems, security surveillance systems, etc. Therefore, it has received considerable attention from both industrial and academic researchers. This dissertation explores schemes for resource management and optimization in WMNs by means of network routing and network coding.

In this dissertation, we propose three optimization schemes. (1) First, a triple-tier optimization scheme is proposed for load balancing objective. The first tier mechanism achieves long-term routing optimization, and the second tier mechanism, using the optimization results obtained from the first tier mechanism, performs the short-term adaptation to deal with the impact of dynamic channel conditions. A greedy sub-channel allocation algorithm is developed as the third tier optimization scheme to further reduce the congestion level in the network. We conduct thorough theoretical analysis to show the correctness of our design and give the properties of our scheme. (2) Then, a Relay-Aided Network Coding scheme called RANC is proposed to improve the performance gain of network coding by exploiting the physical layer multi-rate capability in WMNs. We conduct rigorous analysis to find the design principles and study the tradeoff in the performance gain of RANC.

Based on the analytical results, we provide a practical solution by decomposing the original design problem into two sub-problems, flow partition problem and scheduling problem. (3) Lastly, a joint optimization scheme of the routing in the network layer and network coding-aware scheduling in the MAC layer is introduced. We formulate the network optimization problem and exploit the structure of the problem via dual decomposition. We find that the original problem is composed of two problems, routing problem in the network layer and scheduling problem in the MAC layer. These two sub-problems are coupled through the link capacities. We solve the routing problem by two different adaptive routing algorithms. We then provide a distributed coding-aware scheduling algorithm. According to corresponding experiment results, the proposed schemes can significantly improve network performance.

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

1.1 General Statement of Problem Area

A wireless mesh network is a mesh network implemented over a wireless network system such as wireless LANs. Mesh networks differ from other networks in that the component parts can all connect to each other via multiple hops, and they are assumed to be stationary in this research work. Wireless Mesh Networks(WMNs) is promising for numerous applications. For example, broadband home networking, community and neighborhood networking, enterprise networking, metropolitan area networks, transportation systems, health and medical systems, security surveillance systems, etc. Therefore, it has received considerable attention from both industrial and academic researchers. Let's take enterprise networking for example. Currently, standard IEEE 802.11 wireless networks are widely used in various offices. However, these wireless networks are still isolated islands. Connections among them have to be achieved through wired Ethernet connections, which is the key reason for the high cost of enterprise networks. If the access points are replaced by mesh routers, as shown in Figure 1.1, Ethernet wires can be eliminated. WMNs can grow easily as the size of enterprise expands. The service model of enterprise networking can be applied to many other public and commercial service networking scenarios such as airports, hotels, shopping malls, convention centers, sport centers, etc.

This dissertation explores two important research problems in wireless mesh networks. (1) Load balancing. The goal of load balancing is congestion control. We use load balancing technique to minimize the congestion level given the available network resources. The importance of load balancing lies in the fact it can maximize the utilization of wireless bandwidth and guarantee quality of service (QoS). A network that is not well balanced is more likely to have delay variation, bad QoS and waste of bandwidth. For wireless networks, it is challenging to provide a guaranteed performance due to the dynamic characteristics of wireless channels. Therefore, it is very important to study how to achieve a good load balancing in wireless mesh networks.

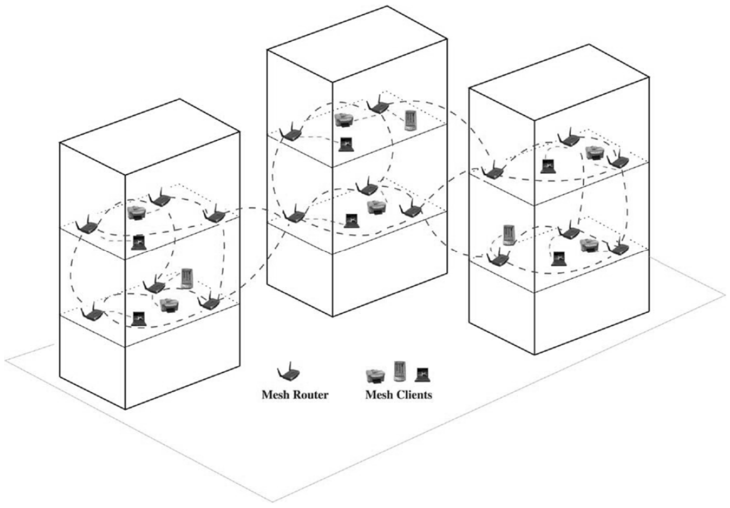


Figure 1.1: WMNs for enterprize networking[Aky05]

(2) Network coding. Communication networks today share the same fundamental principle of operation, whether it is packets over the Internet, or signals in a phone network, information is transported in the same way as cars share a highway or fluids share pipes. That is, independent data streams may share network resources, but the information itself is separate. Routing, data storage, error control, and generally all network functions are based on this assumption. Network coding [KRH$^+$06] is a recent field in information theory that breaks with this assumption. Instead of simply forwarding data, nodes may recombine several input packets into one or several output packets. Network coding has received considerable attention as a mechanism to increase the performance of both wired and wireless networks. The essence of network coding is to convey more information in each transmission by mixing information from different sources. In wireless networks, network coding exploits the broadcast nature of wireless medium to improve system throughput. Since the operation does not require any upgrade of hardware, network coding can be easily imple-

mented and deployed in wireless networks. Therefore, this dissertation focuses on how to improve the performance gain of network coding for wireless mesh networks.

1.2 Research Questions and Hypotheses

Followings are the research questions and hypotheses.

Question 1: How to build a wireless mesh network cost-effectively?

Question 2: How to achieve the goal of load balancing given the dynamic condition of wireless channels?

Question 3: How to improve the performance gain of network coding?

Question 4: How to combat the impact of poor channel conditions on the performance of network coding?

Question 5: How to create more network coding opportunities?

Hypotheses 1: Load balancing may be achieved by proper routing mechanism.

Hypotheses 2: Throughput may be improved by using well-designed network coding techniques.

Hypotheses 3: More network coding opportunities may be created by proper routing.

1.3 Contributions

This dissertation mainly make the following contributions:

1. We propose a triple-tier optimization scheme for load balancing in wireless mesh networks with OSPF routers. The first tier mechanism achieves long-term routing optimization, and the second tier mechanism, using the optimization results obtained from the first tier mechanism, performs the short-term adaptation to deal with the impact of dynamic channel conditions. Based on the average traffic demands and average link capacity, we use linear programming techniques to find the optimal routing policy for load balancing of each

source-destination traffic by setting an appropriate weight to each link. This enable us to use current OSPF routers to build a wireless back-haul. Since current OSPF routers only support discrete traffic splitting, we propose a greedy MIN-MAX Congestion algorithm to find a shortest-path set in order to approximate the optimal routing policy within a certain bound. Due to the impact of dynamic channel conditions, we propose the second tier adaption scheme which runs frequently enough to refine the routing policy to balance the overall traffic according to the channel condition. To utilize the subchannels in orthogonal frequency division multiple access (OFDMA) for the load balancing purpose, a greedy subchannel allocation algorithm is developed and used as the third tier optimization scheme to further reduce the maximum congestion level in the network. We conduct thorough theoretical analysis to show the correctness of our design and give the properties of our scheme. Our solution is also evaluated via simulations and the simulation results show that our work can effectively lower maximum congestion level of the network to a near-optimal value.

2. We propose a *Relay-Aided Network Coding* (RANC) scheme to combat the impact of poor channel condition on the performance of network coding by exploiting the PHY layer multi-rate capability in wireless mesh networks. We investigate the design principles via rigorous analysis and derive the coding structure for RANC. Based on the analytical results, we design the RANC protocol by decomposing the original problem into two sub-problems: the flow partition problem and the scheduling problem. We evaluate our design via simulations. The simulation results show that RANC can significantly outperform COPE [KRH$^+$06] in terms of the throughput of network coding.

3. We propose a method for exploiting the network coding gain via a joint design of both the routing at the network layer and the network coding-aware scheduling at the MAC layer to improve the overall networking performance. We formulate the design problem by nonlinear programming and decompose the original problem into two sub-problems: the routing problem and the scheduling problem. These two subproblems are linked through

the performance factor related to the queuing delay at each node. We propose distributed solutions for both the routing problem and the scheduling problem. We evaluate the performance of the proposed scheme through simulations and the simulation results show that our scheme can significantly improve network throughput.

1.4 Outline of Dissertation

The rest of this dissertation is organized as follows: Chapter 2 presents the related work for topics covered by this dissertation. Chapter 3 proposes a triple-tier load balancing scheme. In Chapter 4, a network coding scheme called RANC is provided and analyzed in detail, while Chapter 5 introduces a joint optimization mechanism of routing and network coding aware scheduling. The performance evaluation of the schemes proposed from chapter 3 to 5 is provided in each chapter. Chapter 6 draws the conclusion and future work is given out.

CHAPTER 2

RELATED WORK

2.1 Load Balancing

With efforts made in improving data rates and reducing cost, wireless access technologies have paved the way for the high-speed broadband wireless data service. Using wired links (such as T1, DSL, T3, etc.) to connect access points or base stations to the Internet has many difficulties due to relatively lower speed of wired links and high deployment cost. Consequently, considerable research and commercial efforts are paid to develop wireless back-hauls as an alternative to costly wired infrastructure via wirelessly multi-hopping to a high-speed and low-cost wired Internet entry point such as a metropolitan network operations center or an institute [GSK04] [Inc].

As broadband wireless data services are expected to have significant growth over the next decade, current and evolving standards for broadband wireless system, such as WiMAX (IEEE 802.16) [STS04] and 3GPP LTE [EFK$^+$06] have adopted or are considering the Orthogonal Frequency Division Multiple Access (OFDMA) as the multiple access technology for the wireless network interface. OFDMA has been demonstrated to be a superior wireless access technology for broadband wireless data network compared with traditional access technologies such as TDMA and CDMA. The main advantages of OFDMA over TDMA/CDMA come from its resiliency to frequency selectivity, low implementation complexity, subchannel orthogonality and flexibility of deployment.

Even though it is technically and economically attractive to deploy wireless back-hauls, it is challenging to provide a guaranteed performance (*e.g.*, throughput and delay) due to the dynamic characteristics of wireless links. For example, the unexpected changes in network topology due to link or router failures would significantly worsen the network performance. Thus, the optimization of bandwidth usage in wireless back-hauls becomes very important. Some works [Wan01] [For00] in traffic engineering for Internet studied how to find optimal path set that minimizes the congestion level given the available network resources. Wang *et al.* gave an optimization framework in which the load balancing problem is transformed to the shortest path routing problem with the careful selection of the weight of each link. Fortz and Thorup [For00] considered the restriction from OSPF routers and proposed a local search heuristic algorithm to approximate the optimal solution. Chim *et al.* [CYL05] proposed a traffic splitting algorithm based on the load sharing statistics collected. However, these schemes do not work well for wireless back-hauls since they did not take into account the impact of dynamic channel conditions. Other works [PSL08] [ZDA06] [JCL05] studied dynamic routing to achieve load balancing, however these schemes are primarily designed for overlay networks. Xie *et. al.* [Xie03] proposed a self-adaptive routing scheme to dynamically adjust routing policy according to network condition. In [NBTD07], a tabu-search heuristic is proposed for choosing link weights to compute routing path. However, these schemes mainly focus on the routing problem in wired networks and require that each router is able to arbitrarily split traffic, which is not the case for OSPF routers.

In this dissertation, we explore the problem of load balancing in wireless back-hauls with OSPF routers. Our objective is to increase the available bandwidth without upgrading existing equipments by minimizing the congestion level. We propose a triple-tier optimization scheme in this dissertation. The first tier mechanism achieves long-term routing optimization, and the second tier mechanism, using the optimization results obtained from the first tier mechanism, performs the short-term adaptation to deal with the impact of dynamic channel conditions. A greedy sub-channel allocation algorithm is developed as the third tier optimization scheme to further reduce the congestion level in the network. In particular, We make following contributions:

First, at the first tier, we apply a prime-dual algorithm to convert the problem of optimal routing with the constraints of traffic demands and link capacity, to the shortest path routing problem by carefully setting link weights. Compared to [Wan01], our prime-dual algorithm has the strong duality property. As a result, routing optimization can be applied to current OSPF routers which execute the Dijkstra algorithm according to the weight of each link. Due to the limited traffic splitting ability of current OSPF routers (say, splitting traffic in a round robin or weighted round robin way), each router may not be able to perform the optimal multi-path routing by splitting the traffic demands with an arbitrary ratio. Thus, according to the shortest-path set derived from the first tier mechanism, we present a Greedy MIN-MAX Congestion (GMC) algorithm which can approximate the optimal congestion level within a sub-linear factor. The GMC algorithm runs at each ingress router and distributes each source-destination traffic demand over a subset of the shortest path set obtained from the first tier module.

Second, due to the dynamic wireless links, the static routing policy derived from the first tier is not sufficient to balance traffic well. Therefore, we present the second tier adaptation algorithm which adjusts routing decisions with the help of the first tier optimization information as well as the real-time path congestion information. As a result, the traffic distribution in the network is well adapted according to channel conditions so that the network congestion level can be reduced. The second tier adaptation algorithm independently runs at each source node, and distributively achieves an equilibrium between different flows in the network. Third, since the discrete traffic splitting may cause unbalanced subchannel usage on each link, we design the dynamic subchannel allocation scheme as the third tier optimization mechanism to allocate the available subchannels to the links according to the traffic volume over the links. As a result, the maximum congestion level can be further reduced. We formally present the problem and propose the Greedy Subchannel Allocation (GSA) algorithm with bounded optimality. The third tier module runs independently at each intermediate router which allocates subchannels according to the real-time traffic volume of each link.

2.2 Network Coding

The concept of network coding was first proposed by Ahlswede *et. al.* [ACLY00] for multi-cast in wired networks. Since then a number of works have studied how to construct efficient network coding [CWJ03] [DFZ05] [KM03] [LYC03] [MEHK03]. Some recent works have studied network coding in wireless networks [KRH$^+$06] [LGT07] [LRK$^+$05b] [SRB07] [FWB06] [YP07]. Lun *et. al.* [LRK05a] [LRK$^+$05b] studied the problem of minimum cost single-session multi-cast. Ramamoorthy [RSW03] derived results for maximum flow achievable in random wireless networks for the same application. Li *et. al.* [LL05] showed that the throughput gain is unbounded in directed networks and the gain becomes marginal in some scenarios in wireless multi-hop networks. Liu *et. al.* [LGT07] studied the upper bound of the throughput gain of network coding in 1-D and 2-D random wireless networks.

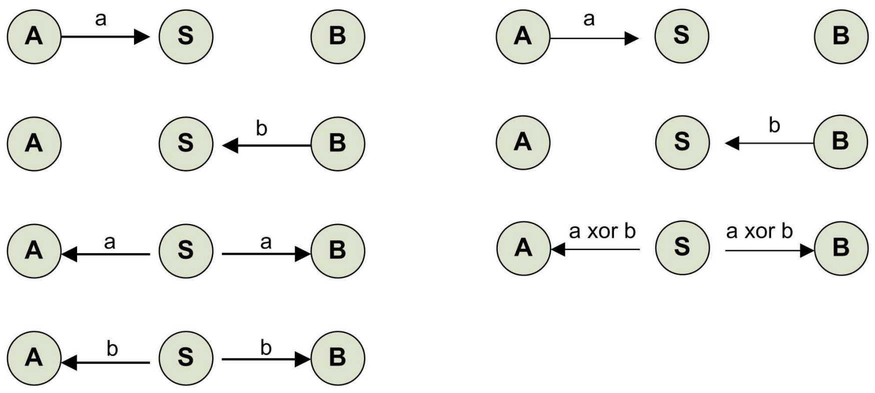


Figure 2.1: Illustration of network coding [Fra06]

While most of the previous works are from information-theoretic perspective and have few consideration on practical situations, Katti *et. al.* [KRH$^+$06] proposed a scheme, called COPE, using XOR-based coding and demonstrated the performance gain with real experimental results. COPE inserts a coding shim between the IP and MAC layers, which identifies coding opportunities and benefits from them by forwarding multiple packets in a single transmission. In wireless networks, network coding exploits the broadcast nature of wireless medium to improve the bandwidth utilization via a simple operation such as bitwise XOR [KRH$^+$06]. In COPE, the performance gain comes from the opportunistic coding and listening which are defined as follows.

Opportunistic Listening: Wireless is a broadcast medium, creating many opportunities for nodes to overhear packets when they are equipped with omni-directional antennae. COPE sets the nodes in promiscuous mode, makes them snoop on all communications over the wireless medium and store the overheard packets for a limited period T(ex,0.5s)

Opportunistic Coding: The key question is what packets to code together to maximize throughput. A node may have multiple options, but it should aim to maximize the number of native packets delivered in a single transmission, while ensuring that each intended nexthop has enough information to decode its native packet.

COPE also defined some terms to help analyze itąŕs performance gain, such as coding gain, coding+MAC gain. Coding Gain is the ratio of the number of transmissions required by the current non-coding approach, to the number of transmissions used by COPE to deliver the same set of packets. Coding+MAC Gain is the expected throughput gain with COPE when an 802.11 MAC is used and all the nodes are backlogged.

We explain the basic idea of COPE through a simple example shown in Figure 2.1. Nodes A and B want to exchange packets but they are not within the transmission range of each other. They have to communicate via an intermediate node S. Without network coding, the number of transmissions needed for node A and node B to successfully exchange a packet is 4. The steps are shown in the left part of Figure 2.1. Node A and node B send their packets to node S first. Then node S can either broadcast(shown in Figure 2.1) or just transmit the packet to its destination. While with network coding, after node S has the information of both packet a and b, it broadcasts a XOR b instead of a and b in sequence. Both A and B can recover the packet of interest, while the number of transmissions is reduced. In this case, Coding Gain=4/3 and Coding+MAC Gain=2. Since the operation needed by network coding (i.e. XOR) does not require any upgrade of hardware, network coding can be easily implemented and deployed in wireless networks.

In COPE, the performance gain (in terms of throughput and bandwidth efficiency) comes from the opportunistic coding and listening, and is an increasing function of the number of flows that benefit from network coding. In order to maximize the performance gain of a given network coding opportunity consisting of a group of nodes that will participate in network coding, in COPE it is necessary that a packet sent by a source node should be successfully overheard by all other nodes except for the destination. It turns out that a poor channel condition from the sender to an overhearing node might be the bottleneck of COPE's performance. Since most of wireless networks have the physical layer multi-rate capability[1], it is possible to *combat the impact of poor channel condition on the performance of network coding by exploiting the PHY layer multi-rate capability*.

Our approach is inspired by the observation that each node only needs a combination of the native (or original) packets from a set of other nodes to decode the packet of interest. This indicates the possibility that a native packet can be combined with other native packets and then be relayed to intended recipients. Following this idea, we propose the *Relay-Aided Network Coding* (RANC) scheme in multi-hop wireless networks. In RANC, some nodes can transmit their native packets to their neighbors at high date rate. Then the receiving nodes XOR these packets with their own native packets, and transmit the coded packet to other nodes with a suitable rate. On one hand, compared to COPE, RANC can improve the performance of network coding in that the node that has poor channel condition can transmit its native packet over a short-range (i.e. sending the packet only to its near neighbors), so that the impact of poor channel condition on the performance of network coding can be significantly alleviated. On the other hand, RANC has the same message complexity as

[1]For example, IEEE 802.11b PHY layer supports a number of transmission rates (say $1, 2, 5.5$ and 11 Mbps) such that the transmission rate can be adapted according to the channel condition.

that in COPE since we require each node to transmit once during each round of network coding.

From the illustration of wireless network coding in Figure 2.1, we can also see that the performance gain of network coding depends on following factors: first, the coding opportunities are related to the network topology and the traffic flowing on each link. These factors determine how often the network coding can be performed. Second, the coding gain depends on the per-link quality and the scheduling policy of the network. Because the coded packets are broadcasted to all the recipients, the transmission rate might be reduced due to bad channel condition at one recipient. The decreased transmission rate may significantly affect the overall networking performance. Following these considerations, several works have been proposed to find new routing and scheduling algorithms that leverage the network coding gain to improve the overall networking performance. Gupta *et. al.* [SRB07] explored schemes to increase the number of network coding opportunities through network coding-aware routing. Le *et. al* [LLC08] proposed a distributed routing discovery mechanism for finding the end-to-end paths that have potential network coding opportunities. In [ZCM08], the protocol called BEND was designed to create more network coding opportunities via locally bending last-2-hop sub-routes. Yomo and Popovskic [Pop06] studied the impact of fading channel on the performance of network coding and proposed an opportunistic scheduling algorithm which carefully selects which packets to be coded based on the instantaneous channel conditions. Zhang *et. al* [ZZ09] exploited the power of cooperative communication to alleviate the impact of channel fading on the performance bottleneck of the network coding (i.e. low-rate broadcasting of coded packets) for both Alice-Bob and X-structure topologies in the network. Chaporkar and Proutiere [Cha07] proposed an centralized optimal scheduling algorithm for network coding. Li and Wang gave a theoretical framework in [LW08], which gives a centralized algorithm for network coding-aware routing, code construction and MAC layer scheduling. However, most of the previous works either focus on the network coding-aware routing and scheduling problems separately, or study the joint design of them under quite strong assumption (e.g. global topology and traffic information or perfect MAC layer functionality).

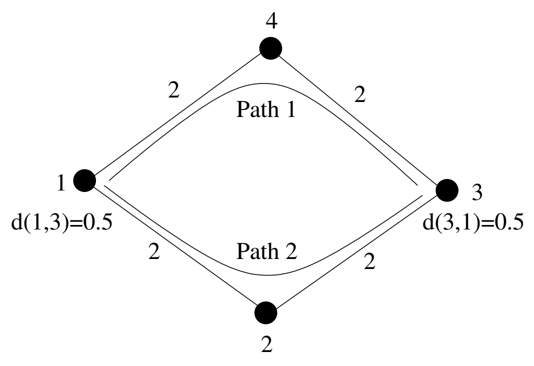


Figure 2.2: The Motivational Example

To practically exploit the network coding gain to improve the overall network performance in wireless mesh network, it is desirable to consider the interaction between the routing and scheduling mechanisms to leverage the network coding gain. We explain the importance in the following motivational example. As shown in Figure 2.2, two symmetric flows between nodes 1 and 3 have the traffic demand of $1/2$. Suppose all the links have the rate of 2 and the MAC layer scheduling is fair and capture every network coding opportunities, if node 2 does not perform network coding while node 4 supports network coding and symmetric traffic flows on each path, then the capacities of path 1 and path 2 are $3/4$ and 1, respectively. Assuming the average end-to-end delay of path i follows $1/(c_i - x_i)$, where c_i and x_i are path capacity and aggregated traffic over path i, respectively. Suppose both flows assign x_1 and x_2 of the traffic over path 1 and path 2 respectively, the problem of minimizing the average end-to-end latency can be modeled to minimize the objective function $x_1/(3/4 - x_1) + x_2/(1 - x_2)$ subject to the constraint $x_1 + x_2 = 1$. The minimal average delay is 1.48 when $x_1 = 0.68 (x_2 = 0.32)$. It is easy to see that under dynamic traffic pattern and scheduling decision, the minimal average delay would be different. For example the routing decision $(x_1, x_2) = (0.68, 0.32)$ is not optimal if node 4 cannot realize all network coding opportunities.

In this dissertation, we also propose a joint design of distributed routing at the network layer and distributed coding-aware scheduling at the MAC layer. The design problem is modeled as a network utility optimization problem. Different from previous works in cross-layer optimization in computer networks [Haj88] and [Tas92], we focus on how to exploit the network coding gain to improve the overall end-to-end transmission performance in network coding-aware wireless mesh network. We first formalize the design problem and decompose it into two sub-problems: the routing problem and the scheduling problem. The cross-layer design is achieved with a simple but crucial linkage between these subproblems. The routing mechanism decides how to distributed the traffic demands over the end-to-end paths in the network. The scheduling mechanism is used to maximize the overall weighted link capacity with network coding awareness. With the joint design, the network coding gain is realized by the scheduling module. The link level gain is further signaled to the routing module which utilizes the information to carefully distribute the traffic to maximize the overall end-to-end networking performance. As a result, more traffic is routed on the path to create more network coding opportunities until the network reaches the equilibrium point where no better routing decisions can be found. In addition to the theoretical framework, we propose mechanisms to implement the framework in an efficient and distributed way by considering the factors of time synchronization, protocol scalability, and the MAC layer coordination. We evaluate our design through extensive simulations and the simulation results show that the joint design of routing and scheduling can greatly increase the capacity of the network by exploiting the network coding-awareness.

CHAPTER 3

TRIPLE-TIER LOAD BALANCING SCHEME

This chapter mainly presents a triple-tier routing optimization scheme for load balancing in wireless back-hauls with OSPF routers. Our objective is to increase the available bandwidth without upgrading existing equipments by minimizing the maximum congestion level. We propose a triple-tier optimization framework (Figure 3.1) in this chapter. The first tier mechanism achieves long-term routing optimization, and the second tier mechanism, using the optimization results obtained from the first tier mechanism, performs the short-term adaptation to deal with the impact of dynamic channel conditions. The third tier mechanism makes a balanced use of orthogonal subchannels in OFDMA and temporarily allocates the subchannels to congested links to further reduce the congestion level in the network. In particular, We make the following contributions:

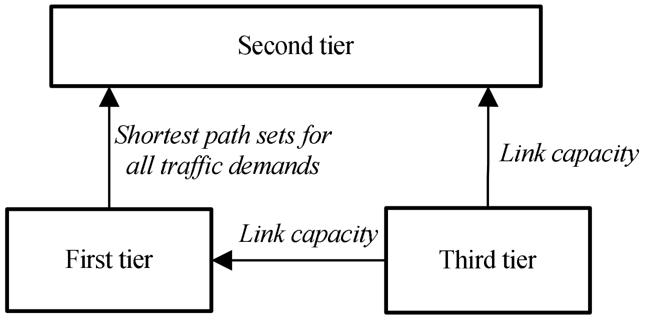


Figure 3.1: System Diagram

First, at the first tier, we design a primal-dual algorithm to convert the problem of optimal routing with the constraints of traffic demands and link capacity, to the shortest path routing problem by carefully setting link weights. Compared to [Wan01], our primal-dual algorithm has the strong duality property. As a result, routing optimization can be applied to current OSPF routers which execute the Dijkstra algorithm [and98] according to the weight of each link. Due to the limited traffic splitting ability of current OSPF routers (say, splitting traffic in a round robin or weighted round robin way), each router may not be able to perform the optimal multi-path routing by splitting the traffic demands with an arbitrary ratio. Thus, according to the shortest-path set derived from the first tier mechanism, we present a Greedy MIN-MAX Congestion (GMC) algorithm which can approximate the optimal congestion level within a sub-linear factor. The GMC algorithm runs at each ingress router and distributes each source-destination traffic demand over a subset of the shortest path set obtained from the first tier module.

Second, due to the dynamic wireless links, the static routing policy derived from the first tier is not sufficient to balance traffic well. Therefore, we present the second tier adaptation algorithm which adjusts routing decisions with the help of the first tier optimization information as well as the real-time path congestion information. As a result, the traffic distribution in the network is well adapted according to channel conditions so that the network congestion level can be reduced. The second tier adaptation algorithm independently runs at each source node, and distributively achieves an equilibrium between different flows in the network.

Third, since the discrete traffic splitting may cause unbalanced subchannel usage on each link, we design the dynamic subchannel allocation scheme as the third tier optimization mechanism to allocate the available subchannels to the links according to the traffic volume over the links. As a result, the maximum congestion level can be further reduced. We formally present the problem and propose the Greedy Subchannel Allocation (GSA) algorithm with bounded optimality. The third tier module runs independently at each intermediate router which allocates subchannels according to the real-time traffic volume of each link.

We give rigorous analysis to prove the correctness of our design. We also use extensive simulations to evaluate the performance of the two-tier load balancing scheme. Simulation results show that, our scheme is able to promptly react to the change of channel conditions and effectively minimize the maximum congestion level in the network accordingly.

This chapter first provides our system model and notations. Then the algorithms used in the first and second tier modules are introduced respectively. Performance evaluation is given out at the end of this chapter.

3.1 System Model and Notations

We study an all-IP wireless back-haul network consisted by a number of access points which are wirelessly connected. With WiMAX technology and OSPF routers, the wireless back-haul delivers each packet from an ingress access point to an egress access point. The topology of the whole network is known and each access point is fixed. We mainly consider the aggregated traffic from an ingress node (source) to an egress node (destination). Since each access point is stationary, the main factor that affects the capacity of each link is slow shadowing, whose coherence time interval is on the order of seconds [KqL99]. Similar to cellular networks, such wireless back-haul network is assumed to have a certain signalling mechanism with which the channel condition of each link can be sampled at a sufficiently higher frequency than that of the shadowing. Therefore, we can get the long-term average capacity of each link.

We aggregate the traffic from a source to a destination as the traffic demand between these two nodes, and the average aggregated source-destination traffic is assumed to be Poisson and, at least, quasi-stationary. According to the channel allocation policy and the actually coding and modulation scheme, the time-average link capacity can be estimated with measurements. All the access points function as OSPF routers, and are able to find multiple shortest paths with Dijkstra algorithm. Based on the subnet mask of the destination's IP address, if there are multiple paths for a flow, each intermediate access point is able to evenly distribute traffic among the paths. This capability can be easily implemented as follows: If m paths use the same link(i, j), the access point i only needs to add the same entry in form of $<$ submask$_{dst}$, link(i, j) $>$ m times in its forwarding table and splits the traffic for the destination in a weighted round robin way. Similarly, the access point can tear down one of these m paths by removing an entry $<$ submask$_{dst}$, link(i, j) $>$ from its forwarding table. $<$ submask$_{dst}$, link(i, j) $>$ in the forwarding table means that flow goes to destination submask$_{dst}$ should be forwarded to link(i, j).

We let a digraph $G = (V, E)$ represent a wireless back-haul network, where V is the set of nodes and E is the set of links. α denotes the maximal link utilization of the entire network and c_{ij} is the measured time-average capacity of link (i, j). Let K be the set of traffic flows in the network and d_k be the traffic demand of the k^{th} flow. Demand k, d_k, can also be denoted as d^{sd} assuming the source-destination pair of the k^{th} flow is (s, d). Let X_{ij}^k ($0 \leq X_{ij}^k \leq 1$) represent the percentage of demand k routed over link (i, j). Given the traffic between source s and destination d, $N(s, d)$ is the set of shortest paths from s to d and N_{sd} is the norm of $N(s, d)$.

Node i has a set of outgoing links (i, j) ($(i, j) \in E$) connecting to its one-hop neighbors. Given the traffic demand of each outgoing link (i, j), the link may have a number of subchannels that are spare in the sense that these subchannels are surplus with the respect of the traffic demand of the link. We assume that each node only controls the subchannel allocation of all the spare links originating from itself and the subchannel allocation is restricted to the node that is the owner of the subchannels, which means we do not consider the inter-node subchannel borrowing in this work.

3.2 Long-term Near-optimal Routing

Although the capacity of wireless links is dynamic, we can perform a long-term optimal routing based on the time-average capacity of each link as well as the aggregate traffic demands. As a result, when the variance of link capacity is small, we can achieve a good load balancing. Since this optimization is performed on large time scale (say, tens of minutes) or can be triggered when the network has significant changes (*e.g.*, link failure), we call it the long term optimal routing and treat it as the first tier optimization module.

3.2.1 The Primal Algorithm

The problem that the primal-dual algorithm trying to solve is that: given a set of traffic demands, how to set the link weights in order to minimize the congestion level of whole network. The traffic demands here are average aggregate source-destination traffic. In [Wan01], it has been proven that, for a given objective, the optimal routing can be reproduced through the shortest path routing based on certain positive link weights, which can be derived through solving the dual problem of the optimal routing problem. According to the theorem, we first model the primal problem with objective to minimize the maximum congestion level over the network. The optimal primal solution gives us the optimal routes. Based on the primal problem, we can derive the dual problem of it. Then, link weights can be easily calculated from the dual problem. This enables us to use OSPF-based access points to build a wireless back-haul network.

We assume that the traffic demands between source-destination pairs are given. Let s_k, t_k be the source node, and the destination node of the k^{th} flow respectively. Then the primal formula for our load balancing problem is as follows:

Primal

$$\min \alpha + r \sum_{k \in K} \sum_{(i,j) \in E} X_{ij}^k \tag{3.1}$$

s.t.

$$\sum_{j:(i,j) \in E} X_{ij}^k - \sum_{j:(j,i) \in E} X_{ji}^k = 0, \quad i \neq s_k, t_k \tag{3.2}$$

$$\sum_{j:(i,j) \in E} X_{ij}^k - \sum_{j:(j,i) \in E} X_{ji}^k = 1, \quad i = s_k \tag{3.3}$$

$$\sum_{j:(i,j) \in E} X_{ij}^k - \sum_{j:(j,i) \in E} X_{ji}^k = -1, \quad i = t_k \tag{3.4}$$

$$\sum\nolimits_{k \in K} d_k X_{ij}^k \leq c_{ij}\alpha, \quad (i,j) \in E \tag{3.5}$$

$$0 \leq X_{ij}^k \leq 1 \tag{3.6}$$

Function (3.1) is the objective function. The second term $r \sum_{k \in K} \sum_{(i,j) \in E} X_{ij}^k$ ensures that the optimization not only minimizes α, but also avoids unnecessarily long paths. The length of a path, $\sum_{k \in K} \sum_{(i,j) \in E} X_{ij}^k$, depends on the number of hops. r is a small positive number which ensures that the minimization of α takes higher priority. Constraints (3.2), (4.10), (3.4) indicate the flow balance equations [Wan01]. Constraint (3.5) requires that the congestion level on any link should not exceed that of the most congested link.

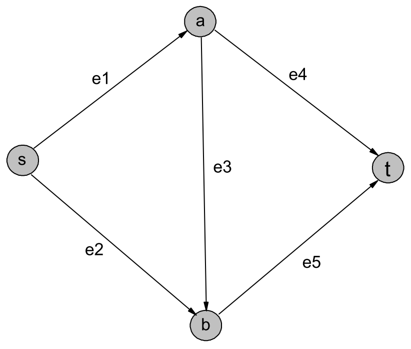


Figure 3.2: Network topology

Since the primal problem is linear programming with continuous real variables, it can be solved in polynomial time. The optimal solution of primal gives a route or a set of routes for each demand. It also gives out the proportions of each demand on different routes if that demand has to be split. Let $\{\bar{X}_{ij}^k\}$ and $\bar{\alpha}$ be the optimal solution of primal. If the routers are capable of arbitrary splitting, which is not the case for OSPF routers, then we can easily reach the optimal congestion level of the network by assigning the percentage of demand to each link according to $\{\bar{X}_{ij}^k\}$.

3.2.2 The Dual Problem

Since it is expensive to modify current OSPF routers to make them support arbitrary splitting [Vil99], the result of primal is hard to be fully utilized. Fortunately, since these optimal routes of primal can be reproduced as shortest paths by setting appropriate link weights which are obtained from the optimal solution of the dual problem, we can rely on the linear programming duality theory [Chv83] to get the following dual of the primal problem.

Dual

$$max \sum\nolimits_{k \in K} U_{t_k}^k \tag{3.7}$$

s.t.

$$U_j^k - U_i^k \leq d_k V_{ij} + r, k \in K, (i,j) \in E \tag{3.8}$$

$$\sum\nolimits_{(i,j) \in E} c_{ij} V_{ij} = 1, (i,j) \in E \tag{3.9}$$

$$V_{ij} \geq 0,\ U_{s_k}^k = 0 \tag{3.10}$$

Now Let us discuss the details of the derivation of Dual problem

Suppose a network has m nodes and n links. Define the (node-arc) incidence matrix A($m \times n$ matrix) similar to [and98]. For example, if there's a network with the topology as shown in Figure(3.2.1). Then the incidence matrix of the network is

$$A = \begin{array}{c} \\ s \\ t \\ a \\ b \end{array} \begin{array}{c} \begin{array}{ccccc} e_1 & e_2 & e_3 & e_4 & e_5 \end{array} \\ \begin{bmatrix} +1 & +1 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & -1 \\ -1 & 0 & +1 & +1 & 0 \\ 0 & -1 & -1 & 0 & +1 \end{bmatrix} \end{array}$$

We denote the routing probability vector of demand k as X^k, which is a n dimensional column vector. Now, we rewrite the Primal as follows:

$$\min \alpha + r \sum\nolimits_{k\in K} \sum\nolimits_{(i,j)\in E} X^k_{ij} \tag{3.11}$$

s.t.

$$AX^k = B^k, k \in K \tag{3.12}$$

$$\sum\nolimits_{k\in K} d_k X^k_{ij} \le c_{ij}\alpha, \quad (i,j)\in E \tag{3.13}$$
$$0 \le X^k_{ij} \le 1$$

We use Lagrange Multipliers U^k(m dimensional row vector) and W(n dimensional row vector) to get the dual. G is a n dimensional row unit vector.

$$\begin{aligned}
&\inf \ \ \alpha + r\sum_K\sum_E X^k_{ij} + \sum_K U^k(B^k - AX^k) \\
&\qquad + \sum_{(i,j)\in E} W_{ij}(\sum_K d_k X^k_{ij} - c_{ij}\alpha) \\
&= \inf \ \sum_K U^k B^k + \alpha(1 - \sum_{(i,j)\in E} c_{ij}W_{ij}) + \\
&(r\sum_K GX^k + \sum_K \sum_{(i,j)\in E} d_k W_{ij} X^k_{ij} - \sum_K U^k AX^k) \\
&= \inf \ \sum_K U^k B^k + \alpha(1 - \sum_{(i,j)\in E} c_{ij}W_{ij}) \\
&\qquad + \sum_K (rG + d_k W - U^k A)X^k
\end{aligned}$$

So, we have

$$\sum_{(i,j)\in E} c_{ij}W_{ij} = 1$$

$$rG + d_k W - U^k A \ge 0, k \in K$$

$$U^k A \le rG + d_k W, k \in K$$

$$U^k_j - U^k_i \le r + d_k W_{ij}, k \in K, (i,j) \in E$$

$$max \sum\nolimits_{k \in K} U^k B^k$$

s.t.

$$U_j^k - U_i^k \leq d_k W_{ij} + r, k \in K, (i,j) \in E$$

$$\sum\nolimits_{(i,j) \in E} c_{ij} W_{ij} = 1, (i,j) \in E$$

$$W_{ij} \geq 0,\ U_{s_k}^k = 0$$

Therefore, we get above dual. Note that $U^k B^k = U_{t_k}^k - U_{s_k}^k$ and $U_{s_k}^k = 0$, so $U^k B^k = U_{t_k}^k$, t_k is the destination node of demand k, then we get the dual (3.7) as described before.

Theorem (Strong LP Duality): Whenever both the primal and the dual LP are feasible, they have optimal solutions with equal value of their objective function.

This theorem holds for our primal and dual problems. Let $\{\bar{U}_i^k\}$ and $\{\bar{W}_{ij}\}$ be the optimal solution of the dual. Any path from s_k to t_k determined by the primal optimal solution is a shortest path with respect to link weights $\bar{w}_{ij} = d_k \bar{W}_{ij} + r$. Different from the formulation in [Wan01], we found that weight $\bar{w}_{ij} = d_k \bar{W}_{ij} + r$ actually depends on each demand. This requires the routers to be able to set the network link weights on per-flow basis.

[Wan01] modified the constraint (3.8) as follows:

$$U_j^k - U_i^k \leq d_k W_{ij} + d_k r, k \in K, (i,j) \in E$$

then define $U^k = U^k / d_k$, we have

$$U_j^k - U_i^k \leq W_{ij} + r, k \in K, (i,j) \in E$$

Therefore, weight $w_{ij} = W_{ij} + r$. This avoids the requirement of refreshing the network link weights on per-flow basis, but the strong duality does not hold for the dual in [Wan01] due to above relaxation. For example, given the network topology shown as in Figure 3.2.2, suppose there are two source-destination pairs (1,11) and (2,12), the traffic demands are 5Mb and 4Mb respectively. The value of r is set to 0.005. The objective function of primal problem is 0.204788 which is the same as the objective function of dual problem we derived. The objective function of dual problem in [Wan01] is 0.316635.

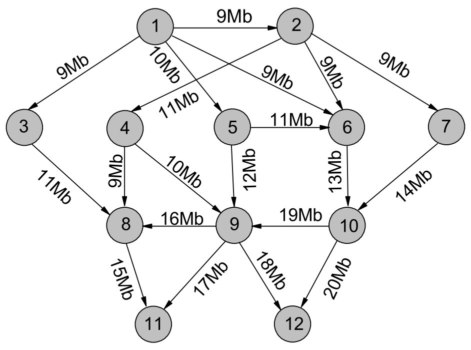


Figure 3.3: Network topology and long-term average link capacity

If the router is powerful enough to set the link weights on per-demand basis, it is possible for us to achieve the optimal routing policy obtained from optimal primal solution. However, if it is not the case, we propose to use the link weights determined by the maximum traffic demand of the demand set. Specifically, we define the network link weights as:

$$w_{ij} = d_{\bar{k}} \bar{V}_{ij} + r \tag{3.14}$$

where $d_{\bar{k}}$ is the maximum demand among the demand set. The reason of using the maximum traffic demand is that the link weights of maximum demand is an upper bound of $U_j^k - U_i^k$ for all the demands in constraint (3.8). It can be shown in the following inequality:

$$U_j^k - U_i^k \leq d_{\bar{k}} V_{ij} + r, k \in K, (i,j) \in E \tag{3.15}$$

Therefore, for this relaxed dual, it has the same optimal solution as the dual problem (3.7) when all the demands are the same. Unlike the relaxed dual in [Wan01], the value of r will not affect the gap between our relaxed dual problem and the original dual problem (3.7). As we mentioned before, when $r = 0$, the relaxed dual in [Wan01] has the same optimal solution as the dual problem (3.7). As r increases, the the gap between the relaxed dual in [Wan01] and the original dual (3.7) will be larger. When $r > 0$, if all the sources have similar traffic demands, our relaxed dual has a smaller gap than that of the dual in [Wan01]. In our study, we assume the difference between the source-destination traffic demands is usually small, and moreover, we prefer shorter path length by setting a reasonable large value of r. Thus, we adopt the relaxed dual specified by equations (3.14) and (3.15).

3.2.3 Greedy MIN-MAX Congestion Algorithm

A shortest-path set is generated under the link weights obtained from the dual problem described in the previous subsection. This set is used to approximate the optimal congestion level obtained by arbitrary traffic splitting. Since current OSPF routers only support discrete splitting (*i.e.*, it can only be slitted in the portion of $\frac{1}{n}$, where n is the possible cardinality of the path set), it has been shown in [Sri03] that finding the optimal splitting can be reduced to the 3-D Matching problem [and79], which is NP-hard. Thus, we propose a polynomial-time approximation algorithm called Greedy Min-max congestion algorithm (GMC). Different from [Sri03], instead of equally splitting the traffic at every router on the way to the destination, our algorithm tries to find a near-optimal traffic splitting in the view of all the shortest paths between the source and destination.

First of all, we need to spit each common link by the number of different paths in a shortest-path set that share the link. For example, given the network topology shown in Figure 1, based on the link weights by solving the dual problem, for flow $1 \rightarrow 11$, the shortest-path set is path(1)=(1-3-8-11), path(2)=(1-2-4-8-11), path(3)=(1-2-4-9-11), path(4)=(1-5-

9-11), and path(5)=(1-6-10-9-11). As can be seen, path(1) and path(2) share link(8, 11). As we discussed in Section 3.1, node 8 adds the entry $< 11, \text{link}(8, 11) >$ twice in its routing able. By distributing the traffic of flow $1 \rightarrow 11$ according to the routing table, node 8 can equally split the traffic to both path (1) and path (2). This rule applies to all the other shared link.

In our approach, the source nodes are sorted in the sequence of decreasing order of the demand of incoming traffic in the network. Then the GMC executes at each source node in this order. Formally, GMC algorithm tries to choose a subset of $N(s, d)$, denoted by M, to achieve

$$\min \max_{(i,j)\in E} \left(\frac{l_{ij} + \frac{d^{sd}}{\|M\|}}{l_{ij} + d^{sd}\bar{X}_{ij}^{sd}}\right) \tag{3.16}$$

where, $\bar{X}_{ij}^{sd}$ denote the optimal primal solution for each source-destination pair (s, d). l_{ij} denotes the current load of link (i, j) before making actual assignment.

The GMC algorithm runs in two steps:

Step 1: For each index $t = 1, 2, ..., N_{sd}$, we perform a *virtual assignment* to a set of t paths to achieve the objective in equation (3.16). This can be achieved as follows:

1. By setting $\|M\| = t$, find the link of each path $n = 1, 2, ..., N_{sd}$ with the maximum value of equation (3.16). We denote the link index of such link of path n under index t as $n(t)$. Thus the load of link $n(t)$ is $l_{n(t)}$.
2. With the virtually assigned demand, sort each path in $N(s, d)$ in an increasing order according to

$$\frac{l_{n(t)} + \frac{d^{sd}}{t}}{l_{n(t)} + d^{sd}\bar{X}_{n(t)}^{sd}}, n \in N(s, d) \tag{3.17}$$

   Then re-index them and virtually assign d^{sd} only to the first t paths.

Step 2: From N_{sd} possible assignments, choose the one with the smallest value of object function (3.16).

Suppose there are two flows $1 \rightarrow 11$ and $2 \rightarrow 12$, the traffic demands are 5Mb and 4Mb respectively. With GMC algorithm, the final path set for flow $1 \rightarrow 11$ is path(1), path(2),

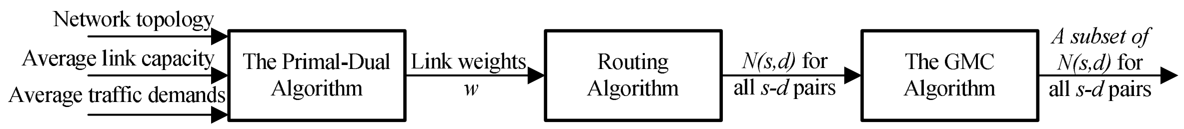


Figure 3.4: System Diagram of First Tier

path(3), path(4) and path(5). For flow $2 \rightarrow 12$, the shortest path set is path(1)=(2-6-10-12), path(2)=(2-7-10-12), path(3)=(2-4-9-12) while the final path set of GMC is path(1) and path(2). The optimal congestion level of Primal $\bar{\alpha}$ is 0.173. The maximum congestion level of the GMC algorithm under the max-demand link weights obtained from the Dual is 0.222 and the maximum congestion level under the link weights obtained from dual in [Wan01] is 0.400. The system diagram of first tier is shown in Figure 3.4.

3.3 The Short-term Adaptive Routing

3.3.1 The Self-adaptive Routing Algorithm

Unlike the wired links, the condition of wireless links is dynamic in nature, which makes it insufficient to select the routing path set for each flow based on long-term average of channel capacity. Therefore, we propose a short-term adaptation scheme which runs frequently enough to refine the routing policy to balance the overall traffic according to the channel condition. Our adaptation algorithm is motivated by stochastic control theory [Gal77, Ber84, Kae96], and runs only at each source node. Our algorithm is asynchronous, so there's no requirement for synchronization between the source nodes. The channel condition of every link consisting a path is feeded back in a reverse direction, namely, from the destination node to the source node. Each node updates to it's previous node in the labelled shortest paths once each update period. The feedback information is used by the source node to get the maximum link congesting level along each shortest path. For the source node, it also maintains the following parameters.

- Internal probability q_i^{sd} from source s to destination d through path i. Later we will show this set of probability converges to Cesaro-Wardrop equilibrium [J.N86] in the path set.
- Internal routing probability p_i^{sd} from source s to destination d through path i. It is this set of probability that the actual routing adaptation approximates to.
- The most recently reported maximum link utilization δ_i^{sd} of the shortest path i from source s to destination d.
- The link load l_i of the most congested link through path i.
- The actual routing decision r_i^{sd} from source s to destination d through path i.

Each source node runs the adaptation algorithm once during a certain time interval, which is called updating period. When source receives an update of path i, it updates δ_i^{sd} and l_i. Then it computes the updated congestion level for path i according to the following equation:

$$\alpha_i^{sd}(n+1) = (1 - a(n))\alpha_i^{sd}(n) + a(n)\delta_i^{sd} \tag{3.18}$$

where $a(n) \in [0, 1]$ is the learning factor of congestion level. If it is large, it means α_i^{sd} is sensitive to the noise. If it is small, α_i^{sd} depends mainly on the previous sample. However, in wireless network, $a(n)$ can not be too small because if it is too small, it can not be responsive enough to the dynamic wireless channel condition.

Then, the source node computes its overall congestion level according to

$$\alpha^{sd}(n+1) = \sum_{i \in N(s,d)} p_i^{sd}(n)\alpha_i^{sd}(n+1) \tag{3.19}$$

where $N(s, d)$ is the labelled shortest-path set from the source node s to the destination node d. After the source node gets the overall congestion level, it updates the internal probability for each path in $N(s, d)$ according to following equation:

$$q_i^{sd} = q_i^{sd}(n) + b(n)[q_i^{sd}(\alpha^{sd}(n+1) - \alpha_i^{sd}(n+1)) + \xi_i^{sd}] \tag{3.20}$$

Where $b(n) > 0$ is routing learning factor and it is used to smooth out the noise. ξ_i^{sd} is i.i.d random vector distributed uniformly on the unit ball of dimension $N(s,d)$. It is used to add disturbance to avoid non-Wardrop solutions.

Then, source node projects q_i^{sd} to $[0,1]^{N(s,d)}$ to ensure that the internal probability is a valid set. It get the new internal probability by solving the optimum problem below:

$$\min \sum_{i \in N(s,d)} (x_i - q_i^{sd})^2 \tag{3.21}$$

s.t.

$$\sum_{i \in N(s,d)} x_i = 1$$

$$0 \leq x_i \leq 1, \;\; i \in N(s,d)$$

The above projection can be solved following

$$x_i = q_i^{sd} + \frac{1 - \sum\limits_{i \in N(s,d)} q_i^{sd}}{N_{sd}} \tag{3.22}$$

Then, we update the new internal routing probability p based on q:

$$p_i^{sd}(n+1) = (1-\epsilon) q_i^{sd}(n+1) + \frac{\epsilon}{N_{sd}} \tag{3.23}$$

where ϵ is a small constant number.

Finally, since each router supports equal splitting in the path set, we propose an algorithm to compute real routing probability r_i^{sd} according to the internal routing probability p_i^{sd}. The principle is to approximate the congestion level under the actual routing probability r_i^{sd} to the congestion level under the internal routing probability p_i^{sd}. We can compute the actual routing probability in every updating period or compute it every several updating periods, which depends on the requirement of responsiveness.

The algorithm for calculating the actual routing probability r_i^{sd} is as follows:

Step 1: Calculate l_{ri} on the most congested link of each path according to following equation, where l_i and r_i^{sd} are the current link load and current actual routing probability respectively.

$$l_{ri} = l_i - d^{sd} r_i^{sd} \tag{3.24}$$

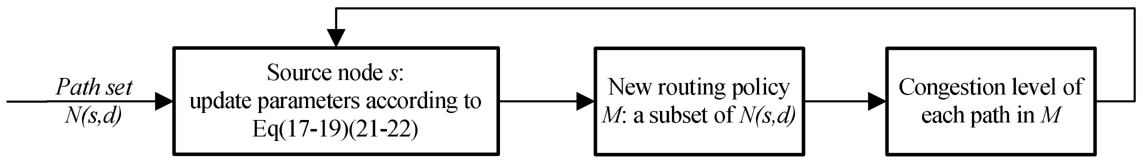

Path set
N(s,d)
Source node s: update parameters according to Eq(17-19)(21-22)
New routing policy M: a subset of N(s,d)
Congestion level of each path in M

Figure 3.5: System Diagram of Second Tier

Step 2: Sort load ratio

$$\frac{l_{ri} + \frac{d^{sd}}{t}}{l_{ri} + d^{sd} p_i^{sd}}, \ (t = 1, 2, , N_{sd})$$

in increasing order, virtually assign d^{sd} only to the first t paths. The denominator is the load determined by internal routing probability p_i^{sd}. The numerator is load determined by equal splitting.

Step 3: From all N_{sd} possible assignments, choose the one with the smallest maximum for an actual assignment.

Step 4: After running the above algorithm, we may get a new paths set. However, due to equal splitting, the granularity of the change of traffic on corresponding links is greater or equal to $1/N_{sd}$, the newly calculated actual routing probability is not necessarily able to reduce the congestion level. Therefore, our last step of the short-term adaptation algorithm is to find the best decision by comparing the congestion level under the previous actual routing probability, the newly generated paths set, and the routing policy obtained from the first tier module. Note that the routing policy of the first tier is useful to prevent divergence of adaptation due to lack of accurate prediction of the channel condition[1]. Suppose the final path set as M, the actual routing probability r_i^{sd} is updated as follows:

$$r_i^{sd} = \begin{cases} \frac{1}{\|M\|}, & i \in M \quad (3.25a) \\ 0, & i \notin M \quad (3.25b) \end{cases}$$

The system diagram of second tier is shown in Figure 3.5.

[1]Through our study, we found that the divergence might last for several updating periods until the estimation of link utilization converges under stationary routing probability.

3.4 The Dynamic Subchannel Assignment

3.4.1 Problem Formation

Suppose the link (i,j)'s bandwidth C_{ij} is composed of two parts: W_{ij} and B_{ij}, where W_{ij} is the bandwidth of the traffic demand of link (i,j) and B_{ij} is the spare bandwidth with the respect of the traffic demand. Let O_i denote the set of outgoing links of node i, where $O_i = \{(i,j) \in E\}$ and $|O_i|$ is the cardinality of O_i. Suppose A_i represents the set of spare subchannels at node i and the bandwidth of the k^{th} spare subchannel is denoted as $D_i^k, i \in A_i$. Let D_{max} and D_{min} be the maximum and minimum bandwidth among all the spare subchannels respectively. We denote $x_{ij}^k((i,j) \in E, k \in A_i)$ the indicator that shows if subchannel k is assigned to link (i,j) or not. Thus, we have $x_{ij}^k = \{0,1\}$. Our objective is to minimize the maximal congestion level among all the outgoing links at node i by reassigning the spare subchannels A_i to O_i. Let $\hat{C}_{ij}$ denote the new bandwidth of link (i,j) after subchannel assignment. Let β_i be the maximum link utilization among the outgoing links of node i after subchannel assignment, that is $\beta_i = \max\{W_{ij}/\hat{C}_{ij}, (i,j) \in O_i\}$. Then, we formalize the problem as follows: For every node i in the network, we have

$$\min \beta_i \tag{3.26}$$

s.t.

$$W_{ij} + \sum_{(i,j)\in O_i, k \in A_i} x_{ij}^k D_i^k = \hat{C}_{ij}, \quad (i,j) \in O_i \tag{3.27}$$

$$\sum_{(i,j)\in O_i} C_{ij} = \sum_{(i,j)\in O_i} \hat{C}_{ij} \tag{3.28}$$

$$W_{ij} \leq \beta_i \hat{C}_{ij}, \quad (i,j) \in O_i \tag{3.29}$$

$$\sum_{(i,j)\in O_i} x_{ij}^k = 1, \quad k \in A_i \tag{3.30}$$

$$x_{ij}^k = 0 \; or \; 1, \quad (i,j) \in O_i, k \in A_i \tag{3.31}$$

(3.26) is the objective. Equation (3.27) is the definition of $\hat{C}_{ij}$, which is the total bandwidth of link (i,j) after subchannel assignment. Equation (3.28) ensures the total bandwidth at node i remains the same before and after subchannel assignment.Constraint (3.29) gives the definition of β_i. Equation (3.30) requires that each spare subchannel is allocated to only one outgoing link of node i. Equation (3.31) says the variables x_{ij}^k are the integer of either 0 or 1.

The optimization problem is in the form of mixed-integer non-linear programming problem, which is NP-hard in general [and79]. Therefore, it is of practical importance to design an approximation algorithm to find the sub-optimal solution of the formulated problem. Our approach to solve this problem is as follows. We first explore a lower bound for the objective, which can be obtained by relaxing the integer value of $x_{ij}^k, (i,j) \in O_i, k \in A_i$. Using this lower bound as a performance benchmark, we then develop a greedy subchannel allocation algorithm to find the sub-optimal solutions.

3.4.2 A Lower Bound For The Objective

By relaxing the integer requirement on $x_{ij}^k = \{0,1\}$ with $0 \le x_{ij}^k \le 1$, the aforementioned optimization problem can be rewritten as the following model: For each node i,

$$\min \max_{(i,j)\in O_i} \frac{W_{ij}}{W_{ij} + \sum\limits_{k\in A_i} x_{ij}^k D_i^k} \tag{3.32}$$

s.t.

$$\sum_{(i,j)\in O_i} x_{ij}^k = 1, \quad k \in A_i \tag{3.33}$$

$$x_{ij}^k \in [0,1], \quad (i,j) \in O_i, k \in A_i \tag{3.34}$$

Based on the relaxed model, we can compute the lower bound of the original integer optimization problem. In particular, the optimal value of objective function (3.32) can be obtained by:

$$\beta_i^{opt} = \frac{\sum_{(i,j)\in O_i} W_{ij}}{\sum_{(i,j)\in O_i} W_{ij} + \sum_{k\in A_i} D_i^k} \tag{3.35}$$

Therefore, the optimal spare bandwidth of link (i,j), denoted by B_{ij}^{opt}, can be calculated by:

$$B_{ij}^{opt} = \frac{W_{ij}}{\beta_i^{opt}} - W_{ij} \tag{3.36}$$

3.4.3 The Greedy Subchannel Allocation Algorithm

Using the derived lower bound of the maximal congestion level of node i as the performance benchmark, we develop a *greedy subchannel allocation* (GSA) algorithm. This algorithm is executed at each node i once every specified time period (say 200 ms). The details of the algorithm are as follows:

Step 1: Determine the set A_i. For each link (i,j) ($(i,j) \in O_i$ and $W_{ij} > 0$), set the initial value of B_{ij} to B_{ij}^{opt}. Initialize $x_{ij}^k = 0$ ($(i,j) \in O_i, W_{ij} > 0, k \in A_i$) and set the state of link (i,j) to active .

Step 2: If there is only one active link left, go to step 5. Else, for each active link (i,j), calculate $\frac{B_{ij}}{D_i^k}$ for all the spare subchannels D_i^k ($k \in A_i$) and sort the sequence of $\{\frac{B_{ij}}{D_i^k}\}$ in an increasing order, in which ties can be broken arbitrarily. Compare the first element of the sequence of each active link and pick the one with the largest value. Suppose the resulting link and subchannel are (i,j') and k', respectively.

Step 3: If the element is greater than or equal to 1, go to step 4. Else compare $\left|\frac{B_{ij'}}{D_i^{k'}} - 1\right|$ and $\left|\frac{B_{ij'}^*}{D_i^*} - 1\right|$, where D_i^* is the bandwidth of the previous subchannel assigned to link (i,j')

and $B^*_{ij'} = B_{ij'} + D^*_i$. If the former is smaller than the latter, go to step 4. Else, deactivate link (i, j'), go to step 2.

Step 4: Assign the subchannel k' to link (i, j') and set $x^{k'}_{ij'} = 1$. Update $B_{ij'} = B_{ij'} - D^{k'}_i$. Remove subchannel k' from A_i. If the element in step 3 is equal to 1, deactivate link (i, j'). Go to step 2.

Step 5: Assign the active subchannel to the link and terminate the algorithm.

We further explain the algorithm with an example under a simple scenario. Suppose there are two outgoing links from node 1, link (1,1) and link (1,2). Suppose $W_{11} = 10Kbps$, $W_{12} = 40Kbps$. Both link (1,1) and link (1,2) have one spare subchannel with the bandwidth of $40Kbps$ and $10Kbps$, respectively. Therefore, $A_1 = \{10Kbps, 40Kbps\}$. We can calculate current link congestion level following $\alpha_{11} = \frac{W_{11}}{W_{11}+B_{11}} = \frac{10}{10+40} = 0.2$. $\alpha_{12} = \frac{W_{12}}{W_{12}+B_{12}} = \frac{40}{40+10} = 0.8$. Therefore, $\beta_1 = 0.8$. Based on the definition of β^{opt}_i and B^{opt}_{ij}, we have $\beta^{opt}_1 = \frac{10+40}{10+40+40+10} = 0.5$, $B^{opt}_{11} = 10Kbps$ and $B^{opt}_{12} = 40Kbps$.

Now we apply our greedy subchannel allocation algorithm: First, initialize $B^{opt}_{11} = 10Kbps$ and $B^{opt}_{12} = 40Kbps$. Set $x^k_{1j} = 0$ $(j, k = 1, 2)$. Activate all the links, link(1,1) and link(1,2) . Then calculate and sort the sequence $\{\frac{B_{1j}}{D^k_1}\}$ $(j, k = 1, 2)$ in an increasing order. The sequence of link $(1, 1)$ is (0.25,1) and that of link $(1, 2)$ is (1,4). Comparing the first element of the two sequences, we pick the larger one. Following steps 3 and 4, we assign the corresponding subchannel $D^2_1 = 40Kbps$ to the link (1,2) and update $B_{12} = 40 - 40 = 0$. Since $B_{12} = 0$, link (1,2) is deactivated. In the next round, the other sub-band $D^1_1 = 10Kbps$ is assigned to link (1,1).

3.5 Performance Analysis

In this section, we give theoretical analysis of the proposed algorithms and show their properties in terms of convergence and optimality bound. We make following assumptions in our analysis, which are widely used in two time-scale stochastic iterative algorithms [Bor97, Bor00, Tad03].

1. The Feller property holds which means the update is frequent enough compared with the network changing rate.

2. The learning factor of congestion level $a(n)$ satisfies:

$$\forall n: \; a(n) \geq a(n+1) > 0$$

$$\sum_n a(n) = \infty, \; \sum_n a(n)^2 < \infty$$

$$\sum_n (\frac{a(n) - a(n+1)}{a(n)})^r < \infty$$

   for some $r \geq 1$.

3. Assume $b(n)$ factors used in internal probability update satisfies:

$$\forall n: \; b(n) \geq b(n+1) > 0$$

$$\sum_n b(n) = \infty, \; \sum_n b(n)^2 < \infty$$

$$\sum_n (\frac{b(n)}{a(n)})^s < \infty$$

   for some $s \geq 1$.

Assumptions 2 and 3 are essential to guarantee the convergence of stochastic iterative algorithms [D.P96]. Learning factor $a(n)$ is the step size of updating congestion level while $b(n)$ is that of updating routing probability. The sum of them should be unbounded in order to reach equilibrium. $\sum\limits_n a(n)^2 < \infty$ and $\sum\limits_n b(n)^2 < \infty$ ensure that the variance of estimation of congestion level and routing probability are bounded. Thus, the decreasing step sizes in the assumptions guarantee that the adaptation algorithm converges to the solution almost surely. We give the properties of our scheme through the following theorems:

Theorem 1 *The GMC algorithm achieves a load that is no more than* $(1 + \ln N_{sd})$ *times the load determined by optimal solution* $\{\bar{X}_{ij}^{sd}\}$ *of primal.*

Theorem 2 *The GMC algorithm achieves a maximum congestion level that is no more than* $(1 + \ln N_{sd})$ *times the optimal congestion level* $\bar{\alpha}$ *of primal.*

Theorem 3 *The short-term adaptation algorithm achieves a load that is no more than* $(1 + \ln N_{sd})$ *times the optimal load with the adaptation algorithm determined by internal routing probability* p.

Theorem 4 *The short-term adaptation algorithm achieves a maximum congestion level that is no more than* $(1 + \ln N_{sd})$ *times the congestion level determined by internal routing probability* p.

Theorem 5 *With the short-term adaptation algorithm, the routing probability of each flow converges, the estimation of maximum link utilization converges under stationary routing probability, and the internal routing probabilities converge in* H_s *almost surely,* where H_s is the set $\{y : y_i^{sd} > 0 \Rightarrow \alpha_i^{sd} = \min_j \alpha_j^{sd}\}$.

The following theorem gives the approximation analysis of the GSA algorithm.

Theorem 6 *For node* i*, suppose the maximum congestion level achieved by the GSA algorithm is* β_i*, then we have:*

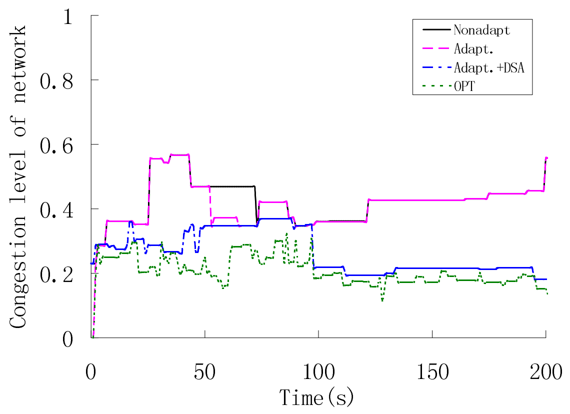
$$\left| \frac{1}{\beta_i} - \frac{1}{\beta_i^{opt}} \right| \leq (|O_i| - 1) \frac{\max(1, \frac{B_{max}^{opt}}{D_{\min}} - 1) D_{\max}}{W_{min}} \tag{3.37}$$

where $B_{\max}^{opt}$ is the maximum value of B_{ij}^{opt} $((i,j) \in E)$and W_{min} is the minimum value of $W_{ij}(W_{ij} > 0, (i,j) \in E)$.

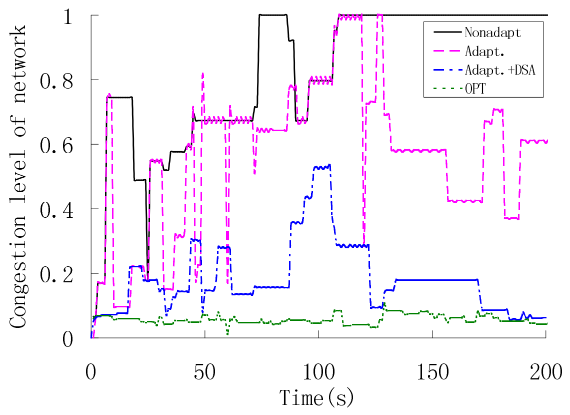
3.6 Performance Evaluation

We evaluate the performance of our scheme through simulations by using ns-2 [Pro]. The simulation topology and average link capacity is the same as those in Figure 1. Even though the setting of link capacity may not match some real networks, since we focus on load balancing, such setting does not affect the way to prove the effectiveness of the proposed scheme. The simulation duration is 200 seconds. There are two traffic sources and the source-destination pairs are still (1,11) and (2,12). Both traffic sources are exponentially distributed. The update period of the adaptive routing is 1 second. The parameter of the noise generator, denoted by std, controls the extent of variance of link capacity. The coherence time of shadowing is exponentially distributed with mean of 5 seconds. We compare the maximum congestion level of the routing scheme that only applies the first tier adaptation mechanism (denoted by Nonadapt), the routing scheme that uses both the first and second tier adaptation mechanisms (denoted by Adapt) and the routing scheme that employs all three tiers adaptation mechanisms (denoted by Adapt+DSA). The dynamic subchannel assignment is run once every 200 ms. We can set this period to different values as far as it is smaller or equal to the updating period of the short term adaptive routing, which is 1 second in our simulation. In addition, we give out the optimal congestion level (denoted by OPT) as the benchmark of the performance evaluation.

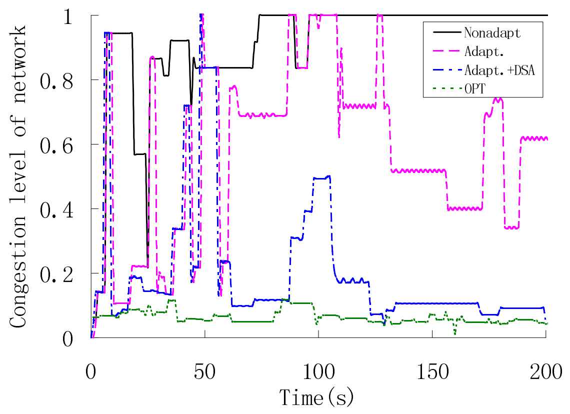
Let us first compare the congestion level between the Adapt routing and the Nonadapt routing. The simulation results using adaptation and without using adaptation when $d^{1,11} = 5$, $d^{2,12} = 4$ and $std = 1$ are shown in Figure 3.6 (a). The congestion level under Nonadapt is obtained by equally splitting among paths under long term average channel condition while the congestion level under Adapt is achieved by the adaptation algorithm based on the short term channel condition in the provided path set. We can see that when the std is small, which means the channel condition is relatively stable, the congestion level of Nonadapt is close to the congestion level of Adapt. This is because the link capacity used to compute the long term routing decision is the average link capacity, when channel is stable, the long term average link capacity is relatively closer to the short term link capacity. Therefore, with relatively stable channel condition, the benefit of short term adaptive mechanism is not



(a) $(d^{1,11}, d^{1,12}) = (5, 4)$,std=1



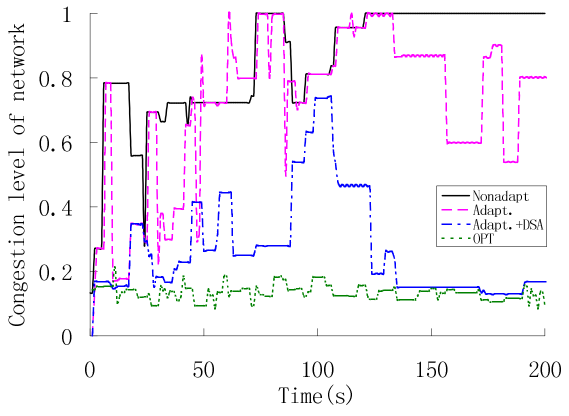
(b) $(d^{1,11}, d^{1,12}) = (1.5, 1)$, std=2.5



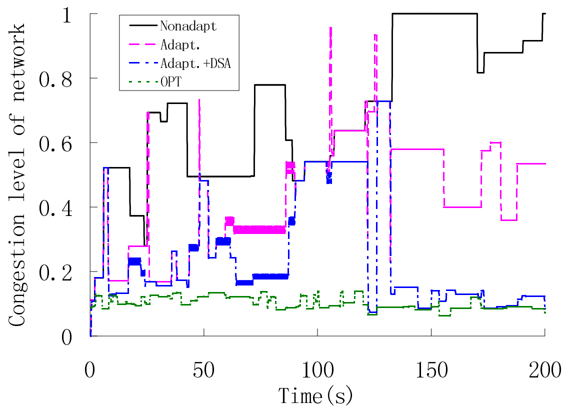
(c) $(d^{1,11}, d^{1,12}) = (1, 1)$, std=3.0

Figure 3.6: Performance of Our Scheme with different stds

quite significant. However, comparing the simulation results shown in Figures 3.6 (b) and 2(c), where $d^{1,11} = 1.5$, $d^{2,12} = 1$, $std = 2.5$ and $d^{1,11} = 1$, $d^{2,12} = 1$, $std = 3$, respectively, we can see that even the traffic demands are smaller than those in Figure 3.6 (a), due to a large variance of link capacity, the maximum congestion level with Nonadapt is very high (*i.e.*, close to 1). In these cases, we observe that when the short term adaptive routing is applied, which is the case of Adapt, the maximum congestion level is reduced significantly. This is because the short term adaptive routing will monitor the congestion levels among the multipath set and adjust traffic volume on each path to reduce the congestion level on the most congested path.



(a) $(d^{1,11}, d^{1,12}) = (3, 2)$, std=2.0



(b) $(d^{1,11}, d^{1,12}) = (2, 2)$, std=2.0

Figure 3.7: Performance of our scheme with std=2.0

The simulation results for $d^{1,11} = 3$, $d^{2,12} = 3$, std=2 and $d^{1,11} = 2$, $d^{2,12} = 2$, std=2 are shown in Figures 3.7 (a) and (b). First, look at the maximum congestion level with Nonadapt. It is obvious and easy to understand that when the std is the same, the larger the traffic demands are, the higher the congestion level is. Therefore, as shown in Figure 3.7 (a), with Nonadapt, the network is more congested than that in Figure 3.7 (b). For the maximum congestion level with Adapt, we can observe that it is significantly reduced from the case of Nonadapt. In summary, from all the simulation results, we can see that the Adapt routing scheme can lower the maximum congestion level significantly, especially when channel is more dynamic (*i.e.*, the std is large). We also notice that there are bursts when Adapt routing scheme is used. The reasons is that sometimes the updating process is not frequent enough to feedback the change of channel conditions. However, we can see that as the adaptation refines the routing policy quickly, and the bursts last very short time.

Next, we evaluate the effectiveness of the Adapt.+DSA routing scheme. Since the spare subchannels on each links as the consequence of the short-term adaptive routing have been carefully exploited, the maximum congestion level is greatly reduced. We can see from all the figures that the maximum congestion level under Adapt.+DSA is further reduced compared to those with Nonadapt and Adapt. In many cases especially when the channel condition is relatively stable, it is very close to the optimal congestion level (OPT), which is obtained by solving the primal linear programming problem in section 3.2. This OPT is the realtime optimal solution under given network topology, realtime link capacity and traffic needs. Please note that it is impractical to implement OPT since it requires a lot of communication resource to transfer the channel condition of each link to the central controller to calculate the optimal congestion level. The gap between Adapt.+DSA and OPT is caused by following reasons. (1) Optimal scheme explore all possible routes in the network and fully utilize the capacity. The routes of optimal scheme are always changing due to the dynamic wireless channel conditions. As we mentioned before, they are updated every 200 ms. The path set as the input of the adaptation algorithm is, however, fixed. It is obtained based on averaged link capacity and GMC algorithm as we introduced before. (2) The optimal congestion level is achieved through arbitrary splitting. Adaptive routing is based on equal splitting and the granularity of the change of traffic on corresponding links is greater or equal to $1/N_{sd}$. Even though the dynamic bandwidth allocation is used to improve adaptive routing, this is still a great factor for the gap of the two curves. (3) For the dual problem in section 3.2, we mentioned if the router is powerful enough to set the link weights on per-demand basis, we propose to use the link weights defined in equation (3.14), which is adopted in our simulation. This is also a reason for the gap between Adapt.+DSA and OPT since OPT is obtained directly from the optimal solution of the Primal.

CHAPTER 4

A NETWORK CODING SCHEME-RANC

In this chapter, we propose the *Relay-Aided Network Coding* (RANC) scheme in multi-hop wireless networks. In RANC, some nodes can transmit their native packets to their neighbors at high date rate. Then the receiving nodes XOR these packets with their own native packets, and transmit the coded packet to other nodes with a suitable rate. We investigate the design principles via rigorous analysis and derive the coding structure for RANC. We also analyze the tradeoff in the performance gain of RANC. Based on the analytical results, we design the RANC protocol by decomposing the original problem into two sub-problems: the flow partition problem and the scheduling problem. As a result, the set of flows in the network is partitioned into a number of subsets in the way that the overall cost is minimized. For each subset of flows, the scheduling algorithm is applied to determine the sequence of packet transmission, the corresponding transmission rate, and the information of each packet. We evaluate our design via simulations.

4.1 System Model and Motivation

4.1.1 System Model

An example of multi-hop wireless networks is shown in Figure 4.1. From the figure, it can be found that: for applications incurring symmetric traffic (e.g. P2P file sharing, multi-person online game), there may be a large number of flows which can benefit from network coding in the area where the node density is high. With appropriate routing algorithms (e.g. [SRB07]), such rich network coding opportunity can be exploited. By studying the topology closely, it can be seen that many network coding scenarios can be abstracted as a wheel topology depicted in Figure 4.1.

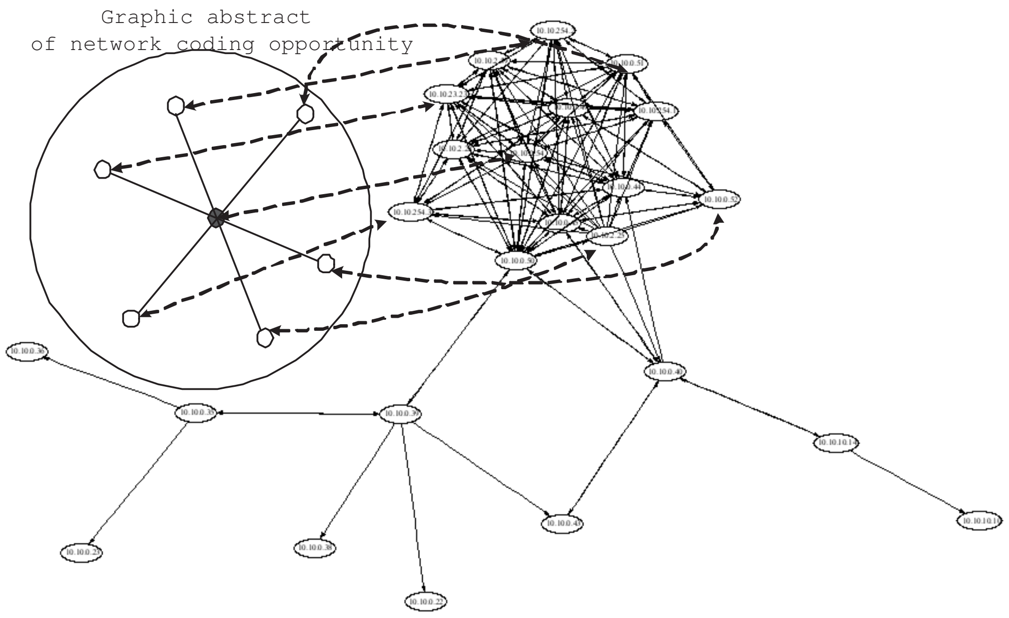


Figure 4.1: The topology of Berlin wireless mesh network [Fre] and the illustration of potential network coding opportunity

For simplicity, we focus on network coding under a wheel topology. We assume there are $n/2$ symmetric uni-cast flow pairs, where n is an *even* number and equal to the total number of participant nodes excluding the hub node. The node IDs are indexed by $0, 1, 2, ..., n-1$. $D(i)$ is denoted as the ID of the destination node of node i. There is a hub node at which all symmetric flows intersect and the packets of each flow need to be forwarded by the hub node. Similar to COPE, at an approximate time instance, the hub node performs XOR-based network coding for all the flows. We assume all nodes are static and the location of source and destination nodes is assumed to be known a priori. Since the network topology is abstracted as a circle, we use polar coordinate system to locate node i by the distance to the center (i.e. the hub node) and the polar angle, which are denoted by d_i and θ_i, respectively. We also denote d_{ij} as the distance between nodes i and j.

We denote P_i and C_i as the native packet from node i and the packet sent by node i, respectively. The XOR-ed packet generated by the hub node is denoted by P. Although network coding does not require the length of each packet to be same, to simplify the analysis, we assume all native packets in the network have the same packet length. As each node is able to adapt the transmission rate to the channel condition, r_i is the transmission rate used by node i and $d(r_i)$ is the distance threshold of r_i. In addition, we assume a fair medium access control (MAC) protocol which provides each node equal transmission opportunity is adopted in the network.

We use X_i^j $(j \neq i)$ to indicate if node i overhears packet C_j. Thus node i has a vector $\vec{X_i} = (X_i^0, X_i^2, ..., X_i^{n-1})$, where $X_i^i \equiv 1$. By the definition, all the packets node i received or overheard can be obtained by the inner product $\vec{X_i} \cdot \vec{C}^T$, where $\vec{C} = (C_0, ..., C_{n-1})$. The module-2 linear combination algorithm $L(\vec{P})$, where $\vec{P}$ is a vector of packets, is used to decode P. For each transmitted packet C_i, we define the coding length of C_i, denoted by $l(C_i)$, to be the number of native packets XOR-ed in C_i. For example, if $C_1 = P_1 \oplus P_2 \oplus P_3$, then $l(C_1) = 3$, and if $C_1 = P_1$, then $l(C_1) = 1$.

4.1.2 Motivation

The performance gain of network coding comes from few aspects. In the example shown in Figure 2.1, one packet transmission can be saved with network coding, which indicates that the *throughput gain* can be obtained by network coding. Furthermore, assume that all nodes continuously have some traffic to send (i.e., backlogged), but are limited by their MAC allocated bandwidth. Then the hub-node becomes the bottleneck since its bandwidth allocation is shared by its own traffic and the traffic to be forwarded. The Coding+MAC gain is the ratio of the bottleneck's draining rate with network coding to its draining rate without network coding. In the example shown in Figure 2.1, the *Coding+MAC gain* of the hub-node is equal to the coding length of P, which is equal to 2. In general, given n participant nodes that create a network coding opportunity, the Coding+MAC gain is maximized when the hub-node has the opportunity to XOR all n packets at a proper time and broadcast it to all the nodes. In order to ensure that each node i can decode the XOR-ed packet P sent from the hub-node, COPE [KRH+06] defines the coding rule as follows.

COPE's coding rule: to transmit n packets, $P_1, ..., P_n$ to nodes $0, ..., n-1$, the hub node can XOR the n packets together only if each node i has all $n-1$ packets P_j for $j \neq i$.

To meet this coding rule, it is necessary to let node i have all $n-1$ packets P_j for $j \neq i$. In fact, COPE simply asks node i to transmit its native packet P_i and Other nodes except for node $D(i)$ to operate in the promiscuous mode and overhear P_i. In the end, the hub node generates P by XOR-ing all overheard native packets, and transmits the coded packet to all nodes. If node i has successfully overheard all $n-1$ packets $P_j (j \neq i)$, it can decode the XOR-ed packet from the hub node. Otherwise, at least one extra transmission is needed to let node i get the desired packet. As a result, if there always exists a node $j(j \neq i, D(i))$ and the channel condition between nodes i and j is poor, in order to maximize the performance gain (especially the Coding+MAC gain when fair MAC is used), in COPE, node i has to transmit its native packet with a low speed, at which node j can successfully overhear the packet.

The observation above shows that, given network coding opportunities, maximizing performance gain of COPE is quite sensitive to the channel condition between node i and all other nodes except for node $D(i)$. However, such sensitivity can be alleviated according to the following observation. *As node i only needs to know the XOR-ed information of $n-1$ packets P_j ($j \neq i$) to decode P, it is not necessary for node i to overhear all individual native packets P_j ($j \neq i$).* For example, if the coding rule requires that $P_2 \oplus P_3 \oplus P_4$ should be present for node 4 to decode $P_1 \oplus P_2 \oplus P_3 \oplus P_4$ to get P_1, node 4 does not have to successfully overhear P_2 and P_3 one by one. Instead, if node 4 can successfully overhear $P_2 \oplus P_3$, it can still have $P_2 \oplus P_3 \oplus P_4$ by XOR-ing $P_2 \oplus P_3$ and P_4, and decode $P_1 \oplus P_2 \oplus P_3 \oplus P_4$. RANC is inspired by the above thought that *forming the a number of coded packets, whose coding length may not necessarily be equal to 1, to satisfy that the*

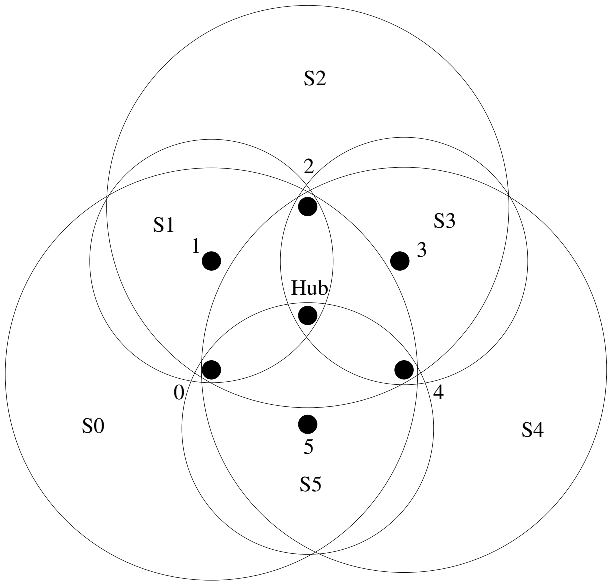


Figure 4.2: An illustration of how RANC works

linear combination of these packets is sufficient for decoding P. As a result, node i does not have to successfully overhear all native packet P_j ($j \neq i$) in order to decode P. In RANC, the nodes are allowed to transmit at different rates based on the channel condition. The packet that each node sends may not necessarily be the native packet. In stead, a packet could be a XOR-ed information of some packets overheard by the node. During each round of network coding, our design requires that each node only transmit once and thus it does not increase message complexity of network coding.

We give an example to illustrate how RANC works: As shown in Figure 4.2, suppose nodes 0 and 3, nodes 1 and 4, nodes 2 and 5 communicate with each other, respectively. Let node 1, 3 and 5 send their native packets P_1, P_3 and P_5, one by one at rate r_1. The transmission range under rate r_1 is marked as S_1, S_2 and S_3, respectively. As shown in Figure 4.2, only the one-hop neighbors of nodes 1, 3 and 5 can successfully overhear the packet transmitted from these nodes. After nodes 1, 3 and 5 finish transmitting, node 0 has P_1 and P_5, node 2 has P_1 and P_3, and node 4 has P_3 and P_5. Then nodes 0, 2 and 4 will act as *relay nodes* and start to transmit coded packets one by one at rate r_2, where $r_2 < r_1$.

Node	The overheard packets
0	P_1, P_5, $P_1 \oplus P_2 \oplus P_3$, $P_3 \oplus P_4 \oplus P_5$
1	$P_1 \oplus P_2 \oplus P_3$, $P_5 \oplus P_0 \oplus P_1$
2	P_1, P_3, $P_3 \oplus P_4 \oplus P_5$, $P_5 \oplus P_0 \oplus P_1$
3	$P_1 \oplus P_2 \oplus P_3$, $P_3 \oplus P_4 \oplus P_5$
4	P_3, P_5, $P_1 \oplus P_2 \oplus P_3$, $P_5 \oplus P_0 \oplus P_1$
5	$P_5 \oplus P_0 \oplus P_1$, $P_3 \oplus P_4 \oplus P_5$

Table 1: The overheard packets at each node

Node	The available packets
0	P_1, P_5, $P_1 \oplus P_2 \oplus P_3$, $P_3 \oplus P_4 \oplus P_5$, $P_5 \oplus P_0 \oplus P_1$
1	$P_1 \oplus P_2 \oplus P_3$, $P_5 \oplus P_0 \oplus P_1$, P_1
2	P_1, P_3, $P_3 \oplus P_4 \oplus P_5$, $P_5 \oplus P_0 \oplus P_1$, $P_1 \oplus P_2 \oplus P_3$
3	$P_1 \oplus P_2 \oplus P_3$, $P_3 \oplus P_4 \oplus P_5$, P_3
4	P_3, P_5, $P_1 \oplus P_2 \oplus P_3$, $P_5 \oplus P_0 \oplus P_1$, $P_3 \oplus P_4 \oplus P_5$
5	$P_5 \oplus P_0 \oplus P_1$, $P_3 \oplus P_4 \oplus P_5$, P_5

Table 2: The available packets at each node

In particular, the packet that node 0, 2 and 4 transmit are $P_5 \oplus P_0 \oplus P_1$, $P_1 \oplus P_2 \oplus P_3$ and $P_3 \oplus P_4 \oplus P_5$, respectively. The transmission region of node 0, 2 and 4 under r_2 is S_0, S_2, S_4, respectively. As a result, after each of these nodes $(0, 1, ..., 5)$ transmits once, all the overheard packets at each node are listed in Table 1.

Since all the packets that a node transmitted is also available to the node, we add the packets that each node transmits to the overheard packets and list all the available packets at each node in Table 2. Then each node simply XORs all the available packets shown in the second column of Table 2, the linear combination of these packets is shown in Table 3. From the results listed in Table 3, it can be easily found that each node meets the coding rule to decode P, which is equal to $P_0 \oplus P_1 \oplus P_2 \oplus P_3 \oplus P_4 \oplus P_5$ in this example. Because three of the total seven transmissions are at high speed in RANC while all the seven transmissions are at low speed in COPE, RANC has a better performance than COPE.

Node	The linear combination of the available packets
0	$P_0 \oplus P_1 \oplus P_2 \oplus P_4 \oplus P_5$
1	$P_0 \oplus P_1 \oplus P_2 \oplus P_3 \oplus P_5$
2	$P_0 \oplus P_1 \oplus P_2 \oplus P_3 \oplus P_4$
3	$P_1 \oplus P_2 \oplus P_3 \oplus P_4 \oplus P_5$
4	$P_0 \oplus P_2 \oplus P_3 \oplus P_4 \oplus P_5$
5	$P_0 \oplus P_1 \oplus P_3 \oplus P_4 \oplus P_5$

Table 3: XOR-based linear combination of the available packets at each node

4.2 The Design Principles of RANC

Overall, RANC tries to solve the problem that: given a network coding opportunity, how to maximize the objective performance gain of network coding? Specifically, the solution that RANC provides should answer following questions:

1. What's the transmission rate for each node?
2. What's the information of the packet that each node transmits?
3. What's the transmission schedule for all the nodes?

Also, RANC should meet the constraint that each node in the network should transmit once in each round of network coding[1]. We start from deriving the design principles that are the foundation for the RANC protocol design.

There are two types of nodes in RANC:

- *Native nodes*: They transmit native packets.
- *Relay nodes*: They transmit coded packets (XOR-ed information of native packets).

[1]A round of network coding is the time period that one packet of all flows is successfully delivered from the source node to the destination node.

4.2.1 Principle 1: Low speed transmission for relay nodes

For the relay node i that carries XOR-ed information of native packets transmitted from other nodes, if there always exists a node j ($j \neq i, D(i)$), and the channel condition between nodes i and j cannot support high speed transmission, then node i should transmit its coded packet C_i at low speed. This principle is the same as that of COPE since the coded packet C_i should be overheard by all other nodes except for node $D(i)$.

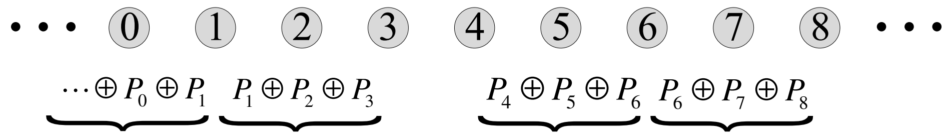

0
1
2
3
4
5
6
7
8
$\cdots \oplus P_0 \oplus P_1$
$P_1 \oplus P_2 \oplus P_3$
$P_4 \oplus P_5 \oplus P_6$
$P_6 \oplus P_7 \oplus P_8$

Figure 4.3: Illustration of overlapping with only one neighboring coded packet

4.2.2 Principle 2: The overlap of neighboring coded packets

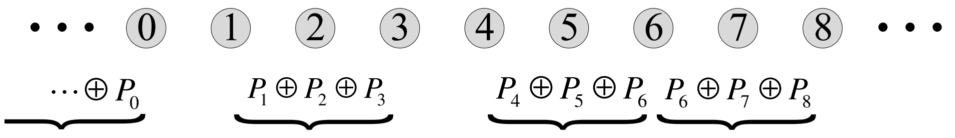
Suppose node i is a relay node and its coded packet is C_i and $l(C_i) > 1$. We can find two neighboring coded packets of C_i whose coding length is greater than 1.

Lemma 1 *Each coded packet ($l(C) > 1$) should have an overlap with both of its neighboring coded packets ($l(C) > 1$).*

proof Suppose the coding length is 3, without loss of generality, let's take the coded packet $P_1 \oplus P_2 \oplus P_3$ for example. We enumerate two cases for the coded packet as follows.
Case 1: As shown in Figure 4.3, the coded packet overlaps with only one of its neighboring coded packets. Without loss of generality, suppose node 2 is relay node, then according to the Principle 1, $D(3)$ will overhear the coded packet $P_1 \oplus P_2 \oplus P_3$. Since the information of ... $\oplus P_3$ is only transmitted once (by node 2), the coding rule is violated at node $D(3)$ since $L(\vec{X}_{D(3)}\vec{C}^T) \neq P \oplus P_3$.
Case 2: As shown in Figure 4.4, the coded packet does not have overlap with any of the two neighboring coded packets. Without loss of generality, suppose node 1 is relay node, then

according to Principle 1, nodes $D(2)$ and $D(3)$ will overhear the coded packet $P_1 \oplus P_2 \oplus P_3$ which is only transmitted once by node 1. This violate the coding rule at both node $D(2)$ and node $D(3)$ since $L(\vec{X}_{D(2)}\vec{C}^T) \neq P \oplus P_2$ and $L(\vec{X}_{D(3)}\vec{C}^T) \neq P \oplus P_3$, respectively.

The proof is similar when the coding length is other than 3. Therefore, by excluding Case 1 and Case 2, we conclude that each coded packet should overlap with both of its neighboring coded packets.



0
1
2
3
4
5
6
7
8
$\cdots \oplus P_0$
$P_1 \oplus P_2 \oplus P_3$
$P_4 \oplus P_5 \oplus P_6$
$P_6 \oplus P_7 \oplus P_8$

Figure 4.4: Illustration of overlapping with none of the neighboring coded packets

4.2.3 Principle 3: The coding length

Lemma 2 *For a coded packet C, its coding length must satisfy $l(C) \leq 3$*

proof Given any coding length $l(C)(l(C) \geq 4)$, if we can prove that any one of the following two conditions is satisfied, then it implies that $l(C) \leq 3$.

1. *Condition 1:* With same efficiency in terms of message complexity, the coded packet with the coding length $l(C)$ can be divided into at least two coded packets which have the coding length less than 4.
2. *Condition 2:* There do not exist a coding scheme for coding length $l(C) \geq 4$.

First, Let's look at the case when $l(C) = 4$. Without loss of generality, we study the coded packet $P_1 \oplus P_2 \oplus P_3 \oplus P_4$ which has the coding length of 4.

We enumerate three cases as follows.

Case 1: As shown in Figure 4.5, the overlap of two neighboring coded packets is 1.

If node 2 transmits the coded packet $P_1 \oplus P_2 \oplus P_3 \oplus P_4$, then the coded packets that node $D(2)$ receives include $... \oplus P_0 \oplus P_1, P_4 \oplus P_5 \oplus P_6 \oplus P_7, ...$, and then the linear combination of these packets does not include the packet P_3. Therefore, the coding rule is violated at node $D(2)$ unless node 3 transmits P_3, which requires an extra packet transmission. In this case, the message complexity of the coded packet $P_1 \oplus P_2 \oplus P_3 \oplus P_4$ is equivalent to that of the packets $P_1 \oplus P_2 \oplus P_4$ and P_3, whose coding length are 3 and 1, respectively. Therefore, *Condition 1* applies. Similar proof can be applied to the case when node 3 transmits the code $P_1 \oplus P_2 \oplus P_3 \oplus P_4$.

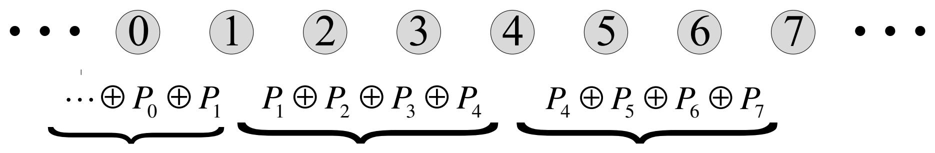

• • •
0
1
2
3
4
5
6
7
• • •
$\cdots \oplus P_0 \oplus P_1$
$P_1 \oplus P_2 \oplus P_3 \oplus P_4$
$P_4 \oplus P_5 \oplus P_6 \oplus P_7$

Figure 4.5: Illustration of the overlap of two neighboring coded packets is 1

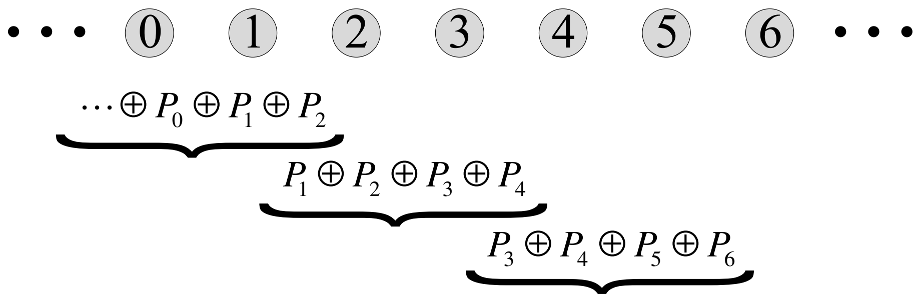

• • •
0
1
2
3
4
5
6
• • •
$\cdots \oplus P_0 \oplus P_1 \oplus P_2$
$P_1 \oplus P_2 \oplus P_3 \oplus P_4$
$P_3 \oplus P_4 \oplus P_5 \oplus P_6$

Figure 4.6: Illustration of the overlap of two neighboring coded packet is 2

If node 1 transmits the coded packet $P_1 \oplus P_2 \oplus P_3 \oplus P_4$, then node $D(1)$ cannot overhear $P_1 \oplus P_2 \oplus P_3 \oplus P_4$. However, node $D(1)$ can overhear $... \oplus P_0 \oplus P_1$. Therefore, the coding rule is violated at node $D(1)$ since $L(\vec{X}_{D(1)}\vec{C}^T) \neq P \oplus P_1$. The proof is similar to the case that node 4 transmits the coded packet $P_1 \oplus P_2 \oplus P_3 \oplus P_4$.

Case 2: As shown in Figure 4.6, the overlap of two neighboring coded packets with coding length of 4 is 2.

Let's take node $D(3)$ for example. By Principle 1, node $D(3)$ can receive at least one of the coded packet: $P_1 \oplus P_2 \oplus P_3 \oplus P_4$ and $P_3 \oplus P_4 \oplus P_5 \oplus P_6$. If node $D(3)$ receives only one coded packet, node $D(3)$ has the the information of packet P_3, which violates the coding rule since $L(\vec{X}_{D(3)}\vec{C}^T) \neq P \oplus P_3$. If node $D(3)$ receives both of the coded packets, then node $D(3)$ does not have the information of packet P_4, which is canceled due to XORing, unless node 4 transmits P_4 again. Therefore, the message complexity of the coded packet $P_1 \oplus P_2 \oplus P_3 \oplus P_4$ is the same as that of $P_1 \oplus P_2 \oplus P_3$ and P_4. Thus, *Condition 1* applies.

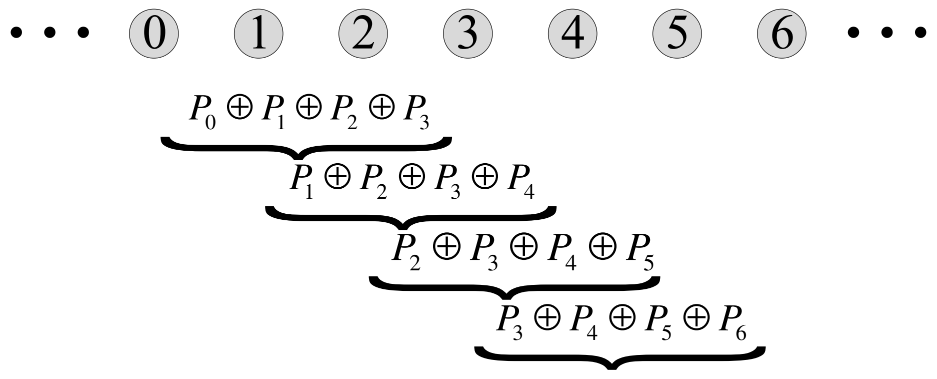

0
1
2
3
4
5
6
$P_0 \oplus P_1 \oplus P_2 \oplus P_3$
$P_1 \oplus P_2 \oplus P_3 \oplus P_4$
$P_2 \oplus P_3 \oplus P_4 \oplus P_5$
$P_3 \oplus P_4 \oplus P_5 \oplus P_6$

Figure 4.7: Illustration of the overlap of two neighboring coded packet is 3

Case 3: As shown in Figure 4.7, the overlap of two neighboring coded packets is 3.

In this case, the number of coded packets with coding length of 4 is the same as the total number of nodes in the network. Since each node only transmits once. For the first node that transmits in the network, the coding length is 1, which means it transmits its native packet. Therefore, it is impossible for it to transmit a code with length equal to 4. Thus, *Condition 2* applies.

Similarly, we can prove that either *Condition 1* or *Condition 2* applies to any coding length greater than 4. Therefore, the lemma holds.

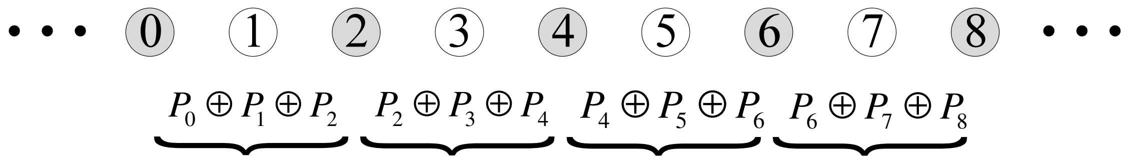
Based on the derived design principles, we can define the coding structure of RANC that is depicted in Figure 4.8. According to Principle 1 and Principle 2, we know that the coding length of RANC should be 3 and the native and coded packets should interleave with each other.

4.3 The Tradeoff in RANC's Performance Gain

In this section, we analyze the performance gain of RANC. For simplicity, we study the gain under the ideal wheel topology, in which the hub node resides in the center and all other nodes are evenly distributed on the circle. In the end of this section, we discuss how to evaluate the gain under irregular topologies.

4.3.1 Calculating the Minimum Transmission Range of Native Nodes under Ideal Wheel Topology

The transmission range of relay nodes are determined and they transmit at low speed. For example, suppose node 3 is relay node and its transmission range is shown in Figure 4.9. Now we need to find the minimum transmission range of native nodes in order to make RANC work.



• • •
0
1
2
3
4
5
6
7
8
• • •
$P_0 \oplus P_1 \oplus P_2$
$P_2 \oplus P_3 \oplus P_4$
$P_4 \oplus P_5 \oplus P_6$
$P_6 \oplus P_7 \oplus P_8$

Figure 4.8: The structure of coded packets in RANC

As shown in Figure 4.9, suppose node $1, 3, 5$ are relay nodes and node 0, 2, 4 are native nodes. Without loss of generality, we calculate the minimum transmission range of node 2.

The coded packets of relay node 1, 3 and 5 are $C_1 = P_0 \oplus P_1 \oplus P_2$, $C_3 = P_2 \oplus P_3 \oplus P_4$ and $C_5 = P_4 \oplus P_5 \oplus ....$ Based on Principle 1, node $D(4)$ can overhear C_1, C_3, C_5. Since information of P_2 is XOR-ed in C_1 and C_3, both of which are necessary for node $D(4)$ to decode P. Since XORing C_1 and C_3 will eliminate P_2 at node $D(4)$, node $D(4)$ has to overhear P_2. As node $D(3)$ can hear C_1 and C_5 and the information of P_2 is XORed only in C_1, which is sufficient for node $D(3)$ to decode P_2 from P. Thus, node $D(3)$

does not need to have more information of P_2. Therefore, packet transmitted from node 2 should be overheard by all the nodes except for node $D(2)$ and its one-hop neighbors on the circle (i.e., nodes $D(1)$ and $D(3)$ in this example). Since node 2 is a native node and node $D(4)$ is one of the furthermost nodes from node 2 (the other node is $D(0)$) that P_2 should be overheard, the minimum transmission range of node 2 should be $d(2, D(4))$. As the topology is an ideal wheel, similar argument can be applied to all other native nodes.

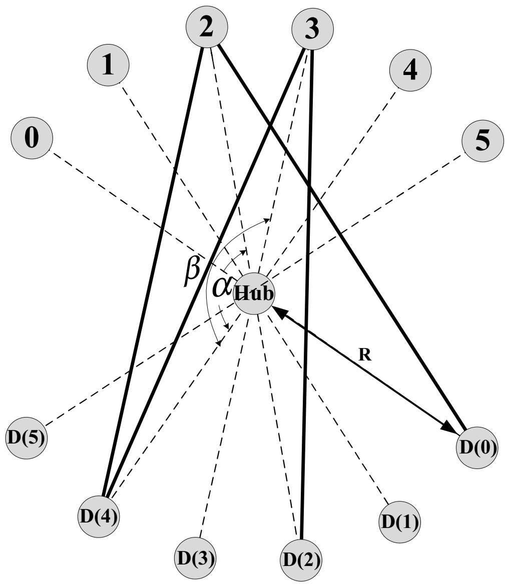


Figure 4.9: Calculate the transmission range of native nodes

4.3.2 The Tradeoff in RANC's Performance Gain

With the coding length of 3, we can analyze the throughput of network coding. We assume the two-ray ground propagation model is used so that the signal-to-noise ratio is determined by the distance between the sender and the receiver. By the Shannon's Theorem [**?**], in high *signal-to-noise ratio* (SNR) region, the transmission rate r follows:

$$r = B\log_2(1 + SNR) \approx B\log_2 SNR \tag{4.1}$$

where $SNR \propto \frac{1}{d^{\gamma}}$ and γ is the path loss exponent which is normally in the range of 2 to 4. Thus, we have

$$r \propto \log_2(d^{-\gamma}) \tag{4.2}$$

α, β and radius R are defined in Figure 4.9. According to the result from Section 4.3.1, the minimum transmission range of node 2(native node) and node 3(relay node) should be $d(2, D(4))$ and $d(3, D(4))$, respectively. Denote the minimum transmission range of native node and relay node as d_n and d_r, respectively. Let's calculate d_n and d_r. By the law of cosines, we have:

$$d_n = d(2, D(4)) = R\sqrt{2[1 - \cos\alpha]} = R\sqrt{2[1 + \cos\frac{4\pi}{n}]} \tag{4.3}$$

$$d_r = d(3, D(4)) = R\sqrt{2[1 - \cos\beta]} = R\sqrt{2[1 + \cos\frac{2\pi}{n}]} \tag{4.4}$$

Denote the transmission rate of native node and relay node as r_n and r_r respectively. From equation (4.2), we have

$$r_n \propto \log_2(d_n^{-\gamma}) \tag{4.5}$$

$$r_r \propto \log_2(d_r^{-\gamma}) \tag{4.6}$$

The throughput of network coding, denoted by T_{nc}, is defined by:

$$T_{nc} = \frac{\text{The number of packets serviced in each round}}{\text{The delay of each round}} \tag{4.7}$$

Let $T_{nc,RANC}$ and $T_{nc,COPE}$ represent the throughput of RANC and COPE respectively. In RANC, half of participating node n are native nodes and half are relay nodes, therefore the throughput of RANC under ideal wheel topology can be calculated as follows.

$$T_{nc,RANC}(n) = \frac{n}{\frac{n}{2}\frac{1}{r_n} + \frac{n}{2}\frac{1}{r_r} + \frac{1}{r(broadcast)}}$$

$$\propto \frac{n}{\frac{n}{2}\frac{1}{\log_2\frac{1}{d_n^\gamma}}+\frac{n}{2}\frac{1}{\log_2\frac{1}{d_r^\gamma}}+\frac{1}{\log_2\frac{1}{R^\gamma}}}$$

$$\propto \frac{n}{\frac{n}{2}\frac{1}{\log_2\frac{1}{(R\sqrt{2[1+\cos\frac{4\pi}{n}]})^\gamma}}+\frac{n}{2}\frac{1}{\log_2\frac{1}{(R\sqrt{2[1+\cos\frac{2\pi}{n}]})^\gamma}}+\frac{1}{\log_2\frac{1}{R^\gamma}}}$$

For COPE, all participating nodes n transmit at the same rate of the relay nodes in RANC. Therefore, the throughput of COPE can be calculated as follows.

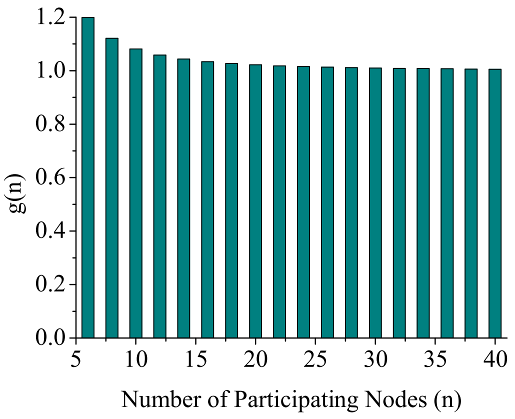
$$T_{nc,COPE}(n)=\frac{n}{n\frac{1}{r_r}+\frac{1}{r(broadcast)}}\propto\frac{n}{n\frac{1}{\log_2\frac{1}{d_r^\gamma}}+\frac{1}{\log_2\frac{1}{R^\gamma}}}$$

From the analysis, we have the following two findings:

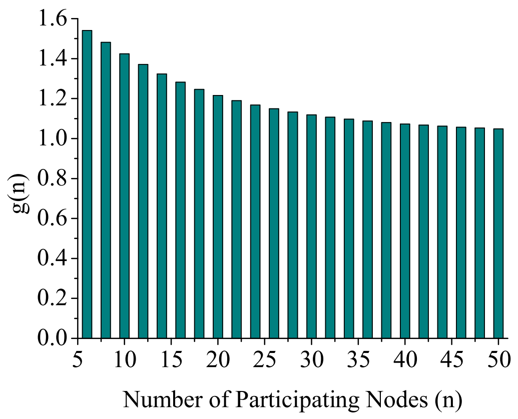
First, the throughput of RANC is always better than that of COPE due to high speed transmissions. We define $g(n)=\frac{T_{nc,RANC}(n)}{Th_{nc,COPE}(n)}$ as the throughput gain of RANC over COPE when the number of participating nodes in the network is n. As n increases, the throughput gain $g(n)$ becomes smaller. When $n\rightarrow\infty$, $g(n)\rightarrow 1$. Based on the structure of RANC, in order to benefit from RANC, n should be at least 6. When n is 2 or 4, there is no need to have relay-aided network coding. Therefore, we compare throughput of RANC with COPE in cases when $n\geq 6$. When SNR$=20$, for the path loss exponent $\gamma=2$ and $\gamma=4$, the numerical results of $g(n)$ are shown in Figure 4.10(a) and (b), respectively. As can be seen, the $g(n)$ increases as γ increases.

Second, the throughput of RANC is maximized when there are three pairs of symmetric flows in the network, which, in other words, means 6 participant nodes consisting the network coding opportunity. This is a very important result for the protocol design of RANC (see the next Section). It implies that when the number of participating nodes is larger than 6, we can partition them into groups, each of which has at most 6 members. For instance, if the number of participating nodes is n, we can partition them into $\lceil\frac{n}{6}\rceil$ groups so that $g(n)\approx\frac{T_{nc,RANC}(6)}{T_{nc,COPE}(n)}$. One thing deserves attention is that any two nodes forms a symmetric flow are always partitioned in the same group. In other words, the basic element in partition set is a symmetric flow.

Remark: For an irregular wheel topology, RANC can be applied to a set of nodes as long as the packet transmitted from node i at low speed can be overheard by any other nodes except for the destination of node i. Given the SNR of each link associated with the set of nodes, the performance gain of RANC can be computed in a similar way.



(a) $\gamma = 2$



(b) $\gamma = 4$

Figure 4.10: Numerical results of the throughput gain of RANC

4.4 The RANC Protocol Design

Based on the analysis of design principles and the performance gain of RANC, we design the protocol and algorithm of RANC via decomposing the original RANC problem into two sub-problems: the flow partition problem and the scheduling problem. On the one hand, $g(n)$ is a non-increasing function of n. If we want to obtain the maximum throughput gain of RANC, we should partition the flow set into subsets each of which has no more than 3 pairs of symmetric flows(6 participating nodes). On the other hand, if we want to the obtain maximum network Coding+Mac gain (especially for the network using IEEE 802.11

MAC), then we should include as many flows as possible to perform network coding in each round. Therefore, there exists a tradeoff between the throughput gain and the network Coding+Mac gain in RANC. The first sub-problem is to partition the flow set according to the cost function that can be used to balance the tradeoff between the throughput and Coding+Mac gain. The output will be a flow partition decision that minimizes the total cost. As the partition problem is NP-hard, we provide a $ln\frac{n}{2}$-approximation algorithm to achieve an approximate solution. After the first sub-problem is solved, we need to solve the second sub-problem, which is to decide the transmission schedule of nodes in each subset.

4.4.1 The Flow Partition Problem

The flow partition problem falls into the category of the weighted set cover problem. Suppose U is the universe of $\frac{n}{2}$ flows, $U = 1 ... \frac{n}{2}$, S is the total subsets of U, $|S| = 2^{\frac{n}{2}} - 1$. s_i is a partition set and $|s_i|$ is the cardinality of s_i. The cost function for each subset s_i is $c(s_i)$.

The objective of the flow partition problem is to find a set $I \subseteq S$ that minimizes $\sum\limits_{i \in I} c(s_i)$, subject to $\bigcup\limits_{i \in I} s_i = U, s_i \cap s_j = \phi$. The members of I are mutually disjoint. We use a binary variable x_{s_i} to indicate whether set s_i is selected. Formally, the problem is:

$$\min \sum_{i \in S} x_{s_i} c(s_i) \tag{4.8}$$

s.t.

$$\sum\nolimits_{j \in S | x_{s_j}=1, i \in s_j} x_{s_j} = 1, \quad i \in U \tag{4.9}$$

$$x_{s_i} = 0 \; or \; 1, \; i \in S \tag{4.10}$$

Equation (4.9) means that each flow i belongs to one and only one set. We define the utility function is as follows:

$$c(s_i) = -(\alpha \cdot T_{nc,RANC}(s_i) + (1-\alpha) \cdot G_{nc+mac}(s_i)) \tag{4.11}$$

where $G_{nc+mac}(s_i)$ is the Coding+Mac gain for the flow set s_i and $\alpha \in [0,1]$ is a tuning parameter used to balance the tradeoff the throughput gain and the Coding+Mac gain. As the weighted set partition problem is NP-hard [and79], it will be costly to compute the exact optimal solution. Therefore, after having the cost for each subset, we use the following approximation algorithm to find the solution for flow partition.

```
Algorithm
Step 1: $I \leftarrow \phi, \bar{U} \leftarrow U$
Step 2: While $\bar{U} \neq \phi$
    (a) $s \leftarrow \underset{X \in S}{\operatorname{argmax}}\, c(X)$
    (b) $I \leftarrow I \cup s$, $S \leftarrow S \backslash \{s, p \,|p \cap s \neq \phi\}$, and $\bar{U} \leftarrow \bar{U} \backslash s$, repeat step 2.
```

Lemma 3 *The algorithm is a lnN-approximation algorithm for the flow partition problem.*

proof We first order the flow pairs in the order that they were picked, breaking ties arbitrarily. At the time that the i-th flow pair was picked, $\bar{U}$ contains at least $\frac{n}{2} - i + 1$ elements (flow pairs). Let OPT denote the optimal objective function value of flow partition problem(equation (4.8). At that point, the "per-element" cost of OPT is at most $OPT/(\frac{n}{2} - i + 1)$. Thus, for at least one of the sets in OPT, we have

$$c(s) \leq \frac{OPT}{\frac{n}{2} - i + 1} \tag{4.12}$$

Since the algorithm is greedy, the set s picked by the algorithm satisfies the above inequality. Thus, sum over all the elements in $\bar{U}$, we have

$$\sum_{i \in \bar{U}} c(s_i) \leq \sum_{i=1}^{\frac{n}{2}} \frac{OPT}{\frac{n}{2} - i + 1} \leq OPT \ln \frac{n}{2} \tag{4.13}$$

4.4.2 The Scheduling Problem

Let $d_r(i,j)$ represent the maximum transmission radius of node i when it transmits a packet to node j at the rate of r subject to the bit error rate should be kept below a pre-defined threshold. According to the analysis in Section 4.3.1, we denote $d_{j,min}$ as the minimum transmission radius of native node j.

This scheduling problem is performed for each subset we get from the solution of the flow partition problem. In particular, the scheduling decision is made on per subset basis until all the flows are serviced. We denote the current subset, the subset of relay nodes and the subset of native nodes as S_c, S_r, and S_n, respectively. Assuming that node IDs in S_c are $0, ..., |S_c| - 1$, the scheduling algorithm is formalized as follows.

$$\max \sum_{j \in S_c} r_j \tag{4.14}$$

s.t.

$$C_i = P_i, \quad i \in S_n \tag{4.15}$$

$$C_i = P_{i-1+|S_c| \bmod |S_c|} \oplus P_{i \bmod |S_c|} \oplus P_{i+1 \bmod |S_c|},\ i \in\ S_r \tag{4.16}$$

$$t_i < t_j, \quad i \in S_n, j \in S_r \tag{4.17}$$

$$\begin{gathered} r_j = argmax_r d_r(j,k) \geq \max(d_{j,min}, d(j,k)), j \in S_n \\ k = j + 1\ mod\ |S_c|\ or\ j - 1 + |S_c|\ mod\ |S_c| \end{gathered} \tag{4.18}$$

$$r_i = argmax_r d_r(i,j) \geq \max(d(i,j)), i \in\ S_r, j \in S_c, j \neq i, D(i) \tag{4.19}$$

Equation (4.15) means that the native nodes send their own packet first. Equation (4.16) gives out the packet that the relay nodes transmit. Constraint (4.17) says the native nodes transmit before the relay nodes. Equation (4.18) and (4.19) define the transmission rate of native nodes and relay nodes, respectively. Given a subset of nodes, since the role of each node is either native node or relay node, the scheduling decision is to divide the nodes into two groups so that the aggregate transmission rate that can be achievable under the constraints (4.15-4.19). Thus, the scheduling decision can be found with polynomial complexity. Regarding the efficient implementation of scheduling, it should be carefully designed if the transmission collision and interference from neighboring nodes of the participant nodes are taken into account. In this paper, we assume a collision free MAC mechanism is adopted for each round of network coding. One example of the MAC mechanism is to reserve a dedicated network coding channel and the participant nodes can switch to the channel and access the channel in a time-division multiple access (TDMA) manner. How to implement the scheduling for RANC in a network under single channel with a random access MAC protocol (e.g. IEEE 802.11 based WLAN [**?**]) will be part of our future work.

Since the hub-node can hear the transmission from any other nodes, it has the best position to perform flow partitioning and packet scheduling. Therefore, we let the hub-node execute the RANC protocol, which is summarized in Table 4. RANC is performed round by round. Similar to COPE [KRH$^+$06], the information regarding channel condition and the status of network coding in the previous round can be piggybacked in each data packet.

4.5 Performance Evaulation

We evaluate the performance of RANC via extensive simulations. Our simulation is based on ns-2 [Pro]. The distance threshold for $11Mbps$, $5.5Mbps$, and $2Mbps$ are $100m$, $200m$, and $250m$ respectively. The data packet length is set to be 1000 bytes and the simulation

```
The process of RANC for each round of network coding
Input: the union of flow pairs $U$;
        the channel condition of all the links in the network;
Calculate the cost function for each subset of $U$;
Run approximation algorithm to compute flow partition $I$;
**for** each subset $s \in I$ **do**
        Compute the transmission schedule of $s$;
**for** each $node \in s$ **do**
        Send out corresponding packet according to the schedule;
**end for**
**end for**
**if** any node failed in decoding $P$ in the previous round
        Request the hub-node to re-transmit the native packet destined to the node;
**end if**
```

Table 4: The pseudo-code of the RANC protocol

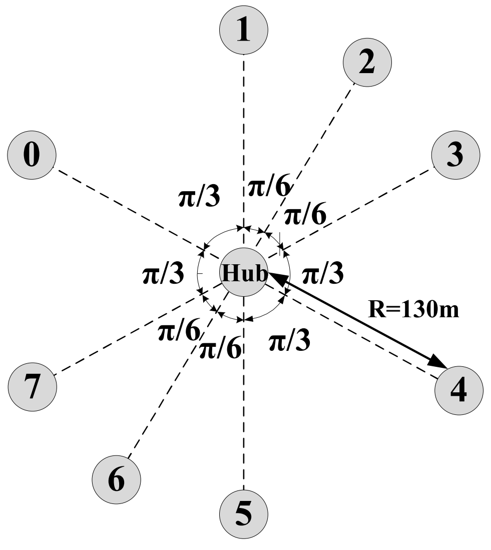
time is set to be 100 seconds. We compare RANC with COPE in terms of the throughput of network coding and the Coding+MAC gain[2]of the hub-node. The control overhead is counted in the performance measurement.

We first evaluate the scenario in which there are 3 flow pairs in an ideal wheel topology, which is same as that in Figure 4.2, with the radius of $130m$. In this case, the rate of each transmission in COPE is $2Mbps$. For RANC, the transmission rates of native nodes and relay nodes are $5.5Mbps$ and $2Mbps$, respectively, in this case. The simulation results are shown in Figure 4.12. As can be seen, due to the existence of high speed transmission, RANC improves the throughput by approximately 50%. As we explained previously, there is no need to do flow partition when $\frac{n}{2} \leq 3$. Therefore, in this case, both RANC and COPE require the hub-node to transmit once in each round. While there is no network coding, hub-node has to transmit 6 times each round under current topology, therefore, the draining rates of hub-node under both RANC and COPE are 6 times faster than that without using

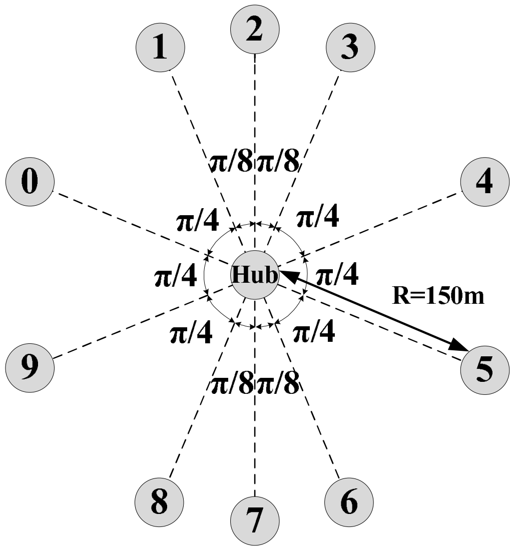
[2]To take channel condition into account, we define the draining rate as the transmission time spent by the hub-node to forward all the packets.

network coding. In other words, the Coding+MAC gain is at most 6. It can be observed that the Coding+MAC gain of RANC and COPE are close to each other and both are below 6.

Then, we evaluate the performance of RANC in more complicated scenarios(Figure 4.11(a)). The participating nodes are all located on the circle and the radius is $130m$. The cost function follows equation (4.11). As the tuning parameter α is used to balance the tradeoff between the throughput gain and the Coding+MAC gain, different value of α leads to different flow partition I. Basically, a large α favors the Coding+MAC gain and a small α favors the throughput gain. For example, given the topology shown in Figure 4.11(a), we have flow partition $I_1 = \{s_1 = \{(0,4),(1,5),(3,7)\}, s_2 = \{(2,6)\}\}$ for $\alpha = 0.9$, and flow partition $I_2 = U = \{(0,4),(1,5),(2,6),(3,7)\}$ for $\alpha = 0.8$. We evaluate the throughput and Coding+MAC gain of RANC with both I_1 and I_2 as well as those of COPE. First, let us analyze the performance of RANC-I_1 and COPE. The rate of each transmission for COPE is still $2Mbps$. For RANC-I_1, there are two subsets in it and each subset has high speed transmission($r > 2Mbps$). The simulation results are shown in Figure 4.13. As can be seen, due to the existence of high speed transmission achieved by flow partition, RANC greatly improves the throughput. However, this is at the expense of less Coding+MAC gain of hub-node. Hub-node has to participate transmission for each subset. In the case of I_1, hub-node has to broadcast in both s_1 and s_2. Therefore, RANC-I_1 requires the hub-node to transmit twice in each round. For COPE, hub-node always transmit once in each round. When $n = 8$, hub-node has to transmit 8 times in each round if there is no network coding. Therefore, the Coding+MAC gain of RANC-I_1 and COPE are 4 and 8, respectively and they are the upper bound. Similarly, we compare the results of RANC-I_2 and COPE in Figure 4.14. Since the flow partition $I_2 = U$, hub-node transmit once in each round, which is same as COPE. It can be observed that the Coding+MAC gain of the two are close to each other. In the terms of throughput, since there still exist high speed



(a) n=8



(b) n=10

Figure 4.11: The network topologies with different n

transmission($r > 2Mbps$), the throughput of RANC-I_2 is greater than COPE. Compare Figure 4.13 and 4.14, we can observe the throughput of RANC-I_1 is higher than that of RANC-I_2 at the expense of less Coding+MAC gain.

We also study the scenario whose topology is shown in Figure 4.11(b). The participating nodes are all located on the circle with radius equal to $150m$. In this case, we have flow partition $I_1 = \{s_1 = \{(0,5),(2,7),(4,9)\}, s_2 = \{(1,6)\}, s_3 = \{(3,8)\}\}$ for $\alpha = 0.95$, $I_2 = \{s_1 = \{(0,5),(1,6),(3,8),(4,9)\}, s_2 = \{(2,7)\}\}$ for $\alpha = 0.9$, $I_3 = U = \{(0,5),(1,6),(2,7),(3,8),(4,9)\}$ for $\alpha = 0.8$. The throughput and Coding+MAC gain of RANC under I_1, I_2, I_3 and COPE are shown in Figure 4.15, 4.16 and 4.17, respectively.

As shown in Figure 4.15, let us first analyze the simulation results of RANC-I_1 and COPE. The rate of each transmission for COPE is $2Mbps$ under this topology. For RANC-I_1, there are three subsets in it and each subset has high speed transmission($r > 2Mbps$) compared to COPE. As can be seen, due to the existence of high speed transmission achieved by flow partition, RANC greatly improves the throughput. However, hub-node has to transmit once in each subset which add up to a total of three times in each round.

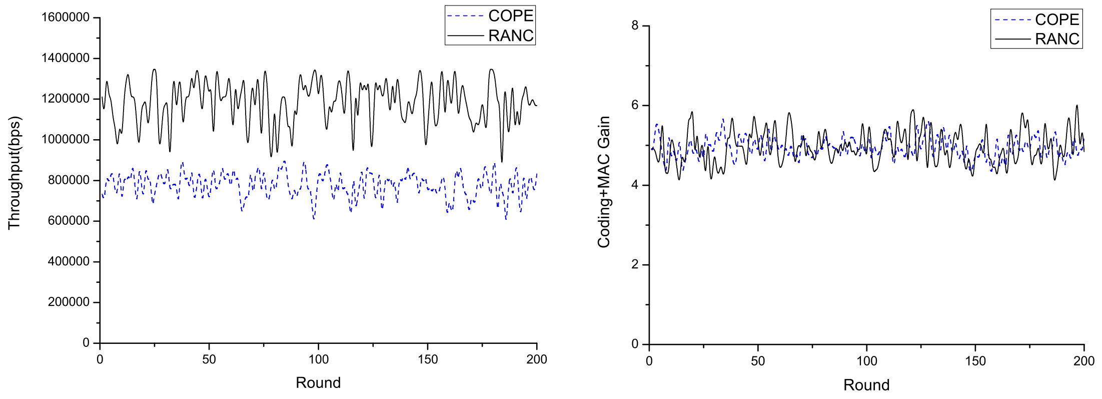


Figure 4.12: n=6

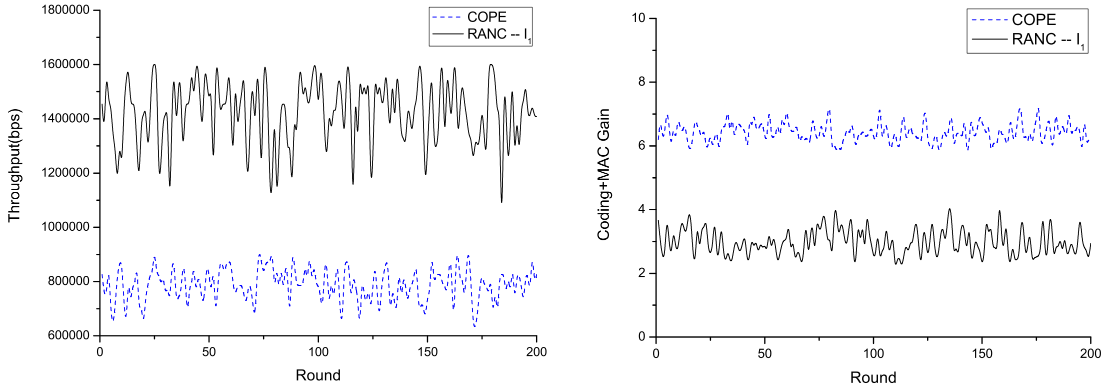


Figure 4.13: n=8, flow partition is I_1

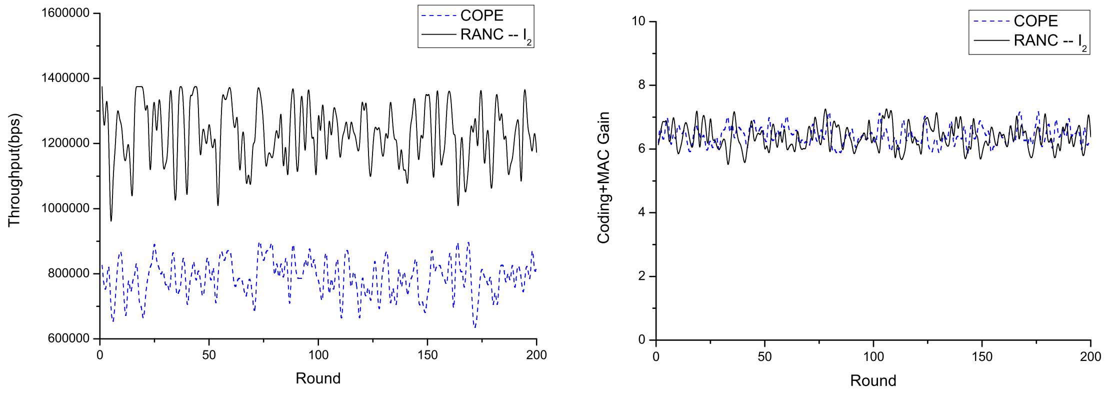


Figure 4.14: n=8, flow partition is I_2

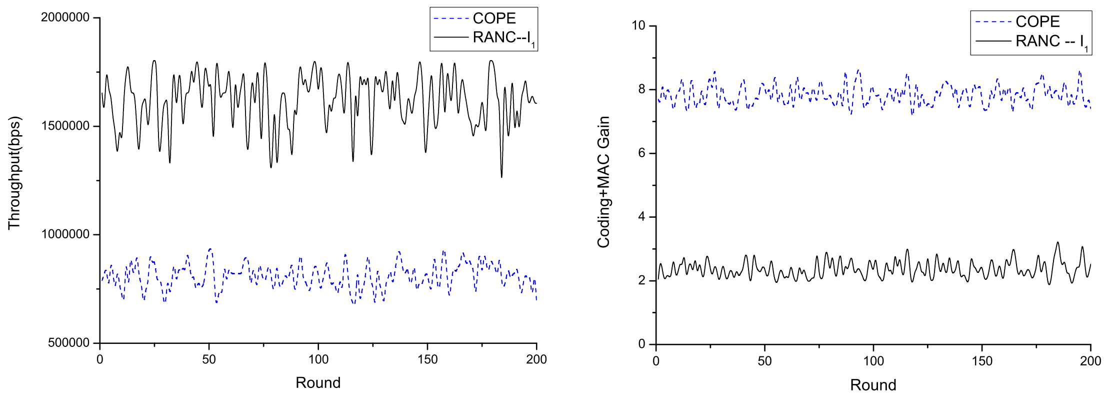


Figure 4.15: n=10, flow partition is I_1

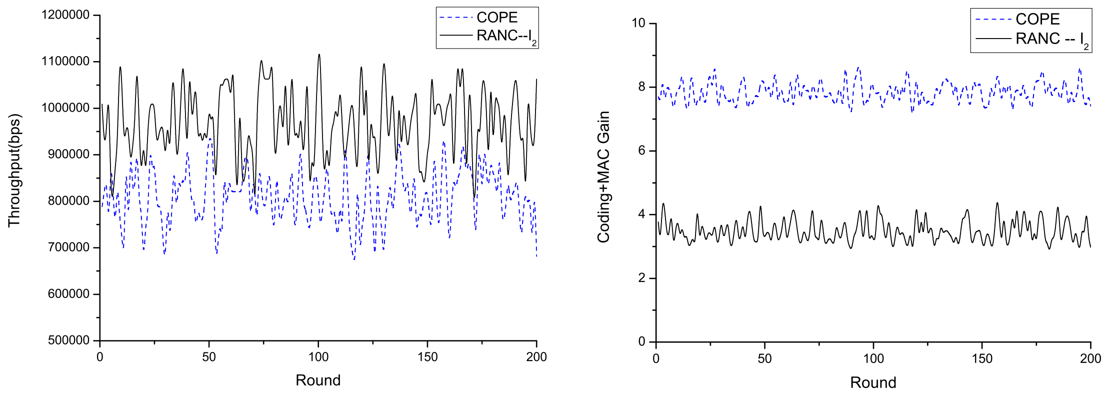


Figure 4.16: n=10, flow partition is I_2

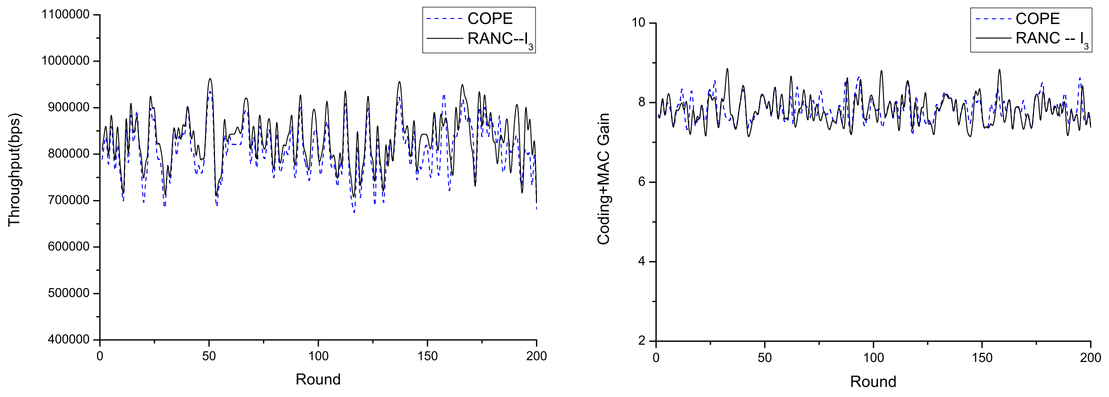


Figure 4.17: n=10, flow partition is I_3

While in COPE, hub-node always transmit once in each round. When $n = 10$, hub-node has to transmit 10 times in each round if there is no network coding. Therefore, the Coding+MAC gain of RANC-I_1 and COPE are 3.3 and 10, respectively and they are the upper bound. For RANC-I_2, there are two subsets and it has less high speed transmissions($r > 2Mbps$) compared to RANC-I_1. Therefore, from simulation results shown in Figure 4.16, we can see that the throughput of RANC-I_2 is lower than that of RANC-I_1. The Coding+MAC gain of RANC-I_2, however, is higher than that of RANC-I_1. Because there are two subsets in RANC-I_2, hub-node has to transmit twice each round. Thus, the Coding+MAC gain is 5 in this case. Similarly, we compare the results of RANC-I_3 and COPE in Figure 4.17. Since the flow partition $I_3 = U$, hub-node transmit once in both RANC and COPE. It can be observed that the Coding+MAC gain of the two are close to each other and the upper limit is 10. In the terms of throughput, since all the transmissions of RANC are at $2Mbps$ for this flow partition, the throughput of RANC-I_3 is close to that of COPE. Compare Figure 4.15, 4.16 and 4.17, we can easily observe the tradeoff between throughput and Coding+MAC gain of RANC. As can be seen, the throughput of RANC-I$_3$($I_3 = U$) has the same throughput as COPE. This is different from the case of RANC-I_2($I_2 = U$) when $n = 8$ (see Figure 4.14, 4.17). The reason is that the requirement of the minimum transmission range of native nodes (see Section 4.3.1) forces the native nodes to transmit at low speed($2Mbps$). It indicates that when the node density is high, it is more beneficial for RANC to set α close to 1.0 in order to achieve higher throughput. However, partition with higher throughput has less Coding+MAC gain.

CHAPTER 5

JOINT ROUTING AND SCHEDULING OPTIMIZATION SCHEME

In this chapter, we introduce a joint design of distributed routing at the network layer and distributed coding-aware scheduling at the MAC layer. The design problem is modeled as a network utility optimization problem. Different from previous works in cross-layer optimization in computer networks [Haj88] and [Tas92], we focus on how to exploit the network coding gain to improve the overall end-to-end transmission performance in network coding-aware wireless mesh network. We first formalize the design problem and decompose it into two sub-problems: the routing problem and the scheduling problem. The cross-layer design is achieved with a simple but crucial linkage between these subproblems. The routing mechanism decides how to distributed the traffic demands over the end-to-end paths in the network. The scheduling mechanism is used to maximize the overall weighted link capacity with network coding awareness. With the joint design, the network coding gain is realized by the scheduling module. The link level gain is further signaled to the routing module which utilizes the information to carefully distribute the traffic to maximize the overall end-to-end networking performance. As a result, more traffic is routed on the path to create more network coding opportunities until the network reaches the equilibrium point where no better routing decisions can be found. In addition to the theoretical framework, we propose mechanisms to implement the framework in an efficient and distributed way by considering the factors of time synchronization, protocol scalability, and the MAC layer coordination. We evaluate our design through extensive simulations and the simulation results show that the joint design of routing and scheduling can greatly increase the capacity of the network by exploiting the network coding-awareness.

The chapter is organized as follows. In Chapter 5.1, first, we present the system model and notations, then we formulate the network optimization problem and exploit the structure of the problem via decomposition. In Chapter 5.2, two different asynchronous routing algorithms are proposed to solve the routing problem. Chapter 5.3 gives the solution for the network coding-aware scheduling problem and presents a low-complexity distributed coding-aware scheduling algorithm. In Chapter 5.4, we evaluate the performance of joint routing and coding-aware scheduling using simulations.

5.1 System Model and Problem Formulation

We represent the network with a directed graph $H = (N, E)$, where N is the set of nodes and E is the set of links. Every link $(i, j) \in E$ interferes with a set of other links with the following interference model: Let I_i denote the one-hop neighboring nodes of node i, link (k, l) interferes with link (i, j) if $k \in I_i \cup I_j$ or $l \in I_i \cup I_j$. Therefore, to avoid collision, when link (i, j) is active, all nodes in $I_i \cup I_j$ should be idle. In the network, there are multiple uni-cast connections from the source nodes to the destination nodes. The set of source nodes is denoted as S and the set of destination nodes is denoted as T. For each source-destination flow pair, we denote it as f. Let $S(f)$, $T(f)$ and d^f be the source node, destination node and traffic demand of flow f, respectively. Let P^f represent the path set of flow f and $|P^f|$ is the cardinality of P^f. Denote $x^{f,p}$ the traffic of flow f which is routed through path p. The vector x^f consists of the elements $x^{f,p}, p \in P^f$. x^f_{ij} then represent the traffic of flow f routed on link (i, j). Clearly, $x^f_{ij} = \sum\limits_{p \in P^f (i,j) \in E} x^{f,p}$. Let F denote the set of all the source destination flow pairs f in the network.

At the MAC layer, we assume the following time division MAC protocol is used: in the network, time is divided into slots of equal length. Each time slot is divided into two parts: a channel contention slot and a data transmission slot. The links that are to be scheduled are chosen in the contention slot and the chosen links transmit their packets in the data transmission slot. More detailed protocol description will be introduced in Chapter 5.3. c_{ij}, $(i, j) \in E$ is used to represent the capacity of link (i, j). Due to restriction from the interference model, vector $\mathbf{c}$ is determined by the scheduling algorithm and lies within a capacity region Π, which is actually the convex hull of $\mathbf{c}$.

From the Little's Theorem [Asm03], given the average number of packets in the stable network, a smaller end-to-end delay implies a higher average traffic arrival rate at the source node. Therefore, our design goal is to minimize the average end-to-end delay of the network. We associate with the network a delay function which reflects the average latency that maps valid routing matrix to real numbers that we seek to minimize. With M/M/1 queuing model, link (i, j)'s delay function can be $\frac{1}{c_{ij}-x_{ij}}$ where x_{ij} is the aggregate traffic traversing link (i, j). It follows that $\frac{x_{ij}}{c_{ij}-x_{ij}}$ is a non-negative, differentiable and non-decreasing convex function with respect to x_{ij}. However, in multi-hop network settings, it is difficult to find a closed-form and tractable delay function. By using the measured link delay, we do not need explicit definition of link delay and only assume the load-latency product function is convex. Thus, for the analysis purpose, we suppose a convex function $U(x_{ij})$ which reflects the average link delay experienced by all flows. Consequently, the network optimization problem can be formalized as following:

$$\min \sum_{(i,j)\in E} U(\sum_{f\in F} x_{ij}^f) \tag{5.1}$$

s.t.

$$\sum_{f\in F} x_{ij}^f \leq c_{ij}, \forall (i,j) \in E \tag{5.2}$$

$$\mathbf{c} \in \mathbf{\Pi} \tag{5.3}$$

$$(\sum_{\{j|(i,j)\in E\}} x_{ij}^f - \sum_{\{j|(j,i)\in E\}} x_{ji}^f) = \sigma_i^f, \forall i \in N, f \in F \tag{5.4}$$

$$\sigma_i^f = \begin{cases} d^f, & \text{if } i \in S \\ -d^f, & \text{if } i \in T \\ 0 & \text{otherwise.} \end{cases} \tag{5.5}$$

The objective is to minimize the delay function over the network. Equations (5.2) and (5.3) are capacity and capacity region constraints. Equations (5.4) and (5.5) are flow conservation constraints. We form the dual problem by introducing Lagrange multipliers λ_{ij} only for the coupling constraints $\sum_{f\in F} x_{ij}^f \leq c_{ij}$. This results in the Lagrangian dual function:

$$\begin{aligned}L(x,c,\lambda) &= \sum_{f\in F,(i,j)\in E} U(x_{ij}^f) + \sum_{(i,j)\in E} \lambda_{ij}(\sum_{f\in F} x_{ij}^f - c_{ij}) \\ &= (\sum_{f\in F,(i,j)\in E} U(x_{ij}^f) + \sum_{(i,j)\in E} \lambda_{ij} \sum_{f\in F} x_{ij}^f) \\ &- \sum_{(i,j)\in E} \lambda_{ij} c_{ij}\end{aligned} \tag{5.6}$$

We observe that the dual function can be separated into two sub-problems: the routing problem and the scheduling problem. In the dual objective function (5.6), the first part $\sum_{f\in F,(i,j)\in E} U(x_{ij}^f) - \sum_{(i,j)\in E} \lambda_{ij} \sum_{f\in F} x_{ij}^f$ is the objective function of a routing problem. The second part $\sum_{(i,j)\in E} \lambda_{ij} c_{ij}$ is the objective function of a scheduling problem. For constraints (5.4) and (5.5) are the constraints of routing problem and constraint (5.3) is associated the scheduling problem. Therefore, we can rewrite the dual problem in the following way:

$$V(\lambda) = V_{rt}(\lambda) + V_{sch}(\lambda) \tag{5.7}$$

where $V_{rt}(\lambda)$ is defined as

$$V_{rt}(\lambda) = \min\{ \sum_{f\in F,(i,j)\in E} U(x_{ij}^f) - \sum_{(i,j)\in E} \lambda_{ij} \sum_{f\in F} x_{ij}^f \left| \begin{array}{c} (\sum_{\{j|(i,j)\in E\}} x_{ij}^f - \sum_{\{j|(j,i)\in E\}} x_{ji}^f) = \sigma_i^f, \forall i \in N, f \in F \\ \sigma_i^f = \begin{cases} d^f, i = s \\ -d^f, i = t \\ 0, otherwise \end{cases} \end{array} \right\} \tag{5.8}$$

and $V_{sch}(\lambda)$ is defined as:

$$V_{sch}(\lambda) = \max\{ \sum_{(i,j)\in E} \lambda_{ij} c_{ij} | c_{ij} \in \Pi \}. \tag{5.9}$$

Given $V_{rt}(\lambda)$ and $V_{sch}(\lambda)$ that are parameterized by the dual variable λ, the dual problem can be solved using the sub-gradient method by updating the dual variable λ as following:

$$\lambda_{ij}(t+1) = \left[\lambda_{ij}(t) + \frac{\alpha}{c_{ij}} (\sum_{f\in F} x_{ij}^f(n) - c_{ij}) \right]^+ \tag{5.10}$$

Since the term $\sum\limits_{f\in F} x_{ij}^{f}(n)$ is the aggregate traffic routed over link (i,j) and c_{ij} is the capacity of link (i,j), the buffer occupancy $b_{ij}(t)$ evolves according to:

$$b_{ij}(t+1) = \left[b_{ij}(t) + \sum_{f\in F} x_{ij}^{f}(n) d^{f} - c_{ij}(t) \right]^{+} \tag{5.11}$$

Dividing both sides by $c_{ij}(t)$ and multiplying both sides by α and identifying $\lambda_{ij} = \alpha \frac{b_{ij}}{c_{ij}}$, we see that λ_{ij} can be understood as the weighted queuing delay of link (i,j) and is an approximate of the subgradient of the dual problem. The small stepsize α is used to ensure convergence. Thus, each link periodically measures the average queuing delay and update its λ. As a result, a more congested link has a larger value of λ.

5.2 Solving the Routing Problem

In this section, we focus on designing distributed and asynchronous routing algorithms to find the optimal solution of the routing problem stated in (5.8). We first introduce the algorithm based on the one proposed in [J.N86]. The algorithm is a gradient based method with the proven property of convergence. However, as we will see, the algorithm is difficult to implement with good robustness. To overcome the difficulty, we introduce another routing algorithm which is not only much easier to implement but also has proven convergence.

5.2.1 The Gradient-based Routing Algorithm

From the analysis in previous section, we observe that the objective function of routing problem can be rewritten in following form, $\sum\limits_{(i,j)\in E} U(x_{ij}) - \sum\limits_{(i,j)\in E} x_{ij} d(x_{ij})$, where $d(.)$ is the delay function and x_{ij} is the amount of traffic over link (i,j). Denote the routing problem as $V_{rt}(x)$. By the definition of x_{ij}, we are actually studying the allocation of $x^{f,p}(p \in P(f))$ for each flow f, which is the multi-path traffic routing problem. Assume each source node of flow $f \in F$ makes the routing decisions and each source node can update the value $x^{f,p}$ by performing the gradient projection update as follows:

$$x^{f,p}(n+1) = [x^{f,p}(n) + \gamma\mu^f \frac{\partial V_{rt}(x^{f,p}(n))}{\partial x^{f,p}}]^+_{G^f}. \tag{5.12}$$

Here, $[a]^+_{Gf}$ is the projection of a on the closed set $W \subset R^n$, with respect to the inner product $\langle \cdot, \cdot \rangle_{G^f}$, where G^f is a symmetric positive matrix. W is the projection plane. The computation of the projection $[\cdot]^+$ on W will be addressed shortly. The above projection update requires perfect synchronization of the all source nodes, which is difficult to implement due to the asynchronous feedback of queuing delay information. Therefore, it is necessary to find an asynchronous gradient projection mechanism for the routing adaptation. In particular, each source node runs the adaptation algorithm once during a certain time interval, which is called updating period. Let $\bar{x}$ represent the vector of desired flows percentage. Each source node makes the routing decision according to the following equation.

$$x^{f,p}(n) = a^{f,p}(n)\bar{x}^{f,p}(n) + (1 - a^{f,p}(n))x^{f,p}(n-1). \tag{5.13}$$

We assume

$$a^{f,p}(n) \geq \xi, \; \forall f \in F, \forall p \in P(f), \forall n \tag{5.14}$$

where $\xi > 0$ and $a^{f,p}$ is the solution of the following equation

$$\sum_p a^{f,p}(n)(\bar{x}^{f,p}(n) - x^{f,p}(n-1)) = 0. \tag{5.15}$$

The desired flow percentage is calculated as follows

$$\bar{x}^f(n+1) = [\bar{x}^f(n) - \gamma\mu^f\beta^f(n)]^+_{Gf} \tag{5.16}$$

$\beta^f(n)$ is a vector consisting $\beta^{f,p}(n), p \in P(f)$ where

$$\beta^{f,p}(n) = \sum_{(i,j)\in p} \frac{\partial \bar{V}_{rt,ij}}{\partial l_{ij}}(\hat{l}^k_{ij}(n)) \tag{5.17}$$

$$\frac{\partial V_{rt,ij}}{\partial x^{f,p}}(x(n)) = \begin{cases} \frac{\partial \bar{V}_{rt,ij}}{\partial l_{ij}}(l_{ij}(n)),\ if(i,j) \in p \\ 0,\ otherwise. \end{cases} \tag{5.18}$$

where l_{ij} is the total traffic on link (i, j). Practically, node i estimates from time to time the amount of traffic through the link (i, j). These estimates are an average of a set of measurements obtained over some period of time T. Therefore, at each time n, node i has an estimate

$$\hat{l}_{ij}(n) = \sum_{m=n-T}^{n} e_{ij}(m) l_{ij}(m). \tag{5.19}$$

Where $\sum\limits_{m=n-T}^{n} e_{ij}(m) = 1$.

The computation of the projection $[\cdot]^+$ on W falls into following problem

$$\|x - x_0\|_2^2 \tag{5.20}$$

s.t.

$$Ax \leq b. \tag{5.21}$$

In our case the projection plane $W = \{x|Ax = b\}$ which is the flow conservation constraint. The solution of above problem, namely the projection of x_0 on plane W is

$$P(x_0) = x_0 + (b - a^T x_0)a / \|a\|_2^2 . \tag{5.22}$$

The convergence properties of the algorithm are shown in the following theorems. Since their proofs are similar to those in [J.N86], we do not give the proofs.

Theorem 7 *Given all the assumptions and step-size γ is small enough, $V_{rt}(x(n))$ of the algorithm converges to $\min_{x \in W} V_{rt}(x)$ and any limit point of $\{x(n)\}$ is a minimizing point. Moreover, $x^f(n) - \bar{x}^f(n)$ converges to zero for all the flow $f \in F$.*

Theorem 8 *Given all the assumptions and step-size γ is small enough, if each $\bar{V}_{rt,ij}$ is strictly convex, then the vector of link flows $l_{ij}(n)$ converges to the unique minimizing vector of the objective function $V_{rt}(x)$.*

Although the algorithm has nice theoretical features, throughout our simulation studies (see more details in Chapter 5.4), it can be shown that the algorithm is very difficult to implement with good robustness so that many routing adaptations not only cannot help the network to minimize the average delay but also create more fluctuations. The main cause of the poor robustness comes from the difficulty in accurate evaluation of the marginal delay of each link $\frac{\partial V_{rt}(x)}{\partial x_{ij}}$.

5.2.2 The Proposed Routing Algorithm

Since it is difficult to implement the asynchronous gradient-based approach to solving the original routing problem, we seek to find an approximate but simpler method to solve the routing problem. The ideal is that we allow each source node selfishly optimize its own routing policy to minimize the average end-to-end delay of its own flows. Under such selfish routing, it is know that the selfish behavior can reach a Wardrop equilibrium in which, for any demand, all paths that have positive flow will have the same path delay [J.N86]. With such relaxation, compared to the gradient-based algorithm, the selfish routing protocol does not need to evaluate the marginal delay of each link, which is exactly the partial derivative of $V_{rt}(x)$ with respect to l_{ij}. Instead, only the information of path delay is need and the computation and communication complexities for measuring and signaling the delay information are much lower. In theory, there exists performance gap between selfish routing and optimal routing (called Price of Anarchy [RT02]), existing work [QYZS03] [Xie03] shows that such performance gap is usually quite small in data networks.

The proposed adaptive routing algorithm is motivated by stochastic control theory [Gal77] [Ber84] [Kae96], and runs at each source node every updating period. This algorithm is asynchronous, so there's no requirement for synchronization between the source nodes. The end-to-end delay of each path is fed back in a reverse direction, namely, from the destination node to the source node. Each node updates the information $V_{rt}(x_{ij}^{f})$ to it's previous node in the path once every updating period. The feedback information is used by the source node to get the sum $\sum\limits_{(i,j)\in p} V_{rt}(x_{ij}^f)$, which is denoted as $v_{rt}^{f,p}$. For the source node, it also maintains the following parameters

- $\bar{x}^{f,p}$, the desired routing probability of flow f on path p. Later we will show this set of probability converges to Cesaro-Wardrop equilibrium [J.N86] in the path set.
- $x^{f,p}$, the actual routing probability.

When source node gets a new value of $v_{rt}^{f,p}$ from path p, it updates $V_{rt}^{f,p}(n+1)$ according to the following equation:

$$V_{rt}^{f,p}(n+1) = (1-a(n))V_{rt}^{f,p}(n) + a(n)v_{rt}^{f,p} \tag{5.23}$$

where $a(n) \in [0,1]$ is the learning factor of $V_{rt}^{f,p}$. If it is large, it means $V_{rt}^{f,p}$ is sensitive to the noise. If it is small, $V_{rt}^{f,p}$ depends mainly on the previous sample. However, in wireless network, $a(n)$ can not be too small because if it is too small, it can not be responsive enough to the dynamic wireless channel condition.

Then, based on the information of $V_{rt}^{f,p}$ from all the paths of flow f, the source node computes an overall V_{rt}^{f} according to

$$V_{rt}^{f}(n+1) = \sum_{p\in P^f} x^{f,p}(n)V_{rt}^{f,p}(n+1) \tag{5.24}$$

After the source node gets the overall value, it updates the desired routing probability for each path in P^f according to following equation:

$$\bar{x}^{f,p} = \bar{x}^{f,p}(n) + b(n)[\bar{x}^{f,p}(V_{rt}^{f}(n+1) - V_{rt}^{f,p}(n+1)) + \xi_i^{sd}] \tag{5.25}$$

Where $b(n) > 0$ is routing learning factor and it is used to smooth out the noise. ξ_i^{sd} is i.i.d random vector distributed uniformly on the unit ball of dimension P^f. It is used to add disturbance to avoid non-Wardrop solutions.

Then, source node projects $\bar{x}^{f,p}$ to $[0,1]^{Pf}$ to ensure that the desired routing probability is a valid set. The new desired routing probability is obtained by solving the optimization problem below:

$$\min \sum_{p \in P^f} (y^{f,p} - \bar{x}^{f,p})^2 \tag{5.26}$$

s.t.

$$\sum_{p \in P^f} y^{f,p} = 1$$

$$0 \le y^{f,p} \le 1, \;\; p \in P^f$$

The solution of above projection can easily be obtained from following

$$y^{f,p} = \bar{x}^{f,p} + \frac{1 - \sum\limits_{p \in P^f} \bar{x}^{f,p}}{|P^f|} \tag{5.27}$$

and $y^{f,p}$ becomes the new desired routing probability $\bar{x}^{f,p}(n+1)$.

Then, we update the new routing probability $x^{f,p}$ based on $\bar{x}^{f,p}$:

$$x^{f,p}(n+1) = (1-\epsilon) y^{f,p}(n+1) + \frac{\epsilon}{|P^f|} \tag{5.28}$$

where ϵ is a small constant number.

The convergence property of the algorithm is shown in the theorem below and its proof is similar to that in [J.N86] [Xie03].

Theorem 9 *With the proposed selfish routing algorithm, the routing probability of each flow converges, the estimation of minimum link delay converges under stationary routing probability, and the internal routing probabilities converge in H_s almost surely,* where H_s is the set $\{x : x^{f,p} > 0 \Rightarrow V_{rt}^{f,p} = \min_j V_{rt}^{f,j}\}$.

5.3 Solving The Scheduling Problem

The scheduling problem ($V_{sch}(.)$) derived from the original optimization problem is the classical weighted maximal link scheduling problem where the link weight is proportional to the average link delay. However, in the objective function, c_{ij} represents the average link capacity that is determined by both the link scheduling and the link transmission rate. As the MAC protocol decides the link schedule in each time slot, we need to transform the original scheduling problem to a MAC layer per time slot scheduling decision problem. Since λ_{ij} reflects the average queuing delay of packets, at the beginning of each time slot, λ_{ij} is expected to be proportional to the queue length of link (i, j). Note the both the queuing delay and the queue length can be used to evaluate the congestion level of the link. Furthermore, c_{ij} is proportional to the link transmission rate which depends on the channel quality of the link. Let h_{ij} denote the channel gain of link (i, j) and follow the gradient scheduling, the original scheduling algorithm can be transformed to the following problem: at the beginning of each time slot t, find a link schedule S such that:

$$\begin{aligned} &\max_{\mathbf{S}} \sum_{(i,j)\in E} b_{ij}(t)h_{ij}(t)I\{(i,j) \in \mathbf{S}\} \\ &s.t\ \mathbf{S} \in \bar{\mathbf{S}} \end{aligned} \tag{5.29}$$

where $I(x)$ is equal to 1 if x is true, and 0 otherwise. $\bar{S}$ is the scheduling constraint that no interference links can be active simultaneously. Thus, the scheduling problem becomes the classical throughput optimal scheduling problem. However, the problem formulation does not consider how to realize the network coding gain for given coding opportunities. Thus, we introduce how to add network coding awareness into the scheduling decision process. Following the opportunistic listening and coding principle in COPE, we assume each node that behaves as a forwarding node to forward packets to different downstream nodes has the knowledge of if the situation where the network coding is applicable to a group of receiving nodes. Such group is called a network coding group N and satisfies the following three conditions: First, all the N nodes are within the transmission range of the forwarding node for a broadcast; second, the relay node has packets to be forwarded to all N receiving nodes; third, each receiving node has the packets sent by the forwarding node to the other N-1 receiving nodes.

Given a network coding group N, we form a super link, denoted by l_N, consists of all links in N and include the super link into the link set to be scheduled. The queue length and the channel gain of the link are defined as:

$$\begin{aligned} b_{l_N} &= \max_{(i,j)\in N} b_{ij} \\ h_{l_N} &= |N| \min_{(i,j)\in N} \{h_{ij}\} \end{aligned} \tag{5.30}$$

From the definitions above, we can see that the weight of link l_N increase proportionally to $|N|$ but may be significantly decreased due to a link with poor channel condition in the group. Given similar channel gains, the network coding group will have a higher priority to be scheduled and the network coding gain can be realized as well.

5.3.1 The Link-initiated Greedy Scheduling Algorithm

Under the two-hop interference model, the scheduling problem has been proven to be NP-hard to approximate within arbitrary constant factor (i.e. APX-hard) in general networks. Therefore, it is not practical to find exactly optimal link schedule in each time slot. Recent research [JLS08] [LNS09] has shown that the greedy maximal scheduling method can derive an approximate solution within a constant factor (e.g., between $1/6$ and $1/3$ in [JLS08]) under the two-hop interference model. The main idea of the greedy maximal scheduling is to pick up the links in a decreasing order of their weights until no link can be scheduled. Once a link is selected, its interference links should be excluded from the link schedule. Although the original scheduling method is a centralized algorithm, it has been relaxed to be implemented in a distributed way with the same optimality as the centralized version. Current implementation of the distributed greedy maximal scheduling requires each link knows the weight of its interference links. This is usually implemented by explicit message exchange between interference links. However, since it is not appropriate to make the size of each time slot too large, the scheduling decisions should be made in a small time scale. Thus, the protocol that requires message passing between the interference links may cause high communication overhead as the per-link message complexity is $2N$ if unicast is used, where N is the number of interference links.

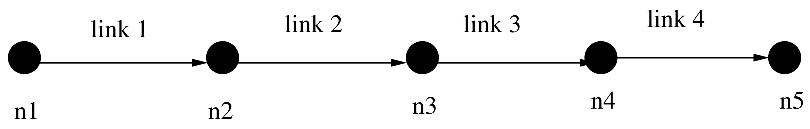


Figure 5.1: A linear network with 4 links

Without the weight information of interferenc links, the greedy maximal scheduling is inherently sequential and it appears almost as unlikely as P=NP [MR95] to implement it fast in a distributed way. To efficiently implement the distributed greedy maximal scheduling, we design a new channel contention mechanism used in each contention slot. The principle of the contention mechanism is to let the link with a larger weight have a shorter backoff period than the link with a smaller weight. As a result, a larger weight link can win the competition of the following data slot by signaling its activation information earlier than a smaller weight link. At the first glance, it is easy to design and implement such contention mechanism. However, a careless design of the mechanism may have significant performance gap compared to that of the sequential implementation of the greedy maximal scheduling. For example, as shown in Figure 5.1, a linear network consists of 5 links. Suppose their weights follow $w_1 > w_3 > w_2 > w_4 > w_5$. According to the principle of the contention mechanism, link 1's backoff timer expires first. However, since $n3$ is out of the transmission range of $n1$, $n3$ cannot receive the signal sent from $n1$ until $n2$ transmits signal to acknowledge $n1$. Thus, $n3$'s timer may expire before $n2$ sends the signal, in which case the signal from $n3$ will collide with the signal from $n1$ at $n2$ and the backoff timer of $n4$ will be stopped once $n4$ hears the signal from $n3$. Consequently, only link 1 will transmit data in the data slot. In contrast, with perfect greedy maximal scheduling, the link 3 should stop competing the channel if link 1 is activated, in which case links 1 and 4 can use the data slot. Thus, it can be seen that the performance loss under the backoff timer based contention mechanism could be quite large (e.g., 50% in this example).

To overcome the problems of high communication overhead due to frequent message passing and the potential large performance loss due to possible collision caused by poorly coordinated backoff processes, we find a new *link-based* channel contention mechanism to implement distributed greedy maximal scheduling. The contention mechanism is designed as follows: Suppose the weight of link (i, j) is w_{ij}, we have $w_{ij} = b_{ij}h_{ij}$, and the sender and receiver of link (i, j) are node i and node j, respectively. Since the information of b_{ij} and h_{ij} is available at node i, at the beginning of each contention slot, node i determines a short backoff period Δ_{ij} following:

$$\Delta_{ij} = \frac{\sigma}{w_{ij}} \tag{5.31}$$

where σ is a pre-specified system parameter. Node i periodically updates node j on the evolution of b_{ij}. Meanwhile, node j also notifies node i about h_{ij}. With the synchronized information of b_{ij} and h_{ij}, both node i and node j can calculate the weight of link (i, j) at the beginning of each contention slot, and set the same backoff period of their timer. If a node (say node i) senses channel idle when its backoff timer expires, it broadcasts a small *busy* packet to all its one-hop neighbors. A *busy* packet simply contains a few bits and has a very small transmission time. If both end nodes of link (i, j) broadcast the busy packet, their two-hop neighboring nodes will hear the packet. After broadcasting the busy packet, node i sends channel reservation packet *resv* to node j. The *resv* packet contains the ID of the receiver node (say j). If node j successfully receives the *resv* packet, it immediately replies the *clear* message to node i. When node i successfully receives the *clear* packet, node i knows link (i, j) wins the channel contention and starts to transmit data in the following data slot. Any node i hears a *busy* packet before its backoff timers expires, it simply aborts the contention for the data slot and resumes the contention at the beginning of the next contention slot. When a node has a collision happens or it decides to abort the contention, the node does not reply *clear* packet even if it can successfully decode the *resv* packet. Then, the sender of the *resv* packet will infer the failure of contention from the absence of the *clear* packet.

The link-initiated channel contention mechanism requires the link weight synchronization between the end nodes of each link, it is only feasible for unicast data transmissions. However, all network coding packets (i.e. XORed packets) are transmitted in a multicast way. How to efficiently support reliable one-hop multicast for network coding packets remains a problem. Since each node opportunistically constructs the packet set for network coding, it is very difficult to synchronize the weight of transmitting each network coding packet among the sender and the receivers. Thus, we give another signaling mechanism for the transmission of network coding packets: Similar to the case of unicast, the sender broadcasts a multicast reservation packet $m - busy$ to its *two-hop* neighbors by using a narrow band to transmit the packet. Note that, given the same transmission power, the transmission range can be increased if the transmission uses less bandwidth. Then the sender broadcasts a multicast reservation $m - resv$ packet using the normal band to its receivers. The $m - resv$ packet contains the IDs of the receivers of the XORed packet. For each intended receivers, if it successfully receive the $m - resv$ packet and does not abort the channel contention, it immediately transmits a multicast clear packet ($m - clear$) to the sender.

The sender infers the success of the channel contention by checking the received signal strength, which is denoted by RSS. If $RSS = \sum_{(i,j)\in N} h_{ij}$, where N is the set of links between the sender and the receivers, the sender knows every intended receiver replied and is ready to receive the network coding packet. Otherwise, the channel contention fails due to the collision caused by some interference link(s). When detecting channel contention failure, the senders of the competing links will start the collision resolution phase. In this phase, the channel is divided into mini-slots (e.g. 10 μs per slot). Each sender competes the channel at the beginning of the next mini-slot with a fixed probability of 0.5 until the collision is resolved or the delay of collision resolution is longer than a threshold value. In the next Chapter, we will analyze the performance and overhead of the link-initiated channel contention mechanism and how to strike the balance between the performance and overhead.

5.3.2 Protocol Analysis

In this section, we analyze the performance of the proposed distributed scheduling scheme in terms of expected contention period and the scheduling performance compared to the ideal greedy maximal scheduling. First, we analyze the expected delay of the contention process. For simplicity, we assume the channel conditions are homogeneous and their gains are i.i.d. Therefore, the cumulative distribution function and probability density function of Δ_{ij} are related to the respective distributions of h_{ij} according to:

$$F(t) = cdf_{\Delta_{ij}}(t) = 1 - cdf_{h_{ij}}(\frac{c_{ij}}{t}) \tag{5.32}$$

$$f(t) = pdf_{\Delta_{ij}}(t) = \frac{d}{dt}cdf_{\Delta_{ij}}(t) = \frac{c_{ij}}{t^2}pdf_{h_{ij}}(\frac{c_{ij}}{t}) \tag{5.33}$$

where $c_{ij} = \frac{\sigma}{b_{ij}}$.

Lemma 4 *Given a set of $M > 2$ competing links, which have the same average queue length b and the same channel condition following the pdf $f(x)$ and the cdf $F(x)$, the expected idle time due to backoff, denoted by I_b, follows:*

$$E(I_b(M)) = \frac{c}{\int_0^\infty M[1-F(t)]^{M-1}f(t)tdt} \tag{5.34}$$

where $c = \frac{\sigma}{b}$.

Proof. The backoff period lasts until the link with the largest weight triggers its presence. During a stable network, the average queue size is assumed to the constant b. Thus, the factor that affects the average idle period is the probability distribution of the highest channel gain among the n competing links. Thus, we denote the maximum channel gain by h_{max} and have its cumulative distribution function following:

$$cdf_{h_{max}}(t) = Pr\{h_{ij} < t, \forall (i,j) \in A\} = (1 - F(t))^n \tag{5.35}$$

and the probability density function is:

$$pdf_{h_{max}}(t) = \frac{d}{dt} cdf_{h_{max}} = n(1 - F(t))^{n-1} f(t) \tag{5.36}$$

Thus

$$E(h_{max}) = \int_0^{\infty} n(1 - F(t))^{n-1} f(t) t dt \tag{5.37}$$

Since $\frac{\sigma}{b}$ is constant, we have

$$E(I_b) = E(\frac{c}{h_{max}}) = \frac{c}{E(h_max)} \tag{5.38}$$

By plugging equation (5.37) into equation (5.38), we complete the proof. □

Lemma 5 *Let X_k be the random variable denoting the number of steps required to resolve a collision with k links involved. Then its expectation satisfies*

$$E(X_k) \leq 2ln(k) \tag{5.39}$$

for all k.

Proof. After each step, a fraction of k links will abort the contention when these links do not trigger at the beginning of the previous step and sense the presence of some other links. If the current number of contending links is k, its proceeds that at the next step, there will be $k - X$ contending links left, where X is a random variable ranging over the integers $1, ..., k-1$. By our collision resolution mechanism, we have $E(X) = \frac{k}{2}$. Let $g(x) = \frac{x}{2}$ and $f(m) = \int_1^m \frac{dx}{g(x)}$ for $m \geq 1$ and assume $E(X_m) \leq f(m)$ for values of m smaller than

k, consider the first step during which the number of contending links is reduced from k to $k - X$. We have:

$$\begin{aligned}
E(X_k) &= 1 + E(f(k-X)) \text{ (by induction)} \\
&= 1 + f(k) - E(\int_{k-X}^{n} \frac{dy}{g(y)}) \\
&\leq 1 + f(k) - E(\int_{k-X}^{n} \frac{dy}{g(k)})) \text{ (g(x) is nondecreasing)} \\
&= 1 + f(k) - \frac{E(X)}{g(k)} \\
&= f(k) \\
&= 2ln(k)
\end{aligned} \quad (5.40)$$

By induction, the inequality holds for all k and we complete the proof. □

A collision of two or more competing links happens in the vulnerable period when the link with the smallest backoff period triggers its presence but the signal is not sensed by other competing links. Denoting the length of the vulnerable period by v, with the results of Lemma 1 and Theorem 1 in [BKRL06] in which detailed proof can be found, we have:

Lemma 6 *Given $M \geq 2$ i.i.d positive random variables $\Delta_i, i \in [1, M]$, each with pdf $f(x)$ and cdf $F(x)$. Without loss of generality, these variables are ordered as $\Delta_1 < \Delta_2 < ... < \Delta_M$. Then the corresponding collision probability, denoted by $P_c(M)$, is given by the following equations:*

$$P_c(M) = 1 - I_c \quad (5.41)$$

where

$$I_c = M(M-1) \int_{v}^{\infty} f(y) \left[1 - F(y)\right]^{M-2} F(y-v) dy \quad (5.42)$$

Theorem 10 *Given a set of $M > 2$ competing links, which have the same average queue length b and the channel condition following the pdf $f(x)$ and the cdf $F(x)$, the expected contention time, denoted by $E(T_c(M))$, follows:*

$$E(T_c(M)) = E(I_b(M)) + D(1 + P_c(M)E(X_M)) \tag{5.43}$$

where D is the time needed for the completion of each link-initiated contention process and is defined by:

$$D = T_{busy} + T_{resv} + T_{clear} \tag{5.44}$$

Proof. The contention time consists of the idle time due to link backoff and the actual contention delay. The expected idle time is given in Lemma 1. The expected contention delay is equal to $D + P_c DE(X_M)$. Simply combining these two components, we complete the proof. $\square$

Theorem 11 *Given a network consisting of $M > 2$ links, each of which has the same queue length distribution and channel condition distribution. Suppose the per-slot aggregated throughput obtained from the sequential greedy maximal scheduling is S_G, the link-initiated scheduling has its throughput of S_L. Then, we have:*

$$(1 - p_c(M))^{\frac{M}{\Delta^*+1}} E(S_G) \leq E(S_L) \leq E(S_G) \tag{5.45}$$

where Δ^* is the average node degree in the link conflict graph of the network.

Proof. Comparing the sequential greedy maximal scheduling and the link-initiated distributed scheduling, the throughput loss of the link-initiated scheduling happens when a link which should have been scheduled to be idle starts to trigger its presence due to the incomplete channel contention information. The case is the same as that when a collision among the contending links happens. As a result, the throughput loss is related to the collision probability during the channel contention slot. To calculate the lower bound of the throughput of the link-initiated distributed channel contention, we order all the links in a non-decreasing list. We first make the pessimistic assumption that the adjacent links in the list are also adjacent in the link conflict graph of the network. Then we follow the greedy maximal scheduling to select the links should be scheduled and remove the neighboring links in the conflict graph from the list. With the link-initiated distributed contention, its expected performance follows:

$$\begin{aligned} E(S_L) &= \prod_{i=1}^{k}(1-p_c(M_k))E(S_G) + B \text{B is some nonnegative constant} \\ &\geq \prod_{i=1}^{k}(1-p_c(M_k))E(S_G) \\ &\geq (1-p_c(M))^{\frac{M}{\Delta^*+1}}E(S_G) \end{aligned} \tag{5.46}$$

The last inequality is derived from the fact that $p_c(M)$ is monotonically decreasing with M and the average number to links being selected is equal to $\frac{M}{\Delta^*+1}$. In addition, since the link-initiated channel contention cannot achieve a better performance than the sequential greedy link scheduling, we complete the proof. $\square$

From the theorem, we can see that the performance of the link-initiated contention mechanism can be very close to that of the sequential greedy maximal scheduling algorithm if the collision probability $p_c(M)$ satisfies $p_c(M)\frac{M}{\Delta^*+1} \ll 1$. With the proposed link-initiated channel contention mechanism. Here, we face a trade-off between the bandwidth efficiency and the optimality of link scheduling. The balance of the trade-off can be achieved by tuning the system parameter σ. We show an example as follows. Assuming each link has an i.i.d. Rayleigh fading channel. For link (i,j), we define $h_{ij} = |a_{ij}|^2$, where $|a_{ij}|$ is the channel gain. Since $|a_{ij}|$ is a Rayleigh random variable, it can be proven that h_{ij} are i.i.d. exponential random variables with parameter β, and the analytical h_{ij}'s distributions are:

$$F(t) = e^{-\beta c/t} \tag{5.47}$$

$$f(t) = \frac{c\beta}{t^2}e^{-\beta c/t} \tag{5.48}$$

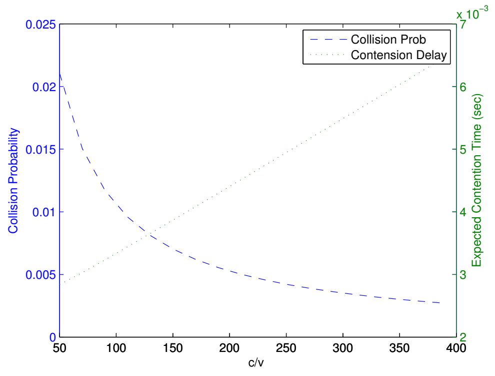


Figure 5.2: The tradeoff between the collision probability and the expected contention delay when M=3

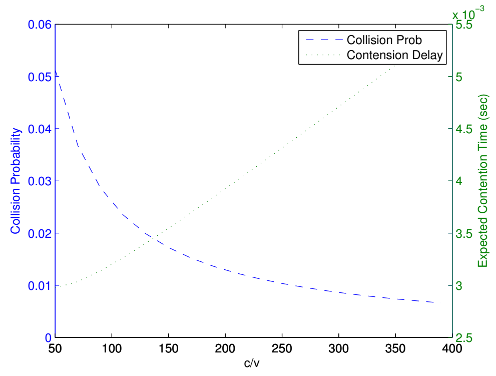


Figure 5.3: The tradeoff between the collision probability and the expected contention delay when M=6

Since the transmission distance is short (e.g. at most several hundreds of meters), the vulnerable period v is dominated by the receive-to-transmit switching time of network interfaces. Assuming the switching time is $2\mu s$, $\beta = 1$ and plugging the distribution functions in the equations (5.47) and (5.48), we calculate the numerical results of the collision probability and the channel contention delay under different values of c/v and M in Figure 5.2, 5.3 and 5.4. From the figures, we can see that there is a trade-off between the collision probability and the expected contention delay. In addition, the numerical results give the guide for deciding the length of each data slot such that the overhead of the control slot can be controlled below a certain threshold (e.g. 10%).

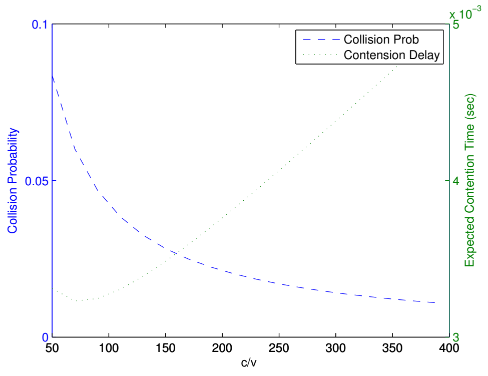


Figure 5.4: The tradeoff between the collision probability and the expected contention delay when M=10

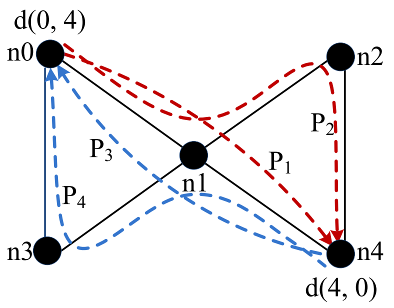


Figure 5.5: Simulation topology 1

5.4 Performance Evaluation

We evaluate the performance of the proposed routing and scheduling algorithms through simulations by using ns-2 [Pro]. For each link, the utility function is defined as $U_{ij} = -d_{ij}$, where d_{ij} is average delay of traffic experienced at link (i, j). For simplicity, for each link, we fix its data rate in the simulations and use the protocol interference model introduced in Chapter 5.3. Similar to COPE, we perform the simulations in the context of an opportunistic network coding scheme in which each node uses only packets in its local queues for coding. This allows benefits of network coding through local decisions without requiring any form of global coordination between different nodes. In addition, opportunistic listening is used to exploit the broadcast nature of the wireless medium. In particular, each node operates its network interface in the promiscuous mode in order to

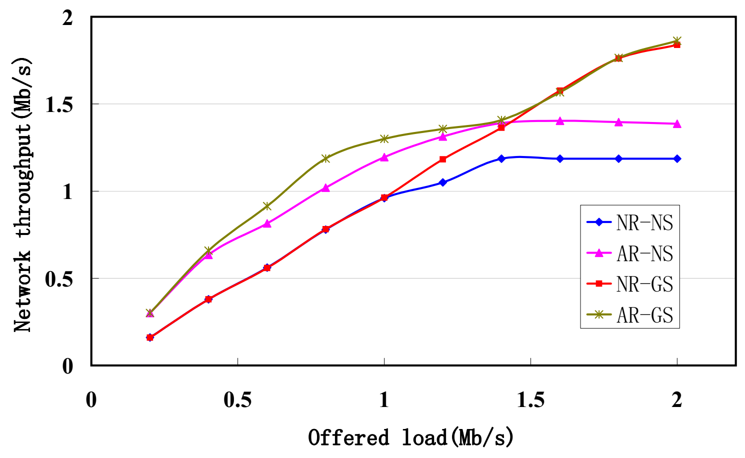
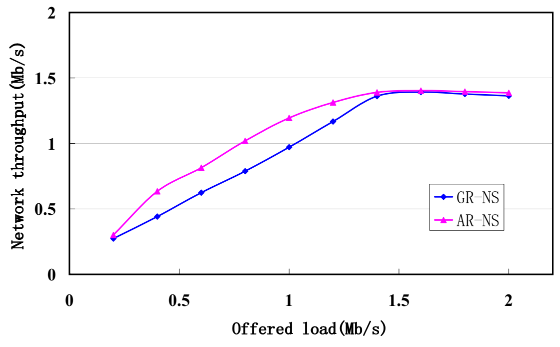
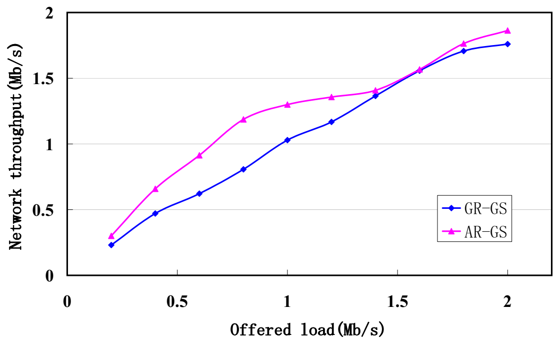


Figure 5.6: Throughput of our schemes compared to NR-NS under topology 1



(a) GR-NS vs AR-NS



(b) GR-GS vs AR-GS

Figure 5.7: Throughput comparison of GR and AR under topology 1

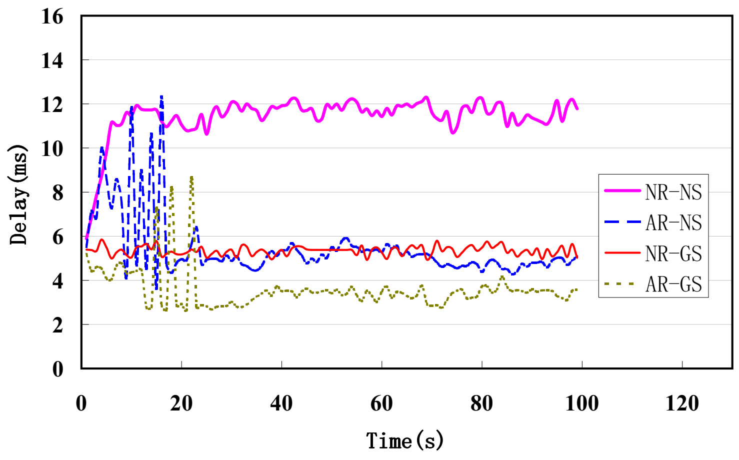
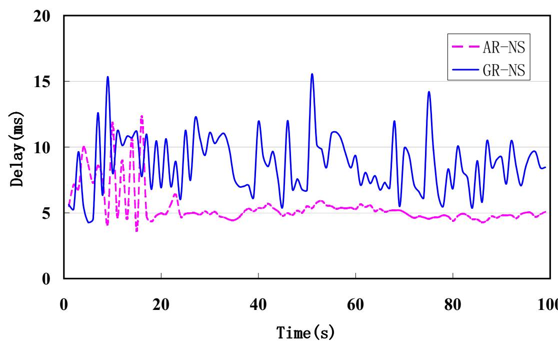
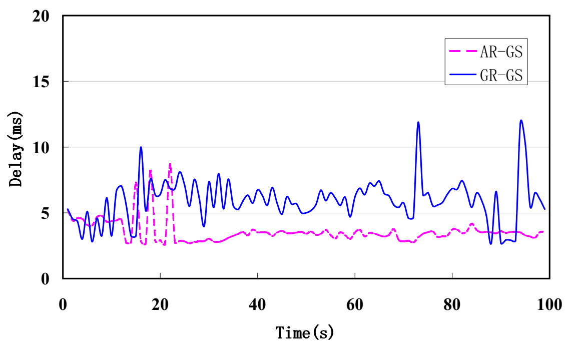


Figure 5.8: Delay of our schemes compared to NR-NS under topology 1



(a) GR-NS vs AR-NS



(b) GR-GS vs AR-GS

Figure 5.9: Delay comparison of GR and AR under topology 1

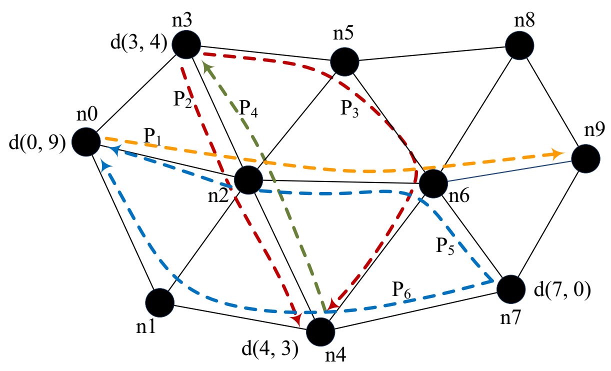


Figure 5.10: Simulation topology 2

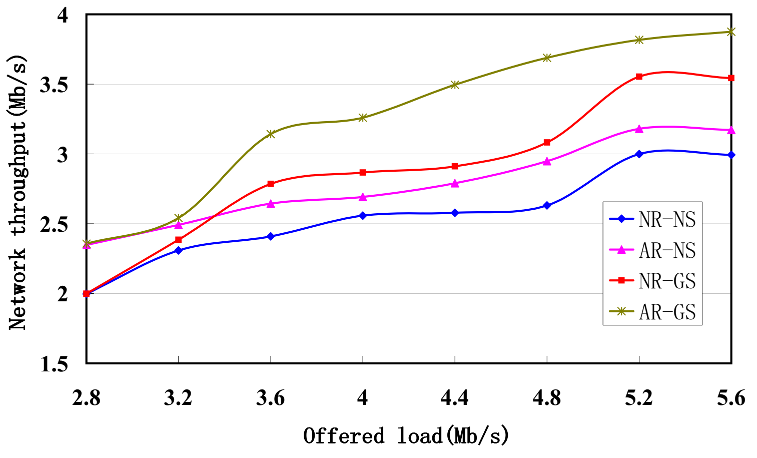
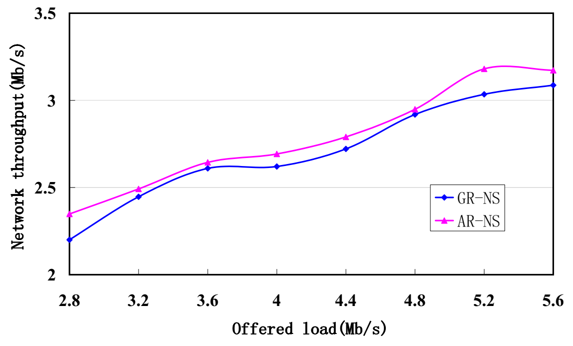
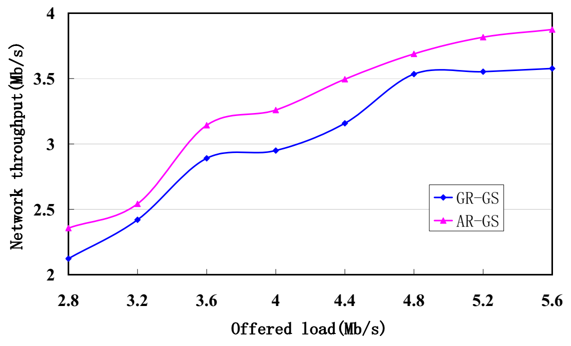


Figure 5.11: Throughput of our schemes compared to NR-NS under topology 2



(a) GR-NS vs AR-NS



(b) GR-GS vs AR-GS

Figure 5.12: Throughput comparison of GR and AR under topology 2

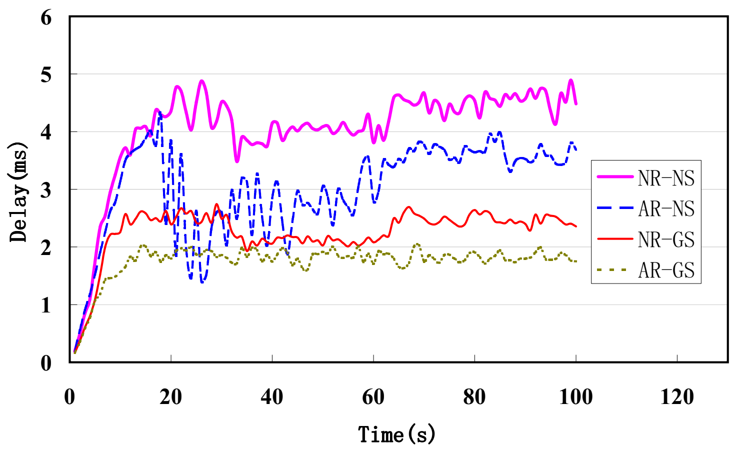
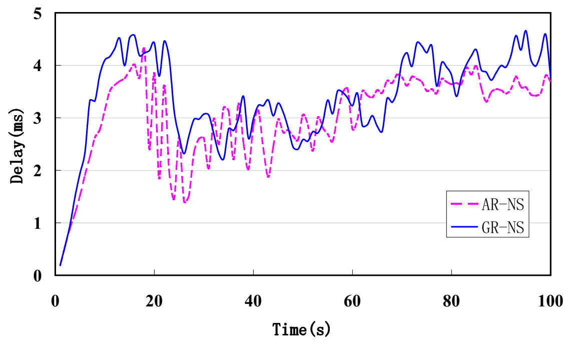
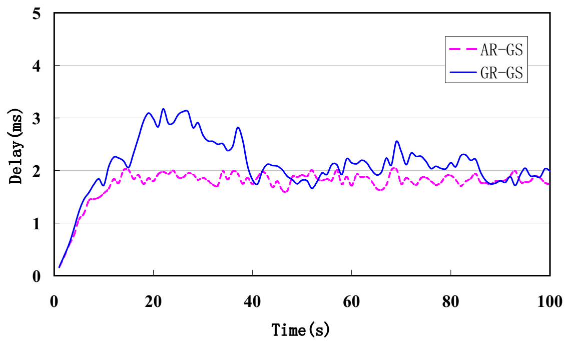


Figure 5.13: Delay of our schemes compared to NR-NS under topology 2



(a) GR-NS vs AR-NS



(b) GR-GS vs AR-GS

Figure 5.14: Delay comparison of GR and AR under topology 2

snoop on all packets communicated by its neighbors. The snooped packets then can be used for making coding decisions. The simulation duration is 100 seconds for both scenarios. The updating period of routing algorithms are 1 second in our simulation. We evaluate the network performance under two topologies shown in Figures 5.5 and 5.10.

We compare the network performance under following schemes denoted by:

- NR-NS: each node uses non-adaptive routing and the distributed greedy scheduling without the network coding awareness. With the non-adaptive routing, each node equally splits traffic along the available path set.
- AR-NS: each node uses adaptive selfish routing mechanism and the distributed greedy scheduling without the network coding awareness.
- GR-NS:each node uses the gradient-based adaptive routing and the distributed greedy scheduling without the network coding awareness.
- NR-GS: each node uses non-adaptive routing (*i.e.* equal splitting) and the network coding-aware distributed greedy scheduling mechanisms.
- GR-GS: each node uses the gradient-based adaptive routing and the network coding-aware distributed greedy scheduling mechanisms.
- AR-GS: each node uses the adaptive selfish routing and the network coding-aware distributed greedy scheduling.

We first study network topology 1 shown in Figure 5.5. The data rate of each link is 3 $Mbps$ and all the links are bidirectional. There are two traffic sources and the source-destination pairs are $(0, 4)$ and $(4, 0)$, whose paths are shown in the figure. Both traffic sources are exponentially distributed. Let P_1, P_2, P_3 and P_4 represent path $(0, 1, 4)$, $(0, 1, 2, 4)$, $(4, 1, 0)$ and $(4, 1, 3, 0)$, respectively. The throughput of our schemes compared to NR-NS is shown in Figure 5.6. The initial flow percentage for path P_1 and P_2 is 0.5, 0.5, same in the case of P_3 and P_4, which means that the network is initialized with equal splitting among paths. We can see that compared to the throughput of NR-NS, AR-NS improved the throughput by $15\% - 30\%$ under this scenario. This is contributed by adaptive selfish routing mechanism alone. From throughput of NR-GS, we can see the benefit of network coding-aware distributed greedy scheduling mechanism. When the traffic in the network is low, the beneficial is trivial, however, as the network becomes more loaded, the contribution of this mechanism is obvious compared to NR-NS. Last but not least, the throughput for AR-GS, where each node uses the adaptive selfish routing and the network coding-aware distributed greedy scheduling, improves significantly. We observe that the coexistence of flows over P_1 and P_3 boost throughput due to network coding. In turn, due to the existence of adaptive selfish routing, source node 0 will assign the flow on P_1 a larger percentage over flow on P_2 if this improves the overall throughput of traffic demand $(0, 4)$. Similarly, node 4 will assign the flow on P_3 a larger percentage over flow on P_4 if this improves the overall throughput of traffic demand $(0, 4)$. In sum, network coding will boost throughput and the adaptive selfish routing algorithm will reflect to this change of flow throughput and adjust the percentage of flows among the path set to maximize the network throughput. The routing decisions will facilitate more coding opportunities.

We also observe that the performance of gradient-based adaptive routing and adaptive selfish routing is close(see Figure 5.7). However, adaptive selfish routing is more robust and easier to implement compared to gradient-based adaptive routing since it avoids calculation of derivatives in (5.18) and the solutions of equation (5.14) (5.15) as well. In our simulation, the derivatives are implemented as following:

$$\frac{\partial \bar{V}_{rt,ij}}{\partial l_{ij}}(l_{ij}(n)) = \frac{\bar{V}_{rt,ij}(l_{ij}(n)) - \bar{V}_{rt,ij}(l_{ij}(n-1))}{l_{ij}(n) - l_{ij}(n-1)} \tag{5.49}$$

This approximation of derivatives may worsen the performance of gradient-based adaptive routing, but this indicates gradient-based adaptive routing may incur some practical issue when implemented in real world. Moreover, the setting for parameters γ and μ^f is also very challenging.

Figure 5.8 and 5.9 shows the end-to-end network delay. In Figure 5.8, we observe that our scheme reduced the delay significantly. We notice that there are bursts when adaptive selfish routing scheme is used. The reasons is that sometimes the updating process is not frequent enough to feedback the change of channel conditions. However, we can see that as the adaptation refines the routing policy quickly, and the bursts last very short time. Network delay under gradient-based adaptive routing and adaptive selfish routing are shown in Figure 5.9. We can see that adaptive selfish routing is much better than gradient-based adaptive routing in this case since it is more stable.

Similarly, We study network topology 2 shown in Figure 5.10. The data rate of each link is 4 $Mbps$ and all the links are bidirectional. There are four traffic sources and the source-destination pairs are $(0, 9)$, $(3, 4)$, $(4, 3)$ and $(7, 0)$, whose paths are shown in the figure. Both traffic sources are exponentially distributed. s-d pairs $(3, 4)$ and $(7, 0)$ have multiple paths. Let P_1=path$(0, 2, 6, 9)$, P_2=path$(3, 2, 4)$, P_3=path$(3, 5, 6, 4)$, P_4=path$(4, 2, 3)$, P_5=path$(7, 6, 2, 0)$ and P_6=path$(7, 4, 1, 0)$. The network is still initialized with equal splitting among multiple paths, so the initial flow percentage for path P_2 and P_3 is 0.5, 0.5, same in the case of P_5 and P_6. The throughput of our schemes compared to NR-NS is shown in Figure 5.11. We can see that compared to the throughput of NR-NS, AR-NS improved the throughput by $5\% - 17\%$ in this case. Since there are more traffic in this topology and some of them have only a single path, therefore the contribution of adaptive selfish routing mechanism is less significant compared to that of topology 1. Similarly, when the traffic in the network is low, we observe that the throughput of NR-GS is close to NR-NS, but when network is more loaded, we can see that the coding-aware distributed greedy scheduling improves throughput a lot. When both adaptive routing and the coding-aware scheduling are implemented, it is obvious that AR-GS improves network throughput significantly. We observe that the coexistence of flows over P_2 and P_4, P_1 and P_5 boost throughput due to network coding. In turn, adaptive routing algorithm will reflect to this change of flow throughput and assign more traffic on path that has larger throughput. Similarly to topology 1, the throughput of GR-NS vs AR-NS and GR-GS vs AR-GS are shown in Figure 5.12(a)(b), respectively and the network delays are shown in Figure 5.13 and 5.14. We can see that adaptive selfish routing achieves better performance than gradient-based adaptive routing in our simulation.

CHAPTER 6

CONCLUSIONS

This dissertation presents solutions to the problem of resource management and optimization in wireless mesh networks. There are three major solutions. They are triple-tier load balancing scheme, network coding scheme RANC, and joint optimization scheme of routing and network coding aware scheduling.

6.1 Triple-tier Load Balancing Scheme

We presented a triple-tier load balancing scheme for wireless back-hauls with OSPF routers. The proposed scheme has following features: First, based on the long-term average traffic demand and average link capacity, the first tier optimization algorithm successfully solves the problem of finding the optimal routing policy to minimize the congestion level by properly setting link weights. As a result, traffic can be distributed among a subset of shortest paths and, theoretically, the congestion level of the network can be minimized to the optimal level. With this approach, we do not need to upgrade the hardware of each router. With the constraint of equal traffic splitting of current OSPF routers, a greedy algorithm, called GMC, is proposed to approximate the optimal routing policy. Second, due to the dynamic wireless channels, the second tier adaptation algorithm is proposed to dynamically adjust the routing policy according to the channel condition. As a result, the congestion level of the network is well adapted. Third, the use of OFDMA communication technology enables us to design the dynamic subchannel assignment scheme as the third tier adaptation mechanism to further reduce the congestion level of the network. We conduct theoretical analysis to prove the properties of our scheme (e.g. approximation ratio, convergence). The performance of our scheme is evaluated through simulations and the simulation results show that our scheme is effective and can achieve near optimal load balance.

6.2 RANC

We studied a relay-aided network coding scheme, called RANC, to improve the performance of network coding in wireless ad hoc networks by exploiting the physical layer multi-rate capability. In RANC, nodes are divided into two categories: relay nodes and native nodes. Native nodes transmit their native packets at high speed to their one-hop neighboring relay nodes. Then relay nodes simply XOR both the received and their own native packets and broadcast the coded packet. Since some packet transmissions are at high speed, the overall throughput of network coding can be significantly improved. We formalize the problem of RANC via integer linear programming and analyze some design principles of RANC. Based on the analytical results, the RANC protocol was design by decomposing the original problem into two sub-problems. We evaluated the performance of RANC through simulations. The simulation results show that RANC can achieve much high throughput of network coding than COPE. In the future work, we will study RANC in irregular topologies and consider how to deal with interference from other node other than the nodes involved in network coding. We will also relax the constraint that each node should transmit only once in each round of network coding and incorporate some MAC layer relay mechanism [ZC06] [TWZA07] to boost the performance of opportunistic listen and coding.

6.3 Joint Routing and Scheduling Optimization Scheme

We have presented a joint optimization of the routing problem and coding-aware scheduling problem to maximize the total utility of wireless network. We formulated the optimization problem in wireless network and showed that it can be decomposed into two elementary problems: routing and scheduling, which are coupled through link performance factor related to queuing delay. To solve the routing problem, we proposed an adaptive selfish routing algorithm. It converges to the optimal routing even in asynchronization system. We provided the properties of the routing algorithm and the proof of them as well. Then we presented a coding-aware distributed greedy scheduling algorithm and give out its properties. We also analyzed the stability of scheduling algorithm. We conducted thorough theoretical analysis and showed the correctness of our design. In simulation, we compared the network performance under following schemes: NR-NS, AR-NS, GR-NS, NR-GS, GR-GS, AR-GS. We observe that the joint optimization of adaptive selfish routing and coding-aware distributed greedy scheduling improves network throughput due to the fact that appropriate routing can create more network coding opportunities.

6.4 Future work

In the future work, we will study RANC in irregular topologies and consider how to deal with interference from other node other than the nodes involved in network coding. We will also relax the constraint that each node should transmit only once in each round of network coding and incorporate some MAC layer relay mechanism to boost the performance of opportunistic listening and coding.

BIBLIOGRAPHY

[ACLY00] R. Ahlswede, N. Cai, S. Y. R. Li, and R. W. Yeung. Network Information Flow. *IEEE Transcations on Information Theory*, 46(4), 2000.

[Aky05] I. F. Akyildiz, X. Wang, and W. Wang. Wireless mesh networks: A survey. *Computer Networks Journal (Elsevier)*, 2005.

[and79] M. R. Garey and D.S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-completeness*. W.H. Freeman and Company, New York, NY, 1979.

[and98] Christos H. Papadimitriou and Kenneth Steiglitz. *Combinatorial Optimization: Algorithms and Complexity*. Dover Publications,INC, 1998.

[Asm03] S. Asmussen. *Applied Probability and Queues*. New York: Springer-Verlag, 2003.

[Ber84] D.P. Bertsekas, E.M. Gafni, and R.G. Gallager. Second direvative algorithms for minimum delay distributed routing in networks. *IEEE Transactions on Communications*, 8:911–919, 1984.

[BKRL06] Aggelos Bletsas, Ashish Khisti, David P. Reed, and Andrew Lippman. A simple cooperative diversity method based on network path selection. *IEEE J. SELECT. AREAS COMMUN*, 24:659–672, 2006.

[Bor97] V. Borkar. Stochastic approximation with two time scales. *Systems and Control Letter*, 29:291–294, 1997.

[Bor00] V. Borkar and S. Meyn. The O.D.E method for convergence of stochastic approximation and reinforcement learning. *SIAM Journal on Control*, 38(2):447–469, 2000.

[Cha07] P. Chaporkar and A. Proutiere. Adaptive network coding and scheduling for maximizing throughput in wireless networks. *in Proceedings of the 13th annual ACM international conference on Mobile computing and networking*, 2007.

[Chv83] V. Chvatal. *Linear Programming*. W H Freeman and Company, 1983.

[CWJ03] P. A. Chou, Y. Wu, and K. Jain. Practical Network Coding. *In Proceedings of Allerton Conference on Communications*, 2003.

[CYL05] T. W. Chim, K. L. Yeung, and K-S. Lui. Traffic distribution over equal-cost-multi-paths. *Computer Networks*, 49(4):465–475, November 2005.

[DFZ05] R. Dougherty, D. Freiling, and K. Zeger. Insufficiency of Linear Coding in Network Information Flow. *IEEE Transcations on Information Theory*, 51(8), 2005.

[D.P96] D.P.Bertsekas and J.N.Tsitsiklis. *Neuro-dynamic Programming*. Athena Scientific, 1996.

[EFK$^+$06] H. Ekstrom, A. Furuskor, J. Karlsson, M. Meyer, S. Parkvall, J. Torsner, and M. Wahlqvist. Technical solutions for the 3g long-term evolution. *IEEE Communication Magazine*, March 2006.

[For00] B. Fortz and M. Thorup. Internet traffic engineering by optimizing OSPF weights. *In Proceedings of Infocom'00*, March 2000.

[Fra06] Jean-Yves; Widmer Jorg Fragouli, Christina; Le Boudec. Network coding: An instant primer. *In ACM Computer Communication Review*, January 2006.

[Fre] Free2air. Snapshot of Berlin MANET Mesh. *http://www.free2air.org*.

[FWB06] C. Fragouli, J. Widmer, and J.-Y. L. Boudec. A Network Coding Approach to Energy Efficient Broadcasting: From Theory to Practice. 2006.

[Gal77] R.G. Gallager. A minimum delay routing algorithm using distributed computation. *IEEE Transactions on Communications*, 1:73–85, 1977.

[GSK04] Violeta Gambiroza, Bahareh Sadeghi, , and Edward W. Knightly. Dynamic routing of bandwidth guaranteed tunnels with restoration. *In Proceedings of ACM Mobicom'04*, September 2004.

[Haj88] B. Hajek and G. Sasaki. Link scheduling in polynomial time. *IEEE Trans. Inform. Theory*, 34(5):910–917, September 1988.

[Inc] Motorola Inc. Motorola Canopy. *http://motorola.canopywireless.com/*.

[JCL05] W. Jiang, D-M. Chiu, and J. C. S. Lui. On the interaction of multiple overlay routing. *Performance Evaluation*, 62(1-4):229 – 246, October 2005.

[JLS08] C. Joo, X. Lin, and N. Shroff. Understanding the capacity region of the greedy maximal scheduling algorithm in multi-hop wireless networks. In *IEEE INFOCOM'08*, 2008.

[J.N86] J.N.Tsitsiklis and D.P. Bertsekas. Distributed asynchronous optimal routing in data networks. *IEEE Transactions on Automatic Control*, 31:325–332, 1986.

[Kae96] L. Kaelbling, M. Littman, and A. Moore . Reinforcement learning: A survey. *Journal of Artificial Intelligence Research*, pages 237–285, 1996.

[KM03] R. Koetter and M. Medard. An Algebraic Approach to Network Coding. *IEEE Transcations on Networking*, 11(5), 2003.

[KqL99] Young Yong Kim and San qi Li. Capturing Important Statistics of a Fading/Shadowing Channel for Network Performance Analysis. *IEEE Journal on Selected Aeras in Communications*, 17(5):888–901, May 1999.

[KRH^{+}06] S. Katti, H. Rahul, W. Hu, D. Katabi, M. Medard, and J. Crowcroft. XORs in The Air: Practical Wireless Network Coding. *In Proceedings of ACM SIGCOMM'06*, September 2006.

[LGT07] J. Liu, D. Goeckel, and D. Towsley. Bounds on the gain of network coding and broadcasting in wireless networks. *In Proceedings of IEEE Infocom'07*, 2007.

[LL05] Z. Li and B. Li. Network coding: The case for multiple unicast sessions. *In Proceedings of Allerton Conference on Communications*, 2005.

[LLC08] Jilin Le, John C.S. Lui, and D-M. Chiu. DCAR: Distributed Coding-Aware Routing in Wireless Networks. *IEEE ICDCS'08*, 2008.

[LNS09] M. Leconte, J. Ni, and R. Srikant. Improved Bounds on the Throughput Efficiency of Greedy Maximal Scheduling in Wireless Networks. In *ACM MobiHoc'09*, 2009.

[LRK05a] D. Lun, N. Ratnakar, and R. Koetter. Efficient Operation of Wireless Packet Networks using Network Coding. *In Proceedings of The International Workshop on Convergent Technologies*, 2005.

[LRK^{+}05b] D. Lun, N. Ratnakar, R. Koetter, M. Medard, E. Ahmed, and H. Lee. Achieving Minimum cost Multicast: A Decentralized Approach Based on Network Coding. *In Proceedings of IEEE INFOCOM'05*, 2005.

[LW08] K. Li and X. Wang. Cross-Layer Design of Wireless Mesh Networks with Network Coding. *IEEE Transactions on Mobile Computing*, 2008.

[LYC03] S. R. Li, R. W. Yeung, and N. Cai. Linear Network Coding. *IEEE Transactions on Information Theory*, 49(2), 2003.

[MEHK03] M. Medard, M. Effros, T. Ho, and D. Karger. On Coding for Non-multicast Networks. *In Proceedings of Allerton Conference on Communications*, 2003.

[MR95] Rajeev Motwani and Prabhakar Raghavan. *Randomized Algorithm*. Cambridge University Press, 1995.

[NBTD07] A. Nucci, S. Bhattacharyya, N. Taft, and C. Diot. Igp link weight assignment for operational tier-1 backbones. *In ACM SIGCOMM Computer Communication Review*, 15(4):789–802, August 2007.

[Pop06] P. Popovski and H. Yomo. The anti-packets can increase the achievable throughput of a wireless multi-hop network. *IEEE International Conference on Communications (ICC'06)*, 2006.

[Pro] VINT Project. Network simulator - ns-2. *http://www.isi.edu/nsnam/ns/.*

[PSL08] D. Pompili, C. Scoglio, and L. Lopez. Multicast algorithms in service overlay networks. *Computer Communications*, 31(3):489–505, February 2008.

[QYZS03] Lili Qiu, Yang Richard Yang, Yin Zhang, and Scott Shenker. On selfish routing in internet-like environments. In *SIGCOMM '03: Proceedings of the 2003 conference on Applications, technologies, architectures, and protocols for computer communications*, pages 151–162, New York, NY, USA, 2003. ACM.

[RSW03] A. Ramamoorthy, J. Shi, and R. Wesel. On the Capacity of Network Coding for Wireless Networks. *In Proceedings of 41st Annual Allerton Conference on Communication Control and Computing*, 2003.

[RT02] Tim Roughgarden and Éva Tardos. How bad is selfish routing? *J. ACM*, 49(2):236–259, 2002.

[SRB07] S. Sengupta, S. Rayanchu, and S. Benerjee. An Analysis of Wireless Network Coding for Unicast Sessions: The Case for Code-Aware Routing. *In Proceedings of IEEE INFOCOM'07*, May 2007.

[Sri03] Ashwin Sridharany, Roch Gueriny, and Christophe Diot. Achieving Near-Optimal Traffic Engineering Solutions for Current OSPF/IS-IS Networks. *IEEE INFOCOM'03*, 2003.

[STS04] IEEE Computer Society, IEEE Microwave Theory, and Techniques Socity. 802.16-2004:Air Interface for Fixed Broadband Wireless Access Systems. *IEEE Standards*, October 2004.

[Tad03] V. Tadic and S. Meyn. Asymptotic properties of two time-scale stochastic approximation algorithms with constant step sizes. *In Proceedings of the 2003 American Control Conference*, June 2003.

[Tas92] L. Tassiulas and A. Ephremides. Jointly optimal routing and scheduling in packet radio networks. *IEEE Trans. Inform. Theory*, 38(1):165–168, January 1992.

[TWZA07] K. Tan, Z. Wan, H. Zhu, and J. Andrian. CODE: Cooperative Medium Access for Multirate Wireless Ad Hoc Network. *in Proceedings of the 4th Annual IEEE Communications Society Conference on Sensor, Mesh, and Ad Hoc Communications and Networks (SECON)*, June 2007.

[Vil99] C. Villamizar. MPLS optimized multipath MPLS-OMP. *INTERNETDRAFT,draft-villamizar-mpls-omp-01.txt*, February 1999.

[Wan01] Yufei Wang, Zheng Wang, and Leah Zhang. Internet Traffic Engineering without Full Mesh Overlaying. *IEEE INFOCOM'01*, 2001.

[Xie03] Haiyong Xie, Lili Qiu, Yang Richard Yang, and Yin Zhang. On Self Adaptive Routing in Dynamic Environments-An Evaluation and Design Using a Simple,Probabilistic Scheme. *IEEE ICNP'03*, 2003.

[YP07] H. Yomo and P. Popovski. Opportunistic Scheduling for Wireless Network Coding. *IEEE ICC'07*, 2007.

[ZC06] H. Zhu and G. Cao. rdcf: A relay-enabled medium access control protocol for wireless ad hoc networks. *IEEE Transactions on Mobile Computing*, 5(9), September 2006.

[ZCM08] J. Zhang, Y. Chen, and I. Marsic. Network Coding Via Opportunistic Forwarding in Wireless Mesh Networks. *IEEE WCNC'08*, 2008.

[ZDA06] Y. Zhu, C. Dovrolis, and M. Ammar. Dynamic overlay routing based on available bandwidth estimation: a simulation study. *Computer Networks: The International Journal of Computer and Telecommunications Networking*, 50(6):742–762, April 2006.

[ZZ09] J. Zhang and Q. Zhang. Cooperative Network Coding-Aware Routing for Multi-Rate Wireless Networks. *IEEE INFOCOM'09*, 2009.

VITA

XIAOWEN ZHANG

2008	M.S Telecommunications and Networking Florida International University Miami, FL, USA.
2010	Doctoral Candidate in Electrical Engineering Florida International University Miami, FL, USA.

PUBLICATIONS AND PRESENTATIONS

Xiaowen Zhang and Hao Zhu, "Two-tier Load Balancing in OSPF Wireless Back-hauls," in Proc. Of *IEEE INFOCOM 2007*, Anchorage, Alaska, May 2007.

Xiaowen Zhang, Hao Zhu and Jinsong Zhang, "RANC: Relay-aided Network Coding in multi-hop Wireless Ad Hoc Networks," *ELSEVIER Journal of Computer Communications*, vol. 32, no. 5, pp. 974-984, 2009.

Xiaowen Zhang and Hao Zhu, "Triple-tier Load Balancing in OFDMA Wireless Back-hauls," in submission to *Journal of Wireless Networks*.

Xiaowen Zhang and Hao Zhu, "Joint Design of Distributed Routing and Scheduling in Network Coding-Aware Wireless Mesh Networks," in submission to *IEEE Transaction on Mobile Computing*.