AN ABSTRACT OF THE DISSERTATION OF

Dong Nguyen for the degree of Doctor of Philosophy in

Electrical Engineering and Computer Science presented on June 10, 2009.

Title: Network Coding for Multi-User Wireless Networks

Abstract approved: ______________________________

Thinh Nguyen Bella Bose

Until a few years ago, wireless-capable laptops were considered novelties by many. It is now hard to find a laptop or a hand-held computing device that is not wireless-ready. As wireless devices are becoming commodities, they have also become an indispensable part of the modern society. Not surprisingly, research in wireless communication has also been significantly advanced in the past decade, to accommodate the growing demand for these wireless devices and applications. Yet, many challenges remain in transmitting information reliably, timely, and efficiently over wireless channels. Unlike wired transmissions, wireless transmissions are subjected to limited bandwidth, and are much more susceptible to environmental noises such as fading and interferences. As a result, it is difficult to transmit information reliably at high data rates. The problem is further compounded by the strict requirements on maximum delay and minimum throughput imposed by current and future multimedia applications. That said, recent advances in coding techniques, communication protocols and architectures provide an optimistic view of future wireless networks that help proliferate high quality wireless multimedia applications. One significant advance in coding theory in the past decade is *Network Coding* (NC). NC refers to the notion of mixing information from different flows at intermediate nodes in the network, and it has been shown to achieve throughput capacity. In this dissertation, we investigate NC theories and practical techniques for improving throughput and reducing delay of wireless networking applications. Specifically, the dissertation will focus on theoretical analysis of NC benefits and limitations as well as design of NC-based practical protocols for improving performance in a wireless access network such as Wi-Fi or WiMax. There are three main contributions of the dissertation. First, we propose a NC-based retransmission protocol for broadcasting information from a wireless base station to multiple users in a wireless access network. The proposed NC protocol exploits the special property of wireless transmissions that users in proximity, can listen to each other's transmissions to code the packets in such a way to increase every user throughputs. Both theoretical analysis and simulation results show a significant throughput gain when using the proposed NC protocol over the standard ARQ protocol. Second, we propose a NC-based packet scheduler at a wireless base station for delivering multimedia streams, particularly scalable video streams to multiple users in a wireless access network. We formulate the NC-based packet scheduler problem in the framework of Markov Decision Process (MDP) in which, packet delay, inter-dependency of packets, and different visual contributions of packet types are taken into account, to optimize for the overall visual qualities. We describe an optimal scheduler for transmitting scalable video streams to a small number of users. For a large number of users, we propose a heuristic, simulation-based algorithm for finding the near-optimal transmission policy. Third, we introduce Random Network Coding (RNC) techniques. More specifically, we present a prioritized RNC scheme for multimedia transmissions for multi-user in a wireless access network. We then study a real-world implementation of RNC. We describe the step-by-step design of encoding and decoding modules of RNC and measure their computational rates.

Network Coding for Multi-User Wireless Networks

by

Dong Nguyen

A DISSERTATION

submitted to

Oregon State University

in partial fulfillment of
the requirements for the
degree of

Doctor of Philosophy

Presented June 10, 2009
Commencement June 2010

Doctor of Philosophy dissertation of Dong Nguyen presented on June 10, 2009.

APPROVED:

Co-Major Professor, representing Electrical Engineering and Computer Science

Co-Major Professor, representing Electrical Engineering and Computer Science

Director of the School of Electrical Engineering and Computer Science

Dean of the Graduate School

I understand that my dissertation will become part of the permanent collection of Oregon State University libraries. My signature below authorizes release of my dissertation to any reader upon request.

Dong Nguyen, Author

ACKNOWLEDGEMENTS

I would like to express my sincere gratitude to my advisor, Dr. Thinh Nguyen, for his extraordinary support and scientific guidance throughout my graduate studies at Oregon State University. I would like to thank my co-advisor, Dr. Bella Bose for his encouragement and for providing useful insights in my research. I am grateful to Dr. Ben Lee, Dr. Alan Fern, and Dr. William H. Warnes for serving on the committee of my final oral examination and their useful feedback on my dissertation.

I am deeply grateful to my colleagues in Multimedia Information Processing and Networking (MAIN) Lab, Tuan Nguyen, Monchai Lertsutthiwong, and Kien Nguyen for their useful and interesting discussions and chatting from research to world politic, economy, and history. I am fortunate to have been in Oregon where I can find the wonderful friendship with my Vietnamese friends.

And finally, yet most importantly, I would like to express my very special gratitude to my family in Vietnam. I would like to express my special thanks to my parents for their love and unconditional support. I will also never forget the encouragement of my dear sisters, Thu Nguyen and Du Nguyen. Without their support and encouragement, I would never have been able to study in the United States and accomplish my degree.

CONTRIBUTION OF AUTHORS

Tuan Tran assisted in developing some theoretical and simulation results for Chapter 3 and Chapter 5.

Dr. Xue Yang provided some discussions on Chapter 4.

TABLE OF CONTENTS

TABLE OF CONTENTS (Continued)

TABLE OF CONTENTS (Continued)

LIST OF FIGURES

LIST OF FIGURES (Continued)

DEDICATION

For my family:

My parents Dung Nguyen and Bua Nguyen

My sisters Thu Nguyen and Du Nguyen

Chapter 1 – INTRODUCTION

1.1 Introduction

Until a few years ago, wireless-capable laptops were considered novelties by many. It is now hard to find a laptop or a hand-held computing device that is not wireless-ready. As wireless devices are becoming commodities, they have also become an indispensable part of the modern society. Not surprisingly, research in wireless communication has also been significantly advanced in the past decade, to accommodate the growing demand for these wireless devices and applications. Yet, many challenges remain in transmitting information reliably, timely, and efficiently over wireless channels.

Unlike wired transmissions, wireless transmissions are subjected to limited bandwidth, and are much more susceptible to environmental factors such as fading and interferences. Signal fading is caused by multi-path effect, a result of constructive and destructive patterns of multiple copies of signals traversing different paths to reach a receiver. Interference results from a variety of sources, usually other transmitters in the area that use the same carrier frequency. As a result, it is difficult to transmit information reliably at high data rates. In his famous 1948 papers, Shannon [1] developed the foundation of modern information theory in which he characterized information as irreducible randomness. At the heart of his information theory is the notion of channel capacity that provides the fundamental limit on what rate information can be sent over a given channel so that the bit error probability can be made to zero. Since then, much research in communications, especially wireless communications, has focused on achieving the channel capacity by utilizing a variety of approaches and their combinations, from channel coding and modulation techniques to communication protocols. It should be noted that it is possible to achieve channel capacity if there is no restriction on delay and computational resources. However, many real-world applications do impose such restrictions. Therefore, an ideal technique is one with small delay, low computational complexity, yet whose achievable throughput approaches channel capacity.

The problem of wireless transmissions is further compounded by the stricter requirements on maximum delay and minimum throughput imposed by current and future multimedia applications. For high quality video conferencing applications, the end-to-end delay should not be more than 150 *ms*, and should not be more than 300ms for a sustainable conversation. Failure to meet this requirement would hinder the ability for the two parties to communicate effectively. As for bandwidth requirement, a sustainable rate of 6 *Mbps* is required to stream a DVD-quality video.

That said, recent advances in coding techniques, communication protocols and architectures provide an optimistic view of future wireless networks that help proliferate high quality wireless multimedia applications. Before discussing the recent advances, we first briefly review some fundamental techniques for reliable transmissions in an error-prone channel.

Approaches to reliable transmission over an error-prone channel can be divided into three categories: Auto Repeat reQuest (ARQ), Forward Error Correcting (FEC), and HyBrid ARQ (HARQ) [2]. Using the ARQ protocol, the sender simply retransmits the corrupted data until it is received successfully at the receiver. For a binary erasure channel, it can be shown that this technique achieves channel capacity, thus it is attractive. However, the ARQ protocol requires a feedback channel in order for the sender to know which data was corrupted, and must be retransmitted. Many existing wireless networks such as Wi-Fi does have feedback channel, and thus for the most part, this requirement is less of a concern. Using an FEC approach, the sender generates some redundancies, then sends both redundant and original information to the receiver. If the amount of corrupted data is sufficiently small (less than the redundant data), a receiver can recover the lost data using some decoding schemes. The FEC technique is most useful when feedback channel is either not available or too costly to implement. As an example, consider satellite broadcast TV. In this setting, data are broadcast from a satellite to potentially millions of TVs, making the probability of any TV not receiving the correct data at any time close to 1. Therefore, it is extremely inefficient to employ an ARQ protocol for retransmissions since most bandwidth will be used for retransmissions. Furthermore, the lack of TV-to-satellite channels makes the retransmission approach infeasible. In these scenarios, it is preferable to employ FEC technique. Last but not least, is the the Hybrid ARQ. Often, FEC technique can introduce too much redundancy that might not be needed when channel condition is good. The HARQ technique combines both FEC and ARQ technique judiciously to improve the throughput.

One significant advance in coding theory in the past decade is *Network Coding* (NC). NC refers to the notion of mixing information from different flows at

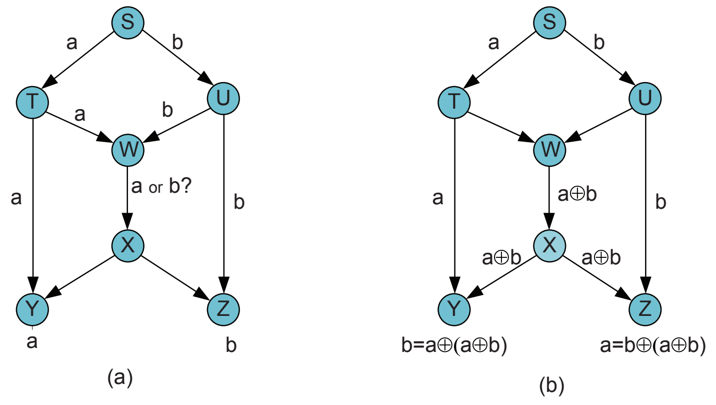


Figure 1.1: An example for using XOR NC to efficiently utilize the network bandwidth: (a) Without NC, two packets a and b are congested at link WX; (b) With XOR NC at node W, two packets can be sent to two sinks without congestion.

intermediate nodes in the network, and it has been shown to achieve throughput capacity. Below is a brief introduction to NC.

1.1.1 Network Coding Overview

Network Coding (NC) was first proposed by R. Ahlswede *et al.* [3] as a way to increase the average rate of data dissemination from a source to multiple receivers, also known as the multicast throughput. In a nutshell, NC is the generalized routing scheme that achieves the multicast capacity for a given network topology. Unlike the existing store and forward routing scheme [1] in which data is relayed hop by hop from a source to a destination without being altered, NC allows intermediate nodes in the network to mix data from different flows before forwarding the

[1] Store and forward scheme is employed in almost all the existing data communication networks.

mixed data to outgoing links. It has been shown that proper mixing of data across flows within a network, will in fact provide the maximum achievable throughput.

Fig. 1.1 shows how mixing data at intermediate nodes can increase the multicast capacity for the butterfly network. In this setting, source S wants to multicast two bits a and b to each of the two receivers Y and Z. Furthermore, assuming that the capacity of each link is 1 bit per second. As shown in Fig. 1.1(a), using the store and forward routing scheme, two bits a and b cannot be transmitted at the same time via link WX. As a result, if W sends a and b alternately, then the average throughput at each receiver is 1.5 bits per second.

Now let us allow W to mix the data coming from T and U by using XOR operation on the inputs to produce $a \oplus b$. $a \oplus b$ is then forwarded to X which then broadcasts to both Y and Z as seen in Fig. 1.1(b). Upon receiving $a \oplus b$, Y is able to reconstruct b as $b = a \oplus (a \oplus b)$ and Z is able to reconstruct a as $a = b \oplus (a \oplus b)$. Clearly, by mixing data, i.e., using NC, every receiver can obtain an average throughput of 2 bits per second.

NC has also brought a new dimension to designing architectures and protocols for emerging wireless networks. Notably, NC has been used to maximize bandwidth efficiency and/or minimize power usage by exploiting the broadcast nature of wireless medium. Often in a typical wireless transmission scenario, a receiver can listen to the transmission intended for its neighbor. This information can be used to increase the receiver's throughput. The classical example using such an approach to exchange information between two wireless nodes is proposed by Y. Wu *et al.* [4]. Fig. 1.2 depicts two users R_1 and R_2 in a wireless network that want to exchange information with each other via a common wireless router R. Using the store and forward scheme, R_1 sends bit a to R, then R relays bit a to R_2. Similarly, R_2 send bit b to R, then R relays bit b to R_1. As described, R has to perform to transmissions: one for relaying packet a and one for relaying packet b. Furthermore, because of wireless interference, total number of time slots (assuming one transmission per time slot) requires for R_1 and R_2 to exchange information is four.

On the other hand, when using NC scheme with XOR operations, the total number of transmissions can be reduced to 3 as follows. First, we note that R_1 and R_2 initially have bits a and b, respectively. In the first time slot, R_1 sends a to R. In the second time slot, R_2 sends b to R. In the third time slot, R broadcasts $a \oplus b$, resulted from bitwise exclusive or-ing of a and b. Since R_1 and R_2 can receive this bit, R_1 can recover b as $b = a \oplus (a \oplus b)$. Similarly, R_2 can recover a as $a = b \oplus (a \oplus b)$. Thus, with one broadcast from R, both R_1 and R_2 can receive the desired packet. Fig. 1.2(a) shows the existing method with 4 transmissions while Fig. 1.2(b) shows the NC approach with the number of transmissions reduced to 3.

1.1.2 Contributions of This Thesis

Despite the benefits above, the utility of NC in a general setting is not entirely clear. One must weigh NC benefits against complexity of implementing NC operations at intermediate routers. While XOR-style NC seems to have minimal computational

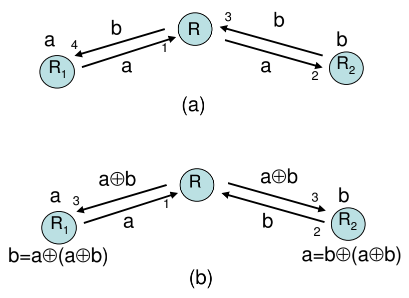


Figure 1.2: Information exchange between R_1 and R_2 via R can be employed with NC: (a) 4 transmissions are required with traditional storing and forwarding; (b) Only 3 transmissions are required with NC.

complexity, it might not be optimal in a general setting. In fact, it has been shown that large finite field operations must be used to achieve throughput optimality. In addition, supporting sophisticated NC operations at intermediate routers goes against the influential end-to-end design principle by Saltzer *et al.* [5] which argues for simple routers to increase performance and scalability. This principle has been cited in part for the huge success of the Internet.

That said, in this dissertation, we investigate NC theories and practical techniques for improving throughput and reducing delay in wireless networking applications. Specifically, the dissertation focuses on theoretical analysis of NC benefits and limitations as well as design of NC-based practical protocols for improving performance in wireless access networks such as Wi-Fi, WiMax, or cellular networks. The motivations for investigating NC in contexts of these networks are three-fold: 1) These wireless networks share similar underlying architecture, and therefore it is likely the case that the proposed NC solutions, with perhaps some slight modifications, can be applied to all of them; 2) A wireless base station or access point in these networks is an ideal place to implement NC since it typically manages less traffic as compared to that of a core router in the Internet. Thus, it is feasible to realize some sophisticated NC-based algorithms and protocols at these devices which would increase the wireless throughput, at some expense of additional complexity; 3) They carry almost all of the current wireless traffic volume. Arguably, Wi-Fi networks have revolutionized communications within homes and enterprizes. This trend will continue with more Wi-Fi devices and applications being developed everyday. In the near future, IPTV (IP based TV) and Video-on-Demand applications will deliver high definition videos from the Internet to any TV set via a wireless Access Point (AP). WiMax network has also emerged as the standard that aims to deliver data over long distances, and can potentially provide wireless broadband access as an alternative to cable and DSL. With potentially larger mobility coverage, WiMax can offer interesting multimedia applications such as mobile TV and multi-modal mobile gaming.

Fig. 1.3 captures the architecture of almost all data transmissions from the Internet to wireless users in a Wi-Fi, a WiMax, or cellular network. This critical data path supports numerous current and future applications, from delivery of digital TV programs to subscribers in a WiMax network to multimedia gaming in mobile networks. Importantly, all packets from the Internet pass through an BS (for a WiMax or cellular network) or an AP (for a Wi-Fi network). This architecture allows an BS to intercept and mix packets from different flows, i.e., performs NC in order to enhance the wireless networking performance.

In this dissertation, we argue for breaking the end-to-end principle from a cod-

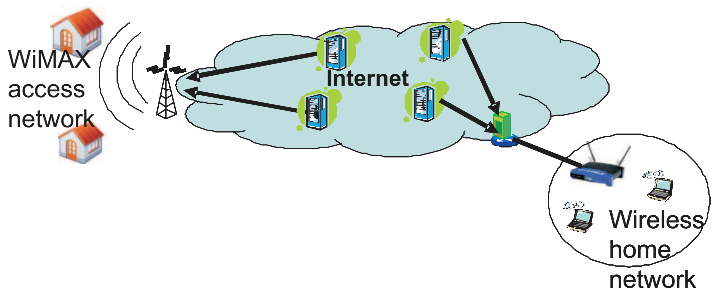


Figure 1.3: Last mile scenarios: Wi-Fi, WiMax, cellular networks.

ing perspective. We show that the wireless bandwidth can be efficiently utilized by allowing retransmissions to be performed at the BS, and *more importantly, by proper mixing of lost packets from multiple flows* [6, 7, 27]. This is in stark contrast to the existing techniques such as the Automatic Request (ARQ) or Hybrid-ARQ (HARQ) protocols [2, 9–11] where lost packets from different flows are retransmitted individually.

There are three main contributions in the dissertation. First, we tackle the problem of broadcast information from an AP to multiple users in a wireless access networks using NC. Second, we consider a packet scheduling at the AP for optimal delivery of multimedia streams to multiple users in a wireless access network using NC. Third, we investigate a prioritized RNC scheme and the actual implementation RNC using C language.

1.1.2.1 Wireless Broadcast With NC

For this contribution, we propose a NC-based retransmission protocol for broadcasting information from a wireless AP to multiple users in a wireless access network. The proposed NC protocol exploits the special property of wireless transmissions that users in proximity, can listen to each other's transmissions to code the packets in such a way to increase every user throughputs. Both theoretical analysis and simulation results show a significant throughput gain when using the proposed NC protocol over the standard ARQ protocol.

1.1.2.2 Media Wireless Transmission With NC Using MDP

Second, we propose a NC-based packet scheduler at a wireless AP for delivering multimedia streams, particularly scalable video streams to multiple users in a wireless access network. We formulate the NC-based packet scheduler problem in the framework of Markov Decision Process (MDP) and Partially Observable Markov Decision Process (POMDP) in which, packet delay, inter-dependency of packets, and different visual contributions of packet types are taken into account, to optimize for the overall visual qualities. We describe an optimal scheduler for transmitting scalable video streams to a small number of users. For a large number of users, we propose a heuristic, simulation-based algorithm for finding the near-optimal transmission policy.

1.1.2.3 Random Network Coding Techniques

Our third contribution is a prioritized RNC scheme for wireless media transmissions along with its time and computational complexities. The proposed prioritized RNC scheme is used to transmit scalable videos over wireless channels with and without feedbacks from the receivers. We specifically consider scenario with full feedback in which every packet sent from the AP has a feedback and limited feedback in which not all packets sent from the AP have feedbacks. We then provide detailed description for the design of RNC in finite field $GF(2^8)$ using C language. The encoding and decoding modules are implemented using an efficient table lookup method.

1.1.3 Thesis Organization

Our thesis is organized as follows. In Chapter 2, we provides the background on NC, wireless networking, and multimedia transmissions. Chapter 3 demonstrates our wireless broadcast using NC. In Chapter 4, we present a joint NC and scheduling scheme for wireless multimedia transmissions using Markov decision processes. We provide RNC techniques in Chapter 5 and conclusion in Chapter 6.

Chapter 2 – BACKGROUND AND PRELIMINARY

2.1 Network Coding

2.1.1 Introduction on Network Coding

NC was firstly proposed by R. Ahlswede *et al.* [3] as a solution to efficiently utilize the network transmission bandwidth. In the original NC solution as shown in the example in Fig. 1.1, XOR combination is used to encode and decode packets. In a general form, NC, which is known as random NC (RNC), uses linear random combinations and coefficients from a finite field $GF(2^q)$. In this NC scheme, all encoding and decode operations are in $GF(2^q), q > 1$. The packets, which are presented as elements of the field, are multiplied with random coefficients which are also elements of the same field, before being combined with each other to produce coded packets. The receivers, upon receiving those coded packets, decode the original packets by solving systems of equations with unknowns being the original packets.

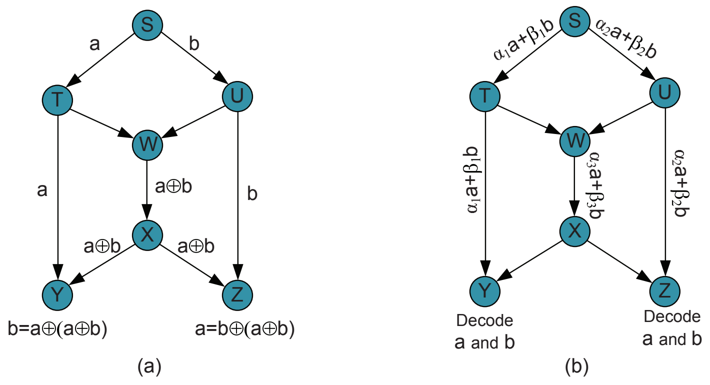


Figure 2.1: RNC achieves the same throughput gain with XOR NC.

An example of RNC is introduced in Fig.5.2. To produce random coded packets, node S generates random coefficients $\{\alpha_1, \beta_1\}$ to encode two packets a and b as $\alpha_1 a + \beta_1 b$ that is sent out on link ST. Similarly, $\{\alpha_2, \beta_2\}$ is generated and used to create another coded packet $\alpha_2 a + \beta_2 b$ that is sent out on link SU. T and U forward the information to the outgoing links without modification. Node W, however, uses random coefficients to encode two incoming packets $\alpha_1 a + \beta_1 b$ and $\alpha_2 a + \beta_2 b$ into one new coded packet $\alpha_3 a + \beta_3 b$ that is sent out on link WT. Now sink Y receives two coded packets $\alpha_1 a + \beta_1 b$ and $\alpha_3 a + \beta_3 b$ and sink Z receives $\alpha_2 a + \beta_2 b$ and $\alpha_3 a + \beta_3 b$. Those coded packets form a system of equations in which the unknowns are the original packets a and b. Solving this system of equations give us a and b. Clearly, RNC provides the same throughput to that of XOR NC. The major difference between RNC and XOR NC is that, in RNC, a node combines incoming packets originated from different sources after multiplying with some random coefficients to generate a coded packet while XOR NC allows only XOR combinations at some specific nodes. In other words, the structure of XOR NC somewhat depends on the network topology while RNC does not.

One important requirement of RNC is that the sets of coefficients used for encoding must be independent so that the systems of equations formed at the destination nodes are solvable. For instance, for the system of equations at sink Y to be solvable, $\{\alpha_1, \beta_1\}$ and $\{\alpha_3, \beta_3\}$ must be independent. It has been proved that if $GF(2^8)$ is used and all coefficients are taken randomly from the field, then the probability of independency among those sets of coefficients is almost close to one [12]. We note that the coefficients must be transmitted with the coded packets;

this can be done by inserting them into the headers of the coded packets.

2.1.2 NC for general-topology networks

NC can be designed for a general-topology network in a simple way [13]. Consider a general network presented by a directed graph $G = (V, E)$ with a vertex set V and an edge set $E \subseteq V \times V$. Assume that node v be the only source and z be the only sink in the network. The source is assumed to have the input processes presented as a vector $\mathbf{x} = (X(v,1), X(v,2), ..., X(v,\mu(v)))$ and the output processes of the sink $\mathbf{z} = (Z(v',1), Z(v',2), ..., Z(v',\nu(v')))$. We will show the relationship between the input processes and the output processes. We first have the relationship between the input processes and output processes of each node (Equation 2.1).

$$\begin{aligned}
Y(e_1) &= \alpha_{e_{1,1}}X(v,1) + \alpha_{e_{1,2}}X(v,2) + \alpha_{e_{1,3}}X(v,3) \\
Y(e_2) &= \alpha_{e_{2,1}}X(v,1) + \alpha_{e_{2,2}}X(v,2) + \alpha_{e_{2,3}}X(v,3) \\
Y(e_3) &= \alpha_{e_{3,1}}X(v,1) + \alpha_{e_{3,2}}X(v,2) + \alpha_{e_{3,3}}X(v,3) \\
Y(e_4) &= \beta e_1, e_4 Y(e_1) + \beta e_2, e_4 Y(e_2) \\
Y(e_5) &= \beta e_1, e_5 Y(e_1) + \beta e_2, e_5 Y(e_2) \\
Y(e_6) &= \beta e_3, e_6 Y(e_3) + \beta e_5, e_6 Y(e_4) \\
Y(e_7) &= \beta e_3, e_7 Y(e_3) + \beta e_4, e_7 Y(e_4) \\
Z(v',1) &= \varepsilon e_{5,1} Y(e_5) + \varepsilon e_{6,1} Y(e_6) + \varepsilon e_{7,1} Y(e_7) \\
Z(v',2) &= \varepsilon e_{5,2} Y(e_5) + \varepsilon e_{6,2} Y(e_6) + \varepsilon e_{7,1} Y(e_7)
\end{aligned}$$

$$Z(v',3) \quad = \quad \varepsilon e_{5,3}Y(e_6) + \varepsilon e_{6,3}Y(e_6) + \varepsilon e_{7,1}Y(e_7) \tag{2.1}$$

From 2.1, we can derive the transfer matrix describing the relationship between the the input processes at the source and the output processes at the sink : $\mathbf{z} = \mathbf{x}M$. We have,

$$A = \begin{bmatrix} \alpha_{e_{1,1}} & \alpha_{e_{1,2}} & \alpha_{e_{1,3}} \\ \alpha_{e_{2,1}} & \alpha_{e_{2,2}} & \alpha_{e_{2,3}} \\ \alpha_{e_{3,1}} & \alpha_{e_{3,2}} & \alpha_{e_{3,3}} \end{bmatrix} \tag{2.2}$$

$$B = \begin{bmatrix} \varepsilon e_{5,1} & \varepsilon e_{6,1} & \varepsilon e_{7,1} \\ \varepsilon e_{5,2} & \varepsilon e_{6,2} & \varepsilon e_{7,1} \\ \varepsilon e_{5,3} & \varepsilon e_{6,3} & \varepsilon e_{7,1} \end{bmatrix} \tag{2.3}$$

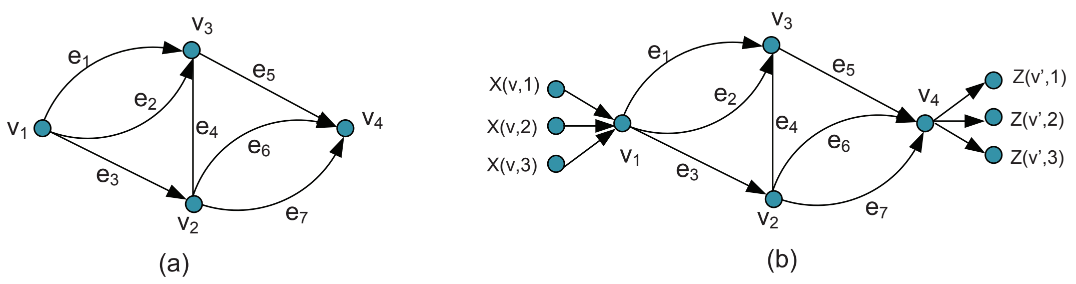


Figure 2.2: a) A point to point connection in a simple network and b) the same network nodes representing the random processes to be transmitted in the networks. [13].

Hence, the transfer matrix M is:

$$M = A \begin{bmatrix} \beta e_1, e_5 & \beta e_1, e_4 \beta e_4, e_6 & \beta e_1, e_4 \beta e_4, e_7 \\ \beta e_2, e_5 & \beta e_2, e_4 \beta e_4, e_6 & \beta e_2, e_5 \beta e_4, e_7 \\ 0 & \beta e_3, e_6 & \beta e_3, e_6 \end{bmatrix} B^T \qquad (2.4)$$

The NC solution exists only if the equation $\mathbf{z} = \mathbf{x}M$ has a solution. Because all coefficients α, β, and ε are elements of a finite field, we can always select them so that determinant of matrix M is nonzero and the NC solution exists. This means that we can always design a coding structure at each node for networks any topology such that the destination can decode the original information. As mentioned above, if finite field $GF(2^8)$ is used and coefficients are randomly selected, the the system of equations has a solution with very high probability.

2.1.3 NC for Wireless Networks

NC is proved to be very efficient for wireless networks thanks to the broadcast nature of wireless medium. In this subsection, we consider some existing NC approaches to wireless networks. Katti *et al.* [14] proposes COPE, a XOR NC model for general wireless mess networks. COPE exploits the broadcast nature of wireless transmissions, combining multiple packets to increase the throughput. The main principle of COPE is that the network nodes buffer any packet they opportunistically hear from the neighbor nodes' transmissions and then use those packets to combine with the packets they want to transmit to the next nodes. To show how COPE works in general, we consider an example of NC for X-topology wireless networks as shown in Fig. 2.3 (a). In this network, source S_1 wants to transmit packet a to destination nodes D_1 and D_2 via a common relay node R and similarly, source S_2 wants to transmit packet b to the two destination nodes via the same relay node. Without using NC, four transmissions are needed as one transmission of the packet a from S_1 to R and D_1 (i.e., one wireless broadcast transmission to two nodes) and one transmission from R to D_2 because D_2 is out of the transmission range of S_1; similarly, two transmissions of the packet b from S_2 to D_1 and D_2 via R. Using NC, however, R does not transmit a immediately; R waits for packet b. Upon receives two packets, R can broadcast XORed packet $a \oplus b$ to two destination nodes. Thus, only three transmissions are required.

NC also helps dealing with the hidden terminal problem. In the Fig. 2.3 (b), the destination D is not in the transmission range of the source S, and all the information from the source has to relay via the several intermediate nodes R_1, R_2, R_3, and R_4. The successful packet transmission probability for each link from S to an intermediate node is 20% and from an intermediate node to D is 100%. Clearly, in current IEEE 802.11 networks, the routing algorithm will pick the best path between the source and the destination to transmit a packet. It is easy to see that any path between S and D will take on average 6 transmissions to transmit a packet from the source to the destination. One improved scheme should use the path diversity to transmit packets. In this scheme (without using NC), the source will send packets to all intermediate nodes, and any node if receives a packets will forward to the destination, resulting a throughput increase to $(1 - 0.8^4) \times 100 =$

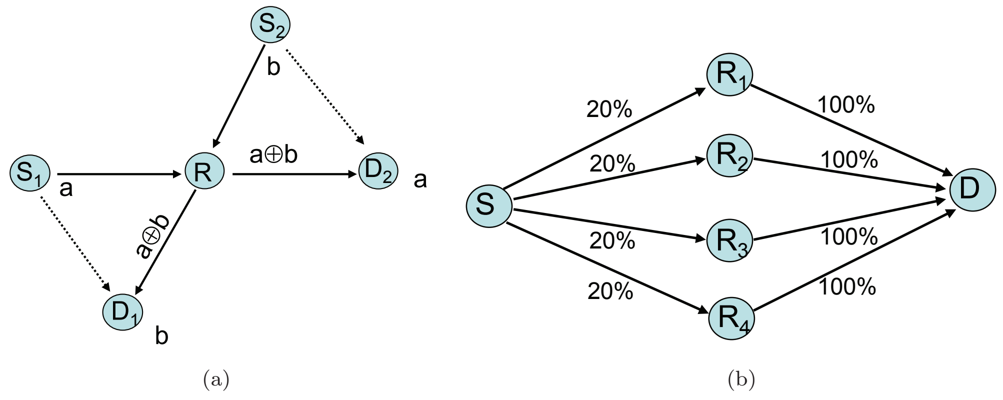


Figure 2.3: NC for wireless mess networks: (a) a X-topology network: two sources S_1 and S_2 transmit information to destination nodes D_1 and D_2 via a common relay node R; (b) a multiple-relay network: RNC reduces the duplications of packet transmissions.

59%. However, this path diversity scheme leads to the duplications of packet transmissions, i.e., the destination node may receive a duplication of a packet. NC can solve this problem. In particular, NC used at the intermediate nodes does not transmit the unmodified packet individually; it waits until receiving several packets and combines them with random coefficients before transmitting to D. By doing so, each packet received by the destination node contains information about several packets, not just only one. The original packets can be recovered by decoding the coded packets. This form of coding has been generalized for wireless mess networks as presented in [15, 16].

2.1.4 Advantages of NC

2.1.4.1 NC Increases Multicast Throughput

As seen in Section 2.1, NC improves the multicast throughput. To quantify this throughput improvement, assume that the transmission time between the source and the sinks only depends on links TY, WX, and UZ; the transmission time on other links is ignored. Therefore, within the same transmission time, NC allows a transmission of four packets to the sinks, but non-NC allows only three packets. As a result, the throughput gain from NC in this case is 33%. For wireless networks, as shown in the information exchange network using NC by Wu *et al.* [4] (Fig. 1.2), exchanging two packets from two receivers via a common router requires totally 3 transmissions instead of 4 transmissions in the traditional store and forward scheme.

2.1.4.2 NC is an Alternative for Channel Coding to Provide Reliable Communications

To provide reliable communications over unreliable channels such as wireless or the Internet, FEC, ARQ, or HARQ are normally used. From the FEC point view, some redundant information is transmitted in addition to the original information. For example, using Reed-Solomon systematic code to protect m information packets, k additional parity packets are transmitted together with m original packets. A receiver, which receives any m packets among $m + k$ packets, is able to recover

m original packets. With NC, to protect packets from errors or losses, the sender generates and transmits $m+k$ random coded packets from m information packets. If a receiver receives any m packets among those packets, it is able to decode m information packets. To this end, NC can be considered as a channel coding method to prevent packet errors or losses [17].

2.1.4.3 NC Avoids Complicated Routing and Scheduling

Because NC combines packets randomly before forwarding, a coded packet might contain information of all other packets and all coded packets have the same importance in reconstructing the original packets. This combination minimizes the complexity of routing and scheduling. This interesting feature of NC is rigorously proved in [16] by S. Chachulski *et al.*. The authors propose a NC scheme for multi-hop wireless multicast. The scheme allows network nodes to opportunistically receive, encode/decode, and forward data to other nodes without using any centralized routing protocol. The authors's emulation proves that NC remarkably outperforms the current state-of-the-art routing schemes which use centralized control.

2.2 Multi-user Single-hop Wireless Networks

Multi-user single-hop wireless transmissions captures the architecture of almost all data transmissions from the Internet to wireless users in a Wi-Fi, a WiMax, or

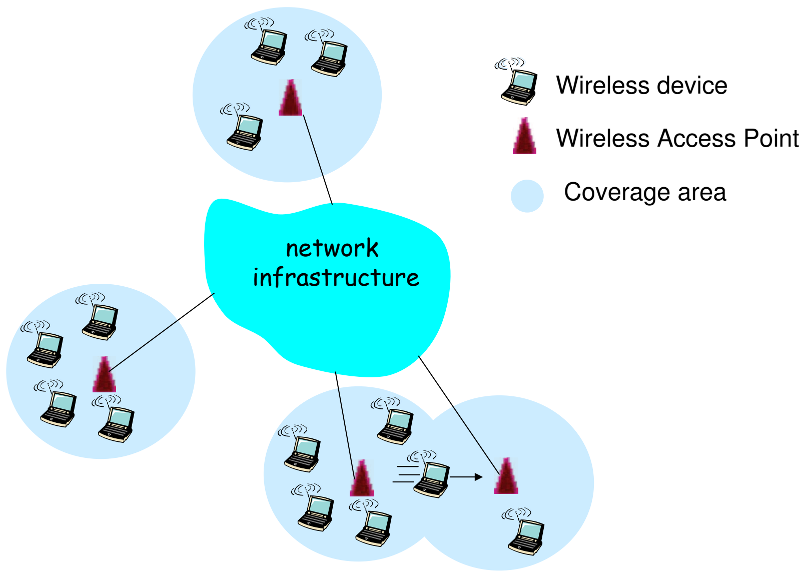


Figure 2.4: Components of wireless networks: wireless links, wireless devices, and wireless APs [18].

cellular network. Those types of networks allow users to have anywhere, anytime, untethered access to the global Internet. As depicted in Fig. 2.4, they consist of wireless links, wireless devices, and wireless APs [18].

Wireless Link: The wireless link connects the devices and the wireless APs. The wireless link has characteristics that are different from the wired links including: decreasing signal strength, interference from other sources, and multi-path propagation. Those characteristics result in more packet losses or errors in comparison with wired links. For those reasons, the wireless link is the major bottle neck in end-to-end communications between the wireless users and the sources from the Internet. The FEC or ARQ techniques are employed in order to avoid this problem.

Wireless Device: A wireless device might be a laptop, a PDA, or a cellphone which are end-systems that run end-user applications. Those devices connect to the global Internet via the wireless link and the AP. Wireless devices now become more powerful thanks to the advancement of signal processing and hardware technologies. They are getting smaller, more portable, yet having functionalities like a desktop computer.

Wireless Access Point: The AP is responsible for sending and receiving data to and from wireless users that are associated with that AP. Different from a switch or a router in wired network, the wireless AP is characterized by its interaction with wireless signal. In computer network protocol design, those characteristics are defined in two layers, the *physical layer* and *link layer*.

- *Link Layer:* The link layer lies above the physical layer and below the network layer among seven layers in the Open Systems Interconnection (OSI) model. To send a packet, a packet is passed from the network layer to the link layer; the link layer processes the packet and then passes it to the physical layer in order to transmit to other network nodes. The link layer provides reliable information delivery on a link basis by providing some possible services including framing by inserting a header into packets, link access, error detection and correction [18]. Framing is the process to encapsulate each network-layer datagram within a link-layer frame before transmission over the link. The link access provides a medium access control (MAC) protocol which handles multiple concurrent wireless connections with wireless users. Error detection and correction are the capabilities of the link layer to detect or correct some erroneous bits in the frame. This is done by inserting redundancy bits using some FEC techniques as described below. ARQ is also used in this layer to avoid lost packets by retransmission upon request from the receiver. In general, there are two types of network links: point-to-point links and broadcast links. While the point-to-point link is a transmission from one sender to one receiver via one link, the broadcast link is a transmission from one sender to multiple receivers. IEEE 802.11 networks use the broadcast link-layer technologies.

- *Physical Layer:* The physical layer is the lowest layer in the network OSI protocol stack. The main task of the physical layer is to move each bit of the data receiving from the link layer to the next network node. The famous IEEE 802.11 wireless technologies consist of 802.11a, 802.11e, and 802.11g. The physical layer, when transmitting data, employs some modulation techniques to embed the digital data into the microwave signal and then broadcasts over the air to the receivers. When receiving data, it demodulates to extract the digital bits from the received microwave signal. The 802.11a wireless LAN operates in the frequency band of 2.4-2.486 GHz with the data rate of 11 Mbps. The 802.11b wireless LAN operates at a higher frequency band, 5.1-5.8 GHz with a data rate up to 54 Mbps. And finally, the 802.11g works in the band 2.4-2.485 GHz with a data rate up to 54 Mbps. A new WiFi standard, 802.11n, employing multiple-input multiple-output (MIMO) antenna technology, is expected to provide the data rate up to 100 Mpbs.

2.2.1 Reliable Wireless Transmissions

Auto Repeat Request: The ARQ strategies are retransmission techniques associated with the link layer to correct the lost or erroneous packets. The main principle of ARQ is to detect packets with error at the receiving side and then requesting the sender to repeat the transmission of those erroneous packets. The simplest type of this protocol is the stop-and-wait ARQ [19]. The stop-end-wait ARQ ensures each packet to be received correctly before transmitting the next packet. The sender transmits a packet and then waits for the acknowledgement from the receiver. If the packet is detected without error, the receiver sends an acknowledgment back to the sender and if the corresponding acknowledgment is correct, the sender transmits the next packet. Alternatively, if the packet or the feedback is detected with error, the sender will retransmit the packet. Go-Back-N ARQ is another type of the retransmission protocol. The basic idea of Go-Back-N ARQ is to send several successive packets without waiting for acknowledgment from the receivers. The sender is not allowed to transmit packet $i + N$ before packet i gets its acknowledgement.

Forward Error Correction: FEC appends some redundant bits into a string of bits in order to correct or detect the erroneous bits. The simplest method of error detection is parity check codes. For instance, one parity bit can be appended to the string for detecting the error of a string. This check bit has value 1 if the number of 1's in the string is odd and 0 otherwise. More complicated parity check codes are horizontal and vertical parity check codes or cyclic redundancy check codes.

Other more advanced codes are Reed-Solomon codes, Digital Fountain Codes, or Low-Density Parity-Check codes (LDPC) [20].

2.3 Multimedia Networking

In recent years, parallel with the surge of wireless networks is the explosive growth of media applications-audio or video applications, primarily due to the increased high-speed and availability of the Internet and wireless devices. Multimedia applications are mainly different from the elastic application such as Web-browsing, email, or file sharing. In particular, they are sensitive to end-to-end transmission delay, transmission delay variation, but can tolerate some amount of packet losses. Multimedia applications can be classified into stored media streaming , live media streaming, or real-time interactive media streaming [18].

In *stored media streaming applications*, media content is prerecored and stored in the server. The user, once connecting to the server to view the media, can play, pause, rewind, or forward through the media content. After a few seconds of receiving the first part of media content from the server, the user can start playing without having to wait until receiving the whole media file. The user proceeds watching the media while the media content downloading is still in the process. Many video web sites such as Youtube, CNN, or Google video are typical examples of this type of applications. The user can stream from a short video clip to a hour-long movie from those sites with a delay of few second after receiving the media file.

In *live media streaming applications*, users are allowed to receive a live audio or video produced by the Internet radio or television stations like traditional radio or television broadcast stations. There are many such online radio or live television broadcast stations nowadays. In a simple streaming architecture, a station with a large upload bandwidth allow users to direct connect to the server to initiate the live streaming from the server. Some online radio station such as online Bloomberg radio or online BBC radio, or live television station such as ESPN or CNN live provide those services. The limitation of those services is that it can serve limited a number of users at the same time due to the shortage of servers' upload bandwidth. The Peer-to-Peer video streaming is promising to be able to solve this problem. Peer-to-Peer is a application-layer multicast network in which each user plays both as the client and the server and functions as a relay node for other users to stream the media content. Thus it is able to reduce the concurrent downloading from the central server. Well-known P2P streaming networks such as PPlive [21] or Joost [22] are examples of this type of application.

Real-time interactive media applications are refereed as telephone or video conference over the Internet. For instance, two users can make both a voice chat and a video chat using Yahoo Messenger, or Skype program. In those interactive live streaming applications, the media transmission delay should not exceed some certain amount, otherwise, can result in frustrating. For instance, the delay in voice chatting should not exceed 400 ms.

2.3.1 Media Streaming over the Internet

Internet provides a best-effort service in the sense that it makes its best effort to move data from the source to the destination as quickly as possible. The best-effort service does not guarantee the packet transmission with a certain amount of end-to-end delay, percentage of packet losses or transmission delay variations. Two popular services of the best-effort Internet are UDP and TCP. UDP is a connectionless service, providing an unreliable data transfer service. It does not guarantee that the message will reach the intended receivers. TCP, on the other hand, is a connection-oriented service, providing a reliable end-to-end data transfer by employing some data protection mechanism such as the ARQ method. While UDP is not a reliable service, it has a smaller delay than TCP. UDP is normally used for P2P streaming networks in which a peer downloads multiple parts of the media content from different server at the same time. P2P streaming networks such as PPlive or Joost use UDP for streaming the media data. TCP, on the other hand, is used in most server-based media streaming systems. Most online video web sites or video-shared networks such as Youtube or Metacafe use this protocol. Youtube or Metacafe [23] use the edge architecture which puts multiple servers containing media contents to the edge of the Internet so that the users can connect to the media sources with low latency and high throughput.

While UDP and TCP are not suitable for real-time applications, the Real-time Streaming Protocol (RTSP), Real-time Transport Protocol (RTP), and the Real-time Transport Control Protocol (RTCP) were specifically designed aiming at

Internet media streaming applications. RTSP allows multimedia users to control the playback of continuous media such as pausing, repositioning, or fast-forwarding the media. The process of exchanging control messages between the client and the media server is similar to that of HTTP. The RTP and its companion protocols, RTCP typically runs on top of UDP. The sender encapsulates a media chunk within a RTP packet and sends via UDP. RTP defines the packet header including payload type, sequence number, timestamps, and synchronization source identifier. RTCP is a protocol which can be used in conjunction with RTP by providing control information and statistics such as number of packets sent, number of packets lost, and inter-arrival jitter for a RTP flow. Those protocols do not provide any mechanism to ensure the timely delivery of packets or any other QoS guarantees.

Chapter 3 – WIRELESS BROADCAST WITH NETWORK CODING

1) D. Nguyen, T. Nguyen, and B. Bose, *"Wireless broadcast using network coding,"* in Third Workshop on Network Coding, Theory, and Applications, January 2007.

2) D. Nguyen, T. Tran, T. Nguyen, and B. Bose, *Wireless Broadcast Using Network Coding,* IEEE Transactions of Vehicular Technology, 2008.

3.1 Introduction

Broadcast is a mechanism for disseminating identical information from a sender to multiple receivers. It is widely employed in many applications, ranging from satellite communications to Wireless Local Area Network (WLAN). Reliable broadcast requires that every receiver must receive the correct information sent by the sender. When the communication channels between a sender and receivers are lossy, some appropriate error control schemes must be used to provide reliable transmissions. Depending on applications, these schemes can be classified into two main approaches: Automatic Repeat ReQuest (ARQ) and Forward Error Correction (FEC).

Using the ARQ approach, the sender may have to rebroadcast the lost packet to all the receivers, even though there may be only one receiver that did not receive that packet correctly. The ARQ approach assumes that a feedback channel is available so that the receiver can communicate to the sender on whether or not it receives the correct data. On the other hand, using the pure FEC approach, the sender generates some redundancies, then broadcasts both redundant and original information to the receivers [9]. If the amount of lost data is sufficiently small (less than the redundant data), a receiver can recover the lost data using some decoding schemes.

For satellite TV applications, the TV signals are broadcasted from a satellite to potentially millions of TVs, making the probability of any TV not receiving the correct signal at any time close to 1. Therefore, it is extremely inefficient to employ an ARQ protocol for retransmissions since most of the bandwidth will be used for retransmissions. Furthermore, the lack of TV-to-satellite channels makes the retransmission approach infeasible. In these scenarios, it is preferable to employ a pure FEC technique.

In other settings, e.g., WLAN, there are relatively few devices and the communication channel is relatively reliable. Therefore, the retransmission approach may be more bandwidth-efficient than that of the FEC approach since redundant information is not added in every transmission. Furthermore, in practice, FEC alone cannot guarantee reliable delivery due to a non-zero chance that a receiver may not be able to recover the data from a single transmission.

This chapter explores the efficient retransmission-based broadcast schemes in single-hop wireless networks. Efficient schemes can be used in WLAN to reduce the time required to copy a large file from one computer to multiple computers simultaneously. It can also be used to broadcast music or important announcements in multiple rooms in a small building, even though a highly reliable audio signal is probably not needed in most situations. In addition, an efficient retransmission-based wireless broadcast scheme might be beneficial to the emerging wireless standard WiMAX (Worldwide Interoperability for Microwave Access). WiMAX aims to enable wireless data transmissions over long distances, and can potentially provide wireless broadband access as an alternative to cable and DSL. In this WiMAX setting, at any point in time, a few homes may want to watch the same broadcast digital TV program through the Internet, i.e., these homes may want to receive the same TV signals. Therefore, a WiMAX broadcast station can view the data delivery as multiple wireless broadcast sessions (TV channels) with each session involving a few homes. By optimizing these wireless broadcast sessions, the wireless bandwidth can be efficiently used. To this end, we propose some broadcast schemes that combine NC and retransmission to utilize bandwidth efficiently. Our proposed NC schemes are designed for broadcast in single-hop wireless networks. The main idea for the proposed schemes is based on the observation that, at a certain point in time, many receivers may have disjoint lost packets. Thus, the sender may XOR these lost packets together and broadcast it to all the receivers. Upon receiving this XOR packet, a receiver will be able to recover its lost packet by XORing the XOR packet with the certain packets that it has received previously. As such, one transmission from the sender will enable multiple receivers to recover their lost packets, thus efficiently utilizing the wireless bandwidth. We will elaborate on how to do this shortly and show that this approach can reduce the transmission bandwidth significantly.

3.2 Broadcast Schemes

To describe our proposed schemes, we make the following assumptions for all the broadcast schemes:

1. There are one sender and $M > 1$ receivers.

2. Data is sent in packets, and each packet is sent in a time slot of fixed duration.

3. The sender has access to the information on packet losses of all the receivers

at any time slot. This can be accomplished through the use of positive and negative acknowledgments (ACK/NAKs). For simplicity, we assume that all the ACK/NAKs are instantaneous, i.e. the sender knows (a) whether or not a packet is lost and (b) the identity of the receiver with the lost packet instantaneously. This implicitly assumes that ACK/NAKs are never lost. This assumption is not critical since one can easily incorporate the delay and bandwidth used by ACK/NAKs into the analysis. One can also consider an alternative view in which, if an ACK or NAK is lost, the corresponding data packet is also considered lost. Thus, the assumption of reliable ACK/NAK messages can be relaxed by changing the packet loss probabilities at the receivers to reflect the loss rates in both data and feedback channels.

4. Packet loss at a receiver i follows a Bernoulli trial with parameter p_i. This model is clearly insufficient to describe many real-world scenarios. However, this model is only intended for capturing the essence of wireless broadcast. One can develop a more accurate model, at the cost of complicated analysis. In fact, we will provide some results when using a slightly more accurate model that reflects the correlated losses among the receivers in Section 3.3.3. We will also provide simulation results using a simple two-state Markov model to characterize packet losses.

3.2.1 Broadcast Schemes without NC

Scheme A *(Memoryless receiver).* In this scenario, a receiver sends a NAK immediately whenever there is a packet loss in the current time slot, regardless of whether it has received this packet correctly in some previous time slots (hence memoryless). This situation arises when a receiver receives a correct packet, but this packet was lost at some other receivers at some previous time slots. Hence, the sender has to retransmit this packet. If this packet is now lost in the current time slot, a memoryless receiver would automatically request a retransmission, even though it has previously received that packet. This scheme is clearly suboptimal in terms of bandwidth utilization as it implies that the sender has to resend a packet until *all* the receivers receive this packet correctly and *simultaneously.*

Scheme B *(Typical ARQ scheme).* In this scenario, the receiver sends a NAK immediately only if there is a packet loss in the current time slot and this packet has not been received correctly in any previous time slot. This scheme is clearly superior to scheme A in terms of bandwidth utilization since it never requests a retransmission for a packet that it already received successfully.

3.2.2 Broadcast Schemes with NC

Scheme C *(Time-based retransmission).* In this scheme, the receiver's protocol is similar to that of the receiver in scheme B in that, it sends the NAK immediately if it does not receive a packet correctly. However, the sender does not retransmit the lost packet immediately when it receives a NAK. Instead, the sender maintains a list of lost packets and their corresponding receivers for which their packets are lost. The sender waits until N packets have been transmitted before any retransmission takes place. During the retransmission phase, the sender forms a new packet by XORing a maximum set of the lost packets from different receivers before retransmitting this combined packet for all the receivers. The *combined* packets may be lost during the retransmission, and these packets will be retransmitted until all the receivers receive this packet. The sender keeps sending out the *combined* packets until there are no more lost packets on the list, it then resumes the transmission of a different set of packets.

Upon successfully receiving a *combined* packet, a receiver is able to recover its lost packet by XORing this combined packet with an appropriate set of previously successful packets. The information on choosing this appropriate set of packets is included in the packets sent by the sender. To illustrate this, Fig. 3.1 shows a pattern of lost packets (denoted by the crosses) for two receivers R_1 and R_2. The combined packets are $a_1 \oplus a_3$, $a_4 \oplus a_5$, a_7, a_9, where a_i denotes the i^{th} packet. Note that, if packet $a_1 \oplus a_3$ is not received correctly at any receiver, this packet is

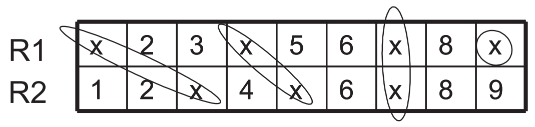


Figure 3.1: *Combined packets for time-based retransmission:* $a_1 \oplus a_3$*,* $a_4 \oplus a_5$*,* a_7*,* a_9*;* $N = 9$

retransmitted until all the receivers receive this packet correctly, but might not be simultaneously. Receiver R_1 recovers packet a_1 as $a_3 \oplus (a_1 \oplus a_3)$. Similarly, receiver R_2 recovers packet a_3 as $a_1 \oplus (a_1 \oplus a_3)$. When the same packet loss occurs at both receivers R_1 and R_2, the encoding process is not needed and the sender just has to retransmit that packet alone. Note that, the sender has to include some bits to indicate to a receiver which set of packets it should use for XORing. Assuming that all the retransmissions are correctly received at all the receivers at the first attempt, then clearly the number of retransmissions for this scheme is only 4 while it is 6 for scheme B.

Scheme D *(Improved time-based retransmission).* Scheme C is suboptimal because the sender has to retransmit the same combined packet even though some receivers may receive it. An improved scheme is to have the sender dynamically changes the combined packets based on what the receivers have received. For example, Fig. 3.2 shows the same pattern of lost packets as in the previous scenario. Now, suppose the packet $a_1 \oplus a_3$ is lost at receiver R_2, but is received correctly at receiver R_1. In this case, instead of retransmitting packet $a_1 \oplus a_3$, the sender can transmit packet $a_3 \oplus a_4$. Clearly, on average, the number of transmissions can be further reduced using this scheme.

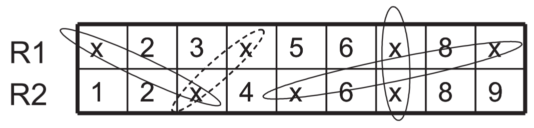


Figure 3.2: *Combined packets for improved time-based retransmission: $a_1 \oplus a_3$, $a_3 \oplus a_4$, $a_5 \oplus a_9$, a_6; $N = 9$.*

Remarks: Note that a larger buffer size N results in better bandwidth efficiency. However, it may incur an unnecessary long delay for some packets. This may be acceptable for file transfer, but may not be suitable for multimedia applications. Choosing an optimal value for N for the multimedia applications with certain delay requirements is beyond the scope of this thesis. However, we envision that a good scheme is one that dynamically changes the value of N based on the current network conditions and the application delay requirement. When $N = 1$, the NC scheme reduces to the scheme B. In the next section, we derive a few theoretical results on transmission bandwidths for different schemes with infinite and finite buffer sizes.

3.3 Transmission Bandwidth Analysis

We define the transmission bandwidth as the average number of transmissions required to successfully transmit a packet to all the receivers. Let η_A, η_B, η_C, and η_D denote the transmission bandwidths using schemes A, B, C, and D, respectively. Let M denote the number of receivers, and p_i denote the packet loss probability of receiver i. We first discuss the non-NC schemes A and B.

3.3.1 Non-NC Schemes A and B

We begin with a special case where there are only two receivers with the packet loss probabilities of p_1 and p_2. We have the following results:

Proposition 3.3.1. The transmission bandwidth of scheme A with two receivers is

$$\eta_A = \frac{1}{(1-p_1)(1-p_2)} \tag{3.1}$$

and using scheme B is

$$\eta_B = \frac{1}{1-p_1} + \frac{1}{1-p_2} - \frac{1}{1-p_1p_2}. \tag{3.2}$$

Proof. For scheme A, the proof is simple. As described in Section 3.2, the sender has to retransmit the packets until both receivers receive the correct packets simultaneously. Since the packet loss is independent and uncorrelated between the receivers (Bernoulli trial), the number of transmission attempts before both receivers correctly receive the data follows the geometric distribution with the parameter $(1-p_1)(1-p_2)$. Therefore, the average number of transmissions per successful event is $\frac{1}{(1-p_1)(1-p_2)}$.

For scheme B, let X_1, X_2 be the random variables denoting the numbers of attempts to successfully deliver a packet to R_1 and R_2, respectively. Then, the number of transmissions needed to successfully deliver a packet to both receivers is the random variable $Y = \max\{X_1, X_2\}$. We have

$$P[Y \le k] = P\left[\max\{X_1, X_2\} \le k\right] = \prod_{i=1}^{2} P[X_i \le k] = \prod_{i=1}^{2}(1-p_i^k). \tag{3.3}$$

Therefore,

$$P[Y = k] = \prod_{i=1}^{2}(1-p_i^k) - \prod_{i=1}^{2}(1-p_i^{k-1}) \tag{3.4}$$

and the average number of transmissions per successful packet is

$$\begin{aligned}\eta_B &= E[Y] = \sum_{k=1}^{\infty} k\left(\prod_{i=1}^{2}(1-p_i^k) - \prod_{i=1}^{2}(1-p_i^{k-1})\right) \\ &= \sum_{k=1}^{\infty} k(p_1^{k-1}-p_1^k) + \sum_{k=1}^{\infty} k(p_2^{k-1}-p_2^k) + \sum_{k=1}^{\infty} k(p_1^k p_2^k - p_1^{k-1}p_2^{k-1}) \\ &= \frac{1}{1-p_1} + \frac{1}{1-p_2} - \frac{1}{1-p_1p_2}. \qquad (3.5)\end{aligned}$$

■

Theorem 3.3.1. The transmission bandwidth of scheme A with M receivers is

$$\eta_A = \frac{1}{\prod_{i=1}^{M}(1-p_i)} \qquad (3.6)$$

and using scheme B is

$$\eta_B = \sum_{i_1,i_2,...,i_M} \frac{(-1)^{i_1+i_2+...i_M-1}}{1-p_1^{i_1}p_2^{i_2}...p_M^{i_M}}, \qquad (3.7)$$

where $i_1, i_2, ..., i_M \in \{0,1\}$, $\exists i_j \neq 0$.

Proof. For scheme A, using the same argument for two receivers, the number of transmissions before all M receivers correctly receive a packet follows the geometric distribution with the parameter $\prod_{i=1}^{M}(1-p_i)$. Therefore, the average number of transmissions per successful packet is $\eta_A = \frac{1}{\prod_{i=1}^{M}(1-p_i)}$.

For scheme B, let $R_1, R_2, ..., R_M$ denote the receivers with the corresponding packet loss probabilities $p_1, p_2, ..., p_M$, respectively. The number of transmissions needed to successfully deliver a packet to all receivers is the random variable

$Y = \max_{i\in\{1,..,M\}}\{X_i\}$, where X_i is the random variable denoting the number of attempts to successfully deliver a packet to R_i. We know that $P[Y \le k] = P\left[\max_{i\in\{1,..,M\}}\{X_i\} \le k\right] = \prod_{i=1}^{M}(1-p_i^k)$. Therefore,

$$P[Y = k] = P[Y \le k] - P[Y \le k-1] = \prod_{i=1}^{M}(1-p_i^k) - \prod_{i=1}^{M}(1-p_i^{k-1}). \tag{3.8}$$

Thus the average number of transmissions per successful packet is

$$\begin{aligned}
\eta_B &= E[Y] = \sum_{k=1}^{\infty} kP[Y=k] = \sum_{k=1}^{\infty} k\left(\prod_{i=1}^{M}(1-p_i^k) - \prod_{i=1}^{M}(1-p_i^{k-1})\right)\\
&= \sum_{k=1}^{\infty} k\Bigg((-p_1^k + p_1^{k-1}) + ... + (-p_M^k + p_M^{k-1}) + (-1)^1((p_1^k p_2^k - p_1^{k-1}p_2^{k-1}) + ...\\
&+ (p_{M-1}^k p_{M-2}^k - (p_{M-1}^{k-1}p_{M-2})^{k-1}) + ... + (-1)^M(p_1^k p_2^k ... p_M^k - p_1^{k-1}p_2 ... p_M^{k-1})\Bigg)\\
&= \sum_{i_1,i_2,...,i_M} \frac{(-1)^{i_1+i_2+...+i_M-1}}{1-p_1^{i_1}p_2^{i_2}...p_M^{i_M}},
\end{aligned} \tag{3.9}$$

where $i_1, i_2, ..., i_M \in \{0,1\}$ and $\exists i_j \ne 0$.

■

Note that, with $p_1 = p_2 = ... = p_M = p$,

$$\eta_B = \sum_{k=1}^{M} \frac{(-1)^{k-1}\binom{M}{k}}{1-p^k}. \tag{3.10}$$

3.3.2 NC Schemes C and D

Unlike the schemes A and B, scheme C has one additional parameter, namely, the size of the buffer used to maintain a list of receivers and their corresponding lost packets. When a small buffer is used, there may not be sufficiently many lost packets for generating the combined packets, which can reduce the bandwidth efficiency. On the other hand, when a large buffer is used, the bandwidth efficiency improves at the expense of increased delays for some packets. This approach is acceptable for file transfer applications. We now provide an asymptotic result when the buffer size N and the number of packets to be sent T are sufficiently large. Since it is not beneficial to have $N > T$, we assume $T = N$ and N is sufficiently large. We have the following results for two receivers.

Proposition 3.3.2. The transmission bandwidth of scheme C with two receivers where $p_1 \leq p_2$ and N is sufficiently large is

$$\eta_C \sim 1 + \frac{p_1}{1-p_1} + \frac{p_2}{1-p_2} - \frac{p_1}{1-p_1p_2}. \tag{3.11}$$

Proof. The transmission bandwidth depends on how many pairs of lost packets one can find in order to generate the combined packets. Let random variable Z_i denote the number of lost packets at receiver R_i after N transmissions. We first prove that $P(Z_2 - Z_1 < 0) \to 0$ as $N \to \infty$. Since the packet loss rates follows Bernoulli trials, Z_1 and Z_2 are Binomial random variables. When $N \to \infty$, according to the central limit theorem, distributions of Z_1 and Z_2 approach the distribution of a Gaussian variable. Let $Z = Z_2 - Z_1$, since Z_1 and Z_2 are independent, we have

$$\begin{aligned} \mu_Z &= E[Z_2] - E[Z_1] \\ &= N(P_2 - P_1) \qquad (3.12) \\ \sigma_Z^2 &= var(Z_2) + var(Z_1) \\ &= N[(p_2(1-p_2) + p_1(1-p_1)] \qquad (3.13) \end{aligned}$$

Thus, the probability density function of Z can be written as

$$P(Z) = \frac{1}{\sqrt{2\pi}\sigma_Z} e^{-\frac{(Z-\mu_Z)^2}{2\sigma_Z^2}} \qquad (3.14)$$

When $N \rightarrow \infty$, both μ_Z and σ_Z increase. In particular, μ_Z increases with an order of N while σ_X increases with an order of $\sqrt{N}$. Hence, the tail area, i.e., $P(Z < 0)$, asymptotically goes to 0 as $N \rightarrow \infty$. Consequently, one can combine Z_1 pairs of lost packets since $Z_1 \leq Z_2$ and there are $Z_2 - Z_1$ lost packets from R_2 that need to be retransmitted alone. Because $Z_1 = Np_1$ and $Z_2 = Np_2$, we have

$$n \sim N + Np_1E[X_1] + N(p_2 - p_1)E[X_2], \qquad (3.15)$$

where X_1 and X_2 are the random variables denoting the numbers of transmission attempts before a successful transmission for the combined and non-combined packets. Now, $E[X_2] = \frac{1}{1-p_2}$ since X_2 follows the geometric distribution. From

Proposition 3.3.1, we have

$$E[X_1] = \frac{1}{1-p_1} + \frac{1}{1-p_2} - \frac{1}{1-p_1p_2}.$$

Replace $E[X_1]$ and $E[X_2]$ in Equation (3.15) and divide n by N, we arrive at Proposition 3.3.2.

■

We can generalize the result to M receivers.

Theorem 3.3.2. The transmission bandwidth of scheme C with M receivers and sufficiently large N is

$$\eta_C \sim 1 + p_1\varphi_M + \sum_{j=1}^{M-1} (p_{j+1} - p_j)\varphi_{M-j}, \tag{3.16}$$

where

$$\varphi_M = \sum_{i_1,i_2,...,i_M} \frac{(-1)^{i_1+i_2+...+i_M-1}}{1-p_1^{i_1}p_2^{i_2}...p_M^{i_M}} \tag{3.17}$$

and $i_1, i_2, ...i_M \in \{0,1\}$, $\exists i_j \neq 0$, $p_1 \leq p_1 \leq ... \leq p_M$.

Proof. Using the same argument for the proof of Proposition 3.3.2, we have $Np_1 \leq Np_2 \leq ... \leq Np_M$. Now, we can conceptually count the number of combinations for XORing the lost packets and transmit these packets in different rounds. In particular, in round 1, there are Np_1 lost packets of R_1 that can be combined with the lost packets of $R_2, R_3, ..., R_M$. After these combinations, the numbers of lost packets remain for R_1, R_2, R_3, ..., R_M are 0, $N(p_2 - p_1)$, $N(p_3 - p_1)$, ..., $N(p_M - p_1)$, respectively. Next in round 2, the remaining $N(p_2 - p_1)$ lost packets at R_2 are combined with the remaining lost packets at R_3, R_4, ... R_M. Thus, the remaining lost packets for receivers R_1 to R_M are now 0, 0, $N(p_3 - p_2)$,..., $N(p_M - p_{M-1})$. The same reasoning applies until there are no more lost packets. Therefore, the average number of transmissions required to successfully deliver all N packets to all the receivers equals

$$n \sim N + Np_1\phi_1 + N(p_2 - p_1)\phi_2 + ... + N(p_M - p_{M-1})\phi_M, \tag{3.18}$$

where ϕ_i denotes the average number of transmissions required to successfully transmit a combined packet in round i.

Now, using Theorem 3.3.1, the average number of transmission attempts in order for all K receivers to correctly receive a packet is

$$\varphi_K = \sum_{i_1, i_2, ... i_K} \frac{(-1)^{i_1+i_2+...+i_K-1}}{1 - p_1^{i_1} p_2^{i_2} ... p_K^{i_K}}, \tag{3.19}$$

where $i_1, i_2, ..., i_K \in \{0, 1\}$, $\exists i_j \neq 0$, and $p_1 \leq p_2 \leq ... \leq p_K$. Set $\phi_i = \varphi_{M+1-i}$ and divide n by N, the proof follows directly.

■

Theorem 3.3.3. The transmission bandwidth of scheme D with M receivers and sufficiently large N is

$$\eta_D \sim \frac{1}{1 - \max_{i \in \{1,..,M\}}\{p_i\}}. \tag{3.20}$$

Proof. We begin with the case of two receivers. Without loss of generality, we assume that $p_1 \leq p_2$. As discussed in Section 3.2, the combined packets in scheme D are dynamically formed based on the feedback from the receivers. If a combined packet is correctly received at one receiver, but not at the other, a new combined packet is generated to ensure that the receivers with the correct packet will be able to obtain the new data using the new combined packet. As proved in the Proposition 3.3.2, in the long run, the number of losses will be dominated by the number of losses at the receiver with the largest error probability (R_2). Therefore, the total number of transmissions to successfully deliver N packets to two receivers equals the number of transmissions to successfully deliver N packets to R_2 alone, i.e. $\frac{N}{1-p_2}$ or $\frac{N}{1-max\{p_1,p_2\}}$. Using a similar argument, we can generalize this result to the case with M receivers:

$$n \sim \frac{N}{1-\max_{i\in\{1,..,M\}}\{p_i\}}. \tag{3.21}$$

Therefore, the transmission bandwidth is

$$\eta_D = \frac{n}{N} \sim \frac{1}{1-\max_{i\in\{1,..,M\}}\{p_i\}}. \tag{3.22}$$

■

For many real-time applications, it is necessary to reduce to the packet delay. This implies that the retransmission of lost packets from the sender to the receivers must be done promptly, i.e., N should be sufficiently small. However, by doing so, the chance of combining the lost packets decreases. Thus, we want to quantify the bandwidth efficiency for scheme D with finite buffer size N. We have the following theorem:

Theorem 3.3.4. The transmission bandwidth of scheme D with M receivers and buffer size N is

$$\eta_D^N = \frac{\sum\limits_{k=N}^{\infty} k\Big(\prod\limits_{j=1}^{M}\sum\limits_{i=0}^{k-N} Q_{ij} - \prod\limits_{j=1}^{M}\sum\limits_{i=0}^{k-N-1} Q_{ij}\Big)}{N}, \tag{3.23}$$

where

$$Q_{ij} = \begin{cases} p_j^i(1-p_j)^N \sum_{l=1}^{i} \binom{N}{l}\binom{i-1}{l-1} & \text{with} i \le N \\ p_j^i(1-p_j)^N \sum_{l=1}^{N} \binom{N}{l}\binom{i-1}{l-1} & \text{with} i > N \end{cases}$$

and p_j is the loss probability at receiver R_j.

Proof. We consider the scenario with one sender and one receiver. Let X denote the number of transmissions for the receiver to get N packets successfully. X can be N, $N+1$, $N+2$, ...We compute the probabilities for different values of X. If exactly N transmissions are required, then there must be no loss during transmitting N packets. We have

$$P[X = N] = \binom{N}{0} p^0(1-p)^N.$$

If $N+1$ transmissions are required, then there must be only one lost packet and one successful retransmission. We have

$$P[X = N+1] = \left[\binom{N}{1}p(1-p)^{N-1}\right](1-p) = \binom{N}{1}p(1-p)^{N}.$$

If $N+2$ transmissions are required, then there could be one loss during the transmissions of the first N packets and two retransmissions before the lost packet is successfully received, or there could be two losses during the transmissions of the first N packets and two successful retransmissions, one for each lost packet. We have

$$\begin{aligned} P[X = N+2] &= \left[\binom{N}{1}p(1-p)^{N-1}\right]p(1-p) + \left[\binom{N}{2}p^2(1-p)^{N-2}\right](1-p)^2 \\ &= \left[\binom{N}{1} + \binom{N}{2}\right]p^2(1-p)^N. \end{aligned} \tag{3.24}$$

Similarly, one can show that

$$P[X = N+i] = \begin{cases} p^i(1-p)^N \sum_{l=1}^{i} \binom{N}{l}\binom{i-1}{l-1} & \text{with} i \le N \\ p^i(1-p)^N \sum_{l=1}^{N} \binom{N}{l}\binom{i-1}{l-1} & \text{with} i > N. \end{cases} \tag{3.25}$$

Now, we consider the case of one sender and two receivers.

The number of transmissions using NC to guarantee that both receivers receive N packets successfully is $Y = \max_{j\in\{1,2\}}\{X_j\}$ where X_j is a random variable denoting the number of transmissions for receiver j to get N packets successfully.

Then,

$$P[Y \le k] = \prod_{j=1}^{2} \sum_{i=0}^{k-N} P[X_j = i + N]. \tag{3.26}$$

Next,

$$P[Y = k] = \prod_{j=1}^{2} \sum_{i=0}^{k-N} Q_{ij} - \prod_{j=1}^{2} \sum_{i=0}^{k-N-1} Q_{ij}, \tag{3.27}$$

where $Q_{ij} = P[X_j = N + i]$, which can be computed from (3.30).

The average number of transmissions so that both receivers receive a packet successfully is

$$\eta_D^N = E[Y] = \sum_{k=N}^{\infty} kP\left[Y = k\right] = \sum_{k=N}^{\infty} k \left(\prod_{j=1}^{2} \sum_{i=0}^{k-N} Q_{ij} - \prod_{j=1}^{2} \sum_{i=0}^{k-N-1} Q_{ij} \right). \tag{3.28}$$

Without much difficulty, we can generalize the result for the case with M receivers as

$$E[Y] = \sum_{k=N}^{\infty} k \left(\prod_{j=1}^{M} \sum_{i=0}^{k-N} Q_{ij} - \prod_{j=1}^{M} \sum_{i=0}^{k-N-1} Q_{ij} \right), \tag{3.29}$$

where

$$Q_{ij} = \begin{cases} p_j^i (1-p_j)^N \sum_{l=1}^{i} \binom{N}{l} \binom{i-1}{l-1} & \text{with} i \le N \\ p_j^i (1-p_j)^N \sum_{l=1}^{N} \binom{N}{l} \binom{i-1}{l-1} & \text{with} i > N \end{cases} \tag{3.30}$$

and p_j is the probability that the packet is lost at receiver R_j.

■

Note that we have not been able to obtain a reasonable closed-form expression for the transmission bandwidth of scheme C with a finite buffer. Thus, we omit the analysis for this case.

3.3.3 Receivers with Correlated Loss

The previous results are obtained based on a simple Bernoulli model for packet loss in a wireless medium. In many scenarios, packet losses at different receivers are highly correlated. For example, if two wireless receivers are located closely to each other, and behind an obstacle, then most likely they will have correlated losses. Thus, the assumption on independent packet losses among the receivers is no longer accurate. In this section, we would like to investigate the performance gain of NC schemes under such scenarios. In particular, we first assume that packet losses at different receivers in a given time slot can be correlated, and their loss probabilities are given by a joint probability. Second, we assume that packet losses in different time slots are uncorrelated. We now have the following results on transmission bandwidth for M receivers with correlated losses.

Theorem 3.3.5. The transmission bandwidth for an M-receiver scenario with correlated losses and sufficiently large buffer using scheme B is

$$\eta_{cor}^{B} = \sum_{k=1}^{\infty} kP\left[Y_M = k\right] \tag{3.31}$$

and using scheme C is

$$\eta_{cor}^{C} = 1 + \sum_{l=0}^{M-1} \sum_{k=1}^{\infty} k(p_{l+1} - p_l) P\left[Y_{M-l} = k\right], \tag{3.32}$$

where $p_1 \leq p_2 \leq ... \leq p_M$, $p_0 = 0$ and $P[Y_{M-l} = k]$ is the probability that the sender needs k transmissions to deliver a packet to all $M-l$ receivers $R_l, R_{l+1}, ..., R_M$ successfully.

Proof. Scheme B:

We start with the case of two receivers, then generalize to M receivers. Denote X_1 and X_2 the number of attempts needed to successfully deliver a packet to receiver R_1 and R_2 respectively, and $Y = \max_{i\in\{1,2\}}\{X_i\}$. We have

$$P[Y \leq k] = \sum_{i=1}^{k}\sum_{j=1}^{k} P[X_1 = i, X_2 = j]. \tag{3.33}$$

Next,

$$\begin{aligned} P[Y = k] &= \sum_{i=1}^{k}\sum_{j=1}^{k} P[X_1 = i, X_2 = j] - \sum_{i=1}^{k-1}\sum_{j=1}^{k-1} P[X_1 = i, X_2 = j] \\ &= \sum_{i=1}^{k} P[X_1 = i, X_2 = k] + \sum_{i=1}^{k-1} P[X_1 = k, X_2 = i]. \end{aligned} \tag{3.34}$$

Fig. 3.3 shows that after $X_1 = i$ and $X_2 = k$ attempts ($i \leq k$), a packet is received successfully at R_1 and R_2 respectively. Assume that we know the joint packet loss probability. Let us denote p_{xx}, p_{xo}, p_{ox}, and p_{oo} as the probabilities that a packet is lost at both receivers, a packet is lost at R_1 but received at R_2, a

Number of attempts	1	2	…	i-1	i	i+1	…	k-1	k
R_1	x	x	…	x	o	-	…	-	-
R_2	x	x	…	x	x	x	…	x	o

Figure 3.3: *Number of attempts to deliver a packet to both receivers successfully: R_1 needs i while R_2 needs k attempts ("x" and "o" indicate an unsuccessful and successful attempt, respectively).*

packet is received at R_1 but lost at R_2, and a packet is received at both receivers, respectively. We have

$$P[X_1 = i, X_2 = j] = \begin{cases} p_{xx}^{i-1} p_{ox} (p_{ox} + p_{xx})^{j-i-1} (p_{oo} + p_{xo}) & \text{with} i \leq j \\ p_{xx}^{j-1} p_{xo} (p_{xo} + p_{xx})^{i-j-1} (p_{oo} + p_{ox}) & \text{with} i > j. \end{cases} \tag{3.35}$$

Therefore, the average number of transmissions required to send a packet successfully to both receivers is

$$\begin{aligned} E[Y] &= \sum_{k=1}^{\infty} \sum_{i=1}^{k} kP[X_1 = i, X_2 = k] + \sum_{k=1}^{\infty} \sum_{i=1}^{k-1} kP[X_1 = k, X_2 = i] \\ &= \sum_{k=1}^{\infty} \sum_{i=1}^{k} k p_{xx}^{i-1} p_{ox} (p_{ox} + p_{xx})^{k-i-1} (p_{oo} + p_{xo}) \\ &+ \sum_{k=1}^{\infty} \sum_{i=1}^{k-1} k p_{xx}^{i-1} p_{xo} (p_{xo} + p_{xx})^{k-i-1} (p_{oo} + p_{ox}). \end{aligned} \tag{3.36}$$

Now examine the case of M receivers. Let X_i denote the number of attempts for a packet to be received successfully at receiver R_i, $Y_M = \max_{i \in \{1,\ldots,M\}} \{X_i\}$, and $P[j_1, j_2, ..., j_M] = P[X_1 = j_1, X_2 = j_2, ..., X_M = j_M]$. We have

$$P[Y_M = k] = \sum_{j_1,...,j_M \in Z_k} P[j_1, j_2, ..., j_M] - \sum_{j_1,...,j_M \in Z_{k-1}} P[j_1, j_2, ..., j_M], \tag{3.37}$$

where $Z_k = \{1, ..., k\}$ and $Z_{k-1} = \{1, ..., k-1\}$. Note that Z_k and Z_{k-1} are defined to make k largest among $j_1, j_2, ...j_M$.

Now, we can compute $P[j_1, j_2, ...j_M]$ in terms of the joint packet loss probabilities. Let the vector $(a_1a_2...a_M)$ denote the status reception of a packet at all the receivers; $a_h =$ "o" and $a_h =$ "x'' indicate successful and unsuccessful receptions at receiver R_i, respectively. Let $p_{a_1a_2...a_M}$ denote the probability of this event, then:

$$\begin{aligned}\sum_{j_1,...,j_M \in Z_k} & P[j_1, j_2, ..., j_M] \\ &= \sum_{j_1,...,j_M \in Z_k} \left(p^{i_1}_{x_1x_2...x_M} p_{o_1x_1...x_M} \prod_{h=1}^{M-1} \left(\sum_{a_h \in \{o_h, x_h\}} p_{a_1...a_hx_{h+1}...x_M} \right)^{l_{h+1}-l_h-1} \right. \\ & \left. \left(\sum_{a_h \in \{o_h, x_h\}} p_{a_1...a_ho_{h+1}x_{h+2}...x_M} \right) \right)\end{aligned} \tag{3.38}$$

where the sequence $\{l_1, l_2, ..., l_M\}$ is is an ascending sorted sequence of $\{j_1, j_2, ..., j_M\}$. For instance, if $\{j_1, j_3, j_2\}$ is the ascending sorted sequence of $\{j_1, j_2, j_3\}$, then then $l_1 = j_1$, $l_2 = j_3$, and $l_3 = j_2$.

The key is to obtain the above equation is to set up a table as shown in Fig. 3.3 and multiply out the joint probability as it was done in Equation (3.35) for the

2-receiver case.

Given $P[Y_M = k]$, the transmission bandwidth for M receivers with correlated losses using scheme B is

$$\eta_{cor}^{B} = E[Y] = \sum_{k=1}^{\infty} kP\left[Y_M = k\right] \tag{3.39}$$

Scheme C:

Consider the 2-receiver scenarios. We use the same notations above and assume that $p_1 \leq p_2$. In the long run, the average number of lost packets at receiver 2 with be larger than that at receiver 1. Then average number of transmissions to successfully deliver a packet to two receivers is

$$\eta_{cor}^{C} = 1 + p_1 E[max\{X_1, X_2\}] + (p_2 - p_1)E[X_2], \tag{3.40}$$

where $E[max\{X_1, X_2\}]$ is obtained from equation 3.36 and $E[X_2] = \frac{1}{1-p_2}$.

For the general case of M receivers, we also assume that $p_i \leq p_j$ for all $i < j$. Similarly, in the long run, the number of lost packets at receiver i is smaller than that of receiver j. Using the same argument for the case of 2 receivers we can derive the average number of transmissions required to successfully deliver a packet to M receivers as

$$\begin{aligned}\eta_{cor}^{C} &= 1 + p_1 E[\max_{i\in\{1,..,M\}}\{X_i\}] + (p_2 - p_1)E[\max_{i\in\{2,..,M\}}\{X_i\}] + ... + (p_M - p_{M-1})E[X_M] \\ &= 1 + \sum_{l=0}^{M-1}(p_{l+1} - p_l)\sum_{k=1}^{\infty} kP\left[Y_{M-l} = k\right]\end{aligned}$$

$$= \quad 1 + \sum_{l=0}^{M-1} \sum_{k=1}^{\infty} k(p_{l+1} - p_l) P\left[Y_{M-l} = k\right], \tag{3.41}$$

where $p_0 = 0$ and $P[Y_{M-l} = k]$ is the probability that the sender needs k transmissions to deliver a packet to all $M - l$ receivers $R_l, R_{l+1}, ...,$ and R_M successfully. $P[Y_{M-l} = k]$ can be computed using Equation (3.38). Note that, to shorten the notation, we add a virtual receiver R_0 with $p_0 = 0$. ■

The theorem above indicate that to compute the transmission bandwidth, one needs to compute the probabilities $P[Y_M = k]$ and $P[Y_{M-l} = k]$. We show how to compute these probabilities in the Appendix.

The transmission bandwidth with correlated losses in Scheme D is the same as that in the case of independent receivers, i.e.,

$$\eta^D_{cor} = \frac{1}{1 - \max_{i \in \{1,..,M\}}\{p_i\}}.$$

This is because, in the long run, regardless of whether the packet losses are correlated or not, the number of transmissions to successfully deliver N packets to M receivers will be dominated by the one with the largest loss probability.

3.3.4 Remarks on NC Gain

In the previous section, we analyzed the transmission bandwidths of different schemes. We now define the coding gain of one scheme over the other by the ratio of their transmission bandwidths. In particular, the coding gains of schemes

C and D over scheme B for two receivers are

$$G_C = \frac{\eta_B}{\eta_C} = \frac{\frac{1}{1-p_1} + \frac{1}{1-p_2} - \frac{1}{1-p_1 p_2}}{1 + \frac{p_1}{1-p_1} + \frac{p_2}{1-p_2} - \frac{p_1}{1-p_1 p_2}} \tag{3.42}$$

and

$$G_D = \frac{\eta_B}{\eta_D} = \frac{\frac{1}{1-p_1} + \frac{1}{1-p_2} - \frac{1}{1-p_1 p_2}}{\frac{1}{1-max\{p_1,p_2\}}}. \tag{3.43}$$

For the case $p_1 = p_2 = p$, Equations (3.42) and (3.43) become

$$G_C = \frac{1+2p}{1+p+p^2} \tag{3.44}$$

and

$$G_D = \frac{1+2p}{1+p}. \tag{3.45}$$

Note that, when p_1 or p_2 is equal to zero, Equations (3.42) and (3.43) indicate no gain for NC schemes, e.g., $G_C = G_D = 1$. However, this scenario is only true when considering only two receivers. A typical scenario is likely to involve more users with different packet loss rates. Even in the presence of lossless receivers, if there are a few lossy receivers (more than 1), our schemes still provide higher bandwidth efficiency. We will continue this discussion in the next section.

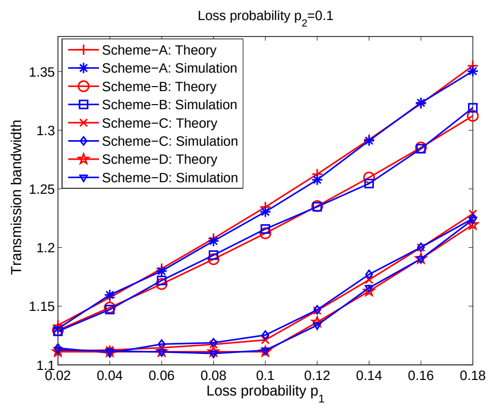
3.4 Simulation Results and Discussion

We use simulations to (a) verify the analytical derivations for the transmission bandwidths and (b) to set light on the typical performances of different broadcast schemes for real world settings. Instead of using Raleigh fading parameters to characterize the wireless channel, we use packet loss rates which is sufficiently characterize the overall health of the channel. For interested readers, in [8], we provide some analysis of our proposed techniques with a detail consideration on the modulation and channel parameters. However, such analysis is beyond the scope of this thesis. Also, our simulations do not take into account the interaction between the MAC protocol and the higher layer protocol such as TCP. Under some settings, this interaction may reduce the coding gain for the proposed schemes. Recently, Dong et al. [24] provide a discussion on possible performance degradation of NC when TCP is employed in wireless ad hoc network. As such, the authors provide a *loop coding* scheme that improves both network throughput and TCP throughput simultaneously. Modeling such a complex interaction is very useful, however it is beyond the scope of this thesis.

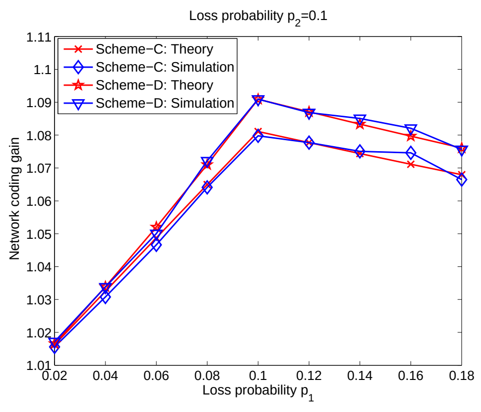
That said, the simulations are divided into four categories. In category one, packet losses are assumed to be independent and uncorrelated across the receivers. In category two, packet losses are also assumed independent across the time slots, but they are correlated among the receivers. In both of these categories, we attempt to model a realistic performance of the proposed scheme by using the simulated packet loss rates for the IEEE 802.11 standard as reported in the literature. In particular, the reported packet loss rates (before the MAC protocol retransmissions) range from 0.5% to 20% [25]. Similar packet loss rates are confirmed in [26]. In category three, we model the channel as a two-state Markov chain to capture the burst losses. Finally, in category four, we employ trace-driven simulations. The traces are collected from measurements of packet losses in IEEE 802.11 based network [25]. We now begin with the simulations for category one.

3.4.1 Independent and Uncorrelated Packet Loss Model

Fig. 3.4 (a) shows the simulation and theoretical results on the transmission bandwidths (e.g., the average number of transmissions per successful packet) of schemes A, B, C, and D for the scenario consisting of one sender and two receivers R_1 and R_2 with independent and uncorrelated packet losses. For schemes C and D, we use the buffer size $N = 1000$ packets, which sufficiently simulates an infinite size buffer in this setting. The packet loss probability of R_1 varies as shown on the x-axis while that of R_2 remains at 10%. As seen, the simulation and theoretical curves match all most exactly for all the schemes, verifying the results of our derivations. Furthermore, the number of transmissions per successful packet in scheme D is the smallest while that of scheme A is the largest, which confirms our earlier intuitions about these schemes. We note that although scheme D is slightly more efficient than scheme C, the hardware implementation of scheme D might be little more complex than that of scheme C due to its dynamic selection of the retransmitted packets.



(a)



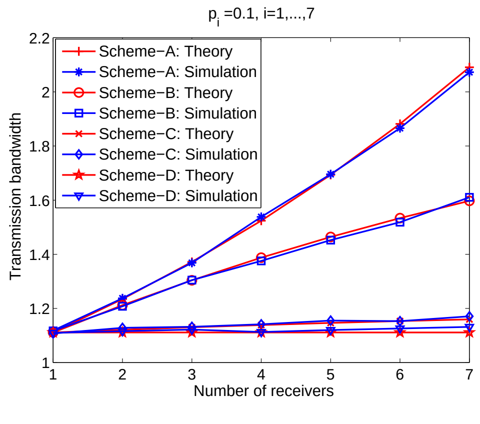
(b)

Figure 3.4: (a) Transmission bandwidth versus packet loss probability; (b) NC gain versus packet loss probability.

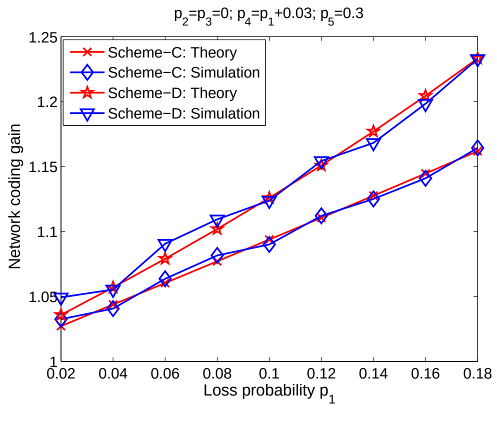
We now show the coding gains of different schemes. Ideally, scheme A has the worst performance and should be used as the baseline for comparison. However, most wireless devices with limited memory will be able to implement scheme B. Furthermore, scheme B is the traditional ARQ scheme. Therefore, we compare our proposed schemes C and D against scheme B by examining their coding gains over scheme B. Fig. 3.4(b) shows the coding gains of schemes C and D as functions of packet loss probability of R_1. It is interesting to note that the gain is largest when both loss probabilities of R_1 and R_2 are equal to each other. This is intuitively plausible as in this special case, the maximum number of lost packet pairs is achieved. In other words, more combined packets can be generated, reducing the number of retransmissions required otherwise.

On the other hand, when two receivers have disparate packet loss rates, e.g., $p_1 = 0.01$, $p_2 = 0.9$, using the NC techniques, roughly 1% of the combined packets and 89% of individual lost packets must be retransmitted. Since NC techniques depend on the number of lost packets that can be combined, it would not produce much coding gain in this scenario. At one extremity, if one receiver does not have any packet loss and the other has some non-zero packet loss rates, e.g., 10%, then the performance of the NC technique is identical to that of scheme B, the traditional ARQ technique, i.e., coding gain over scheme B equals 1.

In addition, the coding gains for schemes C and D over scheme B seem to be the piece-wise linear functions of the loss rate p_1 (with fixed p_2). However, this is simply a coincidence for this range of values for p_1 and p_2. The coding gains of scheme D over B can be easily calculated as $(\frac{1}{1-p_1} + \frac{1}{1-p_2} - \frac{1}{1-p_1 p_2})(1-p_2)$ for the first segment, and $(\frac{1}{1-p_1} + \frac{1}{1-p_2} - \frac{1}{1-p_1 p_2})(1-p_1)$ for the second segment. These functions are clearly not linear functions, but their plots resemble linear plots. Also note that the cusp in the graph is due to the sudden change of $\max\{p_1, p_2\}$ from p_2 to p_1. Recall that for scheme D, the transmission bandwidth is $\frac{1}{1-max\{p_1,p_2\}}$. Since p_2 is fixed and p_1 changes along the x-axis. Therefore, the transmission bandwidth remains constant at $\frac{1}{1-p_2}$ until p_1 exceeds p_2. After this point, the transmission bandwidth starts varying with p_1, creating a cusp in the graphs. To investigate the effectiveness of our proposed techniques as functions of the number of the receivers, Fig. 3.5 (a) shows the average number of transmissions required to successfully deliver a packet to all receivers for schemes A, B, C and D. In this scenario, the loss probabilities of all the receivers are set to 0.1. The NC schemes C and D significantly outperform schemes A and B when the number of receivers is large. As the number of receivers increases, the transmission bandwidths for



(a)



(b)

Figure 3.5: (a) Transmission bandwidth versus the number of receivers; (b) NC gains versus packet loss probability in a 5-receiver scenario.

schemes A and B increase significantly. This is because it is much harder to successfully transmit a packet to all the receivers due to the increase in likelihood that a lost packet can occur at any receiver. For example, if the packet loss rate of receiver R_1 is p_1, then the average number of transmissions required to successfully transmit a packet is $\frac{1}{1-p_1}$. Now if one is required to successfully transmit the same packet also to receiver R_2 with packet loss rate of p_2, then using scheme A, one needs an average of $\frac{1}{(1-p_1)(1-p_2)}$ transmissions. Note that the denominator is the product of terms that are less than 1, which quickly reduces to a small number when the number of receivers increases. As a result, the transmission bandwidth quickly increases. On the other hand, the transmission bandwidth for scheme C increases very slightly and is unchanged for scheme D. As shown in the analysis, the transmission bandwidths of these NC schemes depend more or less on the receiver with the highest packet loss rate. Since the loss rates are set to 0.1 for all the receivers, we should not expect to see much increase in the transmission bandwidth. In fact, for scheme D, the transmission bandwidth should not increase at all since it is equal to $\frac{1}{1-max\{p_1,\ldots,p_7\}} = \frac{1}{1-0.1} = 1.1$, which is not a function of the number of receivers. Intuitively, even though there are more packet losses with more receivers, using NC techniques, most of these lost packets can be combined, effectively reducing the total number of retransmissions.

Fig. 3.5 (b) shows the theoretical and simulated coding gains (over scheme B) for five receivers with different loss probabilities for schemes C and D as a function of packet loss probability p_1 at receiver R_1. The packet loss probabilities at other receivers are set as follows: $p_2 = p_3 = 0$; $p_4 = p_1 + 0.3$; and $p_5 = 0.3$, i.e. there are packet losses at receivers R_1, R_4 and R_5, but not R_1 and R_2. As seen, the NC gain of scheme D is 15% when $p_1 = 0.1$. Note that, even if there are two receivers without packet loss, our NC schemes are still better than the traditional retransmission scheme. This is plausible since whenever there are pairs of disjoint packet losses at two or more receivers, the XOR packets are formed and transmitted in the NC schemes, leading to a better performance.

Up until now, we have shown the results of different schemes under the infinite buffer assumption. In practice, one must use a finite buffer. To characterize the performance of the best scheme (scheme D) with a finite buffer, Fig. 3.6 shows the transmission bandwidth as a function of the buffer size N for scheme D at $p_1 = p_2 = 0.2$. As expected, as the buffer size increases, the number of opportunities for combining lost packets increases, resulting in a smaller number of retransmissions. When the buffer size is large, e.g., more than 40 and $p_1 = p_2 = 0.2$,

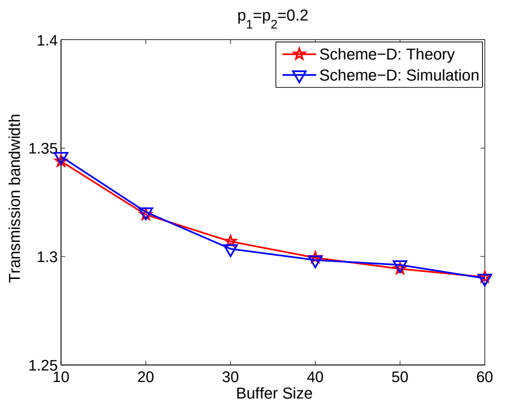


Figure 3.6: *Transmission bandwidth versus buffer size for scheme D in a 2-receiver scenario.*

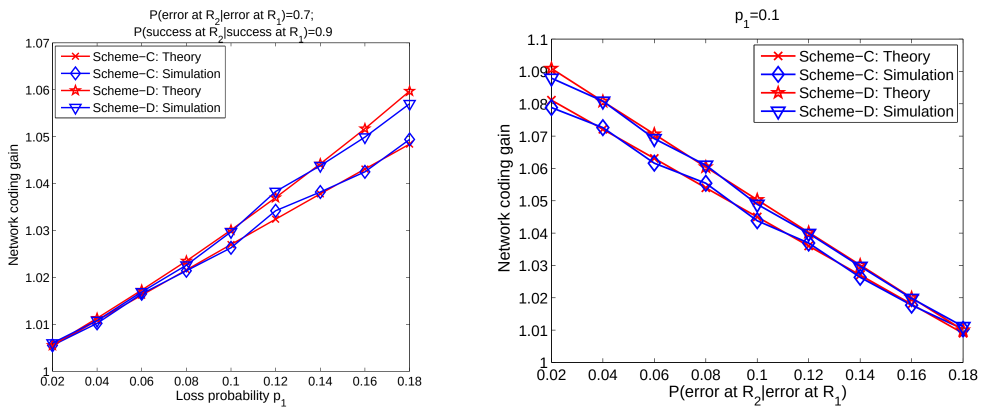
the transmission bandwidth changes very slightly. This is because such a buffer size is sufficiently large and can be thought of having an infinite value, and thus the transmission bandwidth remains constant according to Theorem 3.3.3. Under high loss rates, one can use a shorter buffer and still achieve the performance of the scheme with an infinite buffer. On the other hand, one needs to use a larger buffer when packet losses are infrequent to achieve the performance limit. Clearly, using a larger buffer size results in longer delay for certain packets, and may not be acceptable for some real-time applications. An interesting question is how to find an optimal buffer size under the delay constraints. We have addressed this question partially in [7].

3.4.2 Independent but Correlated Packet Loss Model

In this model, the receivers are assumed to have correlated packet losses. In particular, we assume there are two receivers with the following loss characteristics: whenever there is a packet loss at R_1, with probability of 0.7 that packet is also lost at R_2 and whenever a packet is received successfully at R_1, with probability of 0.9 that packet is also received successfully at R_2. Note that both conditional probabilities are above 0.5 which imply positive correlations between successful receptions as well as packet losses at these two receivers. We note that these conditional probabilities can be computed from the given joint probability mass function. Fig. 3.7 (a) shows the NC gains versus the packet loss probability for schemes C and D in the case of two receivers with correlated losses. As seen, as the packet loss probability of R_1 increases, the NC gain also increases. However, the NC gain for correlated loss receivers is not as large as that of receivers with independent losses. To see why, we consider two correlated loss receivers when $p_1 = 0.1$. This implies that

$$\begin{aligned} p_2 &= P(R_2 = \text{“}x\text{”}|R_1 = \text{“}o\text{”})P(R_1 = \text{“}o\text{”}) + P(R_2 = \text{“}x\text{”}|R_1 = \text{“}x\text{”})(R_1 = \text{“}x\text{”}) \\ &= (1 - P(R_2 = \text{“}o\text{”}|R_1 = \text{“}o\text{”}))(1 - p_1) + P(R_2 = \text{“}x\text{”}|R_1 = \text{“}x\text{”})p_1 \\ &= 0.16 \qquad (3.46) \end{aligned}$$

Now, the coding gain for the correlated loss receivers at this point in Fig. 3.7 is 1.03 while the coding gain for the independent loss receivers at the same point



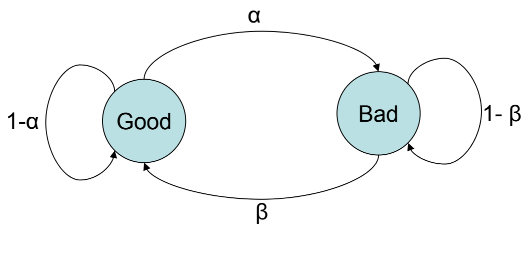
(a) (b)

Figure 3.7: (a) Coding gain versus packet loss probability in a scenario with correlated losses; (b) Coding gain versus conditional packet loss probability in a scenario with correlated losses.

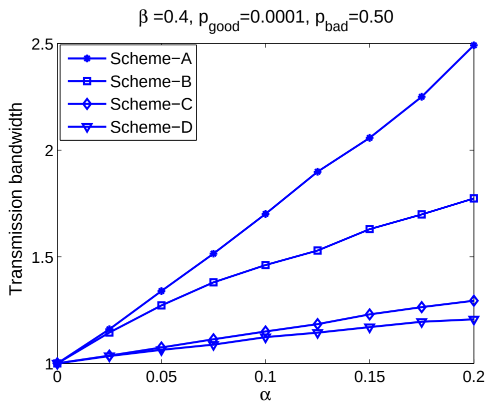
$(p_1 = 0.1, p_2 = 0.16)$ in Fig. 3.4 is 1.08. Clearly, NC techniques are less useful in correlated loss environments. This is intuitively plausible by considering one extremity where packet losses at the receivers are 100% correlated. In this case, the NC scheme simply reduces to the traditional retransmission scheme since the packet receptions at two receivers are completely identical. Fig. 3.7 (b) confirms this phenomenon as the correlation between the two receivers increases, the NC gain reduces.

3.4.3 Two-State Markov Model

We now present the simulation results based on a two-state Markov model for characterizing the bursty packet losses. Using the two-state Markov model, the state of a channel is classified into "bad" and "good" states. When the channel is in



(a)



(b)

Figure 3.8: (a) Transition of channel states in two-state Markov error model.; (b) Transmission bandwidth versus state transition probability using two-state Markov error model in a 5-receiver scenario.

the good state, the packet loss probability p_{good} is small, and when it is in the bad state, the packet loss probability p_{bad} is much larger. The channel state changes at each transmission slot with transition probabilities $\alpha = p_{good->bad}, \beta = p_{bad->good}$ as shown in Fig. 3.8(a). The stationary probabilities for the channel in the good and bad states are $\pi_{good} = \frac{\beta}{\beta+\alpha}$ and $\pi_{bad} = \frac{\alpha}{\beta+\alpha}$, respectively.

We evaluate the performances of different schemes for a 5-receiver scenario, with each receiver having identical channel conditions. For simplicity, we set β to a constant value while varying α. Fig. 3.8(b) shows the transmission bandwidths of different schemes. When $\alpha = 0$, the channel quickly converges and stays in the good state which has very small loss probability. Thus the performances of all schemes are almost identical. As α increases while β is unchanged, the portion of time that the channel has a high loss probability becomes larger. As a result, there are more lost packets, and more combined packets to be transmitted, leading to large performance gaps between NC schemes (C, D) and non-NC schemes (A,B). In other words, the transmission bandwidths for NC schemes do not increase as fast as those of the non-NC schemes as the channel gets progressively worse.

3.4.4 Trace-driven Simulation

We now show the performances our proposed schemes using trace-driven simulations. In particular, Fig. 3.9(a) shows the actual packet error rates of the IEEE 802.11 standard for different traces [25] prior to the MAC layer retransmission. Each trace consists of 20,000 packets; they recorded the average packet loss rates during the transmissions from an AP to a receiver. We note that the packet loss rates are substantially large for traces larger than 36. We ignore these traces since they are measured when the sending rate is set to a very high value. Instead, we use traces 0 to 36 to evaluate the performances of our schemes. Since these traces were collected in an experiment involving only a single sender and a single receiver, we need a way to assign traces to multiple receivers in order to simulate broadcast scenarios. That said, to simulate a scenario consisting of 5 receivers, we arbitrarily picked traces 1 to 12 to represent channel conditions for receivers 1 and 2, traces 20 to 31 for receiver 3, traces 15 to 26 for receiver 4, and traces 25 to 36 for receiver 5. Effectively, each receiver experiences 12 different packet loss rates through time, corresponding to 12 different traces with receivers 1 and 2 having identical channel conditions. Fig. 3.9(b) shows the transmission bandwidths for

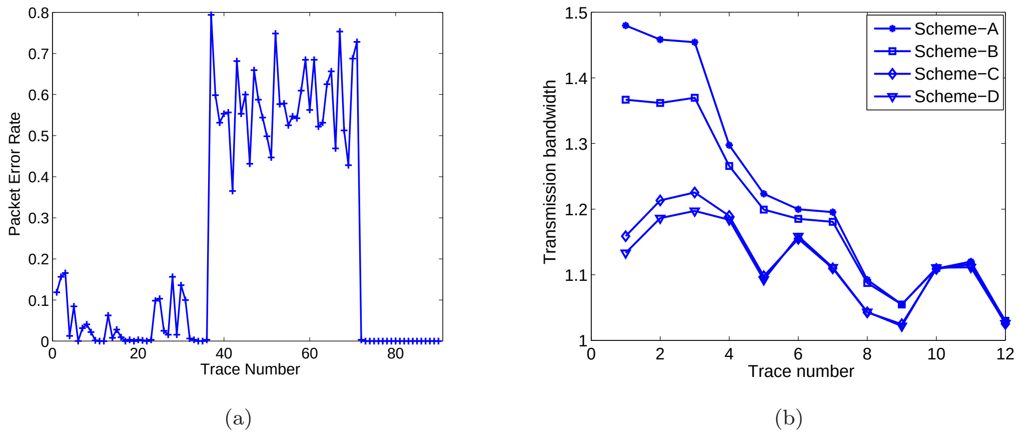


Figure 3.9: (a) Packet error rates for different traces [25].; (b) Transmission bandwidths for different traces in a 5-receiver scenario.

different schemes vs trace number. Each point on the graph represents the transmission bandwidth when the packet loss rate for each receiver is taken from its traces in the increasing order. For example, the first point on the graph for scheme A corresponds to the packet loss rates for receivers 1 to 5 taken from traces 1, 1, 20, 14, 25 respectively. As seen in Fig. 3.9(b), traces with high loss probabilities result in larger performance improvements for NC schemes over non-NC schemes, whereas there are almost no differences among the performances of all schemes for traces with near-zero error rates. This confirms our theoretical analysis.

Remark: We note that the performances of our proposed schemes depend on the packet loss rates, which are functions of many parameters. Among these parameters are the packet size and the bit error rate of the channel. The packet size is tunable. The bit error rate can be reduced to a desired rate by adding appropriate amount of FEC redundancy. Therefore, one can modify the packet error rate for optimal bandwidth transmission by changing the packet size and the amount of redundancy. Of course, when doing so, one must take into account the redundancy introduced by FEC. Even when an optimal FEC is used for a given channel, we emphasize that our proposed NC technique with FEC still outperforms a hybrid-ARQ technique that uses the same amount of FEC. This is because adding the same amount of FEC just simply changes the packet loss probabilities for both techniques by an equal amount. Furthermore, we have shown that NC technique for retransmission is better than the traditional ARQ technique. Therefore, one should expect that a joint NC and channel coding technique is better than a hybrid-ARQ technique. In [27], we provide some analysis for jointly optimizing NC and FEC techniques.

3.5 Related Work

Wireless broadcast is a well-explored problem. In his seminal work, Cover [28] modeled a broadcast channel as multiple binary channels, each with a given channel capacity. He found the lower and upper bounds on the capacity regions of jointly achievable transmission rates. Subsequently, there has been much research on using broadcast bandwidth efficiently. Recently, NC has been applied successfully to many wireless broadcast and multicast applications. Lun *et al.* [29] proved that the problem of minimum-energy multicast in infrastructureless networks can be solved exactly in polynomial time when employing NC. This is in contrast with the traditional routing approaches in [30–32] that result in non-polynomial time solutions. In addition, Li *et al.* [33, 34] proved that NC could provide some benefits over the non-NC approaches. Lun *et al.* [35] showed a capacity-approaching coding scheme for unicast or multicast over lossy packet networks, in which all nodes perform opportunistic coding by constructing the encoded packets with random linear combinations of previously received packets.

Our work is rooted in the recent development of NC for wireless ad hoc networks [4, 36–38]. In [4], Wu *et al.* proposed the basic scheme that uses XOR of packets to increase the bandwidth efficiency of a wireless mesh network. In [36], Katti *et al.* implemented a XOR-based scheme in a wireless mesh network and showed a substantial bandwidth improvement over the current approach. Unlike existing approaches, the focus of our work is on the analysis of the *reliable* wireless broadcast problem in a single-hop wireless network such as WLAN or WiMAX networks. Eryilmaz *et al.* [39] also recently proposed a similar model. In this work, Eryilmaz *et al.* employed a *random NC* scheme for multiple users downloading a single file or multiple files from a wireless base station. Rather than using XOR operations, their scheme encodes every packet using coefficients taken randomly from a sufficiently large finite field [40, 41]. This scheme guarantees that the receivers can decode the original data with high probability. Another work somewhat related to ours is that of Ghaderi *et al.*[42]. In [42], the authors analyzed the reliability benefit of NC for reliable multicast by computing the expected number of transmissions using link-by-link ARQ compared to NC.

Last but not least, there exist standard error control techniques for reliable wireless transmission that employ either FEC, ARQ, or hybrid-ARQ [43]. For example, A. Shiozaki *et al.* [44] and M. Nakamura *et al.* [45] investigated hybrid error control broadcast models which incorporate the ARQ with FEC techniques to achieve higher throughputs to all receivers.

3.6 Summary

In this chapter we propose some NC techniques to increase the bandwidth efficiency of reliable broadcast in a wireless network. Our proposed schemes combine different lost packets from different receivers in such a way that multiple receivers are able to recover their lost packets with one transmission by the sender. The advantages of the proposed schemes over the traditional wireless broadcast are shown through simulations and theoretical analysis. Specifically, we provide a few results on the transmission bandwidths of the proposed schemes under different channel conditions.

Chapter 4 – WIRELESS MEDIA TRANSMISSIONS WITH NC USING MDP

1) D. Nguyen, T. Nguyen, and Xue Yang, *Multimedia Wireless Transmission with Network Coding*, in Packet Video, Lausanne, Switzerland, 2007.

2) D. Nguyen and T. Nguyen, *Network Coding-Based Wireless Media Transmission Using POMDP,* in Packet Video, Seattle 2009.

3) D. Nguyen, T. Nguyen, and Xue Yang, *Joint Network Coding and Scheduling for Media Streaming over Multi-User Wireless Networks*, submitted to IEEE Transactions of Multimedia, under second revision.

4.1 Introduction

Although there has been a tremendous growth in multimedia applications over the Internet, packet loss, delay, and time-varying bandwidth of the Internet have hindered many high-quality multimedia applications. These problems manifest more so in wireless networks, which often exhibit higher loss rate and lower bandwidth. Arguably, all these problems will disappear if bandwidth is infinite or is well provisioned. Unfortunately, the current *best effort* Internet and wireless networks provide neither. Even if the Internet is redesigned entirely to provide proper bandwidth provision mechanisms, doing so may introduce scalability and complexity issues, resulting in degraded performance. Thus, many *above network layer* approaches to multimedia streaming over the Internet and wireless networks have been proposed to deal with packet loss, delay, and time-varying bandwidth, ranging from transport protocols and packet scheduling algorithms [46, 47] to source and channel coding techniques [48, 49]. A number of these techniques are based on the differentiated principle in which, data of various importance levels are treated differently under the resource constraints. For example, several scalable video coding techniques are employed to compress a video bit stream in a layered hierarchy consisting of a base layer and several enhancement layers. The base layer contributes the most to the visual quality of a video, while the enhancement layers provide successive quality refinements [50]. As such, using a scalable video bit stream, the sender is able to adapt a video bit rate to the current available network bandwidth by sending the base layer and an appropriate number of enhancement

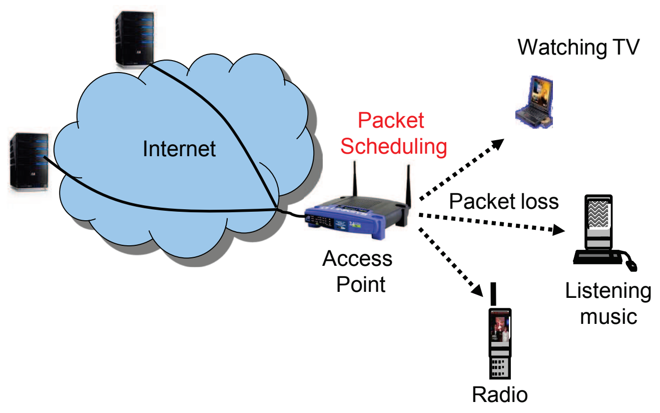


Figure 4.1: Multimedia transmissions from the Internet sources to multiple receivers over the Access Point.

layers [50, 51]. The receiver is then able to view the video with higher quality when more layers are received as a result of larger available bandwidth.

That said, for a number of video streaming applications, their bandwidth requirements are sufficiently small that even without employing sophisticated techniques, a few of these applications can run concurrently over the existing wireless standards (IEEE 802.11(b) and (g)). On the other hand, these standards may not be able to support multimedia applications with much larger bandwidth requirement, e.g., DVD quality video streaming applications. Despite the fact that wireless bandwidth has been increasing significantly, from a theoretical limit of 11Mbps for 802.11b to 54 Mbps for 802.11g, and to 540 Mbps for 802.11n, there has always been high bandwidth demand resulting from new applications. For example, many wireless devices and multimedia applications have been developed, ranging from MP3 streaming on wireless-ready iPods to video conferencing via laptops. In the near future, IPTV and Video on Demand (VoD) applications will rely on wireless network to deliver high quality videos from the Internet to any TV set or computers at home through a wireless Access Point or Base Station. In addition to home networks, WiMAX (Worldwide Interoperability for Microwave Access) has emerged as the wireless standard that aims to deliver wireless data over long distances, and can potentially provide wireless broadband access as an alternative to cable and DSL. With the wireless broadband access, there will be potentially more users (homes), leading to higher bandwidth demand and greater bandwidth constraint. Therefore, it is imperative that an efficient bandwidth sharing/competing scheme among the wireless applications be employed to satisfy bandwidth and delay requirements of each application.

This chapter proposes a new NC technique to improve the overall bandwidth efficiency while optimizing multiple concurrent multimedia applications with heterogeneous requirements in a wireless access network. In particular, we propose a NC based scheduling policy to be used at a WLAN-like AP or at a WiMAX-like broadcast station that optimizes the multimedia transmission in both broadcast and unicast settings 4.1. The contributions of this chapter include (a) a framework for increasing bandwidth efficiency of broadcast and unicast sessions in a wireless network based on NC techniques and (b) optimized scheduling algorithms based on the Markov Decision Process (MDP) to maximize the quality of multimedia applications. In Section 4.2, we present a basic NC based retransmission scheme that improves the bandwidth efficiency of broadcast and unicast sessions in a one-hop wireless network. Next, we present NC based scheduling policy using MDP that optimizes multiple concurrent flows under bandwidth constraint. In Section

4.5, we demonstrate our simulation-based dynamic algorithm as a solution for a large MDP. Section 4.6 shows how our simulation-based algorithm is used to solve the scheduling problem for the case of erroneous feedback. Simulation results and discussions are provided in Section 4.7. Finally, we provide a summary with a few remarks in Section 4.8.

4.2 Background on Wireless Streaming with NC

4.2.1 Multimedia Streaming

Multimedia streaming over best-effort, packet-switched networks is challenging due to a number of factors such as high bit rates, delay, and loss sensitivity. As such, many approaches have been proposed ranging from network protocols to source and channel coding techniques. From channel coding perspective, Forward Error Correction (FEC) techniques have been proposed to increase reliability at the expense of bandwidth expansion [48, 52–54]. From source coding perspective, error-resilient coding techniques have been explored to allow the quality of a video to be degraded gracefully in lossy environments [51, 55, 56]. In addition, layered video coding techniques have been proposed to deal with heterogeneity and time-varying nature of the Internet by adapting its bit rate to the available bandwidth [50, 57, 58]. A layered video bit stream is organized into a number of layers with the base layer being the most important layer in terms of the visual quality contribution. Other layers are used to enhance the video quality with more layers resulting in higher video quality.

Depending on the coding techniques, a video bit stream is composed of different types of bits. Each type generally contributes a different amount of enhancement toward the final reconstructed video quality. Furthermore, some bits are only useful when other bits are present. This property leads to much research on the packet scheduling algorithms that choose which packets to send at which times, in order to produce in the best possible reconstructed video quality under insufficient network resources. Notably, the rate-distortion, MDP-based optimization approach to packet scheduling has produced many fruitful results in the past several years [46, 59, 60]. The main idea of this approach is that using the observations at every single step, the scheduling algorithm chooses the best action to perform (e.g., whether to send a packet or not and which packet to send) in order to maximize the *expected* video quality under limited network resources. The optimal sequence of actions during the duration of interest is the solution to the MDP problem which can be efficiently solved in many settings.

4.3 Optimization with Markov Decision Processes

Let us consider a decision maker or a controller who, at every time step, is in charge of making a decision or choosing an action, which can influence the evolution of a probabilistic system. Assuming that the state of the system evolves in discrete time steps, then the goal of the controller is to choose a sequence of actions that maximizes some cumulative system performance metrics (rewards) at the end of some finite or infinite number of time steps. Since the system states and the performance metrics depend on the chosen action at every time step, it is wise for the controller to consider the future states and the associated rewards in the decision making process at the present state. Finding the optimal sequence of actions is the solution to the MDP problem.

That said, an abstract MDP represents a dynamic system and is specified by a finite set of states S, representing the possible states of the system, a set of control actions A, a transition probability P, and a reward function r. The transition probability specifies the dynamics of the system, and gives the probability $P(s'|s,a)$ of transitioning to state s' after taking action a in state s. The dynamics are Markovian in the sense that the probability of the next state s' depends only on the current state s and action a, and not on any previous history. The reward function assigns a real number to the current state s and the action a taken in that state, so that $r(s,a)$ represents the immediate reward of being in state s and taking action a. A policy π is a mapping from states to actions, which defines a controller that takes actions as specified by the policy. We assume that time is discrete and that the control policy selects one action at each time step. Every policy π is associated with a value function V^{π} such that $V^{\pi}(s)$ gives the expected cumulative reward achieved by π when starting in state s. The solution to a MDP problem is an optimal policy that maximizes the expected cumulative reward over any finite or infinite number of time steps.

When a MDP ends in a finite number of time steps N, we call it a finite-horizon MDP. Mathematically, let us denote S as a set of all the finite states and A a set of all possible actions. Let d_t denote a decision rule, prescribing a procedure for action selection in each state at a specified time step t. In other words, decision rules are functions $d_t : S \rightarrow A$, which specify the choice of action when the system occupies state s at time step t. For each $s \in S$, $d_t(s) = a_t \in A$. A policy $\pi = (d_1, d_2, d_3, ...d_N)$ is a sequence of actions at every time step. Thus, a sample path of the system evolution is a sequence of states and actions pairs $\omega = (s_1, a_1, s_2, ..., a_{N-1}, s_N)$ and the probability of this sample path under policy π is:

$$\begin{aligned} P^{\pi}(\omega) &= P(s_1, a_1, ..., a_{N-1}, s_N) \\ &= P_1(s_1)p_1(s_2|s_1, a_1)...p(s_N|s_{N-1}, a_{N-1}), \end{aligned} \tag{4.1}$$

where $P_1(s_1)$ and $p_t(s_{t+1}|s_t, a_t)$ specify respectively the initial distribution of the states and the transition probability from state s_t to s_{t+1} when action a_t is taken. Let $V(\omega)$ denote a real-valued function of ω, then the expected value of V under policy π is:

$$E^{\pi}\{V\} = \sum_{\omega \in (S \times A)^N} V(\omega)P^{\pi}(\omega) \tag{4.2}$$

$$= \sum_{v \in R^1} vP^{\pi}(\omega : V(\omega) = v). \tag{4.3}$$

Now, let $r_t(s, a)$ denote an immediate reward of taking action in a in state s at time t, then the total reward corresponding a sample path $\omega = (s_1, a_1, s_2, a_2, ...a_{N-1}, s_N)$

is:

$$V(s_1, a_1, ..., s_N) = \sum_{t=1}^{N-1} r_t(s_t, a_t) + r_N(s_N), \tag{4.4}$$

where $r_N(s_N)$ is the optional reward in the final state. Even though there is no action taken in the final state, one can assign an immediate reward. Therefore, given a distribution of the states $P(s)$ and policy π, the expected total reward of a MDP is:

$$\begin{aligned} \overline{V}^{\pi} &= E^{\pi}\left\{V(\omega)\right\} \\ &= E^{\pi}\left\{\sum_{t=1}^{N-1} r_t(s_t, a_t) + r_N(s_N)\right\}, \end{aligned} \tag{4.5}$$

In a typical scenario, it is useful to find the expected total reward given a starting state $s_1 = s$. We may denote this expected reward as:

$$\overline{V}_s^{\pi} = E_s^{\pi}\left\{\sum_{t=1}^{N-1} r_t(s_t, a_t) + r_N(s_N)\right\}, \tag{4.6}$$

where the expectation is taken over a new distribution of the sample paths $P^{\pi}(\omega)$ as:

$$\begin{aligned} P^{\pi}(\omega) &= P(s, a_1, ..., a_{N-1}, s_N) \\ &= p_1(s_2|s, a_1)...p(s_N|s_{N-1}, a_{N-1}), \end{aligned} \tag{4.7}$$

as a result of setting $P(s_1 = s) = 1$ in (4.1).

Typically, $\overline{V}_s^{\pi}$ can be efficiently computed using dynamic programming by denoting U_t^π as the total expected reward obtained by using policy π from the time t, $t+1$, ... $N-1$. Thus, for $t < N$, we have

$$U_t^\pi(s_t) = E_{s_t}^\pi \left\{ \sum_{n=t}^{N-1} r_n(s_n, a_n) + r_N(s_N) \right\}. \tag{4.8}$$

Clearly,

$$\overline{V}_s^\pi = U_1^\pi(s) \tag{4.9}$$

Now, one can compute $U_1^\pi(s)$ using the following recursive equation:

$$\begin{aligned} U_t^\pi(s_t) &= r_t(s_t, a_t) + E_{s_t}^\pi \left\{ U_{t+1}^\pi(s_{t+1}) \right\} \\ &= r_t(s_t, a_t) + \sum_{j \in S} p(j|s_t, a_t) U_{t+1}^\pi(j). \end{aligned} \tag{4.10}$$

This equation is intuitive as it indicates that the expected *remaining* reward is equal to the sum of the immediate reward and the expected *remaining* reward of the next state.

Based on (4.10), it can be shown that the optimal policy $\pi^* = (d^*(s_1), d^*(s_2), ..., d^*(s_N))$ can be solved using the Backward Induction Algorithm [61], in order to produce the maximum final cumulative reward.

$$\pi^* = \arg\max_{a \in A} E_{s_t}^\pi \left\{ \sum_{n=t}^{N-1} r_n(a_n, a_n) + r_N(s_N) \right\}. \tag{4.11}$$

The algorithm is summarized as follows.

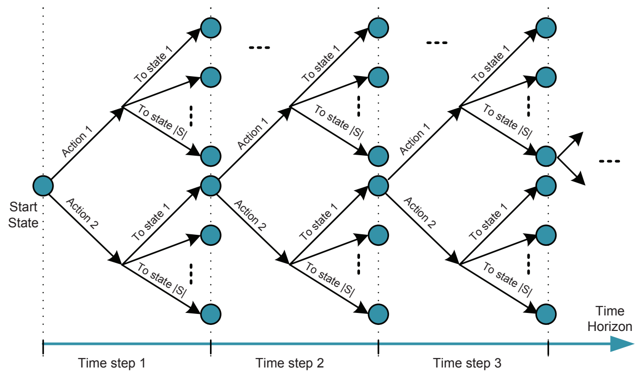


Figure 4.2: Trellis diagram for MDP state transitions.

1. *Set* $t = N$ *and* $U^*_N(s_N) = r_N(s_N)$ *for all* $s_N \in S$*,*

2. *Substitute* $t-1$ *for* t *and compute* $U^*_t(s_t)$ *for each* $s_t \in S$ *by*

$$U^*_t(s_t) = \max_{a \in A} \left\{ r_t(s_t, a) + \sum_{j \in S} p(j|s_t, a) U^*_{t+1}(j) \right\}.$$

   Set

$$d^*(s_t) = \arg\max_{a \in A} \left\{ r_t(s_t, a) + \sum_{j \in S} p(j|s_t, a) U^*_{t+1}(j) \right\}.$$

3. *If* $t = 1$*, stop. Otherwise return to step 2.*

Fig. 4.2 shows the Trellis diagram of the state transitions for the Markov decision process. From the "start state", taking an action results in to a next state with some probability. The process continues until the end of the finite time horizon. The MDP solution tool must find the path in the graph that provide the best sum average reward.

Having presenting a MDP formulation and an efficient solution using Backward Induction Algorithm, we note that solving a typical MDP problem involving two tasks: modeling and selection of solution tools. In the modeling task, a particular real-world problem is translated into an abstract MDP problem. This involves modeling the states, the actions, the immediate rewards, the transition probabilities, and the desired objective [1]. This modeling process can be hard, and often require domain experts. Even when the states are well modeled, computing the transition probabilities or selecting an appropriate reward function to fit a particular purpose can be difficult. In other cases, representing the system states accurately may require a large state and action spaces, making it hard to solve a MDP in practice. Thus, it is necessary for one to use appropriate approximate algorithms in order to solve the problem in a reasonable amount of time. Readers may refer to [61, 62] for the details on Markov Decision Process.

4.3.1 Model and Assumption

We now describe the broadcast and unicast models in WLAN-like and WiMAX-like networks. Based on this model, we show how NC and MDP can be used to increase the bandwidth efficiency while optimizing the concurrent applications based on their requirements. In particular, we are interested in designing a packet scheduling algorithm running at a WLAN-like AP or WiMAX-like broadcast station that optimizes multiple concurrent wireless applications as shown in Fig. 4.3. In this

[1]The desire objective does not have to be the sum of all the immediate rewards. A popular reward is the discount reward where the future reward is weighted less than the current reward.

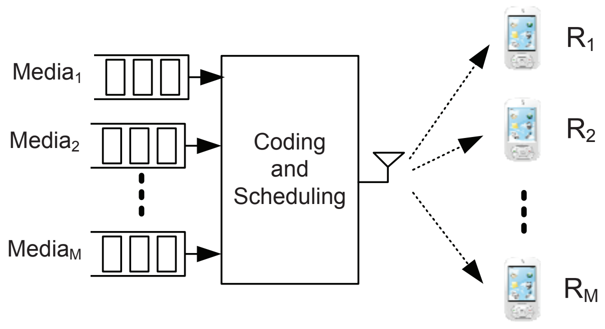


Figure 4.3: A media streaming model: multiple media streams are transmitted to multiple wireless receivers from the Access Point. The transmission can be a broadcast where all streams are broadcasted to all receivers or a concurrent unicast where streams are simultaneously unicasted to multiple receivers. Our joint coding and scheduling scheme is employed at the Access Point to minimize the media distortion.

setting, we assume that data of different applications flow mostly in one direction, from the AP to the users. These applications may have different bandwidth, delay, and loss requirements. For example, video streaming applications may require higher bandwidth and shorter delay than that of file downloading applications.

However for clarity, we present an optimized packet scheduling algorithm, designed exclusively for video broadcast and unicast flows from the AP to one or more receivers. The objective of the algorithm is to maximize the visual quality of videos received at the receivers under a certain bandwidth constraint.

We make the following assumptions for our model:

1. There are M receivers R_1, R_2, ..., R_M.

2. The AP has a set $\Omega = \{l_1, l_2, ..., l_K\}$ of K packets that needed to be delivered

to the receivers after some time slots N. In a broadcast setting, all the receivers request all K packets, while in a unicast setting, each receiver would request different subsets of Ω. In a semi-broadcast setting, there are two or more receivers requesting the same subset of Ω.

3. There is a limit on the total number of time slots N used to transmit these K packets. After N time slots, the AP moves to the next batch of K packets, regardless of whether or not all K packets have been successfully received at the intended receivers. Thus, the quality of the media streams at the receivers might be reduced by some amount proportional to the number and the type of the unavailable packets. One can view this mechanism as a way to impose a delay requirement on a streamed video. Alternatively, this mechanism can also be viewed as a bandwidth allocation that assigns a fixed number of time slots per unit time to a video stream.

4. Any receiver can cache packets transmitted from the AP to other receivers, even though those packets are not directly useful to itself. We assume that appropriate encryption is employed to provide the receiver privacy.

5. Data is divided into packets, each is sent in a time slot of fixed duration.

6. The AP knows which packet from which receiver is lost. This can be accomplished through the use of positive or negative acknowledgments (ACK/NAKs). If an ACK or NAK is lost, the corresponding data packet is also considered lost.

7. The distribution of packet loss at a receiver R_i follows the Bernoulli distribution with parameter p_i. This model is clearly insufficient to describe many real-world scenarios. However, it is only intended to capture the essence of the problem. One can develop a more accurate model, albeit complicate analysis. In [6], we provide an analysis for a slightly more accurate model that reflects the correlated loss among the receivers.

Assumptions on packetized media model: In the media packetized model, the dependence structure of media packets of a frame can be presented as a directed line graph. Specifically, the video sequence is supposed to be divided into frames and each frame is then encoded into K dependent layers (Fig. 4.4) (a) consisting of one base layer and $K-1$ enhancement layers. All the bits in a layer can be packetized into one packet, resulting in K packets in a frame as in 4.4a. Associated with each packet P_i are its contribution $\triangle d_i$ to overall distortion reduction of the frame, the packets it depend on for correct decoding $P_1, P_2, ...$, and P_{i-1} , and its deadline D. As the frame is encoded into K packets, the packet order $P_1, P_2, ...$, and P_K is of decreasing degrees of importance. Packet P_i can be used for playback only if it is received before its deadline and all packets that it depends on are received previously. The average distortion of a media frame at a client can be quantitatively defined as

$$\theta = \sum_{i=1}^{K} \triangle d_i \prod_{j \leq i} (1 - \varepsilon_j), \tag{4.12}$$

where $\varepsilon_i = 1$ is the indication that packet P_i is received correctly and $\varepsilon_i = 0$ indicates otherwise. d_i is the distortion reduction measured in Mean Squared

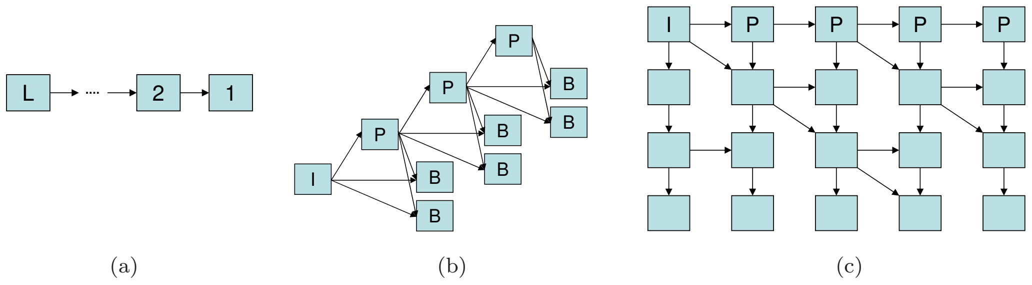


Figure 4.4: Dependencies of coded video packets within a frame are presented as a directed graph: (a) Sequential dependencies as directed line graph; (b) Dependencies of a standard video coder; (c) Typical dependencies for MPEG-4 progressive fine grain scalability mode.

Error (MSE) amount associated with packet P_i not being used for playback. The term $\prod_{j \leq i}(1-\varepsilon_j)$ expresses the dependence of packet P_i on packets $P_1, P_2, ...,$ and P_{i-1}. In our paper, we use average distortion reduction in comparison with θ_{max}, where $\theta_{max} = \sum_{i=1}^{K} \triangle d_i$, as one of metrics to evaluate the performances of different schemes.

4.3.2 Problem Statement

We first again reexamine a wireless broadcast transmission using NC that is presented in the previous chapter. For simplicity, suppose 2 packets a and b are broadcasted from a wireless AP to 2 receivers R_1 and R_2. Let us first examine a non-NC broadcast scheme in which, packets are sent in time slots. In a 802.11x network, if a packet is received correctly, the AP should receive an ACK within an appropriate amount of time after the data packet is sent. Otherwise, the data

Time slots	1	2	3	4
Packet sent	a	a	b	b
Events at R1,R2	Lost at R1, received at R2	Received at R1	Lost at R2, received at R1	Received at R2

(a)

Time slot	1	2	3
Packet sent	a	b	a⊕b
Events at R1,R2	Lost at R1, received at R2	lost at R2, received at R1	Received at both R1, R2

(b)

Figure 4.5: An example of using XOR NC to reduce number of retransmissions for sending two packets a and b to two receivers R_1 and R_2: (a) Two retransmissions for the non-NC model and (b) One retransmission if using NC.

packet is considered lost, and must be retransmitted. Using this scheme, a packet loss at any receiver will require the AP to retransmit that packet. If two distinct lost packets at two different receivers, the AP will need at least two retransmissions as seen in Fig. 4.5(a). In the first time slot, packet a is lost at R_1, but is received correctly at R_2. Therefore, in the second time slot, the AP rebroadcasts packet a, and R_1 receives the packet a correctly at this time. In the third time slot, the AP sends packet b which is lost at R_2, but is received correctly at R_1. In the fourth time slot, the AP rebroadcasts packet b, and R_2 receives b correctly this time. Using this scheme, a total of 4 transmissions are required to transmit both packets a and b to receivers R_1 and R_2.

We now consider a NC technique that requires only one retransmission to recover two lost packets at both receivers. Unlike the traditional scheme, using the NC scheme, the AP does not retransmit the lost packet a at R_1 immediately. Instead, the AP continues to broadcast the next packet until there is a lost packet

b at receiver R_2. At this time, the AP broadcasts the new packet $(a \oplus b)$ to both receivers. If R_1 has packet b but not a, and R_2 has packet a but not b, then both receivers will be able to reconstruct their missing packets by simply XORing the packet they have, with the packet $(a \oplus b)$. As shown in Fig. 4.5(b), R_1 reconstructs a as $b \oplus (a \oplus b)$ and R_2 reconstructs b as $a \oplus (a \oplus b)$. Therefore, one retransmission from the AP will enable both receivers to correctly reconstruct their lost packets.

It is straightforward to extend the NC technique to unicast setting. Assume that R_1 wants to receive packet a while R_2 wants to receive packet b. Clearly, if R_1 is willing to cache packet b intended for R_2, and R_2 is willing to cache packet a intended for R_1, then the two unicast sessions are now equivalent to a single broadcast session in the previous example. Similarly, when there are N receivers that want to receive N different packets, a receiver may want to cache everyone else's data in order to use NC for higher bandwidth efficiency.

In our media streaming system, because a media packet is characterized by its degree of importance, its dependency among packets, and its time-sensitivity, under the NC paradigm, it is beneficial to find a scheduling transmission policy in order to efficiently utilize the available transmission bandwidth. For instance, in the example above, at the first time slot, a is received correctly at R_2 but lost at R_1. In the second time slot, the AP has two options: either retransmitting a or transmitting b. If it chooses to transmit b, there can be an opportunity for transmitting the combined packet (i.e., $a \oplus b$) in the third time slot as shown in the above example. However, if the deadline is short or the packet loss rate is high, then retransmitting a at the second time slot may provide a higher media quality if a is an important packet. Thus the AP has to schedule which packets should be sent at which time slot in order to achieve the best media quality, for given packet loss rate and bandwidth.

We can state our joint coding and scheduling problem as follows. There are K media packets $Q_1, Q_2, ...,$ and Q_K which are buffered at the AP and transmitted to M wireless receivers $R_1, R_2, ...,$ and R_M within N transmission opportunities. Those packets may have different importance and priorities, and dependent on each other in reconstructing the original media. Both the forward and feedback channels are unreliable, resulting in media packet and acknowledge packet losses or corruption. We are interested in designing a joint NC and transmission scheduling algorithm that chooses which packets, i.e., new packets, lost packets or coded packets, to send at which times, to produce in the best possible reconstructed media quality under insufficient network resources.

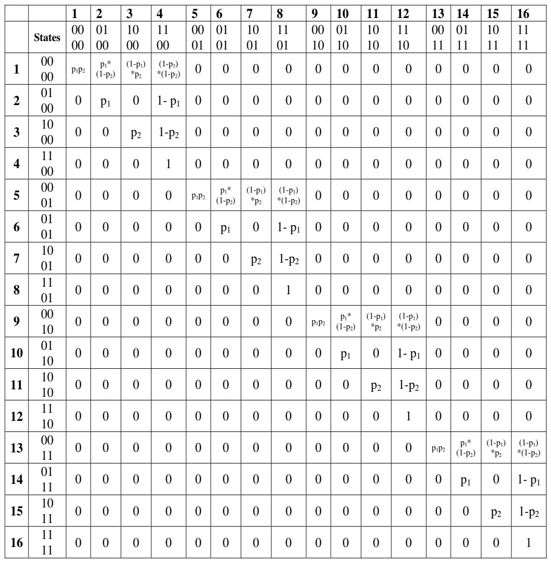
In order to quantify the media quality that we want to maximize, we define that each media packet, if received correctly on time, will reduce an amount of the total media distortion. Thus our optimization problem is to maximize the total amount of distortion reduction of the transmitted media. In the next section, we present the joint NC and scheduling framework based on the MDP with some remarks on the complexity of this method.

4.4 Optimal MDP Based Packet Scheduling

As mentioned earlier, solving a MDP problem typically involves two phases: modeling and solution tool selection. The modeling phase involves translating our wireless broadcast and unicast problems into abstract MDPs. Specifically, in this section, we discuss the modeling of the set of states S, the set of actions A, the immediate reward $r(s_t, a_t)$, the transition probabilities $P(s_{t+1}|s_t, a_t)$, and the cumulative rewards.

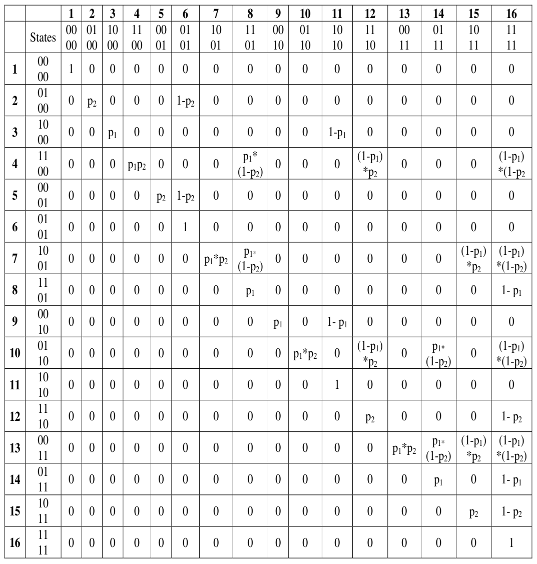
Our packet scheduling algorithm works as follows. At every time step, the AP sends a packet, and waits for an ACK message. If a receiver receives a packet, an ACK is sent back immediately, similar to the 802.11x protocol. If no ACK is received within a specified time frame, the data packet is considered lost. The AP then can choose to send a new packet, retransmit a lost packet, or transmit a XOR packet. We now proceed to model our packet scheduling algorithm as a MDP of finite horizon N where N is the maximum number of allowable time slots to transmit K packets.

State Representation. At any given time slot, receiver R_i possesses a subset of packets belonged to Ω, including the packets that are intended for other receivers. This subset can be represented by an K-bit vector as $(b_j^1, b_j^2, ..., b_j^K)$ where $b_j^i \in \{0, 1\}$. $b_j^i = 1$ indicates the presence of packet l_i at R_j, while $b_j^i = 0$ indicates otherwise. Since there are M receivers, a system configuration or state s can be



		1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**
	States	00 00	01 00	10 00	11 00	00 01	01 01	10 01	11 01	00 10	01 10	10 10	11 10	00 11	01 11	10 11	11 11
1	00 00	p_1p_2	p_1* $(1-p_2)$	$(1-p_1)$ *p_2	$(1-p_1)$ *$(1-p_2)$	0	0	0	0	0	0	0	0	0	0	0	0
2	01 00	0	p_1	0	$1-p_1$	0	0	0	0	0	0	0	0	0	0	0	0
3	10 00	0	0	p_2	$1-p_2$	0	0	0	0	0	0	0	0	0	0	0	0
4	11 00	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
5	00 01	0	0	0	0	p_1p_2	p_1* $(1-p_2)$	$(1-p_1)$ *p_2	$(1-p_1)$ *$(1-p_2)$	0	0	0	0	0	0	0	0
6	01 01	0	0	0	0	0	p_1	0	$1-p_1$	0	0	0	0	0	0	0	0
7	10 01	0	0	0	0	0	0	p_2	$1-p_2$	0	0	0	0	0	0	0	0
8	11 01	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
9	00 10	0	0	0	0	0	0	0	0	p_1p_2	p_1* $(1-p_2)$	$(1-p_1)$ *p_2	$(1-p_1)$ *$(1-p_2)$	0	0	0	0
10	01 10	0	0	0	0	0	0	0	0	0	p_1	0	$1-p_1$	0	0	0	0
11	10 10	0	0	0	0	0	0	0	0	0	0	p_2	$1-p_2$	0	0	0	0
12	11 10	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
13	00 11	0	0	0	0	0	0	0	0	0	0	0	0	p_1p_2	p_1* $(1-p_2)$	$(1-p_1)$ *p_2	$(1-p_1)$ *$(1-p_2)$
14	01 11	0	0	0	0	0	0	0	0	0	0	0	0	0	p_1	0	$1-p_1$
15	10 11	0	0	0	0	0	0	0	0	0	0	0	0	0	0	p_2	$1-p_2$
16	11 11	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

(a)



		1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**
	States	00 00	01 00	10 00	11 00	00 01	01 01	10 01	11 01	00 10	01 10	10 10	11 10	00 11	01 11	10 11	11 11
1	00 00	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	01 00	0	p_2	0	0	0	$1-p_2$	0	0	0	0	0	0	0	0	0	0
3	10 00	0	0	p_1	0	0	0	0	0	0	0	$1-p_1$	0	0	0	0	0
4	11 00	0	0	0	p_1p_2	0	0	0	p_1* $(1-p_2)$	0	0	0	$(1-p_1)$ *p_2	0	0	0	$(1-p_1)$ *(1-p_2
5	00 01	0	0	0	0	p_2	$1-p_2$	0	0	0	0	0	0	0	0	0	0
6	01 01	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
7	10 01	0	0	0	0	0	0	p_1*p_2	p_1* $(1-p_2)$	0	0	0	0	0	0	$(1-p_1)$ *p_2	$(1-p_1)$ *$(1-p_2)$
8	11 01	0	0	0	0	0	0	0	p_1	0	0	0	0	0	0	0	$1-p_1$
9	00 10	0	0	0	0	0	0	0	0	p_1	0	$1-p_1$	0	0	0	0	0
10	01 10	0	0	0	0	0	0	0	0	0	p_1*p_2	0	$(1-p_1)$ *p_2	0	p_1* $(1-p_2)$	0	$(1-p_1)$ *$(1-p_2)$
11	10 10	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
12	11 10	0	0	0	0	0	0	0	0	0	0	0	p_2	0	0	0	$1-p_2$
13	00 11	0	0	0	0	0	0	0	0	0	0	0	0	p_1*p_2	p_1* $(1-p_2)$	$(1-p_1)$ *p_2	$(1-p_1)$ *$(1-p_2)$
14	01 11	0	0	0	0	0	0	0	0	0	0	0	0	0	p_1	0	$1-p_1$
15	10 11	0	0	0	0	0	0	0	0	0	0	0	0	0	0	p_2	$1-p_2$
16	11 11	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

(b)

Figure 4.6: (a) Transition probability matrix associated with action "sending packet l_1"; (b) Transition probability matrix associated with action "sending packet $l_1 \oplus l_2$".

represented by an $M \times K$ matrix with binary entries as:

$$s = \begin{bmatrix} b_1^1 & b_1^2 & \dots & b_1^K \\ b_2^1 & b_2^2 & \dots & b_2^K \\ \dots & \dots & \dots & \dots \\ b_M^1 & b_M^2 & \dots & b_M^K \end{bmatrix} \tag{4.13}$$

Thus, there are $2^{M \times K}$ possible states.

Action Representation. At any given time slot, the AP can (a) broadcast any $l_i \in \Omega$, (b) broadcast any XOR packet resulting from XORing the distinct lost

		1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**
	States	00 00	01 00	10 00	11 00	00 01	01 01	10 01	11 01	00 10	01 10	10 10	11 10	00 11	01 11	10 11	11 11
1	00 00	0	r_{21}	r_{11}	$r_{11}+r_{21}$	0	0	0	0	0	0	0	0	0	0	0	0
2	01 00	0	0	0	r_{11}	0	0	0	0	0	0	0	0	0	0	0	0
3	10 00	0	0	0	r_{21}	0	0	0	0	0	0	0	0	0	0	0	0
4	11 00	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	00 01	0	0	0	0	0	r_{21}	r_{11}	$r_{21}+r_{11}$	0	0	0	0	0	0	0	0
6	01 01	0	0	0	0	0	0	0	r_{11}	0	0	0	0	0	0	0	0
7	10 01	0	0	0	0	0	0	0	r_{21}	0	0	0	0	0	0	0	0
8	11 01	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
9	00 10	0	0	0	0	0	0	0	0	0	r_{21}	r_{11}	$r_{11}+r_{21}$	0	0	0	0
10	01 10	0	0	0	0	0	0	0	0	0	0	0	r_{11}	0	0	0	0
11	10 10	0	0	0	0	0	0	0	0	0	0	0	r_{21}	0	0	0	0
12	11 10	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
13	00 11	0	0	0	0	0	0	0	0	0	0	0	0	0	r_{21}	r_{11}	$r_{11}+r_{21}$
14	01 11	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	r_{11}
15	10 11	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	r_{21}
16	11 11	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

(a)

		1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**
	States	00 00	01 00	10 00	11 00	00 01	01 01	10 01	11 01	00 10	01 10	10 10	11 10	00 11	01 11	10 11	11 11
1	00 00	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	01 00	0	0	0	0	0	r_{22}	0	0	0	0	0	0	0	0	0	0
3	10 00	0	0	0	0	0	0	0	0	0	0	r_{12}	0	0	0	0	0
4	11 00	0	0	0	0	0	0	0	r_{22}	0	0	0	r_{12}	0	0	0	$r_{12}+r_{22}$
5	00 01	0	0	0	0	0	r_{21}	0	0	0	0	0	0	0	0	0	0
6	01 01	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	10 01	0	0	0	0	0	0	0	r_{21}	0	0	0	0	0	0	r_{12}	$r_{21}+r_{12}$
8	11 01	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	r_{12}
9	00 10	0	0	0	0	0	0	0	0	0	0	r_{11}	0	0	0	0	0
10	01 10	0	0	0	0	0	0	0	0	0	0	0	r_{11}	0	r_{22}	0	$r_{11}+ r_{22}$
11	10 10	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
12	11 10	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	r_{22}
13	00 11	0	0	0	0	0	0	0	0	0	0	0	0	0	r_{21}	r_{11}	r_{21}
14	01 11	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	r_{11}
15	10 11	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	r_{21}
16	11 11	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

(b)

Figure 4.7: (a) Reward matrix associated with action "sending packet l_1"; (b) Reward matrix associated with action "sending packet $l_1 \oplus l_2$".

packets from different receivers, and (c) broadcast nothing. This implies that the number of possible actions J at any time step:

$$J = K + \sum_{i=2}^{L} \begin{pmatrix} L \\ i \end{pmatrix} + 1, \tag{4.14}$$

where L denotes the number of packets which are lost at one or more receivers. The maximum number of lost packets L is K, however this case is extremely rare for large K.

Transition Probability. Given the Bernoulli model with parameter p_i for packet loss at each receiver R_i, it is straightforward to compute the transition probability $P(s_{t+1} = s'|s_t = s, a_t = a)$. For example, consider broadcasting two packets to two receivers, i.e., $K = 2$ and $M = 2$. Let us denote

$$s = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \quad s' = \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}$$

Suppose at time t, the system is in the state s, i.e., R_1 has packet l_1 and R_2 has packet l_2, then choosing action a = "send l_1" in state s will move the system to state s' with probability:

$$P(s_{t+1} = s'|s_t = s, a_t = a) = 0 \tag{4.15}$$

while choosing action a' = "send l_2" will move the system to state s' with probability:

$$P(s_{t+1} = s'|s_t = s, a_t = a') = 1 - p_1 \tag{4.16}$$

Reward Modeling. The immediate reward $r(s, a)$ for each state and action pair must be chosen such that the sum of these immediate rewards models our objective accurately. Since our objective is to optimize the quality of multimedia streaming applications, we model the immediate rewards as the sum of the reduction in distortion for one or more receivers upon receiving a particular packet. Thus, maximizing the overall reward is equivalent to minimizing the overall distortion for all the receivers' applications under a particular bandwidth constraint.

In our setting, taking an action a in a state s does not provide an explicit immediate reward. Rather, we know the explicit reward amount $r(s', s)$ when the system moves from state s to state s'. For example, given a detail profile of a layered video, we know the amount of distortion reduction contributed by each layer. If state s indicates that a receiver has layers 1 and 2, and state s' indicate the a receiver has layers 1, 2, and 3, then moving from state s to state s' would reward us an amount $r(s', s)$ equal to the distortion reduction contributed by layer 3. Since we know the transition probability between states under an action a, we can compute $r(s, a)$ as the expected immediate reward by taking action a as :

$$r(s,a) = \sum_{j \in S} P(j|s,a) r(j,s) \tag{4.17}$$

Example. We now present a simple example showing MDP formulations for broadcast and unicast settings with two receivers and two packets. For the state space, there would be a total of 16 states with each state s represented by:

$$s = \begin{bmatrix} b_1^1 & b_1^2 \\ b_2^1 & b_2^2 \end{bmatrix}.$$

$b_j^i = 1$ represents that receiver R_j has packet l_i, and $b_j^i = 0$ represents otherwise.

As for the action space, at any time step, the AP can perform one of the four actions: send l_1, send l_2, send $l_1 \oplus l_2$, and send nothing.

To compute the transition probabilities, let us denote p_1 and p_2 as the packet loss probabilities at R_1 and R_2, respectively. For each action, there is an associated transition probability matrix. In this example, we show two transition probability matrices due to taking actions "sending l_1" and "sending $l_1 \oplus l_2$", respectively.

The transition probability matrix for taking actions "sending l_2" and "not sending anything" can be similarly computed.

First let us consider the transition probability matrix for taking action "sending l_1". This is shown in Fig. 4.6(a). An entry in row i and column j denotes the transition probability from state i to state j under action "sending l_1". For example, the probability of transition from state 1 to 4 when sending packet l_1 is $(1-p_1)(1-p_2)$. The reason is as follows. Since state 1 denotes neither receivers have packets l_1, and the state 4 denotes both receivers have packets l_1, to transition from state 1 to state 4 by sending packet l_1, both receivers must have correctly received l_1, and the probability of this event equals to $(1-p_1)(1-p_2)$. Similarly, other transition probabilities for different states can be computed by using the packet loss probabilities at each receiver.

Let us now consider the transition probability matrix for taking action "sending $l_1 \oplus l_2$". This action is interesting as one transmission by the AP can help two receivers to simultaneously recover two distinct lost packets. Consider a transition from state 10 to state 16 in Fig. 4.6(b). In state 10, R_1 has l_2 but not l_1 while R_2 has l_1 but not l_2. If the AP sends packet $l_1 \oplus l_2$, and the packet is successfully received at both receivers, then both R_1 and R_2 will now obtain $l_1 = l_2 \oplus (l_1 \oplus l_2)$ and $l_2 = l_1 \oplus (l_1 \oplus l_2)$, respectively. The probability of this event is then equal to $(1-p_1)(1-p_2)$. Other probability entries can be calculated in a similar manner.

Now for each action, there is an associated reward matrix. Let us denote r_{ij} as the immediate reward of R_i upon receiving packet l_j. It can be seen that for broadcast setting, $r_{11} = r_{21}$ and $r_{12} = r_{22}$ since both receivers want packets l_1 and

1. Set $t = N$ and $U_N^*(s_N) = 0$ for all $s_N \in S$,
2. Substitute $t-1$ for t and compute $U_t^*(s_t)$ for each $s_t \in S$ as follows
   - Sample each action $a \in A$ for N_i iterations and compute the average
   $$\hat{U}_t(s_t, a) = \frac{1}{N_i} \sum_{N_i} \Big(r_t(s_t, a) + U_{t+1}^*(j) \Big) \tag{4.18}$$
   - Find the highest reward
   $$U_t^*(s_t) = \max_{a \in A} \{ \hat{U}_t(s_t, a) \}.$$
   - Set
   $$d^*(s_t) = \arg\max_{a \in A} \hat{U}_t(s_t, a).$$
3. If $t = 1$, stop. Otherwise return to step 2.

Figure 4.8: Simulation-based Dynamic Programming Algorithm

l_2. For unicast setting, we assume that R_1 wants only l_1 while R_2 wants only l_2, thus $r_{12} = 0$ and $r_{21} = 0$. Given this definition, we can express the reward matrix for both unicast and broadcast settings when sending l_1 as shown in Fig. 4.7(a). For example, the immediate reward when transitioning from state 1 to state 4 under action "sending l_1" is $r_{11} + r_{21}$. The reason is that the reward in state 1 is zero, and with a transition to state 4, both receivers receive l_1 and l_2. Thus, the immediate reward should be equal to the sum of the individual rewards. In the broadcast and unicast settings, this sum equals to $2r_{11}$ and r_{11}, respectively. Similarly, we can write down the reward matrix for sending $l_1 \oplus l_2$ as shown in Fig. 4.7(b).

Remarks on the modeling and computational complexity of BIA: For a small number of receivers, we can use BIA to solve our abstract MDP that corresponds to the scheduling problem. We see that, however, the number of states and actions can be exponentially large under certain settings. Specifically, the number of states is $|S| = 2^{M \times K}$ and the number of actions is $|A| = 2^K$, both increase exponentially as the number of receivers increases. For instance, if we have $M = 6$ receivers and $K = 3$ packets, then the number of states is $|S| = 2^{18}$ and the number of actions is $|A| = 2^3$. From the modeling perspective, BIA requires us to define all state transition probabilities and reward of all transitions. This is infeasible with a large number of states and actions. From the computational perspective, its time complexity of $O(N|S|^2|A|)$ quickly becomes intractable, even for a broadcast session with a moderate number of receivers. To tackle those issues, we propose an algorithm called Simulation-based Dynamic Programming (SDP).

4.5 Simulation-Based Dynamic Programming Algorithm

As noted above, the time complexity of BIA is $O(N|S|^2|A|)$ which is dominated by the number of states $|S| = 2^{M \times K}$. The number of actions $|A| = 2^K$ is also large, but much smaller than the number of states, and is manageable for typical scenarios. Therefore, our goal is to devise an algorithm that reduces the number of states and modeling complexity. Specifically, we propose an dynamic programming algorithm that is based on simulations. The intuition for using such an approach is that, for many problems, going through all the states to determine the optimal actions is not efficient. Rather, only the most likely states will be explored for determining the optimal action. Furthermore, the most likely states can be obtained via simulations. This simulation-based approach has been studied extensively in the Artificial Intelligence literatures [63–67]. In [64, 65] , the authors employed the simulation-based approach to address the MDP problem with either a large state space or a large action space. In addition, [67] presents an algorithm called PEGASUS which is a policy search method for large MDPs. In this thesis, the author proposes a simple simulation-based method, which combines both dynamic algorithm and sampling method to solve our large MDP. Our method addresses both the complexity of large state space and modeling process.

4.5.1 Simulation-based Dynamic Programming for Large MDP

Our SDP algorithm works based on BIA. Specifically, for a given state s, the SDP algorithm sample each action a in the action space A for a number of iterations N_i to compute the average sampling reward of that tuple (s, a). From those average sampling rewards, the action for a given state that results in the largest reward is the best action. The algorithm is shown in Fig. 4.8 and graphically illustrated in Fig. 4.9. The SDP algorithm samples each action in the action space to determine the next state and the transition reward. The process runs backward from $t = N$ to $t = 1$ similarly as in BIA to find the near optimal policy $\pi^* = \{d^*(s_1), d^*(s_2), ..., d^*(s_N)\}$ that produces the maximum final cumulative reward.

$$\pi^* = \arg\max_{a \in A} E^{\pi}_{s_t} \left\{ \sum_{n=t}^{N-1} r_n(s_n, a_n) + r_N(s_N) \right\}. \tag{4.19}$$

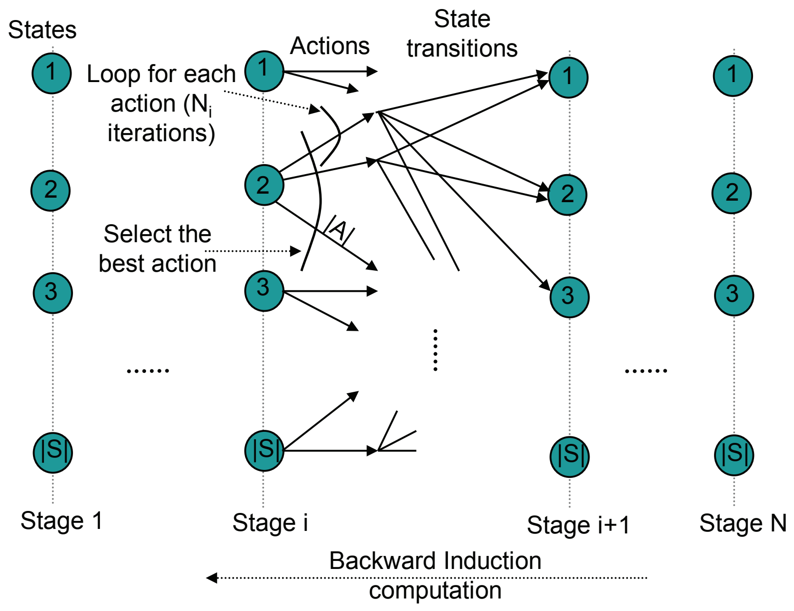


Figure 4.9: Graphical illustration of the SDP algorithm. For each state at stage i, all actions in the action space are sampled for N_i iterations and the mean average reward is computed, the best action is the one with highest mean average reward. The algorithm computes the reward following the Backward Induction Algorithm starting from stage N to stage 1.

4.5.2 Evaluating the Properties of the SDP algorithm

We first examine the SDP algorithm for solving our MPD defined above. Clearly, how close this policy resulted from the SDP algorithm to the optimal policy found by BIA depends on the number of samples per action N_i. According to the law of large number, the larger the number of samples, the closer to the optimal policy the result is. As N_i goes to infinity, the algorithm results in the optimal policy. To see how the algorithm converges, we consider a transmission example in which two packets l_1 and l_2 are broadcasted to two receivers R_1 and R_2. Assume that

	stages			
state	**1**	**2**	**3**	**4**
1	1	1	1	1
2	1	1	1	1
3	1	1	1	1
4	1	3	3	3
5	1	2	2	2
6	1	2	2	1
7	3	3	3	3
8	3	1	1	3
9	1	2	2	2
10	3	3	3	3
11	1	1	1	1
12	1	1	1	1
13	2	3	3	3
14	3	3	3	2
15	2	2	3	2
16	0	0	0	0

(1)

	stages			
state	**1**	**2**	**3**	**4**
1	1	1	1	1
2	1	1	1	1
3	1	1	1	1
4	1	3	1	1
5	1	2	1	1
6	1	2	2	1
7	3	3	3	3
8	3	3	1	1
9	1	2	2	2
10	3	3	3	3
11	1	1	1	1
12	1	1	1	1
13	2	3	2	3
14	3	3	2	2
15	2	2	3	2
16	0	0	0	0

(2)

	stages			
state	**1**	**2**	**3**	**4**
1	1	1	1	1
2	1	1	1	1
3	1	1	1	1
4	1	1	1	1
5	2	2	1	1
6	1	1	1	1
7	3	3	3	3
8	1	1	1	1
9	1	1	1	1
10	1	1	1	3
11	1	1	1	1
12	1	1	1	1
13	2	2	2	2
14	2	2	2	2
15	2	2	2	2
16	0	0	0	0

(3)

Figure 4.10: Different transmission policies resulted from different number of samples per action N_i: (1) transmission policies resulted by running 6 samples per action, (2) transmission policy resulted by running 25 samples per action, and (3) the actual optimal policy by BIA.

when a receiver receives packet l_1 or l_2 correctly, it gets an amount of reward 0.7 or 0.3, respectively. Assume further that l_2 depends on l_1, i.e., l_2 can be useful only when l_1 has been received. The packet error probabilities $p_1 = p_2 = 0.1$. As noted above, there are 16 different states: $\begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$, $\begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix}$, ..., and $\begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}$ and four different actions a_0=“sending nothing”, a_1=“sending l_1”, a_2=“sending l_2”, and a_3=“sending $l_1 \oplus l_2$”. We set the time horizon to be 3 time steps.

Convergence: As seen in Fig. 4.10, we have the transmission policies resulted by different number of samples per action N_i. In Fig. 4.10(1), $N_i = 6$, the actions are quite different from the actions in the optimal policy in Fig.4.10(3). For more number of samples per action, $N_i = 25$, the policy becomes closer to the optimal

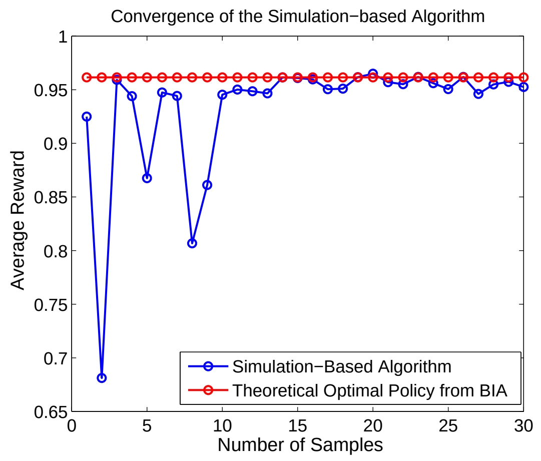


Figure 4.11: The convergence rate of policy by the SDP algorithm to the optimal as the number of samples per action increases.

one. Fig. 4.11 shows the convergence rate to the optimal policy in term of the amount of average reward per receiver. When increasing the number of samples per action, the reward value gets to close to the optimal value.

Complexity: Our simulation-based algorithm avoids the complexity of modeling ("the curse of modeling" problem). When K and M are large, it is intractable to explicitly construct the transition probabilities and transition reward function. Using simulations, these are implicitly captured. Secondly, the SDP algorithm reduces the computational complexity ("the curse of dimensionality" problem). It is easy to see that the time complexity for this algorithm is $O(NN_i|S||A|)$. Because N_i is normally much smaller than $|S|$, this time complexity is significantly less than $O(N|S|^2|A|)$ of BIA.

4.6 Scheduling with Erroneous Feedback

So far, we assume that the feedbacks from the receivers are error-free, so that the AP knows the states of the receivers exactly and our optimization problem can be nicely solved by solving the corresponding abstract MDP. In this section, we further examine the case of erroneous feedback, i.e., the feedback channels are unreliable.

4.6.1 Hidden Markov Model of Receivers' States

With error-free feedback, the AP obtains the states of the receivers correctly and views the state transitions of receivers as a Markov process. With error-prone feedback, however the AP views those states as hidden states, i.e., partially observable states. Thus in the transmission scenarios with erroneous feedback, the AP gets the observations that are not actual the states of the receivers as in the scenarios with error-free feedback. To illustrate the difference between the actual states and the observed states or observation from the AP, let us consider again the example in which two packets l_1 and l_2 are sent to two receivers R_1 and R_2. Assume that the beginning actual state is $s_1 = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$ and the AP, at the beginning also observes the state as $o_1 = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$. Taking action "sending l_1" will result two receivers receiving l_1 correctly. The next actual state becomes $s_2 = \begin{bmatrix} 1 & 0 \\ 1 & 0 \end{bmatrix}$.

Through the feedback, if received correctly at the AP, the AP knows exactly that the receivers' state $s_2 = \begin{bmatrix} 1 & 0 \\ 1 & 0 \end{bmatrix}$. If one of the feedback is error, however, the AP obtains an observation that is different from the actual state of the receivers. For instance, suppose the feedback from the first receiver is erroneous, the AP will get the observation $o_2 = \begin{bmatrix} 0 & 0 \\ 1 & 0 \end{bmatrix}$. Similarly, in the next time step, the AP takes action "sending l_2", which may results in the actual state $s_3 = \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}$ and the observation $o_3 = \begin{bmatrix} 0 & 0 \\ 1 & 1 \end{bmatrix}$. As a result, the actual state transition is $s_1 \to s_2 \to s_3$ that is different from the observation transition $o_1 \to o_2 \to o_3$. The AP, which acts as a decision maker, has to select which packet to send based on the partial observations in order to maximize the sum reward returned at the receivers. Our decision making problem, i.e., scheduling problem, will be modeled as a partially observable Markov decision processes (POMDP) in the next section [67–69].

4.6.2 POMDP for Scheduling with Erroneous Feedback

In order to model our scheduling problem as an POMDP, we need to define following components: 1) the state; 2) the observation; 3) the action; 4) the transition probabilities; 5) the transition reward, and 6) the time horizons, i.e., it is different from the completely observable MDP by the observation component. We define the state, the action, the transition probabilities, the transition reward, and the time horizon exactly the same as in the MDP. The observation is defined as in the example in Section 4.6.1. In general, it is defined as

$$o = \begin{bmatrix} ob_1^1 & ob_1^2 & ... & ob_1^K \\ ob_2^1 & ob_2^2 & ... & ob_2^K \\ ... & & & \\ ob_M^1 & ob_M^2 & ... & ob_M^K \end{bmatrix},$$

where entry $ob_j^i = \{0, 1\}$ indicates that the observation from the AP wether the receiver R_j has been received packet l_i or not: $ob_j^i = 1$ indicates that the AP observes that R_j has been received l_i correctly and $ob_j^i = 0$ indicates otherwise. We note again that if the feedback is error-free, then the observation completely describes the state of receivers, i.e., the observation and the state are the same ($o = s$). Because of the error feedback, however, it partially describes the receivers' state. The solution for the POMDP is a policy from time step $t = 1$ to $t = N$: $\pi^* = \{d^*(s_1), d^*(s_2), ..., d^*(s_N)\}$, where $d^*(s_t)$ is a set of actions at time step t that maximize the sum reward:

$$\pi^* = \arg\max_{a \in A} E_{s_t}^{\pi} \left\{ \sum_{n=t}^{N-1} r_n(o_n, a_n) + r_N(o_N) \right\}. \tag{4.20}$$

We note that this equation is different from Equation 4.19 by the observation o_n in the equation. This means that the AP relies on the observation o rather than exact state s as in the MPD to select action at each time step. In addition, because the observation above is actually the partially presentation of states, the MDP is

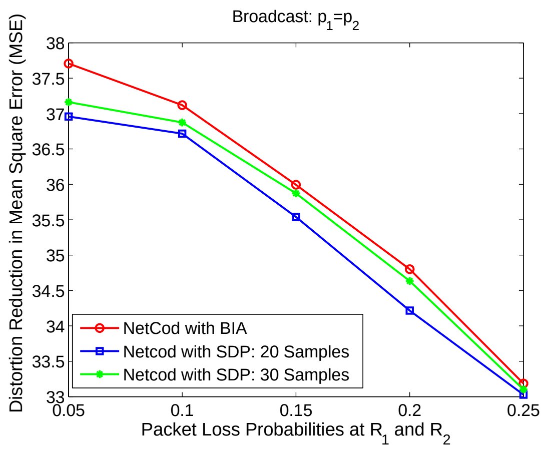


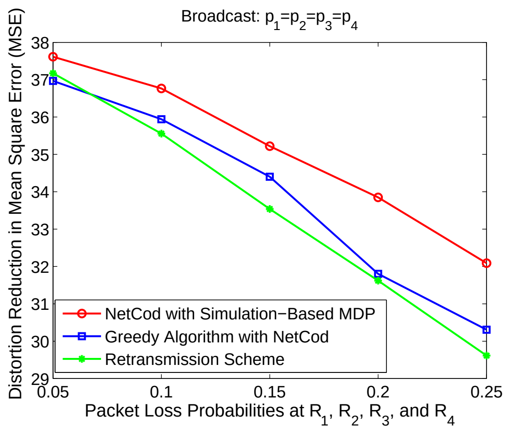
Figure 4.12: Average distortion reduction in MSE of video sequences at each receiver versus the packet loss probabilities for the broadcast setting with a small number of receivers.

just a special case of the POMDP. In other words, the POMDP will reduce to the MDP when the feedback is error-free.

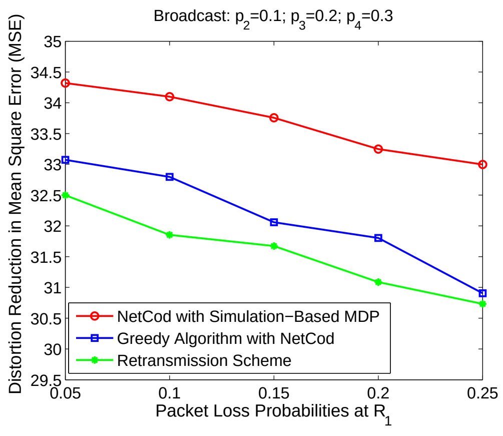
Fortunately, to solve our POMDP, we can again use the SDP algorithm. The difference is that the algorithm, for each observation which is a partial pretention of actual state, will sample all the actions in the action space to select the best action.

4.7 Simulation Results

In this section, we present some simulation results to demonstrate the advantages of our scheduling algorithm using NC with simulation-based MDP. We assume that the AP has a Foreman video sequence which is sent to the receivers. The sequence is assumed to have four layers l_1, l_2, l_3, and l_4. Each layer is packetized into different



(a)

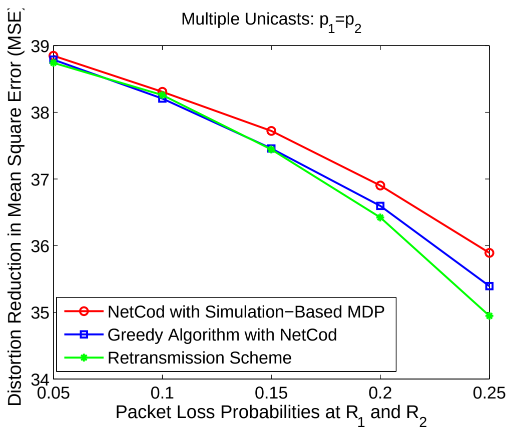


(b)

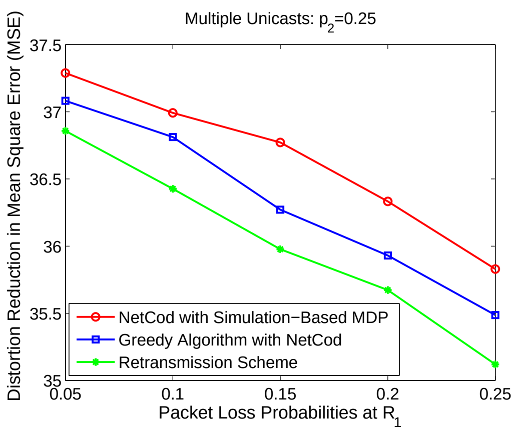
Figure 4.13: Average distortion reduction in MSE of video sequences at each receiver in the broadcast scenario with the fixed number of transmission opportunities: (a) Distortion versus packet loss probabilities at receivers R_1, R_2, R_3, and R_4; (b) Distortion reduction versus packet loss probability at R_1.

type of packets which are denoted as packets l_1^F, l_2^F, l_3^F, and l_4^F, respectively. Associated with each packet is a reward or a distortion reduction amount in term of MSE. In particular, we use the distortion reduction $r_1^F = 14.67, r_2^F = 10.60, r_3^F = 6.85$, and $r_4^F = 5.83$ as provided in [50, 58]. Given that we only have N time slots (i.e., transmission opportunities) to send these packets so that they arrive at the receivers before their deadline, the objective of our NC with SDP is to maximize the total distortion reduction for the media sequence.

Fist, we evaluate the scheduling with NC using BIA and the scheduling with NC using SDP for a broadcast scenario with a small number of receivers. In particular, we measure the performances of those algorithms for a broadcast setting in which a Foreman sequence is broadcasted to two receivers R_1 and R_2. For SDP, we use



(a)

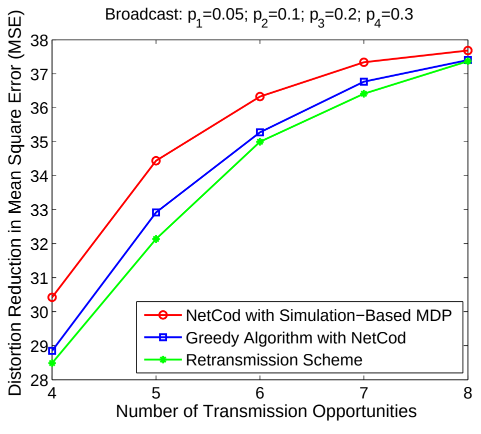


(b)

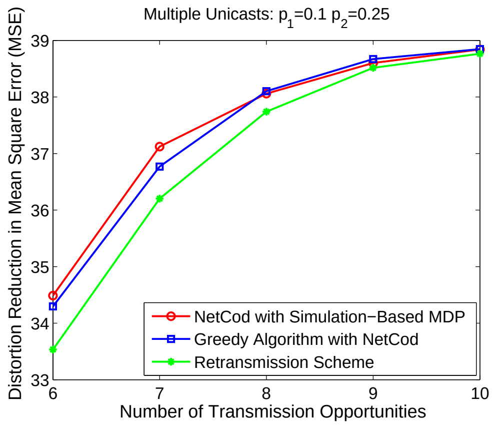
Figure 4.14: Average distortion reduction in MSE of video sequences at each receiver in the concurrent unicast scenario: (a) Distortion versus packet loss probabilities at receivers R_1 and R_2; (b) Distortion reduction versus packet loss probability R_1.

20 and 30 samples per action. As seen from Fig. 4.12, the NC scheme using BIA gives the optimal scheduling policy, leading to the highest distortion reduction. The performance of the NC with SDP gets closer to that of the NC with BIA as we increase the number of samples from 20 to 30. We note that in this setting, there are two receivers in our simulation, so we can define the state transition matrices and reward matrices without much difficulty.

Now we consider for the scenario of more number of receivers in which we apply the NC scheme using SDP. We compare the performance of our NC approach with those of two other algorithms: The ARQ retransmission algorithm without using NC and the greedy algorithm with NC. In the retransmission scheme, the AP sends packets starting from the packet with the largest distortion reduction to the



(a)



(b)

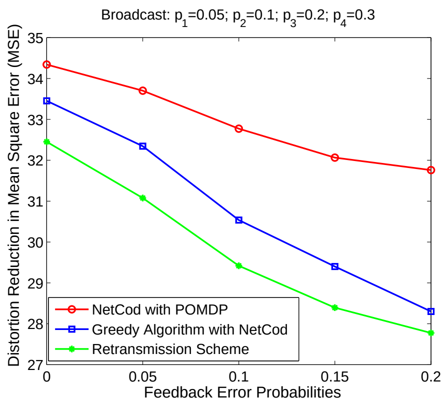
Figure 4.15: Average distortion reduction in MSE of video sequences at each receiver versus the number of transmission opportunities: a) Broadcast to 4 receivers; (b) Concurrent unicast to two receivers.

one with the lowest distortion reduction, i.e., following the order l_1^F, l_2^F, l_3^F, and l_4^F. Each packet is sent until either it is received correctly at the intended receiver(s) or the number of transmissions exceeds N. After N time slots, regardless of whether or not the AP successfully sends all 4 packets, it moves to the next 4 packets (layers) of the next frame. For the greedy algorithm with NC, the AP maintains the set of states and the set of actions similarly as in the MDP method. Those actions includes sending nothing, sending a new packet, sending a lost packet, and sending a NC packet. At each transmission *stage*, it will compute the best action that provides the highest immediate reward instead of computing the policy for the whole time horizon. In fact, this algorithm can be considered as an approximation solution of the MDP problem where the optimization policy is for one stage rather than for the whole time horizon as resulted from BIA.

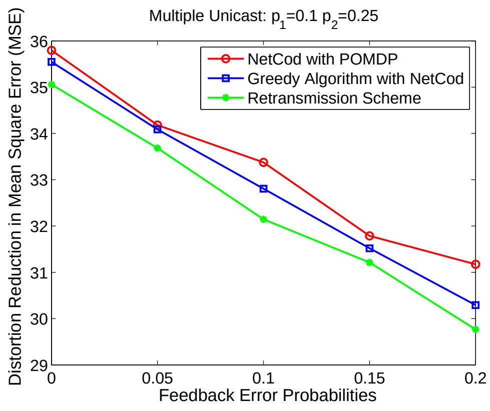
To illustrate the difference between the retransmission approach and greedy algorithm transmission with NC, we consider an example in which the AP broadcasts two packets l_1 and l_2 with their respective normalized rewards 0.7 and 0.3 to three receivers R_1, R_2, and R_3. Assume that at some point, the state of the receivers is $\begin{bmatrix} 1 & 0 \\ 0 & 0 \\ 1 & 0 \end{bmatrix}$, i.e., the first packet l_1 is lost at the receiver R_2. In the next transmission, the retransmission algorithm will retransmit packet l_1. The greedy algorithm with NC, instead, will transmit l_2 because that action provides the largest immediate sum reward.

Fig. 4.13 shows the average distortion reduction as a function of packet loss probabilities in the broadcast setting for three schemes. In Fig. 4.13(a), we vary the packet loss probabilities of all receivers from 5% to 25% and fix the total number of transmission opportunities $N = 5$. The lower distortion amount results in better average video qualities. As seen, the NC scheme with MDP performs the best, followed by the greedy algorithm with NC, then the non-NC retransmission algorithm. For a fixed $N = 5$, the increase in loss rate results in a decrease in throughput. With small delay requirement ($N = 5$), it is becoming critical to schedule the packets optimally to maximize the video qualities at the receivers. When the loss rate is 5% and $N = 5$, the AP has many opportunities to successfully transmit all the packets to receivers, resulting in approximately the maximum distortion reduction. In a second simulation, we simulate for the scenario where packet loss rates of receivers are quite different. In particular, we set packet loss rates for R_2, R_3 and R_4 as $p_2 = 0.1, p_3 = 0.2$, and $p_4 = 0.3$ and vary the loss rate for R_1. The number of transmission opportunities is set to 5. As shown in Fig. 4.13(b), the NC with MDP provides the highest media quality. It is interesting that the performance gap between the MDP scheme and the other two schemes becomes bigger. This means that the MDP always produce a better transmission policy when the packet loss rates vary from a receiver to another.

We now consider the unicast setting in which two receivers R_1 and R_2 want to receive two different sequences Foreman and Coastguard, respectively. We assume that Foreman has three packets l_1^F, l_2^F, and l_3^F with their corresponding distortion reduction in MSE $r_1^F = 14.67, r_2^F = 10.60$, and $r_3^F = 6.85$, and Coastguard has three packets l_1^C, l_2^C, and l_3^C with their corresponding distortion reduction in MSE $r_1^C = 20.84, r_2^C = 15.34$, and $r_3^C = 9.54$. Fig. 4.14(a) shows the distortion reduction as a function of the loss rates for fixed $N = 7$ and the packet loss probabilities at two receivers p_1 and p_2 vary from 0.05 to 0.25. Similarly, with the same N, we fix packet loss rates of R_1 as $p_1 = 0.2$ and vary p_2 from 0.05 to 0.25. As predicted, the smaller packet loss probabilities provides better performance. Again, the NC scheme with simulation-based MDP leads to the largest reward or best video qualities. The performance of the greedy algorithm with NC is very close to that of the MDP non-NC algorithm as shown in both figures. This is plausible as the MDP always selects the packet with largest distortion reduction to send to its intended receiver, resulting in a similar performance to that of the look-ahead algorithm. For the same broadcast and unicast settings, we now examine the scenarios where the loss rates are fixed and the number of transmission opportunities N is varied. Fig.



(a)



(b)

Figure 4.16: Average distortion reduction in MSE of video sequences at each receiver versus the number of transmission opportunities for the case of erroneous feedback: a) Broadcast to 4 receivers; (b) Concurrent unicast to two receivers.

4.15 (a) and (b) shows the rewards as a function of N for three approaches in the broadcast and unicast settings, respectively. As N increases, there are more opportunities to retransmit the lost packets, thus the performances of all three algorithms also increase. Again, the MDP with NC algorithm performs the best, followed by other algorithms. When N increases to some value, the performances of those three schemes converge to a maximum value. This means that there is enough transmission opportunities to send all packets correctly to the receivers. Thus there are no packet losses that causes the reduction in media quality. We note that the NC with MDP converges to the maximum value faster than all other schemes.

We now present the simulation results for the case of erroneous feedback. For the broadcast scenario, the AP broadcasts the Foreman sequence four receivers R_1, R_2, R_3, and R_4 with their corresponding packet loss probabilities $p_1 = 0.05, p_2 = 0.1, p_3 = 0.2$, and $p_4 = 0.3$. For the unicast scenario, the AP concurrently unicasts the Coastguard and Foreman to two receivers R_1 and R_2 with packet loss probabilities $p_1 = 0.1$ and $p_2 = 0.25$, respectively. In both settings, we vary the feedback error probabilities from all receivers to the AP from 0% to 20% and fix the number of transmission opportunities. As seen from Fig. 4.16, when the feedback error probabilities increase, the transmission performance of the retransmission method reduces significantly, while the NC framwork with POMDP maintains the high video quality. We note that, when the feedback error probability equals to 0%, the feedback channel is perfect and our scheduling problem reduces to the scheduling with MDP above.

4.8 Summary

In this chapter we have proposed a NC based scheduling policy at an AP that optimizes the multimedia transmission in both broadcast and concurrent unicast settings. In particular, our contributions include (a) a framework for increasing the bandwidth efficiency of broadcast and unicast sessions in a wireless network based on NC techniques; (b) an optimized scheduling algorithm based on the Markov Decision Process (MDP) to maximize the quality of multimedia applications; (c) simulation-based algorithms for solving the large MDP and POMDP problems. The results demonstrate the advantages of our approach over the current transmission techniques.

Chapter 5 – RANDOM NETWORK CODING TECHNIQUES

D. Nguyen, T. Tran, and T. Nguyen, *Hybrid ARQ-Random Network Coding for Wireless Media Streaming*, in IEEE-ICCE, Vietnam, 2008.

5.1 Introduction

In previous chapters, we consider XOR NC in which the encoding and decoding operations are the XOR operations-the operations in Finite Field $GF(2)$. RNC, on the other hand, uses operations in Finite Field $GF(2^q)$ with $q > 2$. As mentioned in the Chapter 2, using RNC, data packets and coefficients are presented as elements of $GF(2^q)$. RNC can achieve a throughput improvement, in most cases, similar to or even better than XOR NC. As seen in the example in Fig. 5.2, the throughput improvement can be the same as XOR NC. The advantage of RNC is that the AP does not have to know which packet is lost at which receiver. In other words, it may not need to use full feedback for every packet as in the XOR NC scheme.

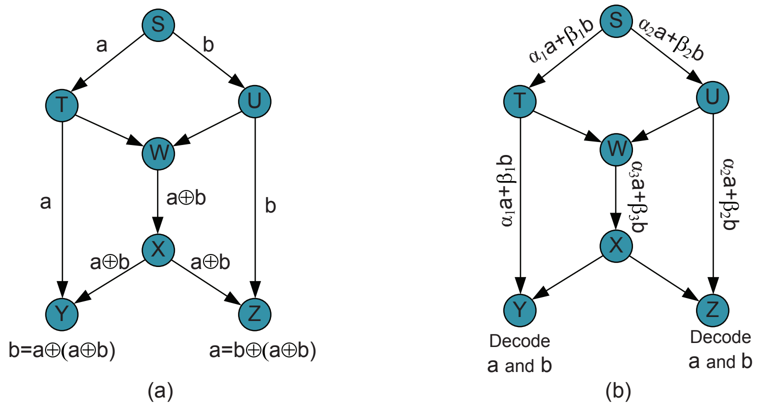


Figure 5.1: An example of RNC: RNC (b) achieves the same throughput as XOR NC (a); α_i and β_i (i={1,2,3}) are random coefficients in a Finite Field $GF(2^q)$, $(q > 2)$.

Generally, to encode K packets $P_1, P_2, ..., P_K$ using RNC, K random coefficients

Timeslot	1	2	3
Packet sent	a	b	a⊕b
R_1	x	o	o
R_2	o	x	o

(a)

Timeslot	1	2	3
Packet sent	$\alpha_1 a+\beta_1 b$	$\alpha_2 a+\beta_2 b$	$\alpha_3 a+\beta_3 b$
R_1	x	o	o
R_2	o	x	o

(b)

Figure 5.2: An example of RNC for multi-user single-hop networks: RNC (b) achieves the same throughput as XOR NC (a); α_i and β_i (i={1,2,3}) are random coefficients in Finite Field $GF(2^q)$, $(q > 2)$.

$\alpha_i (1 \leq i \leq K)$ are generated, and the coded packet is formed as

$$CP = \sum_{i=1}^{K} \alpha_i P_i. \tag{5.1}$$

For a receiver to be able to decode those K original packets, it must receive at least K independent coded packets. This means that K sets of random coefficients must be independent. If finite field $GF(2^8)$ is used, the probability of independency for small K (e.g. about 10) is close to 100%. Those coefficients must be sent together with the coded packet by inserting them into the packet header. This results in a transmission overhead. If l is the packet length (in a number of symbols in the finite field), then this overhead is $\frac{n}{l}(\%)$. In practice, because the packet length is normally much larger than the number of coefficients, the overhead can be ignored. For instance, with $q = 2^8$, $K = 10$ (packets), and $l = 1000$ (bytes), the overhead translates to roughly only 1% of the data. RNC has been recognized to be a very efficient technique for information dissemination in wireless sensor or ad-hoc networks and peer-to-peer networks where link failures are often and the network topology is difficult to obtain.

5.2 RNC for Wireless Media Transmissions

We again consider the wireless media transmission from the AP to multiple receivers as in Chapter 3 and Chapter 4. Suppose there are N transmission opportunities to transmit K media packets $P_1, P_2, ..., P_K$ to M receivers so that they can arrive at the clients before the deadline. The AP employs our RNC technique to encode those packets and send to the client. At the receivers, an appropriate decoding process is used to recover the original media packets. We have following RNC schemes:

5.2.1 Uniform Random Network Coding (URNC)

In this scheme, the AP linearly combines K media packets P_i into coded packet CP as in Equation 5.1 and then broadcasts those coded packets together with their coefficients to the receivers. There are N transmission opportunities, so N coded packets are formed and transmitted. A receiver can decode the original packets if it receives at least K independent coded packets. Clearly, N should be at least equal to K so that the decodability at the receivers is possible. Because of the packet losses, N is normally greater than K. Fig. 5.3(a) shows how URNC encodes 3 dependent media packets P_1, P_2, and P_3 into five coded packets $CP_1, CP_2, CP3, CP_4$, and CP_5 which are then transmitted to the receivers. A receiver must receive at least 3 independent coded packets in order to decode the original packets.

The drawback of URNC is that it does not consider the unequal importance of the media packets. Because, the media packets have different levels of importance in contributing to the overall media quality, it is not efficient to combine all packets into the same coded packets. When packet loss probability is high so that a receiver does not receive enough number of packets, all correctly-received coded packets are unable to be decoded, resulting in highly distorted video. In the example in Fig. 5.3(a), if a receiver receives 2 coded packets, say CP_1 and CP_2 then it is unable to decode any packet because the system of equations formed by CP_1 and CP_2 has three unknowns, thus being unsolvable. In addition, a receiver can start playback only after K original packets being decoded. The PRNC scheme can alleviate this problem.

5.2.2 Prioritized Random Network Coding (PRNC)

PRNC is an improved version of URNC by giving encoding and transmission priority to important packets. PRNC involves three procedures: (1) hierarchically grouping the set of packets into different classes based on packets' importance; (2) applying RNC for each class using a common finite field; and (3) scheduling to transmit coded packets resulted from each class. More specifically, the set of K

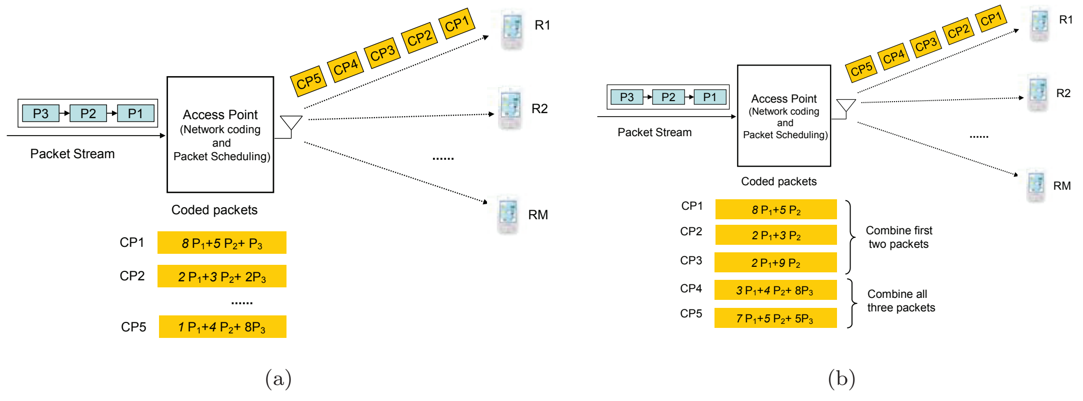


Figure 5.3: An example of wireless media broadcast with RNC; three media packets are broadcast to M receivers: (a) URNC: coded packets are random combinations of all three packets; (b) PRNC: the first three coded packets are random combinations of the first two media packets and the last two coded packets are random combinations of all three media packets.

packets $\{P_1, P_2, ..., P_K\}$ are classified into L classes: $C_1 = \{P_1, P_2, ..., P_{N_1}\}$, $C_2 = \{P_1, P_2, ..., P_{N_2}\}$, ..., and $C_L = \{P_1, P_2, ..., P_{N_L}\}$, where $N_1 < N_2 < ... < N_L = K$. This means that $C_1 \subset C_2 \subset .... \subset C_L$ (hierarchical classification). We propose a heuristic scheduling policy for transmissions of coded packets from classes; with each algorithm, we consider two scenarios of the availability of the channel feedbacks. In particular, we consider the scenario without channel feedback and and the scenario with channel feedback. As will be shown, the coded packets from high-priority classes have higher probability to be recovered earlier than those of low-priority classes.

For the sake of simplicity, we investigate a basic case with $L = 2$; the set of packets is classified into two classes. The scheduling algorithm with $L > 2$ can be inferred in the same manner. There are 3 packets P_1, P_2, and P_3 which are allowed to be transmitted within a maximum of 5 time slots. This set of packets can be grouped into two classes $C_1 = \{P_1, P_2\}$ and $C_2 = \{P_1, P_2, P_3\}$. Note that the important packets are designed to be present in every class. Applying RNC for the first class results in coded packets, which are assigned to be transmitted within the first 3 time slots. Similarly, coded packets from C_2 are transmitted during the last 2 time slots. Now assume that the channel experiences high packet error rate, and there are 3 time slots among 5 in which, the packets are lost at a receiver. Using PRNC, if receiver receives 2 coded packets during the first 3 time slots, it would be able to decode the 2 packets in C_1 whereas if using URNC, a receiver does not have any chance of decoding 3 media packets because it only receives 2 out of 5 coded packets (Fig. 5.3b). Thus, PRNC can be used in a high loss rate environment.

5.2.3 Scheduling Coded Packets

No Channel Feedback: The transmissions without channel feedback is suitable for the case of high data rate transmissions or for the broadcast with a large number of receivers. In high bit rate transmissions, using feedback may render high transmission delay. In the broadcast with a large number of receivers, using feedback may create the "feedback implosion problem" when many receivers send feedback to the AP at the same time. From error protection point view, NC works similarly to FEC techniques which transmit redundancy to help recover lost packets across different receivers. Using URNC, the AP forms N coded packets by combining all K media packets to broadcast to all receivers. URNC does not require any special scheduling algorithm. In particular, the AP combines media packets in class C_1 into coded packets that are transmitted in the first T_1 time slots and combines media packets in class C_2 into coded packets that are transmitted in the remaining T_2 time slots ($T_1 + T_2 = N$). One special case is that $T_2 = 0$. This means the AP dedicates all N time slots to transmit all coded packets in the first class and ignore the second class which has less important packets. In the example above, the AP can use 5 time slots to transmit coded packets of C_1. This way of transmission produces a certain level of media quality under high packet loss rates, at an expense that it never gives full media quality.

With Channel Feedback: We consider two cases: full-feedback and limited-feedback. In full-feedback, a receiver sends feedbacks for all packets it has received while in limited feedback a receiver sends one feedback for several packets together. Clearly, in term of media quality, URNC would not gain from feedback because N coded packets are formed and transmitted in N time slots similarly to the case without feedback. However, in term of transmission overhead, i.e., the number of transmissions per segment group, the AP can stop the transmission when knowing that all receivers receive all media packets. For PRNC, as soon as the AP knows that all receivers can decode the media packets in the first class, it starts transmitting coded packets in the second class. For PRNC, the AP also stops transmissions when it knows all receivers receive all L packets. We note that there can be a number of sophisticated scheduling algorithms in the case of full-feedback. For instance, one can state the scheduling problem as a Markov decision process problem and use the Dynamic Programming method to find an optimal algorithm.

5.2.4 Integer Programming Formulation for PRNC

So far we have described how PRNC works intuitively, but not how to set the parameters optimally that achieves highest visual video quality under a given channel condition. Specifically, we would like to know how to group the coded packets into classes and schedule the transmissions for packets from each class in order to obtain the best possible video quality at the receivers. The answers to these questions can be formulated as an integer programming problem. Let us consider the broadcast scenario for PRNC.

There are three parameters to consider: the number of classes L, the size of each class $N_1, N_2, ...,$ and N_K, and the number of time slots T_i that are allowed to transmit packets from class C_i. These 3 parameters specify completely the scheduling algorithm. Note that we do not need to specify which packets belong to which class as this is done implicitly. In particular, if C_1 has size of N_1, then the first N_1 packets among L packets will belong C_1. Next if C_2 has N_2 packets, then the first N_2 packets among the L packets belong to C_2, and so on. That said, we can formulate the optimal scheduling problem as an integer programming problem as

$$\begin{aligned} \text{minimize} \quad & \theta_{PRNC}, \\ \text{subject to} \quad & N_L = K, \\ & \textstyle\sum_{i=1}^{L} T_i = N, \\ & N_i > N_{i-1} > 0, T_i > 0, K > 0, \end{aligned}$$

where θ_{PRNC} denotes the average media quality as defined in Equation 4.12, and is a function of PERs at receivers and the distortion contributions of packets. Having formulated the problem, we note that finding the solution for a general integer programming problem is hard. Although heuristic algorithms can be employed, when the number of packets K is small (< 10), the problem can be solved by the exhaustive search method.

5.2.5 Remarks On Overhead

There are two types of overheads in RNC: transmission overhead and delay overhead. The earlier is the overhead in which random coefficients must be transmitted with the coded packets and the later is the playback time delay because the client has to wait until getting enough number of coded packets for decoding and then takes time to decode the original packets. To combine K packets, K coefficients must be used to produce each coded packet. As a result, the transmission overhead depends on the length of the packet being combined. In practice, coefficients are normally elements of $GF(2^8)$ (i.e., one byte) and the packet has a length of

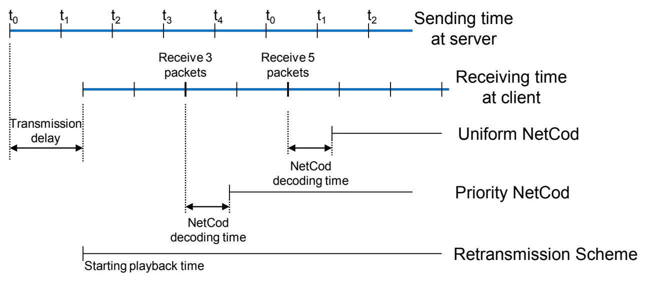


Figure 5.4: The delay in the streaming system using three different schemes: URNC has the largest delay while the retransmission scheme has the smallest delay.

more than thousand bytes, so this overhead can be ignored. The waiting time to get enough number of coded packets are unavoidable for RNC. For example with $K = 5$ and $L = 3$, the client has to wait until getting enough 5 packets and 3 packets if using URNC and PRNC, respectively, before starting decoding. The retransmission scheme or Round-Robin transmission scheme, however, do not have this delay. The decoding is a process of solving a system of equations. The number of unknowns (in the system of equations) is K and L, respectively for URNC and PRNC. If the Gaussian elimination method is used for decoding, the time complexities are $O(K^3)$ and $O(L^3)$. If the packet length is of l symbols, then those complexities are $O(lL^3)$ and $O(lK^3)$. We note that K is not very large since a media frame typically has up to more than 10 layers while l is in the order of hundreds or more than one thousand.

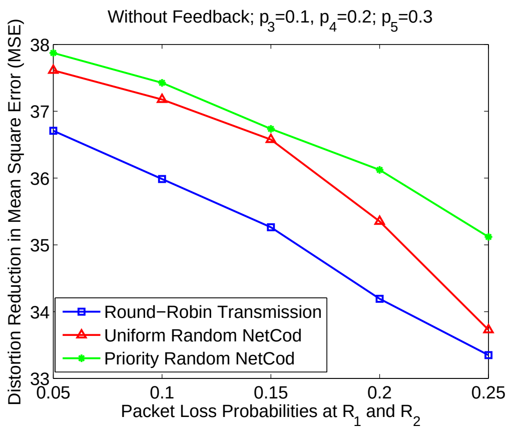
Fig. 5.4 introduces the delay of the media streaming playback with different NC schemes. The server begins transmission at time t_0. After the transmission delay including the propagation delay, the first packet is received at the client.

If using regular retransmission scheme without using NC, the receiver can start playing back immediately. However, if using URNC and PRNC, the receiver must receiver enough number of coded packets for decoding (e.g., 5 packets for URNC and 3 packets for PRNC) and then spends time to decode the source packets before playing back.

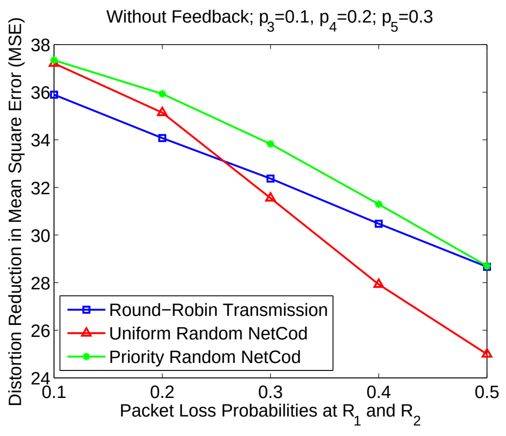
5.3 Simulation Results

We consider the broadcast setting in which the AP broadcasts a Foreman sequence to five receivers R_1, R_2, R_3, R_4, and R_5. For each transmission round of N time slots, the AP transmits five packets of the sequence, P_1, P_2, P_3, P_4, and P_5 with their corresponding distortion reduction in MSE $14.67, 10.60, 6.85, 5.83$, and 4.30.

For the case of non-feedback, we compare the URNC and PRNC performance with that of the Round-Robin transmission scheme. The Round-Robin transmission scheme transmits packet P_1, P_2, P_3, P_4, and P_5 within N time slots in the decreasing order of the distortion reduction amount. After finishing the transmission of those packets for the first time, the AP turns back to retransmit those packets in the same order if the transmission opportunities are still available. For example, if $N = 7$, then the transmission order of the Round-Robin transmission scheme is $P_1, P_2, P_3, P_4, P_5, P_1$, and P_2. Because there are no feedbacks from the receivers, the AP tries to transmit as many times as it can so that a lost packet, if any, can be recovered with some probability. The URNC scheme combines all 5 packets into 7 coded packets $CP_1, CP_2, ...$, and CP_7 and transmits to the re-



(a)

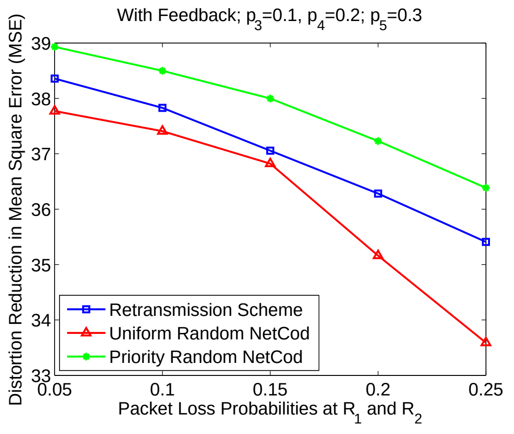


(b)

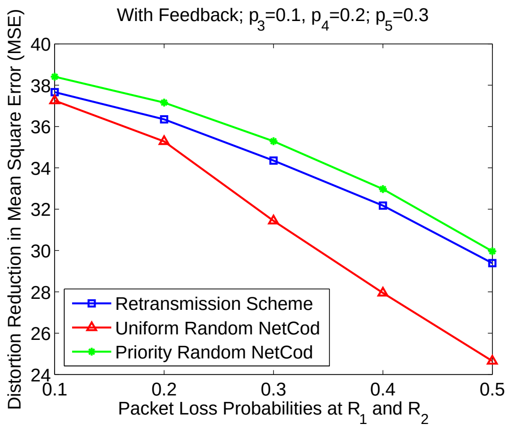
Figure 5.5: Average distortion reduction in MSE of video sequences at each receiver versus packet loss probabilities in the non-feedback transmissions using RNC schemes: (a) Broadcast to five receivers; (b) The case of high packet loss probabilities.

ceivers within 7 time slots. Any receiver receiving equal or more than 5 coded packet can decode those 5 original packets, otherwise, they are undecodable. The PRNC scheme heuristically takes the first, most important 3 packets, combines and transmits them within the first 5 time slots. For the remaining 2 time slots, PRNC transmits the coded packets which are combinations of all 5 packets. Fig. 5.5 shows the transmission performances of three schemes. We see that for both cases of low packet loss probability and high packet loss probability, the performance of PRNC is better than those of other two schemes, while the Round-Robin performs worst.

Similarly, for the transmissions with feedback, we compare the the URNC and PRNC performance with that of the retransmission scheme. The retransmission



(a)



(b)

Figure 5.6: Average distortion reduction in MSE of video sequences at each receiver versus packet loss probabilities in the transmissions with feedback using RNC schemes: (a) Broadcast to five receivers; (b) the case of high packet loss probabilities.

scheme using feedback transmits packet P_1, P_2, P_3, P_4, and P_5 within N transmission time slots in the decreasing order of the distortion reduction amount. Different from the Round-Robin scheme, the retransmission scheme *retransmits* any lost packet immediately until its intended receiver receives it correctly. The URNC scheme does not need feedback to schedule their transmissions. However, it can stop transmission if it knows that all receivers have decoded the original packet, helping reducing the total number of transmissions. For the PRNC scheme, the coded packets from the first class, i.e, the first three most important packets, are transmitted until all 5 receivers are able to decode them. After that, the combinations of all 5 packets are transmitted in the remaining time slots. Fig. 5.6 shows the transmission performances of three schemes. We see that for both cases of low packet loss probabilities and high packet loss probability, the performance of PRNC is better than those of other two schemes, while the retransmission scheme performs worst.

5.4 On Encoding Rate and Decoding Rate of Practical RNC

In this section, we present a practical RNC implementation in C language and we measure the encoding and decoding delay using normal desktop computer. We use finite field $GF(2^8)$ with primitive polynomial $p(x) = x^8 + x^4 + x^3 + x + 1$ for our operations of NC. We consider how the operations in this field work on the data packets. First of all, we examine the encoding and decoding processes for regular data packets.

5.4.1 Encoding and Decoding Data Packets

We consider RNC over the finite field $GF(2^8)$ Because, each data byte, i.e., 8 bits of information, is an element in $GF(2^8)$, each packet which is presented as an array of bytes, is a sequence of elements in the field. Our encoding and decoding use addition, substraction, multiplication, division operations on the byte basic. To illustrate how encoding and decoding works, we assume that there are three packets P_1, P_2, and P_3. Those packets have the same size of N bytes, i.e., 3 arrays of N elements in $GF(2^8)$. For each encoding, three random coefficients-elements of $GF(2^8)$, are generated. As seen in the Fig. 5.7(a), a set of random coefficients,

a_1, b_1, c_1 is used to produce encoded packet CP_1. Similarly, a_2, b_2, c_2 and a_3, b_3, c_3 are generated to create encoded packet CP_2 and CP_3, respectively. We note that each coefficient is an element of $GF(2^8)$, so it is presented as a byte. The encoding is a byte-wise operation. This means that the first byte of CP_1 is created by encoding three fist bytes of P_1, P_2, and P_3; the second byte of CP_1 is created by encoding three second bytes of P_1, P_2, and P_3; and so on. In this example, we need to encode three packets, the receiver needs at least three coded packets to decode the original packets. In practice, because some coded packets can be lost during their transmissions, the sender has to generate more than three coded packets. As mentioned earlier, because the probability of independence of those random coefficients is close to 100%, the probability of being able to decode the original packets is close to 100%. We note that the coefficients must be sent together with the coded data by enclosing them in the coded packet's header.

We now consider the decoding process (Fig. 5.7(b)). We assume that a receiver has received three coded packets CP_1, CP_2, and CP_3 with three sets of independent coefficients, $\{a_1, b_1, c_1\}$, $\{a_2, b_2, c_2\}$ and $\{a_3, b_3, c_3\}$. The receiver has to decode the original data based on those received information. This can be done by solving systems of equations using the Gaussian Elimination method. Let us consider how it solves the first three bytes of three packets. We denote the first byte of P_1 as x, P_2 as y and P_3 as z, that we want to recover; the first byte of coded packet CP_1 as d_1, CP_2 as d_2, and CP_3 as d_3. We then have a system of equations with three

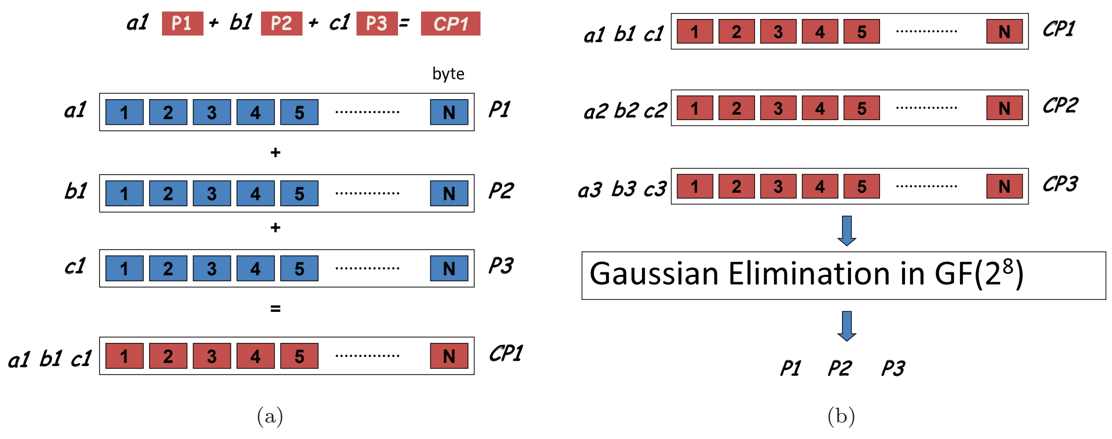


Figure 5.7: An example of random network encoding and decoding: (a) three packets are presented as vector of data bytes, each byte is an element of $GF(2^8)$. a_i, b_i, and c_i are random coefficients in the field; (b) when the receiver receives three coded packets with three sets of independent coefficients, it uses the Gaussian Elimination method to solve the original data.

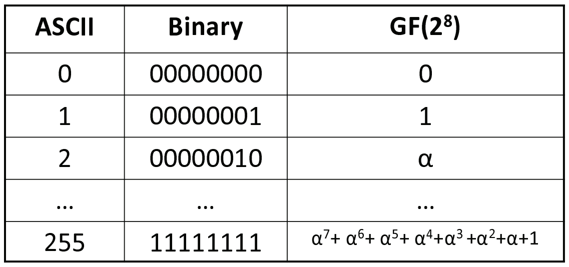
unknown as follows:

$$a_1x + b_1y + c_1z = d_1 \tag{5.2}$$

$$a_2x + b_2y + c_2z = d_2 \tag{5.3}$$

$$a_3x + b_3y + c_3z = d_3 \tag{5.4}$$

Solving this system of equations gives us the first three bytes of three original packets. Because there are N bytes for each packet, N systems of equations need to be solved to recover all three packets P_1, P_2, and P_3.

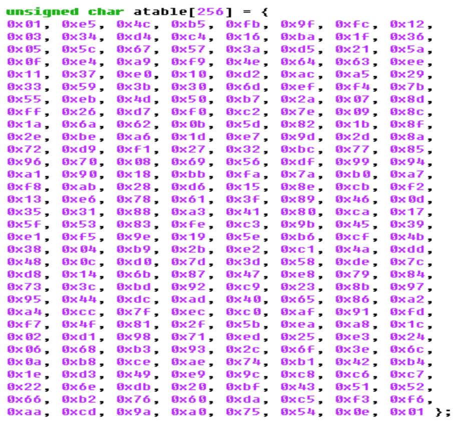


ASCII	Binary	GF(2^8)
0	00000000	0
1	00000001	1
2	00000010	α
…	…	…
255	11111111	$\alpha^7+\alpha^6+\alpha^5+\alpha^4+\alpha^3+\alpha^2+\alpha+1$

Figure 5.8: Byte presentation of a packet used for RNC. Each byte is an element of $GF(2^8)$.

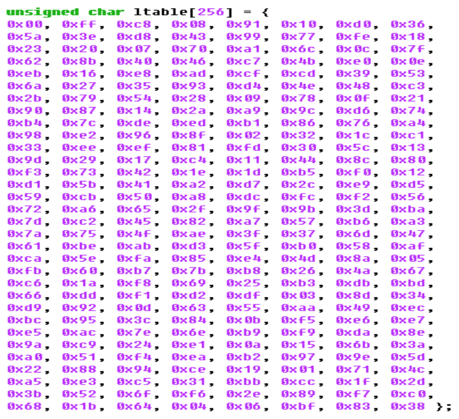
5.4.2 Encoding and Decoding Implementation in C

Because, the encoding and decoding uses operations in finite field $GF(2^8)$, we need to implement all the basic operations including addition, substraction, multiplication, and division. We will show step-by-step how the program using C language works for those encoding and decoding processes. First of all, we introduce how to present a packet as a vector of elements in $GF(2^8)$. Each element of $GF(2^8)$ can be represented as polynomials of degree strictly less than 8 over the basic field $GF(2)$. We take polynomial $p(x) = x^8 + x^4 + x^3 + x + 1$ over $GF(2)$ as a primitive polynomial of $GF(2^8)$. Let α be the root of $p(x) = 0$, then any element of $GF(2^8)$ can be presented as an exponential presentation of α or as a polynomial of α. For example, α^0 is 1 in the polynomial presentation and **00000001** in binary presentation, or α^8 is $\alpha^4 + \alpha^3 + \alpha + 1$ in the polynomial presentation and **00011011** in the binary presentation. Fig. 5.8 shows the presentation of an element of $GF(2^8)$ in different forms. Using the polynomial presentation, the operations in $GF(2^8)$ can be implemented over polynomials, i.e., polynomial addition,



```
unsigned char atable[256] = {
0x01, 0xe5, 0x4c, 0xb5, 0xfb, 0x9f, 0xfc, 0x12,
0x03, 0x34, 0xd4, 0xc4, 0x16, 0xba, 0x1f, 0x36,
0x05, 0x5c, 0x67, 0x57, 0x3a, 0xd5, 0x21, 0x5a,
0x0f, 0xe4, 0xa9, 0xf9, 0x4e, 0x64, 0x63, 0xee,
0x11, 0x37, 0xe0, 0x10, 0xd2, 0xac, 0xa5, 0x29,
0x33, 0x59, 0x3b, 0x30, 0x6d, 0xef, 0xf4, 0x7b,
0x55, 0xeb, 0x4d, 0x50, 0xb7, 0x2a, 0x07, 0x8d,
0xff, 0x26, 0xd7, 0xf0, 0xc2, 0x7e, 0x09, 0x8c,
0x1a, 0x6a, 0x62, 0x0b, 0x5d, 0x82, 0x1b, 0x8f,
0x2e, 0xbe, 0xa6, 0x1d, 0xe7, 0x9d, 0x2d, 0x8a,
0x72, 0xd9, 0xf1, 0x27, 0x32, 0xbc, 0x77, 0x85,
0x96, 0x70, 0x08, 0x69, 0x56, 0xdf, 0x99, 0x94,
0xa1, 0x90, 0x18, 0xbb, 0xfa, 0x7a, 0xb0, 0xa7,
0xf8, 0xab, 0x28, 0xd6, 0x15, 0x8e, 0xcb, 0xf2,
0x13, 0xe6, 0x78, 0x61, 0x3f, 0x89, 0x46, 0x0d,
0x35, 0x31, 0x88, 0xa3, 0x41, 0x80, 0xca, 0x17,
0x5f, 0x53, 0x83, 0xfe, 0xc3, 0x9b, 0x45, 0x39,
0xe1, 0xf5, 0x9e, 0x19, 0x5e, 0xb6, 0xcf, 0x4b,
0x38, 0x04, 0xb9, 0x2b, 0xe2, 0xc1, 0x4a, 0xdd,
0x48, 0x0c, 0xd0, 0x7d, 0x3d, 0x58, 0xde, 0x7c,
0xd8, 0x14, 0x6b, 0x87, 0x47, 0xe8, 0x79, 0x84,
0x73, 0x3c, 0xbd, 0x92, 0xc9, 0x23, 0x8b, 0x97,
0x95, 0x44, 0xdc, 0xad, 0x40, 0x65, 0x86, 0xa2,
0xa4, 0xcc, 0x7f, 0xec, 0xc0, 0xaf, 0x91, 0xfd,
0xf7, 0x4f, 0x81, 0x2f, 0x5b, 0xea, 0xa8, 0x1c,
0x02, 0xd1, 0x98, 0x71, 0xed, 0x25, 0xe3, 0x24,
0x06, 0x68, 0xb3, 0x93, 0x2c, 0x6f, 0x3e, 0x6c,
0x0a, 0xb8, 0xce, 0xae, 0x74, 0xb1, 0x42, 0xb4,
0x1e, 0xd3, 0x49, 0xe9, 0x9c, 0xc8, 0xc6, 0xc7,
0x22, 0x6e, 0xdb, 0x20, 0xbf, 0x43, 0x51, 0x52,
0x66, 0xb2, 0x76, 0x60, 0xda, 0xc5, 0xf3, 0xf6,
0xaa, 0xcd, 0x9a, 0xa0, 0x75, 0x54, 0x0e, 0x01 };
```

(a)



```
unsigned char ltable[256] = {
0x00, 0xff, 0xc8, 0x08, 0x91, 0x10, 0xd0, 0x36,
0x5a, 0x3e, 0xd8, 0x43, 0x99, 0x77, 0xfe, 0x18,
0x23, 0x20, 0x07, 0x70, 0xa1, 0x6c, 0x0c, 0x7f,
0x62, 0x8b, 0x40, 0x46, 0xc7, 0x4b, 0xe0, 0x0e,
0xeb, 0x16, 0xe8, 0xad, 0xcf, 0xcd, 0x39, 0x53,
0x6a, 0x27, 0x35, 0x93, 0xd4, 0x4e, 0x48, 0xc3,
0x2b, 0x79, 0x54, 0x28, 0x09, 0x78, 0x0f, 0x21,
0x90, 0x87, 0x14, 0x2a, 0xa9, 0x9c, 0xd6, 0x74,
0xb4, 0x7c, 0xde, 0xed, 0xb1, 0x86, 0x76, 0xa4,
0x98, 0xe2, 0x96, 0x8f, 0x02, 0x32, 0x1c, 0xc1,
0x33, 0xee, 0xef, 0x81, 0xfd, 0x30, 0x5c, 0x13,
0x9d, 0x29, 0x17, 0xc4, 0x11, 0x44, 0x8c, 0x80,
0xf3, 0x73, 0x42, 0x1e, 0x1d, 0xb5, 0xf0, 0x12,
0xd1, 0x5b, 0x41, 0xa2, 0xd7, 0x2c, 0xe9, 0xd5,
0x59, 0xcb, 0x50, 0xa8, 0xdc, 0xfc, 0xf2, 0x56,
0x72, 0xa6, 0x65, 0x2f, 0x9f, 0x9b, 0x3d, 0xba,
0x7d, 0xc2, 0x45, 0x82, 0xa7, 0x57, 0xb6, 0xa3,
0x7a, 0x75, 0x4f, 0xae, 0x3f, 0x37, 0x6d, 0x47,
0x61, 0xbe, 0xab, 0xd3, 0x5f, 0xb0, 0x58, 0xaf,
0xca, 0x5e, 0xfa, 0x85, 0xe4, 0x4d, 0x8a, 0x05,
0xfb, 0x60, 0xb7, 0x7b, 0xb8, 0x26, 0x4a, 0x67,
0xc6, 0x1a, 0xf8, 0x69, 0x25, 0xb3, 0xdb, 0xbd,
0x66, 0xdd, 0xf1, 0xd2, 0xdf, 0x03, 0x8d, 0x34,
0xd9, 0x92, 0x0d, 0x63, 0x55, 0xaa, 0x49, 0xec,
0xbc, 0x95, 0x3c, 0x84, 0x0b, 0xf5, 0xe6, 0xe7,
0xe5, 0xac, 0x7e, 0x6e, 0xb9, 0xf9, 0xda, 0x8e,
0x9a, 0xc9, 0x24, 0xe1, 0x0a, 0x15, 0x6b, 0x3a,
0xa0, 0x51, 0xf4, 0xea, 0xb2, 0x97, 0x9e, 0x5d,
0x22, 0x88, 0x94, 0xce, 0x19, 0x01, 0x71, 0x4c,
0xa5, 0xe3, 0xc5, 0x31, 0xbb, 0xcc, 0x1f, 0x2d,
0x3b, 0x52, 0x6f, 0xf6, 0x2e, 0x89, 0xf7, 0xc0,
0x68, 0x1b, 0x64, 0x04, 0x06, 0xbf, 0x83, 0x38 };
```

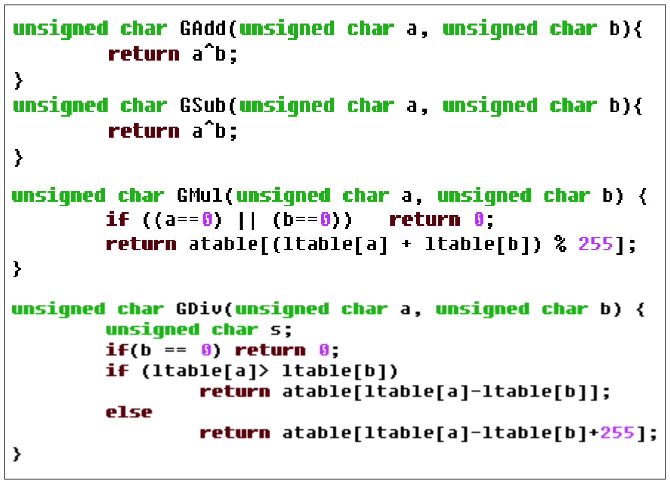
(b)

Figure 5.9: Elements of $GF(2^8)$ is presented as an array in the program.

substraction, multiplication, and division.

As introduced in [43], performing the polynomial addition and substraction has the time complexity $O(n)$, where n is the degree of the polynomial (in our case, $n = 7$). The multiplication and division, however, have the complexity of $O(n^2)$ and $O(n^3)$, respectively, in which the division requires the implementation of the Extended Euclidean algorithm. In the C implementation, the addition and substraction are simply done by the XOR operation, i.e., $a \wedge b$ for $a + b$ or $a - b$, where a, b are elements of $GF(2^8)$ as seen in Fig. 5.10.

For the multiplication and division, one efficient way is using Rijndael's finite field $GF(2^8)$ with lookup table [70]. We consider a Rijndael's finite field with primitive polynomial $p(x) = x^8 + x^4 + x^3 + x + 1$. Under this field, each element of the field is presented as an item in a table of 256 entries. Rijndael algorithm uses one element as a generator to generate all 256 members of the field. This



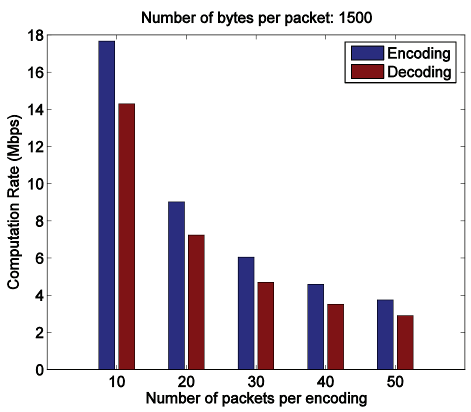
```c
unsigned char GAdd(unsigned char a, unsigned char b){
        return a^b;
}
unsigned char GSub(unsigned char a, unsigned char b){
        return a^b;
}
unsigned char GMul(unsigned char a, unsigned char b) {
        if ((a==0) || (b==0))   return 0;
        return atable[(ltable[a] + ltable[b]) % 255];
}
unsigned char GDiv(unsigned char a, unsigned char b) {
        unsigned char s;
        if(b == 0) return 0;
        if (ltable[a]> ltable[b])
                return atable[ltable[a]-ltable[b]];
        else
                return atable[ltable[a]-ltable[b]+255];
}
```

Figure 5.10: Functions in C to implement operations in $GF(2^8)$: Addition (GAdd), Substraction (GSub), Multiplication (GMul), and Division (GDiv).

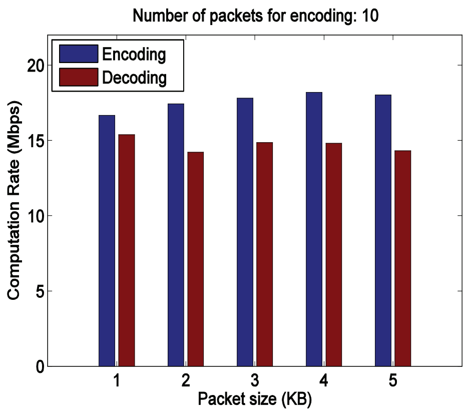
is done by multiplying the generating element with itself for some times. To implement the fast multiplication and division, we first need to construct the tables called logarithm table and anti-logarithm table. Each entry in the logarithm table is the number of times for multiplying the generator while anti-logarithm table is the reversion of the logarithm table. To clarify this, consider the C multiplication function. The logarithm table is generated by the generator 229 or $0xe5$ in hexadecimal and is stored as array *ltable*, while the anti-logarithm is stored as array *atable*. An element $ltable[a]$ in the table (or array) gives us the number of times for multiplying 229. So to multiply a with b we use the *ltable* to find out the exponent presentation of those numbers and the multiplication the addition as $ltable[ab] = ltable[a] + ltable[b]$. Now we need to convert back from the exponent presentation into the decimal by using the anti-logarithm table. In general, using two tables, we can multiply two numbers a and b as $ab = atable[ltable[a]+ltable[b]]$. It is straight forward to divide a by b as $a/b = atable[ltable[a] - ltable[b]]$.

5.4.3 Measuring Encoding and Decoding Time Delay

We are interested in measuring the encoding rate and decoding rate of RNC. We have learned that the encoding requires addition and multiplication in $GF(2^8)$, while the decoding requires addition, multiplication, and division. In the first implementation, we set the packet size of 1500 bytes and the number of packets per encoding of $10, 20, 30, 40$, and 50. Fig. 5.11(a) shows the computation rates of encoding and decoding operation versus the number of packets per encoding. We see that for the same packet size the encoding time is less than the decoding time but they are not much different. For example, with number of packets per encoding 20, the computation rates are 8.5 Mbps and 7.8 Mbps for encoding and decoding, respectively. This is reasonable because using the table-lookup method, the multiplication time complexity and division time complexity are quite similar. Importantly, when the number of packets per encoding increases, the encoding rate and the decoding rate decrease. Fig. 5.11(a) presents the encoding rate and the decoding rate versus the packets sizes with a fixed number of packets per encoding. There is a slight increase in the rates when the packet size increases. From those measurements, it is necessary to design an appropriate coding structure so that the data transmission speed matches with the encoding/decoding speeds, especially for realtime applications using NC.



(a)



(b)

Figure 5.11: Computation rates of random encoding and decoding.

5.5 Summary

We have considered RNC techniques. Our application of RNC to wireless media broadcast shows that it can improve the performance of the media transmissions under various cases of feedbacks. We also show the detailed implementation of RNC in C language in which the table-lookup method for Rijndael's finite field is used to implement efficient operations in $GF(2^8)$. Our implementation shows that the encoding time and the decoding time are not much different and if the number of packet per encoding increases then the delay will increase. This phenomena may effect the performance of realtime applications like media streaming ones.

Chapter 6 – CONCLUSION AND FUTURE WORK

We have presented NC schemes for multi-user single-hop wireless networks - a last mile of the end-to-end communication from the Internet to mobile users. Our proposed scheme is potentially applied in WiFi, WiMAX, or Cellular networks to enhance the overall end-to-end data transmission performance.

In Chapter 3, we have considered a NC scheme for a multi-user wireless broadcast setting in which the AP combines distinct lost packets of different wireless receivers using XOR operations to enhance the bandwidth utilization. We provide detailed theoretical and simulation results of using NC and compare with other traditional transmission schemes in a number of broadcast scenarios.

We provide a joint NC and transmission scheduling scheme using MDP for multimedia streaming to multiple wireless users in Chapter 4. The joint problem is formulated as an MPD abstract so that the solution of MDP will provide an off-line optimal transmission policy, which decides a packet or a combined packet to be transmitted at which time slot in order to produce the best media quality at the receivers. In addition, we also provide a simulation-based dynamic programming algorithm to address the problem of large MDP when the number of receivers increases, and the problem of partial observation of the receivers' states when the feedback from the receiver is in error.

Finally, in Chapter 5, we present RNC techniques which use random combinations for encoding. The prioritized encoding scheme is employed for scheduling the media packets to multiple wireless users. We consider our NC approaches under different feedback scenarios, i.e., full feedback and limited feedback. In addition, we provide the practical RNC implementation in C language and measure the practical encoding and decoding rates versus different packet sizes and varying numbers of packets for combination.

6.1 Challenges of NC

Although, NC provides new design approaches to many problems of networking, and especially, the unreliable and unpredictable wireless medium, there are many challenges associated with its practical implementation such as: delay, new design architectures of the router and the switch, the information security issue, and the transmission reliability issue.

Delay when using NC is unavoidable. The delay can be classified into two types: encoding/decoding delay and waiting delay. The encoding delay is the delay that the router has to combine incoming packets using operations in a selected finite field and the decoding delay is the delay that the receiver have to solve the system of equation in order to recover the original information. The waiting delay, on the other hand, is the waiting time that the router has to wait for getting an enough number of packets before combining and sending out. The receivers also have to wait for getting enough the number of coded packets before decoding. Those delays may affect the time-sensitive applications like multimedia applications.

The second problem in using NC for the current networking infrastructure is that we need to redesign all the routers so that they are capable of encoding or decoding. Current routers are using storing and forwarding mechanism. The data are stored, and unmodified, then forwarded to the next nodes in the network. Using

NC, the routers have to generate the random coefficient and encode packets from different sources into coded packets before forwarding. Consequently, the routers have to perform more complicated operations. The network designers have to make significant modifications for the current router architectures. Furthermore, in the network where there are a lot of users involving in the communications, designing an appropriate and efficient network code structure will require considerable research effort.

The third problem in using NC is the information security. As seen in the example of wireless broadcast using NC, each receiver decodes both information it wants and information intended for other receivers. This capability might not be good in term of information security in wireless networks such as cellphone or WLAN/WiMax networks, where the information privacy must be guaranteed. In wireless mess networks with multicast or multiple unicast, the information that is addressed to different receivers is encoded into same packets. Thus receivers may be able to decode all information of other receivers. Therefore, we need to consider the security into practical NC design.

The reliability is another problem of NC. Because all information is randomly encoded, the most important factor is the ability to decode the information the receiver wants. If a receiver gets a less number of packets than necessary, it is not able to decode the information and the received packets become useless. We have seen in the wireless multiple unicast using random NC is that while some other nodes are able to decode the packet, some nodes are not. Thus designing an appropriate NC structure is very important in this case. In addition, with NC, all the information is mixed and the destination becomes less important. This means that routing from the source to the destination does not play an important role in the end-to-end communication. This may raise a question of the reliability in the end-to-end communication when using NC.

6.2 Future work

In the future work, that we are interested in doing following research on NC: (1) practical implementation of NC for multi-user wireless networks, (2) NC for wireless mess networks, and (3) NC for P2P networks. In our current work, we consider NC for single-hop wireless networks with concentration on the theoretical aspect. It is very interesting to learn how NC practically works on the current network infrastructures such as WLAN. It is very interesting to find answers for questions such as: How does NC interact with the transport and MAC layer and what is the maximum transmission speed can be achieved under the current network architectures?, etc.

Our current research focuses on the application of NC in single-hop networks. NC is also potential for wireless mess networks in which information is transmitted via multiple hops. In such networks, there are normally no centralized control systems that control the overall routing of the information like in wired networks. Using NC, the broadcast nature of wireless can be exploited. Any node, if opportunistically receiving information from other, can encode them with other information using RNC algorithms. Although, it has been proved that doing random encoding and decoding at the destination can avoid complicated routing, there are still a lot of unsolved problems when employing NC with the current routing infrastructures.

NC is also very promising for improving the performance of P2P networks. P2P networks are application-layer multicast networks in which each client act as both a data sink and a source. Designing an efficient P2P network requires us to design efficient application-layer routing algorithms. This means that P2P routing algorithm needs to find the best sharing strategy so that the data is delivered to the intended receiver in the fastest way and consume least transmission bandwidth. In P2P networks, data can travel multiple hops and multiple paths. NC might be a very efficient solution in this scenario as we see in wireless mess networks where data is transmitted via multiple hops and multiple paths.

Bibliography

[1] C. Shannon, "A mathematical theory of communication," *ACM SIGMOBILE Mobile Computing and Communications Review*, 2001.

[2] S. B. Wicker, *Error Control Systems for Digital Communication and Storage.* Prentice-Hall Inc, NJ, USA, 1994.

[3] R. Ahlswede, N. Cai, R. Li, and R. W. Yeung, "Network information flow," *IEEE Trans. Inform. Theory*, vol. 46, pp. 1204–1216, July 2000.

[4] Y. Wu, P. A. Chou, and S. Kung, "Information exchange in wireless networks with network coding and physical-layer broadcast," in *Technical Report MSR-TR-2004-78, Microsoft Research*, August 2004.

[5] J. Saltzer, D. Reed, and D. Clark, "End-to-end arguments in system design," *ACM Transaction on Computer System*, vol. 2, no. 4, pp. 277–288, November 1984.

[6] D. Nguyen, T. Nguyen, and B. Bose, "Wireless broadcast using network coding," in *Third Workshop on Network Coding, Theory, and Applications*, January 2007.

[7] D. Nguyen, T. Nguyen, and X. Yang, "Wireless multimedia transmission with network coding," *Packet Video 2007*, November 2007.

[8] T. Tran, T. Nguyen, and B. Bose, "A joint network-channel coding technique for single-hop wireless networks," in *IEEE NetCod Workshop*, January 2008.

[9] G. C. Clark and J. B.Cain, *Error-Correction Coding for Digital Communications.* New York: Plenum, 1982.

[10] S. Kallel and D. Haccoun, "Generalized type ii hybrid arq scheme using punctured convolutional codes," *IEEE Transactions on Communications*, vol. 38, pp. 1938 – 1946, Nov. 1990.

[11] S. Chandran and S. Lin, "Selective-repeat-arq schemes for broadcast links," *IEEE Transactions on Communications*, vol. 40, pp. 12–19, Jan. 1992.

[12] Y. Wu, P. A. Chou, and K. Jain, "A comparision of network coding and tree packing," in *ISIT*, 2004.

[13] R. Koetter and M. Medard, "An algebraic approach to network coding," *IEEE/ACM Trans. Networking*, vol. 11, no. 3, Oct 2003.

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

[15] C. Fragouli, D. Katabi, A. Markopoulou, M. Medard, and H. Rahul, "Wireless network coding: Opportunities and challenges," in *IEEE Military Communications Conference*, October 2007.

[16] S. Chachulski, M. Jennings, S. Katti, and D. Katabi, "Trading structure for randomness in wireless opportunistic routing," in *ACM SIGCOMM*, 2007.

[17] A. Eryilmaz, A. Ozdaglar, and M. Medard, "On delay performance gains from network coding," in *CISS*, march 2006.

[18] J. F. Kurose and K. W. Ross, *Computer Networking: A Top-Down Approach.* ISBN: 0-13-607967-9 Addison Wesley Longman Inc, 2007.

[19] D. Bertsekas and R. Gallager, *Data Networks.* Prentice Hall; 2rd edition, 1992.

[20] M. V. D. S. P. A. Chou, *Multimedia over* IP *and wireless networks: compression, networking, and systems.* Academic Press, 2007.

[21] http://www.pplive.com/.

[22] http://www.joost.com/.

[23] http://www.metacafe.com/.

[24] Q. Dong, J. Wu, W. Hu, and J. Crowcroft, "Practical network coding in wireless networks," in *Proc. of 13th Annual ACM International Conference on Mobile Computing and Networking*, 2007.

[25] A. Willig, M. Kubisch, and A. Wolisz, "Result of bit error rate measurements with an IEEE 802.11 compliant PHY," *Technical Report: TKN-00-08, Technical University Berlin*, p. 40, September 2000.

[26] C. H. Nam, S. C. Liew, and C. P. Fu, “An experimental study of ARQ protocol in 802.11b Wireless LAN,” in *Wireless Personal Multimedia Communications (WPMC’02)*, October 2002.

[27] T. Tran, T. Nguyen, and B. Bose, “A joint network-channel coding technique for single-hop wireless networks,” in *NetCod Workshop*, January 2008.

[28] T. Cover, “Broadcast channels,” *IEEE Transactions on Information Theory*, vol. IT-18, pp. 2–14, January 1972.

[29] D. S. Lun, M. Medard, and R. Koetter, “Efficient operation of wireless packet networks using network coding,” in *International Workshop on Convergent Technologies*, June 2005.

[30] M. Cagalj, J. Hubaux, and C. Enz, “Minimum-energy broadcast in all-wireless networks: NP-completeness and distribution issues,” in *ACM/IEEE Mobicom*, September 2002, pp. 172–182.

[31] W. Liang, “Constructing minimum energy broadcast trees in wireless ad hoc networks,” in *Proc. of 3rd ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc), Lausanne, Switzerland*, June 2002, pp. 112–122.

[32] A. Ahluwalia, E. Modiano, and L. Shu, “On the complexity and distributed construction of energy-efficient broadcast trees in static ad hoc wireless networks,” *IEEE Transactions on Wireless Communications*, vol. 4, pp. 2136–2147, September 2005.

[33] Z. Li and B. Li, “On increasing end-to-end thoughput in wireless ad hoc networks,” in *Conference on Quality of Service in Heterogeneous Wired/Wireless Networks (QShine)*, 2005.

[34] ——, “Network coding: the case for multiple unicast sessions,” in *Allerton Conference on Communications*, 2004.

[35] D. Lun, M. Medard, R. Koetter, and M. Effros, “On coding for reliable communication over packet networks,” in *Proc. of 42nd Annual Allerton Conference on Communication, Control, and Computing Monticello, IL*, October 2004.

[36] S. Katti, D. Katabi, W. Hu, H. Rahul, and M. Medard, “The importance of being opportunistic: Practical network coding for wireless environments,” in *Proc. of 43rd Annual Allerton Conference on Communication*, September 2005.

[37] C. Fragouli, J. L. Boudec, and J. Widmer, “Network coding: An instant primer,” in *Technical Report, TR2005010, EPFL*, 2005.

[38] S. Deb, M. Effros, T. Ho, D. R. Karger, R. Koetter, D. S. Lun, M. Medard, and N. Ratnakar, “Network coding for wireless applications: A brief tutorial,” in *Proc. of International Workshop on Wireless Ad-hoc and Sensor Networks*, May 2005.

[39] A. Eryilmaz, A. Ozdaglar, and M. Medard, “On delay performance gains from network coding,” in *40th Annual Conference on Information Sciences and Systems*, March 2006, pp. 864–870.

[40] T. Ho, M. Medard, J. Shi, M. Effros, and D. R. Karger, “On randomized network coding,” in *Proc. of 41st Annual Allerton Conference on Communication, Control, and Computing*, October 2003.

[41] T. Ho, M. Medard, D. R. Karger, M. Effros, J. Shi, and B. Leong, “A random linear network coding approach to multicast,” *IEEE Transaction on Information Theory*, 2004.

[42] M. Ghaderi, D. Towsley, and J. Kurose, “Reliability benefit of network coding,” *Technical Report 07-08, Computer Science Department, University of Massachusetts Amherst*, February 2007.

[43] S. Wicker, *Error Control Systems for Digital Communication and Storage.* Prentice-Hall, 1995.

[44] A. Shiozaki, “Adaptive type-II hybrid broadcast ARQ system,” *IEEE Transactions on Communications*, vol. 44, pp. 420–422, April 1996.

[45] M. Nakamura and T. Kodama, “An efficient hybrid ARQ scheme for broadcast data transmission systems (in Japanese),” *IEICE Transactions*, vol. J73-A, pp. 277–283, February 1990.

[46] P. A. Chou and Z. Miao, “Rate-distortion optimized streaming of packetized media,” *IEEE Trans. on Multimedia*, vol. 8, no. 2, 2006.

[47] S. Floyd, M. Handley, J. Padhye, and J. Widmer, "Equation-based congestion control for unicast application," in *Architectures and Protocols for Computer Communication*, October 2000, pp. 43–56.

[48] T. Nguyen and A. Zakhor, "Multiple sender distributed video streaming," *IEEE Trans. on Multimedia*, vol. 6, no. 2, pp. 315–326, April 2004.

[49] W. Tan and A. Zakhor, "Error control for video multicast using hierarchical FEC," in *Proceedings of 6th International Conference on Image Processing*, vol. 1, October 1999, pp. 401–405.

[50] W. Li, "Overview of fine granularity scalability in MPEG-4 video standard," *IEEE Trans. Circuits and Systems for Video Technology*, vol. 11, no. 3, pp. 301–317, 2001.

[51] W. Tan and A. Zakhor, "Real time internet video using error resilient scalable compression and TCP friendly transport protocol," *IEEE Trans. on Multimedia*, vol. 1, pp. 172–186, June 1999.

[52] H. Ma and M. E. Zarki, "Broadcast/multicast MPEG-2 video over wireless channels using header redundancy FEC strategies," in *Proceedings of The International Society for Optical Engineering (SPIE)*, vol. 3528, November 1998, pp. 69–80.

[53] P. Chou, A. Mohr, A. Wang, and S. Mehrotra, "Error control for receiver-driven layered multicast of audio and video," *IEEE Trans. on Multimedia*, vol. 3, pp. 108–22, March 2001.

[54] A. Mohr, E. Riskin, and R. Ladner, "Unequal loss protection: Graceful degradation over packet erasure channels through forward error correction," *IEEE Journal on Selected Areas in Communication*, vol. 18, pp. 819–828, April 2000.

[55] G. D. L. Reyes, A. Reibman, S. Chang, and J. Chuang, "Error-resilient transcoding for video over wireless channels," *IEEE Trans. on Multimedia*, vol. 18, pp. 1063–1074, June 2000.

[56] J. Robinson and Y. Shu, "Zerotree pattern coding of motion picture residues for error resilient transmission of video sequences," *IEEE Journal on Selected Areas in Communications*, vol. 18, pp. 1099–1110, June 2000.

[57] U. Horn, K. Stuhlmuller, M. Link, and B. Girod, "Robust internet video transmission based on scalable coding and unequal error protection," *Signal Processing: Image communication*, vol. 15, pp. 77–94, 1999.

[58] H. Radha, M. van der Schaar, and Y. Chen, "The MPEG-4 fine-grained scalable video coding method for multimedia streaming over IP," *IEEE Trans. on Multimedia*, vol. 3, pp. 53–68, March 2001.

[59] P. Chou and A. Sehgal, "Rate-distortion optimized for receiver-driven streaming over best effort networks," in *Packet Video Workshop*, April 2002.

[60] M. Kalman and B. Girod, "Techniques for improved rate-distortion optimized video streaming," *ST Journal of Research-Networked Media*, vol. 2, no. 1, 2004.

[61] M. L. Puterman, *Markov Decision Processes Discrete Stochastic Dynamic Programming.* John Wiley and Sons, Inc., New York, NY, 1994.

[62] R. Howard, *Dynamic Programming and Markov Decision Processes.* MIT Press, 1960.

[63] D. Bertsekas and J. Tsitsiklis, *Neuro-Dynamic Programming.* Athena Scientific, 1996.

[64] H. S. Chang, M. C. Fu, J. Hu, and S. I. Marcus, *Simulation-based Algorithms for Markov Decision Processes (Communications and Control Engineering).* Springer, 2007.

[65] H. S. Chang, R. Givan, and E. K. P. Chong, "On-line scheduling via sampling," in *Artificial Intelligence Planning Systems, Breckenridge, CO*, April 2000.

[66] Y. M. M. Kearns and A. Ng., "A sparse sampling algorithm for near optimal planning in large Markov decision processes," June 1999.

[67] A. Y. Ng and M. Jordan, "PEGASUS: A policy search method for large MDPs and POMDPs," June 2000.

[68] M. T. J. Spaan and N. Vlassis, "PERSEUS: Randomized point-based value iteration for POMDPs," pp. 195–220, August 2005.

[69] N. Meuleau, K. E. Kim, L. P. Kaelbling, and A. R. Cassandra, "Solving POMDPs by searching the space of finite policies," in *Proceedings of the Fifteenth International Conference on Uncertainty in Artificial Intelligence*, August 1999.

[70] *Rijndael's Galois Field*, http://www.samiam.org/galois.html.