Coding and Scheduling Optimization over Packet Erasure Broadcast Channels

MASSACHUSETTS INSTITUTE OF TECHNOLOGY
MAR 2 0 2012
LIBRARIES

by

Weifei Zeng

BASc., University of Toronto (2009)

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

ARCHIVES

Master of Science in Electrical Engineering and Computer Science

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

February 2012

© Massachusetts Institute of Technology 2012. All rights reserved.

Author .....
Department of Electrical Engineering and Computer Science
February 1st, 2012

Certified by .....
Muriel Médard
Professor of Elec. Eng
Thesis Supervisor

Accepted by .....
Leslie A. Kolodziejski
Chairman, Department Committee on Graduate Theses

Coding and Scheduling Optimization over Packet Erasure Broadcast Channels

by

Weifei Zeng

Submitted to the Department of Electrical Engineering and Computer Science
on February 1st, 2012, in partial fulfillment of the
requirements for the degree of
Master of Science in Electrical Engineering and Computer Science

Abstract

Throughput and per-packet delay can present strong trade-offs that are important in the cases of delay sensitive applications. In this thesis, we investigate such trade-offs using a random linear network coding scheme for one or more receivers in single hop wireless packet erasure broadcast channels. We capture the delay sensitivities across different types of network applications using a class of delay metrics based on the norms of packet arrival times. With these delay metrics, we establish a unified framework to characterize the rate and delay requirements of applications and optimize system parameters. In the single receiver case, we demonstrate the trade-off between average packet delay, which we view as the inverse of throughput, and maximum ordered inter-arrival delay for various system parameters. For a single broadcast channel with multiple receivers having different delay constraints and feedback delays, we jointly optimize the coding parameters and time-division scheduling parameters at the transmitters. We formulate the optimization problem as a Generalized Geometric Program (GGP). This approach allows the transmitters to adjust adaptively the coding and scheduling parameters for efficient allocation of network resources under varying delay constraints. In the case where the receivers are served by multiple non-interfering wireless broadcast channels, the same optimization problem is formulated as a Signomial Program, which is NP-hard in general. We provide approximation methods using successive formulation of geometric programs and show the convergence of approximations. Practical issues of implementing proposed coding and optimization scheme on existing layered network architecture are also discussed.

Thesis Supervisor: Muriel Médard
Title: Professor of Elec. Eng

We gratefully acknowledge the funding support from the Air Force Office of Scientific Research (AFOSR) under award No. 016974-002, from France Telecom S.A. under award number: 018499-00 and from BAE Systems National Security Solutions, Inc, under award number: 739532-SLIN 0002.

Acknowledgments

Foremost, I owe my deepest gratitude to my supervisor, Prof. Muriel Médard. This thesis would not have been possible without her support the last two years. Her extraordinary patience and understanding, insightful guidance and mentoring have helped me tremendously. I feel really fortunate to work in the Network Coding and Reliable Communication Group.

Besides my supervisor, I am really grateful to Dr. Chris Ng at Bell Labs, for this insightful comments and helpful feedback on major parts of the work. My sincere thanks also goes to Prof. Catherine Rosenberg at the University of Waterloo and Dr. Laurent Massoulie at Technicolor Paris Research Lab for inspiring discussions during my visits and internship. I would also like to send my gratitude to Prof. Frank Kschischang and Prof. Baochun Li, at the University of Toronto, for guiding me into the field of communications and enlightening me with the first glance of research with so much patience and enthusiasm.

I would also like to thank my friends and colleagues in my office, the NCRC group and in the EECS department. In particular, I would like to thank Flávio du Pin Calmon, Arman Rezaee, Jason Cloud, Matt Carey and Georgios Angelopoulos for many interesting discussions and memories throughout the last two years. My thanks also goes to Soheil Feizi, Ali Parendeh-Gheibi, Danail Traskov and Viveck Cadambe, for many inspiring discussions on various exciting topics and problems.

Last, and most importantly, I would like to thank my parents. I could not have come this far without the unwavering love and support from them.

Contents

List of Figures

List of Tables

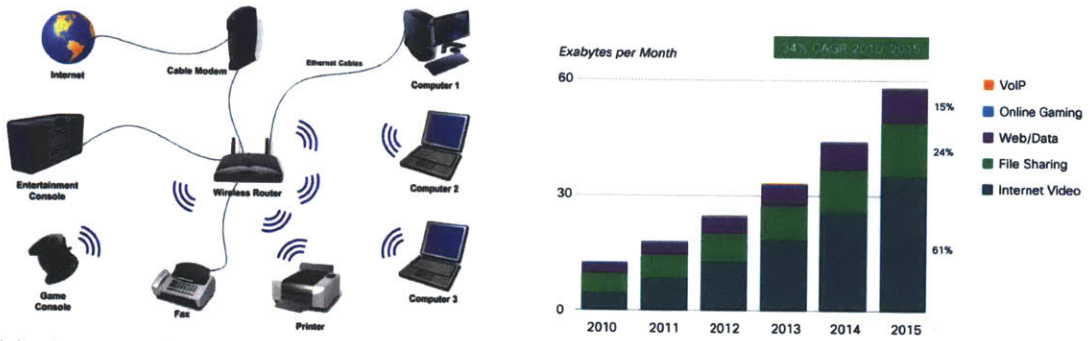
Chapter 1

Introduction

1.1 Background and Motivation

The growing diversity of network applications, protocols and architectures makes it increasingly challenging to understand and to measure the needs of various applications in terms of delay and rate for desired Quality of User Experience (QoE). One of the typical scenarios happens in the case of modern home network environment (Figure 1-1a), where multiple networking devices are usually connected with the same wireless broadcast channels associated with one or more WiFi access points (AP). The network is shared by a wide variety of applications, such as media streaming to media players, file downloading at PCs, online gaming traffic on gaming consoles and real time voice or video traffic from VoIP devices (Figure 1-1b). While there is great heterogeneity of network applications, the service pattern or the transmission strategies used in today's networking device are largely based on simple prioritization models that may be inefficient. Fundamental trade-offs between throughput and delay in most communications systems increases the importance of capturing precisely the delay-throughput requirements and constraints of applications in order for systems to allocate resources efficiently and to enhance QoE.

In many of the communication systems, the trade-off between rate and delay is most evident in the choice of coding methods and coding block sizes. Generally, a large coding



(a) An example of modern home network : various devices and applications share the same wireless channel from a single access point

(b) Cisco Visual Networking Index (Feb 2011): A simple categorization and prediction of Internet data traffic trends shows fast increasing of delay sensitive traffice such as media streaming

Figure 1-1: Home network and various networking applications

block does not only make systems more resilient to channel variations and erasures, but also provides higher throughput. As a trade-off, large coding block size usually introduces longer decoding delay, as the receivers need to collect longer coding blocks before decoding. They may also suffer longer computational delay because of the increased decoding complexity associated with longer coding blocks. The tolerable decoding delay durations vary drastically depending on a series of factors, such as channel and network statistics. But predominantly, the tolerable delay is limited by application's QoE requirements. For instance, applications like file downloading or FTP protocols aim solely to maximize the average transmission rate, thus minimizing the overall completion time. Since non-integral file parts provide no utility for users, such downloading protocols are not concerned with the inter-delivery times between consecutive packets. On the other hand, applications such as real-time video conferencing are highly sensitive to delay of any consecutive packets. Failure to meet continuous delivery deadlines in stream of packets quickly deteriorates QoE. However, the two extremes in delay sensitivities by no means represent all types of networking applications today. Progressive downloading video, for example, would be more delay sensitive than file downloading, as the video player has to catch up with certainly playback deadline for sequenced video content or packets. On the other hand, its delay sensitivity is usually not as high as a live video conference, since in a non-real time scenario, a receiver is usually allowed to buffer sufficient

content before playback starts or whenever the playback is to be resumed after some interruptions. Therefore, a range of sensitivity, instead of two extremes, is a better reflection of common applications. Furthermore, at different stage of a transmission session, variations of delay sensitivity maybe observed for the same application, as in the case of progressive video downloading, depending on the amount of sequentially buffered content.

The proliferation of low cost gateways with fast-increasing computation capability does not only bring more flexibility in packet coding options, but also provides the possibility to optimize the efficiency of network resource allocation based on coding methods and the rate-delay requirements of different applications. In this thesis, motivated by the typical home network scenario, we focus the problems in the scope of single hop wireless broadcast channel with packet erasures. We explore the possibility of applying linear network coding between the senders and receivers in such systems, from a perspective of designing and optimizing coding and scheduling parameters different delay sensitivities in the existence of delay-rate trade-offs. We investigate how properly designed and optimized coding and scheduling would allow the system to accommodate better the delay and rate requirements of various applications.

1.2 Thesis Outline and Main Contributions

The thesis starts by developing a unified framework to study rate and delay trade-off of packet based linear block coding schemes. We use a class of delay metrics based on the ℓ_p-norms of the packet inter-arrival times to represent delay-rate characteristics and requirements of applications. At one extreme of the p values, the class of delay metrics could capture the average packet delay and therefore the average rate of transmission. At the other extreme of the p values , the metrics measure the maximum ordered inter-arrival delay. We apply the metrics to linear intra-session network coding in our system and express the delay cost function in terms of coding parameter and system parameters. With these delay cost measures, we establish methods to optimize coding and scheduling parameters in the networking system, where various devices with different delay requirements are served by

single-hop wireless erasure broadcast channels, each associated with an access point (AP).

For the main parts of the thesis, we use a coding scheme that is a variation of the generation-based random linear network coding, presented in [1] and [2]. Specifically, the sender maintains a *coding bucket* for each receiver. When a transmitter is ready to send a packet to some receiver, it reads the all the packet in the coding bucket for that receiver and produces an encoded packet by forming a random linear combination of all the packets in the coding bucket. The encoded packet is then broadcasted to all the receivers. Once the receiver collects enough packets to decode all packets in the coding bucket through Gaussian elimination, it uses a separate feedback channel to send an ACK message back to the sender. The sender always receives the ACK message after a certain delay. It then purges all the packets in the coding bucket and moves new packets into the bucket. The respective delay constraints of the receivers are known to the sender, who determines adaptively the number of packets to put in the coding buckets for each receiver, by solving system-wise optimization problems. A precise description of the transmission scheme is given in Chapter 2. The coding buckets act as the Head of Line (HOL) generations in generation based schemes. However, unlike most generation-based schemes, packets are not partitioned prior to transmission and the bucket sizes in our scheme may vary over time and across different receivers, depending on each receiver's changing delay constraints.

Under this coding scheme, for various system configurations, the coding parameters are optimized jointly with time division resource allocation parameters to exploit optimal trade-off between rate and delay. We first illustrate such trade-off in the case of point-to-point erasure channels. Then, in the case of multiple receivers with one AP, we formulate the delay constrained optimization problem as a Generalized Geometric Program, which can be very efficiently solved. We compare the solutions with fixed generation size schemes for specific examples. Finally, in the case of multiple APs with non-interfering erasure broadcast channels, we formulate the problem as a Signomial Program and provide methods to approximate this non-convex optimization with successive GPs.

While the emphasis of the work is on the optimization of adaptive intra-session coding

between a centralized sender (the gateway) and the receivers, we also briefly explore the possibility of using inter-session coding when the feedback delay is insignificantly short compared to a transmission time slot. Such short feedback delay allows the sender to capture the erasure patterns experienced by all the receivers and perform opportunistic inter-session coding based on the stochastic erasure patterns. Coded packet generated by coding across multiple sessions benefits more than one receivers, and thus reduces the total number of transmissions needed for serving all packets and increases the average rates.

At the end of the thesis, we discuss and address some of the practical issues of implementing the designed coding and scheduling schemes, with minimum changes to the current network layering architecture.

The main contributions of the thesis can be summarized as follows:

1. We develop a new framework for quantifying the delay sensitivity of a variety of applications based on ℓ_p-norm of packet in-order inter-delivery times. With the variation in the p value for specific applications, the metrics allow us to capture the optimal trade-off between average delay per packet and the maximum expected in-order inter-packet delay. We illustrated these optimal trade-off with various system parameters, such as feedback delay and erasure probability in single receiver case.

2. For the single wireless packet erasure broadcast channel with multiple receivers, we construct a geometric optimization program to optimize the coding bucket size and the service time allocated to each receiver. The optimization problem can be solved efficiently and we illustrate the benefit of such optimized adaptive scheme against normal fixed size block coding schemes.

3. When there are multiple non-interfering broadcast channels serving the same set of receivers, we formulate the optimization problem into non-convex signomial problem and approximate the local optimal solutions with successive solutions of GPs.

4. We investigate the rate gain of using opportunistic pair-wise inter-session coding in the case of perfect feedback. We quantify the gain in the single packet erasure broadcast

channel with by solving a Markov chain model.

Parts of the thesis is presented in [3].

1.3 Related Works

There exists a significant amount of related literature and we shall only examine a incomplete set of relevant ones. Previous work by Walsh *et al.* [4] considers the rate and delay trade-off in multipath network coded networks, while [5] studies the related issue of rate-reliability and delay trade-off by constructing various network utility maximization (NUM) problems. The concept of network coding is introduced in [6] and linear network coding is extensively studied in [7] and [8]. Other typical rateless codes that are asymptotically optimal for erasure channels are seen in [9] [10] [11]. However, unlike linear network codes which allow intermediate nodes to recode packets, the class of fountain codes are generally only used for one-hop communication systems, as the packets can not be recoded, owing to stringent packet degree distribution requirements. In our system, the delay constraints make it difficult to apply fountain codes efficiently, as the asymptotic optimality is only achieved with coding over relatively large number of packets. On the other hand, we have feedback which will allow us to change dynamically the coding parameters. The network coding gain in overall delay of file downloading with multicast over packet erasure broadcast channel is characterized in [12] and [13]. With the use of similar linear network codes, broadcast coding schemes based on perfect immediate feedback are proposed and their delay characteristics are analyzed in [14] [15] [16]. An analysis of random linear codes with finite block size is given in [17].

The scenario considered in the thesis is the multiple unicasts from a single source to a group of receivers through several wireless erasure broadcast channels. Such scenario can arise in typical home networks with a single gateway and multiple access points, in overlapping regions of cellular networks and in satellite networks for example. The channel model we consider in this thesis is basically a broadcast channel, which is one of the most important and widely studied wireless channel models [18]. Even though there is no general charac-

terization of the capacity region of the broadcast channel (BC), advances have been made in many sub-classes of the BC [19]. Among them, Gaussian BC [20] has received the most attention. In network settings, however, an erasure channel model may provide simpler abstractions to facilitate analysis. In a wireless network, when some mechanism of interference avoidance is reinforced, every hyper-arc can be treated as independent broadcast erasure channel, which captures successfully the media sharing and lossy transmission properties of wireless links. Thus, broadcast erasure channels have been increasingly considered [21,22]. Most the recent works [12,21,23] on the channel focus mainly on the capacity and coding gains in the case of multicast on the packet erasure broadcast channel. In general, the capacity of broadcast erasure channel without feedback is given by [21]. More importantly, such capacity can be achieve with network coding with timing sharing among various receivers. Most recently, Gatzianas *et al.* [24] and Wang *et al.* [25] have shown that perfect feedback does expand the capacity region of single input broadcast erasure channel.

1.4 Thesis Organization

The remainder of the thesis is organized as follows. In Chapter 2, we provide a short overview of Geometric Programming and Network Coding. We then establish the transmission model, coding and scheduling schemes considered in the later part of the thesis. We also give the definition of a class of delay metrics, as well as showing how the metrics apply specifically to the scheme we consider. Chapter 3 constitutes the main part of the thesis. It focuses on the optimization of coding and scheduling parameters. The chapter is organized into two parts. In the first part, we consider a single wireless broadcast channel with packet erasures and one or more receivers. We construct a joint optimization program for the coding parameters and scheduling coefficients. The program is converted into a Generalized Geometric Program and solved efficiently using GP techniques. We illustrate the delay and throughput trade-offs with different system parameters and compare the solutions with the fixed generation size schemes. In the second part, the same problem is investigated in the case of multiple non-interfering wireless packet erasure broadcast channels. The approach extends the methods

used in the first part. We construct a similar optimization problem into a Signomial Program and we provide approximation algorithm to this non-convex problem. In Chapter 4, we focus on the scenario where the feedback delay is insignificant and we consider the possibility of using inter-session network coding, based on the erasure pattern the sender sees from the feedback. Chapter 5, we investigate the practical side of implementing similar schemes with the existing protocol architecture and discuss the difficulties and challenges. Finally, Chapter 6 concludes the thesis and highlight some future works.

Chapter 2

Preliminaries, Models and Metrics

Before we proceed to consider the packet erasure broadcast channel with multiple receivers in Chapter 3, we first establish the delay sensitivity model and the coding scheme in this chapter. We start with a review of the basics of network coding, especially generation based practical network coding schemes. We define the delay metrics or cost functions for measuring the delay performance of coding schemes. A very brief introduction to geometric programming is also presented, since geometric programming techniques serve as the primary tool for solving the optimization problem that is constructed in the Chapter 3.

2.1 Network Coding

The introduction of network coding by Ahlswede *et al.* [6] marks a paradigm shift in how information is treated over the networks. Traditionally, each node in a network simply relays information from input links to output links, although from a information theoretic perspective, there is no reason for restricting nodes to relay function. While unicast capacity can be readily solved and achieved by Ford-Fulkerson algorithm with routing, the min-cut capacity of multicast transmission cannot be achieved by routing in general. Network coding, on the other hand, allows intermediate nodes not only to relay information, but also to encode received information which is shown to achieve significant throughput advantages

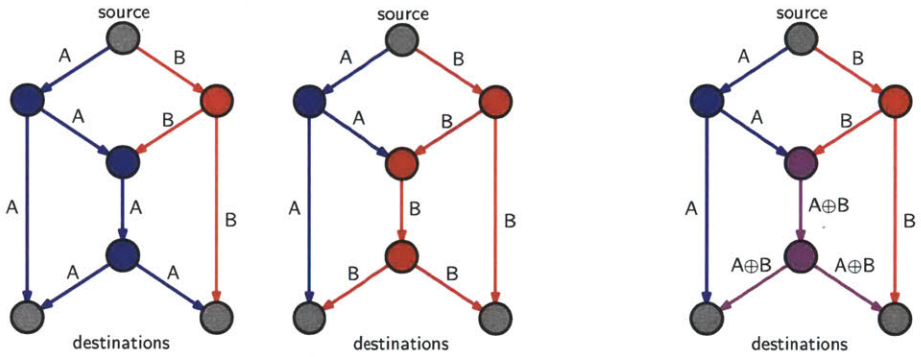
over non-coding solutions in many cases. For single multicast on directed acyclic or cyclic graphs, it is proven that network coding achieves the min-cut capacity for each receiver [6]. Furthermore, multicast capacity can be achieved with *linear network codes* [7] [8]. The preservation of transmission efficiency with linear network codes, allows linear multicast solutions to be constructed efficiently under algebraic frameworks [8] [26]. In particular, Ho *et al.* [1] propose randomly linear network codes (RLNC) and show that, when the finite field $\mathbb{F}_q$, over which coding is performed, is large enough, the linear coding coefficients can be randomly chosen from $\mathbb{F}_q$ without sacrificing throughput or transmission efficiency.

2.1.1 Butterfly graph

One of the classical examples for illustrating the advantage of network coding over routing is on the butterfly graph, as shown in Figure 2-1. Each edge on the graph has a unit capacity. The source at the top aim to multicast two packets A and B, both of one unit size, to two destinations at the bottom of the figure. Routing solutions result in either Figure 2-1a or 2-1b, where the central edge transmits either A or B. For these solutions, it is impossible for both destinations to receive both packets in one time slot and the average throughput to each destination is 1.5 packets. Figure 2-1c gives a network coding solution to the same problem. Instead of transmitting the original packets, the central edge now carries the coded packet $A \oplus B$. Each destination in this case will receive one original packet and the packet $A \oplus B$, which allow the receivers to recover both original packet by solving a simple linear system. Hence, a network coding solution gives each receiver a rate of 2, which is the min-cut capacity for each of the node. It is shown that the throughput gap between routing and network coding solutions can be arbitrarily large [26], for multicast on certain directed acyclic graphs.

2.1.2 Intra-session vs inter-session coding

In the butterfly example shown in Figure 2-1, we have a multicast scenario, in which all destinations requesting all the packets (or source processes) A and B. Therefore, all packets



(a) Butterfly graph routing solution A

(b) Butterfly graph routing solution B

(c) Butterfly graph network coding solution

Figure 2-1: Comparison between routing and network coding for multicast on the butterfly graph. The source looks to transmit A and B to both destinations: Routing in (a) and (b) forces central edge to transmit original packet, resulting 1.5 average throughput. Network coding in (c) achieves min-cut capacity of 2 for both destinations.

are said to be in the same session. Coding among these packets are therefore referred to as *intra-session coding*. However, in a slightly modified version of the butterfly, shown in Figure 2-2, we have two source-sink pairs. Source node s_1 look to transmit A to t_1, while source node s_2 aim to deliver B to t_2. The same coding $A \oplus B$ on the central edge is then referred to as *inter-session coding*, since A and B belong to different transmission sessions in this case. For a more detailed treatment of intra-session and inter-session coding, we refer the reader to [27]. In most of the thesis, we have a multiple unicast scenario, where each receiver requests a independent flow of packets. The choice of inter and intra session codings becomes important, especially regarding the decodability of the received packets, as covered in later chapters. In Chapter 3, we focus on intra-session coding, i.e. coding packet within the same flow, while in Chapter 4 we consider inter-session coding for rate gain. The capacity of non-multicast transmission over general network remains unsolved even for very simple cases. In this thesis, we shall not attempt to answer the problem regarding the rate region and achievability. Rather, we examine the existing generation based coding scheme on a particular network and investigate the possibility of improving coding performance under certain delay constraints characterized in our delay-rate trade-off framework.

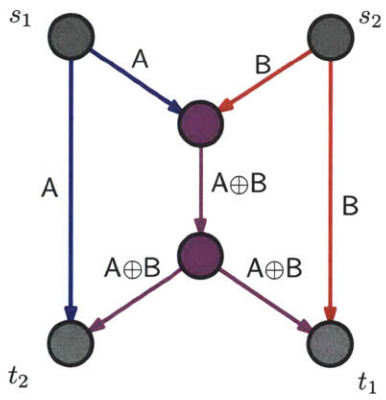


Figure 2-2: A inter-session coding example of the butterfly graph: Source s_1 transmits A to sink t_1, while s_2 transmit B to t_2.

2.2 Coding Schemes and Delay Metrics

We start with the simplest transmission model. Consider a point-to-point communication system illustrated in Figure 2-3. The sender, denoted by Tx, is to communicate with the receiver, denoted by Rx, through a wireless erasure channel with packet erasure probability of ε. In addition, a perfect feedback channel with delay D allows the receiver to send *delayed* feedback to the transmitter. The system is assumed to be time-slotted. At any time slot, the wireless packet erasure channel delivers at most one packet successfully, in the case when the packet is not erased. Otherwise, no packet is received at the receiver. A packet flow f, arrives at the sender and is to be delivered to the receiver. The packets in this flow are denoted as $\{P_1^f, P_2^f, \cdots, P_N^f, \cdots\}$. Each of them is treated as a length m vector in the space $\mathbb{F}_q^m$, over some finite field $\mathbb{F}_q$, i.e. $P_i^f = [P_{i1}^f P_{i2}^f \ldots P_{im}^f]^T$, where $P_{ij}^f \in \mathbb{F}_q$. For simplicity, we consider only the delivery of the first N packets of the flow and assume that all N data packets are assumed to be available at the sender prior to any transmissions. The results can be generalized to infinite packet flows. Furthermore, we assume that N is sufficient large such that $N \gg K_{\max}$, where $K_{\max}$ is the maximum linear code block size, in terms of number of packets. Next, we consider the typical fixed size generation-based network coding scheme and the adaptive scheme we propose.

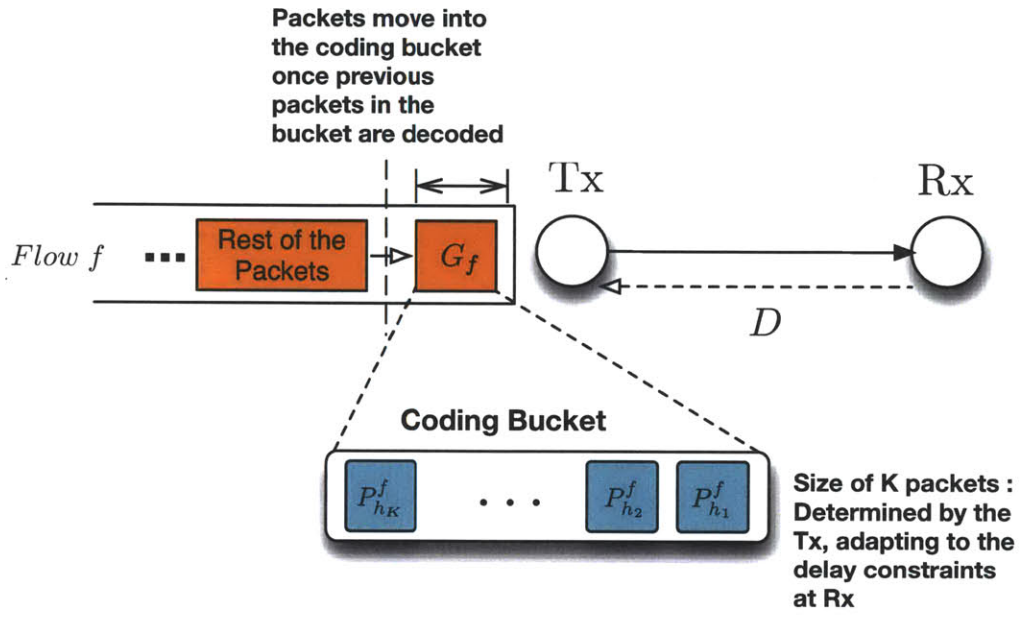


Figure 2-3: Adaptive linear network coding based transmission model: For the packet flow f, the transmitter (Tx) maintains a coding bucket G_f, whose size is adaptively determined based on the delay constraints at the receivers.

2.2.1 Fixed generation-based Coding Scheme

In a fixed generation-based linear network coding scheme, the sender first chooses an integer $K \geq 1$. It sequentially partitions the N packets into $\lceil \frac{N}{K} \rceil$ *generations* $\{G_1^f, \cdots, G_{\lceil \frac{N}{K} \rceil}^f\}$. Each generation G_i^f consists of K consecutive packets, i.e. $G_i^f = \{P_{iK+1}^f, \cdots, P_{\min((i+1)K,N)}^f\}$. The generations of packets are transmitted sequentially according their generation indices. We use $G_h^f = \{P_{h_1}^f, \cdots, P_{h_K}^f\}$ to denote the Head of the Line (HOL) generation for transmission, in which $h = 1, \cdots, \lceil \frac{N}{K} \rceil$ is the generation index, and $h_k, k = 1, \cdots, K$ are the indices of packets within the generation. In a time-slotted communication system, it is assumed that the sender has read all the packets in the HOL generation once the previous HOL generation is delivered. At every time slot t, the transmitter generates a coded packet $P[t]$ that is a

linear combination of all packets in the HOL generation G_h^f (shown in Figure 2-5), i.e.

$$P[t] = \sum_{k=1}^{K} a_k[t] P_{h_k}^f, \tag{2.1}$$

where $a[t] = (a_1[t], \cdots, a_K[t])$ is referred to as the coding coefficient vector for packet $P[t]$. Each coding coefficient vector is uniformly and independently chosen at random from $\mathbb{F}_q^K$ [1]. The coded packet, with the coefficient vector appended in the header, is then sent to the receiver over the erasure channel.

The receiver collects coded packets over time, until it obtains K linearly independent packets to construct a full rank linear system of the K packets. For example, if the HOL generations contains $P_1, \ldots, P_K$, and the K linearly independent packets collected are $Y_1, \ldots, Y_K$, with coding coefficient vector $\mathbf{a}_i = [a_{i1} \ldots a_{iK}]$ for Y_i. We have the linear system,

$$\begin{bmatrix} Y_1 \\ Y_2 \\ \vdots \\ Y_K \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1K} \\ a_{21} & a_{22} & \cdots & a_{2K} \\ \vdots & \vdots & \ddots & \vdots \\ a_{K1} & a_{K2} & \cdots & a_{KK} \end{bmatrix} \begin{bmatrix} P_{11} & P_{12} & \cdots & P_{1K} \\ P_{21} & P_{22} & \cdots & P_{2K} \\ \vdots & \vdots & \ddots & \vdots \\ P_{K1} & P_{K2} & \cdots & P_{KK} \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1K} \\ a_{21} & a_{22} & \cdots & a_{2K} \\ \vdots & \vdots & \ddots & \vdots \\ a_{K1} & a_{K2} & \cdots & a_{KK} \end{bmatrix} \begin{bmatrix} P_1 \\ P_2 \\ \vdots \\ P_K \end{bmatrix}. \tag{2.2}$$

With all Y_i and a_{ij} known to the receiver, the full rank linear system can be solved by Gaussian elimination to recover all P_{ij} and thus all packets P_i. Moreover, given a large enough field $\mathbb{F}_q$, it is shown that with high probability [1], any K received packets are linearly independent. Therefore, in this thesis, we assume that a receiver is always able to decode whenever it receives K packets.

Once the receiver decodes the HOL generation successfully, it sends an ACK message through the feedback channel to the sender. The sender, who receives the ACK after a delay of D time slots, will purge the old HOL generation and move on to the next generation in the line. The process repeats until all the packets are delivered to the receiver.

2.2.2 Adaptive Linear Coding Scheme

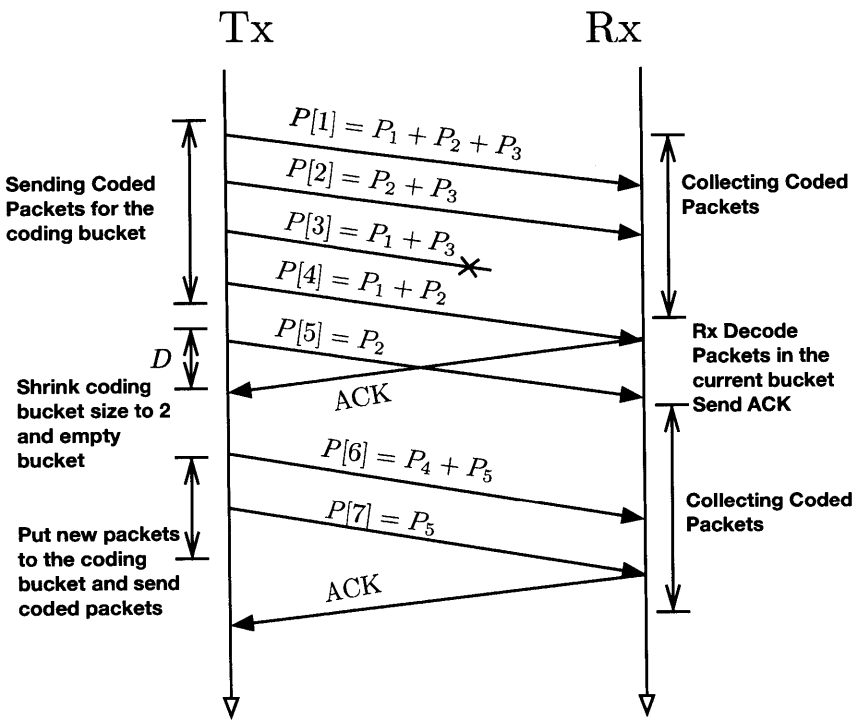


Figure 2-4: A session with varying coding bucket size. The initial coding bucket size is 3, containing packets $\{P_1, P_2, P_3\}$. After finishing the three packets, the receiver decide to shrink the bucket size to 2, containing packets $\{P_5, P_6\}$.

Our scheme modifies such generation-based network coding in the following ways. The packets in the flow are not partitioned into generations prior to the transmissions. Instead, a *coding bucket* is created and acts like the HOL generation. We use the term *bucket* to avoid confusion with normal generation-based schemes. The size of the bucket in term of number of packets is denoted as K, like in the case of fix generation schemes. The sender collects information about user-end delay constraints and chooses the bucket size K dynamically. Therefore, the coding bucket size may change over time for the same packet transmission session. Figure 2-4 gives a simple example of the timing diagram of our adaptive scheme. At the beginning, the coding bucket has a size of 3 and contains three packets $\{P_1, P_2, P_3\}$. The sender keeps transmitting encoded packets, i.e. $P[1]$ to $P[5]$, of these three packets. Some

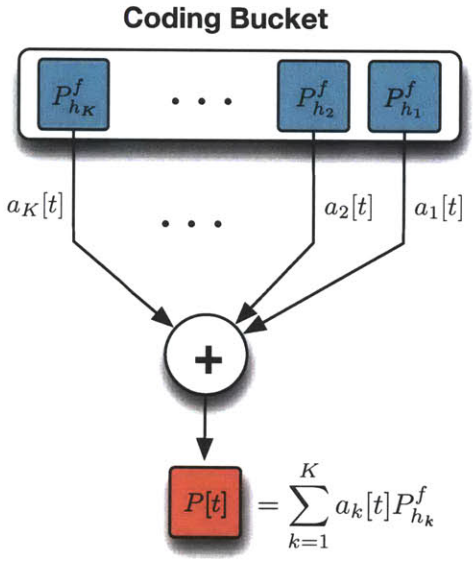


Figure 2-5: An illustration of encoding: the encoded packet $P[t]$ is the linear combination of all packet in the coding bucket with coefficients $a_k[t]$ chosen randomly from some finite field.

of the coded packets, such as $P[3]$ in this case, may be erased during the transmission. The receiver collects these coded packets until it has enough coded packets to decode all packets currently in the coding bucket, i.e. $\{P_1, P_2, P_3\}$. This is done when it receives $P[4]$ in this case. At this time, the receiver sends a ACK message back to the transmitter, and informs the latter to move on to new packets. Upon receiving the ACK feedback, the transmitter empties the bucket. We define all the transmissions between two bucket emptying actions to be a *round of transmissions*. A round of transmissions therefore includes all the transmissions taken place for all the packets in the current coding bucket to be delivered. In this example, the sender further decides to shrink bucket size to 2 for the next round of transmissions, possibly because of the tighter delay constraint experienced at the receiver. Therefore, only two packets $\{P_4, P_5\}$ go into the bucket for the new round of transmissions. The process repeats all packets are decoded at the receiver. We leave the details of adaptively determining coding bucket size to Chapter 3.

There are a few remarks regarding the adaptive coding scheme:

1. Packets stay in the coding bucket for the entire round of transmission. They are moved into the coding bucket at the beginning of the round and purged from the bucket at the end of the round. This is to guarantee the random linearly coded packets are generated from the same group of original data packets and thus the correct decoding of the original data packets at the receiver. All the packets delivered in the same round of transmission are decoded together, as in the case of generation based schemes.

2. The size of the coding bucket remains the same throughput each round of transmission. In principle, the coding bucket size can be change within a round of transmission. For example, if the size increases, new packets can be introduced into the bucket. However, it is then more complicated to guarantee the correct decoding at the receiver. For simplicity, the change of bucket size within a round of transmission is not considered in the thesis.

3. The availability of feedback is important and is seen in may practical systems at different communication layer as discussed in Chapter 5. Feedback is only used when the receiver collects enough linearly independent coded packets. The transmitter is informed about the delay constraints at the receiver through some feedback channel or from the ACK message.

2.2.3 ℓ_p-Norm Delay Metrics

Now we define the delay metrics used in the paper. Following the notations used in the previous part, let T_i be the time slot in which the packet P_i^f is decoded at the receiver, and is delivered to upper layer. It is important that we require the delivery of original data packets $\{P_1^f, \cdots, P_N^f\}$ to be *in order*. In the case when the sequence of packets decoded is out-of-order, we assume that they are buffered at the receiver first, to ensure in-order final delivery. Let T_i represent the final in-order delivery times of packets P_i^f. Because of the in-order delivery, we have $T_1 \leq T_2 \leq \cdots \leq T_N$. Consequently, it allows us to define the

ordered inter-arrival times ΔT_i of the original packets to as follows,

$$\Delta T_1 \triangleq T_1 + D \tag{2.3}$$

$$\Delta T_i \triangleq T_i - T_{i-1}, \quad i = 2, \cdots, N, \tag{2.4}$$

where D is the feedback delay from the receiver to the sender. Note that a feedback message ACK is always assumed to be received correctly after D time slots. However, in the case when there is more than one receiver, we assume that, in general, receivers experience different feedback delays across the system owing to its location and channel variations. Let the size of each data packet be L. We define the delay cost function as a metric of the following form,

$$d(p) \triangleq \frac{1}{L}\left(\frac{\sum_{i=1}^{N}(\mathbf{E}[\Delta T_i])^p}{N}\right)^{1/p}, \quad p \in [1, \infty), \tag{2.5}$$

where $\mathbf{E}[\Delta T_i]$ is the expected value of ΔT_i. The expectation is taken over the distribution of packet erasures over the system and all the randomness associated with the coding and scheduling scheme, which are discussed in more detail in the next Chapter.

Mathematically, the delay metric is the ℓ_p-norm of the vector of expected inter-delivery times, i.e. $[\mathbf{E}[\Delta T_1] \cdots \mathbf{E}[\Delta T_N]]^T$, normalized by the total packet number and individual packet size. Physically, however, p measures the sensitivity of the receiver toward the inter-packet delays. As discussed before, such delay sensitivity and is predominantly dependent on the type of applications running on the receiver. As the value of p varies from 1 to ∞, the delay function becomes increasingly biased towards the large components in the vector because of the ℓ_p-norm, hence indicating increasing user sensitivity toward large inter-packet delay. In particular, the two extreme values of p give two interesting interpretations of the delay metrics. First, consider the case when $p = 1$. Since $\sum_{i=1}^{N} \mathbf{E}[\Delta T_i] = \mathbf{E}[T_N] + D$, the delay in (2.5) simplifies to,

$$d(1) = \frac{\mathbf{E}[T_N] + D}{LN}. \tag{2.6}$$

That is, $d(1)$ is the average delay per packet, normalized by the size of a packet. Minimizing

$d(1)$, therefore, is equivalent to maximizing the average rate the receiver. On the other hand, consider the case when $p = \infty$. Because of the *infinity norm*, the delay function in (2.5) reduces to,

$$d(\infty) = \frac{\max_i \mathbf{E}[\Delta T_i]}{L}. \tag{2.7}$$

Effectively, minimizing $d(\infty)$ translates into minimizing the maximum expected inter-arrival time between any two successive packets. We call this the *per-packet delay.*

The flexibility in choosing various p-value for delay metrics provides a unified way of looking at the delay sensitivity at the user side. If a user is downloading a file, he is certainly more concerned about shortening the overall completion time or average delay per packet. Consequently, $d(1)$ is the appropriate delay cost function to be measured and optimized. On the other hand, if the user is running a real-time video applications that is extremely delay sensitive, then $d(\infty)$ is more likely to be the right cost function to be minimized as it allows sequence of packets to catch up quickly with respective delivery deadlines. Moreover, in between the two extremes, we can choose appropriate sensitivity value p for applications like progressive video downloading and even dynamically adjust the value according to the changing delay requirements.

2.2.4 Delay In Adaptive Coding Scheme

Next, we consider how these delay metrics can be applied in the coding scheme we discussed previously. We only need to examine the adaptive scheme, as the fixed generation scheme can be viewed as a special case when coding bucket size is a constant. In the adaptive coding scheme, a receiver will decode all packets in the current bucket before informing the sender to empty the bucket and move in new packets. Assume that the rate at which the coded packets are transmitted is r. Consider a round of transmissions of a bucket of K packets $\{P_{i_1}, \cdots, P_{i_K}\}$. Once the receiver collects K linearly independent coded packets of the bucket, it decodes all K packets together. Hence, the ordered inter-arrival times of original packets will satisfy, $\mathbf{E}[\Delta T_{i_1}] = \frac{K}{r} + D$ and $\Delta T_{i_1} = \cdots \Delta T_{i_K} = 0$. In general, consider the case when the bucket size remains the same for a sequence of N packets, $\{P_{i_1}, \cdots, P_{i_N}\}$.

N is divisible by K, as the bucket size may only change when the bucket is emptied. The packets will sequentially enter the bucket in groups of K packets. Then, for the inter-arrival time of the j-th packet, we have,

$$\mathbf{E}[\Delta T_{i_j}] = \begin{cases} \frac{K}{r} + D, & \text{if } j \equiv 1 \pmod{K}, \\ 0, & \text{otherwise.} \end{cases} \tag{2.8}$$

Therefore, if the adaptive scheme chooses bucket size of K of a sequence of N packets, we can simplify (2.5) to measure the delay cost function for the transmission of the N packets, resulting in:

$$d(p) = \frac{1}{L}\left(\frac{\frac{N}{K}\sum_{j=1}^{K}(\mathbf{E}[\Delta T_{i_j}])^p}{N}\right)^{1/p} \tag{2.9}$$

$$= \frac{1}{L}\left(\frac{\frac{N}{K}(\frac{K}{r}+D)^p}{N}\right)^{1/p} \tag{2.10}$$

$$= \frac{\frac{K}{r}+D}{LK^{1/p}}. \tag{2.11}$$

In particular, under this coding scheme, the delay $d(p)$ seen by the receiver over the period is independent of N as long as the coding bucket size remains to be K. Hence, we drop N and only consider the bucket size K for rest of the paper. Furthermore, in practice, K takes only positive integer values in $[1, K_{max}]$, where K_{max} is the maximum bucket size, limited by the maximum tolerable computation complexity of the target system. In this work, for simplicity, we assume that the coding bucket size K are real value in $[1, K_{max}]$.

2.3 Geometric Programming

We give a concise primer of Geometric Programming (GP) techniques and terminology before looking specifically into the optimization problems associated with our system, as GP serves as the primary tool for the next chapter. For more comprehensive coverage of the topic, we refer the reader to [28], [29].

GP is a class of mathematical optimization problems characterized by some special forms of objective functions and constraints. A typical GP is nonlinear and non-convex, but can be converted into a convex program so that a local optimum is also a global optimum. The theory of GP has been well studied since the 60s [30]. The convexity and duality of the these problems are very well understood. Well developed solution techniques, such as *interior point methods* are capable of solving GPs efficiently even for large scale problems. Many high-quality GP solvers are available (e.g. MOSEK package and CVX [31]) for providing robust numerical solutions for generalized GPs (GGP). GP or GGP see many applications in engineering design and analysis, as they can be used to model or approximate a wide variety of practical problems, especially in areas such as digital circuit design. In wireless communication, GP is often used to solve transmission power control problems [29].

Consider a vector of decision variables $\mathbf{x} = [x_1 \dots x_n]^T$. A real function $g : \mathbf{R}^n \to \mathbf{R}$ is said to be a *monomial* if it can be written in the form,

$$g(\mathbf{x}) = c \prod_{i=1}^{n} x_i^{a_i}, \tag{2.12}$$

where the coefficient c is *positive*, and the exponents $a_1, \dots, a_n$ are arbitrary real numbers. A function $f : \mathbf{R}^n \to \mathbf{R}$ in the form

$$f(\mathbf{x}) = \sum_{k=1}^{K} c_k \prod_{i=1}^{n} x_i^{a_{ik}}, \tag{2.13}$$

with all c_k being positive real numbers, is called a posynomial. A posynomial is the sum of arbitrary number of monomials. On top of this, any function $\tilde{f}$, which can be constructed

with posynomials using addition, multiplication, positive power and maximum operations is called a *generalized posynomial.*

A *standard form* geometric program is presented as follows,

$$\begin{aligned} \text{minimize} \quad & f_0(\mathbf{x}) \\ \text{subject to} \quad & f_i(\mathbf{x}) \leq 1, \ i = 1, \ldots, m, \\ & g_j(\mathbf{x}) = 1, \ j = 1, \ldots, p, \end{aligned} \tag{2.14}$$

where $f_i(x)$ are posynomials and $g_i(x)$ are monomials, and x_i are the decision variables, which are also implicitly assumed to be positive, i.e. $x_i > 0, \ i = 1, \ldots, n$. In particular, the objective of the optimization has to be minimizing some posynomial. That says, for solving maximization problems with GP, the objective function has to be in the form of some monomial $g(\mathbf{x})$, so that instead of maximizing $g(\mathbf{x})$, we can minimize $\frac{1}{g(\mathbf{x})}$, which is itself a monomial. In the case where any $f_i(\mathbf{x})$ is a generalized posynomial, the optimization program is said to be a *generalized geometric program* (GGP). All generalized geometric programs can be converted into standard geometric programs and solved efficiently.

Note that GP in its standard form is non-convex, because posynomials in general are non-convex functions. In order to apply general convex optimization methods, a GP is usually transformed into its convex form through logarithmic change of variables and multiplicative constants as follows. Let $y_i = \log x_i$ so that $x_i = e^{y_i}$, the standard form GP can be transformed into its equivalent convex form,

$$\begin{aligned} \text{minimize} \quad & \log f_0(e^{\mathbf{y}}) \\ \text{subject to} \quad & \log f_i(e^{\mathbf{y}}) \leq 1, \ i = 1, \ldots, m, \\ & \log g_j(e^{\mathbf{y}}) = 1, \ j = 1, \ldots, p. \end{aligned} \tag{2.15}$$

In particular, a monomial constraints

$$g_j(\mathbf{x}) = d_j \prod_{k=1}^{n} x_k^{a_{jk}} = 1 \tag{2.16}$$

is converted to

$$\log g_j(e^{\mathbf{y}}) = \log d_j + \sum_{k=1}^{n} a_{ik}\, y_k = 0, \tag{2.17}$$

which is affine and convex, while a posynomial constraint

$$f_i(\mathbf{x}) = \sum_{k=1}^{K_i} c_{ik} x_1^{a_{ik}^{(1)}} x_2^{a_{ik}^{(2)}} \dots x_n^{a_{ik}^{(n)}} \leq 1, \tag{2.18}$$

is converted to

$$p_i(\mathbf{y}) = \log \sum_{k=1}^{K_i} \exp(\mathbf{a}_{ik}^T \mathbf{y} + \log c_{ik}) \leq 0, \tag{2.19}$$

where $\mathbf{a}_{ik} = [a_{ik}^{(1)} a_{ik}^{(2)} \dots a_{ik}^{(n)}]$ is the vector of exponents. Since $p_i(\mathbf{y})$ is a log-sum-exp function, which is shown to be convex, all the constraints turn out to be convex after the transformation. Therefore, although the original standard formulations of GPs are nonlinear and non-convex, they can be converted into convex form as in (2.15) and solved efficiently with standard convex program solution techniques.

Chapter 3

Joint Coding and Scheduling Optimization with Varying Metrics

Now we are ready to investigate the coding and scheduling optimization in our motivating scenario of typical home network settings. The home network we consider is generally configured as follows. Multiple user networking devices, such as computers, media players and gaming consoles, are wirelessly connected to one or more WiFi access points (AP). These access points are then linked with a home gateway to the Internet. All the flow of packets from the Internet to the user devices goes through the gateway and the access points. The applications running on different devices may have very different delay sensitivities and constraints, as discussed before. The gateway and the access points look for the optimal coding and scheduling parameters to ensure the QoE of all the users devices within the network.

3.1 Single Broadcast Channel With Packet Erasures

First, we consider the case where is only a single WiFi access point and a single broadcast channel connecting all user devices. This is the typical scenario with most of the existing small home networks.

3.1.1 System Model

Conceptually, we represent the system using the following model. We assume that the link between the AP and gateway has a high capacity and is lossless. It is reasonable as the bottleneck of such system is usually the wireless connection between AP and the user devices. Furthermore, in many cases, the gateway and the access point is one integrated device itself. Consequently, we represent both the gateway and the AP together as a single node s. We denote the set of receivers by $T = \{t_1, \cdots, t_M\}$. Each receiver needs to obtain a flow of packets from some source over the Internet. Let $\mathcal{F} = \{f_1, \cdots, f_M\}$ be the set of flows, where f_i is the packet flow requested by receiver t_i. Note that all f_i enter the system from node s, which in turn acts as a source node in our model. The flows for different receivers are assumed to be independent. Hence, we have a single source multiple unicast session in the network. The original data packets in each flow are numbered, with $P_j^{f_i}$ representing the j-th packet in flow f_i. We assume that there are always enough packets to be served for each flow, since that is the case when there is a heavy traffic condition. Furthermore, all packets are assumed to have the same size L in the system and the system is time slotted. As in the previous chapter, at every time slot, the node s is able to broadcast a size L packet to all receivers, through the packet erasure broadcast channel. Erasures happen independently across all receivers and all time slots, i.e. the channel is memoryless. We denote the erasure vector by $\mathbf{e} = [\varepsilon_1 \cdots \varepsilon_M]$, where ε_i represents the erasure probability seen by receiver t_j. Figure 3-1 gives an illustration of the system model in the discussion. Note that the transmission is wireless broadcast, which means that receivers are able to overheard packets destined to other receivers.

Scheduling Strategies

Most of the works we discussed in Chapter 1 focus on linear network codes for multicast, in which all the receivers request the same content from the sources and any coding is done within the same session. In the system we consider here, however, we have a multiple unicast scenario, as each sink looks to receive its own flow, independently from others. Unlike the

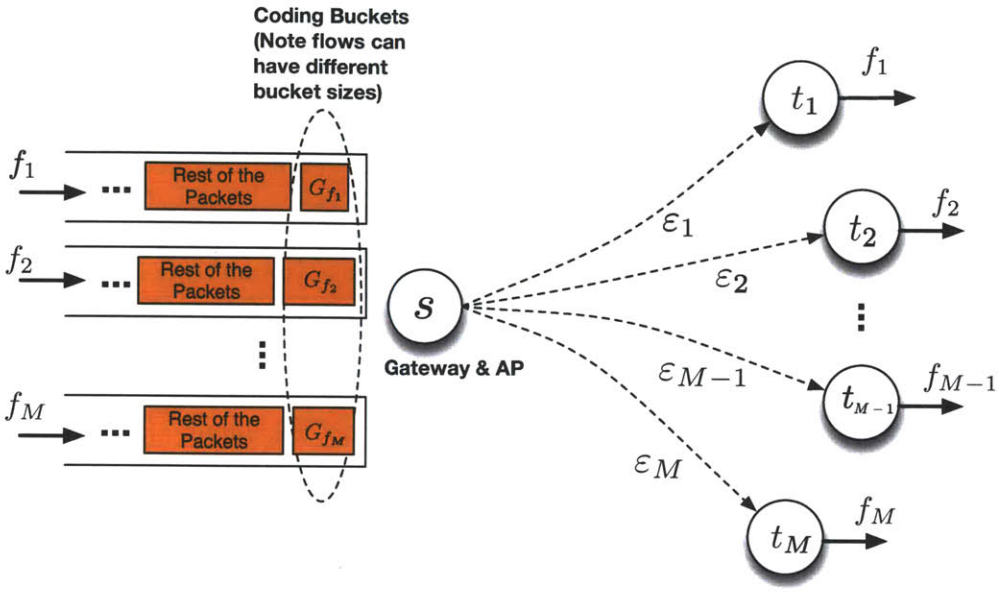


Figure 3-1: System model with single transmitter s and M receivers $\{t_1, \ldots, t_M\}$. Flow f_i is requested by t_i, while G_{f_i} represents the coding bucket for flow f_i.

simple scenario described in the previous chapter, the resource at node s has to be shared among all the receivers. In this chapter, only intra-session coding is considered. As a result, for every time slot, the sender s has to make a decision on which receiver to transmit to. It will then broadcast the encoded packet for that particular receiver. Moreover, if no inter-session coding is present, a receiver will discard any overheard packets which are targeted to some other receiver, since these packets are not useful for him. In this case, some scheduling algorithm is necessary to determine which receiver to serve for each time slot. While many sophisticated scheduling algorithms are available, for simplicity, we use a simple stochastic scheduling algorithm as follows. At every time slot, the source node s serves receiver t_j or flow f_j with probability a_j, independently from of any other time slots. In the long run, equivalently, the transmitter node s is spending a_j portion of time serving receiver t_j. We call the vector $\mathbf{a} = (a_1, \cdots, a_M)$ the vector of *scheduling coefficients.*

Intra-session Coding

We use intra-session or intra-flow coding in this case with multiple receivers, i.e. each unicast flow is coded independently and separately from others. Within each flow, the adaptive coding scheme described in Section 2.2 is used. That means, the transmitter maintains a coding bucket for each of the flows in $\mathcal{F}$. The coding bucket sizes and scheduling coefficients, however, are determined by solving system-wise optimizations. That means that each packet flow will have a different bucket size in general, depending on the flow delay characteristics. Under this scheme, for a given time slot, if the transmitter scheduling algorithm decides to serve receiver t_j, it looks for packets in the coding bucket of flow f_j, and encodes these packets using random linear network codes. The coded packet is broadcast to all the receivers. With probability $1 - \varepsilon_j$, the targeted receiver t_j will receive it correctly. We are only concerned with whether the targeted receiver successfully receives the packet, since overheard packets are discarded. Note that we assume the coding coefficients are embedded in the header of the packet and the size is negligible compared to the size of the packet L. The coding bucket size for flow f_j is denoted as K_j. In general, $K_i \neq K_j$ for $i \neq j$. Furthermore, $K_j, \forall j$ may vary over time as the delay requirements at the receivers changes. Let $\mathbf{K} = (K_1, \cdots, K_M)$. We aim to optimize both $\mathbf{a}$ and $\mathbf{K}$, based on the varying delay constraints at the receivers. There constraints are expressed in terms of the ℓ_p norm delay cost function, introduced in Chapter 2.

3.1.2 Delay Optimization

We first consider the special case where there is only a single receiver, i.e. $M = 1$. Since there is no scheduling issue or system-wise fairness consideration in the case, it makes sense to minimize the delay cost function associated with the receiver. As there is no ambiguity of notation, we drop all the subscripts. It is easy to conclude that the packet transmission rate in this case is $1 - \varepsilon$ for the receiver and therefore the expected time for receiving K coded packets is $\frac{K}{1-\varepsilon}$. Subsequently, the ℓ_p-norm delay cost function minimization problem is given

as follows,

$$\text{minimize} \quad d(p) = \frac{\frac{K}{1-\varepsilon} + D}{LK^{1/p}} \tag{3.1}$$

$$\text{subject to} \quad 1 \leq K \leq K_{\max}, \tag{3.2}$$

where $K_{\max}$ is the maximum possible coding bucket size because of system or computation complexity constraints. The optimal block size K^* can be obtained by setting zero the gradient of the Lagrangian of objective function, i.e.,

$$K^* = \left(\frac{(1-\varepsilon)D}{p-1} \right)_{[1,K_{max}]}, \quad 0 < \varepsilon < 1, \tag{3.3}$$

where the subscript denotes the projection,

$$(x)_{[a,b]} \triangleq \min(\max(a, x), b).$$

Because of the obvious practical interpretation of $d(1)$ and $d(\infty)$, one of important aspect to understand may be the region of achievable pairs of $(d(1), d(\infty))$ in a two dimensional space, for optimizing $d(p)$ with different p values. First, let us consider lower bounds on the two delay metrics individually. For the average packet delay or expected completion, let $\hat{d}(1) \triangleq \inf d(1)$. This infimum can be obtained when $K \to K_{\max} = \infty$, and is given by,

$$\hat{d}(1) = \frac{1}{(1-\varepsilon)L}, \tag{3.4}$$

On the other hand, as $K \to 1$, a theoretical lower bound for the per-packet delay $d(\infty)$ is obtained as follows,

$$\hat{d}(\infty) = \frac{1}{(1-\varepsilon)L} + \frac{D}{L}. \tag{3.5}$$

Therefore, the achievable region on a two-dimensional space of $(d(1), d(\infty))$ is lower bounded by the lines $d(1) = \hat{d}(1)$ and $d(\infty) = \hat{d}(\infty)$. Furthermore, consider the relation between every

achievable pairs of $d(1)$ and $d(\infty)$. From (2.5), we have,

$$K = \frac{D(1-\varepsilon)}{(1-\varepsilon)Ld(1)-1}. \tag{3.6}$$

Since $d(1) = \frac{d(\infty)}{K}$, the trade-off between $d(1)$ and $d(\infty)$ can be expressed as follows,

$$d(\infty) = \frac{D}{L - \frac{1}{d(1)(1-\varepsilon)}}. \tag{3.7}$$

Therefore, $d(1)$ and $d(\infty)$ have a inverse relationship. Ignoring the bucket size constraints for simplicity, given D, we can also vary K from 1 to ∞, and plot the values of $d(\infty)$ against $d(1)$ for the trade-off curve. Each point on the curve corresponds to a choice of K, which is equivalent to a choice of optimizing $d(p)$ for some p, because of (3.3). Therefore, the choice of p at the receiver indicates a point on the trade-off curve of $d(1)$ and $d(\infty)$ that is desired by the receiver.

The ℓ_p-norm delay optimization can be also view from the perspective of geometric programming. With the zero duality gap in GP, we can obtain the optimal $d(p)$ of the unconstrained problem (3.1) directly from the dual function by solving linear equations.

In general consider any posynomial $u(\mathbf{t})$ of variable $\mathbf{t} = [t_1 t_2 \dots t_m]$,

$$u(\mathbf{t}) = \sum_{i=1}^{n} u_i(\mathbf{t}) = \sum_{i=1}^{n} c_i \prod_{j=1}^{m} t_j^{a_{ij}}.$$

Let $u_i(\mathbf{t}) = \beta_i v_i(\mathbf{t})$, such that $\sum_i \beta_i = 1$. Then, from the inequality of Arithmetic and Geometric Mean, we have,

$$u(\mathbf{t}) = \sum_{i=1}^{n} u_i(\mathbf{t}) \geq \prod_{i=1}^{n} v_i(\mathbf{t})^{\beta_i} \tag{3.8}$$

$$= \prod_{i=1}^{n} \left(\frac{u_i(\mathbf{t})}{\beta_i}\right)^{\beta_i} \tag{3.9}$$

$$= \prod_{i=1}^{n} \left[\left(\frac{c_i}{\beta_i}\right)^{\beta_i} \left(\frac{u_i(\mathbf{t})}{c_i}\right)^{\beta_i}\right] \tag{3.10}$$

$$= \left(\frac{c_1}{\beta_1}\right)^{\beta_1} \left(\frac{c_2}{\beta_2}\right)^{\beta_2} \cdots \left(\frac{c_n}{\beta_n}\right)^{\beta_n} t_1^{D_1} t_2^{D_2} \ldots t_m^{D_m} \tag{3.11}$$

$$= V(\beta) \cdot t_1^{D_1} t_2^{D_2} \ldots t_m^{D_m}, \tag{3.12}$$

where $D_j = \sum_i a_{ij} \cdot \beta i \geq 0$, $V(\beta)$ is called the *dual* function while the last line gives the *pre-dual* function of the posynomial. By setting the partial derivatives of the pre-dual function to zero, it is easy to show that the maximum value the lower bound (i.e. the maximum value of the pre-dual function) is achieved when the β_i satisfies both *normality constraints* and *orthogonality constraints*, i.e.

$$\beta_1 + \beta_2 + \cdots + \beta_n = 1 \tag{3.13}$$

$$a_{1j} \cdot \beta_1 + a_{2j} \cdot \beta_2 + \cdots + a_{nj} \cdot \beta_n = 0, \qquad j = 1, 2, \ldots m. \tag{3.14}$$

Moreover, the greatest lower bound is always tight, if there are β_j that satisfy these conditions. In this case, the minimum value of the original posynomial is given by

$$V(\beta^*) = \left(\frac{c_1}{\beta_1^*}\right)^{\beta_1^*} \left(\frac{c_2}{\beta_2^*}\right)^{\beta_2^*} \cdots \left(\frac{c_n}{\beta_n^*}\right)^{\beta_n^*}, \tag{3.15}$$

where $\beta^* = [\beta_1^* \ldots, \beta_n^*]$ is the value of $\beta = [\beta_1 \ldots, \beta_n]$ that satisfies both conditions in (3.14) and gives the maximum value to $V(\beta)$.

Now, recall the delay minimization by (3.1), its dual function is given as,

$$V(\beta) = \left(\frac{1}{(1-\varepsilon)L\beta_1}\right)^{\beta_1} \left(\frac{D}{L\beta_2}\right)^{\beta_2}. \tag{3.16}$$

The maximum of the dual function, according to the (3.14), can be obtained at $\beta^* = (\beta_1^*, \beta_2^*)$, which is the solution to the simple linear system,

$$\begin{cases} (1-1/p)\beta_1 + (-1/p)\beta_2 &= 0, \quad \text{(normality condition)} \\ \beta_1 + \beta_2 &= 1, \quad \text{(orthogonality condition)} \end{cases} \tag{3.17}$$

Note that this system always has a unique solution. The approach can be easily generalized to the case where K is constrained [30].

3.1.3 Delay Constrained Optimization with GP

GP Formulation

For $M > 1$, in general, the delay cost functions of multiple receivers cannot be optimized at the same time. On the other hand, optimizing the delay function for some specific receiver may not be a fair objective. Therefore, instead of delay optimization, we are interested in optimizing certain system-wise utility function under the constraints that the ℓ_p-norm delay requirements must be satisfied at each receiver. We assume that each receiver t_j monitors the delay constraints for targeted QoE of its applications and sets a maximum acceptable delay $\hat{d}(p_j)$, corresponding to some delay sensitivity p_j for its performance requirements. For the objective function, we choose to maximize the *min rate* of all receivers for simplicity and fairness considerations.

First, consider the packet transmission rate from the sender s to a receiver t_j. If the transmission rate is r_j packet per slot, then the actual average data rate received by t_j is $\frac{LK_j}{\frac{K_j}{r_j}+D_j}$, where D_j is the feedback delay from t_j to s. Moreover, since the portion of time s serving t_j is a_j and the maximum transmission rate for t_j, when s serves t_j only, is $1-\varepsilon_j$, r_j must be upper bounded by $a_j(1-\varepsilon_j)$. Let $\mathbf{r} = (r_1, \cdots, r_M)$. The list of variables and constants and their definitions in this case are summarized in Table 3.1. The optimization can be formulated as follows:

$$\max_{\mathbf{K},\mathbf{r},\mathbf{a}} \quad \min_j \frac{LK_j}{\frac{K_j}{r_j}+D_j} \tag{3.18}$$

$$\text{subject to} \quad \frac{\frac{K_j}{r_j}+D_j}{LK_j^{1/p_j}} \le \hat{d}_j(p_j) \qquad \forall j = 1,\ldots,M \tag{3.19}$$

$$r_j \le a_j(1-\varepsilon_j) \qquad \forall j = 1,\ldots,M \tag{3.20}$$

$$\sum_j a_j \le 1 \tag{3.21}$$

$$1 \leq K_j \leq K_{max} \qquad \forall j = 1, \ldots, M. \tag{3.22}$$

In this formulation, constraints (3.19) and (3.20) represent the delay and rate constraints respectively for receiver t_j, while (3.21) is the scheduling probability constraint at the sender node s. The solution of the problem, if exists, will provides the coding bucket sizes of all flows, $\mathbf{K}$ as well as the scheduling coefficients, $\mathbf{a}$. These allow the transmitter s to dynamically adapt its coding and scheduling strategy based on the variations of $\hat{\mathbf{d}}(\mathbf{p})$ at the receivers, and therefore accommodate delay sensitive application better, as illustrated later this chapter.

Variables/Constants	Definition
M	Number of receivers
L	Packet size: all packets are assume to have the same size
K_j, $\mathbf{K}$	Coding bucket size for receiver t_j, $\mathbf{K} = [K_1 \ldots, K_M]$
$K_{\max}$	Maximum possible coding bucket size for the system
a_j, $\mathbf{a}$	Portion of time for serving receiver t_j, $\mathbf{a} = [a_1 \ldots a_M]$
ε_j	Erasure probability between s and t_j
r_j, $\mathbf{r}$	Packet transmission rate for receiver t_j. $\mathbf{r} = [r_1 \ldots r_M]$

Table 3.1: Summary of variables, constants and terms for the single broadcast channel case

For the solution techniques of the optimization program, it is straightforward to verify that the problem is a generalized geometric program. All constraints can be converted into upper bound of posynomials of $\mathbf{K}, \mathbf{r}$ and $\mathbf{a}$ in the standard form shown in (2.14). For constraints (3.19) and (3.21), we have the standard form as,

$$(\hat{d}_j(p_j))^{-1} r_j^{-1} L^{-1} K_j^{1-1/p_j} + D^{-1} L^{-1} K_j^{-1/p_j} \leq 1$$

$$a_j^{-1}(1-\varepsilon_j)^{-1} r_j \leq 1.$$

That leaves with only one non-posynomial part, the objective function. In the case of min rate maximization, the objective can be transformed into upper bounding posynomial

constraints and monomial objective by adding auxiliary variable x, such that,

$$\max_{\mathbf{K},\mathbf{r},\mathbf{a},x} \quad x \tag{3.23}$$

$$\text{subject to} \quad \frac{x(\frac{K_j}{r_j} + D_j)}{LK_j} \leq 1, \qquad \forall j. \tag{3.24}$$

Combing this with (3.19) to (3.22), we have a GP that can be efficiently solved. Many alternative objective function are also possible with this formulation. For example, the rate product maximiztion,

$$\min \quad \prod_j \frac{\frac{K_j}{r_j} + D}{LK_j},$$

or the weight sum of delay functions,

$$\max \quad \sum_j \omega_j d(p_j),$$

for some assigned weight $W = [\omega_1 \dots \omega_m]$. The maximization of the sum rate, $\sum_j \frac{\frac{K_j}{r_j}+D}{LK_j}$, however, is a posynomial maximization, which cannot be converted to GGP and belongs to a class of more general problem called *signomial programming*. We defer the discussion to the next section.

3.1.4 Illustrations of trade-offs and advantages

Trade-off: average delay versus per-packet delay

First, consider the delay optimization for a single receiver system. Figure 3-2 demonstrate the trade-off between $d(1)$ and $d(\infty)$ following Equation (3.7) with various values of D and erasure probability $\varepsilon = 0.4$ in the single receiver case. As discussed previously, if we parameterize $d(1)$ and $d(\infty)$ on the optimal bucket size K^*, as p varies from 1 to ∞, we obtain the same curves. The shaded area bounded by each curve is the area of all achievable pairs $(d(1), d(\infty))$ for the specific feedback delay. With small D, both low delay in $d(1)$ and

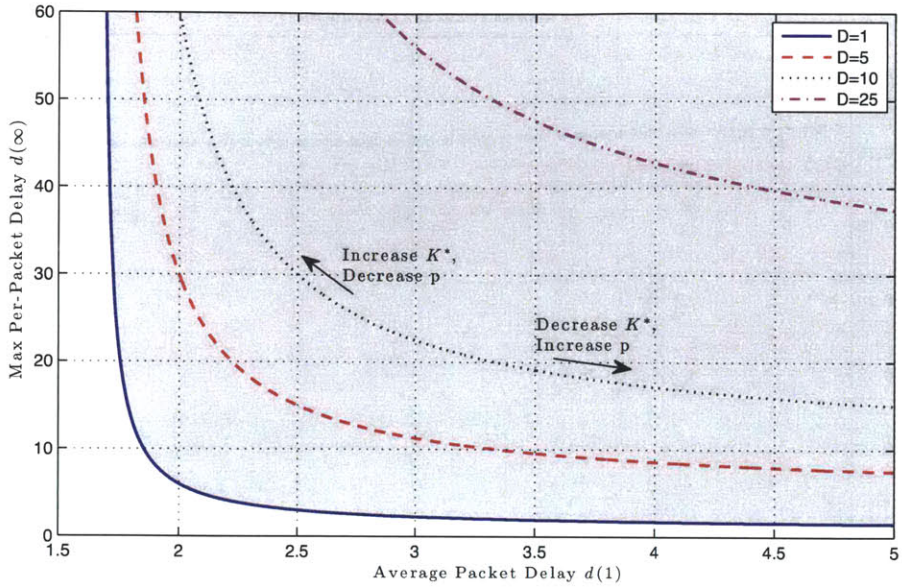


Figure 3-2: Tradeoff of $d(1)$ vs $d(\infty)$ with varying feedback delay D. The shaded region represents all the achievable delay pair $(d(1), d(\infty))$. Moving from left to right on the optimal trade-off curve for fixed D, we are decreasing bucket size K and increasing the delay sensitivity p.

$d(\infty)$ can be achieved, as shown be the blue curve. However, when feedback delay increases, the trade-off becomes increasingly stronger. This is evident from Equation (3.7), where D appears in the numerator. It is expected, since for average delay, coding over larger bucket sizes amortizes the feedback delay over more packets. On the contrary, to obtain small per-packet delay $d(\infty)$, increased feedback delay must be compensated by even smaller generation size or coding bucket size for more frequent decoding. This is also consistent with Equation (3.3) where K^* increases with feedback delay D and decreases as delay sensitivity p.

Adaptive Scheme vs Fixed Generation Coding

Next, consider the delay constrained min rate maximization for multiple receiver systems. Figure 3-3 to 3-5 shows some comparisons between adaptive coding schemes with fixed generation size coding schemes, as the delay sensitivity p_1 of the first receiver increases. In this example, we have 5 receivers, with erasure $\mathbf{e} = [0.4, 0.1, 0.15, 0.2, 0.25]$, the same $D = 5$,

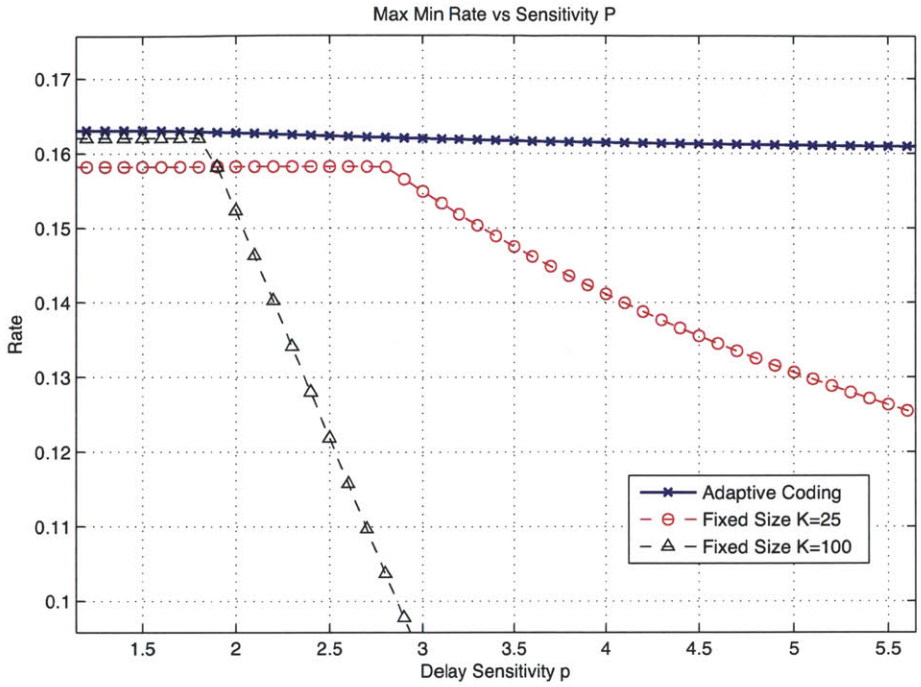


Figure 3-3: Min rate vs delay sensitivity p_1 of different coding methods for an example system with 5 receivers, $D = 5, L = 1$ and erasures $\mathbf{e} = [0.4, 0.1, 0.15, 0.2, 0.25]$. In the case of $K = 100$ and $K = 25$, the min rate drops considerably beyond certain threshold p_1, and the optimization is infeasible before $p_1 = 2.9$, while with the adaptive scheme, the min rate drops much less as p_1 increases.

$L = 1$ and $\hat{d}_j = 50/L$. Except for receiver 1, whose p_1 value varies, we have $p_j = 1$ for all other receivers. For the fixed generation size schemes, we choose $K = 25$ and $K = 100$ for representing small and large generation respectively. In these cases, scheduling coefficients are the only decision variables in the optimization for min rate.

From Figures 3-3 and 3-4, we can see that, initially at low values of p, the min rates for different schemes are relatively close. Because of the low delay sensitivity level, the adaptive scheme is able to choose a large coding bucket size to obtain some rate gain with respect to the $K = 25$ case. As p_1 increases, the fixed coding generation schemes are unable to reduce generation size. In order to meet the growingly stringent delay constraint at the first receiver, the sender has to devote increasingly more time to receiver t_1, as seen in Figure 3-5. Inevitably, the portion of time for serving other receivers is greatly reduced and the min

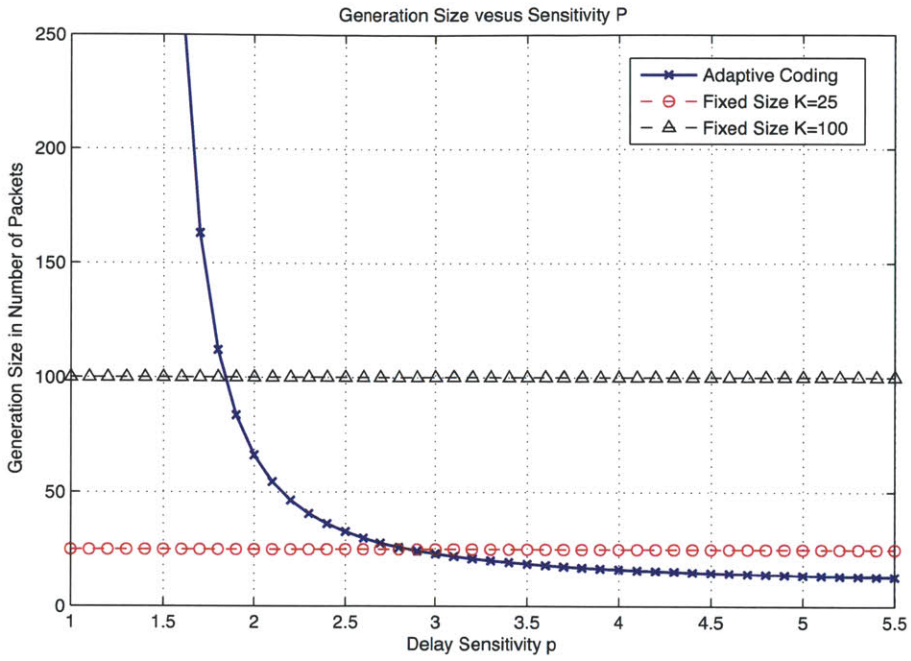


Figure 3-4: Coding bucket size vs delay Sensitivity p_1 of different coding methods for an example system with 5 receivers. $D = 5, L = 1$ and erasures $\mathbf{e} = [0.4, 0.1, 0.15, 0.2, 0.25]$. The adaptive scheme reduces the coding bucket size accordingly, as delay sensitivity increases, to meet the delay constraints.

rate of the system decreases sharply. In the case of $K = 100$, the delay requirements cannot be satisfied for $p_1 > 2.9$, and the optimization becomes infeasible. On the contrary, for the adaptive scheme, which optimizes bucket size and scheduling jointly, there is little decrease in min rate. For low delay sensitive receivers t_2 to t_5, the scheme will assign them large coding bucket sizes to allow rate gain. As a result, the sender is able to meet their delay-rate constraints with less serving time and save time for higher receivers. On the other hand, as p values for some receiver increase, coding bucket size is reduced to quickly decreases the per-packet delay. Hence, the scheme is able to accommodate high delay sensitive receivers much better than the fix generation size schemes.

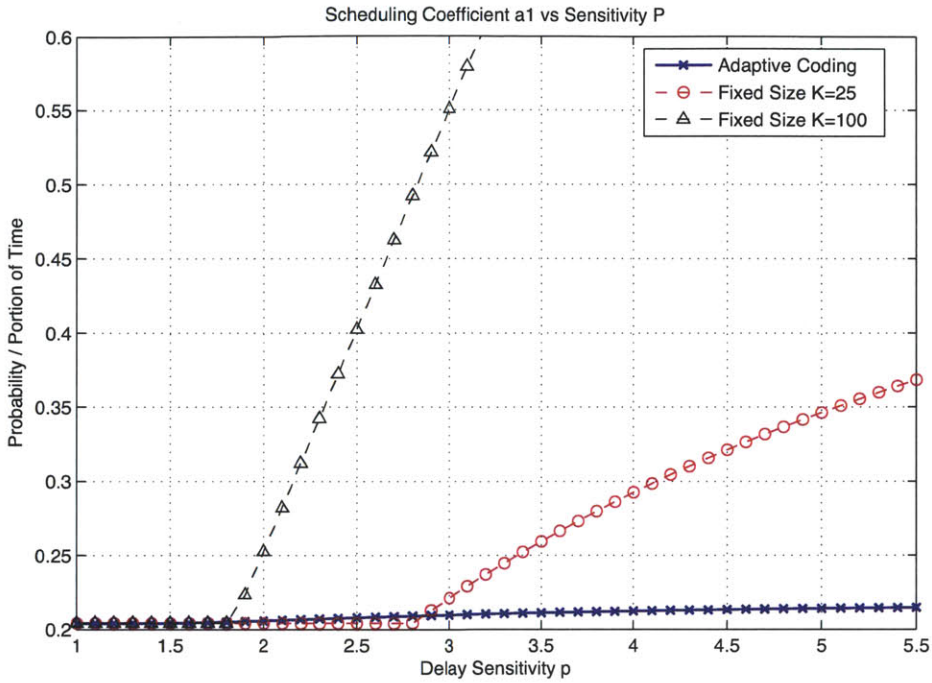


Figure 3-5: a_1 vs Delay Sensitivity p_1 of different coding methods for an example system with 5 receivers. $D = 5, L = 1$ and erasures $\mathbf{e} = [0.4, 0.1, 0.15, 0.2, 0.25]$. In fixed generation size schemes, the service time a_1 has to rapid increase in compensation for large generation size to meet the increasing tight delay constraints, while service time a_1 remains relatively constant for the adaptive scheme.

3.2 Multiple Non-interfering Wireless Packet Erasure Channels

With the proliferation of low cost gateway and access points, many devices may be covered by more than one access points in wireless home, campus or enterprise networks. This motivates us to extend of the approach in the previous section to the case of multiple broadcast erasure channels covering the same set of receivers. As in the previous section, we still have the same set of receivers, $T = \{t_1, \cdots, t_M\}$. However, there are now W access points, denoted by the set $S = \{s_1, \cdots, s_W\}$, each transmitting packets to the receiver through a separate broadcast channel. For simplicity, we assume that these broadcast channels are orthogonal or non-interfering. The assumption is justified if the WiFi access points use

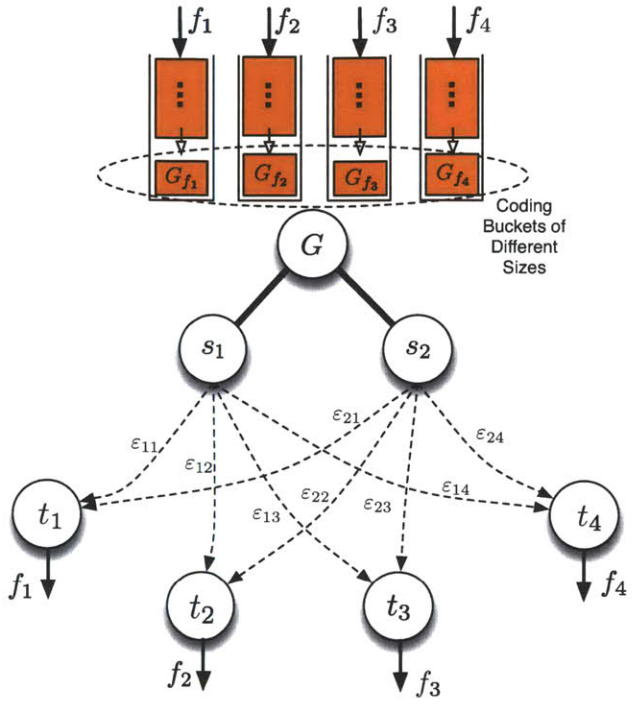


Figure 3-6: An example system with 2 senders and 4 receivers. The broadcast channels associated with s_1 and s_2 do not interfere, while all receivers are connected to both channels simultaneously.

different frequency bands in their respective downlink broadcast channels. In this system, instead of an erasure probability vector, we have an erasure probability matrix $\mathbf{e} = [\varepsilon_{ij}]$, where ε_{ij} is the erasure probability between node s_i and t_j. Furthermore, we also assume that each device is simultaneously connected to and receive packets from all W access points, i.e. has Multi-Packet Reception (MPR) capability. An example of the system is illustrated in Figure 3-6. Note that the access points may be associated with different heterogeneous networks, such as WiFi and cellular networks, with different link characteristics.

The same coding and scheduling scheme is used for the new system. We use $\mathbf{a} = [a_{ij}]$ to represent the probability of transmitter s_i serving the flow f_j at any time slot. The scheduling and coding optimization is done at the gateway node G, who coordinates all senders who perform encoding. Furthermore, for each flow f_j in $\mathcal{F}$, all senders in S have the same coding

bucket size K_j, dictated by node G. This ensures that, for each flow, every sender sends coded packets in the coding buckets consisting of the same data packets and guarantees the decodability. An important feature of the randomly linear coding is that all coded packets from the same bucket are exchangeable. That avoids complicated scheduling based on sequence numbers of the uncoded packets and helps to reduce transmission redundancy in erasure channels.

3.2.1 Signomial Program Formulation

Similarly to the single sender case, we can formulate an optimization program for determining $\mathbf{K}, \mathbf{a}$ and $\mathbf{r}$. For example, the rate product maximization is given as,

$$\min_{\mathbf{K},\mathbf{r},\mathbf{a},\mathbf{R}} \prod_j R_j^{-1} \tag{3.25}$$

$$\text{subject to } \frac{\frac{K_j}{r_j} + D_j}{LK_j^{1/p_j}} \leq \hat{d}_j \qquad \forall j = 1, \ldots, M \tag{3.26}$$

$$r_j \leq \sum_i a_{ij}(1 - \varepsilon_{ij}) \qquad \forall j = 1, \ldots, M \tag{3.27}$$

$$R_j \leq \frac{LK_j}{\frac{K_j}{r_j} + D_j} \qquad \forall j = 1, \ldots, M \tag{3.28}$$

$$1 \leq K_j \leq K_{max} \qquad \forall j = 1, \ldots, M \tag{3.29}$$

$$\sum_j a_{ij} \leq 1 \qquad \forall i = 1, \ldots, W, \tag{3.30}$$

where the delay constraint (3.26) and complexity constraint (3.29) remain the same as in (3.19) and (3.22). Auxiliary variables R_j and (3.28) are used to represent the average packet transmission rates for the receiver. Maximizing rate product is equivalent to minimizing the product of average delays R_j^{-1}, hence the objective $\prod_j R_j^{-1}$.

The packet transmission rate for each receiver in this case is bounded by $\sum_i a_{ij}(1 - \varepsilon_{ij})$. However, owing to the existence of this new transmission rate constraint (3.27), the problem

becomes truly non-convex. In particular, the constraint can be written as

$$r_j + \sum_i (-a_{ij})(1 - \varepsilon_{ij}) \leq 0, \tag{3.31}$$

which is an upper bound constraint on a signomial. A signomial is a sum of monomials whose multiplicative coefficients can be either positive or negative. The problem therefore belongs to a more general class of problem called *Signomial Program*, which is truly non-convex and NP-hard in general. Only local optimal solutions can be efficiently computed. Based on the most widely used monomial condensation methods, we provide an efficient way to approximate the solution with successive GP solutions.

3.2.2 Successive GP Approximation

Consider an arbitrary signomial $h(\mathbf{x})$. It can always be written as the difference between two posynomials, i.e. $h(\mathbf{x}) = f^+(\mathbf{x}) - f^-(\mathbf{x})$, where $f^+(\mathbf{x})$ contains the terms with positive coefficients, and $-f^-(\mathbf{x})$ contains the therms with negative coefficients in $h(\mathbf{x})$. As a results, the inequality $h(\mathbf{x}) \leq 0$ is then equivalent to

$$\frac{f^+(\mathbf{x})}{f^-(\mathbf{x})} \leq 1. \tag{3.32}$$

We can approximate the left hand side of this inequality with a posynomial using common *condensation methods* [29].

In the *single condensation method*, the posynomial denominator $f^-(\mathbf{x})$ is approximated using a monomial $g^-(\mathbf{x})$, which in turn allows $\frac{f^+(\mathbf{x})}{f^-(\mathbf{x})}$ to be approximated by $\frac{f^+(\mathbf{x})}{g^-(\mathbf{x})}$, which is a posynomial itself, since it is generated from dividing a posynomial by a monomial. In the *double condensation method*, both f^+ and f^- are approximated using monomials, which creates a monomial approximation of $\frac{f^+(\mathbf{x})}{f^-(\mathbf{x})}$. In our case, both methods are equivalent, since we have $f^+(\mathbf{x}) = r_j$, which is itself a monomial. One of the commonly used monomial approximation of posynomial is based on the following Lemma [29].

Lemma 1 *Given a posynomial* $f(\mathbf{x}) = \sum_i u_i(\mathbf{x})$, *choose* $\beta_i > 0$, *such that* $\sum_i \beta_i = 1$, *then*

the following bound holds,

$$f(\mathbf{x}) \geq g(\mathbf{x}) = \prod_i \left(\frac{u_i(\mathbf{x})}{\beta_i} \right)^{\beta_i}. \tag{3.33}$$

Furthermore, equality holds when $\mathbf{x} = \mathbf{x}_0$ *and* $\beta_i = \frac{u_i(\mathbf{x}_0)}{f(\mathbf{x}_0)}$.

Proof: The results can be easily proved using the Inequality of Arithmetic and Geometric Mean, similarly to (3.12). Let $u_i(\mathbf{x}) = \beta_i \cdot v_i(\mathbf{x})$, such that $\sum_i \beta_i = 1$, $\beta_i > 0$. From the AM-GM inequality, we have,

$$\sum_i u_i(\mathbf{x}) = \sum_i \beta_i v_i(\mathbf{x}) \geq \prod_i \left(v_i(\mathbf{x})\right)^{\beta_i}.$$

Therefore, $f(\mathbf{x}) \leq g(\mathbf{x})$, $\forall \mathbf{x}$. The equality part can be easily verified by setting $\beta_i = \frac{u_i(\mathbf{x}_0)}{f(\mathbf{x}_0)}$.

Using Lemma 1, we can approximate constraint (3.27) in the signomial program with the following,

$$r_j \leq \prod_i \left(\frac{a_{ij}(1-\varepsilon_{ij})}{\beta_{ij}} \right)^{\beta_{ij}}. \tag{3.34}$$

In particular, when we substitute constraints (3.27) with (3.34), the resulting optimization program is a geometric program. Furthermore, given the monomial approximation in (3.34), we can construct successive GP based on refined approximations of constraint (3.27) to approach local optimal solutions of the original signomial problem.

Let t be the iteration index of the successive approximations and $(K^t, \mathbf{a}^t, \mathbf{r}^t, \mathbf{R}^t)$ be the solution obtained by solving the GP approximation of the SP program at the t-th iteration. Assume that some feasible solution $(K^0, \mathbf{a}^0, \mathbf{r}^0, \mathbf{R}^0)$ is known initially. At iteration $t+1$, the algorithm computes β_i, such that the constraint (3.34) is tight at the solution obtained in the previous iteration, i.e. the new β_i obtained allow (3.34) to be satisfied with equality. This is achieved because β_i are computed based on the equality condition given in Lemma 1. The constraints (3.34) is then reconstructed with the new β_i values. The resulting GP is solved again to obtain an approximated solution at iteration $t+1$. The process iterates until some convergence criterion is reached. The detailed algorithm is summarized in Algorithm 1 and it will output a local minimum solution that fulfills the Karush-Kuhn-Tucker (KKT)

Algorithm 1: Successive GP Approximation of SP

Begin: A feasible solution $(\mathbf{K}^0, \mathbf{a}^0, \mathbf{r}^0, \mathbf{R}^0)$, $t = 0$;
repeat
Compute $f(\mathbf{a}^t) = \sum_i a_{ij}^t(1 - \varepsilon_{ij})$;
Compute $\beta_{ij} = \frac{a_{ij}^t(1-\varepsilon_{ij})}{f(\mathbf{a}^t)}$;
Construct the t-th approximation and replace constraint (3.27) with the monomial constraint,

$$r_j \leq g(\mathbf{a^t}) = \prod_i \left(\frac{a_{ij}^t(1 - \varepsilon_{ij})}{\beta_{ij}} \right)^{\beta_{ij}}$$

$t = t + 1$;
Solve the resulting GP to get $(\mathbf{K}^t, \mathbf{a}^t, \mathbf{r}^t, \mathbf{R}^t)$;
until *Convergence*;

Condition.

The condensation methods and successive GP approximation also provide the possibility of choosing a wider variety of optimizable objective functions, for finding local optimums, including the sum rate maximization, for both single and multiple broadcast packet erasure channel cases. In sum rate maximization, the objective function is given as,

$$\max \quad \sum_j \frac{\frac{K_j}{r_j} + D}{LK_j}.$$

By introducing auxiliary variables $\mathbf{R} = [R_1 \; R_2 \; \ldots R_M]$ representing the average rate of each receiver, and variable x for the sum rate, we can convert the original maximization objective into the following,

$$\max_{\mathbf{K},\mathbf{r},\mathbf{a},x,\mathbf{R}} \quad x \tag{3.35}$$

$$\text{subject to} \quad \sum_j R_j \leq x \tag{3.36}$$

$$R_j \leq \frac{\frac{K_j}{r_j} + D_j}{LK_j}, \qquad \forall\, j. \tag{3.37}$$

In particular, the new objective function and the sum rate constraints (3.36) are both gen-

eralized GP constraints. We can apply exactly the same condensation and approximation method to the remaining constraint (3.37) to converted the sum rate optimization program into a series of standard GPs. Similar methods applies to a variety of objective functions, such as weighted sum rate.

3.2.3 Convergence

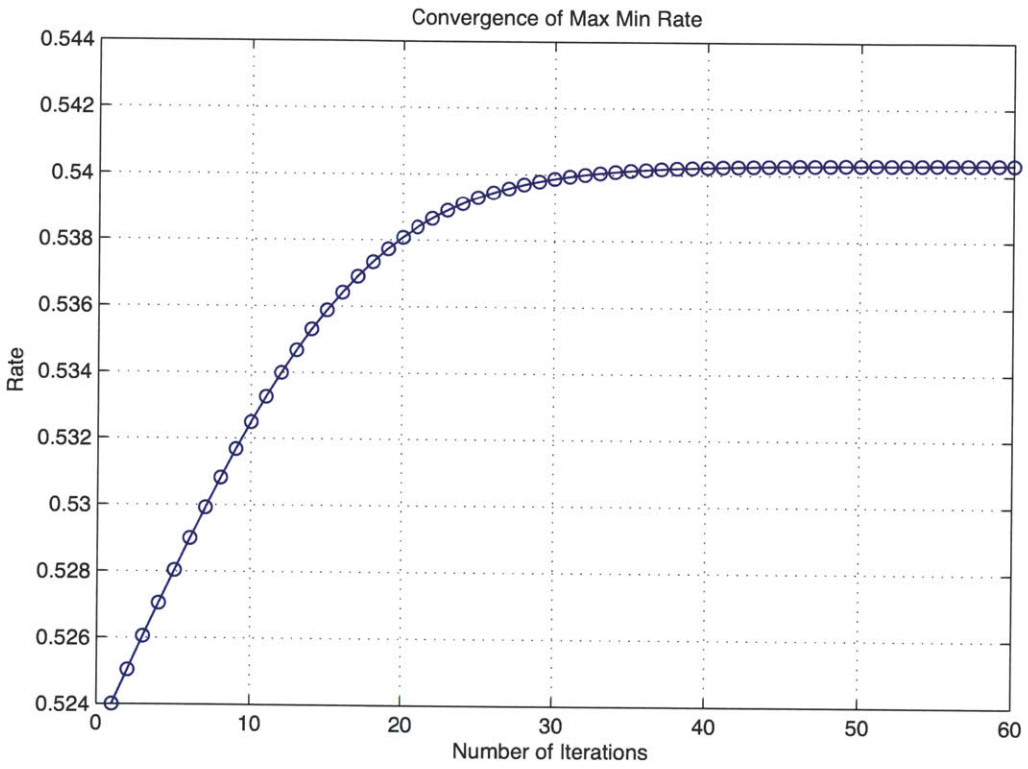


Figure 3-7: Convergence of maximum min rate in the example system, as the number of iterations increases. $p = [1.6\ 1.7\ 1.8]$, $D = 5, L = 1$,$\mathbf{e} = [0.2\ 0.1\ 0.15;\ 0.15; 0.2\ 0.25]$

Next, we show that the proposed Algorithm 1 always converges to some local optimum that meets the KKT condition, based on the results of Lemma 1:

1. First, according to Lemma 1, the values β_{ij} are chosen such that, for the local approx-

imation at $\mathbf{a}^t$ in the t-th iteration, we have,

$$g(\mathbf{a}^t) = f(\mathbf{a}^t) \geq f(\mathbf{a}^{t-1}). \tag{3.38}$$

The equality is from the fact that the local approximation at $\mathbf{a}^t$ is tight, while the inequality part follows from the rate constraint of the previous GP iteration.

2. Let the optimal objective for iteration t be $Z^{*,t}$. Then we have $Z^{*,t} \leq Z^{*,t-1}$.

3. Finally, at local optimal $\mathbf{a}^*$, it can be verified, that $f(\mathbf{a}^*) = g(\mathbf{a}^*)$ and $\nabla f(\mathbf{a}^*) = \nabla g(\mathbf{a}^*)$.

These shows that the algorithm will indeed converge to an optimal that satisfies the KKT condition [32]. In fact, in many cases, it may converge to the global optimum.

Figure 3-7 and 3-8 shows the convergence of min rate and bucket sizes for a example system with 3 receivers and 2 transmitters, $D = 5$ for all receiver, delay sensitivities $p =$ [1.6 1.7 1.8], while erasure probabilities $\mathbf{e} =$ [0.2 0.1 0.15; 0.15; 0.2 0.25]. Figure 3-9 shows the convergence of the scheduling coefficients to the local optimums, for the same system.

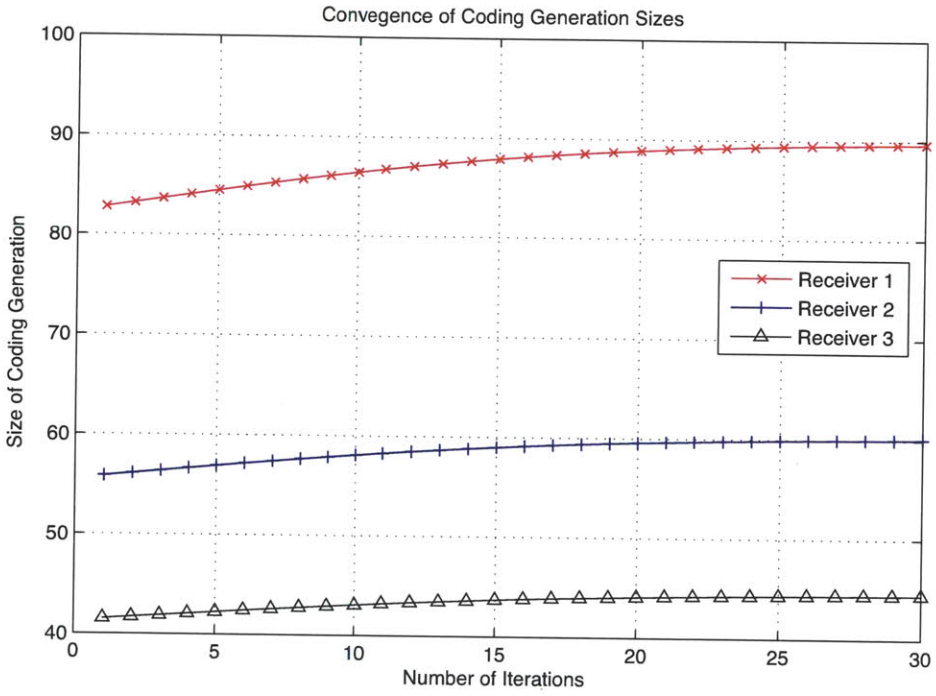


Figure 3-8: Convergence of bucket sizes in the example system as the number of iterations increases. Note that the bucket size variation is very quite small. $p = [1.6\ 1.7\ 1.8]$, $D = 5, L = 1$, $\mathbf{e} = [0.2\ 0.1\ 0.15;\ 0.15; 0.2\ 0.25]$

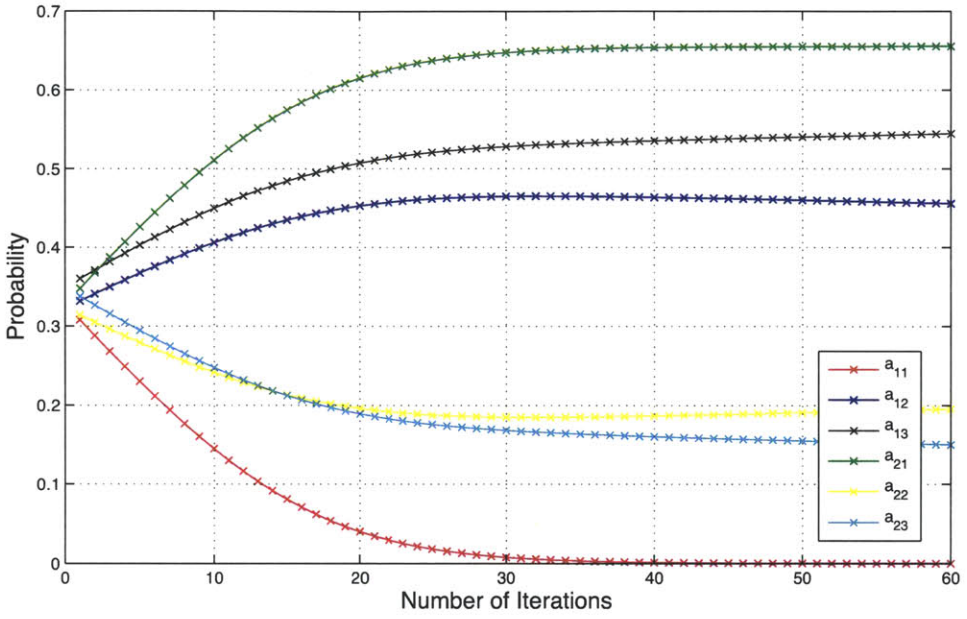


Figure 3-9: Convergence of scheduling coefficients the example system as the number of iterations increases. $p = [1.6\ 1.7\ 1.8]$, $D = 5, L = 1$, $\mathbf{e} = [0.2\ 0.1\ 0.15;\ 0.15; 0.2\ 0.25]$

Chapter 4

Inter-Session Coding in Short Feedback Delay Settings

4.1 Model and Assumptions

In this chapter, we consider again the single broadcast channel with packet erasures, as described in Section 3.1 and Figure 3-1. In the discussion of the previous chapter, we show that the trade-off between $d(1)$ and $d(\infty)$ is increasing strong, as the feedback delay D increases. When the D is very small, the rate gain from having long coding blocks is minimal. In this chapter, we consider the case when the feedback delay is insignificant compared to time slot length. In this case, the transmitter has the knowledge of the erasure pattern seen by all the receivers shortly after each transmission. Accordingly, the transmitter can be perform *inter-session coding* to reduce the total number of transmissions needed to deliver all the packets, or equivalently, to increase the average transmission rate for the receivers.

Perfect Immediate Feedback

Consider the same system described in Figure 3-1. A single source node s is looking to transmit independent packet flows f_i for receiver t_i through a broadcast channel with packet erasures. In this chapter, we make the following assumptions that are different from Chapter

3:

1. There is perfect immediate feedback available from any receiver t_i to the source node s. Equivalently, prior to a new transmission, the transmitter s has the knowledge of erasures happening at all receivers for all previous transmissions. While the perfect immediate feedback assumption may not be very practical, it is usually used for studying upper bounds of the performance of coding and scheduling algorithms. Furthermore, many wireless systems use the Stop-and-Wait medium access control (MAC) protocol, which essentially provides the transmitter with the knowledge of packet erasures before new transmissions.

2. The transmitter maintains the record of erasure information about all past transmissions for all receivers for making its inter-session coding decision at future transmissions.

3. A receiver does not discard overheard packets that are targeted to other receivers, until it receives all its requested packets. The overheard packets are used to decode inter-session coded packets. We start with assuming the buffer size at each receiver is the same and is finite. This assumption is relaxed later.

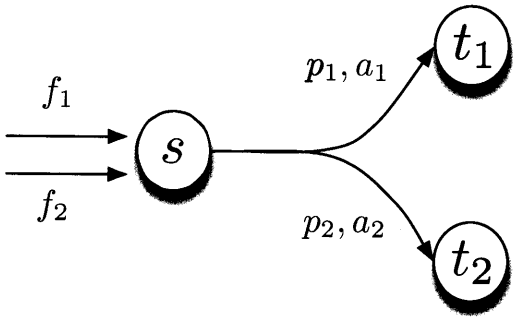


Figure 4-1: A packet erasure broadcast channel with 2 receivers

4.1.1 An Example

A simple example may give a clear illustration of the benefit of inter-session coding based on previous erasure. Figure 4-1 shows a packet erasure broadcast channel with only 2 receivers. Suppose the source node s intends to transmit packet A to t_1, and packet B to t_2. The transmitter broadcasts A and B respectively for the first 2 time slot. The results of the transmissions are shown in Table 4.1. The packet A targeted to t_1 at the first time slot is overheard at t_2 but erased at t_1, while B, targeted to t_2, is erased at t_2 but received at t_1. The overheard packets are buffered in the receivers. The transmitter is assumed to have the knowledge of erasures happen in the first two time slots. Therefore, in the third time slot, instead of transmitting A or B, the best strategy is to transmit $A \oplus B$, which will benefit both receivers and possibly reduce the total number of transmission if both receive it correctly, since $A \oplus B$ allows the decoding of A at receiver t_1 with the knowledge of B and similarly for receiver t_2.

Time slot	1	2	3
Transmitted packet	A	B	$A \oplus B$
Receiver t_1	✗	✓	✓
Receiver t_2	✓	✗	✓

Table 4.1: Transmission of packets in the first few time slots: "✓" indicates successfully received or overheard, "✗" indicates erased.

4.2 Outer bound on the capacity of packet erasure broadcast channels

Recent works show that feedback indeed expands the capacity region of broadcast channel with packet erasures, in contrast to AWGN point-to-point channels, where feedback does not increase capacity [33]. However, the capacity region of the erasure broadcast channel with feedback is in general unknown, except for certain special cases.

Rate Regions

No Feedback

Dana *et al.* [34] shows that, without feedback, the rate region of any multiple input broadcast erasure channel is achieved by time-sharing among the receivers at each available transmitter. Hence, a rate tuple for M receivers $\mathbf{R} = \{R_1, R_2, \ldots, R_M\}$ is achievable if and only if there exist a matrix $[\alpha_{ij}]$, satisfying:

$$0 \leq R_j \leq \sum_{i=1}^{m} \alpha_{ij}(1 - \varepsilon_{ij}) \qquad \forall j \tag{4.1}$$

$$\sum_{j} \alpha_{ij} = 1 \qquad \forall i \tag{4.2}$$

$$\alpha_{ij} \geq 0 \qquad \forall i, j. \tag{4.3}$$

From capacity region specified by (4.3), we conclude that, given a large enough field size q, a intra-session coding over $\mathbb{F}_q$, together with a time-sharing packet scheduling suffices to asymptotically achieve all rate tuples at the boundary capacity region of the multiple input broadcast packet erasure network with increasing block size.

With Perfect Immediate Feedback

When perfect immediate feedback is available, the problem becomes much more complicated, even for linear coding capacity, because of the combinatorial hardness of finding the optimal inter-session codes. There is little work on multiple input broadcast erasure channel with feedback. However, there is some very recent works, providing sets of outer and inner bounds on the capacity region of single input broadcast erasure channels under perfect feedback, as well as some suboptimal coding algorithms to achieve rate-tuples close to capacity region. One outer bound for the capacity region of the single input broadcast erasure channel is shown [24].

Let $\pi = \{\pi_1, \ldots, \pi_M\}$ be some permutation of the index set $\{1, 2, \ldots, M\}$. Recall that

$\{\varepsilon_1, \ldots, \varepsilon_M\}$ are the erasure probabilities to all the receivers. Let $\hat{\varepsilon}_{\pi_i} = \varepsilon_{\cup_{j=1}^{i}\pi_j}$, then,

$$\hat{\mathcal{C}}_{\pi,f} = \{\mathbf{R} \geq 0 \mid \sum_i \frac{R_{\pi_i}}{1 - \hat{\varepsilon}_{\pi_i}}\} \tag{4.4}$$

is the capacity region of a physically degraded broadcast erasure channel with erasure probability $\hat{\varepsilon}_{\pi_i}$ for receiver π_i. As shown in [35], feedback on a physically degraded broadcast channel do not expand the capacity region. Hence, it is also the capacity region of the same degraded broadcast erasure channel with feedback, which serves as an outer bound for the original single input broadcast erasure channel. Furthermore, let $\mathcal{P}$ be the set of all permutations, the capacity region $\mathcal{C}_f$ of the original channel with feedback, is outer bounded by,

$$\mathcal{C}_f \subseteq \cap_{\pi \in \mathcal{P}} \hat{\mathcal{C}}_{\pi,f}. \tag{4.5}$$

References [24] and [25] show that the bound is tight for symmetric channels as well as for $n \leq 3$ cases. Reference [25] further shows that the bound is also tight for spatially independent packet erasure broadcast channel with one-sided fairness constraints with arbitrary number of receivers.

Coding Algorithms

For the no feedback case, intra-session coding within each flow, combined with time-sharing packet scheduling achieves the capacity region of the channel with sufficiently large generation size. When perfect immediately feedback is available, intra-session coding is not sufficient to achieve the whole capacity region. Furthermore, it is unknown whether linear inter-session network codes suffice to achieve all points in the capacity region. Reference [24] provides some exhaustive book keeping strategies for coding packets to achieve the capacity outer bound for symmetric channel conditions and for $n \leq 3$ case. The Packet Evolution scheme proposed in [25] provides a inner bound for the capacity region, as shown in *Proposition 3* of [25]. Some more practical schemes are provided in [36], [37] and [38].

4.3 Pair-wise Opportunistic Coding

4.3.1 Coding Method

We consider a suboptimal pair-wise opportunistic coding scheme. In general, the scheme does not achieve capacity with perfect immediate feedback. However, it is simple to implement and analyse. Furthermore, it is compatible with the time-division scheduling among the receivers in a single packet erasure broadcast channel case. The scheme can be readily extended to multiple non-interfering broadcast channels. First, we consider how this scheme works in the case of two receivers, t_1 and t_2.

When the transmitter sends a packet from f_1 to t_1, if the packet is erased at t_1 but overheard at t_2, we call the packet a Type A packet. Similarly, we define Type B packets to be the packets that are intended to t_2, but missed by t_1 and overheard by t_1. Let the set of type A packets be S_1 and the set of Type B packets be S_2. The algorithm works as follows

At any time slot, the transmitter s randomly picks a receiver to serve based on a stochastic vector (a_i, a_2), similarly to that in Chapter 3, where a_i is the probability that the transmitter broadcast a packet from f_i at any time slot. The decision is made independently for each time slot. Let $|S_i|$ be the cardinality of the set S_i. At some time slot, assume that receiver t_i is chosen,

- If $|S_1| > 0, |S_2| > 0$, then randomly pick $X_1 \in S_1$ and $X_2 \in S_2$, send $Y = X_1 \oplus X_2$. Furthermore, if Y is received at t_i, then remove X_i from S_i, $i = 1, 2$.

- Otherwise, $|S_1| = 0$ or $|S_2| = 0$. Simply send an new packet from f_i for the target receiver t_i. If the packet is erased at t_i and received at at the other receiver, update S_i to include this packet. Otherwise, S_i stays the same.

4.3.2 Markov Model For Two Receivers

Next, we use a Markov chain to analyze the gain of using inter-session coding method in the case of two receivers. Notice that inter-session coding opportunity only occur when

both S_1 and S_2 are nonempty sets. Consider the system states denoted by the tuple of the cardinalities of the two sets $(|S_2|, |S_1|)$. The Markov chain presenting the evolution of the states is drawn in Figure 4-2.

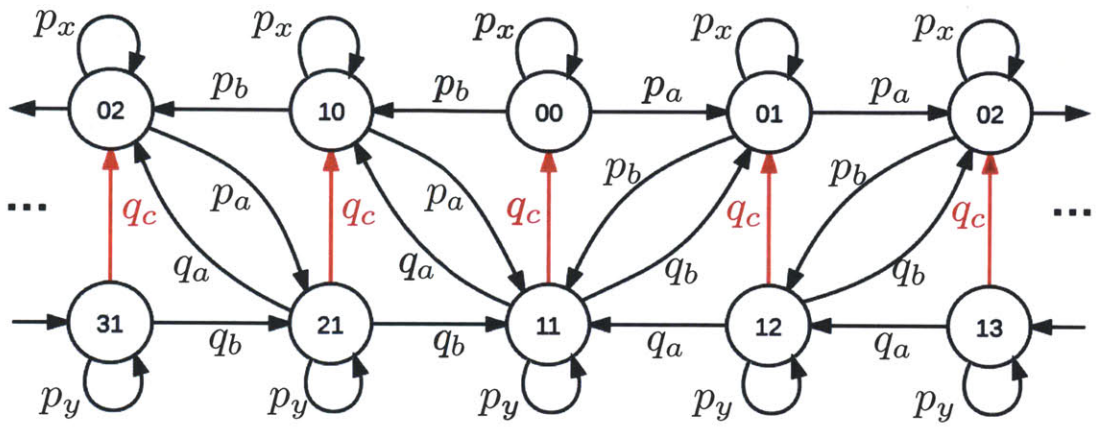


Figure 4-2: Markov chain for pair-wise inter-session coding based on erasure patterns

Note that there are two layers of states. When the system is at the upper layer states, there is no inter-session coding is done. The transmitter chooses a receiver to serve, based on the scheduling probability vector (a_1, a_2). On the other hand, at the lower layer states, coding can always be done and the coded packet can benefit both receivers. The state transition probability is defined as follows,

1. At any upper layer state, let Event A (or B) be the event that a packet is sent to t_1 (or t_2), but erased at t_1 (or t_2) and overheard at t_2 (t_1), i.e. a Type A (or B) packet is generated. We have,

$$P(\text{Event A}) = p_a = a_1\varepsilon_1(1-\varepsilon_2) \tag{4.6}$$

$$P(\text{Event B}) = p_b = a_2\varepsilon_2(1-\varepsilon_1). \tag{4.7}$$

Starting at state 00, if none of the receiver receives the transmitted packet or none of

Event A or B happens, the chain stays in the same state with probability

$$p_x = a_1(1-\varepsilon_1) + a_2(1-\varepsilon_2) + \varepsilon_1\varepsilon_2. \tag{4.8}$$

If Event A happens, S_1 is increased by 1. Therefore, the chain moves to state 01 with probability p_a. Similarly, Event B causes the transition from 00 and 10 with probability p_b.

In some upper layer state other than 00, the occurrence of Event A or B may transit the state into lower layer state, depending on the current state. For example, at 01 if Event B happens, S_2 increases by 1 and 11 state is reached. A similar approach applies to the states on the left of 00.

2. Now, consider the lower layer states. The fact that $|S_1|$ and $|S_2|$ are both greater than zero, means that the sender always transmits inter-session coded packets when it is in lower layer states. Consider the case when a inter-session coded packet is transmitted,

   - If both receivers receive the packet successfully, the packets X_1 and X_2 are removed from S_1 and S_2 respectively, and the chain moves to the non-coding state directly above. This happens with probability,

     $$q_c = (1-\varepsilon_1)(1-\varepsilon_2). \tag{4.9}$$

     Note that this is when the coded transmission generates rate gain, as the delivery of the coded packet benefit both receivers.

   - If only one of the receiver receives the packet, then the rate gain is not observed for this particular transmission. For example, if only t_1 receives $X_1 \oplus X_2$, it is equivalent to it receiving X_1 individually, and X_1 is removed from S_1. Therefore, in state transition, $|S_1|$ is reduced by 1. Furthermore, if $|S_1|$ is reduced to 0, the chain will jump to a upper layer state, depending on the current state. That

happens with probability,

$$q_a = (1-\varepsilon_1)\varepsilon_2, \qquad \text{or} \tag{4.10}$$

$$q_b = (1-\varepsilon_2)\varepsilon_1. \tag{4.11}$$

- Finally, it none of the receiver receives the coded packet, the state remains the same, with probability,

$$q_y = \varepsilon_1\varepsilon_2. \tag{4.12}$$

4.3.3 Steady-state Distribution

In order to obtain the steady distribution of the Markov chain shown in Figure 4-2, we first examine the same problem when the buffer size for overheard packet at each receiver is limited to B. This results in a finite Markov chain is similar to the original infinite chain, except that edge states. Specifically, the edge states are, $0B$, $B0$, $1B$ and $B1$. Assume that there is a stable distribution and π_{ij} is the steady state probability of state ij in the finite state Markov chain. We can solve the chain by applying balance equations for various cuts. We use the right side of the chain in the following explanation, but it is understood that the same applies to the left side of the chain by symmetry.

Edge Cuts

First consider the $C1$ and $C2$ cuts at the right edge of the chain, as shown in Figure 4-3. For the balance equation cross the cuts, we have

$$(q_c + q_a)\pi_{1B} = p_a\pi_{0B-1} \qquad (C1) \tag{4.13}$$

$$p_b\pi_{0B} = p_a\pi_{0B-1} + q_b\pi_{1B} \qquad (C2). \tag{4.14}$$

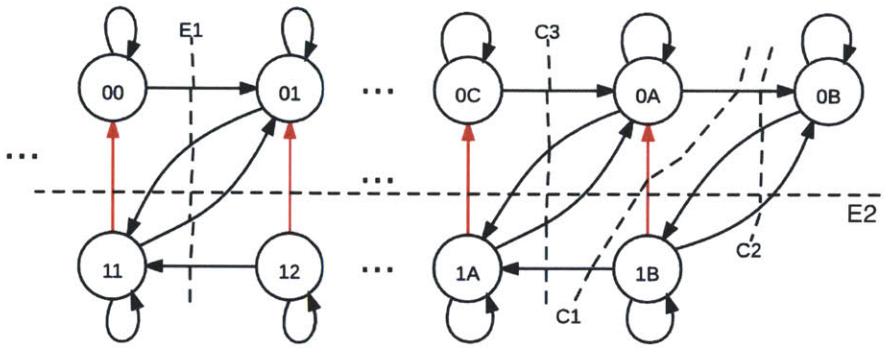


Figure 4-3: Cuts at the right part of the finite state Markov chain. $A = B - 1, C = B - 2$, where B is the buffer size.

Combining the two equations, and performing the same cuts on the left edges of the chain (or simply use symmetry), we have

$$\pi_{0B} = \frac{q_a + q_b + q_c}{p_b} \pi_{1B} = r_{re}\, \pi_{1B} \tag{4.15}$$

$$\pi_{B0} = \frac{q_a + q_b + q_c}{p_a} \pi_{B1} = r_{le}\, \pi_{B1}, \tag{4.16}$$

where we use

$$r_{re} = \frac{q_a + q_b + q_c}{p_b} \quad , \qquad r_{le} = \frac{q_a + q_b + q_c}{p_a}, \tag{4.17}$$

to represent the ratio between the steady state probabilities of two edge states on each side of the chain. Furthermore, we can cut the chain in the middle in the same way as $C1$, i.e. for the cut between state $0j$, $1j$ on one side and $0j+1$ and $1j+1$ on the other side, we have

$$(q_c + q_a)\pi_{1j+1} = p_a \pi_{0j} \tag{4.18}$$

$$(q_c + q_b)\pi_{j+11} = p_b \pi_{j1}. \tag{4.19}$$

alternatively,

$$\pi_{0j} = \frac{q_a + q_c}{p_a}\pi_{1j+1} = r_r\pi_{1j+1} \tag{4.20}$$

$$\pi_{j0} = \frac{q_b + q_c}{p_b}\pi_{j+11} = r_l\pi_{j+11}, \tag{4.21}$$

in which, the steady state probabilities of upper layer states are expressed with those of the lower layer states, and the ratios are denoted as $r_r = \frac{q_a+q_c}{p_a}$ and $r_l = \frac{q_b+q_c}{p_b}$ respectively.

Vertical Cuts

Next, consider any vertical cut in the middle portion of the chain, such as $C3$, shown in Figure 4-3. The states on the left side of the cut are $0j-1$ and $1j$, the balance equation can be written as,

$$p_a\pi_{0j-1} + q_b\pi_{1j} = p_b\pi_{0j} + q_a\pi_{1j+1}. \tag{4.22}$$

Combining this with (4.21), we can express π_{1j+1} in terms of π_{1j}, for $j \geq 2$, and similarly for π_{1j+1} and π_{j1}. These are given as,

$$\pi_{1j+1} = \frac{p_aq_a + p_aq_b + p_aq_c}{p_aq_a + p_bq_a + p_bq_c}\,\pi_{1j} = r_1\pi_{1j} \tag{4.23}$$

$$\pi_{j+11} = \frac{p_bq_a + p_bq_b + p_bq_c}{p_bq_b + p_aq_b + p_aq_c}\,\pi_{j1} = r_2\pi_{j1}. \tag{4.24}$$

where the ratios are denoted by constants r_1 and r_2. Now, putting together Equations (4.24) (4.21) and (4.16), we can express any steady state probability $\pi_{ij}, (i,j) \neq (1,2)$ on the right (or left) side of the chain in terms of π_{12} (or π_{21}). That is,

- On the left side of the chain,

$$\pi_{1j} = r_1^{j-2}\,\pi_{12}, \qquad 2 < j < B \tag{4.25}$$

$$\pi_{0j} = r_r\pi_{1j+1} = r_1^{j-1}r_r\,\pi_{12} \qquad 1 \leq j \leq B \tag{4.26}$$

$$\pi_{0B} = r_{re}\pi_{1B} = r_{re}r_1^{B-2}\pi_{12} \tag{4.27}$$

- On the right side of the chain,

$$\pi_{j1} = r_2^{j-2}\pi_{21}, \qquad 2 < j < B \tag{4.28}$$

$$\pi_{j0} = r_l\pi_{j+11} = r_1^{j-1}r_r\,\pi_{21} \qquad 1 \leq j \leq B \tag{4.29}$$

$$\pi_{B0} = r_{le}\pi_{B1} = r_{le}r_1^{B-2}\pi_{21} \tag{4.30}$$

Following these equation, we only need to solve $\pi_{00}, \pi_{11}, \pi_{12}$ and π_{21} for all the steady state probabilities, which requires four equations.

Linear Equations

The four equations can be found in the following ways.

1. Using cut $E1$, shown in Figure 4-3, we have,

$$p_a\pi_{00} + q_b\pi_{11} = p_br_r\pi_{12} + q_a\pi_{12},$$

   i.e.

$$p_a\pi_{00} + q_b\pi_{11} - (p_br_r + q_a)\pi_{12} = 0 \tag{4.31}$$

2. Similarly, we can cut on the left side of the chain, in the same way as $E1$, in which we obtain,

$$p_b\pi_{00} + q_a\pi_{11} - (p_ar_l + q_b)\pi_{21} = 0 \tag{4.32}$$

3. Consider the horizontal cut $E2$ which separates the upper and lower layers. All the transition probability from upper to lower layer is given by,

$$P(\text{upper to lower}) = p_b\sum_{j=1}^{B}\pi_{0j} + p_a\sum_{j=1}^{B}\pi_{j0} \tag{4.33}$$

$$= p_b(\sum_{j=1}^{B-1} r_r r_1^{j-1} + r_{re}r_1^{B-2})\pi_{12} + p_a(\sum_{j=1}^{B-1} r_l r_2^{j-1} + r_{rl}r_2^{B-2})\pi_{21} \tag{4.34}$$

$$= p_b(r_r \frac{1-r_1^{B-1}}{1-r_1} + r_{re} r_1^{B-2})\,\pi_{12} + p_a(r_l \frac{1-r_2^{B-1}}{1-r_2} + r_{re} r_2^{B-2})\,\pi_{21} \tag{4.35}$$

$$= p_b c_{12}\pi_{12} + p_a c_{21}\pi_{21}, \tag{4.36}$$

where we use c_{12} and c_{21} to denote constants $r_r \frac{1-r_1^{B-1}}{1-r_1} + r_{re} r_1^{B-2}$ and $r_l \frac{1-r_2^{B-1}}{1-r_2} + r_{re} r_2^{B-2}$ respectively for convenience. On the other hand, for the transition from lower to upper layer, we have,

$$P(\text{lower to upper}) = (q_c + q_b) \sum_{j=2}^{B} \pi_{1j} + (q_c + q_a) \sum_{j=2}^{B} \pi_{j1} + (q_a + q_b + q_c)\pi_{11} \tag{4.37}$$

$$= (q_c + q_b) \frac{1-r_1^{B-1}}{1-r_1} \pi_{12} + (q_c + q_a) \frac{1-r_2^{B-1}}{1-r_2} \pi_{21} + (q_a + q_b + q_c)\pi_{11} \tag{4.38}$$

$$= (q_c + q_b) d_{12}\pi_{12} + (q_c + q_a) d_{21}\pi_{21} + (q_a + q_b + q_c)\pi_{11}, \tag{4.39}$$

where $d_{12} = \frac{1-r_1^{B-1}}{1-r_1}$ and $d_{21} = \frac{1-r_2^{B-1}}{1-r_2}$ are constants. Equating (4.36) and (4.39), we have the third linear equation in terms of $\pi_{00}, \pi_{11}, \pi_{12}$ and π_{21}.

4. Finally, all the steady state probabilities must sum up to 1, which gives us the fourth equation,

$$\pi_{00} + \pi_{11} + (c_{12} + d_{12})\pi_{12} + (c_{21} + d_{21})\pi_{21} = 1 \tag{4.40}$$

where, c_{12}, c_{21} and d_{12}, d_{21} are defined as constants in (4.36) and (4.39) respectively.

Solving this linear system, we can recover the four variables previously states. But more importantly, we can compute the probability that the system is in a coding state, i.e. in a lower layer state. This probability π_c is given by the sum of all steady state probabilities of lower layer states, parametrized by $a_1, a_2, \varepsilon_1, \varepsilon_2$ and the buffer size B, i.e.

$$\pi_c^B = d_{12}\pi_{12} + d_{21}\pi_{21} + \pi_{11}. \tag{4.41}$$

Furthermore, sum rate gain can be obtained by computing the frequency at which the red edges are traversed in Figure 4-2. That is simply,

$$G_{\text{sum}} = q_c \pi_c^B, \tag{4.42}$$

while the rate gain for each individual receiver is given as,

$$G_{t_1} = a_1 G_{\text{sum}} , \qquad G_{t_2} = a_2 G_{\text{sum}} \tag{4.43}$$

Note that, the solution gives the steady state probably for coding states and coding gain when the buffer size at each receiver is limited to B. When infinite buffer is allowed, simply take the limit of π_c^B as $B \to \infty$.

The detailed solution procedure, which uses standard algebraic techniques, is omitted. Let

$$\gamma_i = \frac{a_i \varepsilon_i (1-\varepsilon_j)(1-\varepsilon_i \varepsilon_j)}{(1-\varepsilon_i)\varepsilon_j (a_j(1-\varepsilon_i) + a_i \varepsilon_i (1-\varepsilon_j))},$$

where $j \in 1, 2, j \neq i$. After some tedious algebra (Appendix A), we can obtain that, as $B \to \infty$, if $\gamma_1 < \gamma_2$, then

$$\pi_c^B \to \pi_c^* = \frac{a_1 \varepsilon_1 (1-\varepsilon_2)}{(1-\varepsilon_1) + a_1 \varepsilon_1 (1-\varepsilon_2)}. \tag{4.44}$$

Intuitively, in this case, the probability of t_2 overhearing t_1 is lower, in the long run, the probability of of getting into a coding state is dominated by the probability of t_2 overhearing missed packet by t_1, while t_1 always overhear more than enough t_2 packets in the long run.

4.3.4 M Receivers

When there are more than two receivers, the coding becomes much more complicated. Coding opportunities may happen between any pair of receivers. In this thesis, we keep the inter-session coding limited to packets belong to two sessions. While the optimal inter-session coding for such channel is unknown, it is easy to see that the method is not optimal.

However, on the other hand, the code design is much tractable with minimal amount of erasure bookkeeping needed.

When the number of receivers is even, one method is to pair up the receivers statically and perform the pair-wise coding only within the pair. In this case, the previous results still apply. We just have to perform the same analysis for each pair of receivers individually. Hence, if t_i is paired with t_j, we just have to replace $a_1, a_2, \varepsilon_1, \varepsilon_2$ with $a_i, a_j, \varepsilon_i, \varepsilon_j$ in the above Markov chain to obtain the rate gain from (4.43).

However, such rigidly pairing up is obviously not a good option and restricts the coding to pre-defined pairs or pairwise codings. An alternative is to keep track the global erasure pattern and determine online, packets from which two sessions can be coded together. In this method, at any time slot, the transmitter picks a receiver t_i. The algorithm check all possible receivers t_j, $j \neq i$, to see if there exist a receiver j, such that t_i and t_j both overheard each other's missing packet in previous transmission. If there exist such a t_j, the code the two packets together, otherwise we send the original packet for t_i. However, the analysis is more difficult, since the steady state probability of coding states for any two pairs of receivers are not independent. It is, on the other hand, more straightforward to simulate the coding algorithms to understand the coding gains numerically, which is our current work in progress. Most importantly, with better understanding of pair-wise inter-session coding schemes, we look to investigate more general practical inter-session codes in our future works to exploit the benefits of increased capacity from feedbacks.

Chapter 5

Architecture and Protocols

In this chapter, we briefly discuss the practical aspects of implementing the scheme proposed in Chapter 3 using existing protocol architectures with possible modifications. Specifically, we consider the modification of network layers in home networking devices, especially the gateway, access points and possible user devices to efficiently leverage the benefits of network coding and the proposed optimization schemes from previous chapters.

5.1 Session Definition

We start with the definition of a *session* (or a *flow*) on the network layers from a network coding perspective. The coding method we propose is essentially an End-to-End coding between the gateway and the user devices. In the layered architecture within a home network, there are multiple possible ways of defining a transmission session or flow. For example, a session defined in the application or transport layer means that some user devices, such as a PC, may be identified as multiple receivers, when running multiple application or opening multiple connections at the application level. On the other hand, a session may be defined at a much lower level, for example, at the MAC layer. Such a session will contain all packets destined to the same MAC address regardless higher layer entities.

For the purpose of our scheme, we define a session to be all the data packets destined

to the same receiver. That is, a session is defined on a receiver basis. We believe that the definition is most suitable for our model for several reasons.

- Firstly, instead of having any coding operations at the access points, encoding and decoding of packets happen at the home gateway and at the receivers, respectively. That is because, compared to what is possible at the gateway, the computation power at access points is usually much more limited in today's home network. It may be desired to keep dumb devices like access points unaware of network coding related operation as far as possible. The gateway is a network layer entity, while the access points sit at the link layer. Hence it is more reasonable to group packets to the same receiver into a session at the network layer.

- Secondly, such session definition may help to avoid starvation of encoding buffers. A session defined on a receiver basis may include one or more upper layers (TCP or UDP) connections. Therefore, packets from multiple upper layer connections are buffered in the same queue for encoding, which allows more packets to be coded together.

- The optimization complexity is reduced because of reduced number of sessions at the per receiver level, compared to session defined at connection level.

As an alternative, it may be beneficial to define a user session at the upper level, so that the algorithm may be able to optimize the coding and scheduling parameters more specifically with respect to the QoE requirements of individual applications. That means defining a session to correspond to one or more transport layer connections associated with a specific application running on a particular receiver.

However, that may come with significant difficulties and challenges, such as the layering issue and possible starvation of encoding buffers. The number of such connections may be too large to be managed efficiently for coding operations, especially when there are multiple access points whose scheduling operations are to be optimized. Furthermore, setting up the network coding sublayer and coding within a TCP session requires the gateway to intercept the TCP level connection, which may introduce extra complexity to the existing protocol

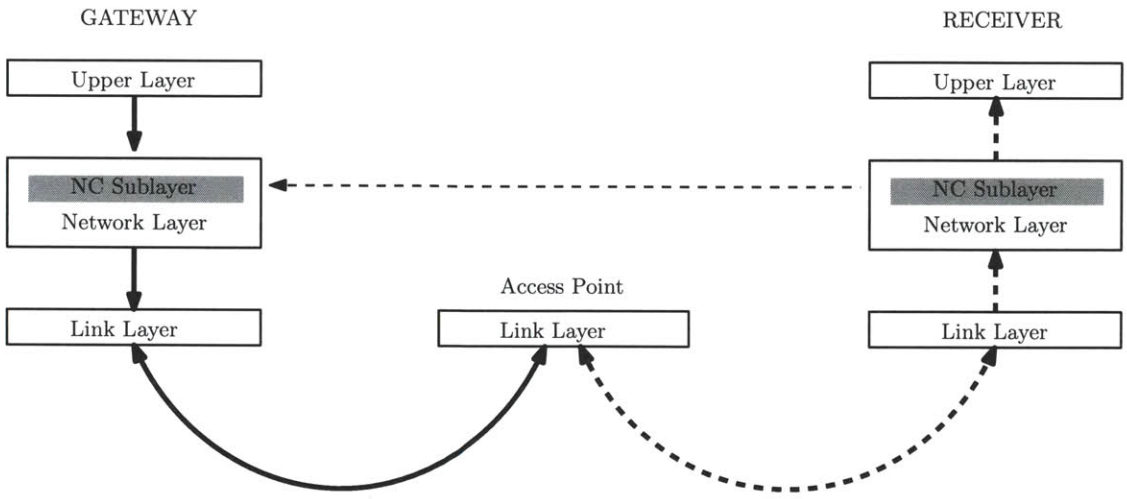


Figure 5-1: Network Coding sublayer design: the sublayer is inserted directly above network layer and a session at this sublayer includes all IP packets from the gateway, destined to the same receiver.

stack and bring the common problems encountered by many TCP proxy solutions.

For the rest of the discussion, we follow the definition of a session on a receiver basis. We consider placing a *network coding sublayer* within the network layer at the gateway as well as at the end receivers. Figure 5-1 illustrates the layer design of our system. Notice that the protocol requires modifications at the gateway as well as at the receivers. Moreover, as we will mention later, some modifications at the access points may be also necessary and come with more challenges.

Briefly, the basic functionalities and operations of the network coding sublayer can be summarized as follows.

(a) **Buffering Packets**. The network coding sublayer queues packet according to their sessions. Packets from multiple TCP or UDP sessions are placed in the same queue as long as they are destined to the same user. Thus, in the case of multiple access points, where each receiver may have multiple IP addresses associated with the access points, the network coding sublayer at the gateway needs to keep track of the IP address belonging to the same receiver in order to queue packets accordingly. Moreover, the sublayer needs to collect information from multiple interfaces in the corresponding sublayer on the user

device side.

(b) **Encoding and Decoding**. The adaptive intra-session network coding discussed in Chapter 3 is implemented in the network coding sublayer. Note that the source in the original model is the NC sublayer at the gateway in the layering structure. The details of the encoded transmission are covered in the next section.

(c) **Block Acknowledgement**. When the end receiver has collected sufficient number of packets from multiple access points, the NC sublayer at the receiver is responsible for sending feedback to the gateway NC sublayer, indicating that the packets in the current coding bucket are correctly received, so that the gateway is able to empty its buffer and start to arrange the next round of transmission for new packets in the coding bucket.

(d) **Optimization**. Finally, the coding bucket size and scheduling coefficient optimizations need to be carried out at the gateway NC sublayer and some results of the optimization, such as scheduling coefficients need to be communicated to the access points.

5.2 Packet Transmissions and Acknowledgements

5.2.1 Encoded Transmissions

We consider a typical *round of transmission* for some receiver t_j in the layered system. All the packets in current coding bucket are buffered at the sublayer at the gateway. Suppose that there are two access points which are connected to the gateway by lossless high bandwidth wireline links. It is straightforward to generalize to systems with only one or more than two access points.

Let time $t = 0$ be the start of the round of transmission for receiver t_j. At $t = 0$, the NC sublayer at the gateway produces $T_1 + T_2$ coded packets, which are linear combinations of the K_j packets in the coding bucket, with coefficients randomly chosen from some sufficiently large field $\mathbb{F}_q$. The network layer then forwards the first T_1 coded packets to access point s_1 and the last T_2 packets to access point s_2. We assume that the transmission between the

gateway and the access points are costless because of the high capacity links. Therefore, for simplicity we set $T_i = (1+\sigma)\frac{K_j}{1-\varepsilon_{ij}}$, for some small positive σ. That is to ensure that, with high probability, the receiver can obtain all packets from a single access point if necessary. We later discuss the case when the receiver fails to receive enough coded packets.

At the next stage, each access point, whenever it is ready to serve receiver t_i, picks a coded packet to transmit to receiver t_i. At the receiver side, the NC sublayer will collect all packets received for the current coding block and place them in a buffer. When the receiver collects K_i packets, it tries to decode the block. With high probability, the decoding will be successful with K_i coded packets. If it fails, then there are linearly dependent packets, and it waits for one more packet before attempting to decode again. Eventually, all packets in the coding bucket are recovered and the decoded packets are delivered to their respective connections at the upper layer.

The link layer retransmission does not guarantee reliable communications. The WiFi link layer makes a maximum number of attempts (usually 3 to 5) for every packet, before the packet is dropped. Hence, the probability of a packet erasure is the probability that the packet fail to be delivered after several attempts. On the other hand, with the coding scheme, there is no incentive to attempt the same packet multiple times. Hence, the maximum number of attempts should be set as low as possible to avoid unwanted retransmissions, whose need is eliminated by the use of coding. As soon as the decoding succeeds, the receiver transmits a feedback message to the gateway at the NC sublayer to acknowledge the current coding bucket. The receiver also sends extra acknowledgements to the gateway if more packets of the same coding bucket are received after successful decoding, generating repeated ACK for the same round of transmission. Note that the ACK happens at the NC sublayer.

As the gateway sees the acknowledgement from receiver t_j about the current round of transmission, it empties the coding bucket and starts a new round of transmissions by moving in new packets from flow f_j. This process repeats until receiver i finishes its session.

However, there are still some challenges to be addressed in the layering framework.

1. Firstly, the scheme may not work well with TCP at the upper layer. Specifically, the TCP congestion control mechanism is largely built on the Round-Trip Time (RTT) and acknowledgements of individual TCP packets. Coding bucket or generations at lower layer means End-to-End TCP sessions observe delayed feedback and larger round trip time, which may largely reduce the efficiency of TCP control and retransmission mechanism.

   Two possible solution may be used. The first one requires larger TCP packet sizes, so that the lower layer can segment TCP packet into smaller frames, each as a unit for coding at the NC sublayer level. At the same time, the timeout value for TCP session can be adjusted carefully to match the corresponding choices of packet and frame sizes. These will mask the lower layer coding from the higher level TCP connections. However, the effectiveness is not tested.

   Alternatively, one could created a per packet ACK messages at NC sublayer to acknowledge the delivery of each packet, or essentially the delivery of each degree of freedom in the linear system. This is first introduced with the *seen packet* concept by Sundararajan *et al.* [39]. With each "seen packet" ACK forwarded to its corresponding TCP connection, the problem can be solved efficiently. For the detail of the NC/TCP ACK, we refer the reader to [39].

2. At the start of each round of transmission, the gateway will empty the coding bucket. However, with high probability, there are still some remaining coded packets at the access points, waiting to be transmitted to the receivers. Given that the receiver already decode the round of transmission, it is very inefficient to allow these packets to be transmitted by access points. A mechanism need to be presented to allow access point to purge these packets. In particular, such feature is usually not observed in today's access point devices. Therefore, the scheme may require slight modification in access points.

   A possible mechanism can function as follows. Each access points are required to check a round of transmission ID associated with the packets that the gateway pushes to the

access points. At soon as the access points see a packet with a new ID that is pushed to it, it will purge all the packets associated with any older round of transmission IDs to prevent transmission redundancy.

5.2.2 Acknowledgement and Retransmission

When the channel status is very poor, it may happen that after an access point s_i finish its attempts on the T_i packets of current coding bucket, receiver t_j has not collected enough coded packets for decoding. Therefore, we need a mechanism to allow more coded packets to be supplied to access points for transmission. We propose two solutions to the problem:

A. **Timeout At The Gateway**. An appropriate timeout value can be chosen at the gateway for the delivery of round of transmission. If no acknowledgement from the NC sublayer at the receiver side is received by the gateway before timeout, the gateway starts to send more coded packets to each access point. With this solution, the timeout value and the number of coded packets sent have to be carefully tuned to prevent buffer overflow at the access points.

B. **Requests By APs**. Alternatively, each access point can send messages back to the gateway as soon as it finishes the current K_i coded packets. The gateway responds by feeding the particular access point with more coded packets of the current coding bucket. However, that requires further modifications at the access points and adding cross layer control message, which may not be desired.

Chapter 6

Conclusions and Future Work

In this thesis, we study the coding and scheduling optimization in typical home networks, between the gateway and user devices. The home network is modelled as a single source, multiple input broadcast channels with packet erasures. Each access point acts as an input to a non-interfering broadcast channel.

In Chapter 2, we establish a unified framework for characterize the delay and rate requirements of various user applications or devices. The delay metrics or cost functions compute the normalized ℓ_p-norms of the in-order inter packet arrivals times of the original data packets, independent from coding algorithms. The exponent p functions as a single parameter characterization of delay sensitivity. When p is close to 1, the delay metrics measure some delay cost function whose value is close to the average delay incurred per packet, $d(1)$, which is also viewed as the inverse of the average rate. On the other hand, as p grows large, the normalized ℓ_p-norm grows increasingly biased toward larger inter-packet delays, which is the critical parameter that affects the QoE of delay sensitive applications. Such a characterization allows user devices to choose a targeted delay value corresponding to a spectrum of delay sensitivity. Compared to the common characterization of delay sensitivity into two extremes (extreme sensitive or not sensitive at all), such delay metrics give much greater flexibility for user device to optimize the right objective and for network system to explore the various trade-off between delay and throughput for different applications.

Under this delay-rate framework, we study the performance of the common fixed generation based coding scheme and propose an adaptive coding scheme that allows the change of coding bucket size based on delay constraints. The delay cost functions for these coding schemes are derived and expressed in terms of sensitivity p, system parameters like erasure probability and coding parameters such as coding bucket sizes. These expressions give us some clear insight to design system and tune codes to match the QoE requirements of heterogeneous user application and devices. In Chapter 3, we derive and illustrate the trade-off of $d(1)$ and $d(\infty)$ when $d(p)$ is optimized for different values of p in a point-to-point communication system. For multiple receivers systems, it is not possible to optimize the ℓ_p-norm delay cost function for all receivers at the same time. In this case, an optimization program is set up to obtain the optimal coding bucket sizes and time-division scheduling coefficients for each receivers for some system wise utility objective, provided that the delay requirements of various user applications are met. We show that the optimization program can be converted into generalized geometric program and solved efficiently.

The adaptive coding scheme demonstrate clear advantages over the commonly used fixed generation schemes. Our experiments show that as the delay sensitivity of certain receivers increases, the adaptive coding scheme is able to reduce the coding bucket size of the more delay sensitive receivers and shorten its inter-packet delivery times. On top of that, the optimization of the coding and scheduling parameters allows large coding bucket sizes to be assigned to relatively delay insensitive users to increases they average rate. This helps the system to save some time and more aggressively serve the delay sensitive users to meet their delay requirements for desired QoE.

The same coding and optimization procedures can be carried out in the case of multiple receivers. However, the optimization difficulty increases sharply, as the problem is a signomial optimization problem, which is no longer convex. Nevertheless, we can approximate the signomial constraints using simple posynomial condensation methods, and use a series of GP optimizations to find a local optimal solution. We show the convergence of the algorithm and illustrate it with examples.

In Chapter 4, we change some assumptions on the system model and investigate the case when the feedback delay is negligible. In this case, inter-session coding needs to be performed for coding gains. The problem of inter-session coding capacity on broadcast erasure channels is largely unknown. We examine recent literature on outer bound and coding methods for the single transmitter cases. We propose a pair-wise opportunistic inter-coding methods for such systems. A Markov chain model is established for the 2 receivers case which provides the full solution to the coding gain in this case. Furthermore, we extend the algorithm for the case with an arbitrary number of receivers.

The final part of the thesis is devoted to discuss practical issues associated with the adaptive coding scheme between gateway and receivers. We propose to add a network coding sublayer directly on top of the network layer to perform the intended functions. Various challenges and difficulties are discussed and some solutions are proposed.

There are several directions for possible future work. Firstly, the link between ℓ_p-norm delay metrics or cost functions and the performance of a variety of different networking applications can be explored and tested. The optimization or the fulfillment of ℓ_p-norm itself solely may not guarantee desirable QoE for many application categories. In this case, it is valuable to understand further the needs of specific applications in terms of various possible metrics and network resource allocations. In the case of inter-session coding based on erasure patterns, neither the capacity nor the optimal coding scheme for arbitrary number of receivers are known. Even for suboptimal coding schemes, for obtaining sufficient gains, exhaustive book-keeping and computations are needed, such as that in [24]. Any further exploration along this direction, especially towards practically implementable coding scheme will be very helpful for todays networks, since broadcast channels with packet erasures are widely present in many typical networking systems.

In summary, instead of the widely assumed two extreme delay sensitivities model for network application today, it is worthwhile to devise better delay sensitivity models given the enormous heterogeneity of modern networking applications. Network resource allocation may be conducted in a much more efficient manner with appropriate models of applications delay

requirements. Subsequently, many networking needs may be accommodated better with slight modification of the protocol structure. This approach is even more promising, with the proliferation of low cost networking device with ever-increasing computation capabilities.

Appendix A

Steady State Distribution

As discussed in Chapter 4, for a fixed buffer size B, we can consider only variables $\pi_{12}, \pi_{21}, \pi_{00}$ and π_{11}, and formulate four equations based on different cuts and sum of probabilities. These equations are given by,

$$p_a\pi_{00} + q_b\pi_{11} - (p_b r_r + q_a)\pi_{12} = 0 \tag{A.1}$$

$$p_b\pi_{00} + q_a\pi_{11} - (p_a r_l + q_b)\pi_{21} = 0 \tag{A.2}$$

$$(q_a + q_b + q_c)\pi_{11} + ((q_c + q_b)d_{12} - p_b c_{12})\pi_{12} + ((q_c + q_a)d_{21} - p_a c_{21})\pi_{21} = 0 \tag{A.3}$$

$$\pi_{00} + \pi_{11} + (c_{12} + d_{12})\pi_{12} + (c_{21} + d_{21})\pi_{21} = 1, \tag{A.4}$$

where the constants are given by,

$$p_a = a_1\varepsilon_1(1-\varepsilon_2), \qquad p_b = a_2\varepsilon_2(1-\varepsilon_1), \tag{A.5}$$

$$q_a = (1-\varepsilon_1)\varepsilon_2, \qquad q_b = (1-\varepsilon_2)\varepsilon_1, \qquad q_c = (1-\varepsilon_1)(1-\varepsilon_2), \tag{A.6}$$

$$r_{re} = \frac{q_a + q_b + q_c}{p_b}, \qquad r_{le} = \frac{q_a + q_b + q_c}{p_a}, \tag{A.7}$$

$$r_r = \frac{q_a + q_c}{p_a}, \qquad r_l = \frac{q_b + q_c}{p_b}, \tag{A.8}$$

$$r_1 = \frac{p_a q_a + p_a q_b + p_a q_c}{p_a q_a + p_b q_a + p_b q_c}, \qquad r_2 = \frac{p_b q_a + p_b q_b + p_b q_c}{p_b q_b + p_a q_b + p_a q_c}, \tag{A.9}$$

$$c_{12} = r_r\frac{1 - r_1^{B-1}}{1 - r_1} + r_{re}r_1^{B-2}, \quad c_{21} = r_l\frac{1 - r_2^{B-1}}{1 - r_2} + r_{re}r_2^{B-2}, \tag{A.10}$$

$$d_{12} = \frac{1 - r_1^{B-1}}{1 - r_1}, \qquad d_{21} = \frac{1 - r_2^{B-1}}{1 - r_2}. \tag{A.11}$$

Solving the equations, we obtain,

$$\pi_c^B = d_{12}\pi_{12} + d_{21}\pi_{21} + \pi_{11} \tag{A.12}$$

$$= \frac{a_1 a_2 \varepsilon_1 (1-\varepsilon_1)\varepsilon_2 (1-\varepsilon 2)(\gamma_1^B - \gamma_2^B)}{a_1(1-\varepsilon_1)^2\varepsilon_2 + a_1\varepsilon_1(1-\varepsilon 2)\left(\gamma_1^B - \varepsilon_2\left(\gamma_1^B - a_2(1-\varepsilon_1)(\gamma_1^B - \gamma_2^B)\right)\right)}, \tag{A.13}$$

where γ_i is defined as,

$$\gamma_i = \frac{a_i \varepsilon_i (1-\varepsilon_j)(1-\varepsilon_i\varepsilon_j)}{(1-\varepsilon_i)\varepsilon_j(a_j(1-\varepsilon_i) + a_i\varepsilon_i(1-\varepsilon_j))}, \tag{A.14}$$

for $j \in 1, 2, j \neq i$. Now, take the limit of π_c^B as $B \to \infty$, assume $\gamma_1 > \gamma_2$, we have,

$$\pi_c^* = \lim_{B\to\infty} \pi_c^B = \frac{a_1\varepsilon_1(1-\varepsilon_2)}{(1-\varepsilon_1) + a_1\varepsilon_1(1-\varepsilon_2)}. \tag{A.15}$$

The case of $\gamma_1 > \gamma_2$ can be solved by symmetry. These gives the probability of generating inter-session coding packet at any time slot.

Bibliography

[1] T. Ho, M. Médard, R. Koetter, D. R. Karger, M. Effros, J. Shi, and B. Leong, "A random linear network coding approach to multicast," *IEEE Transactions on Information Theory*, vol. 52, no. 10, pp. 4413–4430, Oct. 2006.

[2] P. Chou, Y. Wu, and K. Jain, "Practical network coding," in *Proceedings of the Annual Allerton Conference on Communication Control and Computing*, vol. 41, 2003, p. 4049.

[3] W. Zeng, C. Ng, and M. Médard, "Joint coding and scheduling optimization in wireless systems with varying delay sensitivities," in *Submitted*, 2011.

[4] J. MacLaren Walsh, S. Weber, and C. wa Maina, "Optimal rate delay tradeoffs and delay mitigating codes for multipath routed and network coded networks," *IEEE Transactions on Information Theory*, vol. 55, no. 12, pp. 5491–5510, Dec. 2009.

[5] Y. Li, M. Chiang, A. R. Calderbank, and S. N. Diggavi, "Optimal Rate-Reliability-Delay tradeoff in networks with composite links," *IEEE Transactions on Communications*, vol. 57, no. 5, pp. 1390–1401, May 2009.

[6] R. Ahlswede, N. Cai, S. Li, and R. Yeung, "Network information flow," *IEEE Transactions on Information Theory*, vol. 46, no. 4, pp. 1204–1216, 2000.

[7] S. Y. Li, R. W. Yeung, and N. Cai, "Linear network coding," *IEEE Transactions on Information Theory*, vol. 49, no. 2, pp. 371–381, Feb. 2003.

[8] R. Koetter and M. Médard, "An algebraic approach to network coding," *IEEE/ACM Transactions on Networking*, vol. 11, no. 5, pp. 782–795, 2003.

[9] M. Luby, "LT codes," in *The 43rd Annual IEEE Symposium on Foundations of Computer Science, 2002. Proceedings.* IEEE, 2002, pp. 271– 280.

[10] A. Shokrollahi, "Raptor codes," *IEEE Transactions on Information Theory*, vol. 52, no. 6, pp. 2551–2567, Jun. 2006.

[11] D. J. MacKay, "Fountain codes," *Communications, IEE Proceedings-*, vol. 152, no. 6, pp. 1062– 1068, Dec. 2005.

[12] A. Eryilmaz, A. Ozdaglar, M. Médard, and E. Ahmed, "On the delay and throughput gains of coding in unreliable networks," *IEEE Transactions on Information Theory*, vol. 54, no. 12, pp. 5511–5524, 2008.

[13] D. Nguyen, T. Tran, T. Nguyen, and B. Bose, "Wireless broadcast using network coding," *IEEE Transactions on Vehicular Technology*, vol. 58, no. 2, pp. 914–925, Feb. 2009.

[14] J. K. Sundararajan, P. Sadeghi, and M. Médard, "A feedback-based adaptive broadcast coding scheme for reducing in-order delivery delay," in *Workshop on Network Coding, Theory, and Applications, 2009. NetCod '09.* IEEE, Jun. 2009, pp. 1–6.

[15] L. Keller, E. Drinea, and C. Fragouli, "Online broadcasting with network coding," in *Fourth Workshop on Network Coding, Theory and Applications, 2008. NetCod 2008.* IEEE, Jan. 2008, pp. 1–6.

[16] P. Sadeghi, D. Traskov, and R. Koetter, "Adaptive network coding for broadcast channels," in *Workshop on Network Coding, Theory, and Applications, 2009. NetCod '09.* IEEE, Jun. 2009, pp. 80–85.

[17] D. S. Lun, P. Pakzad, C. Fragouli, M. Médard, and R. Koetter, "An analysis of Finite-Memory random linear coding on packet streams," in *2006 4th International Symposium on Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks.* IEEE, Apr. 2006, pp. 1– 6.

[18] T. Cover, "Broadcast channels," *IEEE Transactions on Information Theory*, vol. 18, no. 1, pp. 2–14, 1972.

[19] ——, "Comments on broadcast channels," *IEEE Transactions on Information Theory*, vol. 44, no. 6, pp. 2524–2530, 1998.

[20] H. Weingarten, Y. Steinberg, and S. Shamai, "The capacity region of the gaussian Multiple-Input Multiple-Output broadcast channel," *IEEE Transactions on Information Theory*, vol. 52, no. 9, pp. 3936–3964, 2006.

[21] A. Dana and B. Hassibi, "The capacity region of multiple input erasure broadcast channels," in *Information Theory, 2005. ISIT 2005. Proceedings. International Symposium on*, 2005, pp. 2315–2319.

[22] Y. E. Sagduyu and A. Ephremides, "On broadcast stability region in random access through network coding," in *44th Allerton Annual Conference on Communication, Control and Computing*, 2006.

[23] P. Chaporkar and A. Proutiere, "Adaptive network coding and scheduling for maximizing throughput in wireless networks," in *Proceedings of the 13th annual ACM international conference on Mobile computing and networking*, 2007, p. 135146.

[24] M. Gatzianas, L. Georgiadis, and L. Tassiulas, "Multiuser broadcast erasure channel with feedback - capacity and algorithms," *1009.1254*, Sep. 2010. [Online]. Available: http://arxiv.org/abs/1009.1254

[25] C. Wang, "Capacity of 1-to-K broadcast packet erasure channels with channel output feedback (Full version)," *1010.2436*, Oct. 2010. [Online]. Available: http://arxiv.org/abs/1010.2436

[26] S. Jaggi, P. Sanders, P. A. Chou, M. Effros, S. Egner, K. Jain, and L. M. Tolhuizen, "Polynomial time algorithms for multicast network code construction," *IEEE Transactions on Information Theory*, vol. 51, no. 6, pp. 1973– 1982, Jun. 2005.

[27] T. Ho and D. Lun, *Network coding: an introduction.* Cambridge Univ Pr, 2008.

[28] S. Boyd, S. Kim, L. Vandenberghe, and A. Hassibi, "A tutorial on geometric programming," *Optimization and Engineering*, vol. 8, pp. 67–127, Apr. 2007. [Online]. Available: http://www.springerlink.com/content/9941458381562w82/

[29] M. Chiang, *Geometric programming for communication systems.* Now Publishers Inc, 2005.

[30] R. Duffin, E. Peterson, and C. Zener, *Geometric programming: theory and application.* Wiley, 1967.

[31] M. Grant and S. Boyd, "CVX: matlab software for disciplined convex programming," *Available httpstanford edu boydcvx*, 2008.

[32] B. Marks and G. Wright, "A general inner approximation algorithm for nonconvex mathematical programs," *Operations Research*, p. 681683, 1978.

[33] T. Cover, J. Thomas, J. Wiley *et al.*, *Elements of information theory.* Wiley Online Library, 1991, vol. 6.

[34] A. Dana, R. Gowaikar, R. Palanki, B. Hassibi, and M. Effros, "Capacity of wireless erasure networks," *IEEE Transactions on Information Theory*, vol. 52, no. 3, pp. 789–804, 2006.

[35] A. E. Gamal, "The feedback capacity of degraded broadcast channels (Corresp.)," *IEEE Transactions on Information Theory*, vol. 24, no. 3, pp. 379– 381, May 1978.

[36] F. Xue and X. Yang, "Network coding and Packet-Erasure broadcast channel," in *Sensor, Mesh and Ad Hoc Communications and Networks Workshops, 2008. SECON Workshops '08. 5th IEEE Annual Communications Society Conference on*, 2008, pp. 1–4.

[37] P. Larsson and N. Johansson, "Multi-user ARQ," in *Vehicular Technology Conference, 2006. VTC 2006-Spring. IEEE 63rd*, vol. 4, 2006, p. 20522057.

[38] E. Rozner, A. P. Iyer, Y. Mehta, L. Qiu, and M. Jafry, "ER: efficient retransmission scheme for wireless LANs," in *Proceedings of the 2007 ACM CoNEXT conference on - CoNEXT '07*, New York, New York, 2007, p. 1. [Online]. Available: http://portal.acm.org/citation.cfm?id=1364654.1364665

[39] J. K. Sundararajan, D. Shah, M. Medard, S. Jakubczak, M. Mitzenmacher, and J. Barros, "Network coding meets TCP: theory and implementation," *Proceedings of the IEEE*, vol. 99, no. 3, pp. 490–512, Mar. 2011.