CONTENT NETWORKING WITH PACKET-LEVEL CODING

by

YAO LI

A Dissertation submitted to the

Graduate School—New Brunswick

Rutgers, The State University of New Jersey

in partial fulfillment of the requirements

for the degree of

Doctor of Philosophy

Graduate Program in Electrical and Computer Engineering

Written under the direction of

Predrag Spasojević

and approved by

_____

_____

_____

_____

New Brunswick, New Jersey

October, 2012

ABSTRACT OF THE DISSERTATION

Content Networking with Packet-Level Coding

By Yao Li

Dissertation Director:
Predrag Spasojević

This dissertation focuses on linear packet combining (coding) strategies for bulk content data distribution in packet-switching networks and their effects on transmission cost. The employment of packet level codes aims to reduce the communication overhead needed to exchange real-time network states to restore packet losses, or to plan for efficient sharing of communication links. The aim is achievable due to the additional diversity introduced by coding, which increases the possibility of extracting innovative information by receivers. Nevertheless, increased computational complexity is entailed and may lower the throughput of network node processors and hence the communication throughput. This dissertation, in particular, studies coding strategies generating linear combinations of packets that are decodable with reduced complexity.

Major efforts to reduce computational complexity include limiting the number of packets combined in each coded packet and working with simple operations such as xoring. Two representative classes of codes are LT codes and coding with generations (hyperblocks). LT codes are the first class of fountain codes, especially suitable for multicast applications. Coding with generations is also of interest due to other practical reasons, such as source clustering.

We study these codes in two settings: (1) Content delivery to heterogeneous users experiencing varied channel conditions and having diverse demand volumes; (2) One or more users collecting packets from a "cloud" of source nodes storing the content. Throughput performance is characterized analytically and optimized or improved by parameter selection and code design. In particular, with scenario (2), a probability analysis is conducted using balls-into-bins models. We show that (1) the achievable throughput and energy performance with coding and without receiver feedback can beat that achievable with receiver feedback but without coding; (2) fast encoding and decoding is essential to increasing the throughput limits; (3) employing coding such as LT codes with optimized degree distribution can increase the efficiency of a simultaneous service to heterogeneous users; (4) coding with generations compensates for the lack of coordination between sources or between collectors; and (5) the proposed random annex codes based on coding with generations effectively reduces transmission redundancy by introducing overlaps between generations.

Acknowledgements

First of all, I would like to give my hearty thanks to Emina and Predrag for patiently guiding and supporting me all along, not only in academics and research, but also in life. Emina and Predrag are like parents to me. Words are not sufficient to express my gratitude. I would like to thank Prof. Roy D. Yates for valuable advice and suggestions on the improvement of this work. And I would like to thank WINlAB director Prof. Dipankar Raychaudhuri and all the WINLAB faculty, staff, and fellow students for keeping up a lively, efficient, and productive environment for research.

I would like to thank my parents for their selflessness. They gave out the best they could have offered me - the knowledge, the values, and everything. I would like to thank my extended family and family friends who helped me through the odyssey of coping with losses. I would also like to give special thanks to my friends Mengmeng Liu, Zhuowei Bao, Tingting Sun, Yan Wang, Tong Jin, my landlady Ping, my college classmates, and many others for generously extending their warmest hands in my dire days.

Thanks to all WINLAB alumni and colleagues for their friendship, including but not limited to Xiaojun Tang, Silvija Kokalj-Filipovic, Goran Ivkovic, Manik Raina, Manlin Xiao, Yu Zhang, Shu Chen, Jing Lei, Ruoheng Liu, Song Liu, Shengchao Yu, Liang Xiao, Lin Luo, Suli Zhao, Zhibin Wu, Xiangpeng Jing, Zang Li, Ke Na, Bin Zan, Wenjia Yuan, YiLin Tsai, Chenren Xu, Feixiong Zhang, Kai Su, Zhuo Chen, Tianming Li, Xiruo Liu, Ben Firner, Narayanan Krishnan, Hajar Mahdavi-Doost, Tam Vu, Mehrnaz Tavan, and Shridatt (James) Sugrim.

Lastly, I would like to acknowledge the thousands of individuals who have coded for the LaTeX project for free. It is due to their efforts that we can generate professionally typeset

PDFs now.

Chapter 3 contains material from [1]. Chapter 4 is based on [2]. Chapters 5 and 6 contain materials from [3].

Part of this dissertation work was supported by NSF CNS Grant No. 0721888.

Dedication

To Mom and Dad,

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Motivation

From audio tracks to video news clips and movies, today's internet is filled with traffic delivering bulk digital media content to billions of electronic devices, including desktop computers, laptops, smartphones, and even internet-enabled TV sets. In order for the users to be able to share the communications medium, large files are usually broken into smaller packets before injected into the internet for circulation. Due to unreliable physical links and network congestions, packets could be dropped on the way as a result of uncorrectable bit errors or buffer overflows. To recover packet losses, traditionally a retransmission mechanism (TCP, for example) is employed between the sender and the receiver. For single-link point-to-point duplex communications, such a mechanism is straightforward. However, when there are more than one senders and (or) more than one receivers involved, or when there is not timely feedback from the receiver to the sender, the mechanism could become complicated and less efficient, lowering service rate.

Service availability and latency can be improved through augmenting the accessible bandwidth and computing resources. This is attainable by deploying additional dedicated infrastructure, such as in Content Delivery Networks(CDN) (e.g. Akamai [**?**]; see [4] for a comprehensive survey), or by allowing peer users to assist each other, such as in peer-to-peer (P2P) systems (e.g. BitTorrent [5]; see [6] for a comprehensive survey). Nevertheless, the problem of planning these resources for efficient use is still nontrivial. Especially when facing heterogeneous and dynamic network environments and user requests, it becomes hard

to optimally schedule the content packets for delivery to a multitude of users.

Another approach is to encode across the packets instead of sending the original packets only. In this way, the information contained in a packet is "stitched" into multiple packets, making it possible to recover a missing packet by decoding the other packets received. In this work, we explore the use of packet-level coding in content networking in order to (1) combat packet losses without receiver feedback and (2) reduce the transmission overhead due to the lack of smart scheduling and coordination between users. Our coding schemes are developed on top of fountain codes and random linear network coding.

The concept of digital fountains was first proposed in [7], under which receivers are able to retrieve the content of interest as soon as they have collected a sufficient number of indistinguishable "drops" of the content, regardless of the specific collection of drops. The digital fountain is particularly appropriate for scenarios where a sender is unaware of or has difficulty keeping track of channel erasures. Fountain codes (rateless codes), such as LT codes [8] and Raptor codes [9] have been proposed for practical realization of digital fountains.

Meanwhile, the seminal work of Ahlswede et al. [10] reveals that in multicasts it is in general not optimal to regard the information as a "fluid" that can merely be routed or replicated. By employing coding at network nodes, an approach referred to as network coding, bandwidth required to deliver information can be saved. Following this information theoretic revelation, random linear network coding was proposed in [11] for "robust, distributed transmission and compression of information in networks". Later, the idea of circulating random linear combinations in the network was applied to peer-to-peer file distribution in [12]. In [12] it was observed from experimental results that sending random linear combinations of the packets shortens the downloading time and reduces the impact of peer churn and selfish peers on the overall throughput. However, analysis is lacking in characterizing the benefits. Our work proposes practical packet-level coding schemes and characterizes the achievable throughput with these coding schemes in content delivery and

content collection.

1.2 Organization

In the following Chapter 2, we introduce the basic packet-level coding strategies studied throughout this thesis. The letter notation used in this disseration is as defined in Chapter 2 unless otherwise stated.

The remainder of this thesis is presented in two parts. Part I studies coded content delivery from one server to one or more receivers when receiver feedback is not available for selective retransmissions. In Chapter 3, the throughput performance of a basic round-robin transmission scheme with generations (hyperblocks) is characterized theoretically. The accuracy of the analysis is verified against experimental measurements. It is also shown through experiment results that the achievable throughput and energy performance with coding and without receiver feedback can beat that achievable with receiver feedback but without coding, and that fast encoding and decoding is essential to increasing the throughput limits. Chapter 4 studies coded broadcast to heterogeneous users having varied demand volumes and experiencing diverse link qualities. In this chapter, LT codes, growth codes, and coding with generations are adapted for the heterogeneous scenario and the total number of transmissions required to deliver the demands of all users are evaluated and compared. Especially, LT codes are adapted by the optimization of the "degree distribution" of coded packets. It is shown that employing coding such as LT codes with optimized degree distribution can increase the efficiency of a simultaneous service to heterogeneous users. Part II studies another content networking scenario, where one or more collectors collect(s) content fragments from a "cloud" of storage nodes. The coding schemes studied in this scenario are based on coding with generations, and analysis is performed from the balls-into-bins perspective. Chapter 5 introduces mathematical preliminaries of some balls-into-bins models applicable to our studies. In particular, our extended results of the coupon brotherhood

problem [13, 14] are presented. These results are essential to the analysis of the throughput performance of random collection with coding with generations. Chapter 6 analyzes random collection with a single collector. It is shown that the throughput performance improves with increasing generation size. In addition, allowing overlaps between generations further improves the throughput. Random Annex Codes, a coding scheme with overlapping generations, are proposed. The throughput performance of the codes is analyzed and significant improvement is shown over previously proposed schemes with overlapping generations. Chapter 7 studies collection with multiple collectors, particularly, the union collection of the collectors.

Chapter 2

Packet-Level Coding Strategies for Content Networking

Throughout the thesis we consider the circulation of a content file $\mathcal{F}$ that is divided into N blocks (or packets, used interchangeably throughout this work), denoted as $\xi_1, \xi_2, \ldots, \xi_N$. Each packet $\xi_j (j = 1, 2, \ldots, N)$ can be represented as a bit vector of length b, or a vector of $b_q = b/\log_2 q$ symbols from $\mathbb{F}_q$, a finite field of size q. With a slight abuse of notation, ξ_j is used to denote any of the above representations, where q is determined by context.

Definition 1. *The file $\mathcal{F}$ is expressed as a set of N packets:*

$$\mathcal{F} = \{\xi_1, \xi_2, \ldots, \xi_N\}.$$

In this work, we focus on coding strategies that output coded packets as linear combinations of the original file packets.

One simple coding strategy is to generate and send random linear combinations of the file blocks by applying N linear combining coefficients $c_1, c_2, \ldots, c_N$ that are independently and equally probably drawn from $\mathbb{F}_q$. The coding coefficients (or a compressed version) are sent along with the coded block $\bar{\xi} = \sum_{j=1}^{N} c_j \xi_j$. Vector $c = (c_1, c_2, \ldots, c_N)$ is also referred to as the *coding vector* of the coded packet. Coded packets are linearly independent if their corresponding coding vectors are linearly independent. A receiver is said to have collected *k degrees of freedom (dofs)* if it has collected k linearly independent coded packets.

When a receiver has collected k dofs, the probability that a newly received coded packet is a linear combination of the existing coded packets is $q^k/q^N = q^{k-N}$, and therefore the expected number of coded packets required to increase the dofs in the collection by 1, which is the expected waiting time of the first success of Bernoulli trials with success

probability $1-q^{k-N}$, is $1/(1-q^{k-N})$. Thus, the expected number of coded packets required to accumulate 1 dof is at most a constant $1/(1-q^{(N-1)-N}) = q/(q-1)$. Hence, the receiver is able to collect N dofs as soon as it has collected around $N \cdot q/(q-1) = \mathcal{O}(N)$ coded packets. With each coded packet the receiver collects an equation in the form of

$$\sum_{j=1}^{N} c_j \xi_j = \bar{\xi},$$

where the original packets ξ_js are N unknown variables and c_js and $\bar{\xi}$ are the coding coefficients and the resulting linear combination known to the receiver. When N dofs have been collected, $\xi_1, \xi_2, \ldots, \xi_N$ can be decoded by solving the system of equations formed from all the coded packets.

A major concern in implementing such a coding scheme lies in the extra computational complexity involved that may hurt throughput and increase energy consumption. To solve for the packets by Gaussian elimination, $\mathcal{O}(N^3)$ $\mathbb{F}_q$-operations are required for matrix inversion and $\mathcal{O}(N^2 b_q)$ for multiplication. Experimental works such as [15] have reported difficulty in decoding more than 512 blocks together on a hand-held device. There have been attempts to reduce the complexity of encoding and decoding by sparsifying the coding vectors, that is, by reducing the number of nonzero entries in the coding vectors. In the remainder of this chapter, two representative approaches to coding complexity reduction are described, and these two approaches serve as the basis of the coding schemes proposed and investigated in our work.

2.1 LT Codes

LT codes were first proposed by Michael Luby in [8] as a class of codes designed to overcome variations in channel quality via ratelessness. Unlike their rateless predecessors, the ordinary random linear codes, LT codes can be decoded by simple suboptimal decoders with little loss in performance.

2.1.1 Encoding

The LT codes defined in Luby's original paper [8] are binary codes. An LT encoder outputs a potentially infinite stream of random linear combinations of the N original content packets, Each coded packet is generated independently as follows: first, pick a *degree* d according to a distribution defined by its moment generating function

$$P^{(N)}(x) = p_1 x + p_2 x^2 + \ldots + p_N x^N, \tag{2.1}$$

where $p_j \triangleq \text{Prob}[d = j]$ and the superscript $^(N)$ indicates that the degree distribution is defined for coding with a total of N packets; then, select with equal probability one of the $\binom{N}{d}$ possible combinations of d distinct packets from the N content packets, and form a coded packet of degree d by linearly combining the chosen packets over $\mathbb{F}_2$. The binary coding vector $c = (c_1, c_2, \ldots, c_N)$ has exactly d non-zero components corresponding to the chosen packets. The generated coded packet is given by $\sum_{j=1}^{N} c_j \xi_j$ and is of the same size b as each original file block. The N-bit coding vector c can be embedded into the coded packet and transmitted to the user. In wireless communications where data packets are generally of at most a few kilobytes' size, if $N \sim$1K, it is costly to add an extra N bits. Instead, the source can embed into the packet the degree of the coding vector and the seed of a pseudo-random number generator that is in shared knowledge of both the source and all the sink nodes, to allow the sink node to regenerate the coding vector locally.

2.1.2 Decoding

Decoding of LT codes is done by a belief propagation decoder [8, 9]. After receiving a certain number of coded packets, the receiver decoder attempts to decode, and the decoding process is described as follows. The decoder maintains a set called the *ripple*. The initial ripple is composed of all received coded packets of degree-1. In each decoding step, one coded packet in the ripple is processed by removing it from the ripple, substituting it into all the coded packets it participates in, which reduces the degrees of these coded packets. The

coded packets whose degrees are reduced to 1 are added to the ripple, and the decoder continue to process another coded packet in the ripple. The decoding process halts when the ripple becomes empty. If decoding halts before enough file packets have been decoded, the receiver waits to collect more coded packets and make another attempt to decode when more degree-1 coded packets have been received. It is provable that the order in which the degree-1 packets are processed does not affect the decoding result when the decoding process halts. Hence, for performance analysis purposes, we may as well assume each decoding attempt starts from forming the initial ripple from the whole set of (non-reduced) received coded packets.

As shown in [8, 9], the relationship between the number of coded packets collected and the number of decodable file packets is determined by the degree distribution used to generate the coded packets. This relationship will be elaborated in Section 4.3.1.

2.1.3 Computational Complexity

It was shown in [8] that, with the *robust soliton distribution* defined therein, each coded packet can be generated, independently of all other coded packets, on average by $\mathcal{O}(b \ln(N/\delta))$ xor operations, and the N original packets can be recovered from any $N + \mathcal{O}(\sqrt{N} \ln^2(N/\delta))$ of the coded packets with probability $1 - \delta$ by on average $\mathcal{O}(bN \ln(N/\delta))$ xor operations.

With the robust solition distribution, the fraction of degree-1 coded packets is low, and the growth of the number of decodable file packets with the growth of the number of collected coded packets displays a threshold behavior. That is, few packets are decodable when fewer than N coded packets have been collected, and most of the packets become decodable only after more than N coded packets have been collected. Such a property makes it inefficient to use the code if some receivers only demand a subset of the N packets. In Chapter 4, the degree distribution of LT codes are optimized for efficient delivery of multiple description coded content to heterogeneous users having diverse demand volumes and experiencing varied channel qualities.

2.2 Coding with Generations

Another approach to sparsify coding vectors is to group blocks into "hyperblocks" or "generations" so that the file packets involved in each coded packet always belong to the same generation. LT codes are of lower complexity than the random linear combination strategy introduced at the beginning of this chapter. However, if coded packets are to be further mixed at network nodes, it is hard to tune the degree distribution and maintain the sparsity of the coding vectors. Coding with generations, on the other hand, limits the maximum number of packets involved in each coded packet and keeps the size of the decoding problem small.

2.2.1 Forming Generations

Generations are non-empty subsets of the content file $\mathcal{F}$, which is a set of N packets, as defined at the beginning of this chapter.

Suppose that n generations, $G_1, G_2, \ldots, G_n$, are formed s.t. $\mathcal{F} = \cup_{j=1}^{n} G_j$. To introduce the basic coding scheme here, we assume the generations to be disjoint, i.e., $\forall i \neq j$, $G_i \cap G_j = \emptyset$. The size of each generation G_j is denoted by g_j, and its elements $\xi_1^{(j)}, \xi_2^{(j)}, \ldots, \xi_{g_j}^{(j)}$. For convenience, we will occasionally also use G_j to denote the matrix with columns $\xi_1^{(j)}, \xi_2^{(j)}, \ldots, \xi_{g_j}^{(j)}$. When all generations have a uniform size, use $g = g_1 = \cdots = g_n$ to denote the generation size.

In each transmission, the server first selects one of the n generations according to a certain rule, e.g. round-robin or at random. Once generation G_j is chosen, the source chooses a coding vector $c = (c_1, c_2, \ldots, c_{g_j})$, with each of the g_j components chosen from $\mathbb{F}_q$. A coded packet

$$\bar{\xi} = \sum_{i=1}^{g_j} c_i \xi_i^{(j)}$$

is then formed by linearly combining packets from G_j by c.

2.2.2 Decoding

For coding with disjoint generations, the decoder attempts decoding of generation G_j whenever a coded packet generated from G_j is received. Once there are g_j coded packets with linearly independent coding vectors in the receiver collection, the receiver can successfully decode G_j by Gaussian elimination. If the coding vectors are generated following some special rule, a faster decoding algorithm can be used, though a few more coded packets might be required, such as with LT codes. The decoding of file $\mathcal{F}$ is successful when all generations and hence all N file packets are decoded.

2.2.3 Packet Overhead

Contained in each coded packet are the index of a generation G_j and a linear combining vector for G_j which together take up $\lceil \log_2 n \rceil + g_j \lceil \log_2 q \rceil$ bits. Meanwhile, the data in each coded packet comprise $b_q \lceil \log_2 q \rceil$ bits. The generation size makes a more significant contribution to packet overhead than the number of generations, and such contribution is non-negligible due to the limited size ($\sim$ a few KB) of transmission packets in practical networks. This gives another reason to keep generations small, besides reducing computational complexity.

2.2.4 Computational Complexity

The computational complexity for encoding is $\mathcal{O}(b_q \max\{g_j\})$ per coded packet for linearly combining the g_j information packets in each generation (recall that b_q is the number of $\mathbb{F}_q$ symbols in each information packet, as defined in Section 2.2.1). For decoding, the largest number of unknowns in the systems of linear equations to be solved is not more than $\max\{g_j\}$, and therefore the computational complexity is upper bounded by $\mathcal{O}((\max\{g_j\})^2 + b_q \max\{g_j\})$ per original file packet.

Part I

Coded Content Delivery

Chapter 3

Round-Robin File Transfer with Generations

3.1 Introduction

We start Part I with a study of three coding-with-generations-based rateless coding schemes to combat random packet losses in a single-hop, feedback-less scenario. Two of the schemes are based on random linear coding, and the third is based on structured MDS coding such as Reed-Solomon (RS). All schemes follow round-robin scheduling. Since there is no feedback until the entire file has been downloaded, the round-robin protocol may result in many superfluous transmissions for already decoded generations.

This chapter is organized as follows: In Section 3.2, we introduce our coding and scheduling models, and define our performance measures. In Section 3.3, we present an analysis of the schemes. In Section 3.4, we describe the experimental setup, and present measurement results collected on a mobile platform. In Section 3.5, we discuss the results and future work.

3.2 Round-Robin Generation Scheduling

We consider transmission without feedback over a memoryless binary erasure channel between a sender and a receiver. In a packet network, the erasure rate is evaluated as the packet loss rate, denoted as ϵ. For the theoretical analysis, we assume that ϵ stays constant, regardless of time and the transmission protocol.

3.2.1 Performance Measures

We characterize the delivery packet count, which is defined as the number of packets that have to be sent until the receiver is able to recover the entire file of N packets. Delivery time is the time the receiver has to spend in the system until it is able to recover the content. It is determined by the delivery packet count, the packet size, and the rate of data transmission.

3.2.2 Coding within Generations

We study coding with generation with generations selected in a round-robin fashion: send one coded packet from each generation sequentially and wrap around.

As for the encoding scheme within the generations, we study three schemes: the random linear combination approach, the random linear combination approach with a systematic phase, and using an MDS (maximum distance separable) erasure code. We assume a uniform generation size of g.

Random Linear Combinations over $\mathbb{F}_q$ (RL)

In this scheme, each component $c_j (j = 1, \ldots, g)$ of each coding vector is chosen independently and equiprobably from $\mathbb{F}_q$. The resulting coded packet is $\sum_{i=1}^{g} c_j \xi_j$.

Random Linear Combinations Including a Systematic Phase (RLS)

This is a variation of the RL scheme that includes a systematic phase at the beginning: send the original packets from ξ_1 to ξ_N each once before starting to send random linear combinations of the original packets.

Maximum Distance Separable Codes (MDS)

Over a finite field of small size, such as the common binary field, random linear combinations chosen in the way specified in the RL scheme inevitably introduces non-negligible linear dependency between the coded packets. For short lengths of data, we can use low-rate

Maximum Distance Separable (MDS) Codes instead. An MDS(K,g) code is a code with which g packets are encoded into K coded packets, and all the g packets are recoverable as soon as any g of the K distinct coded packets have been collected. To extend transmission after the sender has exhausted all the K coded packets, the sender repeats the coded packets in a round-robin fashion. The parity check code is a binary MDS code where $K = g + 1$. Reed-Solomon codes are another important class of MDS codes that operate on GF(2^l) with $g < K < 2^l$. The increased complexity that comes with operations on a finite field of large size, however, can possibly undo the benefit brought by the MDS property, as we will later show in our experimental results.

3.3 Statistics of the Delivery Packet Count

In this section, we characterize T, the delivery packet count of coding within disjoint generations following the round-robin generation scheduling scheme. We assume that in each round, one coded packet is created from a generation that is selected from the n generations in a wrap-around fashion. After the tth transmission, $m_t = \lfloor t/n \rfloor$ rounds have been completed. By that time, $m_t + 1$ packets will have been sent from each of the first $r_t = t - m_t n$ generations, and m_t packets from each of the rest $n - r_t = (m_t + 1)n - t$ generations.

Since the generations are disjoint, each generation is decoded independently. Let $M_{g,\epsilon}$ be the number of coded packets needed to be sent over a link of packet erasure rate ϵ from a generation of size g so that the receiver can decode all file packets in the generation. Let $p_{m,g,\epsilon}$ be the probability that $M_{g,\epsilon} \leq m$. Let p_t be the probability that $T \leq t$. Then, since each generation is decoded independently, p_t is the product of the probabilities that each generation is decodable after t transmissions. We have

$$p_t = p_{m_t+1,g,\epsilon}^{r_t} p_{m_t,g,\epsilon}^{n-r_t}.$$

Note that since $p_{m,g,\epsilon}$ is the cumulative probability function of $M_{g,\epsilon}$, $p_{m,g,\epsilon}$ is non-decreasing

in m, and hence p_t is bounded as follows:

$$p^n_{m_t,g,\epsilon} \leq p_t < p^n_{m_t+1,g,\epsilon}. \tag{3.1}$$

Hence,

$$\begin{aligned} E[T] &= \sum_{t=0}^{\infty}(1-p_t) && (3.2)\\ &= \sum_{m_t=0}^{\infty}\sum_{r_t=0}^{n-1}(1-p^{r_t}_{m_t+1,g,\epsilon}p^{n-r_t}_{m_t,g,\epsilon}) \\ &= \sum_{m=0}^{\infty}\Big(n - p_{m,g,\epsilon}\frac{p^{n-1}_{m+1,g,\epsilon}-p^{n-1}_{m,g,\epsilon}}{p_{m+1,g,\epsilon}-p_{m,g,\epsilon}}\Big). && (3.3)\end{aligned}$$

And by (3.1) and (3.2), we have

$$n\sum_{m=1}^{\infty}(1-(p_{m,g,\epsilon})^n) < E[T] \leq n\sum_{m=0}^{\infty}(1-(p_{m,g,\epsilon})^n). \tag{3.4}$$

In the following, we characterize $p_{m,g,\epsilon}$ for different coding schemes within each generation.

3.3.1 RL Scheme

In this scheme, each coded packet is statistically the same; it is simply a random linear combination of the source packets. To decode a generation of size g, a number g of linearly independent coded packets must be received. When m coded packets have been transmitted over the channel with erasure rate ϵ, some $j \geq g$ have to be received, and among them g have to be linearly independent. Therefore, the probability $p^{\mathrm{RL}}_{m,g,\epsilon}$ of successful decoding, given $m \geq g$ coded packets have been transmitted is given as follows:

Claim 1.

$$p^{RL}_{m,g,\epsilon} = \sum_{j=g}^{m}\binom{m}{j}(1-\epsilon)^j\epsilon^{m-j}\prod_{s=0}^{g-1}(1-q^{s-j}) \tag{3.5}$$

The product in the equation is the probability that a $j \times g$ matrix with random entries chosen independently and equiprobably from GF(q) is of full column rank g. It is equal to

the probability of having g linearly independent coded packets among j coded packets. We can lower bound this product as follows (see [16, Lemma 7]):

$$\prod_{s=0}^{g-1}(1-q^{s-j}) \geq \begin{cases} 0.288, & \text{if } q=2 \text{ and } g=j; \\ 1-\frac{1}{q^{j-g}(q-1)}, & \text{otherwise.} \end{cases} \tag{3.6}$$

When q is large, we can further approximate $p^{\mathrm{RL}}_{m,g,\epsilon}$ as follows:

$$p^{\mathrm{RL}}_{m,\,g,\epsilon} \gtrsim \sum_{j=g}^{m}\binom{m}{j}(1-\epsilon)^j\epsilon^{m-j} - \frac{1}{q-1}\sum_{j=g}^{m}\binom{m}{j}(1-\epsilon)^j(q\epsilon)^{m-j}q^{g-m}$$

3.3.2 RLS Scheme

This scheme consists of two phases. In the first (systematic) phase, only uncoded packets are sent, and the second phase is the same as the RL scheme described above. For each generation, first each of the original g packets is transmitted once, followed by random linear combinations of all the packets.

Without loss of generality, we examine the decoding of generation G_1. None of the generations can be fully decoded before the systematic phase is finished. Hence,

$$p^{\mathrm{RLS}}_{m,g,\epsilon} = 0, \quad \text{for } m < g.$$

When $m = g$, it is clear that the generation can be successfully decoded if and only if none of the g uncoded transmissions is erased over the channel, and hence

$$p^{\mathrm{RLS}}_{g,g,\epsilon} = (1-\epsilon)^g.$$

Now consider $m > g$. Let A_S be the event that in the initial g uncoded transmissions, all the packets in some set $S \subseteq G_1$ have been received. Let $l = |S|$ be the number of packets in S. G_1 is fully decodable if $S = G_1$. Otherwise, $S \subset G_1$. The packets in S can be substituted into the equations formed from the coded packets received subsequently. The receiver needs to solve for $|G_1 \backslash S| = g - l$ unknowns instead of $|G_1| = g$ unknowns. Delete from the coding vectors the entries corresponding to the packets in S, and we refer to the resulting vectors of length $g - l$ as the residual coding vectors. Then, G_1 is decodable if and only if in the next

$m-g$ transmissions, at least $g-l$ coded packets with linearly independent residual coded vectors reach the receiver. Since each entry of the residual coding vectors are then still independently and equally probably chosen from $\mathbb{F}_q$, the probability that G_1 is decodable given A_S is exactly $p^{\mathrm{RL}}_{m-g,g-l,\epsilon}$. On the other hand,

$$\mathrm{Prob}[A_S] = (1-\epsilon)^{|}S|\epsilon^{g-|S|},$$

where $|S|$ is the number of packets in S. Therefore,

$$\begin{aligned} p^{\mathrm{RLS}}_{m,g,\epsilon} =& \mathrm{Prob}[A_{G_1}] + \sum_{S\subset G_1} \mathrm{Prob}[A_S] p^{\mathrm{RL}}_{m-g,g-|S|,\epsilon} \\ =& (1-\epsilon)^g + \sum_{S\subset G_1} (1-\epsilon)^{|S|}\epsilon^{g-|S|} p^{\mathrm{RL}}_{m-g,g-|S|,\epsilon} \\ =& (1-\epsilon)^g + \sum_{l=|S|=0}^{g-1} \binom{g}{l} (1-\epsilon)^l \epsilon^{g-l} p^{\mathrm{RL}}_{m-g,g-l,\epsilon}. \end{aligned}$$

We state in the following Claim 2 the above result along with a lower bound for $p^{\mathrm{RLS}}_{m,g,\epsilon}$.

Claim 2. *With the RLS scheme, the probability of receiving g linearly independent packets from m transmissions is*

$$\begin{aligned} p^{RLS}_{m,g,\epsilon} =& (1-\epsilon)^g + \sum_{l=0}^{g-1} \binom{g}{l} (1-\epsilon)^l \epsilon^{g-l} p^{RL}_{m-g,g-l,\epsilon} \\ \gtrsim& \sum_{j=g}^{m} \binom{m}{j} (1-\epsilon)^j \epsilon^{m-j} + \frac{(1-\epsilon)^g}{q-1} (\frac{1-\epsilon}{q} + \epsilon)^{m-g} - \\ & - \frac{1}{q-1} \sum_{j=g}^{m} \binom{m}{j} (1-\epsilon)^j \epsilon^{m-j} q^{g-j} \end{aligned} \tag{3.7}$$

Proof. Please refer to Appendix A. □

3.3.3 MDS Scheme

Suppose we use an MDS code which encodes g symbols into K symbols such that the g symbols can be entirely recovered as long as g distinct symbols have been received. We apply the code to generate K encoded packets from g original packets, and transmit the K encoded packets in a round-robin fashion.

Let $u_m = \lfloor \frac{m}{K} \rfloor$ and $v_m = m - u_m K$ be the quotient and the remainder of the number of transmissions m divided by the code block length K. Then, after m transmissions, the first v_m of the K encoded packets have been transmitted $u_m + 1$ times and the last $K - v_m$ of the K encoded packets have been transmitted u_m times. The probability that an encoded packet has been received is then $1 - \epsilon^{u_m+1}$ for any packet among the first v_m, and $1 - \epsilon^{u_m}$ among the last $K - v_m$. The probability $p^{\text{MDS}}_{m,K,g,\epsilon}$ of successful decoding of all the g packets, given m encoded packets have been transmitted, is equal to the probability that at least g of the K encoded packets have been received, or at most $K - g$ encoded packets have never been received. Therefore, $p^{\text{MDS}}_{m,K,g,\epsilon}$ can be computed by summing up the probability that l of the first v_m packets are absent and j of the remaining $K - v_m$ packets are absent in the receiver collection for all integers l and j satisfying $0 \le l + j \le K - g$.

Claim 3.

$$p^{\text{MDS}}_{m,K,g,\epsilon} = \sum_{l=0}^{K-g} \binom{v_m}{l} (\epsilon^{u_m+1})^l (1 - \epsilon^{u_m+1})^{v_m - l} \cdot \sum_{j=0}^{K-g-l} \binom{K - v_m}{j} (\epsilon^{u_m})^j (1 - \epsilon^{u_m})^{K - v_m - j}. \tag{3.8}$$

where $u_m = \lfloor \frac{m}{K} \rfloor$, $v_m = m - u_m K$, *and* $\binom{a}{b} = 0$ *for* $b > a$.

When $m \le K$, $u_m = 0$, $v_m = m$, (3.8) becomes

$$p^{\text{MDS}}_{m,K,g,\epsilon} = \sum_{l=0}^{m-g} \binom{m}{l} \epsilon^l (1 - \epsilon)^{m-l} = \sum_{j=g}^{m} \binom{m}{j} (1 - \epsilon)^j \epsilon^{m-j}.$$

When $K = g$, the code is the repetition code, and the right hand side of (3.8) becomes $(1 - \epsilon^{u_m+1})^{v_m} (1 - \epsilon^{u_m})^{K - v_m}$ because

$$\begin{aligned} &\sum_{l=0}^{K-g} \binom{v_m}{l} (\epsilon^{u_m+1})^l (1 - \epsilon^{u_m+1})^{v_m - l} \sum_{j=0}^{K-g-l} \binom{K - v_m}{j} (\epsilon^{u_m})^j (1 - \epsilon^{u_m})^{K - v_m - j} \\ &= \binom{v_m}{0} (\epsilon^{u_m+1})^0 (1 - \epsilon^{u_m+1})^{v_m} \binom{g - v_m}{0} (\epsilon^{u_m})^0 (1 - \epsilon^{u_m})^{K - v_m} \\ &= (1 - \epsilon^{u_m+1})^{v_m} (1 - \epsilon^{u_m})^{K - v_m}. \end{aligned}$$

3.4 Numerical and Experimental Results

To evaluate the performance of the schemes discussed in the previous section, the schemes were implemented on an experimental platform consisting of a laptop computer and a smartphone[1]. The time and the energy consumption required for the receiver to recover the whole file are measured. In this section, the experimental results are presented along with the theoretical predictions.

3.4.1 Experimental Setup

The experimental setup consists of an HP Pavilion dv5-1120eg laptop computer as a transmitter and a Nokia N8 smartphone as a receiver. The specifications for the Nokia N8 are shown in Table 3.1.

Table 3.1: Specifications of the Nokia N8

Operating System	Symbianˆ3
CPU	ARM11 @ 1 GHz
Memory	256 MB SDRAM
Display	640 x 360 pixels, 3.5 inch
Battery	BL-4D (3.7 V, 1200 mAh Li-Ion)

Both the laptop and the smartphone runs the same native C++ application (in sender and receiver mode, respectively) implemented using the Qt cross-platform application framework. The laptop transmits a file at a nominal application-layer data rate of 1000KB/s via UDP and using IEEE 802.11b at a physical layer rate of 11 Mbps. A transmitted file consists of 512 random packets having 1400 data bytes each. These data packets are encoded following the three encoding schemes described in Section 4.2. The receiving cell phone tries to decode the original file without sending any feedback information to the sender except for a final completion indicator transmitted only when the file is fully decoded. The sender stops transmission once it has received this completion signal.

[1]The experiment results and artworks presented in this section were contributed by Péter Vingelmann and Morten Videbæk Pedersen.

During the measurements the following information is recorded:

1. Number of packets sent before receiving the completion signal.
2. Number of packets received before sending the completion signal.
3. Time elapsed from the time when the first packet is received to the completion time.
4. Energy consumption by the receiver during the elapsed time. The test application uses the Control API of the Nokia Energy Profiler [17] to programmatically monitor (and record) the energy consumption of the mobile phone. The margin of error for these energy readings is 3%.

Each test was repeated 100 times for each generation size and encoding scheme pair. The following section presents the experimental results observed.

3.4.2 Results

Figure 3.1 shows theoretical predictions for the normalized delivery time (i.e. how many packets are needed to successfully deliver one packet) under typical channel conditions in our experimental setup. The predictions are calculated from (3.3) where $p_{m,g,\epsilon}$ is obtained from (3.5), (3.7), or (3.8). We observed that the packet loss rate (ϵ) is around 15% on an idle receiver when the sender is transmitting at a nominal rate of 1000KB/s. The RL and RLS schemes encode over the binary field, and the MDS schemes are represented by a Reed-Solomon (RS) code ($n = 255, K = g$) and a simple Parity Check (PC) code ($n = g + 1, K = g$) that has one parity symbol (all original symbols XORed together). We observe that the overhead per packet drops as the generation size increases, and thus the probability of transmitting a packet for an already decoded generation decreases. This is not true for PC(g+1,g) that can only cope with very low packet loss rates. The incorporation of a systematic phase in the random linear combination approach helps to reduce overhead for small generation sizes, but the gap quickly closes as the generation size increases. The Reed-Solomon code curve is near optimal since the code rates we use are much lower ($R < 0.51$)

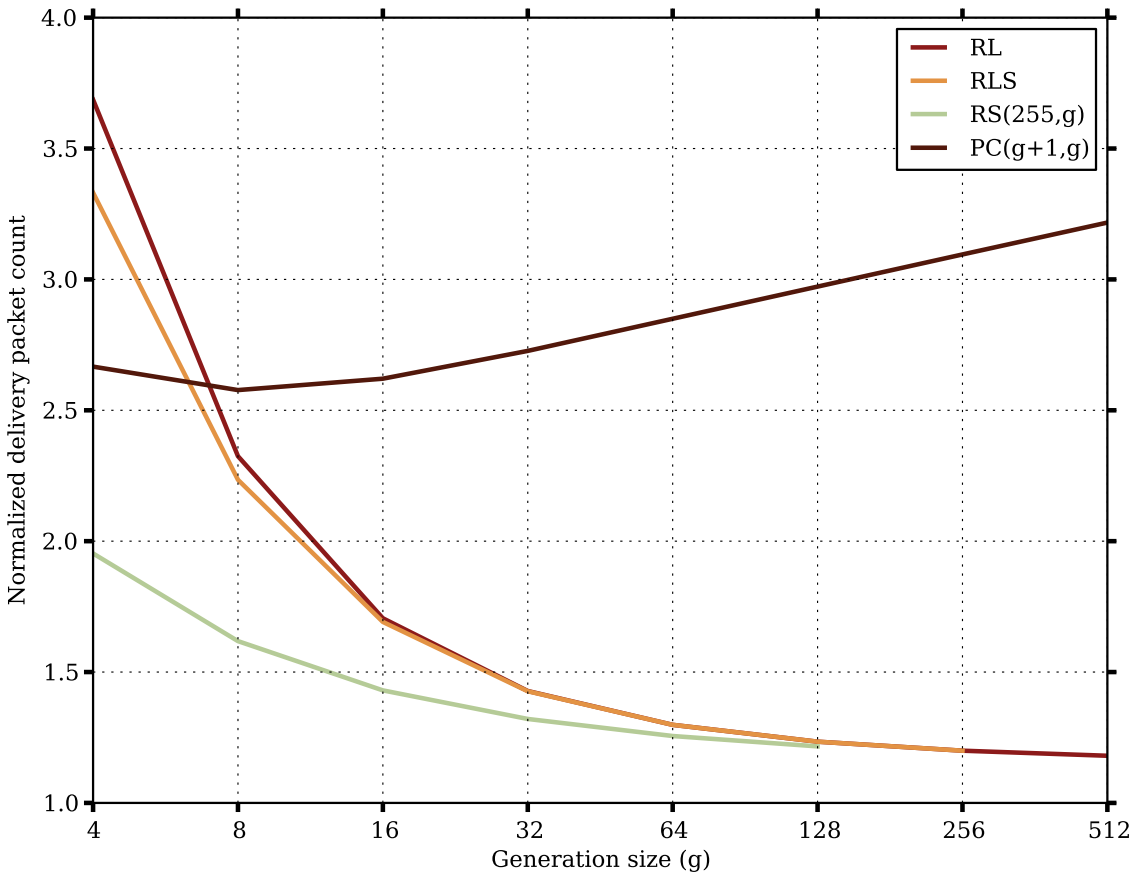


Figure 3.1: Predicted normalized expected delivery packet count (number of transmitted packets required to recover the entire file) versus generation size (assuming packet loss rate $\epsilon = 0.15$). RL: Random linear combinations. RLS: Random linear combinations with a systematic phase. RS(255,g): Reed-Solomon codes. PC($g+1$,g): A systematic code with a single coded packet as the bit-by-bit xor-sum of all file packets.

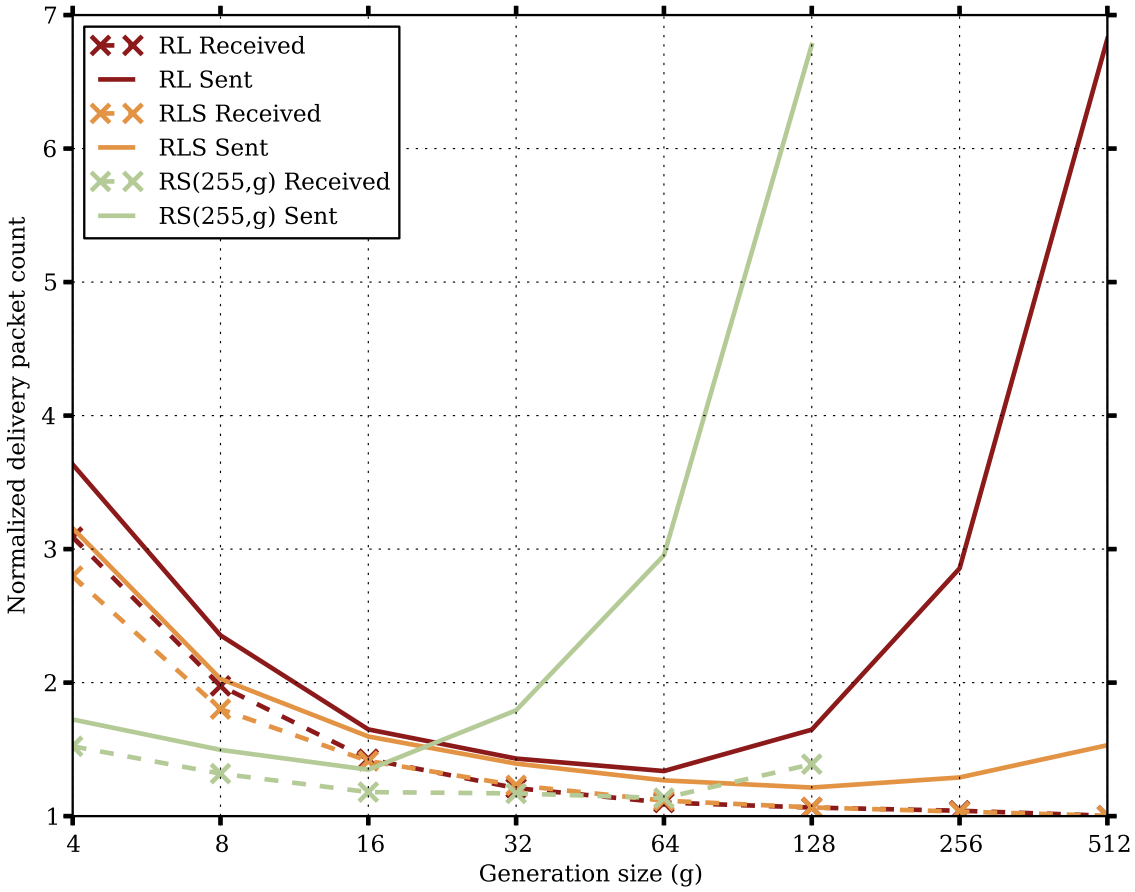


Figure 3.2: Measured delivery packet count versus generation size

than the packet loss rate, and with a high probability the transmission finishes before the sender runs out of the 255 coded packets for each generation.

Figure 3.2 shows the average number of packets sent per successfully delivered packet as measured in our experiments. This was calculated using the total number of packets sent and received divided by the number of packets in the test file (i.e. 512). For small generation sizes, we observe that increasing the generation size lowers the overhead per packet. These values are in accordance with the predictions in Figure 3.1. We would expect this trend to continue, since ideally we would use a single generation for the entire file. This would eliminate the possibility of transmitting packets that belong to an already decoded generation. However, this is not the optimal strategy in practice due to the increasing computational complexity. Figure 3.2 shows that the overhead per sent packet increases

significantly for the RS(255,g) scheme when $g > 16$, and for the RL scheme when $g > 64$. This indicates that the computational load on the receivers was too high, and they were unable to keep up with the transmission rate of the sender. The offset between the RS(255,g) and RL scheme may be explained by the larger field size used by the RS(255,g) scheme. Utilizing large fields (e.g. $q = 2^8$) typically requires some form of memory based look-up table to perform multiplication and division, whereas all operations in the binary field ($q = 2$) may be implemented using CPU instructions for binary XOR and AND operations. The RS implementation was based on a non-systematic Vandermonde matrix, other approaches such as utilizing binary Cauchy matrices [18] should be considered to further increase the performance of this implementation.

The lower computational requirements associated with the systematic packets in the RLS scheme clearly benefit the overall system performance. It is however worth noting that the systematic phase assumes that the receivers did not receive any packets previously. The systematic phase might lead to an additional overhead if the state of the receivers is initially unknown. The curve of the RLS scheme only deviates from the predicted values for very high generation sizes, 256 and 512.

Figure 3.3 shows that when we plug the average (application-layer) packet loss rate observed from the experiments (the loss rates are higher for larger generation sizes) into Claims 1-3, the theoretical predictions still match experimental data. This confirms the validity of our characterization.

The energy consumption of the communication system is especially important on battery-driven mobile devices. In Figure 3.4, we show the average energy consumption in Joules per file download. This is compared to the net throughput observed throughout the test. Due to the dominant impact of the wireless radio on the power consumption, we observe a significant connection between these two measured quantities. As the power consumption of the wireless radio remains relatively stable, when not in power-save or sleep mode, the energy consumption largely depends on the time needed to complete a test, and thus it is

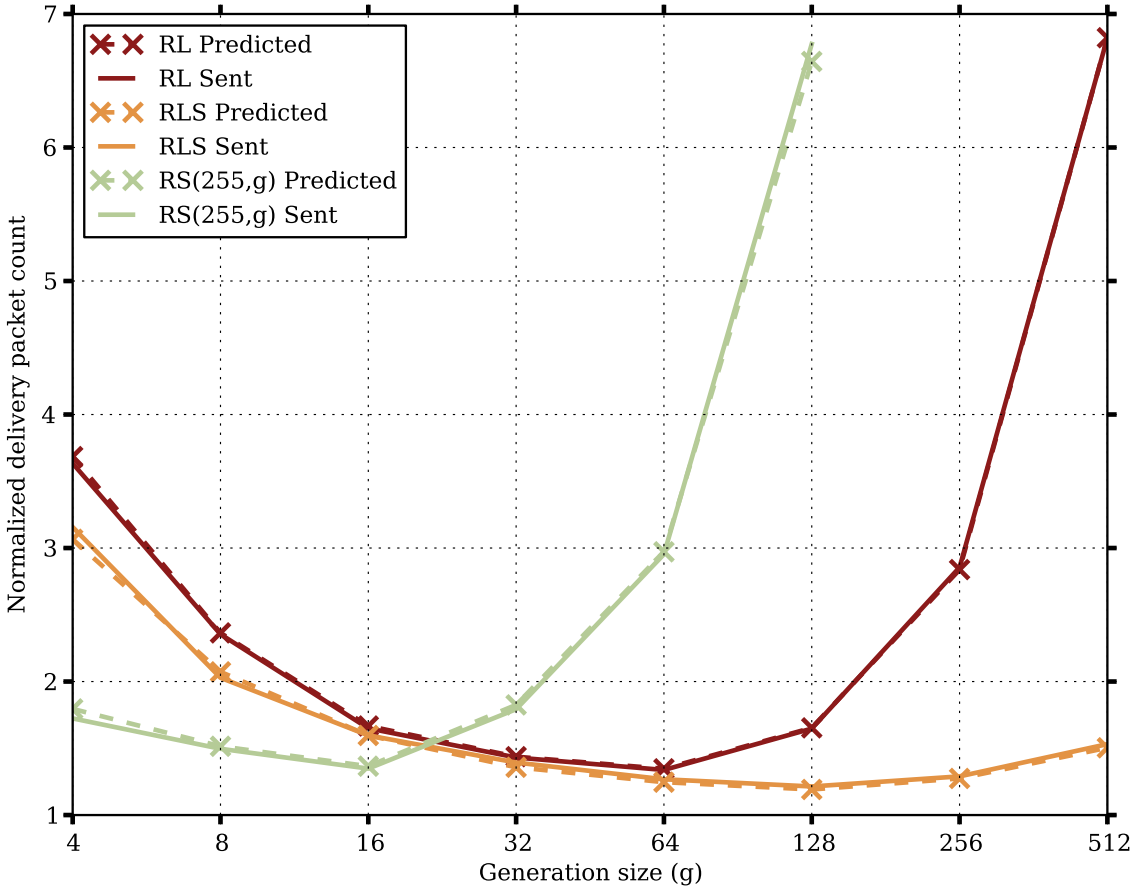


Figure 3.3: Measured delivery packet count compared to predictions calculated for measured packet loss rates

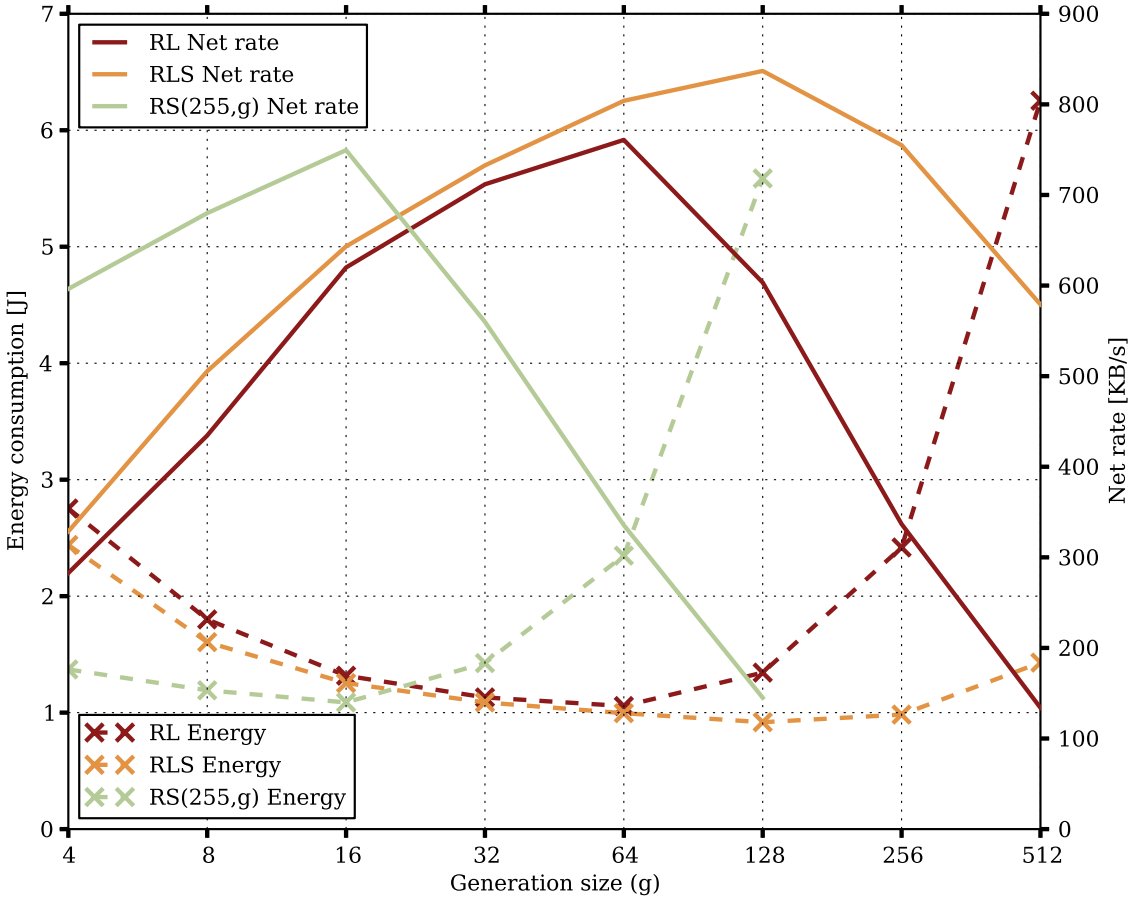


Figure 3.4: Net data rate ($= \frac{\text{file size}}{\text{delivery time}}$) and energy consumption versus generation size

inversely proportional to the net rate. In order to minimize the energy consumption of the protocol, we need to maximize the net rate.

When comparing the RS(255,g) to the binary-field RL and RLS codes, we observe that using a larger field size yields a better code performance at lower generation sizes. On the other hand, it is unable to sustain the low overhead as the generation size and thereby the computational complexity increases.

Although these results and the specific optimal values are certainly device- and system-dependent, we expect that other devices would exhibit similar tendencies, but the actual values would be shifted depending on the capabilities of the given platform. Faster devices might be able to support higher generation sizes and higher data rates.

3.5 Conclusion and Future Work

In this chapter, we considered three packet-level coding schemes for streaming over lossy links: random linear coding (RL), systematic random linear coding (RLS), and maximum distance separable coding (MDS). We characterized the exact distribution and the expected value of the delivery packet count of coding within disjoint generations following the round-robin generation scheduling scheme, taking into account the effect of field size and generation size. Our characterization matches experimental results.

The three coding schemes were implemented on a laptop computer and a Nokia N8 smartphone using the Qt cross-platform application framework. We presented measurement results collected during numerous experiments with various settings. Results show that the computational complexity has a significant impact on the performance of these schemes. The RLS scheme is the least computationally intensive, thereby it is able to achieve the highest net data rate and the lowest energy consumption.

In the future, we plan to implement other codes such as LT codes, Raptor codes, and systematic Reed-Solomon codes on the same testbed in order to compare their performance to the coding schemes discussed in this paper. The cost of random memory access and finite

field operations is non-negligible on a terminal with constrained capacity. A model should be devised that can account for these factors to give predictions on other platforms with different capabilities and constraints.

Acknowledgments

This work was partially financed by the CONE project (Grant No. 09-066549/FTP) granted by the Danish Ministry of Science, Technology and Innovation as well as by the collaboration with Renesas Mobile throughout the NOCE project.

Chapter 4

Coded Broadcast to Heterogeneous Users

4.1 Introduction

4.1.1 Motivation

In the past decade, the development of wireless networks has provided a fertile soil for popularization of portable digital devices, and the digital distribution of bulk media contents. This, in return, is stimulating new leaps in wireless communication technology. Today, devices retrieving digital video content in the air vary from 1080p HDTV sets to smartphones with 480×320-pixel screen resolution. Under assorted restraints in hardware, power, location, and mobility, these devices experience diverse link quality, differ in computing capability needed to retrieve information from received data streams, and request information of varied granularity. Consequently, a transmission scheme designed for one type of users may not be as suitable for another even if they demand the same content.

Today's technology implements a straightforward solution of separate transmissions to individual users, that is, multiple unicasts. Nonetheless, the key question is whether it is possible to deliver all users' demands with fewer data streams and less traffic. Especially in transmitting bulk data through wireless channels or over other shared media, reducing the amount of traffic is vital for reducing collision/interference, which in turn will also positively affect the quality of the channel in use. An additional concern in the environment conscious world is, of course, energy.

There is no surprise then that the problem of delivering more efficient service to a heterogeneous user community has attracted a great deal of both technical and academic

interest. On the source coding side, layered coding (e.g.[19, 20]) and multiple description coding [21] have been widely studied as solutions to providing rate scalability. In particular, with multiple description coding, a user is able to reconstruct a lower-quality version of the content upon receiving any one of the descriptions, and is able to improve the reconstruction quality upon receiving any additional description. Thus, the users' content reconstruction quality is commensurate with the quality of their connection. On the channel coding side, rateless codes [8, 9], or fountain codes, can generate a potentially infinite stream of coded symbols that can be optimized simultaneously for different channel erasure rates as long as the users have uniform demands.

In this work, we explore an achievable efficiency of serving users having heterogeneous demands while using a single broadcast stream. Whereas the information theoretical aspect of the problem is of interest and under investigation (see for example [22] and references therein), we focus on three practical coding schemes and explore their suitability for the described communications scenario. Two important features that make codes suitable for such scenarios are (1) the ability to support partial data recovery and (2) the ability to efficiently adapt to different channel conditions. Based on these desirable features, we chose to investigate three candidates: LT [8], growth [23], and chunked [24, 25] codes. In this chapter, we are particularly interested in the *total number of source transmissions* needed to deliver the demands of all users. This quantity determines the amount of required communication resources, and also translates to the amount of time required for delivery. In the streaming of temporally-segmented multimedia content that is delay-constrained, it is important for the source to finish transmitting a segment as soon as possible so as to proceed to the next one. Some performance measures of interest are addressed in [26].

4.1.2 Main Results

We compare three coding schemes and their variations for broadcasting to heterogeneous users: users suffer different packet loss rates and demand different amounts of data. The

coding schemes discussed include:

1. optimized LT codes, with or without a systematic phase, that is, one round of transmission of the original uncoded packets;
2. growth codes; and
3. chunked codes, equivalent to coding with generations (Section 2.2) with randomly scheduled disjoint generations of equal size, and using the RL coding scheme (Section 3.2.2) within each generation.

We also compare these schemes to a reference scheme for the heterogeneous scenario based on time-shared broadcast of degraded message sets [27]. We find that including a systematic phase is significantly beneficial towards delivering lower demands, but that coding is necessary for delivering higher demands. Different user demographics result in the suitability of very different coding schemes. Growth codes and chunked codes are not as suitable to this communication scenario as are optimized LT codes.

4.1.3 Organization

The rest of the chapter is organized as follows. Section 4.2 introduces the model of wireless broadcasting of multiple description coded content to heterogeneous users. Section 4.3 introduces the coding schemes of interest, and provide theoretical characterization of code performance. In particular, the LT codes are optimized both with and without a systematic phase. In Section 4.4 we provide numerical and simulation results of the achievable server delivery time, and discuss the suitability of these coded schemes for broadcast to heterogeneous users. The last section concludes.

4.2 System Model

Consider a wireless single-hop broadcast network consisting of a source (server) node holding content for distribution, and l sink (user) nodes waiting to retrieve the content from the

broadcast stream aired by the source, as shown in Figure 4.1. Suppose the content held

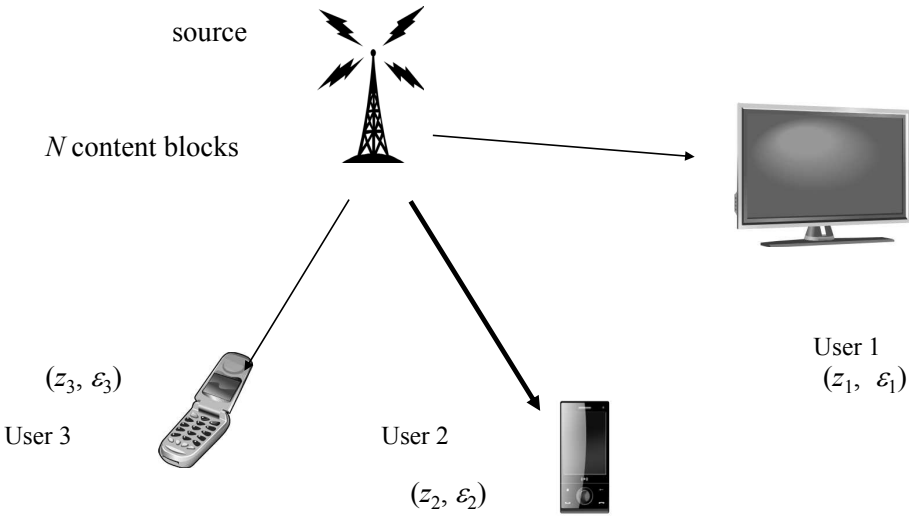


Figure 4.1: Network model

at the source node is multiple description coded into N descriptions, using, for example, one of the coding schemes described in [28] or in [29], and each description is packaged into one packet for transmission. Each packet is represented as a binary vector of length b, and denoted with ξ_j for $j = 1, 2, \ldots, N$. A low-quality version of the content can be reconstructed once a user is able to recover any description. The reconstruction quality improves progressively by recovering additional descriptions, and depends solely on the number of descriptions recovered, irrelevant of the particular recovered collection.

The source broadcasts one packet per unit time to all the sink nodes in the system. However, the sinks are connected to the source by lossy links, and at every sink, a packet either arrives at the sink intact or is entirely lost. Such an assumption is practical if we only consider data streams at the network level or higher. Under the multiple description

coding assumption, a sink node can choose to demand a smaller number of descriptions rather than wait to collect all N. This may not only shorten its own waiting time but also reduce the system burden. The demand of a sink node is characterized by the number of descriptions it needs to reconstruct the content within the desired distortion constraint.

Sink nodes are characterized by parameter pairs (z_i, ϵ_i), $i = 1, 2, \ldots, l$, describing their demands and link qualities:

- $z_i \in \{1/N, 2/N, \ldots, 1\}$ is the fraction of content demanded by user i, that is, each node demands $z_i N$ distinct descriptions.
- ϵ_i is the packet erasure (loss) rate on each link from the source to user i. A packet transmitted by the source fails to reach user i with probability ϵ_i. All links are assumed to be memoryless.

We further assume no feedback from the sinks to the source except for the initial requests to register the demands and the final confirmation of demand fulfillment. We also assume that the packet erasure rates ϵ_is on the links are known to the server. In wireless broadcast, feedback from multiple nodes increases the chance of collision and compromises the point-to-point channel quality. Hence, it is desirable to keep the feedback at a minimum.

Definition 2. *(General Delivery Time) The delivery time T_i of a sink is a random variable defined as the number of packets transmitted from the server until the fulfillment of the demand z_i of user i. The server delivery time $T = \max_{i=1}^{l} T_i$ is the number of packets transmitted from the server required to fulfill the demands of all the users.*

Since all users retrieve information from the same broadcast stream through channels of random erasures, T_i's and T are dependent random variables. Nevertheless, even their marginal probability distributions are not easy to characterize. Instead, in this paper, we normalize the delivery time by the total number of content packets N, and restrict our attention to either the expectation $t_i = E[T_i]/N$ or the asymptotics $t_i = \lim_{N\to\infty} T_i/N$, and study $\max_{i=1}^{l} t_i$ as the (normalized) server delivery time. When $t_i = E[T_i]/N$, $\max_{i=1}^{l} t_i$ is

a lower bound for $E[T]/N$, the normalized expected delivery time. For brevity we further abuse our notation and terminology and specialize the definition of delivery time to these lower bounds. The specialization of a definition will be stated where appropriate. In the rest of this paper, we describe and analyze three randomized coding schemes for broadcast in a heterogeneous setting, optimize if appropriate, compare their performance, and discuss their suitability to our end.

Note that with multiple unicasts, the total number of packets transmitted would be (in the mean) at least $\sum_{i=1}^{l} \frac{z_i}{1-\epsilon_i}$. We will show in our results that it is possible to do better using broadcast.

4.3 Three Coding Schemes for Broadcast to Heterogeneous Users

In this section, we describe three coding schemes and characterize their delivery time performance for broadcast to heterogeneous users. All the coding schemes studied in this chapter are applied at the packet level. Each coded packet is obtained as a random linear combination of the original packets, but either the number or the range of the packets involved in each linear combination is restricted.

4.3.1 LT Coded Broadcast

LT codes were first proposed by Michael Luby in [8] as a class of codes designed to overcome variations in channel quality via ratelessness. Unlike their rateless predecessors, the ordinary random linear codes, LT codes can be decoded by simple suboptimal decoders with little loss in performance. In [30], Sujay Sanghavi pointed out the inadequacy of the original LT codes for partial data recovery, and demonstrated the possibility of redesigning the LT codes. His focus was on scenarios with users having uniform demands. Here we extend the code optimization framework of [30] to adapt the code to the heterogeneous scenario in which both the channel conditions and the demands vary among users.

Characterization of the Recoverable Fraction of LT codes

Each user i needs to recover $z_i N$ content packets from the received coded packets.

Definition 3. *(Recoverable Fraction) A fraction z is said to be recoverable by the belief propagation decoder if the ripple size stays positive (at least) until zN packets get decoded.*

Based on Theorem 2 in [31], Corollary 5 gives the expected ripple size as a function of the recovered fraction of the content packets. We restate the part of Theorem 2 in [31] that concerns the expected size of the ripple as Theorem 4.

Assume $w \cdot N$ coded packets have been collected and fed into an LT decoder, for some positive constant w. Let $u \cdot N$ be the number of decoded content packets, for a constant $u \in [0, 1)$. Let $r^{(N)}(u)$ be the expected size of the ripple, normalized by N.

Theorem 4. *(Maatouk and Shokrollahi [31, Thm. 2]) If an LT code of N content packets has degree distribution specified by the moment generating function $P^{(N)}(x)$ (see (2.1)), then*

$$r^{(N)}(u) = wu\Big(P^{(N)\prime}(1-u) + \frac{1}{w}\ln u\Big) + O\Big(\frac{1}{N}\Big), \tag{4.1}$$

where $P^{(N)\prime}(x)$ stands for the first derivative of $P^{(N)}(x)$ with respect to x.

We further extend Theorem 4 to the case where the number of collected coded packets is random to accommodate random packets losses over the channel. Assume the number of collected coded packets is now $W \cdot N$, where W is a random variable with mean v. Denote the normalized expected ripple size as $N \to \infty$ as $r_W(u)$. Assuming that $P^{(N)}(x)$ converges to $P(x) = \sum_{i\geq 1} p_i x^i$ as $N \to \infty$, then the following holds:

Corollary 5.

$$r_W(u) = u\Big(vP'(1-u) + \ln u\Big). \tag{4.2}$$

Proof. Although Theorem 4 (Theorem 2 in [31]) is stated for the case where the number of coded packets collected by the sink node is more than the total number of content blocks, i.e., $w > 1$, its proof suggests that the theorem also holds for any constant $w < 1$.

Take $N \to \infty$ on both sides of 4.1, we have

$$r(u) = \lim_{N\to\infty} r^{(N)}(u) = u\big(wP'(1-u) + \ln u\big). \tag{4.3}$$

Replace w in (4.3) with W. Due to the linearity of the expected ripple size in W for given u and P,

$$r_W(u) \quad = \quad E\left[u\Big(WP'(1-u) + \ln u\Big)\right] = u\Big(vP'(1-u) + \ln u\Big)$$

□

If we use the expected value as a rough estimate of the ripple size during the decoding process, we should have $r_W(u) > 0$ for $u \in (1-z, 1]$. Applying (4.2), we have

$$vP'(1-u) + \ln u > 0, \quad \forall u \in (1-z, 1], \tag{4.4}$$

We next use (4.4) as a constraint to formulate an optimization problem for LT code degree distribution design to minimize the number of transmissions required to meet all sink demands.

Minimizing Server Delivery Time by Degree Distribution Design

We are concerned with the delivery time of LT coded broadcast in the asymptotic regime when $N \to \infty$.

Definition 4. *(LT Delivery Time) For the LT coded scheme, the (normalized) delivery time t_i is defined as the ratio of the number of transmissions required to fulfill the demand z_i of user i to the the total number N of content packets, as $N \to \infty$. The (normalized) server delivery time is taken as $t_0 = \max_{i=1}^{l}\{t_i\}$.*

The normalized number of coded packets user i receives by its delivery time t_i over a channel with the packet erasure rate ϵ_i is on average $t_i(1-\epsilon_i)$. Let $x = 1-u$ in (4.4) and $\omega = t_i(1-\epsilon_i)$; we have

$$(1-\epsilon_i)t_iP'(x) + \ln(1-x) > 0, \quad \forall x \in [0, z_i), \tag{4.5}$$

and consequently,

$$t_i = \inf\{\tau : (1-\epsilon_i)\tau P'(x) + \ln(1-x) > 0, \forall x \in [0, z_i)\}. \tag{4.6}$$

On the other hand, the recovered fraction by the user on a link of loss rate ϵ as a function of the number t of transmitted packets (normalized by N) is

$$z(t, \epsilon) = \sup\{\zeta : (1-\epsilon)tP'(x) + \ln(1-x) > 0, \forall x \in [0, \zeta)\}, \tag{4.7}$$

i.e., the recoverable fraction is the maximum ζ such that the expected ripple size stays positive in the whole range of $[0, \zeta]$.

We now have all elements to state an optimization problem for minimize the server delivery time t. Using (4.5), the optimization problem is expressed as follows:

$$\begin{aligned} \min._{P,t_1,\ldots,t_l} \quad & t_0 = \max_i \ t_i \\ \text{s.t.} \quad & t_i(1-\epsilon_i)P'(x) + \ln(1-x) > 0, \quad 0 \le x \le z_i, \quad \text{for } i = 1, 2, \ldots, l; \\ & P(1) = 1, \end{aligned} \tag{4.8}$$

or equivalently,

$$\begin{aligned} \min._{P,t_0} \quad & t_0 \\ \text{s.t.} \quad & t_0(1-\epsilon_i)P'(x) + \ln(1-x) > 0, \quad 0 \le x \le z_i, \quad \text{for } i = 1, 2, \ldots, l. \\ & P(1) = 1. \end{aligned} \tag{4.9}$$

To solve the above optimization problems, we first observe that there exist optimal $P(x)$ which is a polynomial of finite degree, as in the following Claim 6.

Claim 6. *There must exist an optimal solution to Problem (4.9) with a polynomial $P(x)$ of degree no higher than*

$$d_{\max} = \lceil \frac{1}{1 - \max_i\{z_i\}} \rceil - 1.$$

Proof. This claim is provable with an argument similar to Lemma 2 of [30]. Suppose (t^*, P^*) is an optimal solution of Problem (4.9). construct $\bar{P}$ such that

$$\bar{p}_j = p_j^*$$

for $j = 1, 2, \ldots, d_{\max} - 1$, and

$$\bar{p}_{d_{\max}} = \sum_{j \geq d_{\max}} p_j^*.$$

Then $\bar{P}(x)$ still represents a degree distribution, and meanwhile,

$$\begin{aligned}
\bar{P}'(x) - P^{*\prime}(x) &= \sum_{j=1}^{d_{\max}} j\bar{p}_j x^{j-1} - \sum_{j \geq 1} j p_j^* x^{j-1} \\
&= \sum_{j=1}^{d_{\max}-1} j p_j^* x^{j-1} + d_{\max} \sum_{j \geq d_{\max}} p_j^* x^{d_{\max}-1} - \sum_{j=1}^{d_{\max}-1} j p_j^* x^{j-1} - \sum_{j \geq d_{\max}} j p_j^* x^{j-1} \\
&= \sum_{j \geq d_{\max}} [d_{\max} p_j^* x^{d_{\max}-1} - j p_j^* x^{j-1}] \\
&= \sum_{j \geq d_{\max}+1} p_j^* x^{d_{\max}-1} [d_{\max} - j x^{j-d_{\max}}] \\
&\geq \sum_{j \geq d_{\max}+1} p_j^* x^{d_{\max}-1} [d_{\max} - (d_{\max}+1)x].
\end{aligned}$$

As long as

$$x \leq \frac{d_{\max}}{d_{\max}+1},$$

i.e.,

$$d_{\max} \geq \frac{x}{1-x} = \frac{1}{1-x} - 1,$$

we have

$$\bar{P}'(x) \geq P^{*\prime}(x).$$

Thus, set

$$d_{\max} = \lceil \frac{1}{1 - \max_i z_i} \rceil - 1 \geq \lceil \frac{1}{1 - z_i} \rceil - 1 \geq \frac{1}{1 - z_i} - 1,$$

we have for $i = 1, 2, \ldots, l$,

$$t^*(1 - \epsilon_i)\bar{P}'(x) + \ln(1-x) \geq t^*(1-\epsilon_i)P^{*\prime}(x) + \ln(1-x) > 0, \quad 0 \leq x \leq z_i,$$

and hence $(t^*, \bar{P})$ is also a feasible and optimal solution of Problem (4.9) with optimal value t^*, and the highest degree of $\bar{P}$ is no more than

$$d_{\max} = \lceil \frac{1}{1 - \max_i\{z_i\}} \rceil - 1.$$

□

Thus, Problem (4.9) can readily be converted into a linear programming problem by the method proposed in [30]. For $j = 1, 2, \ldots, d_{\max}$, let $a_j = tp_j$, and Problem (4.9) becomes

$$\begin{aligned} \min._{a_1,\ldots,a_{d_{\max}}} \quad & \sum_{j=1}^{d_{\max}} a_j \\ \text{s.t.} \quad & \sum_{j=1}^{d_{\max}} j a_j x^{j-1} > -\frac{\ln(1-x)}{1-\epsilon_i}, \quad 0 \le x \le z_i, \quad \text{for } i = 1, 2, \ldots, l. \\ & a_j \ge 0, j = 1, 2, \ldots, d_{\max}. \end{aligned} \tag{4.10}$$

To solve the linear programming problem (4.10) numerically, the constraints defined on a continuous interval of parameter x are written out by evaluating x at discrete points within the interval. Lower bounds for the minimum value of (4.10), and (4.8) can thus be obtained. (In Section 4.4, we interestingly observe that in a 2-user scenario, the optimal server delivery time obtained from the optimization described here is close to the delivery time achievable by using a time-sharing scheme to broadcast degraded message sets. The time-sharing scheme is described in 4.3.4.)

LT Coding with a Systematic (Uncoded) Phase

We also study a variation of the LT codes that start with a systematic phase, namely, transmission of all the original uncoded content packets (systematic packets) followed by parity packets.

The formulation of the degree distribution optimization problem is essentially the same, except that (4.5) (describing the condition which the number of transmissions and the degree distribution must satisfy to allow the delivery of the demand of each user i) becomes constraint (4.11) in the following Claim 7.

Claim 7. *Suppose after the systematic phase, the degree distribution of the parity packets transmitted subsequently follows the distribution represented by $P(x)$, as $N \to \infty$. Then, to recover a fraction z_i, the normalized number of parity packets transmitted should satisfy*

$$-\ln(\epsilon_i) + (1-\epsilon_i)(t_i - 1)P'(x) + \ln(1-x) > 0, \quad \forall x \in (1-\epsilon_i, z_i). \tag{4.11}$$

Proof. Please refer to Appendix B. □

The new optimization problem is still readily transformable into a linear programming problem.

In addition, we have

$$t_i = \begin{cases} \frac{z_i}{1-\epsilon_i}, & z_i \leq 1-\epsilon_i; \\ \inf\{\tau : -\ln(\epsilon_i) + (1-\epsilon_i)(\tau_i - 1)P'(x) + \ln(1-x) > 0, \forall x \in [0, z_i)\}, & z_i > 1-\epsilon_i. \end{cases} \tag{4.12}$$

The systematic phase delivers the demand of a user with $z_i \leq 1-\epsilon_i$, which helps reduce the server delivery time. It is similarly possible to formulate optimization problems to allow transmitting less or more than one round of systematic symbols, and find out the tradeoff between the fraction of systematic symbols and non-systematic symbols. However, this is beyond the scope of this chapter.

4.3.2 Growth Codes

Growth codes were proposed by Kamra et al. in [23] to improve data persistence in sensor networks in face of sensor node failure. Growth codes were not designed for our heterogeneous scenario described in Section 4.2. However, their feature of progressive partial recovery suggests that they may be a good candidate scheme. Extensions and applications of growth codes to video streaming has been studied in, e.g., [32, 33]. Particularly, in [32], a systematic version of growth codes with unequal protection for layer coded video content was proposed.

Growth coded packets are, as LT coded packets, binary random linear combinations of the original source packets, and can be decoded by a belief propagation decoder that is essentially identical to that of the LT codes, as described in Section **??**. But, unlike an LT coded stream which produces statistically identical packets, a growth coded stream starts with degree-1 coded packets and gradually move on to send coded packets of higher degrees. The encoding scheme described in [23] operates as follows. Let $R_j = \frac{jN-1}{j+1}$ for $j = 0, 1, 2, \ldots, N-1$. Let

$$A_j = \sum_{s=\lfloor R_{j-1} \rfloor + 1}^{\lfloor R_j \rfloor} \frac{\binom{N}{j}}{\binom{s}{j}(N-s)}$$

for $j = 1, 2, \ldots, N-1$. Then, on a perfect erasure free channel, the source node sends A_1 degree-1 coded packets followed by A_2 degree-2 coded packets, A_3 degree-3 packets, and so on. Demand of size between R_{j-1} and R_j is expected to be fulfilled during the phase when degree-j coded packets are sent.

Growth codes are based on the design philosophy to greedily ensure the highest probability of recovering a new content packet upon receiving each additional coded packet. On a link with loss probability ϵ, it is reasonable to scale each duration A_j in which degree-j coded packets are transmitted by $1/(1-\epsilon)$ so as to keep the degree distribution of the packets reaching the sink approximately the same as if the code runs on a perfect channel. There is no straightforward way to extend such design philosophy and scaling approach to broadcasting over channels of different erasure rates, but we can still scale A_js for one of the users and see the resulting delivery time for other users and the server, and search for the scaling factor with which the server delivery time is minimized. We use the belief propagation LT decoder to decode growth codes, and use (4.6) to predict the delivery time (as defined in Definition 4) and recoverable fraction of the scaled version of growth codes as $N \to \infty$. We compare growth codes with the optimized LT codes in our heterogenous scenarios in Section 4.4.

4.3.3 Chunked Codes

Chunked codes was first proposed and studied in [24]. Chunked codes are coding with generations with uniformly random generation scheduling. It is also investigated in Chapter 6 via a coupon collection analysis. The N content packets are grouped into n disjoint generations of g packets(assuming N is a multiple of g), and these generations are also called "chunks". Packets are represented as vectors of symbols from $\mathbb{F}_q$. In each transmission, we first uniformly, randomly select a generation, and then sample from $\mathbb{F}_q^g$ a coding vector uniformly at random and form a linear combination of the content packets within the selected chunk. As soon as a sink node has collected g coded packets with linearly independent coding vectors generated from the same chunk, all the packets of this chunk can be decoded by performing Gaussian elimination on $\mathbb{F}_q$. As opposed to full network coding, with which coding vectors are chosen from $\mathbb{F}_q^N$, this scheme has lower computational complexity and also to some extent allows partial recovery.

From Chapter 6, we know that, for a sufficiently large field size q, the expected number of transmissions needed to decode any k of the n chunks on a unicast link subject to packet loss rate of ϵ is given as

$$E[T(n,k,\epsilon)] = \frac{n}{1-\epsilon}\int_0^\infty \left\{\sum_{j=0}^{k-1}\binom{n}{i} S_h^{n-j}(x)\left[e^x - S_g(x)\right]^j\right\} e^{-nx}dx, \tag{4.13}$$

where

$$\begin{aligned} S_m(x) =& 1 + \frac{x}{1!} + \frac{x^2}{2!} + \cdots + \frac{x^{m-1}}{(m-1)!} \quad (m \geq 1) \\ S_\infty(x) =& \exp(x) \text{ and } S_0(x) = 0. \end{aligned}$$

The above result is based on the generalized birthday problem in [34].

Based on the above, we specialize our definition for delivery time when coding with chunked codes.

Definition 5. *(Chunked Codes Delivery Time) With the chunked-coding scheme, the normalized delivery time of user i is defined as $t_i = E[T(n, \lceil z_i n \rceil, \epsilon_i)]/N$. The normalized*

server delivery time is defined as $t_0 = \max_{i=1}^{l} t_i$.

4.3.4 Lower Bound and Reference Schemes

A Lower Bound

An obvious lower bound for the minimum server delivery time is

$$t = \max_{i=1}^{l}\{\frac{z_i}{1-\epsilon_i}\}.$$

Multiple Unicasts

If instead of broadcast, the server transmits separate streams to each user, the total normalized number of transmissions required will be at least

$$t = \sum_{i=1}^{l} \frac{z_i}{1-\epsilon_i}.$$

Braodcast Degraded Message Sets by Timesharing

Here we describe a reference coding scheme. Without loss of generality, assume $z_0 = 0 < z_1 \leq z_2 \leq \cdots \leq z_l$. Then, segment N descriptions/packets into l layers, with Layer $i (i = 1, 2, \ldots, l)$ containing $L_i = (z_i - z_{i-1})N$ packets. Protect Layer i by an erasure code of rate $R_i = 1 - \max\{\epsilon_i, \epsilon_{i+1}, \ldots, \epsilon_l\}$, and transmit the protected layers sequentially. When N goes to infinity, there exist erasure codes that allow the server to deliver Layers 1 through i, that is $z_i N$ packets, to user i for all $i = 1, 2, \ldots, l$ in $\sum_{i=1}^{l} \frac{L_i}{R_i}$ time. Later we will find in Section 4.4.2 that in a 2-user scenario the server delivery time of this scheme is close to that of the optimized LT codes without a systematic phase.

4.4 Performance Comparison

In this section we evaluate the schemes described in Section 4.3 by numerical calculation and simulation for a 2-user scenario.

4.4.1 Partial Recovery Curves

We demonstrate in Figure 4.2 the evolution of the fraction of recoverable content packets at sink nodes with the growth of the number of transmissions from the source in the 2-user broadcast scenario. The erasure rates are $\epsilon_1 = 0.1$ and $\epsilon_2 = 0.5$. User 2 has a worse channel. Shown are the performance curves of optimized LT codes (with and without a systematic phase), growth codes, and chunked codes at both users. The details in obtaining the numerical results are listed below.

- LT codes (systematic and non-systematic)
  - The degree distributions are obtained by solving the optimization problems in 4.3.1 and 4.3.1 by setting $(z_1, \epsilon_1) = (15/16, 0.1)$ and $(z_2, \epsilon_2) = (9/16, 0.5)$. For the non-systematic version, the optimal

    $$P(x) = 0.0195x + 0.7814x^2 + 0.1991x^3;$$

    for the systematic version, the optimal

    $$P(x) = 0.7061x^2 + 0.2939x^3.$$

  - The delivery time is computed for given z_1 and z_2 based on (4.6) in the nonsystematic case and on (4.12) in the systematic case. The optimal server delivery time is 1.5178 for the nonsystematic version, and 1.2488 in the systematic version.
- Growth codes
  - $N = 1024$. The time spent transmitting coded packets of each degree i, A_i, is scaled by a factor of $1/(1 - \epsilon_1)$, that is, the code is adapted to the channel conditions of user 1.
  - The recoverable fraction is computed by evaluating (4.7) on the empirical degree distribution of received packets. Note that (4.7) assumes $N \to \infty$.

- Chunked Codes
  - We set the number of packets to $N = 1024$, the number of chunks to $n = 16$, and thus the chunk size becomes $h = N/n = 64$.
  - We compute $E[T(n, \lceil zn \rceil, \epsilon_i)]/N$ with $E[T(n, k, \epsilon)]$ given by (4.13).

Simulation results show the average time required for the user to recover a given fraction z of all the packets. The average is taken over 100 runs, and the number of source packets N is set to be 1024 in all runs. For chunked codes, the finite field size q is set to be 256. It is shown that the simulation results are close to the theoretical performance prediction for the coding schemes studied, except for the difference in the latter stage of growth codes, which comes from the slight difference in the coding scheme numerically evaluated and the one simulated. In our simulations, after finishing sending A_m degree-m coded packets where $m = \lceil \frac{1}{\max_{i=1,2}\{1-z_i\}} \rceil$, instead of proceeding to sending packets of higher degrees, as defined in the original scheme, we allow the server to send coded packets according to a degree distribution $p_j = \frac{A_j}{\sum_{j=1}^{m} A_j}$ for $j = 1, 2, \ldots, m$, the cumulative degree distribution up to stage A_m. This provides the user on the worse channel with the low-degree packets that are needed but have been lost due to a higher packet loss rate, and allows the user to proceed in packet recovery. All the coding schemes exhibit partial recovery property to a corresponding degree, namely, the recoverable fraction gradually increases with time. With both optimized LT (without the systematic phase) and the chunked codes, there is an initial stage where no packets are recoverable.

4.4.2 Delivery Time

We continue with a study of the server delivery time performance of the coding schemes in the 2-user scenario. We keep $\epsilon_1 = 0.1$ and $\epsilon_2 = 0.5$, set $z_1 = 15/16$, let z_2 vary from 0 to 15/16, and plot the server delivery time versus z_2. With LT codes, for each set of (z_1, z_2) values, we solve for the optimal degree distribution. With growth codes, for each z_2, we

search all scaling factors between $[\frac{1}{1-\epsilon_1}, \frac{1}{1-\epsilon_2}]$ for the one that minimizes the server delivery time. With chunked codes, we examine all power-of-2 chunk sizes between $h = 2^0 = 1$ and $h = 2^{10} = N$, and find the chunk size that minimizes the server delivery time. The numerical evaluation of the minimized server delivery time is plotted versus the demand z_2 of user 2 in Figure 4.3. Note that the optimal degree distribution changes as the demands vary.

The server delivery time of the reference scheme via time-shared broadcast of degraded message sets, as described in 4.3.4, is also plotted. In the two-user case discussed here, $z_1 \geq z_2$ and $\epsilon_1 > \epsilon_2$. Therefore, two layers are formed, the first consisting of $L_1 = z_2 N$ blocks and the second another $L_2 = (z_1 - z_2)N$ blocks, and respectively protected with erasure codes of rates $1 - \epsilon_1$ and $1 - \epsilon_2$. It is particularly interesting to find that in the setting demonstrated here, the reference scheme performance almost coincides with the non-systematic optimized LT coded scheme with optimized degree distribution. The reason of such a phenomenon, however, awaits further investigation and is beyond the scope of this work.

In addition, in Figure 4.3, we plot two other reference lines:

$$t = \max\{\frac{z_1}{1-\epsilon_1}, \frac{z_2}{1-\epsilon_2}\}$$

and

$$t = \frac{z_1}{1-\epsilon_1} + \frac{z_2}{1-\epsilon_2}.$$

The former represents an apparent lower bound for server delivery time, and the latter represents a lower bound for the total number of packet transmissions if the demands are delivered by unicasts to each individual user.

Comparing the server delivery time of systematic and nonsystematic LT codes optimized with the knowledge of channel state and user demands, we find that the systematic phase significantly shortens the server delivery time when the demand of the user on the worse channel is low. As the demand of user 2 rises, however, including a systematic phase

becomes suboptimal, until the demand rises further closer to 1, when the delivery time seems to converge to the delivery time with the nonsystematic scheme.

Growth codes and chunked codes do not have a competitive server delivery time performance compared with other schemes. When the demand of the user on the worse channel is low, using these schemes requires more transmissions to deliver the demands than using multiple unicasts. One advantage of chunked codes, however, is that all the content packets in the same chunk are decoded simultaneously, allowing these packets to be dependent on each other for the reconstruction purpose, which may reduce redundancy in the source coding stage.

It is also of interest to study performance measures other than the delivery time of the server. For example, one could wish to maximize the minimum of users' throughputs $\min_i\{z_i/t_i\}$, the ratio of the demand and the time needed to complete the demand, or to maximize the minimum channel utilization $\min_i\{z_i/((1-\epsilon_i)t_i)\}$, the ratio of the information pushed through the channel to the channel bandwidth. These criteria are of interest for acieving, e.g., fairness in the network, and optimization for these criteria can yield very different code design parameters (degree distributions). The optimization of the degree distributions for LT coded broadcast for these different criteria has been studied in our work [26] and interested readers are kindly referred to this paper for detailed problem formulation and results. An additional type of heterogeneity can also be treated in the framework provided in Section 4.3.1, namely, when some of the sink nodes cannot decode but can only recover content from degree-1 coded packets. Readers are also referred to [26] for results regarding the coexistence of nodes able and unable to decode in the system.

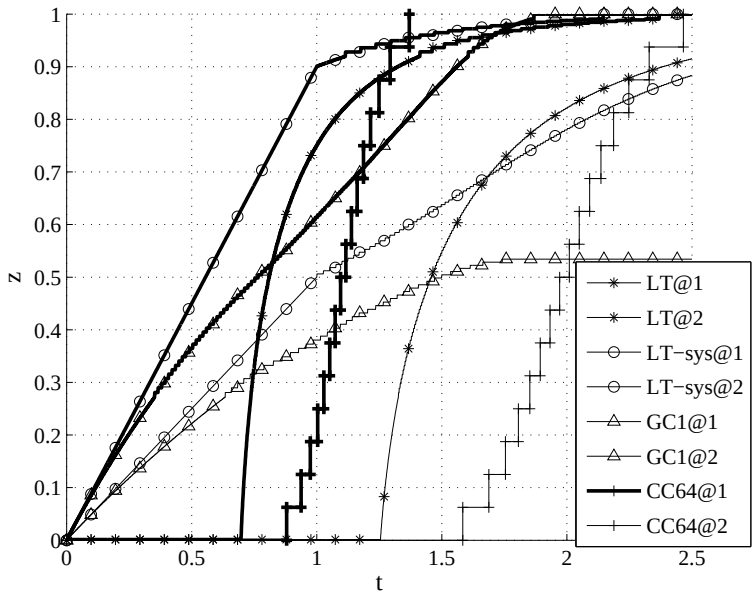
4.5 Conclusion and Future Work

We investigated the usage of three coded schemes for broadcasting multiple description coded content in a single-hop wireless network with heterogeneous user nodes of nonuniform demand over links of diverse packet loss rates. The three coded schemes include

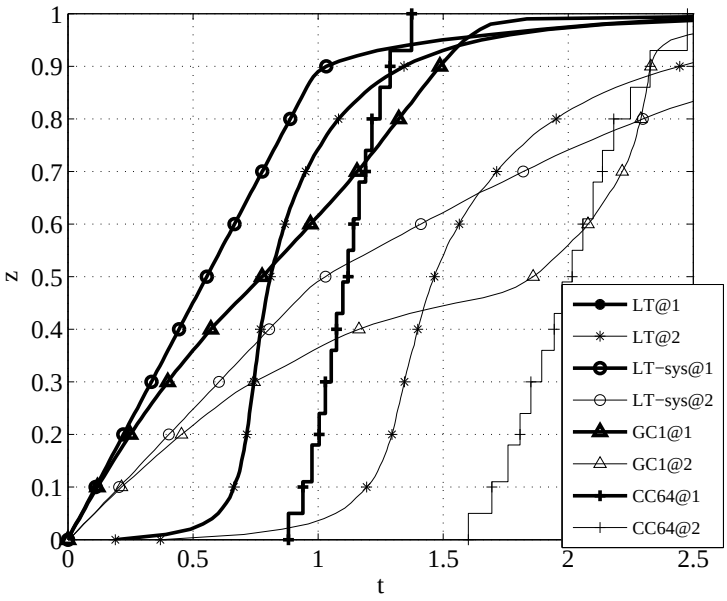
the LT codes with specially optimized degree distribution (with or without a systematic phase), growth codes, and chunked codes. Particularly, with the LT codes, we are able to formulate the degree distribution design problem in the heterogeneous scenario into linear optimization problems. For the schemes compared, we characterize numerically the number of transmissions needed to deliver the demand of all users. The numerical evaluations agree with simulation results.

A systematic phase delivers the demands efficiently when the fraction of content requested by the users does not exceed the link capacity. On the other hand, for higher demands, coding significantly improves the delivery time. Different user demographics result in very different coding schemes being suitable for efficient delivery of demands. Growth codes and chunked codes, are not found to be as suitable to the communication scenario as the optimized LT codes. Interestingly, time-shared broadcast of degraded message sets is found to give a comparable delivery time performance as that of the non-systematic optimized LT codes.

As for future work, we are interested in incorporating the source coding stage into code design, since there is clearly an interplay between the compression efficiency in source coding and the efficiency of channel coding. We are also interested in the exploration and analysis of more competitive schemes for broadcasting to heterogeneous users.



(a) Numerical Results



(b) Simulation Results

Figure 4.2: Evolution of recoverable fraction with time at each user in a 2-user scenario. $\epsilon_1 = 0.1$, $\epsilon_2 = 0.5$. $N = 1024$. Schemes: (1) LT optimized with $z_1 = 15/16$ and $z_2 = 9/16$. (2)optimal systematic LT (3) growth codes with a scaling factor of $\frac{1}{1-\epsilon_1}$ (4) chunked codes with $n = 16$ chunks of 64 packets. LT(-sys): Optimized LT codes (with a systematic phase); GC1: Growth codes scaled by $\frac{1}{1-\epsilon_1}$; CC_{64}: Chunked codes with 64 packets per chunk.

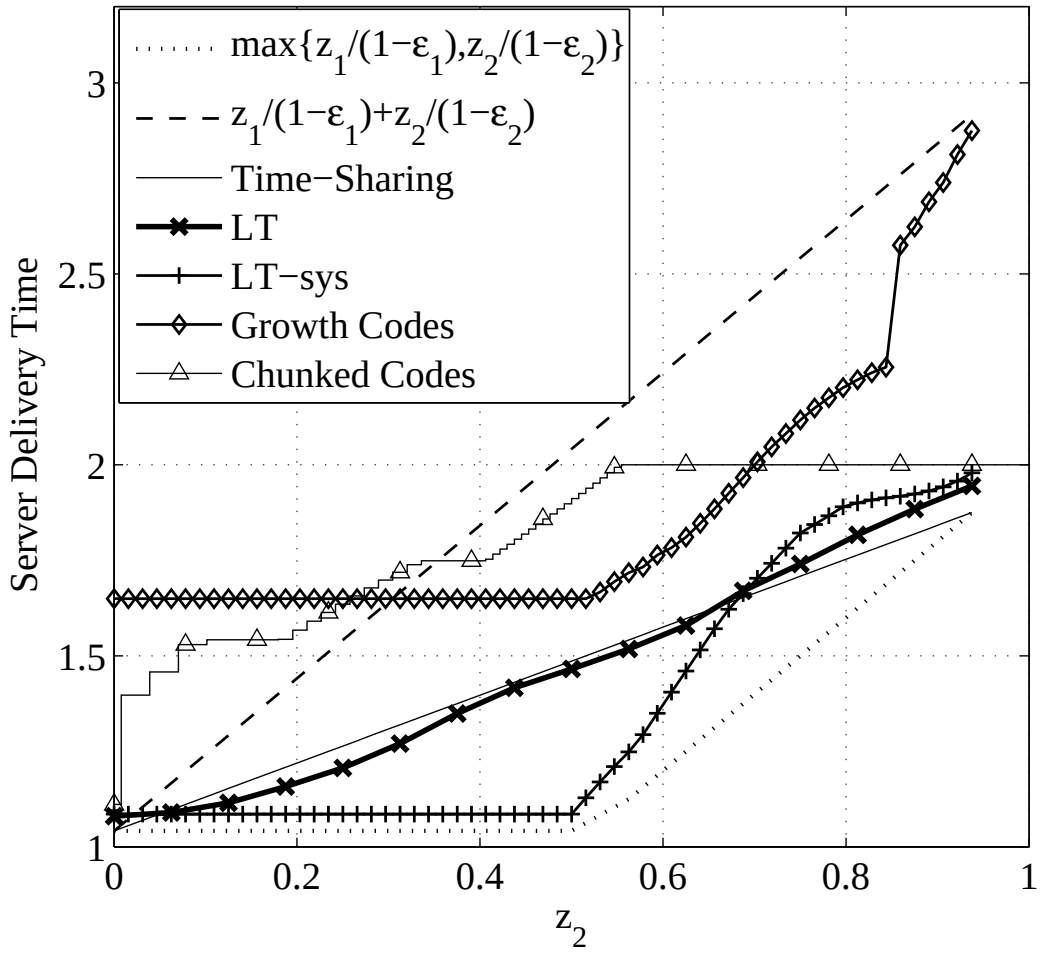


Figure 4.3: Comparison of the server delivery time t: $N = 1024$ descriptions/packets ($N \to \infty$ for LT codes and growth codes), $\epsilon_1 = 0.1$, $\epsilon_2 = 0.5$, $z_1 = 15/16$, z_2 ranging from 0 to 15/16.

Part II

Content Collection in a Balls-into-Bins Perspective

Chapter 5

Collecting Content Blocks as Allocating Balls into Bins

In this part, we consider a content collection scenario. Suppose a data collector (sink, receiver) in the network wishes to collect a file of N packets from a number of servers in the network. Each server may have a complete or incomplete collection of the N packets, and sends a packet to the receiver in each transmission. The collection process can be viewed as the shooting (allocation) of balls from the server into N bins kept by the receiver. The receiver collects Packet ξ_j if a ball lands in Bin j, and when each bin is filled with at least 1 ball the receiver has collected the whole file. We are interested in the number of shots (packet transmissions) required for the receiver to collect the whole file, and refer to this number as the *collection time*.

The servers are considered collectively as a "super server", and packets are transmitted as if from the super server to the receiver(s). In Chapter 6, a single receiver is considered and the communication is deemed to be on a unicast link. In Chapter 7, multiple receivers are considered and multiple unicast links.

If the servers send packets regardless of which packets have already been received by the receiver, which packets are sent by other servers, and which packets have been sent previously, we can approximately deem the super server as to be selecting (scheduling) each transmission packet independently and equally likely from the N possible choices. That is, each shot independently lands in one of the N bins with equal probability. This is the classical coupon collector's problem, and the expected collection time is well known as $N \sum_{i=1}^{N} \frac{1}{i} = \mathcal{O}(N \log N)$ [35]. The overhead in this scenario increases superlinearly as the file size grows.

Instead of sending randomly selected packets, the servers can send encoded packets that are linear combinations of the file packets over $\mathbb{F}_q$, a finite field of size q, as discussed in Chapter 2. For practical reasons, such as computational complexity, coding with generations (Chapter 2) is introduced. The concept of generation in network coding was first proposed by Chou et al. in [36] to handle the issue of network synchronization. Coding with randomly scheduled generations was first theoretically analyzed by Maymounkov et al. in [24]. Coding with generations scheduled round-robin was studied in Chapter 3 of this dissertation.

If there are n generations, each generation is viewed as a bin, and each coded packet is a ball. Shooting (allocating) a ball into Bin j is analogous to transmitting a coded packet formed from generation G_j. To successfully decode, the receiver needs to collect a certain number of coded packets for each generation $G_j (j = 1, 2, \ldots n)$, and the number is $\mathcal{O}(g_j)$, g_j being the generation size. We are interested in the collection time, the total number of coded packets transmitted when enough has been collected from each generation. In the balls-and-bins language, we are interested in the total number of balls allocated when each bin has been filled with the required number of balls.

In the remainder of this chapter, we present the mathematical theory of random allocation (of balls into bins) that is used for the analysis of coded content collection with generations in Chapters 6 and 7. Balls are allocated into n bins. There are two basic types of random allocation: random allocation with replacement and without replacement. With the former, balls are allocated into bins one-by-one and independently, not barring the possibility of allocating a ball into a bin in which some previous ball has landed, whereas with the latter, each ball is ensured to be allocated into a different bin.

5.1 Coupon Collector's Problem and Collector's Brotherhood Problem

The coupon collector's problem [13, 14] is based on the simplest random allocation model with replacement. The coupon collector's brotherhood problem studies quantities related to the completion of m sets of n distinct coupons by sampling a set of n distinct coupons

uniformly at random with replacement. Here a coupon is a bin, and sampling a coupon is allocating a ball into a bin. In analogy, coded packets belonging to generation G_j can be viewed as copies of coupon G_j(with a slight abuse of terminology), and hence the process of collecting coded packets when generations are scheduled uniformly at random can be modeled as collecting multiple copies of distinct coupons. To be more general, we assume that the probability of sampling a coupon G_j is ρ_j, for $j = 1, 2, \ldots, n$.

Because of possible linear dependence among coded packets between generations, the numbers of coded packets needed for each of the n generations to ensure successful decoding, however, are n random variables. Therefore, we must generalize the coupon collector's brotherhood model from collecting a uniform number of copies for all coupons to collecting different numbers of copies for different coupons, before it can be applied to the analysis of the collection time of a file coded with generations. In this section, the original collector's brotherhood model is generalized in two ways. And later in this paper, the analysis of the collection time of coding with disjoint generations in Section 6.2 rests on the first generalization, whereas that of coding with overlapping generations in Section 6.3 rests on the second generalization.

5.1.1 Generating Functions, Expected Values and Variances

For any $m \in \mathbb{N}$, we define $S_m(x)$ as follows:

$$S_m(x) = 1 + \frac{x}{1!} + \frac{x^2}{2!} + \cdots + \frac{x^{m-1}}{(m-1)!} \quad (m \geq 1) \tag{5.1}$$

$$S_m(x) = 0 \quad (m \leq 0) \text{ and } S_\infty(x) = e^x. \tag{5.2}$$

Let the total number of samplings needed to ensure that at least $m_j (\geq 0)$ copies of coupon G_j are collected for all $j = 1, 2, \ldots, n$ be $T(\boldsymbol{\rho}, \mathbf{m})$, where $\mathbf{m} = (m_1, m_2, \ldots, m_n)$. The following Theorem 8 gives $\varphi_{T(\boldsymbol{\rho},\mathbf{m})}(z)$, the generating function of the tail probabilities of $T(\boldsymbol{\rho}, \mathbf{m})$. This result is generalized from [13] and [14], and its proof uses the Newman-Shepp symbolic method in [13]. Boneh et al. [37] gave the same generalization, but we restate it

here for use in our analysis of coding with disjoint generations (Section **??**). If for each $j = 1, 2, \ldots, n$, the number of coded packets needed from generation G_j for its decoding is known to be m_j (which can be strictly larger than the generation size g_j), $T(\boldsymbol{\rho}, \mathbf{m})$ then gives the total number of coded packets needed to ensure successful decoding of the entire content when the generations are scheduled according to the probability vector $\boldsymbol{\rho}$.

Theorem 8. *(Non-Uniform Sampling) Let*

$$\varphi_{T(\boldsymbol{\rho},\mathbf{m})}(z) = \sum_{i\geq 0} \text{Prob}[T(\boldsymbol{\rho}, \mathbf{m}) > i] z^i. \tag{5.3}$$

Then,

$$\varphi_{T(\boldsymbol{\rho},\mathbf{m})}(z) = \int_0^\infty \left\{ e^{-x(1-z)} - \prod_{i=1}^{n} \left[e^{-\rho_i x(1-z)} - S_{m_i}(\rho_i x z) e^{-\rho_i x} \right] \right\} dx. \tag{5.4}$$

Proof. Please refer to Appendix C.2, where we give a full proof of the theorem to demonstrate the Newman-Shepp symbolic method [13], which is also used in the proof of our other generalization in Theorem 10. □

The expected value and the variance of $T(\boldsymbol{\rho}, \mathbf{m})$ follow from the tail probability generating function derived in Theorem 8.

Corollary 9.

$$\begin{aligned} E[T(\boldsymbol{\rho}, \mathbf{m})] &= \varphi_{T(\boldsymbol{\rho},\mathbf{m})}(1) \\ &= \int_0^\infty \left\{ 1 - \prod_{i=1}^{n} \left[1 - S_{m_i}(\rho_i x) e^{-\rho_i x} \right] \right\} dx, \\ Var[T(\boldsymbol{\rho}, \mathbf{m})] &= 2\varphi'_{T(\boldsymbol{\rho},\mathbf{m})}(1) + \varphi_{T(\boldsymbol{\rho},\mathbf{m})}(1) - \varphi^2_{T(\boldsymbol{\rho},\mathbf{m})}(1). \end{aligned}$$

Proof. Please refer to Appendix C.2. □

Note that in Theorem 8 and Corollary 9, m_i-s are allowed to be 0, thus including the case where only a specific subset of the coupons is of interest. Theorem 8 and Corollary 9 are also useful for the analysis of coding over generations when there is a difference in priority among the generations. For instance, in layered coded multimedia content, the

generations containing the packets of the basic layer could be given a higher priority than those containing enhancement layers because of a hierarchical reconstruction at the receiver.

In the following, we present another generalization of the collector's brotherhood model. Sometimes we are simply interested in collecting a coupon subset of a certain size, regardless of the specific content of the subset. This can be further extended to the following more complicated case: for each $i = 1, 2, \ldots, A (A \geq 1)$, ensure that there exists a subset of $\{G_1, G_2, \ldots, G_n\}$ such that each of its k_i elements has at least m_i copies in the collected samples. Such a generalization is intended for treatment of coding over equally important generations, for example, when each generation is a substream of multiple-description coded data. In this generalization, the generation scheduling (coupon sampling) probabilities are assumed to be uniform, i.e., $\rho_1 = \rho_2 = \cdots = \rho_n = 1/n$.

Suppose that for some positive integer $A \leq n$, integers $k_1, \ldots, k_A$ and $m_1, \ldots, m_A$ satisfy $1 \leq k_1 < \cdots < k_A \leq n$ and $\infty = m_0 > m_1 > \cdots > m_A > m_{A+1} = 0$. We are interested in the total number $U(\mathbf{m}, \mathbf{k})$ of coupons that needs to be collected, to ensure that the number of distinct coupons for which at least m_i copies have been collected is at least k_i, for all $i = 1, 2, \ldots, A$, where $\mathbf{m} = (m_1, m_2, \ldots, m_A)$ and $\mathbf{k} = (k_1, k_2, \ldots, k_A)$. The following Theorem 10 gives the generating function $\varphi_{U(\mathbf{m},\mathbf{k})}(z)$ of $U(\mathbf{m}, \mathbf{k})$.

Theorem 10. *(Uniform Sampling)*

$$\varphi_{U(\mathbf{m},\mathbf{k})}(z) \tag{5.5}$$
$$= n \int_0^\infty e^{-nx} \Big\{ e^{nxz} - \sum_{\substack{(i_0, i_1, \ldots, i_{A+1}): \\ i_0 = 0, i_{A+1} = n \\ i_j \in [k_j, i_{j+1}] \\ j = 1, 2, \ldots, A}} \prod_{j=0}^{A} \binom{i_{j+1}}{i_j} \Big[S_{m_j}(xz) - S_{m_{j+1}}(xz) \Big]^{i_{j+1} - i_j} \Big\} dx.$$

Proof. Please refer to Appendix C.2. □

Same as for Corollary 9, we can find $E[U(\mathbf{m}, \mathbf{k})] = \varphi_{U(\mathbf{m},\mathbf{k})}(1)$. A computationally wieldy representation of $E[U(\mathbf{m}, \mathbf{k})]$ is offered in the following Corollary 11 in a recursive form.

Corollary 11. *For $k = k_1, k_1 + 1, \ldots, n$, let*

$$\phi_{0,k}(x) = [(S_{m_0}(x) - S_{m_1}(x))e^{-x}]^k;$$

For $j = 1, 2, \ldots, A$, let

$$\phi_{j,k}(x) = \sum_{w=k_j}^{k} \binom{k}{w} \left[(S_{m_j}(x) - S_{m_{j+1}}(x))e^{-x}\right]^{k-w} \phi_{j-1,w}(x),$$

$$\text{for } k = k_{j+1}, k_{j+1} + 1, \ldots, n.$$

Then,

$$E[U(\mathbf{m}, \mathbf{k})] = n \int_0^\infty (1 - \phi_{A,n}(x))\, dx. \tag{5.6}$$

It is not hard to find an algorithm that calculates $1 - \phi_{A,n}(x)$ in $(c_1 m_1 + c_2(n-1) + c_3 \sum_{j=1}^{A} \sum_{k=k_{j+1}}^{n} (k - k_j))$ basic arithmetic operations, where c_1, c_2 and c_3 are positive constants. As long as $m_1 = \mathcal{O}(An^2)$, we can estimate the amount of work for a single evaluation of $1 - \phi_{A,n}(x)$ to be $\mathcal{O}(An^2)$. The integral (5.6) can be computed through the use of an efficient quadrature method, for example, Gauss-Laguerre quadrature. For reference, some numerical integration issues for the special case where $A = 1$ have been addressed in Part 7 of [34] and in [37].

In Section 6.3, we will apply Corollary 11 to find out the expected throughput of the *random annex code*, an overlapping coding scheme in which generations share randomly chosen file packets. The effect of the overlap size on the throughput can be investigated henceforth.

5.1.2 Limiting Mean Value and Distribution

In the previous subsection, we considered collecting a finite number of copies of a coupon set of a finite size. In this part, we present some results from existing literature on the limiting behavior of $T(\boldsymbol{\rho}, \mathbf{m})$ as $n \to \infty$ or $m_1 = m_2 = \cdots = m_n = m \to \infty$, assuming $\rho_1 = \rho_2 = \cdots = \rho_n = \frac{1}{n}$. By slight abuse in notation, we denote $T(\boldsymbol{\rho}, \mathbf{m})$ here as $T_n(m)$.

By Corollary 9,

$$E[T_n(m)] = n \int_0^\infty \left[1 - (1 - S_m(x)e^{-x})^n\right] dx. \tag{5.7}$$

The asymptotics of $E[T_n(m)]$ for large n has been discussed in literature [13], [38] and [39], and is summarized in the following Theorem 12, (5.9), and Theorem 13.

Theorem 12. *([38]) When $n \to \infty$,*

$$E[T_n(m)] = n \log n + (m-1)n \log\log n + C_m n + \mathcal{O}(n), \tag{5.8}$$

where $C_m = \gamma - \log(m-1)!$, γ is Euler's constant, and $m \in \mathbb{N}$.

For $m \gg 1$, on the other hand, we have [13]

$$E[T_n(m)] \to nm. \tag{5.9}$$

What is worth mentioning is that, as the number of coupons $n \to \infty$, for the first complete set of coupons, the number of samplings needed is $\mathcal{O}(n \log n)$, whereas the additional number of samplings needed for each additional set is only $\mathcal{O}(n \log\log n)$.

In addition to the expected value of $T_n(m)$, the concentration of $T_n(m)$ around its mean is also of great interest to us. This concentration leads to an estimate of the probability of successful decoding for a given number of collected coded packets. We can specialize Corollary 9 to derive the variance of $T_n(m)$, as a measure of probability concentration.

Further, since the tail probability generating functions derived in the last subsection are power series of non-negative coefficients and are convergent at 1, they are absolutely convergent on and inside the circle $|z| = 1$ in the complex z-plane. Thus, it is possible to compute the tail probabilities using Cauchy's contour integration formula. However, extra care is required for numerical stability in such computation.

Here we instead look at the asymptotic case where the number of coupons $n \to \infty$. Erdös and Rényi have proven in [40] the limit law of $T_n(m)$ as $n \to \infty$. Here we restate Lemma B from [38] by Flatto, which in addition expresses the rate of convergence to the limit law.

We will later use this result to derive a lower bound for the probability of decoding failure in Theorem 16 in Section 6.2.2.

Theorem 13. *([38]) Let*

$$Y_n(m) = \frac{1}{n}\left(T_n(m) - n\log n - (m-1)n\log\log n\right).$$

Then,

$$\Pr[Y_n(m) \le y] = \exp\left(-\frac{e^{-y}}{(m-1)!}\right) + \mathcal{O}\left(\frac{\log\log n}{\log n}\right).$$

Remark 1. *(Remarks 2&3, [38]) The estimation in Theorem 13 is understood to hold uniformly on any finite interval $-a \le y \le a$. i.e., for any $a > 0$,*

$$\left|\text{Prob}\left[Y_n(m) \le y\right] - \exp\left(-\frac{\exp(-y)}{(m-1)!}\right)\right| \le C(m,a)\frac{\log\log n}{\log n},$$

$n \ge 2$ and $-a \le y \le a$. $C(m,a)$ is a positive constant depending on m and a, but independent of n. For $m = 1$, the convergence rate to limit law is much faster: the $\mathcal{O}\left(\frac{\log\log n}{\log n}\right)$ term becomes $\mathcal{O}\left(\frac{\log n}{n}\right)$.

5.2 Random Allocation with Complexes

A more advanced type of random allocation combines the two basic types and is referred to as allocation by m-complexes [41]. This means allocating balls in groups of $m (m < n)$. Each group of m balls are allocated without replacement, but the allocation of one group is independent of that of any other group.

Use $\mu_0(l,n,m)$ to denote the number of empty bins after l throws of m-complexes. Then, as given in [41],

$$\Pr[\mu_0(l,n,m) = 0] = \binom{n}{m}^{-l} \sum_{k=0}^{n} \binom{n}{k} (-1)^k \binom{n-k}{m}^{l}, \tag{5.10}$$

and

$$E[\mu_0(l,n,m)] = n(1 - \frac{m}{n})^l. \tag{5.11}$$

In Chapter 7, random allocation with m-complexes is used to model some generation scheduling strategies that are "cleverer" than random scheduling.

Chapter 6

Effects of Generation Size and Overlaps on the Throughput of Coding with Generations

In this chapter we analyze the effects of generation size and overlaps on the collection time (as defined in Chapter 5) of a receiver trying to retrieve a content file from the "cloud". "Random annex codes", which introduces overlaps between generations, are proposed, and analytical and simulation results show that with a proper choice of overlap size and structure, the collection time can be reduced using these codes.

The coding scheme studied is based on the coding with generations scheme first introduced in Chapter 2. Coded packets come from the n generations at random as opposed to the round-robin scheduling in Chapter 3, since we assume no coordination between the content holding servers in the "cloud", and these servers blindly transmit to the collector. We just think of the "cloud" as a super server which in each transmission "selects" one of the n generations at random. The probability of choosing generation G_i is ρ_j, $\sum_{j=1}^{n} \rho_j = 1$. Let $\boldsymbol{\rho} = (\rho_1, \rho_2, \ldots, \rho_n)$. Once G_j is chosen, a coded packet is formed as a random linear combination of the packets belonging to G_j over $\mathbb{F}_q$, as with the RL scheme defined in Chapter 3.

6.1 Collecting Coded Packets and Decoding

A generation G_i is not decodable until the number of linearly independent equations collected for G_i reaches the number of its file packets not yet resolved by decoding other generations. The connection between the number of coded packets collected and the linear

independence among these coded packets has to be established before we can predict the collection time of codes with generations using the collector's brotherhood model (Section 5.1).

Let $M(g,x)$ be the number of coded packets from a generation of size g adequate for collecting x linearly independent equations. Then $M(g,x)$ has expected value [42]

$$E[M(g,x)] = \sum_{j=0}^{x-1} \frac{1}{1-q^{j-g}}. \tag{6.1}$$

Approximating summation by integration, from (6.1) we get

$$\begin{aligned} E[M(g,x)] \lessapprox & \int_0^{x-1} \frac{1}{1-q^{y-g}} dy + \frac{1}{1-q^{x-1-g}} \\ =& x + \frac{q^{x-1-g}}{1-q^{x-1-g}} + \log_q \frac{1-q^{-g}}{1-q^{x-1-g}}. \end{aligned} \tag{6.2}$$

Let

$$\eta_g(x) = x + \frac{q^{x-1-g}}{1-q^{x-1-g}} + \log_q \frac{1-q^{-g}}{1-q^{x-1-g}}. \tag{6.3}$$

We can use $\eta_g(x)$ to estimate the number of coded packets needed from a certain generation to gather x linearly independent equations.

In addition, we have the following Claim 14 which upper bounds the tail probability of $M(g,g)$, the number of coded packets needed for a certain generation to gather enough linearly independent equations for decoding.

Claim 14. *There exist positive constants $\alpha_{q,g}$ and $\alpha_{2,\infty}$ such that, for $s \geq g$,*

$$\begin{aligned} \text{Prob}[M(g,g) > s] &= 1 - \prod_{k=0}^{g-1} (1-q^{k-s}) \\ &< 1 - \exp(-\alpha_{q,g} q^{-(s-g)}) < 1 - \exp(-\alpha_{2,\infty} q^{-(s-g)}). \end{aligned}$$

Also, since $1-\exp(-x) < x$ for $x > 0$,

$$\text{Prob}[M(g,g) > s] < \alpha_{q,g} q^{-(s-g)}. \tag{6.4}$$

Proof. Please refer to Appendix C.1. □

We will use Claim 14 and Theorem 15 in Section 6.2 to derive an upper bound to the expected overhead of coding over disjoint generations.

6.2 Coding Over Disjoint Generations

In this section, we study the performance of coding over disjoint generations. We derive both an upper bound and a lower bound for the expected collection time (as defined in Chapter 5. We also derive the variance of the collection time.

6.2.1 Expected Collection Time and Its Variance

Let M_i $(i = 1, 2, \ldots, n)$ be the number of collected coded packets from generation G_i when G_i first becomes decodable. Then M_i is at least g_i, has the same distribution as $M(g_i, g_i)$, the number of coded packets needed for a certain generation to gather enough linearly independent equations for decoding, as defined and studied in Section 6.1. M_i's are independent random variables. Let the collection time over a perfect channel be $W(\boldsymbol{\rho}, \mathbf{g})$, where $\mathbf{g} = (g_1, g_2, \ldots, g_n)$. Use $W_\epsilon(\boldsymbol{\rho}, \mathbf{g})$ to denote the collection time on a BEC(ϵ).

Let X_k $(k = 1, 2, \ldots)$ be i.i.d. geometric random variables with success rate $1 - \epsilon$. Therefore, $E[X_k] = \frac{1}{1-\epsilon}$ and $E[X_k^2] = \frac{1+\epsilon}{(1-\epsilon)^2}$. Then

$$W_\epsilon(\boldsymbol{\rho}, \mathbf{g}) = \sum_{i=1}^{W(\boldsymbol{\rho}, \mathbf{g})} X_i,$$

and therefore,

$$E[W_\epsilon(\boldsymbol{\rho}, \mathbf{g})] = \frac{1}{1-\epsilon} E[W(\boldsymbol{\rho}, \mathbf{g})], \tag{6.5}$$

$$Var[W_\epsilon(\boldsymbol{\rho}, \mathbf{g})] = \frac{1}{(1-\epsilon)^2} \left(Var[W(\boldsymbol{\rho}, \mathbf{g})] + \epsilon E[W^2(\boldsymbol{\rho}, \mathbf{g})] \right). \tag{6.6}$$

By definition, $E[W(\boldsymbol{\rho}, \mathbf{g})]$ is lower bounded by $E[T(\boldsymbol{\rho}, \mathbf{g})]$, the expected number of coded packets necessary for collecting at least g_i coded packets for each generation G_i, and $E[T(\boldsymbol{\rho}, \mathbf{g})]$ is as given in Corollary 9.

The following Theorem 15 gives the exact expression for the first and second moments of $W(\boldsymbol{\rho}, \mathbf{g})$, along with an upper bound for $E[W(\boldsymbol{\rho}, \mathbf{g})]$ considering the effect of finite finite field size q. Then, the expected value and the variance of $W_\epsilon(\boldsymbol{\rho}, \mathbf{g})$ can be derived from (6.5) and (6.6).

Theorem 15. *The expected number of coded packets needed for successful decoding of all N file packets*

$$E[W(\boldsymbol{\rho}, \mathbf{g})] = \int_0^\infty \left(1 - \prod_{i=1}^{n} \left(1 - e^{-\rho_i x} E_{M_i}\left[S_{M_i}(\rho_i x)\right]\right)\right) dx \tag{6.7}$$

$$< \int_0^\infty \left(1 - \prod_{i=1}^{n} \left(1 - e^{-\rho_i x}\left(S_{g_i}(\rho_i x) + \alpha_{q,g_i} q^{g_i} e^{\rho_i x/q} - \alpha_{q,g_i} q^{g_i} S_{g_i}(\rho_i x/q)\right)\right)\right) dx, \tag{6.8}$$

$$\begin{aligned} &E[W^2(\boldsymbol{\rho}, \mathbf{g})] \\ &= 2\int_0^\infty x\left(1 - \sum_{i=1}^{n} \rho_i \frac{1 - E_{M_i}[S_{M_i-1}(\rho_i x)] e^{-\rho_i x}}{1 - E_{M_i}\left[S_{M_i}(\rho_i x)\right] e^{-\rho_i x}} \prod_{j=1}^{n} \left(1 - E_{M_j}\left[S_{M_j}(\rho_j x)\right] e^{-\rho_j x}\right)\right) dx \\ &\quad + \int_0^\infty \left(1 - \prod_{i=1}^{n} \left(1 - e^{-\rho_i x} E_{M_i}[S_{M_i}(\rho_i x)]\right)\right) dx \end{aligned} \tag{6.9}$$

where $\alpha_{q,g_i} = -\sum_{k=0}^{g_i-1} \ln\left(1 - q^{k-g_i}\right)$, $i = 1, 2, \ldots, n$.

Proof. Please refer to Appendix C.3. □

In the case where generations are of equal size and scheduled uniformly at random, we can estimate the asymptotic lower bound for $E[W(\boldsymbol{\rho}, \mathbf{g})]$ by the asymptotics of $T_n(m)$ given in (5.8) and (5.9).

Figure 6.1(a) shows several estimates of $E[W(\boldsymbol{\rho}, \mathbf{g})]$, and Figure 6.1(b) shows the standard deviation of $W(\boldsymbol{\rho}, \mathbf{g})$ calculated from Theorem 15 and simulation results, when $\rho_i = \frac{1}{n}$ and $g_i = g$ for $i = 1, 2, \ldots, n$. The estimates are plotted versus the uniform generation size g for fixed $N = ng = 1000$.

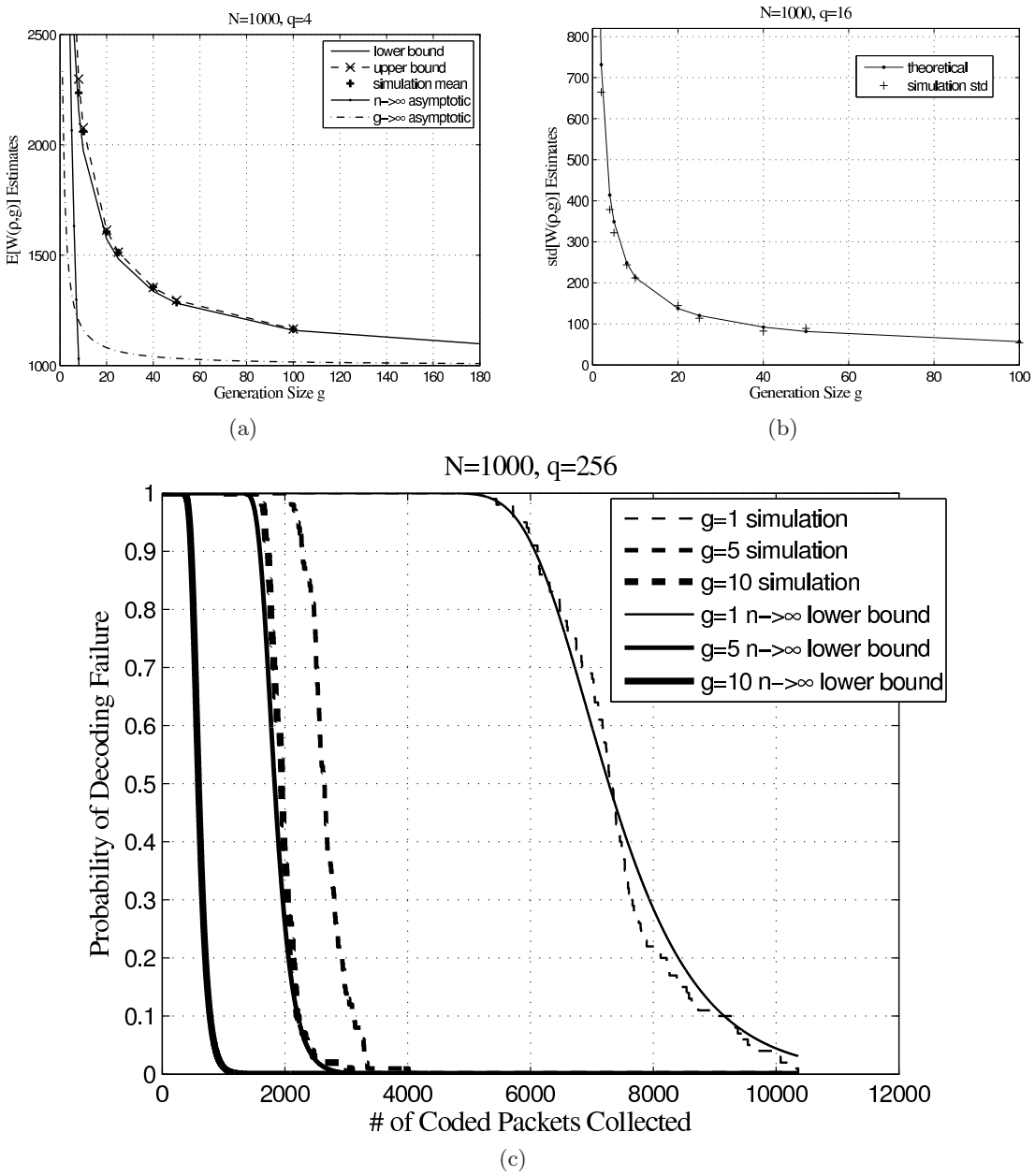


Figure 6.1: (a) Estimates of $E[W(\boldsymbol{\rho}, \mathbf{g})]$, the expected number of coded packets required for successful decoding when the total number of file packets is $N = 1000$, and both $\mathbf{g}$ and $\boldsymbol{\rho}$ are uniform. Estimates shown: lower bound $E[T(\boldsymbol{\rho}, \mathbf{g})]$; upper bound (6.8); mean of $W(\boldsymbol{\rho}, \mathbf{g})$ in simulation; $n \to \infty$ asymptotic (5.8); $m \gg 1$ asymptotics (5.9); (b) Estimates of the standard deviation of $W(\boldsymbol{\rho}, \mathbf{g})$; (c) Estimates of probability of decoding failure versus the number of coded packets collected: Theorem 16 along with simulation results.

For coding over disjoint generations and a fixed total number of file packets, both the expected value and the standard deviation of the collection time drop significantly as the generation size g grows to a relatively small value from the case where no coding is used ($g = 1$). Hence, throughput is improved by a moderate increase in the computational cost that scales quadratically with the generation size (see Section 2.2.4). On the other hand, we also observe that past a moderate generation size ($\sim 50-100$ coded packets for $N = 1000$), the decrease in collection time becomes slower by further increasing the encoding/decoding complexity. We therefore argue for a "sweet spot" generation size which characterizes the tradeoff between throughput and complexity.

6.2.2 Probability of Decoding Failure

In this subsection we assume uniform generation size and scheduling probability, i.e., $\rho_i = \frac{1}{n}$, $g_i = g$ for $i = 1, 2, \ldots, n$. For short, we denote $W(\boldsymbol{\rho}, \mathbf{g})$ as $W_n(g)$. From Theorem 13, we obtain the following lower bound to the probability of decoding failure as $n \to \infty$:

Theorem 16. *When $n \to \infty$, the probability of decoding failure when t coded packets have been collected is greater than*

$$1 - \exp\left[-\frac{1}{(g-1)!} n (\log n)^{g-1} \exp\left(-\frac{t}{n}\right)\right] + \mathcal{O}\left(\frac{\log\log n}{\log n}\right)$$

.

Proof. The probability of decoding failure after acquiring t coded packets equals $\text{Prob}[W_n(g) > t]$. Since $W_n(g) \geq T_n(g)$,

$$\begin{aligned} \text{Prob}[W_n(g) > t] &\geq \text{Prob}[T_n(g) > t] \\ &= 1 - \text{Prob}\left[Y_n(g) \leq \frac{t}{n} - \log n - (g-1)\log\log n\right]. \end{aligned}$$

The result in Theorem 16 follows directly from Theorem 13. □

Corollary 17. *When g is fixed and $n \to \infty$, in order to make the probability of decoding failure smaller than δ, the number of coded packets collected has to be at least $E[T_n(g)] -$*

$n \log\log \frac{1}{1-\delta}$. *If* $\delta = \frac{1}{N^c}$ *for some constant* c*, then the number of coded packets necessary for successful decoding has to be at least* $E[T_n(g)] + cn\log(ng)$.

Theorem 4.2 in [24] also gives the number of coded packets needed to have the probability of decoding failure below $\delta = \frac{1}{N^c}$, but under the assumption that $\ln(N/\delta) = \mathcal{O}(N/n) = \mathcal{O}(g)$. In comparison, Corollary 17 treats the case where g is constant.

Figure 6.1(c) shows the estimate of the probability of decoding failure versus T, the number of coded packets collected. As pointed out in Remark 1, for $m \geq 2$, the deviation of the CDF of $T_n(m)$ from the limit law for $n \to \infty$ depends on m and is on the order of $\mathcal{O}(\frac{\log\log n}{\log n})$ for $m \geq 2$, which is quite slow, partly explaining the deviation of the limit law curves from the simulation curves for $m = 5$ and $m = 10$ in Figure 6.1(c).

6.3 Coding Over Overlapping Generations

Even when generations are scheduled uniformly at random, there will be more coded packets accumulated in some of the generations than in others. The "slowest" generation is the bottleneck for file decoding. It is then advisable to design a mechanism that allows "faster" generations to help those lagging behind. In this section, we propose the *random annex code*, a new coding scheme in which generations share randomly chosen packets, as opposed to previously proposed "head-to-toe" overlapping scheme of [43].

We provide a heuristic analysis of the code throughput based on our results for the coupon collection model and an examination of the overlapping structure. Previous work on coding over overlapping generations, [44] and [43], lacks accurate performance analysis for information blocks of moderate finite lengths. On the other hand, the computational effort needed to carry out our analysis scales well with the length of information, and the performance predictions coincide with simulation data. In addition, we find that our random annex code outperforms the "head-to-toe" overlapping scheme of [43] over a unicast link.

In this section we conveniently assume that the coded packets are sent over a perfect channel. The scheme studied in this section is rateless. Each coded packet is statistically

the same as any other packet, and therefore the collection time over a link with packet loss rate ϵ is simply the collection time on a perfect channel scaled up by a factor of $\frac{1}{1-\epsilon}$. Please be reminded that the "unicast" model comes from an abstraction of the scenario of collecting coded packets from a "cloud" of servers.

6.3.1 Forming Overlapping Generations

We form n overlapping generations out of a file with N file packets in two steps as follows:

1. Partition the file set $\mathcal{F}$ of N packets into subsets $B_1, B_2, \ldots, B_n$, each containing h consecutive packets. These $n = N/h$ subsets are referred to as *base generations*. Thus, $B_i = \{\xi_{(i-1)h+1}, \xi_{(i-1)h+2}, \ldots, \xi_{ih}\}$ for $i = 1, 2, \ldots, n$. N is assumed to be a multiple of h for convenience. In practice, if N is not a multiple of h, set $n = \lceil N/h \rceil$ and assign the last $[N - (n-1)h]$ packets to the last (smaller) base generation.

2. To each base generation B_i, add a random *annex* R_i, consisting of l packets chosen uniformly at random (without replacement) from the $N - h = (n-1)h$ packets in $\mathcal{F} \backslash B_i$. The base generation together with its annex constitutes the *extended generation* $G_i = B_i \cup R_i$, the size of which is $g = h + l$. Throughout this paper, unless otherwise stated, the term "generation" will refer to "extended generation" whenever used alone for overlapping generations.

6.3.2 Generation Scheduling and Encoding within the Generation

The generation scheduling probabilities are chosen to be uniform, $\rho_1 = \rho_2 = \cdots = \rho_n = 1/n$. Encoding follows the RL scheme defined in Chapter 3.

6.3.3 Decoding

As with the decoding process of coding with disjoint generations, decoding starts with any generation G_j for which the receiver has collected g_j coded packets with linearly independent coding vectors. The file packets making up this generation are decoded by solving a

system of g_j linear equations in $\mathbb{F}_q$ formed by the coded packets on one side and the linear combinations of the file packets by the coding vectors on the other. Since generations are allowed to overlap, a decoded file packet may also participate in other generations, from the equations of which the file packet is then removed as an unknown variable. Consequently, in all the generations overlapping with the decoded generations, the number of unknown packets is reduced. As a result, some generations may become decodable even if no new coded packets are received from the source. Again, the newly decoded generations resolve some unknowns of the generations they overlap with, which in turn may become decodable and so on. We declare successful decoding when all N file packets have been decoded.

Note that whereas such a decoding scheme is optimal with disjoint generations, it is sub-optimal with overlapping generations.

6.3.4 Analyzing the Overlapping Structure

The following Claims 18 through 21 present combinatorial derivations of quantities concerning the frequency at which an arbitrary file packet is represented in different generations.

Claim 18. *For any packet in a base generation B_k, the probability that it belongs to annex R_r for some $r \in \{1, 2, \ldots, n\}\backslash\{k\}$ is*

$$\pi = \binom{N-h-1}{l-1} \Big/ \binom{N-h}{l} = \frac{l}{N-h} = \frac{l}{(n-1)h},$$

whereas the probability that it does not belong to R_r is $\bar{\pi} = 1 - \pi$.

Claim 19. *Let X be the random variable representing the number of generations an file packet participates in. Then, $X = 1 + Y$, where Y is* Binom$(n-1, \pi)$.

$$E[X] = 1 + (n-1)\pi = 1 + \frac{l}{h},$$

and

$$Var[X] = (n-1)\pi\bar{\pi}.$$

Claim 20. *In each generation of size $g = h + l$, the expected number of file packets not participating in any other generation is $h\bar{\pi}^{(n-1)} \approx he^{-l/h}$ for $n \gg 1$; the expected number of file packets participating in at least two generations is*

$$l + h[1 - \bar{\pi}^{(n-1)}] \approx l + h\left[1 - e^{-l/h}\right] < \min\{g, 2l\}$$

for $n \gg 1$ and $l > 0$.

Claim 21. *The probability that two generations overlap is $1 - \binom{N-2h}{l,l,N-2h-2l} / \binom{N-h}{l}^2$. The number of generations overlapping with any one generation G_i is then*

$$\text{Binom}\left(n-1, \left[1 - \binom{N-2h}{l,l,N-2h-2l} / \binom{N-h}{l}^2\right]\right).$$

The following Theorem 22 quantifies the expected amount of help a generation may receive from previously decoded generations in terms of common file packets. In the next subsection, we use Corollary 11 and Theorem 22 for a heuristic analysis of the expected throughput performance of the random annex code.

Theorem 22. *For any $I \subset \{1, 2, \ldots, n\}$ with $|I| = s$, and any $j \in \{1, 2, \ldots, n\} \backslash I$,*

$$\Omega(s) = E[|\left(\cup_{i \in I} G_i\right) \cap G_j|] = g \cdot [1 - \bar{\pi}^s] + sh \cdot \pi\bar{\pi}^s \tag{6.10}$$

where $|B|$ denotes the cardinality of set B. When $n \to \infty$, if $\frac{l}{h} \to \alpha$ and $\frac{s}{n} \to \beta$, and let $\omega(\beta) = \Omega(s)$, then $\omega(\beta) \to h\left[(1+\alpha)\left(1 - e^{-\alpha\beta}\right) + \alpha\beta e^{-\alpha\beta}\right]$.

Proof. Please refer to Appendix C.4. □

6.3.5 Expected Throughput Analysis: The Algorithm

Given the overlapping structure, we next describe an analysis of the expected number of coded packets a receiver needs to collect in order to decode all N file packets of $\mathcal{F}$ when they are encoded by the random annex code. We base our analysis on Theorem 22 above, Corollary 11 in Section 5.1, and also (6.3) in Section 6.1, and use the mean value for every quantity involved.

By the time when s $(s = 0, 1, \ldots, n-1)$ generations have been decoded, for any one of the remaining $(n-s)$ generations, on the average $\Omega(s)$ of its participating file packets have been decoded, or equivalently, $(g - \Omega(s))$ of them are not yet resolved. If for any one of these remaining generations the receiver has collected enough coded packets to decode its unresolved packets, that generation becomes the $(s+1)$th decoded; otherwise, if no such generation exists, decoding fails.

The quantity $\eta_g(x)$ defined in (6.3) in Section 6.1 estimates the number of coded packets from a generation of size g adequate for collecting x linearly independent equations. By extending the domain of $\eta_g(x)$ from integers to real numbers, we can estimate that the number of coded packets needed for the $(s+1)$th decoded generation should exceed $m'_s = \lceil \eta_g(g - \Omega(s)) \rceil$. Since in the random annex code, all generations are randomly scheduled with equal probability, for successful decoding, we would like to have at least m'_0 coded packets belonging to one of the generations, at least m'_1 belonging to another, and so on. Then Corollary 11 in Section 5.1 can be applied to estimate the total number of coded packets needed to achieve these minimum requirements for the numbers of coded packets.

The algorithm for our heuristic analysis is listed as follows:

1. Compute $\Omega(s-1)$ for $s = 1, \ldots, n$ using Theorem 22;
2. Compute $m'_s = \lceil \eta_g(g - \Omega(s-1)) \rceil$ for $s = 1, 2, \ldots, n$ using (6.3);
3. Map m'_s $(s = 1, 2, \ldots, n)$ into A values m_j $(j = 1, 2, \ldots, A)$ so that $m_j = m'_{k_{j-1}+1} = m'_{k_{j-1}+2} = \cdots = m'_{k_j}$, for $j = 1, 2, \ldots, A$, $k_0 = 0$ and $k_A = n$;
4. Evaluate (5.6) in Corollary 11 with the A, k_js, and m_js obtained in Step 3), as an estimate for the expected number of coded packets needed for successful decoding.

Remark 2. *The above Step 3) is viable because $\Omega(s)$ is nondecreasing in s, $\eta_g(x)$ is non-decreasing in x for fixed g, and thus m'_s is non-increasing in s.*

Although our analysis is heuristic, we will see in the next section that the estimate closely follows the simulated average performance curve of the random annex coding scheme.

6.3.6 Numerical Evaluation and Simulation Results

Throughput vs. Complexity in Fixed Number of Generations Schemes

Our goal here is to find out how the annex size l affects the collection time of the scheme with fixed base generation size h and the total number of file packets N (and consequently, the number of generations n). Note that the generation size $g = h + l$ affects the computational complexity of the scheme, and hence we are actually looking at the tradeoff between throughput and complexity.

Figure 6.2 and Figure 6.3 show both the analytical and simulation results when the total number N of file packets is 1000 and the base generation size h is 25. Figure 6.2 shows $h + l - \Omega(s)$ for $s = 0, 1, \ldots, n$ with different annex sizes. Recall that $\Omega(s)$ is the expected size of the overlap of the union of s generations with any one of the leftover $n - s$ generations. After the decoding of s generations, for any generation not yet decoded, the expected number of file packets that still need to be resolved is then $h + l - \Omega(s)$. We observe that the $h + l - \Omega(s)$ curves start from $h + l$ for $s = 0$ and gradually descends, ending somewhere above $h - l$, for $s = n - 1$.

Recall that we measure throughput by the collection time (Chapter 5). Figure 6.3(a) shows the expected performance of the random annex code, along with the performance of the head-to-toe overlapping code and the non-overlapping code ($l = 0$). Figure 6.3(b) shows the probability of decoding failure of these codes versus the number of coded packets collected.

- Our analysis for the expected collection time closely matches the simulation results.
- Figure 6.3(a) shows that by fixing the file size N and the base generation size h, the expected collection time decreases roughly linearly with increasing annex size l, up to $l = 12$ for the random annex scheme and up to $l = 8$ for the head-to-toe scheme. Meanwhile, the decoding cost per file packet is quadratic in $g = h + l$. Beyond the optimal annex size, throughput cannot be further increased by raising computational

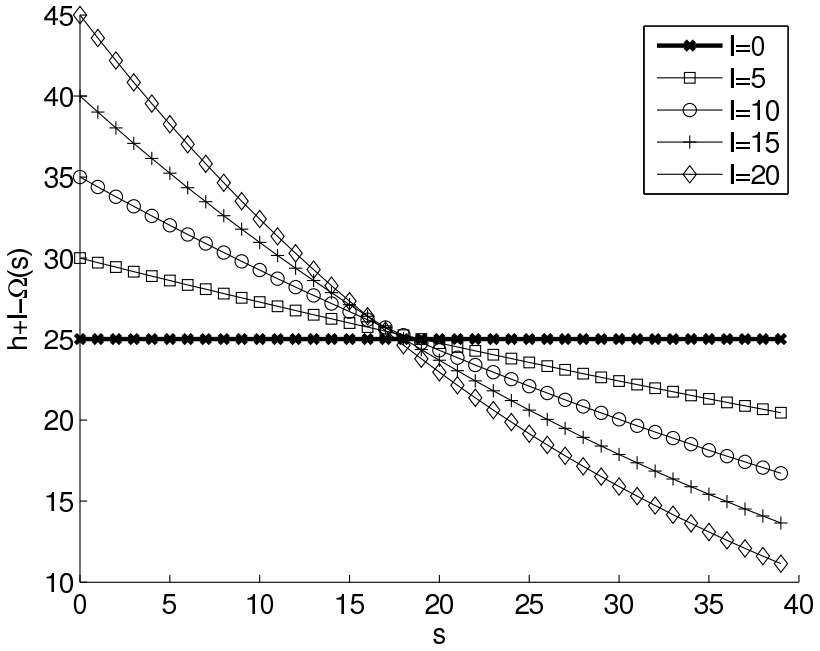


Figure 6.2: $N = 1000$, $h = 25$, $q = 256$: Difference between the generation size and the expected size of overlap with previously decoded generations ($h + l - \Omega(s)$).

cost.

- The random annex code outperforms head-to-toe overlapping at their respective optimal points. Both codes outperform the non-overlapping scheme.

- As more coded packets are collected, the probability of decoding failure of the random annex code converges to 0 faster than that of the head-to-toe and that of the non-overlapping scheme.

Overlaps provide a tradeoff between computational complexity and collection time.

Enhancing Throughput in Fixed-Generation-Size Schemes

Our goal here is to see if we can choose the annex size to optimize the throughput with negligible sacrifice in complexity. To this end, we fix the extended generation size $g = h + l$ and vary only the annex size l. Consequently, the computational complexity for coding does not increase when l increases. Actually, since some of the file packets in a generation of

size g could already be solved while decoding other generations, the remaining file packets in this generation can be solved in a system of linear equations of fewer than g unknowns, and as a result increasing l might decrease the decoding complexity.

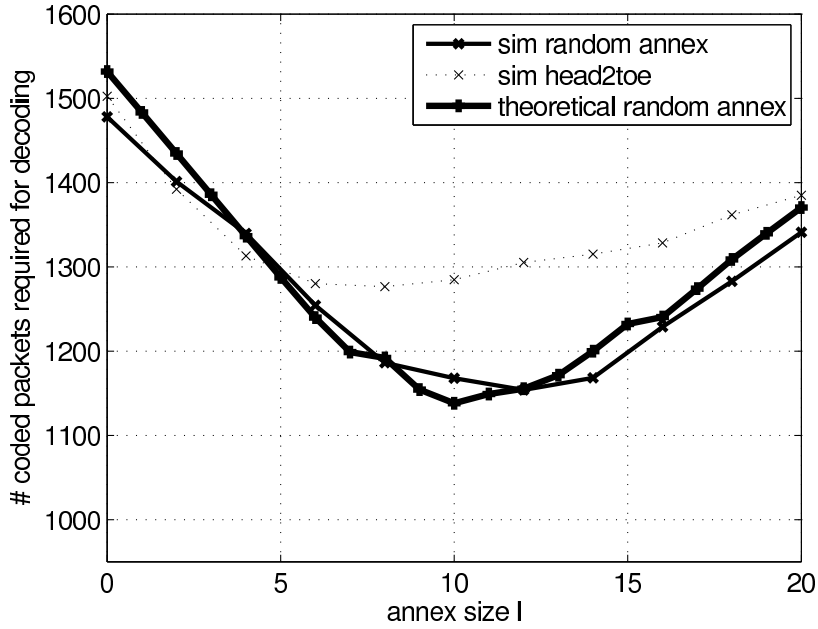
Figure 6.4 shows both the analytical and simulation results for the code performance when the total number N of file packets is fixed at 1000 and size g of extended generation fixed at 25.

- Again our analytical results agree with simulation results very well;
- It is interesting to observe that, without raising computational complexity, increasing annex size properly can still give non-negligible improvement to throughput;
- Figure 6.4(a) shows a roughly linear improvement of throughput with increasing l, up to $l = 10$ for the random annex scheme and up to $l = 6$ for the head-to-toe scheme. Increasing l beyond affects throughput adversely;
- The random annex code again outperforms head-to-toe overlapping at their optimal points. Both codes outperform the non-overlapping scheme;
- We again observe that the probability of decoding failure of the random annex code converges faster than those of the head-to-toe and the non-overlapping schemes.

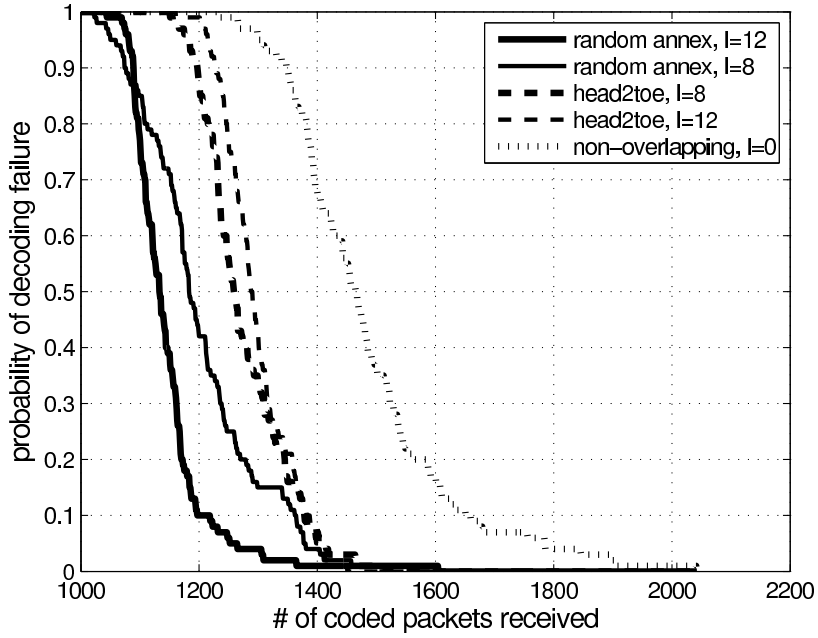
When the overlap size increases, we either have larger generations with unchanged number of generations, or a larger number of generations with unchanged generation size. In both cases the collection time would increase if we neglected the effect of overlaps during the decoding process. If we make use of the overlap in decoding, on the other hand, the larger the overlap size, the more help the generations can lend to each other in decoding and, hence, reducing the collection time. Two canceling effects result in a non-monotonic relationship between throughput and overlap size.

The effect of generation size on the throughput of random annex codes is further illustrated in Figure 6.5. Figure 6.5 plots the optimal expected collection time achievable by

random annex codes and the corresponding optimal annex size versus the generation size for $N = 1000$ and $q = 16$. The plotted values are calculated using the algorithm listed in Section 6.3.5. We can see from Figure 6.5 that with the random annex code and a generation size of 20, the expected throughput is better than what can be achieved with coding over disjoint generations and a generation size of 50. The reduction in computational complexity is considerable. Capturing the optimal overlap size in terms of other parameters of the code is our object of interest in the future.

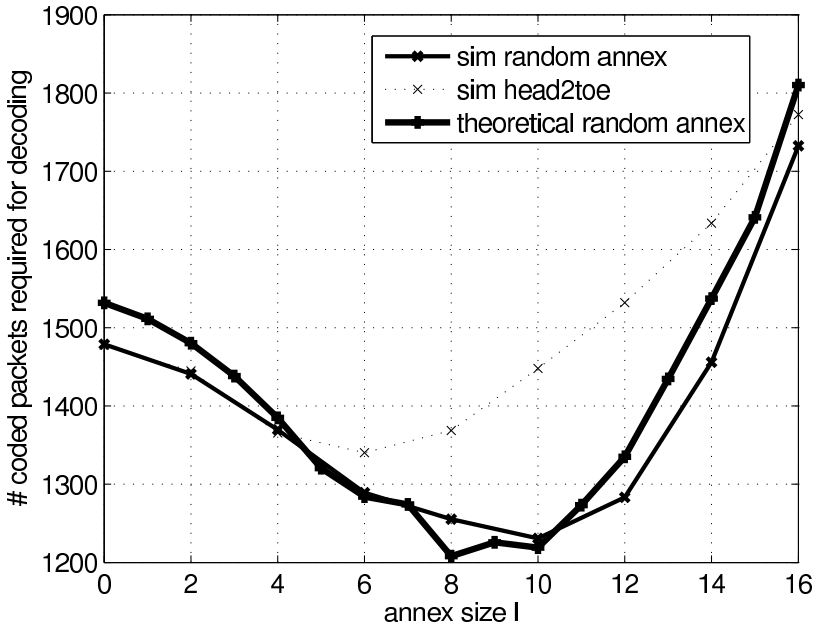


(a)

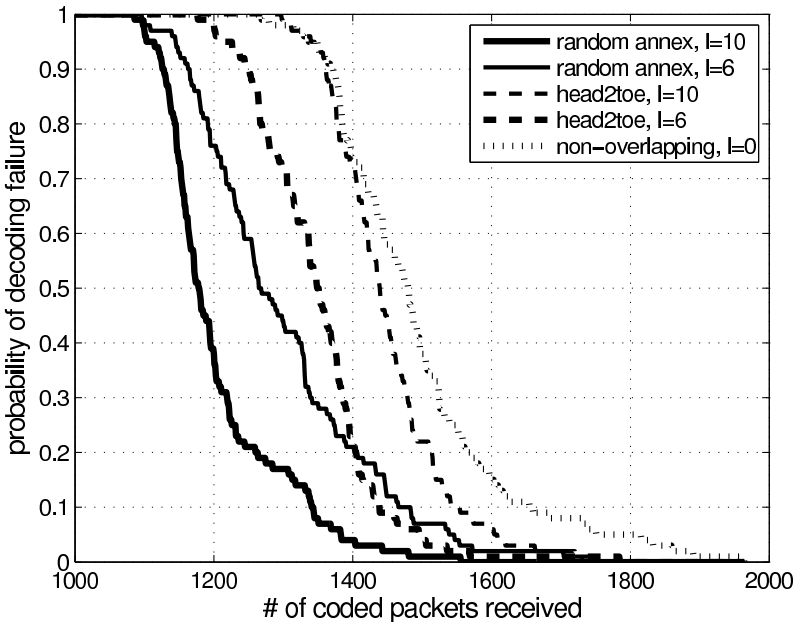


(b)

Figure 6.3: $N = 1000$, $h = 25$, $q = 256$: (a) Expected number of coded packets needed for successful decoding versus annex size l; (b) Probability of decoding failure



(a)



(b)

Figure 6.4: $N = 1000$, $g = h + l = 25$, $q = 256$: (a) Expected number of coded packets needed for successful decoding versus annex size l; (b) Probability of decoding failure

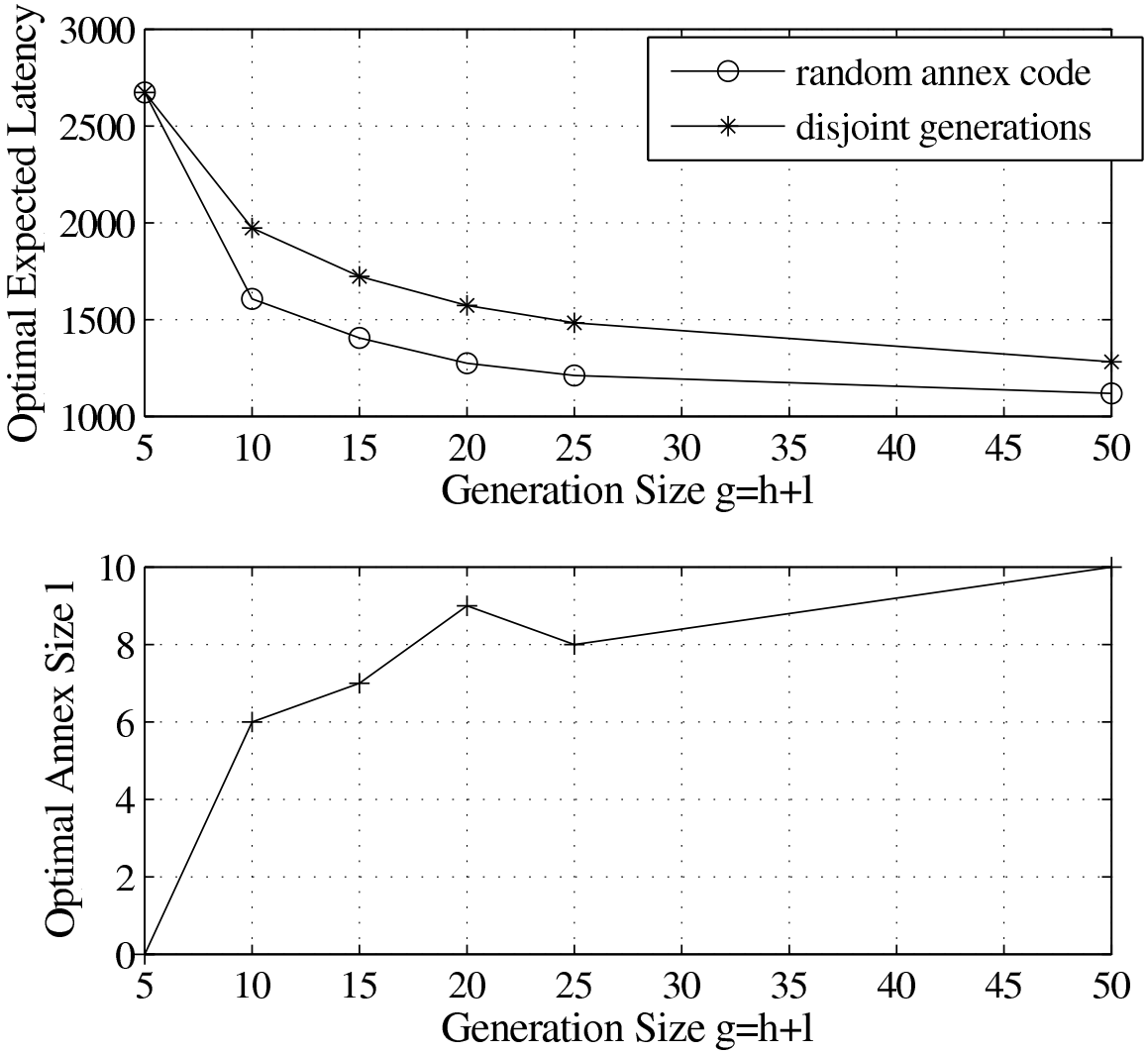


Figure 6.5: Optimal expected collection time and the optimal overlap size with random annex codes. $N = 1000$, $q = 16$

Chapter 7

Multiple Collectors: Union Power

7.1 Introduction

In BitTorrent-like [5] peer-to-peer(P2P) systems, content distribution involves dividing the content into smaller blocks at its source, and using swarming techniques to disseminate these blocks among peers. However, it has been observed from a number of experimental studies (e.g. [45]) that peer-churn(peers joining and leaving the system) and the availability of seeds(peers with the complete content) in the system can cause the "last-piece problem" and hamper downloading efficiency.

To improve downloading efficiency, Gkantsidis and Rodriguez proposed and experimented in [12] to incorporate the concept of network coding into P2P content distribution and to circulate random linear combinations of the content blocks. To help appreciate the benefit of using network coding in P2P systems, consider a simple network where a number of peers download simultaneously from a seeding node with a complete file of N blocks. Let us make a brief comparison of an uncoded system and a network coded system in terms of the growth rate of the union collection of the downloading peers with the growth of t, the number of file blocks or coded blocks downloaded by each peer from the source, and with the growth of l, the number of peers included in the union. In an uncoded system, the size of the union collection is defined as the number of distinct file blocks possessed by the set of peers in consideration, and in a network coded system, it is defined as the number of linearly independent coded blocks. If each peer collects uncoded file blocks from the seed uniformly at random, the expected size of the union collection of l peers will be

$N(1-(1-1/N)^{lt}) \approx N(1-e^{-lt/N})$ for large N. If each peer requests only the file blocks they do not have(sampling without replacement), the expected size of the union collection will be $N(1-(1-t/N)^{l}) \approx N(1-e^{-lt/N})$ for large N, which is not much better than collecting uniformly at random, if the total number of file blocks is large. However, if coding is used, with high probability all coded blocks are linearly independent, even across peers, and so the union collection size grows almost linearly both in t and in l. Without coordination between peers, only with coding can the downloading power be doubled when the number of peers doubles(given uploading bandwidth limit not yet achieved).

In this chapter, we want to study the effects of using coding with generations in P2P file distribution systems. We approach the problem by studying a simpler scenario: multiple collectors collecting packets from a super server, and study the union collection of the collectors. The generation scheduling (selection) strategies are not limited to being uniformly random.

Examining the union collection will help to answer the following questions:

1. How many coded blocks must be injected into the peer community so that the peers can jointly decode the file?

2. How many peers need a newly joining peer contact to acquire a complete copy of the file?

3. What fraction of the content remains in the system upon peer departures?

This work is our first attempt to study the peer union collection in P2P content distribution on the theoretical basis that could disclose the effect of coding and scheduling strategies on interesting quantities of the system.

We are particularly interested in the effects of the generation size and the generation selection strategies that take advantage of the information on individual collections. We compare three generation selection strategies: random selection, individual-min, and individual-max. The first strategy requires minimum accounting and serves as a reference point, the

second is a coded equivalent of the simplest uncoded strategy that each peer downloads its own missing blocks regardless of the collection of other peers, and the last is a greedy strategy that ensures linear growth of decodable content in each individual collection. We find that individual-min performs the best in terms of the rate of growth in union collection. With both random selection and individual-min, larger generation size delays initial decoding, but shortens the time to download the complete file. With individual-max, coding does not help with the growth of the number of decodable file blocks in the union collection, although the individual decodable collection can grow linearly since the beginning.

The rest of the chapter is organized as follows. In Section 7.2, we introduce our system model. In Section 7.3, we study the union collection of peer collectors when they use different generation selection strategies when requesting coded file blocks from other nodes. These strategies include random selection, individual-minimum and individual-maximum, and we use the balls-into-bins model for our analysis. We shall show numerical results and discuss the effects of the generation size and generation selection strategies on the growth of the union collection of peers. The last section concludes.

7.2 System Model

We consider l peer collectors collecting file $\mathcal{F}$ from the super server. Each collector communicates with the server via a perfect unicast link. The N file packets are divided into n disjoint generations of equal size g. For the sake of simplicity, N is assumed to be a multiple of g. Initially, none of the collectors has any content blocks.

Each peer collector contacts the super server and informs the server of the generation the collector is interested in. The server responds by sending a coded packet formed from the requested generation following the RL scheme (Section 3.2.2). The number of coded packets with linearly independent coefficients in the collection of the peer(s) is referred to as the *degrees of freedom(dofs)* collected by the peer(s). This terminology also applies to individual generations. The size of the finite field is assumed to be large enough such

that (with high probability) the dofs collected for a generation equals the number of coded packets collected from that generation, or the generation size g, whichever is smaller. Such an assumption is reasonable (see for example Section 6.1). Each node accumulates coded blocks from all generations and a generation becomes decodable when the dofs collected for that generation reaches the generation size g. Note that $g = 1$ means that there is no coding, and $g = N$ means coding over the entire content.

We are interested in the effects of generation size g and generation selection strategy on the union collection of peers. No coordination between the peers is assumed when requests to the super server are issued.

We assume that the communications in the network is synchronized. Let us suppose that all peers start at the same time(like an incoming "flash crowd"). Time is measured in discrete slots, and requests for coded packets are made at the beginning of a time slot, and responded by the end of the same slot. Each peer makes one request to the server in every time slot. All packets reach the intended receivers.

7.3 The Union Collection

In this section we investigate the union of the collection held at the peers in the system model described in Section 7.2. Use ${}^{l}T^{(k)}$ to denote the time elapsed when the number of decodable generations in the union collection of $l \geq 1$ peers reaches k, that is, when there are h dofs for each one of some arbitrary set of k generations. ${}^{l}T^{(n)}$ is the time needed to get a complete decodable copy of the file in the union collection of l peers. Use $W^{(k)}$ to denote the total number of coded blocks needed to collect k decodable generations. We will use a subscript to refer to the corresponding generation selection schedule, e.g., $W^{(k)}_{\text{rand}}$ refers to the random selection strategy. In a balls-into-bins perspective, a decodable generation corresponds to a bin with at least g balls in it.

7.3.1 Random Selection

With the random selection strategy, a receiver node requests a generation selected uniformly at random among all n generations. Note that this implies that a peer may select an already decodable generation. Nevertheless, this simple strategy offers a reference point and is easier to analyze. Random selection can be modeled as random allocation of balls into n bins with replacement. The problem of collecting k decodable generations is then readily modeled by the collector's brotherhood problem [13, 14].

The expected total number of coded blocks (balls) needed to collect k decodable generations (bins having g or more balls), $E[W_{\text{rand}}^{(k)}]$, can be inferred from Corollary 11 in Chapter 5, which treats an extension of the collector's brotherhood problem:

$$E[W_{\text{rand}}^{(k)}] = n \int_0^\infty \left\{ \sum_{i=0}^{k-1} \binom{n}{i} S_g^{n-i}(x) \left[e^x - S_g(x)\right]^i \right\} e^{-nx} dx, \tag{7.1}$$

where

$$\begin{aligned} S_g(x) =& 1 + \frac{x}{1!} + \frac{x^2}{2!} + \cdots + \frac{x^{g-1}}{(g-1)!} \quad (g \geq 1) \\ S_\infty(x) =& \exp(x) \text{ and } S_0(x) = 0. \end{aligned}$$

Since the time required to have k decodable generations in the union collection of $l \geq 1$ peers is ${}^lT_{\text{rand}}^{(k)} = \lceil W_{\text{rand}}^{(k)}/l \rceil$, we have its expectation sandwiched as

$$\frac{1}{l} E[W_{\text{rand}}^{(k)}] \leq E[{}^lT_{\text{rand}}^{(k)}] < 1 + \frac{1}{l} E[W_{\text{rand}}^{(k)}], \tag{7.2}$$

Plotted in Figure 7.1 is the expected time needed for l peers to acquire in their union collection certain decodable fractions of the file, for a few exemplary generation sizes. With smaller generation sizes, the decodable fraction in the union collection grows fast at the beginning, but it takes longer to collect the whole file, while with larger generation sizes, the decodable fraction in the union collection grows at a more steady rate and a complete file copy could be downloaded in a shorter time. We also note that, when downloading a file of 50 20-packet generation, for as few as $l = 2$ peers, the expected time to acquire the

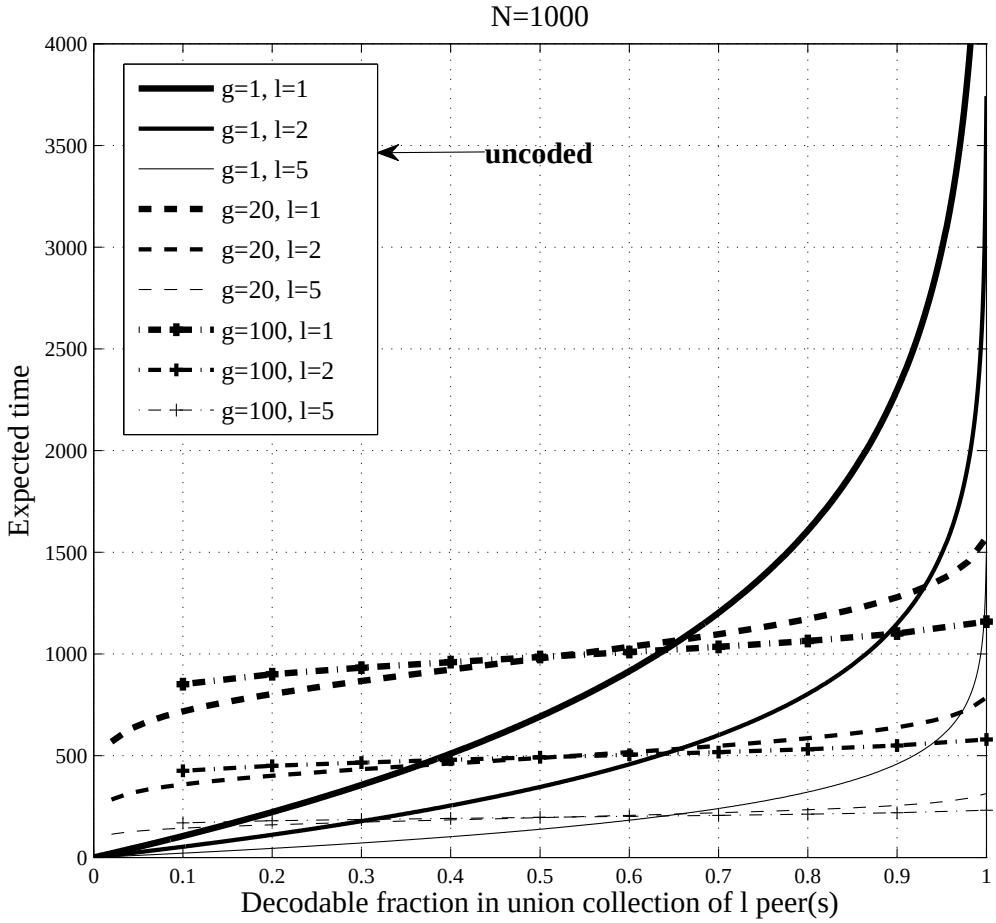


Figure 7.1: Expected time needed to collect decodable fraction of the file in the union collection of l peers. Generation selection strategy: uniformly at random. File Size $N = 1000$.

whole file in the union collection is much shorter than the expected time to collect enough dofs to decode the first generation for a single peer. This implies that our results hold well even if a peer does exclude decodable generations when requesting coded blocks from the seed.

The readers are also referred to Chapter 6 for the effects of generation size on the expected time needed to obtain a complete decodable copy of the file, when generations are selected uniformly at random. Interesting asymptotic results for large n, the number of generations have also been presented there.

7.3.2 Individual-Min

Here each peer requests the generation with the smallest dofs in their respective individual collections. A uniformly random draw is used to break the tie among multiple generations with minimum dofs. For $g = 1$, the uncoded case, this simply means randomly requesting one of the *missing* blocks.

Given our assumption that all peers start collection at the same time and proceed at the same pace, each peer collects the same number of coded packets after a certain time. Meanwhile, under the individual-min strategy, at each collector, the numbers of dofs collected for different generations, that is, the numbers of balls in different bins, differ by at most 1, since the bins are filled "level-by-level". With each collector, by the end of time slot t ($t < N = ng$), $a(t) = t - n\lfloor t/n \rfloor$ of the n bins will each have $\lfloor t/n \rfloor + 1$ balls, and the remaining $n - a(t)$ bins will each have $\lfloor t/n \rfloor$ balls. The $a(t)$ bins with extra balls are distributed among the n bins as if they were chosen uniformly at random from the n generations without replacement. After the bins are merged among the l peers by the bin (generation) indices, there are at least $l\lfloor t/n \rfloor$ balls in each bin. Thus, in the union collection of any l peers, there will be (with high probability) at least a total of $\min\{l\lfloor t/n \rfloor, g\}$ dofs collected for each generation. Hence, we have

$$ {}^{l}T^{(n)} \leq n\lceil \frac{g}{l} \rceil. \tag{7.3} $$

On the other hand,

$$ {}^{l}T^{(n)} \geq n\lfloor \frac{g}{l} \rfloor, \tag{7.4} $$

because otherwise, the total number of dofs collected by all l peers is

$$ l \cdot {}^{l}T^{(n)} < l \cdot n \cdot \lfloor \frac{g}{l} \rfloor \leq l \cdot n \cdot g/l = N, $$

making it impossible to get a decodable copy of all N file packets in the union collection.

Let $y = g - l\lfloor g/l \rfloor$. We have $0 \leq y \leq l - 1$.

Claim 23. *If g is a multiple of l, ${}^{l}T^{(n)}_{\text{indv\_min}} = ng/l = N/l$.*

If $y \geq 1$, $\lfloor g/l \rfloor = \lceil g/l \rceil - 1$. We further have

$$ {}^{l}T^{(n)} > n\lfloor \frac{g}{l} \rfloor, \tag{7.5}$$

since otherwise, the total number of coded packets collected by all l peers is

$$ l \cdot {}^{l}T^{(n)} \leq l \cdot n \cdot \lfloor \frac{g}{l} \rfloor < l \cdot n \cdot g/l = N, $$

impossible to get a fully decoded copy. By (7.3) and (7.5), we have ${}^{l}T^{(n)} \in \{n\lfloor g/l \rfloor + 1, n\lfloor g/l \rfloor + 2, \ldots, n\lceil g/l \rceil\}$. $\lceil {}^{l}T^{(n)}/n \rceil = \lceil g/l \rceil = \lfloor g/l \rfloor + 1$. At time ${}^{l}T^{(n)}$, if we remove $\lfloor g/l \rfloor$ balls from each bin at each peer, there will be $b({}^{l}T^{(n)}) =^{l} T^{(n)} - n\lfloor g/l \rfloor$ bins with one ball in each of them, and the remaining $n - b({}^{l}T^{(n)})$ bins will be empty. Then merge the bins among the l peers by the bin(generation) indices. Given that by time ${}^{l}T^{(n)}$ there should be a fully decodable copy in the union collection, the distribution of the $l \cdot b({}^{l}T^{(n)})$ balls after merging should ensure that there are at least y balls in each bin. Hence, ${}^{l}T^{(n)}$ is the smallest t required to accumulate at least y balls in each of the n bins after allocating balls in l $b(t)$-complexes (see Section 5.2 for the definition of allocation in complexes).

Claim 24. *If l divides $g - 1$,*

$$ E[{}^{l}T^{(n)}_{\text{indv\_min}}] = n\frac{g-1}{l} + \sum_{m=1}^{n} \binom{n}{m} (-1)^{m-1} \sum_{\beta=1}^{n-m} \binom{n}{\beta}^{-l} \binom{n-m}{\beta}^{l} + 1. $$

Proof. For simplicity, in this proof we use T to represent ${}^{l}T^{(n)}$.

If l divides $g-1$, $y = g - l\lfloor g/l \rfloor = 1$, the problem becomes filling all n bins by throwing $b(T)$-complexes of balls l times. The possible values of T are limited to $\{n\frac{g-1}{l} + 1, n\frac{g-1}{l} + 2, \ldots, n\frac{g-1}{l} + n\}$. $b(T) = T - n(g-1)/l$, and takes values in $\{1, 2, \ldots, n\}$.

The number of empty bins after l throws of β-complexes ($\beta \in \{1, 2, \ldots, n\}$) is $\mu_0(l, n, \beta)$,

as defined in Section 5.2. Then, $\Pr[b(T) > \beta] = \Pr[\mu_0(l, n, \beta) > 0]$, and thus by (5.10),

$$\begin{aligned} E[b(T)] &= \sum_{\beta=0}^{n} \Pr[b(T) > \beta] \\ &= 1 + \sum_{\beta=1}^{n-1} \left(1 - \binom{n}{\beta}^{-l} \sum_{m=0}^{n} \binom{n}{m} (-1)^m \binom{n-m}{\beta}^{l} \right) \\ &= 1 + \sum_{\beta=1}^{n-1} \binom{n}{\beta}^{-l} \sum_{m=1}^{n} \binom{n}{m} (-1)^{m-1} \binom{n-m}{\beta}^{l} \\ &= 1 + \sum_{m=1}^{n} \binom{n}{m} (-1)^{m-1} \sum_{\beta=1}^{n-m} \binom{n}{\beta}^{-l} \binom{n-m}{\beta}^{l} \end{aligned} \tag{7.6}$$

Then,

$$E[T] = n\frac{g-1}{l} + E[b(T)], \tag{7.7}$$

where l has to be a divisor of $g - 1$. □

Some example values of $E[T_{\text{indv\_min}}]$ are shown in Table 7.1, when $N = 1000$. The growth of the union collection with the number of peers l is almost halved when the generation size $g = 25$. Also, increasing g from 25 to 40 slightly reduces $E[T_{\text{indv\_min}}]$. It seems that for $N = 1000$, $g = 25$ is good enough to effectively increase downloading power by adding peers, and this increase does not require exchange of information of respective collections among peers.

Table 7.1: $E[T_{\text{indv\_min}}]$ and $E[T_{\text{rand}}]$ for $N = 1000$, two generation sizes $g = 25$ and $g = 40$, with given numbers of peers l (*upperbounds)

	l	1	2	3	6	12	13
$E[T_{\text{indv\_min}}]$	$g = 25, n = 40$	1000	515.4	350.4	180.7	92.5	80*
$E[T_{\text{rand}}]$	$g = 25, n = 40$	1483.2	741.6	494.4	247.2	123.6	114.1
$E[T_{\text{indv\_min}}]$	$g = 40, n = 25$	1000	500	343.2	175*	94.5*	81.9
$E[T_{\text{rand}}]$	$g = 40, n = 25$	1336.2	668.1	445.4	222.7	111.3	102.8

The expected number of decodable file packets in the union collection of l peers, on the other hand, is given in the following Claim 25.

Claim 25. *If l divides $g - 1$, at time t, the expected number of decodable file packets in the*

union collection of l peers is

$$N\left(1-\left(1-\left(\frac{t}{n}-\frac{g-1}{l}\right)\right)^l\right)$$

for $t > n(g-1)/l$, and is 0 when $t \le n(g-1)/l$.

Proof. The number of decodable generations equals $n-\mu_0(l,n,b(t))$. By (5.11), the expected number of decodable file packets in the union collection is

$$\begin{aligned} g(n-E[\mu_0(l,n,b(t))]) &= N\left(1-\left(1-\frac{b(t)}{n}\right)^l\right) \\ &= N\left(1-\left(1-\left(\frac{t}{n}-\frac{g-1}{l}\right)\right)^l\right), \end{aligned} \tag{7.8}$$

□

In Figure 7.2 we compare the growth of decodable fraction in the union collection for $N = 1000$, $g = 1, 10, 100$ and $l = 3, 9$. Note that when $g = 1$ (the uncoded case), all $l \in \mathbb{N}$ divides $g-1 = 0$, and hence both Claim 24 and Claim 25 apply. In particular, the expected number of decodable file packets at time $t \le N$ is

$$N - E[\mu_0(l,N,t)] = N[1-(1-t/N)^l]. \tag{7.9}$$

Here we observe a phenomenon similar to what we have seen in the random selection scheme: with larger generations, it takes longer to get the initial decodable fraction, but it is easier to collect the last fraction of the file. Also, using the individual-min selection strategy, the time needed to download the file is greatly reduced compared to the random selection strategy.

For $y \ge 1$, We also give an upper bound of the expected time T to get a full file copy.

Claim 26. *If $y = g - l\lfloor g/l \rfloor \ge 1$,*

$$E[{}^lT^{(n)}_{\text{indv\_min}}] < n\frac{g-y}{l} + \min\{n, \frac{n}{l}\int_0^\infty \{1-[1-S_y(x)e^{-x}]^n\}dx + 1\}. \tag{7.10}$$

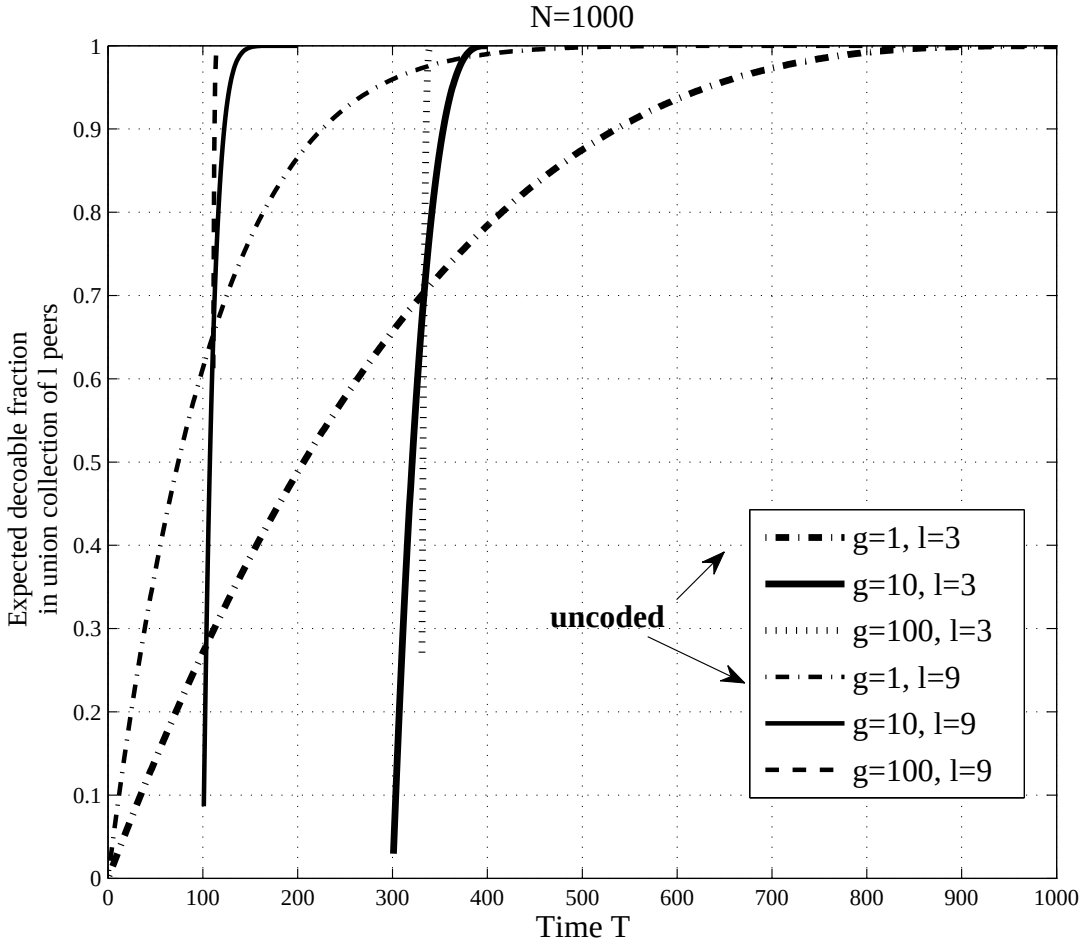


Figure 7.2: Expected decodable fractions of the file in the union collection of l peers versus downloading time. Generation selection strategy: individual-min. File Size $N = 1000$.

Proof. Again in this proof, we abbreviate ${}^{l}T^{(n)}$ to T. First of all, by time $n(g-y)/l + n = n\lceil g/l \rceil$, each peer collects $\lceil g/l \rceil$ dofs for each generation, and therefore in the union collection, $l\lceil g/l \rceil > g$ dofs are collected for each generation, and a fully decodable copy exists. Hence, $T \leq n(g-y)/l+n$. Since here we assume that g is not a multiple of l, smaller values of T is possible. Therefore, $E[T] < n(g-y)/l + n$.

On the other hand, at each peer, consider instead allocating the last $b(T) = T-n(g-y)/l$ balls with replacement (see Chapter 5), and then throw away the extra balls in the bins with more than one ball (counting the last $b(T)$ balls only). Merge the collectors' bins by the indices. Then, the number of balls in each bin will not exceed what is achieved with

allocation without replacement. We need to get at least y balls in each bin. By (7.1) and (7.2), we have

$$E[b(T)] < \frac{n}{l}\int_0^\infty \{1-[1-S_y(x)e^{-x}]^n\}dx + 1.$$

□

We see that

Finally, for the special case of $g = 1$ (the uncoded case), we derive the expected value of the number of peers needed for the union collection to include one complete copy of the file.

Claim 27. *In the uncoded case ($g = 1$), the expected value of the number of peers L needed for the union collection to include one complete copy of the file when each collector has been collecting for a duration of t is*

$$E[L] = \sum_{m=1}^{N} \binom{N}{m}(-1)^{m-1}\frac{1}{1-\binom{N-m}{t}/\binom{N}{t}}. \tag{7.11}$$

Proof.

$$\begin{aligned}
E[L] &= \sum_{l\geq 0}\Pr[L > l] = \sum_{l\geq 0}\Pr[\mu_0(l,N,t) > 0] \\
&= \sum_{l\geq 0}\binom{N}{t}^{-l}\sum_{m=1}^{N}\binom{N}{m}(-1)^{m-1}\binom{N-m}{t}^{l} \\
&= \sum_{m=1}^{N}\binom{N}{m}(-1)^{m-1}\sum_{l\geq 0}\binom{N}{t}^{-l}\binom{N-m}{t}^{l} \\
&= \sum_{m=1}^{N}\binom{N}{m}(-1)^{m-1}\frac{1}{1-\binom{N-m}{t}/\binom{N}{t}}.
\end{aligned} \tag{7.12}$$

□

It can be estimated from (7.9) that $E[L]$ decreases with t approximately as $\mathcal{O}(-1/\log(1-t/N))$.

Imagine a new collector joins after time t, and unfortunately the super server has disconnected from the system. Then it would need to contact about $E[L]$ peers to be able to obtain the complete file.

7.3.3 Individual-Max

Here, in each time slot a peer requests a coded packet from any one of the non-decodable generations with the largest dofs collected. This is a greedy strategy in which each peer ensures linear growth of decodable content in its own personal collection. With this strategy, at each peer, the first g balls land into one of the n bins, the next g balls land into one of remaining $n-1$ empty bins, and so on. After time t, each peer will have a random set of $\lfloor t/g \rfloor$ decodable generations, one generation with $(t - g\lfloor t/g \rfloor)$ dofs, and the rest of the generations with no dofs collected yet. When t is a multiple of g, each bin is either empty or filled with g balls. The number of decodable generations in the union collection of l peers is the same as the number of bins with at least one ball, when balls are shot into the bins as (t/g)-complexes, independently and l times. In the light of this interpretation, by time t, the average number of decodable generations in the union collection of l peers is given by

$$n - E[\mu_0(l, n, t/g)] = n\{1 - [1 - t/(ng)]^l\}.$$

The number of decodable packets is then $N[1-(1-t/N)^l]$, exactly the same as that of the uncoded case (7.9). This means coding does not provide much benefit to the system, if any.

The individual maximum a strategy ensures that individuals can decode coded blocks as soon as possible, but it is adverse to the growth of the union collection of a number of peers, compromising the power of sharing among peers.

7.4 Conclusions and Future Work

We analyzed the growth of union collection in a peer-to-peer system that employs network coding with generations using different versions of the classical balls-into-bins model. We study the effects of the generation size and the generation selection strategies that take advantage of the information on individual collections. Three generation selection strategies are compared: random selection, individual-min, and individual-max. The individual-min strategy is found to perform the best in terms of the rate of growth in the union collection.

With both random selection and individual-min, larger generation size delays initial decoding, but shortens the time to download the complete file. With individual-max, however, coding does not help with the growth of the number of decodable file blocks in the union collection, although the individual decodable collection can grow linearly since the beginning. When properly employed, coding reduces the need to optimize system performance by co-ordination between peers.

The study of the union collection is helpful to the investigation of both the throughput and the peer-dynamics of a peer-to-peer system. We believe our work has already shed some light on the effects of using network coding with generations in peer-to-peer file distribution. The results should also be transplantable to distributed caching systems. In the near future, we expect to extend our analysis to more complicated topologies and generation scheduling with balancing schemes such as local rarest first. We would also like to compare the analytical results to experimental results on large-scale systems.

Appendix A

Proof of Claim 2 in Chapter 3

$$
\begin{aligned}
&\sum_{l=0}^{g-1}\binom{g}{l}(1-\epsilon)^l\epsilon^{g-l}p^{\mathrm{RL}}_{m-g,g-l,\epsilon}\\
&=\sum_{l=0}^{g-1}\binom{g}{l}(1-\epsilon)^l\epsilon^{g-l}\sum_{j=g-l}^{m-g}\binom{m-g}{j}(1-\epsilon)^j\epsilon^{m-g-j}\prod_{s=0}^{g-l-1}(1-q^{s-j})\\
&=\sum_{l=0}^{g-1}\binom{g}{l}\sum_{j=g-l}^{m-g}\binom{m-g}{j}(1-\epsilon)^{j+l}\epsilon^{m-l-j}\prod_{s=0}^{g-l-1}(1-q^{s-j})\\
&=\sum_{l=0}^{g-1}\binom{g}{l}\sum_{j=g}^{m-g+l}\binom{m-g}{j-l}(1-\epsilon)^{j}\epsilon^{m-j}\prod_{s=0}^{g-l-1}(1-q^{s-j+l})\\
&=\sum_{j=g}^{m-1}\sum_{l=0}^{g-1}\binom{g}{l}\binom{m-g}{j-l}(1-\epsilon)^{j}\epsilon^{m-j}\prod_{s=0}^{g-l-1}(1-q^{s-j+l})\\
&\gtrsim\sum_{j=g}^{m-1}\sum_{l=0}^{g-1}\binom{g}{l}\binom{m-g}{j-l}(1-\epsilon)^{j}\epsilon^{m-j}(1-\frac{1}{q-1}q^{g-j}) \qquad (*)
\end{aligned}
$$

(*) follows from (3.6). Applying Vandermonde's identity $\sum_{l=0}^{g}\binom{g}{l}\binom{m-g}{j-l}=\binom{m}{j}$ to (*), we

have

$$
\begin{aligned}
(*) &= \sum_{j=g}^{m} \left(\binom{m}{j} - \binom{m-g}{j-g} \right) (1-\epsilon)^j \epsilon^{m-j} (1 - \frac{1}{q-1} q^{g-j}) \\
&= \sum_{j=g}^{m} \binom{m}{j} (1-\epsilon)^j \epsilon^{m-j} - \sum_{j=g}^{m} \binom{m-g}{j-g} (1-\epsilon)^j \epsilon^{m-j} \\
&\quad - \frac{1}{q-1} \sum_{j=g}^{m} \binom{m}{j} (1-\epsilon)^j \epsilon^{m-j} q^{g-j} + \frac{1}{q-1} \sum_{j=g}^{m} \binom{m-g}{j-g} (1-\epsilon)^j \epsilon^{m-j} q^{g-j} \\
&= \sum_{j=g}^{m} \binom{m}{j} (1-\epsilon)^j \epsilon^{m-j} - (1-\epsilon)^g \sum_{j=0}^{m-g} \binom{m-g}{j} (1-\epsilon)^j \epsilon^{m-g-j} \\
&\quad - \frac{1}{q-1} \sum_{j=g}^{m} \binom{m}{j} (1-\epsilon)^j \epsilon^{m-j} q^{g-j} + \frac{(1-\epsilon)^g}{q-1} \sum_{j=0}^{m-g} \binom{m-g}{j} (\frac{1-\epsilon}{q})^j \epsilon^{m-g-j} \\
&= \sum_{j=g}^{m} \binom{m}{j} (1-\epsilon)^j \epsilon^{m-j} - \frac{1}{q-1} \sum_{j=g}^{m} \binom{m}{j} (1-\epsilon)^j \epsilon^{m-j} q^{g-j} - (1-\epsilon)^g \\
&\quad + \frac{(1-\epsilon)^g}{q-1} (\frac{1-\epsilon}{q} + \epsilon)^{m-g}.
\end{aligned}
$$

Appendix B

Proof of Claim 5 in Chapter 4

We give two independent arguments to validate the claim.

After the systematic phase, each content packet successfully reaches user i independently at probability $1-\epsilon_i$, and an average of $(1-\epsilon_i)N$ packets reaches the user. If $z_i \leq 1-\epsilon_i$, the demand of user i is considered fulfilled. We consider the case where $z_i > 1-\epsilon_i$.

In the first argument, we find out the non-systematic equivalent degree distribution $\bar{P}(x)$. To do that we only need to find out the (normalized) number of degree-1 packets(containing content packets selected uniformly at random with replacement) required to be transmitted in order for user i to recover the $(1-\epsilon_i)N$ distinct content packets received in the systematic phase. This number, $-\frac{\ln \epsilon_i}{1-\epsilon_i}$, can be obtained by setting $P(x) = 1 \cdot x$ (all-one degree distribution) in (4.6). A coupon collector's argument brings to the same conclusion [46, Ch. 2] (see also [42]): the expected number of samplings required to collect zN distinct coupons is

$$N\Big(\frac{1}{N} + \frac{1}{N-1} + \cdots + \frac{1}{N-zN+1}\Big) \gtrapprox N \ln \frac{N}{N-zN+1} = -N \ln\Big(1 - \frac{zN-1}{N}\Big). \quad \text{(B.1)}$$

Divide (B.1) by $N(1-\epsilon_i)$, take $N \to \infty$ and let $z = 1-\epsilon_i$, we find the same result as found from (4.6).

Hence,

$$\bar{p}_1 = -\frac{\ln \epsilon_i}{1-\epsilon_i} + p_1$$

and

$$\bar{P}(x) = -\frac{\ln \epsilon_i}{1-\epsilon_i} x + P(x).$$

In (4.5), replace $P(x)$ by $\bar{P}(x)$ and t_i by $(t_i - 1)$ to subtract the time spent in the systematic phase, and we get (4.11).

Alternatively, one can consider the packets received in the systematic phase as side information to the decoder, such as in [47, 48]. These packets are removed from the subsequent coded packets, and we need to decode another $(z_i - (1 - \epsilon_i))N$ packets from the remaining $\epsilon_i N$ packets. Let d and $\hat{d}$ be the random variables representing the degree of a randomly generated coded packet and the degree of the coded packet after the removal of the packets received in the systematic phase. Thus, $\hat{d} = \sum_{j=1}^{d} X_j$ where $X_j (j = 1, 2, \ldots, D)$ are i.i.d Bernoulli(ϵ_i) random variables indicating if the jth content packet participating in the coded packet has *not* been received in the systematic phase and remains in the coded packet. Thus, the moment generating function $\hat{D}$ is

$$\hat{P}(x) = P(Q(x)) = P(1 - \epsilon_i + \epsilon_i x),$$

where

$$Q(x) = 1 - \epsilon_i + \epsilon_i x$$

is the moment generating function of the i.i.d. X_js. In (4.5), replace $P(x)$ by $\hat{P}(x)$, z_i by

$$\hat{z}_i = \frac{(z_i - (1 - \epsilon_i))N}{\epsilon_i N} = \frac{z_i - (1 - \epsilon_i)}{\epsilon_i},$$

and t_i by

$$\hat{t}_i = \frac{(t_i - 1)N}{\epsilon_i N} = \frac{t_i - 1}{\epsilon_i},$$

we have

$$(1 - \epsilon_i)\frac{t_i - 1}{\epsilon_i} P'(Q(x))Q'(x) + \ln(1 - x) > 0, \quad \forall x \in \left[0, \frac{z_i - (1 - \epsilon_i)}{\epsilon_i}\right). \tag{B.2}$$

Let $y = Q(x) = 1 - \epsilon_i + \epsilon_i x$, we get

$$(1 - \epsilon_i)\epsilon_i \frac{t_i - 1}{\epsilon_i} P'(y) + \ln\left(1 - \frac{y - 1 + \epsilon_i}{\epsilon_i}\right) > 0, \quad \forall y \in \left[1 - \epsilon_i, 1 - \epsilon_i + \epsilon_i \frac{z_i - (1 - \epsilon_i)}{\epsilon_i}\right), \tag{B.3}$$

which is exactly (4.11).

Appendix C

Chapter 6 Proofs

C.1 Proof of Claim 14

For $i = 1, 2, \ldots, n$ and any $s \geq g$, we have

$$\begin{aligned}
&\ln \mathrm{Prob}\big\{M(g,g) \leq s\big\} \\
&\quad = \ln \prod_{k=0}^{h-1} (1 - q^{k-s}) = \sum_{k=0}^{g-1} \ln(1 - q^{k-s}) \\
&\quad = -\sum_{k=0}^{g-1} \sum_{j=1}^{\infty} \frac{1}{j} q^{(k-s)j} = -\sum_{j=1}^{\infty} \frac{1}{j} \sum_{k=0}^{g-1} q^{j(k-s)} \\
&\quad = -\sum_{j=1}^{\infty} \frac{1}{j} q^{-js} \frac{q^{jg} - 1}{q^j - 1} \\
&\quad = -\, q^{-(s-g)} \sum_{j=1}^{\infty} \frac{1}{j} q^{-(j-1)(s-g)} \frac{1 - q^{-jg}}{q^j - 1} \\
&\quad > q^{-(s-g)} \sum_{j=1}^{\infty} \frac{1}{j} \frac{1 - q^{-jg}}{1 - q^j} \\
&\quad = q^{-(s-g)} \ln \mathrm{Prob}\big\{M(g,g) \leq g\big\} \\
&\quad > q^{-(s-g)} \lim_{h \to \infty, q=2} \ln \mathrm{Prob}\big\{M(g,g) \leq g\big\}
\end{aligned}$$

The claim is obtained by setting

$$\alpha_{q,g} = -\ln \mathrm{Prob}\big\{M(g,g) \leq g\big\},$$

and

$$\alpha_{2,\infty} = -\lim_{g \to \infty, q=2} \ln \mathrm{Prob}\big\{M(g,g) \leq g\big\}.$$

C.2 Proofs of Generalized Results of Collector's Brotherhood Problem

Proof of Theorem 8

Our proof generalizes the symbolic method of [13].

Let ξ be the event that the number of copies of coupon G_i is at least m_i for every $i = 1, 2, \ldots, n$. For integer $t \geq 0$, let $\xi(t)$ be the event that ξ has occurred after a total of t samplings, and let $\bar{\xi}(t)$ be the complementary event. Then, the tail probability $\text{Prob}[T(\boldsymbol{\rho}, \mathbf{m}) > t] = \text{Prob}[\bar{\xi}(t)] = \nu_t$.

To derive ν_t, we introduce an operator f acting on an n-variable polynomial g. f removes all monomials $x_1^{w_1} x_2^{w_2} \ldots x_n^{w_n}$ in g satisfying $w_1 \geq m_1, \ldots, w_n \geq m_n$. Note that f is a linear operator, i.e., if g_1 and g_2 are two polynomials in the same n variables, and a and b two scalars, we have $af(g_1) + bf(g_2) = f(ag_1 + bg_2)$.

Each monomial in $(x_1 + \cdots + x_n)^t$ corresponds to one of the n^t possible outcomes of t samplings, with the exponent of x_i being the number of copies of coupon G_i. Since the samplings are independent, the probability of an outcome $x_1^{w_1} x_2^{w_2} \ldots x_n^{w_n}$ is $\rho_1^{w_1} \rho_2^{w_2} \ldots \rho_n^{w_n}$. Hence, the probability of $\bar{\xi}(t)$ is $f((x_1 + \cdots + x_n)^t)$, when evaluated at $x_i = \rho_i$ for $i = 1, 2, \ldots n$, i.e.,

$$\nu_t = f((x_1 + \cdots + x_n)^t)|_{x_i = \rho_i, i = 1, \ldots, n}. \tag{C.1}$$

Hence, (C.1) and (5.3) lead to

$$\varphi_{T(\boldsymbol{\rho}, \mathbf{m})}(z) = \sum_{t \geq 0} f\left((x_1 + \cdots + x_n)^t\right) z^t |_{x_i = \rho_i, i = 1, \ldots, n}.$$

The identity

$$\int_0^\infty \frac{1}{t!} y^t e^{-y} dy = 1$$

and the linearity of the operator f imply that

$$\begin{aligned}\varphi_{T(\boldsymbol{\rho},\mathbf{m})}(z) &= \int_0^\infty \sum_{t\geq 0} \frac{f\left((x_1+\cdots+x_n)^t\right)}{t!} z^t y^t e^{-y} dy \\ &= \int_0^\infty f\Big(\sum_{t\geq 0} \frac{(x_1 zy+\cdots+x_n zy)^t}{t!}\Big) e^{-y} dy \\ &= \int_0^\infty f\left(\exp(x_1 zy+\cdots+x_n zy)\right) e^{-y} dy \end{aligned} \tag{C.2}$$

evaluated at $x_i = \rho_i, i = 1, \ldots, n$.

We next find the sum of the monomials in the polynomial expansion of $\exp(x_1+\cdots+x_n)$ that should be removed under f. Clearly, this sum should be $\prod_{i=1}^n \left(e^{x_i} - S_{m_i}(x_i)\right)$, where S is defined in (5.1) and (5.2)). Therefore,

$$\begin{aligned} & f\left(\exp(x_1 zy+\cdots+x_n zy)\right)|_{x_i=\rho_i, i=1,\ldots,n} \\ &= e^{zy} - \prod_{i=1}^n \left(e^{\rho_i zy} - S_{m_i}(\rho_i zy)\right). \end{aligned}$$

$$\varphi_{T(\boldsymbol{\rho},\mathbf{m})}(z) = \int_0^\infty \left[e^{zy} - \prod_{i=1}^n \left(e^{\rho_i zy} - S_{m_i}(\rho_i zy)\right)\right] e^{-y} dy \tag{C.3}$$

Proof of Corollary 9

Note that

$$\begin{aligned}\varphi_{T(\boldsymbol{\rho},\mathbf{m})}(z) &= \sum_{t=0}^\infty \text{Prob}[T(\boldsymbol{\rho},\mathbf{m}) > t] z^t \\ &= \sum_{t=0}^\infty \sum_{j=t+1}^\infty \text{Prob}[T(\boldsymbol{\rho},\mathbf{m}) = j] z^t \\ &= \sum_{j=1}^\infty \text{Prob}[T(\boldsymbol{\rho},\mathbf{m}) = j] \sum_{t=0}^{j-1} z^t \end{aligned}$$

$$E[T(\boldsymbol{\rho},\mathbf{m})] = \sum_{j=1}^\infty j\text{Prob}[T(\boldsymbol{\rho},\mathbf{m}) = j] = \varphi_{T(\boldsymbol{\rho},\mathbf{m})}(1).$$

Similarly,

$$\begin{aligned}\varphi'_{T(\boldsymbol{\rho},\mathbf{m})}(z) &= \sum_{t=0}^{\infty} t\text{Prob}[T(\boldsymbol{\rho},\mathbf{m}) > t]z^{t-1} \\ &= \sum_{j=1}^{\infty} \text{Prob}[T(\boldsymbol{\rho},\mathbf{m}) = j] \sum_{t=0}^{j-1} tz^{t-1} \\ \varphi'_{T(\boldsymbol{\rho},\mathbf{m})}(1) &= \sum_{j=1}^{\infty} \frac{1}{2} j(j-1)\text{Prob}[T(\boldsymbol{\rho},\mathbf{m}) = j].\end{aligned}$$

Hence,

$$\begin{aligned}E[T(\boldsymbol{\rho},\mathbf{m})^2] =& \sum_{j=1}^{\infty} j^2 \text{Prob}[T(\boldsymbol{\rho},\mathbf{m}) = j] \\ =& 2\varphi'_{T(\boldsymbol{\rho},\mathbf{m})}(1) + \varphi_{T(\boldsymbol{\rho},\mathbf{m})}(1),\end{aligned}$$

and consequently,

$$\text{Var}[T(\boldsymbol{\rho},\mathbf{m})] = 2\varphi'_{T(\boldsymbol{\rho},\mathbf{m})}(1) + \varphi_{T(\boldsymbol{\rho},\mathbf{m})}(1) - \varphi^2_{T(\boldsymbol{\rho},\mathbf{m})}(1).$$

We have

$$\begin{aligned}\varphi'_{T(\boldsymbol{\rho},\mathbf{m})}(z) = \int_0^{\infty} x \Bigg(& e^{-x(1-z)} - \sum_{i=1}^{n} \rho_i \frac{e^{-\rho_i x(1-z)} - S_{m_i-1}(\rho_i xz)e^{-\rho_i x}}{e^{-\rho_i x(1-z)} - S_{m_i}(\rho_i xz)e^{-\rho_i x}} \cdot \\ & \cdot \prod_{j=1}^{n} \left(e^{-\rho_j x(1-z)} - S_{m_j}(\rho_j xz)e^{-\rho_j x} \right) \Bigg) dx,\end{aligned}$$

and from there, we can get $\varphi'_{T(\boldsymbol{\rho},\mathbf{m})}(1)$ and $\text{Var}[T(\boldsymbol{\rho},\mathbf{m})]$.

Proof of Theorem 10

We again apply the Newman-Shepp symbolic method. Similar to the proof of Theorem 8, we introduce an operator f acting on an n-variable polynomial g. For a monomial $x_1^{w_1} \dots x_n^{w_n}$, let i_j be the number of exponents w_u among $w_1, \dots, w_n$ satisfying $w_u \geq k_j$, for $j = 1, \dots, A$. f removes all monomials $x_1^{w_1} \dots x_n^{w_n}$ in g satisfying $i_1 \geq k_1, \dots, i_A \geq k_A$ and $i_1 \leq \dots \leq i_A$. f is again a linear operator. One can see that

$$\varphi_{U(\mathbf{m},\mathbf{k})}(z) = \int_0^{\infty} f\left(\exp(x_1 zy + \dots + x_n zy)\right) e^{-y} dy \big|_{x_1 = x_2 = \dots = x_n = \frac{1}{n}}.$$

We choose integers $0 = i_0 \le i_1 \le \cdots \le i_A \le i_{A+1} = n$, such that $i_j \ge k_j$ for $j = 1, \ldots, A$, and then partition indices $\{1, \ldots, n\}$ into $(A+1)$ subsets $\mathcal{I}_1, \ldots, \mathcal{I}_{A+1}$, where $\mathcal{I}_j (j = 1, \ldots, A+1)$ has $i_j - i_{j-1}$ elements. Then

$$\prod_{j=1}^{A+1} \prod_{i \in \mathcal{I}_j} (S_{m_{j-1}}(x_i) - S_{m_j}(x_i)) \tag{C.4}$$

equals the sum of all monomials in $\exp(x_1 + \cdots + x_n)$ with $(i_j - i_{j-1})$ of the n exponents smaller than m_{j-1} but greater than or equal to m_j, for $j = 1, \ldots, A+1$. (Here S is as defined by (5.1)-(5.2).) The number of such partitions of $\{1, \ldots, n\}$ is equal to $\binom{n}{n-i_A, \ldots, i_2 - i_1, i_1} = \prod_{j=0}^{A} \binom{i_{j+1}}{i_j}$. Finally, we need to sum the terms of the form (C.4) over all partitions of all choices of $i_1, \ldots, i_A$ satisfying $k_j \le i_j \le i_{j+1}$ for $j = 1, \ldots, A$:

$$f\left(\exp(x_1 zy + \cdots + x_n zy)\right)\big|_{x_1 = \cdots = x_n = \frac{1}{n}} \tag{C.5}$$

$$= \exp(zy) - \sum_{\substack{(i_0, i_1, \ldots, i_{A+1}): \\ i_0 = 0, i_{A+1} = n \\ i_j \in [k_j, i_{j+1}] \\ j = 1, 2, \ldots, A}} \prod_{j=0}^{A} \binom{i_{j+1}}{i_j} \left[S_{m_j}(\frac{zy}{n}) - S_{m_{j+1}}(\frac{zy}{n}) \right]^{i_{j+1} - i_j}. \tag{C.6}$$

Bringing (C.5) into (C.2) gives our result in Theorem 10.

C.3 Proof of Theorem 15

$$\begin{aligned} E[W(\boldsymbol{\rho}, \mathbf{g})] &= \sum_{\mathbf{m}} \left(\prod_{i=1}^{n} \Pr[M_i = m_i] \right) E[T(\boldsymbol{\rho}, \mathbf{m})] \\ &= \int_0^\infty \left[1 - \prod_{i=1}^{n} \sum_{m_i} \Pr[M_i = m_i](1 - S_{m_i}(\rho_i x) e^{-\rho_i x}) \right] dx \qquad \text{(C.7)} \\ &= \int_0^\infty \left(1 - \prod_{i=1}^{n} \left(1 - e^{-\rho_i x} E_{M_i}\left[S_{M_i}(\rho_i x)\right]\right) \right) dx. \end{aligned}$$

(C.7) comes from the distributivity.

Since

$$E_{M_i}\left[S_{M_i}(\rho_i x)\right] = \sum_{j=0}^{\infty} \frac{(\rho_i x)^j}{j!} \Pr[M_i > j],$$

by Claim 14,

$$E_{M_i}\left[S_{M_i}(\rho_i x)\right] < S_{g_i}(\rho_i x) + \sum_{j=g_i}^{\infty} \frac{(\rho_i x)^j}{j!} \alpha_{q,g} q^{-(j-g)}$$

$$= S_{g_i}(\rho_i x) + \alpha_{q,g_i} q^{g_i} e^{\rho_i x/q} - \alpha_{q,g_i} q^{g_i} S_{g_i}(\rho_i x/q),$$

where

$$\alpha_{q,g_i} = -\ln \Pr\{M(g_i, g_i) \le g_i\} = -\sum_{k=0}^{g_i-1} \ln\left(1 - q^{k-g_i}\right)$$

for $i = 1, 2, \ldots, n$.

Hence, we have (6.8).

Expression (6.9) for $E[W^2(\boldsymbol{\rho}, \mathbf{g})]$ can be derived in the same manner, and then the expression for $\mathrm{Var}[W(\boldsymbol{\rho}, \mathbf{g})]$ immediately follows.

C.4 Proof of Theorem 22

Without loss of generality, let $I = \{1, 2, \ldots, s\}$ and $j = s+1$, and define $\mathcal{R}_s = \cup_{i=1}^{s} R_i$, $\mathcal{B}_s = \cup_{i=1}^{s} B_i$, and $\mathcal{G}_s = \cup_{i=1}^{s} G_i$ for $s = 0, 1, \ldots, n-1$. Then, $E\left[|\left(\cup_{i\in I} G_i\right) \cap G_j|\right] = E\left[|\mathcal{G}_s \cap G_{s+1}|\right]$. For any two sets X and Y, we use $X + Y$ to denote $X \cup Y$ when $X \cap Y = \emptyset$.

$$\mathcal{G}_s \cap G_{s+1} = (\mathcal{B}_s + \mathcal{R}_s \backslash \mathcal{B}_s) \cap (B_{s+1} + R_{s+1})$$

$$= \mathcal{B}_s \cap R_{s+1} + \mathcal{R}_s \cap B_{s+1} + (\mathcal{R}_s \backslash \mathcal{B}_s) \cap R_{s+1},$$

and therefore

$$E[|\mathcal{G}_s \cap G_{s+1}|] = E[|\mathcal{B}_s \cap R_{s+1}|] + \tag{C.8}$$

$$E[|\mathcal{R}_s \cap B_{s+1}|] + E[|(\mathcal{R}_s \backslash \mathcal{B}_s) \cap R_{s+1}|].$$

Using Claim 18, we have

$$E[|\mathcal{B}_s \cap R_{s+1}|] = sh\pi, \tag{C.9}$$

$$E[|\mathcal{R}_s \cap B_{s+1}|] = h[1 - (1-\pi)^s], \tag{C.10}$$

$$E[|(\mathcal{R}_s \backslash \mathcal{B}_s) \cap R_{s+1}|] = (n - s - 1)h\pi[1 - (1-\pi)^s], \tag{C.11}$$

where π is as defined in Claim 18. Bringing (C.9)-(C.11) into (C.8), we obtain (6.10).

Furthermore, when $n \to \infty$, if $l/h \to \alpha$ and $s/n \to \beta$, then

$$\begin{aligned}
E[|\mathcal{G}_s \cap G_{s+1}|] =& g \cdot [1 - \bar{\pi}^s] + sh \cdot \pi\bar{\pi}^s \\
\to& h(1+\alpha)\Big[1 - \Big(1 - \frac{\alpha}{n-1}\Big)^{n\beta}\Big] + h\alpha\beta\Big(1 - \frac{\alpha}{n-1}\Big)^{n\beta} \\
\to& h\Big[(1+\alpha)(1-e^{-\alpha\beta}) + \alpha\beta e^{-\alpha\beta}\Big] \\
=& h\Big[1 + \alpha - (1 + \alpha - \alpha\beta)e^{-\alpha\beta}\Big]
\end{aligned}$$

References

[1] Y. Li, P. Vingelmann, M. V. Pedersen, and E. Soljanin, "Round-robin streaming with generations," in *Proc. Symposium on Network Coding, Theory, and Applications (NetCod '12)*, Boston, Massachusetts, Jun. 2012. v

[2] Y. Li, E. Soljanin, and P. Spasojević, "Three schemes for wireless coded broadcast to heterogeneous users," *Physical Communication*, no. 0, pp. –, 2012. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1874490712000420 v

[3] Y. Li, E. Soljanin, and P. Spasojevic, "Effects of the generation size and overlap on throughput and complexity in randomized linear network coding," *IEEE Trans. Inf. Theory*, vol. 57, no. 2, pp. 1111 –1123, Feb. 2011. v

[4] M. Hofmann and L. R. Beaumont, *Content Networking: Architecture, Protocols, and Practice*. Morgan Kaufmann, 2005. 1

[5] "Bit torrent file sharing protocol," http://bitconjurer.org/bittorrent/. 1, 77

[6] S. Androutsellis-Theotokis and D. Spinellis, "A survey of peer-to-peer content distribution technologies," *ACM Comput. Surv.*, vol. 36, pp. 335–371, December 2004. [Online]. Available: http://doi.acm.org/10.1145/1041680.1041681 1

[7] J. Byers, M. Luby, and M. Mitzenmacher, "A digital fountain approach to asynchronous reliable multicast," *Selected Areas in Communications, IEEE Journal on*, vol. 20, no. 8, pp. 1528 – 1540, oct 2002. 2

[8] M. Luby, "LT codes," in *The 43rd Annual IEEE Symposium on Foundations of Computer Science*, Nov. 2002, pp. 271–280. 2, 6, 7, 8, 29, 33

[9] A. Shokrollahi, "Raptor codes," *IEEE Trans. Inf. Theory*, vol. 52, no. 6, pp. 2551–2567, Jun. 2006. 2, 7, 8, 29

[10] R. Ahlswede, N. Cai, S.-Y. Li, and R. Yeung, "Network information flow," *IEEE Trans. Inf. Theory*, vol. 46, no. 4, pp. 1204–1216, Jul. 2000. 2

[11] T. Ho, R. Koetter, M. Medard, D. Karger, and M. Effros, "The benefits of coding over routing in a randomized setting," in *Proc. IEEE International Symposium on Information Theory (ISIT'03)*, 2003, p. 442. 2

[12] C. Gkantsidis and P. Rodriguez, "Network coding for large scale content distribution," in *Proc. the 24th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM'05)*, vol. 4, Miami, FL, Mar. 2005, pp. 2235–2245. 2, 77

[13] D. Newman and L. Shepp, "The double dixie cup problem," *The American Mathematical Monthly*, vol. 67, no. 1, pp. 58–61, Jan. 1960. 4, 52, 53, 54, 57, 81, 96

[14] D. Foata and D. Zeilberger, "The collector's brotherhood problem using the Newman-Shepp symbolic method," *Algebra Universalis*, vol. 49, no. 4, pp. 387–395, 2003. 4, 52, 53, 81

[15] Z. Liu, C. Wu, B. Li, and S. Zhao, "UUSee: Large-scale operational on-demand streaming with random network coding," in *Proc. the 30th IEEE Conference on Computer Communications (INFOCOM'10)*, San Diego, California, Mar. 2010. 6

[16] R. Brent, G. Shuhong, and A. Lauder, "Random Krylov spaces over finite fields," *SIAM J. Discret. Math.*, vol. 16, no. 2, pp. 276–287, Feb. 2003. [Online]. Available: http://dx.doi.org/10.1137/S089548010139388X 16

[17] "Nokia energy profiler." [Online]. Available: http://www.developer.nokia.com/Resources/Tools_and_downloads/Other/Nokia_Energy_Profiler 20

[18] J. S. Plank, S. Simmerman, and C. D. Schuman, "Jerasure: A library in C/C++ facilitating erasure coding for storage applications - Version 1.2," University of Tennessee, Tech. Rep. CS-08-627, August 2008. 23

[19] M. Khansari, A. Zakauddin, W.-Y. Chan, E. Dubois, , and P. Mermelstein, "Approaches to layered coding for dual-rate wireless video transmission," in *Proc. IEEE Int'l Conf. Image Processing*, Austin, TX, Oct. 1994, pp. 258–262. 29

[20] M. Gallant and F. Kossentini, "Rate-distortion optimized layered coding with unequal error protection for robust internet video," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 11, no. 3, pp. 357 –372, mar 2001. 29

[21] V. K. Goyal, "Multiple Description Coding: Compression Meets the Network," *IEEE Signal Proc. Magazine*, vol. 18, no. 5, pp. 74–94, 2001. 29

[22] L. Tan, A. Khisti, and E. Soljanin, "Quadratic Gaussian source broadcast with individual bandwidth mismatches," in *2012 IEEE International Symposium on Information Theory (ISIT 2012)*, 2012, p. to appear. 29

[23] A. Kamra, J. Feldman, V. Misra, and D. Rubenstein, "Growth codes: Maximizing sensor network data persistence," in *Proc. of ACM SIGCOMM*, 2006. 29, 39, 40

[24] P. Maymounkov, N. Harvey, and D. S. Lun, "Methods for efficient network coding," in *Proc. 44th Annual Allerton Conference on Communication, Control, and Computing*, Monticello, IL, Sept. 2006. 29, 41, 52, 65

[25] Y. Li, E. Soljanin, and P. Spasojevic, "Effects of the generation size and overlap on throughput and complexity in randomized linear network coding," *IEEE Trans. Inf. Theory*, vol. 57, no. 2, pp. 1111 –1123, Feb. 2011. 29

[26] Y. Li and E. Soljanin, "Rateless codes for single-server streaming to diverse users," in *Proc. the 47th Annual Allerton Conference on Communication, Control, and Computing*, Monticello, IL, Sept. 2009. 29, 46

[27] J. Korner and K. Marton, "General broadcast channels with degraded message sets," vol. 23, no. 1, pp. 60–64, Jan. 1977. 30

[28] P. A. Chou, H. J. Wang, and V. N. Padmanabhan, "Layered multiple description coding," in *Packet Video Workshop, Institute of Electrical and Electronics Engineers, Inc.*, Apr. 2003. 31

[29] S. Kokalj-Filipovic, E. Soljanin, and Y. Gao, "Cliff effect suppression through multiple-descriptions with split personality," in *Proc. 2011 IEEE International Symposium on Information Theory (ISIT 2011)*, pp. 948–952. 31

[30] S. Sanghavi, "Intermediate Performance of Rateless Codes," in *Information Theory Workshop ITW 2007*. 33, 37, 38

[31] G. Maatouk and A. Shokrollahi, "Analysis of the second moment of the LT decoder," in *Proceedings of the 2009 IEEE International Conference on Symposium on Information Theory - Volume 4*, ser. ISIT'09. Piscataway, NJ, USA: IEEE Press, 2009, pp. 2326–2330. 34

[32] A. Dimakis, J. Wang, and K. Ramchandran, "Unequal growth codes: Intermediate performance and unequal error protection for video streaming," in *9th IEEEWorkshop on Multimedia Signal Processing (MMSP)*, oct 2007, pp. 107–110. 39

[33] R. Razavi, M. Fleury, M. Altaf, H. Sammak, and M. Ghanbari, "H.264 video streaming with data-partitioning and growth codes," in *16th IEEE International Conference on Image Processing (ICIP)*, nov. 2009, pp. 909 –912. 39

[34] P. Flajolet, D. Gardy, and L. Thimonier, "Birthday paradox, coupon collectors, caching algorithms and self-organizing search," *Discrete Applied Mathematics*, vol. 39, no. 3, pp. 207–229, 1992. 41, 56

[35] E. Soljanin, "On collecting coupons, degrees of freedom, and content distribution." 51

[36] P. A. Chou, Y. Wu, and K. Jain, "Practical network coding," in *Proc. 41st Annual Allerton Conference on Communication, Control, and Computing*, Monticello, IL, Oct. 2003. 52

[37] A. Boneh and M. Hofri, "The coupon-collector problem revisited – a survey of engineering problems and computational methods," *Stochastic Models*, vol. 13, pp. 39 – 66, 1997. [Online]. Available: http://www.informaworld.com/10.1080/15326349708807412 53, 56

[38] L. Flatto, "Limit theorems for some random variables associated with urn models," *The Annals of Probability*, vol. 10, no. 4, pp. 927–934, 1982. [Online]. Available: http://www.jstor.org/stable/2243548 57, 58

[39] A. N. Myers and H. S. Wilf, "Some new aspects of the coupon collector's problem," *SIAM Rev.*, vol. 48, no. 3, pp. 549–565, 2006. 57

[40] P. Erdös and A. Rényi, "On a classical problem of probability theory," *Magyar Tud. Akad. Mat. Kutató Int. Közl.*, vol. 6, pp. 215–220, 1961. 57

[41] V. F. Kolchin, B. A. Sevast'Yanov, and V. P. Chistyakov, *Random Allocations*. Washington, D.C.: V.H.Winston & Sons, a Division of Scripta Technica, Inc., 1978. 58

[42] C. Fragouli and E. Soljanin, *Network Coding Applications*. now Publishers Inc., 2007, vol. 2, no. 2, pp. 144–145. 60, 93

[43] A. Heidarzadeh and A. Banihashemi, "Overlapped chunked network coding," in *IEEE Information Theory Workshop (ITW'10)*, Jan. 2010, pp. 1–5. 65

[44] D. Silva, W. Zeng, and F. Kschischang, "Sparse network coding with overlapping classes," in *Proc. Workshop on Network Coding, Theory, and Applications (NetCod '09)*, Lausanne, Switzerland, Jun. 2009, pp. 74–79. 65

[45] A. R. Bharambe, C. Herley, and V. N. Padmanabhan, "Analyzing and improving bittorrent performance," Microsoft Research, Tech. Rep. MSR-TR-2005-03, February 2005. 77

[46] W. Feller, *An Introduction to Probability Theory and Its Applications*, 3rd ed. John Wiley & Sons, 1970, vol. 1, p. 225. 93

[47] D. Sejdinovic, R. Piechocki, A. Doufexi, and M. Ismail, "Fountain code design for data multicast with side information," *Wireless Communications, IEEE Transactions on*, vol. 8, no. 10, pp. 5155 –5165, october 2009. 94

[48] R. Gummadi, A. Shokrollahi, and R. Sreenivas, "Broadcasting with side information using fountain codes," in *IEEE Information Theory Workshop*, Cairo, Egypt, 2010. 94

Curriculum Vita

Yao Li

2006-2012 Ph.D. in WINLAB, Electrical and Computer Engineering, Rutgers University

2002-2006 B.S. in Information Engineering and Electronics, Tsinghua University, China

2011 Visiting Research Scientist, ETIS Lab, Centre National de la Recherche Scientifique(CNRS), Cergy-Pontoise, France

2007-2011 Graduate assistant, WINLAB, Department of Electrical and Computer Engineering, Rutgers University

2006-2007 Teaching assistant, Department of Electrical and Computer Engineering, Rutgers University

2012 Yao Li, Emina Soljanin and Predrag Spasojevic, "Three Schemes for Coded Broadcast in a Wireless Network of Heterogeneous Users", in press, *Elsevier Physical Communications: Special Issue on Network Coding*, 2012

Yao Li, Péter Vingelmann, Morten Videbæk Pedersen, and Emina Soljanin, "Round-Robin Streaming with Generations", *NetCod 2012*, Boston, MA.

2011 Yao Li, Emina Soljanin and Predrag Spasojević, "Effects of Generation Size and Overlap on Randomized Linear Network Coding", *the Special Issue of the IEEE Transactions on Information Theory: Facets of Coding Theory: from Algorithms to Networks*, vol. 57, no. 2, pp. 1111 – 1123, Feb. 2011.

2010 Yao Li, Emina Soljanin and Predrag Spasojević, "Collecting Coded Coupons Over Generations", *ISIT 2010*, Austin, Texas

Yao Li, Emina Soljanin and Predrag Spasojević, "Collecting Coded Coupons Over Overlapping Generations", *NetCod 2010*, Toronto, Canada

2009 Yao Li and Emina Soljanin, "Rateless Codes for Single-Server Streaming to Diverse Users", Invited Paper, *47th Annual Allerton Conference on Communication, Control, and Computing*, Monticello, IL, 2009