CODING AND SCHEDULING IN NETWORKS FOR ERASURES AND BROADCAST

BY

SUBHA RAMAKRISHNA GUMMADI

DISSERTATION

Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy in Electrical and Computer Engineering
in the Graduate College of the
University of Illinois at Urbana-Champaign, 2011

Urbana, Illinois

ABSTRACT

This dissertation is concerned with the design and analysis of algorithms that address two related issues in communication networks, namely erasures and broadcast. Erasures are an appropriate model for communication channels from a network layer perspective. A class of efficient and flexible codes known as fountain codes, is available to deal with erasures for the basic erasure channel. However, in the network applications that we consider, it remains a challenging problem to design efficient and scalable codes. For an erasure code, the efficiency of encoding and decoding algorithms is distinct from the efficiency of reconstructing erased code symbols from other code symbols, which is of importance in storage applications. In our work, we propose new codes together with algorithms to efficiently repair lost code symbols, simultaneously with low encoding and decoding complexities. Our work on codes for storage also leads us to systematic fountain codes with improved complexity. We also study the design and analysis of degree distributions for fountain codes when the receivers have side information, and we provide upper and lower bounds on the overhead. In a network with multiple hops from the source, we construct a code to import the one-hop traits of LT codes end-to-end using an idea based on online encoding, which is also one of the components of the repair algorithm for storage codes that we propose.

We then consider wireless erasure networks, where local broadcast is another property which influences the role of coding beyond that of merely dealing with erasures. We show that feedback signaling is a critical factor that defines the role of coding in this situation, in the sense that it is one way to avoid the extensive feedback signaling that is necessary for routing policies. To characterize this more precisely, we consider a formal notion of restricted feedback signaling and derive the throughput of routing policies with restricted feedback on a two-hop network. This allows us to obtain a lower bound on the throughput when the losses are independent, and also to show that it is possible to have arbitrary degradation of throughput with dependent losses.

Finally, we consider optimization problems involving the control of a queue whose server is defined by the broadcast property, where each service satisfies all the customers simultaneously. Customers in the queue incur holding costs. We consider two constraints on the server and derive the associated optimal controls. For the first constraint, a constant non-negative cost is charged per service whereas in the second type, we consider an online running constraint on the ability to operate the broadcast server. To address this, we solve a more general problem called the online knapsack problem where one needs to choose a sequence of actions over time, with each action incurring a stochastic cost and also consuming a resource, which is replenished stochastically over time with a given rate. The objective is to minimize the total cost subject to the constraint of not exhausting the resource at any point. We derive a limiting characterization of the optimal policy by showing the convergence of the scaled value function to that of a continuous time problem when the discount factor vanishes. In other words, this provides a method to approximate such problems when the discount factor is effective over a long duration and consequently, the magnitude of transactions in each time slot vanishes in comparison to the long-term utility.

ACKNOWLEDGMENTS

I would like to thank my adviser, Professor R. S. Sreenivas, for his support and guidance over the course of my graduate study. I would also like to thank my committee members, Professors Chandra Chekuri, Bruce Hajek, P. R. Kumar, Olgica Milenkovic and R. Srikant, for spending their valuable time to serve on my dissertation committee and also for their feedback, which improved the dissertation significantly. I would also like to thank all my collaborators over the course of my graduate study with whom I had the privilege of working on many interesting topics both related and unrelated to the dissertation. They include Kyomin Jung, Devavrat Shah, Laurent Massoulie, Amin Shokrollahi, Carl Nuzman, Phil Whiting, Alexandre Proutiere and Peter Key. I am grateful to all of them for the enriching learning experiences they have offered me.

I would like to thank friends and colleagues at UIUC and, especially, at CSL over the years for both their intellectual as well as social companionship. Needless to say, I am indebted to my near and dear, especially my family, for their unconditional support and encouragement.

Last but not the least, I wish to thank the NSF and the Vodafone foundation fellowship grants which supported a large part of my graduate study. I am also grateful to the ECE department and Professor Kumar for the teaching assistantship opportunity during the Spring 2011 semester. I am also thankful for the student travel grants that allowed me to attend some of the conferences.

TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION

An erasure refers to an event where any *symbol* that is intended to convey information (in the form of a transmission over a communication channel, or alternately, stored in a memory unit) is lost. A communications channel consists of an input alphabet and an output alphabet. In general, errors can happen in arbitrary ways transforming any input symbol to any output symbol. The erasure channel represents one specific model of this channel where input to output symbol transformations have a special *one-way* structure. More precisely, let A be the input alphabet to this channel. The output alphabet consists of $A \bigcup \{\times\}$, i.e. it has exactly one additional character, $\times$, also called the erasure symbol. The errors are one-way, in the sense that the only possible transformation of an input symbol to output symbol that does not faithfully represent what was transmitted is by way of the input symbol being converted into the special erasure symbol, $\times$ [1–3]. At a low enough level, this model is not an accurate representation of a communications channel because errors need not have this special one-way structure. However, at an abstraction relevant to the network layer, where messages are represented in terms of *packets*, this kind of a channel model is most accurate from an operational point of view.

Arguably, the erasure channel is the *simplest* nontrivial communication channel. However, the relative simplicity of the erasure channel as opposed to more *difficult* channels, comes with a commensurately higher bar for the codes designed for this channel, with regards to their rate optimality, flexibility and also the efficiency of the associated algorithms. Presumably due to the increased relevance of this channel model due to the proliferation of networks, there has been a seminal body of work that introduced paradigms for code designs that are very versatile, rate optimal and highly efficient at encoding/decoding [4–8]; in addition, they have also enjoyed significant practical success [9,10]. When compared to other channels, one may comparatively see the erasure channel as a *solved problem* due to their optimality on basic metrics like ratelessness, overhead optimality and encoding/decoding complexity. Nevertheless, various evolving network applications provide a rich source of important and challenging issues that remain to be addressed which fundamentally involve dealing with erasures. This is the primary motivation and a starting point for this dissertation.

Broadcast refers to an abstraction derived from the wireless transmission model, where a single transmission can potentially address a group of clients. Although this is ubiquitous in any wireless transmission, the way most practical systems deal with this is essentially by ignoring it. The protocol model, which is also the basis for much theoretical work utilizes a graph based model in which local broadcast is essentially treated as interference (e.g. [11–14]). In reality, local broadcast represents a diversity gain associated with the channel that could be exploited by either a well-adapted routing scheme or by network coding. In general, coding in the network has two distinct roles in the absence of this local broadcast feature. The first arises from the fact that even without any errors/losses, information flow is fundamentally different from commodity flow, an aspect that points to the field of network coding [15], starting with the pioneering work of [16]. In wireline settings, network coding addresses strictly more general settings than that of a unicast transmission. On the other hand, forward error correction (FEC) addresses another aspect that stems from lossy transmissions. The role of coding in the context of wireless broadcast is something that needs to be distinguished from the previous two concerns, which are relatively better understood. Notably, in this context, the actual performance of optimal coding schemes is better articulated than are the limits of routing policies, both with and without restrictions on feedback signaling. This creates an avenue to discuss the role of coding by way of a better understanding about the limits of optimal routing schemes and then contrast it with optimal coding schemes.

The broadcast property also motivates us to study a very basic queuing model, with the defining property that the service time is completely disconnected from the actual number of customers awaiting the said service/broadcast. In such a model, the number of customers is repeatedly reset to zero every time the broadcast service is rendered. Two alternate views of this situation lead to two different types of Markov decision problems (MDP) in which decisions have to be repeatedly made on how aggressively the server needs to be operated so as to optimize certain customer holding costs over an infinite horizon. In the first viewpoint, we consider a completely *observable* system, where the decision maker can fine tune the broadcast rate, after observing the exact number of customers in the system at each given time. However, the broadcast constraint is kept simple in this setting with each service being charged a fixed non-negative cost at any time. The challenge here mainly stems from the holding cost structure, in which we tackle non-monotone convex holding costs on the customer state space. From the alternate viewpoint, we consider a less responsive decision maker, who only decides the broadcast rate once per broadcast. Since the customers are reset to zero each time the server makes its decision, this allows us to expand our consideration to a more challenging cost/constraint structure on the service, coupled across time. More precisely, we consider a resource, replenished at a fixed rate, and consumed at a rate that depends on the actions taken by the server. The server is constrained to operate subject to the availability of the resource at any time. This is modeled in the form of a general class of Markov decision problems on a continuous state space representing the resource, which is replenished and consumed at each instant. Although the general problem we address has nothing to do with broadcast as such, the ability to fit the given broadcast model problem into the general framework is based on the fact that the number of awaiting customers is reset to zero for each individual decision instance. Techniques like repeated value iteration to numerically estimate the value function for the continuous state space problem involves issues with error propagation that compromise their accuracy. These issues underscore the utility for an analytical procedure to approximating the value function with an easy and efficient characterization.

1.1 Overview and Organization

In Chapter 2, we begin with consideration of storage systems in Section 2.3 and outline the primary obstacle that erasure codes face over replication despite the large gains they offer with respect to redundancy. A metric of repair complexity intended to capture the latency associated with repairing erased code symbols is proposed in Section 2.3.1. In Sections 2.3.4 and 2.3.7, we design codes that achieve order-optimal repair complexity while simultaneously importing many efficient characteristics of conventional fountain codes. Subsequently, one of these constructions also leads to a new solution to the problem of designing efficient systematic rateless codes, as seen in Section 2.4.1. This construction offers an order reduction in the encoding complexity for systematic raptor codes compared to the state of the art, at the expense of a minimal loss in overhead. These sections of the Chapter are the basis for a paper, [17], under submission. We then explore the design of fountain codes for a side information problem with incomplete information at the encoder in Section 2.5.1 and provide upper and lower bounds on the overhead along with an alternate solution that can exploit the newly constructed systematic raptor codes of Section 2.4. This part of the chapter is based on the paper [18]. We then consider the problem of broadcasting a fountain code over multiple hops without having to decode and re-encode the entire source message at each intermediate hop in Section 2.6. We propose a construction that simulates the characteristics of an LT code for single hop on an end-to-end level and also provide an analysis that justifies this construction. The final section is based on the paper [19].

In Chapter 3, we study a wireless erasure network model with a focus on understanding the role of coding in dealing with the local broadcast property of the wireless medium. Even without erasures and in wireline networks, network coding has an essential role for general multicast problems. In order to separate out this aspect from local broadcast, we focus on the case of wireless unicast. In the absence of constraints on feedback signaling, it is argued that coding has no role in achieving the capacity for wireless unicast due to a max-flow min-cut equivalence. To explore the issue of feedback signaling further, we consider a two-hop network and formulate a notion of restricted feedback signaling to reflect the constraint of routing packets without dynamically exploiting information about packet losses on other links in the neighborhood. We then characterize the throughput of the model under the restricted feedback signaling constraint. Using this characterization, we obtain a lower bound on the loss in throughput in the case of independent erasures across different links. Finally, it is shown that this independence assumption is critical by providing a counterexample with unbounded degradation of throughput in case of dependent losses. This chapter is based on the paper, [20]

In Chapter 4, we introduce a queuing model called the broadcast queuing model, in which each service to the queue clears all the customers. We study control of the server to optimize the infinite horizon discounted holding cost of customers in the queue subject to two types of constraints on the server. From the first constraint, we charge a fixed cost for each broadcast of the server, considered in Section 4.4. The structure of the optimal control is shown to be of the threshold type for any convex holding costs on the customer queue. This material was published in [21]. For the second constraint, we consider an online constraint on operation of the server in Section 4.5. The server is associated with a resource queue which has a fixed rate of arrivals. The broadcast server consumes this resource as fuel for its operation, proportional to its rate of operation. To derive the optimal control for this problem, we step back and study a more general problem in Section 4.6, which we call the generalized online knapsack problem. For this setting, a discrete time system is considered where a resource balance process with stochastic arrivals of a given rate evolves over time with departures defined according to the control actions chosen at each time. The control actions also lead to a sequence of costs at each time. The goal is to optimize the infinite horizon discounted cost subject to the maintaining a positive balance on the resource queue at all times. For this problem, we derive a limiting characterization of the optimal value function when the discount factor goes to zero. The value function as a function of the balance becomes unbounded when the discount factor tends to zero, but by scaling both the value and its argument appropriately with the discount factor, we identify its nontrivial limit by showing convergence to a continuous time approximated system. We then use this characterization to compute the value function and optimal control for the corresponding broadcast server problem with an online constraint in Section 4.6. A version of this result with focus on an alternate application to dynamic auctions with budget constraints is in preparation [22].

CHAPTER 2

FOUNTAIN CODES FOR STORAGE AND BROADCAST APPLICATIONS

2.1 Introduction

In a network, when a packet is transmitted, the intended receiver is typically able to detect when the received packet is corrupted, in which case it is meant to be discarded. Each packet is composed of a fixed number, l, of bits. This represents an erasure channel, with an input alphabet of size 2^l corresponding to the total number of potential messages that can be stored as a packet. In addition to these 2^l possible elements, the output alphabet has the erasure symbol, $\times$, representing a failed transmission. Upon passing through the channel, each packet, equivalently also referred to as a *symbol*, is either received in a form identical to that of the transmitted version, or transformed into the erasure symbol, $\times$. Without any forward error-correction, the receiver has to send an acknowledgment for each transmitted packet and the sender has to retransmit each erased packet repeatedly until the transmission is erasure-free. The feedback represents an overhead that can be avoided: by using coding across packets, messages can be transmitted at the maximum rate without the need for feedback. The erasure channel was first introduced by Elias [23] where each symbol is erased independently with a fixed probability, p. It was shown that the information theoretic capacity of this channel is $1 - p$ and that a random linear code could be used to transmit at any rate up to the capacity. Another classical block coding approach involves Reed-Solomon codes [3, 24], in which k of the message packets are coded together to produce n coded packets. A Reed-Solomon code is optimal with respect to storage redundancy: i.e., any k coded packets from among the n are sufficient to decode the original k source packets, which is obviously optimal in this regard. The operations performed by a Reed-Solomon code to achieve this are in the $q-$ary field with $q = 2^l > N$, which represents the total number of possible l-bit sequences that constitute a distinct packet. However, standard implementations have an encoding/decoding complexity of order $k(n-k)\log n$ packet operations, which is inefficient. More recently, researchers have designed codes to improve on the encoding/decoding efficiency of erasure codes, while maintaining the optimality with respect to redundancy. Alon and Luby [25] designed codes with encoding/decoding complexity $O(n\ln(1/\epsilon)/\epsilon)$ while achieving a rate which is order ϵ from the capacity. The encoding/decoding complexity was improved to $O(n\ln(1/\epsilon))$ for a similar overhead guarantee by using a design based on sparse graph codes by Luby et al. [26]. More generally, constructions based on sparse graphs and iterative decoding have also been critical in designing efficient codes for other channels, originally pioneered by Gallager [27] and further developed in relatively more recent work such as [28–32]. (See [33] for a comprehensive introduction.) For the erasure channel, the constructions in [26, 34] provide highly efficient encoding and decoding algorithms in the asymptotic regime of large block length; however, an important limitation of all the aforementioned designs is that the encoder needs to have an estimate of the channel erasure probability for designing the code. In other words, the parameters n and k need to be known beforehand to compute the code. If it turns out that more than n code symbols are necessary, or that fewer than n are necessary, the code cannot be extended by computing new symbols or trimmed by deleting some of the computed symbols on-the-fly.

In many practical settings, the sender may not be able to estimate the channel erasure probability. Even when estimation is not a problem, it is desirable in broadcast settings with heterogeneous receivers for the same code to simultaneously transmit at rates close to the respective capacities of various receivers. A code that can accomplish this is said to be *rateless* because its throughput guarantees can be made without conditioning on the channel characteristics. The concept of rateless/fountain codes was developed [4–8] to achieve efficient encoding/decoding along with optimal overhead without requiring a fixed rate for design a priori; this paradigm has enjoyed remarkable success even in terms of practical use [9, 10]. A rateless encoder is able to generate an endless stream of random code symbols, which is the basis for the eponym, *fountain code*. The decoding guarantees are conditioned on the total number of symbols that a receiver collects. Each receiver, depending on its channel, can adaptively wait until it has enough symbols to decode.

Although their guarantees are primarily intended for the erasure channel, variants can also be applied to other channels [35, 36].

Consider a message divided into a sequence of packets, called *source symbols*. Each symbol comes from an alphabet of size $q = 2^l$, which corresponds to packets composed of l bits. A convenient feature of fountain codes is that the alphabet size has no lower bound and can even be binary for simplicity, i.e. $l = 1$. Fountain codes are designed for the asymptotic case where the interest is in codes designed for use on a large number, k, of source symbols, typically of the order of thousands, for practical effectiveness. The *block length* refers to k, the number of source symbols that constitute the message.[1] The code symbols are defined through binary addition (XOR) of subsets of source symbols. These subsets are random; therefore each code symbol represents a random variable that indicates the subset of source symbols XOR-ed to form the code symbol. They are independent and identically distributed (i.i.d.), and the source symbol indices forming code symbols are uniformly distributed over $\{1, \ldots, k\}$. In the case of LT codes, the *degree* of a given code symbol is the number of source symbols that are XOR-ed to compute the code symbol. The average degree, which is the average of the expected degree of the individual code symbols, is a measure of the *per-symbol* encoding complexity of a (random) code. Unless otherwise stated, we typically measure encoding/decoding complexity in per-symbol units in this chapter. The code is associated with a *degree distribution*, which is the probability distribution for the number of source symbols XOR-ed in each (i.i.d.) code symbol. An important feature of the LT code is the use of an iterative decoder, whose success is probabilistic. Note that if a code symbol has degree one, it can be processed to decode the source symbol that it represents. At each iteration, the LT decoder operates by identifying a code symbol of degree one, and then uses it to reduce the degree of other code symbols, thereby possibly increasing the population of degree one symbols that could be processed during the next iteration. If the decoder does not run out of degree one symbols till the end, then the decoding process succeeds. For a large enough k and for any $\delta > 0$, Luby [6] proposed a degree distribution, called the *robust soliton* distribution, which ensures this event with probability at least $1 - \delta$, when starting with any set of $k + o(k)$ code

[1] In other contexts, block length usually refers to n, the number of code symbols that form the code, but this is not of relevance here because of the rateless nature of the codes.

symbols, where $o(k)$ is a term of order roughly equal to $\sqrt{k}\ln^2\frac{k}{\delta}$. The average degree for this distribution is of the order $O(\log k)$. The analysis for the case of non asymptotic block length has also been studied by Karp et al. [37].

The set of all code symbols of reduced degree one that have not yet been decoded (because they have not been processed yet) is called the *gross ripple*, using the terminology from [38]. For any given degree distribution, the asymptotic limit of the evolution of the gross ripple over the decoding process was characterized by Darling and Norris [39] using a Poisson approximation for the number of coded symbols. The size of the gross ripple changes due to three reasons during each iteration of decoding: (1) A symbol is removed to be processed from the gross ripple. (2) Duplicate symbols corresponding to the processed symbol from the gross ripple are eliminated. (3) New symbols with reduced degree one appear because of subtracting the processed symbol. This process specifies an ODE for the fluid limit of the gross ripple, x_t, where $t \in [0,1]$ is the fraction of source symbols decoded at any given point. This ordinary differential equation (ODE) is given by $\dot{x}_t = -1 - \frac{x_t}{1-t} + (1-t)r\Omega''(t)$ with initial condition $x_0 = r\Omega'(0)$, where the three terms in the ODE correspond to the three factors in the stated order. The solution can be written as $x_t = (1-t)(r\Omega'(t) + \log(1-t))$. From this, the fraction of source symbols recovered asymptotically can be calculated as the point at which the gross ripple process reaches zero, which is given by $\inf\{t \geq 0 : r\Omega'(t) + \log(1-t) < 0\}$.[2]

The equation for the fluid limit suggests more flexibility for designing the degree distribution if the requirement of complete recovery is loosened to near-complete recovery. Shokrollahi proposed raptor codes [7],[3] which exploit this property to result in a much better encoding/decoding complexity compared to LT codes. A truncated version of the soliton distribution can be used with a constant per symbol encoding/decoding complexity, while providing near complete recovery. To avoid this problem, raptor codes use a high rate *precode*, which transforms the source symbols to a set of *intermediate symbols*, which are then encoded by the designed degree distribution. The source symbols can now be decoded even when a small fraction of the

[2]This statement assumes that the trajectory does not touch zero before crossing it, in which case the characterization is more subtle.

[3]This is independent from the work of Darling and Norris, whose connections to coding were published by Maneva and Shokrollahi in [40].

intermediate symbols are left undecoded by the LT/iterative decoder.

To summarize the critical aspects that make fountain codes useful in practice, we have:

1. **Ratelessness:** This refers to the property that the code symbol generated does not depend on the total number of symbols that constitute the code. In other words, the number of code symbols can be arbitrary and does not have to be specified for the encoder a priori. For decoding, one needs to collect a sufficient number of these code symbols and this guarantee is independent of the actual channel.

2. **Encoding/Decoding Complexity:** Raptor codes have (per-symbol) encoding/decoding complexities that do not scale with the block length. Since the guarantees provided are asymptotic in k, this property is quite important for scalability in terms of complexity. LT codes in contrast have a complexity of $O(\log k)$ per symbol.

3. **Overhead:** Overhead refers to the smallest $\delta > 0$ such that any $k(1+\delta)$ code symbols are sufficient to decode the k code symbols. This can be made arbitrarily small by designing the degree distributions appropriately.[4]

The scenarios addressed in this chapter have constraints in addition to the above three basic concerns. The first of these considers the use of erasure codes in storage systems. In addition to encoding/decoding complexity, overhead and the rateless property, an additional concern involves the latency associated with reconstructing the *erased* subset of code symbols from other non-erased *code symbols*, rather than directly from *source symbols*, as in the process of encoding. The requirement of encoding code symbols from other subsets of code symbols is closely related to the notion of locally decodable codes [41]. However, the primary concerns of our work, which differs from the setup of [41], are two fold: (1) we are interested in codes that work specifically for the erasure channel, and (2) we are interested in rateless designs. In addition, the use of network codes specifically for storage applications has been a vibrant research topic in recent past (e.g. [42–59]). We comment on the distinctions of our problem with the repair bandwidth considerations in

[4]In traditional coding theory terminology, this is also called the maximum distance separable (MDS) property.

Section 2.3.2. To the best of our knowledge, our constructions provide the first designs that provide guarantees on the repair complexity notion considered, and therefore set a benchmark for further research to improve the considered metrics. Another variant of the original problem that we address in this thesis involves the design of *systematic* versions of fountain codes, for which the state of the art construction has an encoding complexity that is much worse than the complexity without the systematic requirement. We propose constructions to address this issue by way of achieving a minimal tradeoff with the overhead. We then consider the problem of adapting fountain codes when the receivers already possess a subset of the data as side information. It turns out that the degree distributions optimized for the original problem without any side information are not well applicable to this variant; however, modified degree distributions based on the extent of side information available can do much better. Yet another application we consider involves the design of a rateless code that broadcasts to receivers over multiple hops without having to decode and re-encode at every step. This requires some novel constructions to ensure that intermediate nodes can start producing coded symbols while waiting to receive a full set to decode.

2.2 Contributions and Organization

The contributions made in this chapter are listed below:

- **Efficient repair for storage applications:** Typically, scalability of codes is reflected in their encoding/decoding complexity. In storage systems, however, one has an additional constraint due to the need to dynamically repair the erased code symbols without having to decode the entire source message each time a failure occurs. In this chapter, we first propose a definition of repair complexity that can be used to capture this notion in Section 2.3.1. We then illustrate with examples in Section 2.3.3 why common codes provide either optimal (constant) repair complexity with arbitrarily bad overhead (repetition coding) or, alternatively, optimal overhead with no better than a linear per symbol repair complexity. We then propose the first constructions of codes along with the repair algorithms that ensure an efficient repair property, based on a variant of the standard raptor/fountain code in

Sections 2.3.4, 2.3.7. With our new constructions, we provide a means to effectively achieve useful non-extreme points on the tradeoff curve between repair complexity and overhead.

- **Systematic fountain codes with efficient encoding/decoding:** A systematic code has the defining property that the uncoded source symbols appear as a subset of the coded symbols. The many practical reasons for its utility are motivated in [7]. Shokrollahi also gave a construction that modifies a raptor code into a systematic version in [7]. However, this construction involves a step during encoding which has linear complexity per symbol. In Section 2.4, we propose a new method to construct systematic fountain codes based on the results of Section 2.3. This provides a technique to avoid the linear encoding complexity of systematic raptor codes by taking on a negligible sub-optimality in terms of the overhead.

- **Broadcasting with side information among receivers:** We consider the problem of broadcasting to an audience of receivers when the receivers are in possession of unknown subsets of the source data as side information. The perfect information version of this problem is called index coding, which has been shown to be equivalent to an intractable rank minimization problem. We study the analysis and design of fountain codes for this problem with corresponding guarantees and lower bounds on the optimal code. We then also propose a solution based on systematic fountain codes that avoids the lower bound, which in turn can be obtained from Section 2.4. The use of fountain codes for the side information problem with incomplete information has also been considered by multiple independent works [60, 61], but to the best of our knowledge, our work is the first to provide results involving explicit bounds on the overhead.

- **Broadcasting a fountain code over multiple hops:** Coding in an erasure network while treating each link as point-to-point involves decoding and re-encoding entire message blocks at each node. This is not a scalable solution, especially in the case of fountain code, which requires relatively large block lengths for effectiveness. In Section 2.6, we devise a scheme that imitates the characteristics of a fountain code end-to-end across multiple hops, without having to decode and re-encode at every step.

2.3 Fountain Codes for Repair in Storage Problems

While the benefits of erasure coding for storage are well understood with regard to the savings on the number of storage units for a given level of redundancy, other important aspects that favor replication as opposed to coding are not as well understood. The most significant of these issues [62] is the latency related to recovery from failures. Another related issue that has been pointed out is the large state dependency metric, which refers to the need to contact a potentially large number of servers/storage units for each recovery. An efficient decoding/encoding algorithm does not, by itself, provide an efficient recovery algorithm because decoding efficiency in the context of codes designed for communication has a basic assumption of recovering the entire message block.

Even though the storage gains of coding over replication are extremely large (by a factor that scales with the number of message blocks being coded), storage cost itself could be less important than other factors like repair latency. As long as a code delivers the order of magnitude level improvements related to storage over replication, the need for strict information theoretic optimality of storage overhead of the code is less significant than the ability to repair destroyed code symbols efficiently.

Another issue where fixed rate code designs face an obstacle unlike replication is related to the flexibility of adding or deleting storage units dynamically [63]. When a code is designed with a fixed rate beforehand, adding or deleting code symbols no longer keeps the guarantees that were provided for the original design. Modern data centers use a large number of commodity components that are less reliable than customized hardware to achieve reliability [62].

2.3.1 Repair Complexity

The repair complexity of a code refers to the complexity of operations required to reconstruct subsets of its code symbols using other code symbols.

More precisely, let C be an erasure code with its symbols indexed by integers from $[n]$. For some $D \subseteq [n]$ (that is destroyed) and another (disjoint) subset $R \subseteq [n]$ (the recovery set), a repair algorithm then reconstructs the code symbols indexed in D using those from R, when feasible. Typically, one may consider $R = [n]/D$. The repair complexity for a given reconstruction instance is then defined to be the average (over the symbols in D) number of symbol operations performed by the repair algorithm.

$$\mathcal{R}(D) = \frac{\text{\# of symbol operations in repairing } D}{|D|} \tag{2.1}$$

A code on source data divided into k symbols is said to have an overhead δ if one has the guarantee that any $k(1+\delta)$ code symbols are sufficient to reconstruct the source data. When $\delta = 0$, we have an optimal code with respect to its overhead. The ease of repair is closely related to the overhead. Loosely speaking, the closer to optimal the overhead is, the harder it gets to repair the code. This tradeoff has been investigated in prior work extensively for a broadly related problem called the repair bandwidth minimization problem. However, our work has some fundamental differences with this line of work as explained below.

2.3.2 Distinction from the Bandwidth Minimization Problem

There has been substantial work on the problem of optimizing the repair bandwidth against the overhead (e.g. [42, 43] and the references therein). There are, however, two fundamental differences with the notion of repair complexity considered here. (1) In this dissertation, we consider a code symbol as a single atomic unit of memory which can not be split further. In [42], each node stores data that can be split into subunits and the goal is to optimize the bandwidth across the nodes. In other words, any operational complexity within a node is not a part of the optimization. (2) Our goal of optimizing the number of symbol operations required for reconstruction is more closely aligned to disk access than the bandwidth communicated across nodes itself. For bandwidth optimization [42], the codes designed are expected to download minuscule uniform amounts of compressed/coded data from each node over a large number of nodes to save on the final bandwidth, whereas the total number of symbol operations performed prior to communication could be quite expensive due to large number of symbols accessed.

2.3.3 Repair Complexity of Examples

1. **Parity Code:** A parity code of dimension k is $\{v \in \{0,1\}^{k+1} : v_{k+1} = \sum_{i=1}^{k} v_i\}$. Let $D = \{i\}$ for some $i \in [k+1]$ and $R = \{[k+1]/D\}$. The repair is performed by the relation $v_i = \sum_{j \neq i} v_j$, which involves k symbol operations. So the repair complexity is k.

2. **Repetition Code:** Let the code be $\{v \in \{0,1\}^{kl} : v = [\underbrace{v^* \ v^* \dots v^*}_{l \text{ times}}]\}$ for some $v^* \in \{0,1\}^k$. Let $D \subseteq [kl]$ such that the number of indices in D which are equivalent modulo k is at most $l-1$. To repair D, we need to perform at most $|D|$ symbol operations in copying the corresponding symbol from R, implying a repair complexity of 1. But the overhead for this code is $\delta = l\frac{k-1}{k} + \frac{1}{k} - 1$, which is arbitrarily bad for large values of l.

3. **Rateless Codes - Random Linear Codes (RLC), LT Codes, Raptor Codes:** Consider a rateless code with dimension k, i.e. the source message consists of k symbols whose encoding/decoding complexities are on average α and β per symbol respectively and the overhead is δ. For RLC, $\alpha = \beta = \theta(k), \delta = o(k)$; for LT codes, $\alpha = \beta = O(\log k), \delta = o(k)$; and for raptor codes, $\alpha = \beta = O(1)$ and $\delta = \epsilon$, a small positive number. Consider a recovery set R with $|R| = k(1+\delta)$. A natural repair algorithm is: (1) Decode the source symbols, S from R. (2) Encode the missing code symbols D using S. Under the given assumptions, step 1 incurs βk symbol operations and step 2 incurs $\alpha|D|$ symbol operations. Therefore, the repair complexity for D is:
$$\frac{\beta k + \alpha|D|}{|D|} = \beta\frac{k}{|D|} + \alpha$$
This could be efficient if $|D| \approx k$, but when D is small, even constant encoding/decoding complexities do not provide a corresponding efficient repair guarantee.

The examples above represent extremes on the repair complexity/overhead tradeoff. In what follows, we will propose new code designs to achieve other non-extreme points on this tradeoff.

2.3.4 The Augmented LT Code

We now propose the augmented LT code, which includes the source data concatenated with degree distribution based code symbols. Formally, let $S = \{s_1, \ldots, s_k\}$ represent the data to be stored, divided into fragments which are also called source symbols. Let $c_1, c_2, \ldots$ represent an LT coded stream generated on S with degree distribution Ω on $[k]$. The augmented LT code is defined as the rateless code formed by adjoining the uncoded source symbols to the LT coded stream, i.e. $\{s_1, \ldots, s_k, c_1, c_2, \ldots\}$.

2.3.5 Repair Algorithm for the Augmented LT Code

Let D be the set of indices of the code symbols to be repaired and R be a recovery set. Let $R_C = R \bigcap \{c_1, c_2, \ldots\}$ and $R_S = R \bigcap S$ (so $R = R_C \bigcup R_S$). Let $|D| = t$ and denote $D_S = D \bigcap S$ and $D_C = D/S = D \bigcap \{c_1, \ldots\}$. Consider the case when $|R_C| \geq k(1+\epsilon)$ where $\epsilon > 0$ and k is large. The repair algorithm for D from R is given below.

1. Since $|R_C| \geq k(1+\epsilon)$, w.h.p. it is feasible to iteratively decode S from R_C. Let $s_{\pi(1)}, s_{\pi(2)}, \ldots s_{\pi(k)}$ be a sequence in which the source symbols could be decoded from R_C. Let $c_{\xi(1)}, \ldots, c_{\xi(k)}$ be the corresponding sequence in which code symbols from R_C are processed from the gross ripple during the decoding process. π and ξ can be computed without performing any symbol operations by processing the packet headers or the random number seed generator used for the code construction. We have:
$$s_{\pi(i)} = c_{\xi(i)} \oplus \sum_{j \in S_i} s_j \tag{2.2}$$
where $S_i \subseteq \{\pi(1), \ldots, \pi(i-1)\}$ and $|S_i| = deg(c_{\xi(i)}) - 1$.

2. Repair symbols from D_S in the order of their appearance in π, using Equation (2.2). To rephrase, recover the symbols in D_S in the sequence $(s_{\pi(i_1)}, s_{\pi(i_2)}, \ldots, s_{\pi(i_t)})$ where $i_1, \ldots, i_t$ are ascending. This is feasible

because for each i, the entire set S_i would have been repaired by the time it is used in Equation (2.2).

3. Recover D_C by a standard Ω-degree distribution encoder.

2.3.6 Properties of the Augmented LT Code

Lemma 1. *The expected repair complexity for arbitrary sets D is at most $(1+\epsilon)\Omega'(1)$ while the overhead δ is at most $1+\epsilon$.*

Proof of Lemma 1. Let $|D_S| = t$. The number of symbol operations in restoring $s_{\pi(i)}$ is equal to $deg(c_{\xi(i)})$ from Equation 2.2. Since the sequence π is uniform over all possibilities when averaged over the randomness of the code, $c_{\xi(i_1)}, \ldots, c_{\xi(i_t)}$ has to be a uniform subset from $c_{\xi(1)}, \ldots, c_{\xi(k)}$, which are themselves a subset of the set $c_1, \ldots, c_{k(1+\epsilon)}$. Therefore,

$$\begin{aligned}
\mathbf{E}\sum_{j=1}^{t}[deg(c_{\xi(i_j)})] &= t\,\mathbf{E}[deg(c_{\xi(i_1)})] \\
&= \frac{t}{k}\sum_{j=1}^{k}\mathbf{E}[deg(c_{\xi(j)})] \\
&\leq \frac{t}{k}\sum_{j=1}^{k(1+\epsilon)}\mathbf{E}[deg(c_j)] \\
&= t(1+\epsilon)\Omega'(1)
\end{aligned}$$

The number of symbol operations in the rest of the reconstruction is $|D_C|\Omega'(1)$. Therefore, the repair complexity is at most

$$\frac{t(1+\epsilon)\Omega'(1) + |D_C|\Omega'(1)}{t + |D_C|} \leq (1+\epsilon)\Omega'(1)$$

The overhead of the augmented LT code is $\delta \leq 1+\epsilon$ because any set of $2+\epsilon$ surviving symbols ensures at least $1+\epsilon$ surviving symbols from the set $\{c_1, c_2, \ldots\}$, using which iterative decodability is guaranteed. □

The overhead is suboptimal for the above code, but it achieves a tradeoff between the extremes of bad repair complexity and bad overhead. To further reduce the overhead achieved with the augmented LT code, we next propose another design called the augmented raptor code.

2.3.7 Augmented Raptor Codes

Let $\epsilon > 0$ be small. Consider a (high rate) precode with code symbols $S_I = \{s_1, \dots, s_{k(1+\epsilon)}\}$ such that any subset of k symbols from S_I can recover the message. Let Ω be a degree distribution on $[k(1+\epsilon)]$. Let $c_1, c_2, \dots$ be a fountain coded stream generated by using Ω on S_I. The augmented raptor code is defined as the rateless code formed by adjoining S_I to this fountain coded stream, i.e. $\{s_1, \dots, s_{k(1+\epsilon)}, c_1, c_2, \dots\}$. The next section is on the overhead properties of this code, which is necessary for establishing the repair complexity claims.

2.3.8 Overhead Properties of Augmented Raptor Codes

Overhead is the smallest $\delta > 0$ such that an arbitrary set, R, of $k(1+\delta)$ code symbols can recover the source symbols. We first briefly recall the recovery constraint implied by [26, 39]. See also Section 2.5.1 for more details.

Proposition 2. *[26, 39] A total of rk Ω-coded symbols on k input symbols can be used to decode θk input symbols by iterative decoding if (for large k and w.h.p.)*

$$r\Omega'(t) + \log(1-t) > 0 \quad \forall \;\; t \in (0, \theta)$$

Let $R_C \triangleq R \bigcap \{c_1, c_2, \dots\}$ and $R_S \triangleq R \bigcap S_I = R/R_C$. Since $|R| = k(1+\delta)$, let $|R_C| = k(1+\delta)\alpha$ and $|R_S| = k(1+\delta)(1-\alpha)$ for some $\alpha \in [0,1]$. We have

$$\begin{aligned} |S_I/R_S| &= k(1+\epsilon) - k(1+\delta)(1-\alpha) \\ &= k(\alpha(1+\delta) - (\delta - \epsilon)) \end{aligned}$$

Define:

$$\lambda \triangleq \frac{|S_I/R_S|}{|S_I|} \tag{2.3}$$

$$= \frac{\alpha(1+\delta) - (\delta-\epsilon)}{1+\epsilon} \tag{2.4}$$

The projected code: Since the code symbols from R_S are essentially decoded to begin with, it is useful to view the code symbols from R_C as being generated by a degree distribution, $\Psi(x)$, on S_I/R_S. The "projected code" has $k(1+\delta)\alpha$ code symbols with degree distribution Ψ on a set of $|S_I/R_S| = k(\alpha(1+\delta)-(\delta-\epsilon))$ input symbols. Since the degree distribution of the code symbols in R_C with respect to S_I is $\Omega(x)$, the degree distribution projected onto S_I/R_S is given by $\Psi(x) \triangleq \Omega(1-\lambda+\lambda x)$ (see Corollary 5). Denote

$$f = \frac{\delta}{\epsilon} - 1 \tag{2.5}$$

and

$$M = \frac{1}{\epsilon} + 1 \tag{2.6}$$

A sufficient condition for recovery is to decode at least k total symbols from S_I. This translates to a recovery requirement of $k - |R_S| = k(1-(1+\delta)(1-\alpha)$ symbols. A sufficient condition for recovery can be obtained by expressing the recovery requirement and overhead available as fractions of the number of input symbols ($|S_I/R_S|$) for the projected code. These are:

Recovery fraction requirement:

$$\frac{1-(1+\delta)(1-\alpha)}{\alpha(1+\delta)-(\delta-\epsilon)} = 1 - \frac{1}{M\lambda}$$

Code symbols fraction available:

$$\frac{(1+\delta)\alpha}{\alpha(1+\delta)-(\delta-\epsilon)} = 1 + \frac{f}{M\lambda}$$

The recovery constraint (Proposition 2) now becomes:

$$\forall \lambda \in \left(\frac{1}{M}, 1\right),$$

$$\left(1+\frac{f}{M\lambda}\right)\Psi'(t) + \log(1-t) > 0 \quad \forall \;\; t \in \left(0, 1-\frac{1}{M\lambda}\right) \tag{2.7}$$

Note that f maps to an overhead $\delta = \frac{f+1}{M-1}$ by Equations (2.5) and (2.6). To rephrase, we are therefore interested in the smallest $f > 0$, for which we can design Ω satisfying the following constraint:

$$\forall \lambda \in \left(\frac{1}{M}, 1\right),$$

$$\left(\lambda + \frac{f}{M}\right)\Omega'(1-\lambda+\lambda t) + \log(1-t) > 0 \quad \forall \;\; t \in \left(0, 1 - \frac{1}{M\lambda}\right) \tag{2.8}$$

Let $\Omega(t) = \sum_i p_i t^i$. Since $1-\lambda+\lambda t \le \frac{1}{M} \;\; \forall t \in \left(0, 1-\frac{1}{\lambda M}\right)$, it follows that $i(1-\lambda+\lambda t)^i$ is dominated by $M(1-\lambda+\lambda t)^M \quad \forall \;\; i > M$. So it suffices to restrict attention to degree distributions with $p_i = 0 \;\; \forall \;\; i > M$ to search over distributions to minimize f. Given $\mathcal{P}$, any probability distribution with support on integers less than M and $\lambda \in (\frac{1}{M}, 1)$ and $t \in (0, 1-\frac{1}{M\lambda})$, define:

$$\mathcal{D}(M, \mathcal{P}, \lambda, t) \triangleq \frac{M}{M-1}\left(\frac{-\log(1-t)}{\sum_{i=1}^{M} i(1-\lambda+\lambda t)^{i-1} p_i} - \lambda\right) + \frac{1}{M-1} \tag{2.9}$$

Then, we can achieve an overhead δ that is arbitrarily close to δ_{opt} defined as:

$$\delta_{opt} = \inf_{\substack{M>1 \\ \mathcal{P}}} \sup_{\substack{\lambda \in (\frac{1}{M}, 1) \\ t \in (0, 1-\frac{1}{M\lambda})}} \mathcal{D}(M, \mathcal{P}, \lambda, t) \tag{2.10}$$

It will also be useful to consider the "overhead profile" for each fixed degree distribution ($\mathcal{P}$) and precode (M), which is the overhead that is necessary as a function of λ. More precisely, let:

$$\delta(\lambda, \mathcal{P}, M) \triangleq \sup_{t \in (0, 1-\frac{1}{M\lambda})} \mathcal{D}(M, \mathcal{P}, \lambda, t) \tag{2.11}$$

Although the overhead of a given design is captured by $\sup_{\lambda \in \left(\frac{1}{M}, 1\right)} \delta(\lambda, \mathcal{P}, M)$, it will be of interest to consider the entire profile as a function of λ if we need to optimize the design with assumptions on λ (which translates to assumptions on the relative composition of the recovery set between S_I and $\{c_1, c_2, \ldots\}$, captured by the parameter $\alpha \in [0, 1]$).

Theorem 3. *Let $\mu > 0$ and $D > M$. Let Ω be the same distribution as used in raptor codes [7]:*

$$\Omega(x) = \frac{1}{\mu+1}\left(\mu x + \sum_{i=1}^{D} \frac{x^i}{i(i-1)} + \frac{x^{D+1}}{D}\right)$$

Then, we have the following upper bound on the overhead profile:

$$\delta(\lambda, \Omega, M) < \frac{1}{M-1} + \frac{M}{M-1}\left(\frac{(1+\mu)\log(M\lambda)}{\mu + \log M} - \lambda\right) \tag{2.12}$$

Proof of Theorem 3.

$$\Omega(x) = \frac{1}{\mu+1}\left(\mu x + \sum_{i=1}^{D} \frac{x^i}{i(i-1)} + \frac{x^{D+1}}{D}\right) \tag{2.13}$$

$$\Omega'(x) = \frac{1}{\mu+1}\left(\mu - \log(1-x) + x^D - \sum_{d=D+1}^{\infty} \frac{x^d}{d}\right)$$

$$(\mu+1)\Omega'(1-\lambda+\lambda x) = \mu - \log\lambda - \log(1-x) + \\ +(1-\lambda+\lambda x)^D - \sum_{d=D+1}^{\infty} \frac{(1-\lambda+\lambda x)^d}{d}$$

First consider the following lower bound on one of the terms:

$$\begin{aligned}
&(1-\lambda+\lambda x)^D - \sum_{d=D+1}^{\infty} \frac{(1-\lambda+\lambda x)^d}{d} \\
&= (1-\lambda+\lambda x)^D \left(1 - \sum_{D+1}^{\infty} \frac{(1-\lambda+\lambda x)^{d-D}}{d}\right) \\
&\geq (1-\lambda+\lambda x)^D \left(1 - \frac{1}{D+1}\sum_{t=1}^{\infty}(1-\lambda+\lambda x)^t\right) \\
&= (1-\lambda+\lambda x)^D \left(1 - \frac{1-\lambda+\lambda x}{(D+1)\lambda(1-x)}\right) \\
&= (1-\lambda+\lambda x)^D \left(1 + \frac{1}{D+1} - \frac{1}{(D+1)\lambda(1-x)}\right) \\
&\geq (1-\lambda+\lambda x)^D \left(1 - \frac{M}{D}\right) \quad \because x \in \left(0, 1 - \frac{1}{M\lambda}\right)
\end{aligned}$$

The sufficient condition becomes

$$\forall \;\; \lambda \in \left(\frac{1}{M}, 1\right), \quad \forall \;\; x \in \left(0, 1 - \frac{1}{M\lambda}\right)$$

$$\left(\lambda+\frac{f}{M}\right)\left(\mu-\log\lambda-\log(1-x)+(1-\lambda+\lambda x)^D\left(1-\frac{M}{D}\right)\right)$$
$$>-(1+\mu)\log(1-x)$$

which is equivalent to:

$$\mu+(1-\lambda+\lambda x)^D\left(1-\frac{M}{D}\right)>-\log(1-x)\left(\frac{1+\mu}{\lambda+\frac{f}{M}}-1\right)-\log\lambda \quad (2.14)$$

Both LHS and RHS are increasing in x, so this would be implied (although this could be far from being necessary) by choosing $D \geq M$ in what follows:

$$\mu>\log(M\lambda)\left(\frac{1+\mu}{\lambda+\frac{f}{M}}-1\right)-\log\lambda \quad (2.15)$$

which is equivalent to

$$\mu+\log M>\log(M\lambda)\frac{1+\mu}{\lambda+\frac{f}{M}} \quad (2.16)$$

which is equivalent to

$$\frac{f}{M}>\frac{(1+\mu)\log(M\lambda)}{\mu+\log M}-\lambda \quad (2.17)$$

Equivalently, this shows that the Ω specified above already achieves an overhead profile $\delta(\lambda)$ equal to:

$$\delta(\lambda)=\frac{1}{M-1}+\frac{M}{M-1}\left(\frac{(1+\mu)\log(M\lambda)}{\mu+\log M}-\lambda\right) \quad (2.18)$$

The actual overhead profile achieved will be strictly better than the bound because of a gap between inequalities (2.14) and (2.15). □

Figure 2.1 shows a plot of this bound corresponding to a specific choice of the parameters.

We now revisit Equation (2.10) to numerically optimize the overhead. Consider any M and $\delta^* > 0$. We see that an overhead $\delta = \frac{M}{M-1}\delta^* + \frac{1}{M-1}$ is feasible iff the following LP (with variables $p_i,\ \ i \in [M]$) has a feasible solution for

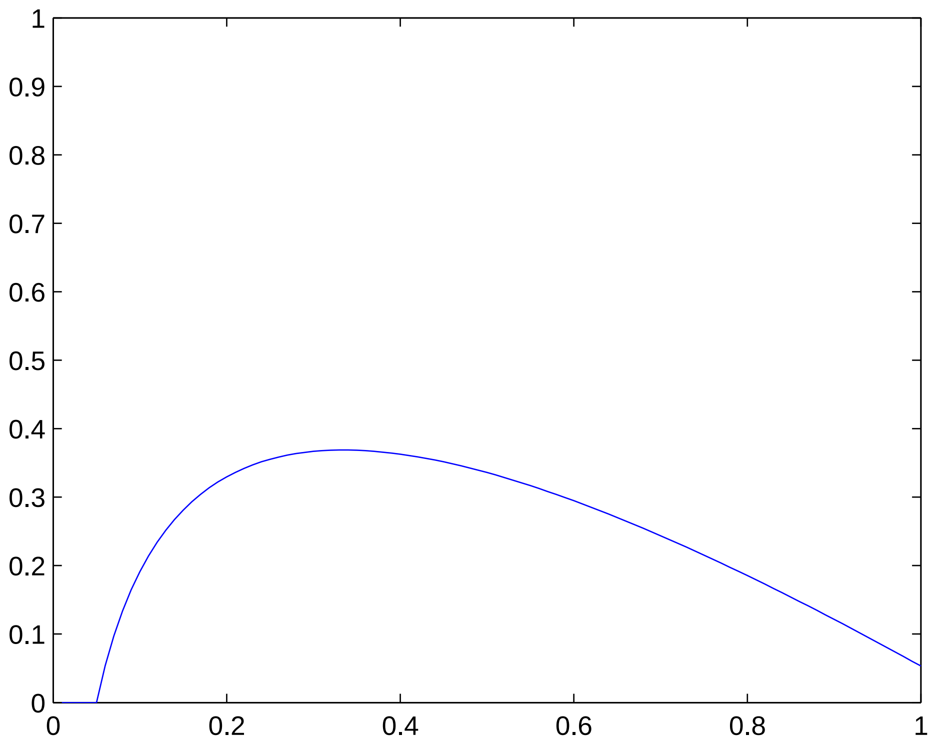


Figure 2.1: A plot of the bound on overhead profile in Equation (2.12) obtained for $M = 20, \mu = 0.001$.

some M.

$$\forall \lambda \in (\frac{1}{M}, 1), t \in (0, 1 - \frac{1}{M\lambda}) \tag{2.19}$$

$$(\delta^* + \lambda) \sum_{i=1}^{M} i(1 - \lambda + \lambda t)^{i-1} p_i \geq -\log(1 - t) \tag{2.20}$$

$$\sum_{i=1}^{M} p_i = 1 \quad \text{and} \quad p_i \geq 0 \;\; \forall \;\; 1 \leq i \leq M \tag{2.21}$$

We can approximate this system by choosing a fine grid for the parameters λ, t to obtain an LP that has a finite number of constraints. This LP can be solved to obtain a candidate degree distribution, whose overhead profile can then be explicitly evaluated numerically using Equation (2.11) (which is, of course, unconditional on the approximations made while obtaining the candidate degree distribution itself). Figure 2.2 is a plot of some feasible overhead profiles obtained by optimizing the degree distributions while minimizing the average degree on a support of $M = 20$.

We note from Figure 2.2 that it is possible to design degree distributions on a support of $M = 20$ that provide an overhead guarantee of at most than $\delta = 0.25$ across all ranges of λ. The reason we plot green and red curves in Figure 2.2 even though they have ranges of λ which perform worse than the blue curve, is to show that it is possible to further optimize the overhead guarantees, if one were to assume restrictions on λ. In Figure 2.2, the green overhead profile was obtained for a distribution that was optimized for $\lambda \in (0, 0.5)$ whereas the red curve corresponds to $\lambda \in (0, 0.2)$. Recall that λ corresponds to the fraction of symbols in S_I that need to be repaired. If, for example, the designer is confident that the systematic part will never have more than 50% of symbols destroyed, one may use the degree distribution represented by the green curve in its design.

2.3.9 Repair Algorithm for Augmented Raptor Codes

Let D be the set of code symbols to be repaired and let R be a recovery set. Let $D_S = D \bigcap S_I$ and $D_C = D \bigcap \{c_1, \ldots\}$. Let $R = R_S \bigcup R_C$ where $R_S = S_I / D_S$ and $R_C = R / R_S$. There are three steps: (1) Repair $D^* \subseteq D_S$ such that $|D_S / D^*| < \epsilon k$ (if $|D_S| \geq k\epsilon$). This can be accomplished using

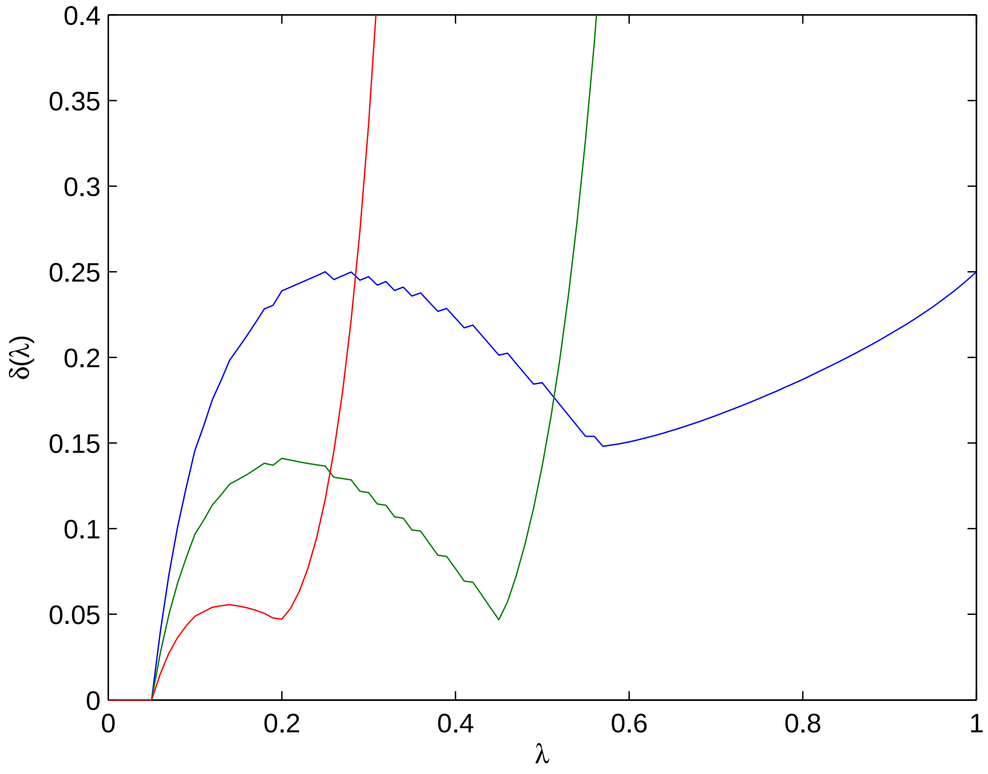


Figure 2.2: A plot of three overhead profiles corresponding to three different candidate degree distributions that were obtained by optimizing the average degree over discretized versions of LP constraints (2.19). The green and red profiles are for distributions that were specifically optimized to minimize the maximum overhead attained for the ranges $\lambda \in (0, 0.5)$ and $\lambda \in (0, 0.2)$ respectively.

an iterative repair procedure similar to the augmented LT code. (2) Repair D_S/D^* using a repair algorithm for the precode. (3) Repair D_C using an Ω-degree distribution encoder.

To ensure that Step 1 is feasible, R_C is assumed to be a set of random code symbols (of the smallest size, for repair complexity claims) that assures the recovery of at least k total symbols from S_I w.h.p. This in turn depends on the size of R_S which determines the λ in Equation (2.4). Let $\delta(\lambda)$ denote the overhead profile for the code used. We have:

$$|R_C| = (1+\delta(\lambda))\alpha \tag{2.22}$$

2.3.10 Repair Complexity for Augmented Raptor Codes

To establish the repair complexity, we need to evaluate the number of symbol operations involved in Step 1 while repairing D^*. Let $\delta(\lambda)$ be the overhead profile for the given degree distribution (Section 2.3.8). The expected number of symbol operations involved in the repair is at most $\Omega'(1)|R_C|$. Therefore, the expected per symbol repair complexity while reconstructing D^* is:

$$\begin{aligned}\frac{\Omega'(1)|R_C|}{|D^*|} &= \frac{k(1+\delta(\lambda))\alpha}{k(1-(1+\delta(\lambda))(1-\alpha))} && (2.23)\\ &= 1+\frac{\delta(\lambda)}{(1+\delta(\lambda))\alpha-\delta(\lambda)} && (2.24)\\ &= 1+\frac{M-1}{M}\frac{\delta(\lambda)}{\lambda-\frac{1}{M}} \text{ using relations (2.4), (2.6)} && (2.25)\end{aligned}$$

For example, using the bound at Equation (2.12) for the raptor code distribution Ω, we can now establish a constant bound on the repair complexity. For any $\lambda > \frac{1}{M}$, we have the following claim on the derivative of the bound on the overhead profile with respect to λ:

$$\begin{aligned}\delta'(\lambda) &= \frac{M}{M-1}\left(\frac{1+\mu}{(\mu+\log M)\lambda}-1\right)\\ &\leq \frac{M}{M-1}\left(\frac{(1+\mu)M}{\mu+\log M}-1\right)\end{aligned}$$

Since $\delta(\frac{1}{M}) = 0$, this implies $\forall \lambda \in (\frac{1}{M}, 1)$:

$$1 + \frac{M}{M-1} \frac{\delta(\lambda)}{\lambda - \frac{1}{M}} \leq \frac{(1+\mu)M}{\mu + \log M}$$

Using Equation (2.23), we now conclude that the per symbol repair complexity for any D^* is at most $\frac{(1+\mu)M}{\mu+\log M}$. As discussed earlier, if we assume a bound for λ's range of interest, we would be able to further optimize the repair complexity guarantee using a corresponding optimization in the overhead profile as shown in Figure 2.2.

2.4 Systematic Rateless Codes with Low Complexity

A systematic code is defined as a code in which the source symbols appear as a subset of the code symbols. It is desirable for a code to be systematic in many applications. We now briefly outline the construction of systematic raptor codes proposed by Shokrollahi in [7]. Let $x_1, \ldots, x_k$ be the original source symbols. In the first step of encoding, the source symbols are transformed into *intermediate symbols*, $y_1, \ldots, y_k$. The raptor code is now constructed by taking $y_1, \ldots, y_k$ as the source symbols. The intermediate symbols are computed by solving for the system of linear equations which sets the k source symbols to match some subset of k positions in the first $k(1+\epsilon)$ generated code symbols, where $\epsilon > 0$ is small. However, this construction has a bottleneck step in the encoding process, which corresponds to an inverse matrix multiplication (Step 1 of Algorithm 11 in [7]) and consequently has a linear per symbol encoding complexity. The systematic versions of LT codes proposed in [7] avoid this step, but they do not enjoy the constant per symbol encoding/decoding complexity advantage of raptor codes.

2.4.1 A Systematic Construction Based on Augmented Raptor Codes

Consider the augmented raptor codes we proposed in Section 2.3.7 where the precode selected to compute $\{s_1, \ldots, s_{k(1+\epsilon)}\}$ is chosen to be a systematic fixed rate erasure code with constant encoding/decoding complexity (e.g. [26]). We can now choose the degree distribution Ω to optimize the overhead properties as outlined in Section 2.3.8. This provides a rateless code with constant encoding/decoding complexity with an overhead penalty of less than 25%. This is a useful tradeoff in any situation when the reduction in encoding complexity (from linear per symbol to constant per symbol) is attractive in comparison to the additional bandwidth cost due to extra overhead (which can be guaranteed to be at most 25% and can be reduced further if a guarantee can be provided on the fraction of correctly transmitted symbols within the systematic part).

2.5 Broadcasting with Side Information

In this section, we consider the problem of designing codes to multicast data from a source to receivers that possess arbitrary subsets of the data a priori as *side information*. We formulate and study an optimization problem over degree distributions to minimize the overhead necessary for complete decoding, and prove that: (i) Degree distributions converging to the standard soliton distribution cannot exploit side information in terms of the overhead necessary for complete decoding. (ii) An asymptotic *shifted* soliton distribution achieves an overhead which is within a constant factor (< 2) of the optimal overhead. (iii) There exist no degree distributions which achieve asymptotically optimal overhead for any non-trivial constant fraction of the data as side information. We then propose a solution using the systematic versions of raptor codes designed in Section 2.4 to which the computed lower bound does not apply.

The presence of side information at the receiver could arise from a previous incompletely decoded download session or from alternate data sources. The issue of designing codes which exploit the side information present at the receivers has been considered previously by a number of authors. Metzner [64] originally identified random linear coding as a useful approach to the context of designing an efficient broadcast retransmission protocol. More recently, Birk and Kol [65] have considered code design to minimize the number of transmissions, assuming that the encoder is provided with the complete side information pattern at the receivers. This problem is known as *index coding* in its form where each receiver requests a unique packet. Reference [66] showed that the optimal linear index code can be formulated as a rank minimization problem on finite field matrices. The unique requests constraint can also be generalized to arbitrary subsets of possibly non-disjoint requests, for which multicast is a practically important non-trivial subclass. However, conveying the side information at each receiver is a task that requires too much overhead in the form of ARQ. Further, processing the collected data to compute the optimal code is likely to be impractical because arbitrary rank minimization problems on finite field matrices are computationally intractable. Consequently, approaches that are oblivious to the precise pattern, and those that can also simultaneously deal with lossy transmissions, are useful.

2.5.1 Optimal Overhead with Side Information

Let $\lambda > 0$ represent a parameter which corresponds to each receiver having a subset of $(1-\lambda)k$ packets as side information. Assume that the decoder subtracts the symbols available as side information from each code symbols received to give a resulting code symbol, which we shall refer to as the "projected code." Let $\Omega^{(k)} = \{\Omega_1^{(k)}, \ldots, \Omega_k^{(k)}\}$ be a degree distribution with support on $[k]$ (i.e. corresponding to block length k), which is used at the encoder. Let $\Psi^{(k)}$ be the corresponding projection of $\Omega^{(k)}$ obtained on subset of λk unknown source symbols. The relation between them is (for $1 \le i \le \lambda k, 1 \le j \le k$):

$$\Psi_i^{(k)} = \sum_{j=i}^{k} \Omega_j^{(k)} \frac{\binom{k\lambda}{i}\binom{k(1-\lambda)}{j-i}}{\binom{k}{j}} \tag{2.26}$$

For small i and j independent of k, the term $\frac{\binom{k\lambda}{i}\binom{k(1-\lambda)}{j-i}}{\binom{k}{j}}$ is approximated by $\binom{j}{i}\lambda^i(1-\lambda)^{j-i}$, though this not an approximation in general. For LT code on block length k, the average degree is of the order $\log k$. Suppose as $k \to \infty$, the pointwise limits converge to valid probability distributions Ω and Ψ respectively on Z^+. The relation between the limiting distributions Ω and Ψ is given by the following relation, which follows by the dominated convergence theorem.

Lemma 4. *For limiting distributions Ω and Ψ formed as above, we have:*

$$\Psi_i = \sum_{j=i}^{\infty} \Omega_j \binom{j}{i} \lambda^i (1-\lambda)^{j-i} \tag{2.27}$$

We then have the following relation between the generating functions (defined as $\Omega(x) = \sum_i \Omega_i x^i$):

Corollary 5. *The relation between the generating functions Ψ and Ω is:*

$$\Psi(x) = \Omega(1 - \lambda + \lambda x)$$

Definition 1. *Let*

$$r_k = \frac{\textit{no. of coded packets received}}{\lambda k}$$

$$z_k = \frac{\textit{no. of unknown packets recovered}}{\lambda k}$$

When the limits exist, let r denote the limit of r_k and $s_\lambda(r, \Omega)$ denote the limit of z_k, where Ω is the generating function of the limit of the degree distributions used for coding.

For a sequence of distributions that converges to a distribution with generating function, $\Omega(x)$, when $\lambda = 1$, the recovered fraction (for a vanishingly small perturbed distribution) converges to [39]:[5]

$$s(r, \Omega) = \inf\{z \in [0, 1) : r\Omega'(z) + \log(1-z) < 0\} \wedge 1$$

When the decoder uses the side information of $(1 - \lambda k)$ source symbols, it decodes additional source symbols given by an asymptotic fraction (of λk) corresponding to $s_\lambda(r, \Omega) = s(r, \Psi)$. Thus:

Proposition 6. $s_\lambda(r, \Omega) = \inf\{z \in [0, 1) : r\lambda\Omega'(1-\lambda+\lambda z) + \log(1-z) < 0\} \wedge 1$

To define the overhead necessary for successful decoding of all source blocks, we are now interested in the following optimization problem for a

[5] Although the conditions stated do not apply verbatim here either, the justification for it comes from Lemma 1 of [67].

given $0 < \lambda \leq 1$:

$$r_\lambda^* = \min_\Omega r \tag{2.28}$$

$$\text{Subject to} : s_\lambda(r, \Omega) = 1 \tag{2.29}$$

Note that the optimal overhead is $r_\lambda^* - 1$. To contrast, the problem solved in Section 2.3 involves consideration of all values of λ simultaneously while requiring only partial recovery. In the current situation, we require complete recovery, but for a specific λ. A trivial lower bound for r_λ^* for any $0 < \lambda \leq 1$ is 1. We can also obtain an easy upper bound by ignoring the side information and using the soliton distribution. Since we know that asymptotically k packets suffice to recover all the n packets without even considering the side information, this achieves an $r_\lambda = \frac{k}{k\lambda} = \frac{1}{\lambda}$. This gives us:

$$1 \leq r_\lambda^* \leq \frac{1}{\lambda} \tag{2.30}$$

Clearly, this implies $r_1^* = 1$, which is attained by the capacity achieving soliton distribution used for LT codes [6]. Some natural questions arise for the problem under consideration: Is $r_\lambda^* = 1$ for $\lambda < 1$? (i.e., Can we have capacity achieving degree distributions for general λ?) How bad is the soliton distribution as a solution to the optimization problem in 2.28? (We know that it is no worse than the upper bound even after throwing away the side information, but could it actually achieve a better ratio because of side information?) If the soliton is bad, how do we design degree distributions that do well for $\lambda < 1$?

2.5.2 Performance of the Soliton Distribution

Proposition 7. *For the soliton distribution, a necessary condition for complete recovery is $r > 1/\lambda$. This means that side information gives no advantage for complete recovery.*

Proof. Consider a sequence of degree distributions that converge to the soliton distribution, whose generating function, $\Omega(x) = \sum_{i\geq 2} \frac{x^i}{i(i-1)}$. With $\Psi(z) = \Omega(1-\lambda+\lambda z)$, the recovered fraction z_n (of a vanishingly small perturbed distribution) converges to: $s(r, \Psi) \triangleq \inf\{z \in [0, 1) : r\Psi'(z) + \log(1-z) < 0\} \wedge 1$.

Consider:

$$\begin{aligned}
&r\Psi'(z) + \log(1-z) \\
&= r\frac{d}{dz}\Omega(1-\lambda+\lambda z) + \log(1-z) \\
&= r\lambda\frac{d}{dz}\left(\sum_{i\geq 2}\frac{(1-\lambda+\lambda z)^i}{i(i-1)}\right) + \log(1-z) \\
&= r\lambda\sum_{i\geq 1}\frac{(1-\lambda+\lambda z)^i}{i} + \log(1-z) \\
&= r\lambda|\log\lambda| + (r\lambda-1)|\log(1-z)|
\end{aligned}$$

From the above equation, it is clear that $r\Psi'(z) + \log(1-z) > 0 \ \ \forall z \in [0,1)$ iff $r\lambda > 1$. □

2.5.3 k-lifted Soliton Distribution

Consider the $k-$lifted soliton distribution defined by the following generating function (where k will be chosen later):

$$\Psi(x) = \sum_{i\geq k+1}\frac{k}{i(i-1)}x^i$$

Shifted distributions for the side information problem were also considered independently in [61] with different parameters (though they do not provide the theoretical guarantees given here).

Proposition 8. *For $k = \lfloor 1/1.82\lambda \rfloor$, the k-lifted soliton distribution requires at most $r = 1/(\lambda\lfloor 1/1.82\lambda\rfloor)$ for complete recovery.*

Proof. First we lower bound $Q'(z)$ for $z \in [0, 1)$.

$$\begin{aligned}
Q'(z) &= \sum_{i \geq k} \lambda k \frac{(1-\lambda+\lambda z)^i}{i} \\
&= \lambda k \left(-\log(\lambda - \lambda z) - \sum_{i=1}^{k-1} \frac{(1-\lambda+\lambda z)^i}{i} \right) \\
&\geq \lambda k \left(|\log \lambda| + |\log(1-z)| - H_{k-1} \right) \\
&\Big(\text{where } H_k \text{ is the } k^{th} \text{ Harmonic number} \Big) \\
&\geq \lambda k \Big(|\log \lambda| + |\log(1-z)| - \Big(\log k + \gamma + \frac{1}{2n + \frac{1}{3}} \Big) \Big) \\
&\text{(Using a bound from [68], where } \gamma \approx 0.578 \text{ is the Euler constant)} \\
&\geq \lambda k \left(|\log \lambda| - \log(1.82k) + |\log(1-z)| \right)
\end{aligned}$$

Given r, we thus have for $z \in [0, 1)$:

$$rQ'(z) + \log(1-z) \geq r\lambda k \left(|\log \lambda| - \log(1.82k) \right) + (r\lambda k - 1)|\log(1-z)|$$

The choice of $k = \lfloor 1/1.82\lambda \rfloor$ ensures that the first term above is non-negative. For such a k, the second term is also non-negative for all $z \in [0, 1)$ for the given choice of r, implying that $s(r, \Psi) = 1$, which assures asymptotically complete recovery. □

2.5.4 Lower Bounds on the Optimal Overhead

We now compute lower bounds for the optimal decoding overhead required for any degree distribution for a given fraction λ defining the side information. This shows that r^*_λ could be strictly greater than 1 for general λ, which was conjectured in [60]. This is done by considering an intermediate performance problem inspired by [67]. The optimization in Equation (2.28) can be rewritten as:

$$r^*_\lambda = \min_P r$$

Subject to ($\forall\ 0 \leq t < 1$):

$$\sum_{i \geq 1} r\lambda i p_i (1-\lambda+\lambda t)^{i-1} + \log(1-t) \geq 0 \tag{2.31}$$

Choose some large integer m and consider the following related problem (which can also be interpreted as an intermediate performance problem):

$$r_{\lambda,m} = \min_{P} r$$

subject to:

$$\sum_{i\geq 1} r\lambda i p_i (1-\lambda+\lambda t)^{i-1} + \log(1-t) \geq 0 \tag{2.32}$$

$$\forall \quad 0 \leq t < 1 - 1/\lambda(m+1)$$

We replaced the constraints in Equation (2.31) with a proper subset in Equation (2.32) and so

$$r_{\lambda,m} \leq r^*_\lambda \ \ \forall \ \ m$$

Further, it can be verified that:

$$\begin{aligned} i(1-\lambda+\lambda t)^{i-1} &< m(1-\lambda+\lambda t)^{m-1} \\ &\forall \ \ i > m, \ t \leq 1 - 1/\lambda(m+1) \end{aligned}$$

The above condition can be shown to imply that we can restrict attention to $p_i = 0 \ \forall \ i > m$ in the above optimization defining $r_{\lambda,m}$. Hence we obtain:

$$r_{\lambda,m} = \min_{P} r$$

Subject to:

$$\sum_{i=1}^{m} r\lambda p_i i (1-\lambda+\lambda t)^{i-1} + \log(1-t) \geq 0$$

$$\forall \ \ 0 \leq t < 1 - 1/\lambda(m+1)$$

With $a_i = rp_i$, we get the LP:

$$r_{\lambda,m} = \min_{A} \sum_{i=1}^{m} a_i$$

subject to:

$$\sum_{i=1}^{m} \lambda i (1 - \lambda + \lambda t)^{i-1} a_i \geq -\log(1-t)$$
$$\forall \;\; 0 \leq t < 1 - 1/\lambda(m+1)$$

The dual of the above LP can be written as :

$$\xi_{\lambda,m} = \max_{\mu} \int_{t \in [0, 1-1/\lambda(m+1))} -\log(1-t) d\mu(t)$$
$$\text{Subject to: } (\forall \; 1 \leq i \leq m)$$
$$\int_{t \in [0, 1-1/\lambda(m+1))} \lambda i (1 - \lambda + \lambda X)^{i-1} d\mu(t) \leq 1$$

where μ is a measure with support defined on $[0, 1 - 1/\lambda(m+1)]$. Further, any feasible solution above is a lower bound to

$$\xi_{\lambda,m} \leq r_{\lambda,m} \leq r^*_{\lambda}$$

Thus, we can optimize over a subclass to obtain the required lower bounds. One possibility is to take μ as a discrete distribution with finite support. In this case, the dual defined above becomes a finite dimensional linear program, for which we can use linear programming to compute the bound. We plot the bounds we have numerically computed using this approach in Figure 2.3 as a function of λ in the range $[0, 1]$. For most λ (except when it is very close to 1), a lower bound strictly greater than 1 is obtained.

2.5.5 A Solution Based on (Systematic) Augmented Raptor Codes

Consider the systematic augmented raptor codes proposed in Section 2.4.1. Rather than the LT coding based design which was analyzed in the prior sections, the above proposed code provides for a more naturally suited design

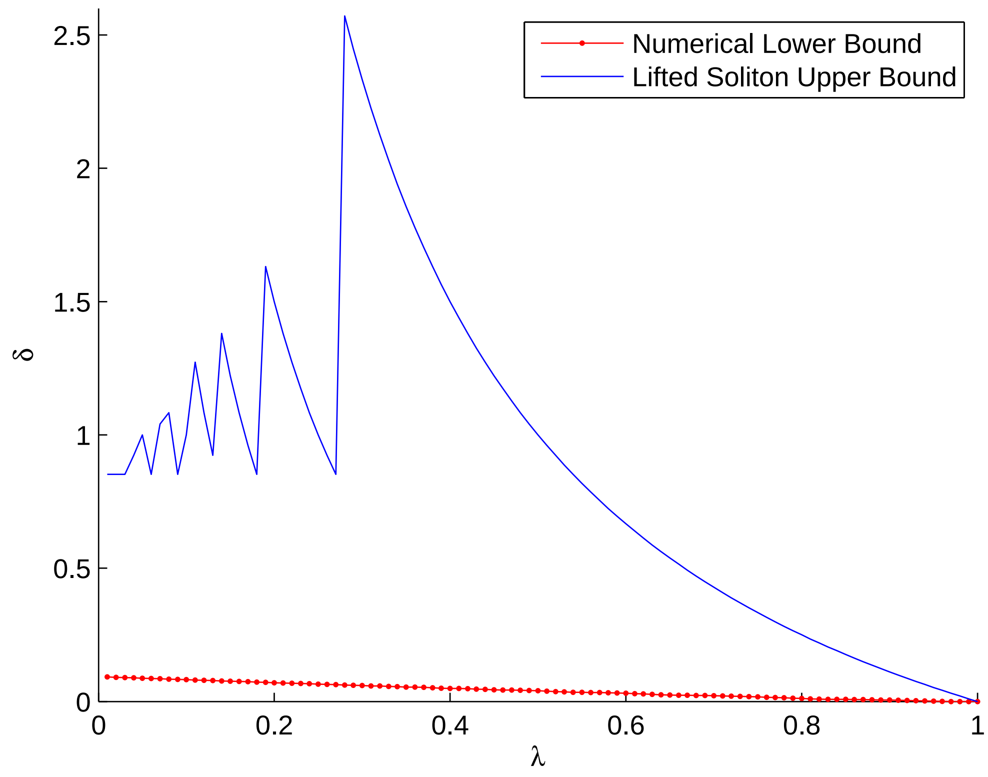


Figure 2.3: A plot of the numerically computed lowerbound along with the analytical upper bound (from Proposition 8) on the optimal overhead, $\delta^*_\lambda = r^*_\lambda - 1$.

for the side information case. To see this, consider a code scheme for the side information problem based on the systematic augmented raptor code, for which the code symbols are defined as the part of the code excluding the systematic part. The missing source symbols at each receiver can now be considered as erased symbols in the systematic augmented raptor code proposed. This implies that the overhead guarantees derived in Section 2.3.8 apply for this code along with the constant encoding/decoding complexity and ratelessness.

2.6 Broadcasting a Fountain Code over Multiple Hops

Figure 2.4 illustrates the problem being considered in this section. LT codes [6] work for a single erasure channel with a per symbol logarithmic complexity. As the block length $k \to \infty$, it was shown in [6] that a set of $k + O(\sqrt{k}\ln^2 \frac{k}{\delta})$ coded packets is sufficient to recover the k packets with a probability of at least $1 - \delta$ through the simple belief propagation decoder. $\delta > 0$ can be arbitrarily small, and both encoding and decoding have a complexity of $O(\log \frac{k}{\delta})$.

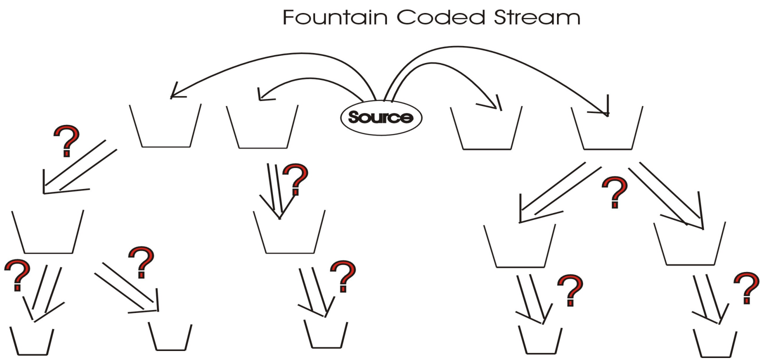


Figure 2.4: Illustration of the basic problem being considered. We are interested in devising *good* coding schemes over the links indicated by the question marks.

The only straightforward way to use a fountain code in a general network is to completely decode and re-encode the source message block at each intermediate node. Though this strategy might be considered a capacity achieving scheme in terms of the channel uses, it involves a large delay that compounds at each hop. There has also been work addressing the specific case of a line network of erasure channels in [69], in which some solutions were discussed that tradeoff between complexity, delay and adaptability. *Delay* is defined as the additional time it takes for the coding scheme to complete the transfer of all packets *beyond* what it would have taken if the throughput was precisely the min-cut capacity. This delay is analyzed in [70] as *overhead*, which is defined as $n-k$ where n is the total number of received coded symbols (packets) and k is the number of source symbols being encoded. Asymptotic throughput optimality in these cases should correspond to a sub-linear (in k) delay/overhead metric.

Is it possible to have the low complexity of LT codes and low *delay* while maintaining *ratelessness* and achieving a throughput equal to capacity for anything beyond a single erasure channel? In this chapter, we consider the above question, with two caveats: (1) We consider networks that can be represented as a tree of discrete memoryless erasure channels (DMC) of unknown erasure probabilities. However from here on, we describe the scheme for a line network of erasure DMCs (as in [69]). Due to the ratelessness, it is easy to apply the same to a tree network wherein each node broadcasts packets to all of its children. This easy extensibility to tree networks is another advantage of the ratelessness. (2) While the scheme is still rateless (assumes no estimate of the erasure probabilities and involves no feedback), it needs an estimate for some universal upper bound $0 \leq \alpha < 1$ on all erasure probabilities. A practical motivation for this scenario is a sequence of relay nodes with reasonably good, but unknown, erasure channels that drop packets at rates anything between say, 0 - 10% of the time. Most networks do not have erasure channels that are arbitrarily bad, hence it would not be too unreasonable to assume such an α for design. We first consider two toy problems to motivate the scheme proposed.

2.6.1 Online Encoding

Consider a set of k source symbols. By online encoding, we refer to an algorithm to generate a set of approximately k coded symbols that are statistically identical to a set of LT coded packets with the following restriction: *The i^{th} coded packet being generated should be a combination of only the first i information packets.* Using the same idea that lies behind the augmented

LT code repair, we describe the algorithm below:

Algorithm *SEQCODE*

(1) Generate a random set of $k+o(k)$ *symbols* according to the LT encoding process.

(2) Obtain a permutation π from the encoded symbols as follows:

   (a) Run an LT decoder on the encoded symbols.

   (b) If the index decoded at i^{th} iteration of the decoding process has a label j, then set $\pi(i) = j$.

(3) Now perform actual packet encoding using π as follows iteratively:

   (i) Pick a new coded symbol that includes index $\pi(i)$, and involves other indices of values $\pi(j)$ for $j \leq i$.

   (ii) Generate a coded output packet according to the rule given by the symbol obtained in step (i).

Proposition 9. *SEQCODE generates a set of $k + o(k)$ packets which are LT distributed in an* online fashion.

Proof. Consider packets indexed $\pi(1) \dots \pi(i)$. The fact that all these packets have been decoded by the i^{th} stage implies that there were at least i coded packets that were all combinations of $\pi(1) \dots \pi(i)$. In fact, the decoding process runs to completion implies that each new index generates at least one new symbol to encode. This implies that step 3-(i) will be successful. $\square$

2.6.2 Recoding Coded Packets at Intermediate Hops

Ignoring the real-time encoding aspect, one approach at intermediate nodes is to do concatenated coding. In other words, an intermediate node encodes packets that it receives, treating them as information packets themselves. The decoder could peel off the successive coding layers and hence will involve t successive instances of the LT decoding process at a node which is t hops away from the source. Consider a sequence of $n + 1$ nodes with source node labeled as node 0 that starts off with k message packets. Fix a sequence of block lengths, $k_0 \dots k_n$ to be specified later, for the $n+1$ nodes to perform the recoding. The lengths are defined with $k_0 = k$ and for each $1 \leq i \leq n$, k_i is set to ensure that collecting a total of k_i LT coded packets at node i is enough to recover the k_{i-1} packets that were recoded by node $i-1$, with high probability. However, for this coding to be optimal, we need to ensure that the asymptotic *overhead* incurred at each node does not accumulate over hops.

Asymptotic Overhead: By appropriately scaling the block length k, in terms of the number of nodes, n, it is possible to maintain cumulative rate optimality. For LT codes, a total of at least $k(1 + \frac{\log^2 \frac{k}{\delta}}{\sqrt{k}})$ packets gives an error probability of at most δ. On a line network of $n+1$ nodes for a union bound error probability of δ, fix a $\frac{\delta}{n}$ error at each intermediate node. For each i, we choose $k_{i+1} = k_i(1 + \frac{\log^2 \frac{k_i n}{\delta}}{\sqrt{k_i}})$ to ensure this, where $k_0 = k$. Since $\frac{\log^2 k}{\sqrt{k}}$ is a decreasing function of k, we can upper bound k_i as

$$k_i \leq k_0(1 + \frac{\log^2 \frac{kn}{\delta}}{\sqrt{k}})^i \tag{2.33}$$

Finite nodes: Clearly, for $i \leq n$ and n constant, Equation (2.33) implies $k_i \sim k$ as $k \to \infty$. The complexity of encoding at node i is the complexity of LT coding for block length $k_i(\sim k)$ which is $O(k \log k)$. Decoding complexity is $O(ik \log k) = O(k \log k)$ since it involves i successive instances of LT decoding.

Unlimited nodes($n \to \infty$): We now show that even for an arbitrary number of nodes, the overhead can be maintained asymptotically optimal. From Equation (2.33), we have $k_n = k_0(1 + \frac{\log^2 \frac{kn}{\delta}}{\sqrt{k}})^n$. Setting $k = \Omega(n^3)$, it can be verified that this bound is asymptotically similar to $k = k_0$. Hence, we can still maintain a negligible overhead even with an arbitrarily growing number of nodes, provided we scale the block length appropriately. Since $k_n \sim k_0$, the encoding complexity is still the same as before - $O(\log k)$ per symbol. The decoding complexity now depends on the relation between n and k and is given as $O(n \log k)$ (per symbol) at the final node. Since we require k to grow as at least $\Omega(n^3)$, this complexity is at most $O(k^{1/3} \log k)$ and can be further reduced by scaling k higher than n^3. This is again not restrictive, since block length usually is of much higher order compared to the number of nodes.

2.6.3 Heuristic "LT-Relay"

Assume slotted time and a line network of DMC erasure channels as in [69] with i^{th} erasure probability ϵ_i. Intermediate nodes can send a packet that results from coding operations that involve all the packets that have been received until and including the current time slot. Online encoding in Section 2.6.1 deals with an artificial situation of producing coded packets where exactly one new information packet is made available to the encoding node at each time slot. We need to devise a coding scheme when new packets are revealed based on a random process that represents erasures on the previous channel.

Fix a sequence of block lengths, $k_0 \dots k_n$, as defined in Section 2.6.2, with $k_0 = k$. Node 0 generates standard LT coded packets. The operations performed by the intermediate nodes will be divided into two well defined distinct phases called the online phase - which is roughly defined as while the node is still receiving useful packets from its predecessor - and the post-online phase. The main challenge is to construct a coding procedure for the online phase so as to ensure that the set of all packets generated during the online phase will be equivalent to an LT code. Beyond the online phase, the stream of coded packets can continue using standard LT coding. Note that the notion of a node's online phase does not involve its downstream nodes, and hence is feedback independent.

Code symbol. A code symbol is a set of indices that correspond to packet indices which will be summed (XOR-ed) to form a coded packet. For instance, one can say that LT coding is performed using a set of *code symbols* generated according to a random LT distribution. *Example of a code symbol:* The set $\{1, 3, 5\}$ represents a code symbol that specifies the rule of summing the first, third and fifth packets to form a coded symbol. Some definitions and examples are given next before describing the scheme being proposed.

Online code copy: An online code copy at node i is an ordered sequence of k_{i+1} code symbols that can be generated in an online fashion. *Example of a procedure to generate a random instance of an online code copy*: Consider an instance/realization of a set of k_{i+1} (random) LT coded symbols generated from the k_i indices, $1, 2, \dots, k_i$. Using the procedure $SEQCODE$ in Section 2.6.1, w.h.p., we compute a permutation π of $\{1, 2, \dots, k_i\}$ such that the number of code symbols containing exclusively the indices $\pi(1), \pi(2), \dots, \pi(i)$

is at least i. Now, consider the set of code symbols in the sequence in which they lend themselves to an online encoding. This set of code symbols taken in the sequence π is an instance of an online code copy.

Code matrix: A code matrix at node i is a $T(k) \times k_{i+1}$ random matrix of code symbols in which each row corresponds to an independently generated online code copy. The number of rows of the code matrix, $T(k) = k^{1+\delta}$ where $\delta > 0$ is a constant.

We assume below that erasure probabilities are strictly increasing down any path from the source. However, it is possible to imagine schemes (that can be ratelessly implemented on top of the main scheme) to artificially force monotonically worsening channels with equivalent min-cut capacity. This leaves a sequence of effective channels of equivalent min cut capacity, $\min_i\{1 - \epsilon_i\}$, with modified erasure probabilities, $\epsilon^{'}_i \approx \max_{1 \leq j \leq i} \epsilon_i$. This is because a good channel following a bad channel is redundant for the min-cut capacity.

Online phase: The online phase at node i is defined to be the period until the time slot at which node i collects a total of k_{i+1} coded packets. We further partition the online phase into $k_{i+1} + 1$ states, indexed through $0, 1, \ldots, k_{i+1}$, where the node is in state i when it has collected a total of i packets. Denote by $\{s_0, s_1, s_2 \ldots s_{k_{i+1}}\}$ the random variables representing the number of erasures seen in the corresponding states. The number of time slots spent in state j is thus $s_j + 1$ including the first time slot upon entering state j.

Note the distinction that we define the online phase to require collecting k_{i+1} packets, although node i needs only k_i packets to decode w.h.p. Also, node i uses only the first k_i packets it received for further encoding, even though the online phase state is maintained till k_{i+1}.

2.6.3.1 The Coding Scheme

We now describe the coding scheme at node i. It is implicit below that the coded packets received are indexed sequentially for the subsequent coding layer.

Procedure $LT - RELAY$ (node i)

1. First generate:

   (i) A random *code matrix*, $\mathcal{M}_i = [c_{ij}]_{1 \le i \le T, 1 \le j \le k_{i+1}}$

   (ii) Another independent random *online code copy*, $\mathcal{R}_i = \{\theta_j\}_{1 \le j \le k_{i+1}}$

2. Initialize state to 0 and define the state as j, when j packets have been successfully received from node $i-1$.

3. *Online phase (i.e. while in state $j, 0 \le j \le k_{i+1}$):*

   (i) In the first time slot upon entering state j, send a packet coded according to the code symbol θ_j.

   (ii) After the first time slot (i.e. for the remaining s_j slots): Choose a code symbol uniformly at random from $\hat{e}_j$, the j^{th} column of $\mathcal{M}_i$. Use it for encoding[6], unless it was used in an earlier time slot. Else, it becomes an *idle slot.*

4. Beyond the online phase, generate independent coded packets at each time slot using the standard LT coding procedure for a block length of k_i.

2.6.3.2 Analysis and Justification

We now develop some arguments to justify how the proposed solution works.

Lemma 10. *If n balls are thrown into T bins uniformly, the probability $\mathcal{C}(n, T)$ that there is at least one collision is upper bounded by $2n^2/T$.*

Proof. For $n > T/2$, the bound is trivial since $2n^2/T \ge 1$. Assume $n \le T/2$. It is easy to verify that $1 - x \ge e^{-4x}$ for $0 \le x \le 1/2$.

$$P(\text{No collision}) = \prod_{i=1}^{n-1} (1 - i/T) \ge \prod_{i=1}^{n-1} e^{-4i/T} = e^{-2i(i-1)/T}$$

Hence, $P(\text{collision}) \le 1 - e^{-2i(i-1)/T} \le 2i^2/T$. □

Theorem 11. *(w.h.p.) For any node the number of idle slots during encoding, N_i, satisfies $N_i \le \log k$.*

[6]Note that we have ensured that $\hat{e}_i$ contains only indices of at most i by construction.

Proof. Let I_j denote the number of idle slots in state j for $0 \leq j \leq k_{i+1}$, so that $N_i = \sum_{j=0}^{k_{i+1}} I_j$. Then,

$$\begin{aligned} P(N_i > \log k) &= P(\sum_{j=0}^{k_{i+1}} I_j > \log k) \\ &\leq P\left(\{I_0 > \log k\} \bigcup \{\cup_{j=1}^{k_{i+1}} \{I_j > 0\}\}\right) \\ &\leq P(I_0 > \log k) + k_{i+1} P(I_1 > 0) \\ &\leq \alpha^{\log k} + 2kP(I_1 > 0) \end{aligned}$$

I_1 is the number of idle slots (duplications) observed when choosing *code symbols* uniformly from among $T(k)$ choices for a total of s_1 time slots, and s_1 is the number of consecutive erasures in state 1 and hence is a geometric random variable. Let $P(s_1 = j) = (1-q)q^j$ for $j \geq 0$ where $0 \leq q < \alpha < 1$.

$$\begin{aligned} P(I_1 > 0) &= \sum_{j \geq 0} P(s_1 = j)\mathcal{C}(j, T(k)) \\ &\leq \frac{2}{T(k)} \sum_{j \geq 0} j^2 P(s_1 = j) \text{ (from Lemma 10)} \\ &= \frac{2(1+q)}{1-q} \frac{1}{T(k)} < \frac{4}{1-\alpha} k^{-(1+\delta)} \\ \Rightarrow P(N_i > \log k) &< \alpha^{\log k} + \frac{8}{1-\alpha} k^{-\delta} \\ &= o(k^{-\beta}) \text{ for some } \beta > 0 \end{aligned}$$

□

Definition 2. *Let $\Lambda = [\lambda_{ab}]_{1 \leq a \leq T(k), 1 \leq b \leq k}$ be a random matrix with elements taking $\{0, 1\}$ values be defined as follows. Each element λ_{ab} is a Bernoulli random variable defined as:*

$$\lambda_{ab} = \begin{cases} 1 & \text{, if } c_{ab} \text{ was used for encoding} \\ 0 & \text{, otherwise} \end{cases} \tag{2.34}$$

Theorem 12. *Given any integer c, and arbitrary $S \subset \{1, 2, \ldots, T(k)\} \times \{1, 2, \ldots k\}$ such that*

$$|\{x : (x, j) \in S\}| \leq c \qquad (1 \leq j \leq k) \; ,$$

the random variables $\{\lambda_{ab}\}_{(a,b)\in S}$ are mutually independent and identically distributed as $k \to \infty$.

Proof. First, note that $b_1 \neq b_2 \Rightarrow \lambda_{a_1b_1}$ and $\lambda_{a_2b_2}$ are independent. Hence, without loss of generality we only need to show that $\lambda_{a_1b}, \lambda_{a_2b} \ldots, \lambda_{a_cb}$ are independent where the a_j are all distinct. Denote, for ease of notation, $\beta_j = \lambda_{a_jb}$ for $1 \leq j \leq c$. Since β_j are binary valued, we only need to show that the events $\{\beta_j = 0\}_{j=1}^c$ are independent.

Let $P(s_b = i) = (1-q)q^i$ for $i \geq 0$. For any $1 \leq j \leq c$

$$\begin{aligned} P(\beta_j = 0) &= \sum_{i\geq 0} P(s_b = i)P(\text{ All } i \text{ slots missed } c_{a_jb}) \\ &= \sum_{i\geq 0}(1-q)q^i(1-1/T)^i = \frac{1-q}{1-q+q/T} \end{aligned}$$

Consider an arbitrary subset of these events $\{\beta_{k_j} = 0\}_{1\leq j\leq l}$ where $l \leq c$.

$$\begin{aligned} &P\left(\bigcap_{j=1}^{l}\{\beta_{k_j} = 0\}\right) \\ &= \sum_{i\geq 0} P(s_b = i)P(\text{ All } i \text{ slots missed all } c_{a_{k_j}b} \text{ for } 1 \leq j \leq l) \\ &= \sum_{i\geq 0}(1-q)q^i(1-l/T)^i = \frac{1-q}{1-q+ql/T} \end{aligned}$$

As $k \to \infty$ (and hence, $T(k) \to \infty$),

$$\begin{aligned} (P(\beta_j = 0))^l &\approx \frac{(1-q)^l}{(1-q)^l + l(1-q)^{l-1}q/T} \\ &= \frac{1-q}{1-q+ql/T} = P\left(\bigcap_{j=1}^{l}\{\beta_{k_j} = 0\}\right) \end{aligned}$$

□

Theorem 12 implies the following corollary, which says that the code symbols chosen by the algorithm are scattered "uniformly random" in the following sense:

Corollary 13. *Let $\chi = \{0,1\}^{T(k)\times k}$ denote the ensemble of all possible realizations of the random matrix, Λ. For $\Psi = [\psi_{ij}] \in \chi$ and for any*

$S \subset \{1,\ldots,T(k)\} \times \{1,\ldots,k\}$, denote

$$W_S(\Psi) = \sum_{(i,j)\in S} \psi_{ij}$$

Consider any given integer r, and $E \subset \{1,\ldots,T(k)\} \times \{1,\ldots,k\}$, with $|E| = r$ denoted as $E = \{e_1,\ldots,e_r\}$. For any $\phi = (\phi_1,\ldots,\phi_r) \in \{0,1\}^r$ let $\Theta_\phi = \{\Psi \in \chi : \psi_{e_j} = \phi_j \text{ for } 1 \leq j \leq r\}$. Then, as $k \to \infty$, in the probability space generated by "LT-Relay", $P(\Theta_\phi)$ depends solely on $\sum_{i=0}^{r} \phi_i = W_E(\Psi)$ $\forall\ \Psi \in \Theta_\phi$.

If s_j, the number of time slots spent in column j, had been Poisson rather than geometric, we could have had perfect independence due to the Poisson splitting property. Although this is not quite the case here, we do have a reasonable notion of "asymptotic uniformity" as argued by Corollary 13.

Theorem 14. *Given that (i) the subset of* code symbols *from $\mathcal{M}_i$ used by the algorithm at node i, is uniformly random and (ii) t denotes a time slot past the online phase, the* set *of all coded packets generated by node i till time slot t has the same degree distribution as an LT code.*

Proof. Under the above assumptions, the set of code symbols used till time t is the union of

1. $\mathcal{R}_i$.
2. An (almost) uniform random subset of code symbols from $\mathcal{M}_i$ (partially justified by Corollary 13).
3. The independent LT coded packets generated past the online phase.

Clearly, the components (1), (2) and (3) are mutually independent by their construction. Also, each of them has the degree distribution of an LT coded set of packets - (1) by construction, (2) because of the fact that a uniformly random subset of a set of independent identically distributed (i.i.d.) random variables is also i.i.d. as the original set. In our case, $\mathcal{M}_i$ is the original set of LT coded random variables. Also, (3) is an LT coded set. Hence, their union has the same degree distribution as a set of LT coded packets. □

The above theorem implies that, as long as we ensure monotonically decreasing number of net successful packet receptions across a path from the source, we have a code whose packets have a degree distribution asymptotically identical to the LT code. Also, coded packets are being sent by node $i-1$ to node i at *every* time slot except for the the vanishing fraction of $O(\log k)$ idle slots (Theorem 11) out of a total number of at least $\Omega(k)$ time slots. Hence subject to successful decoding, which could be justified by equivalent LT degree distribution observed, we have a throughput rate to node i equal to the success rate on the channel between nodes $i-1$ to i, $\min_{1\leq j\leq i}(1-\epsilon_j)$.

2.7 Conclusion

In this chapter, we considered design and analysis pertaining to four important practical problems in erasure networks: (1) efficient repair for storage systems, (2) efficient systematic rateless codes, (3) broadcasting with side information, and (4) broadcasting end-to-end across multiple hops. We proposed solutions to each of them in such a way as to import the many desirable characteristics of fountain codes. The constructions have a good deal of synergy in terms of the ideas that lie behind their solutions. For example, the repair algorithm for the augmented LT code is originally motivated by the online encoding procedure envisioned in the context of relaying a fountain code across multiple hops in Section 2.6. Also, the problems of designing the degree distributions for augmented raptor codes and for using LT codes in the presence of side information both involve optimizing the overhead, utilizing the result on the fluid limit of the evolution of degree one packets for any given degree distribution. Moreover, the systematic rateless codes that we come up with in Section 2.4 are a byproduct of the augmented raptor code construction motivated by the storage problem in Section 2.3. Finally, our proposed construction of systematic rateless codes in Section 2.4 that is based on the augmented raptor codes of Section 2.3.7 ended up being useful as an efficient solution to the broadcasting with side information problem, as seen in Section 2.5.5.

CHAPTER 3

THE ROLE OF CODING IN LOCAL BROADCAST FOR ERASURE NETWORKS

3.1 Introduction

In this chapter, we consider the role of coding, specifically in exploiting the local broadcast property of the wireless medium. We first argue that for unicast, the throughput achieved with network coding is the same as that achieved without any coding. This argument highlights the role of a general max-flow min-cut duality and is more explicit than previous published proofs of this fact. The maximum throughput can be achieved in multiple ways without any coding, for example, using backpressure routing, or using some centralized flow scheduler that is aware of the network topology. However, all such schemes, in order to utilize the local broadcast property, require dynamic routing decisions for choosing the next hop for each packet from among the nodes where it is successfully received. This choice seems to depend critically on feedback signaling information like queue lengths, or ARQ. In contrast, the use of network coding can achieve the same without such feedback, in exchange for decoding overhead.

A key task in comparing routing and coding is to assess the importance of feedback signaling to the throughput of routing policies. With this motivation, we explore how feedback at a given node affects its throughput, for an arbitrary vector of given rates of its one hop neighbors to the destination. *Static* routing policies which are essentially *feedback independent*, are considered. An explicit characterization of the optimal policies under such a feedback constraint is obtained, which can be interpreted as a natural generalization of both flooding and traditional routing (which does not exploit local broadcast, because the next hop is fixed prior to the transmission). When losses at the receivers are independent (still allowing for dependencies on transmissions by two different nodes, to model interference), the reduction in capacity due to constraining the feedback is limited to a constant fraction $(1 - e^{-1} = 63\%)$ of the coding capacity, and gets arbitrarily close to optimal as the capacity itself is low. This result is also extended to a more general version on feed-forward networks without any assigned rates of the one hop neighbors to the destination. However, if there are dependencies in the losses seen by receivers from a single broadcast, the reduction could be arbitrarily bad, even with just two hops.

Network coding was originally introduced in [16] as a general framework to achieve the optimal multicast rate from a data source to a set of receivers in wired networks. In contrast to "store-and-forward," also called "routing" operation, network coding performs recombinations of data packets at network nodes, while the former operation never alters original packets. Since then, many other applications of network coding have been identified. In particular it has been considered in the context of wireless networks [71] for unicast communications, where such a wireless setting was identified as an especially good candidate for network coding because of the local broadcast. The wireless scenario of [71] features lossy transmissions as well as local broadcast. The experimental evaluations of [71] show benefits of network coding over routing policies in terms of the transmission rates achieved. This is in contrast with recent work of Smith and Hassibi [72], which implies that routing policies can achieve the capacity for unicast. This raises the question: Are there any benefits of network coding over routing in wireless unicast communications? If so, where do they stem from, and how large can they be?

In a nutshell, we show that the benefits of coding over routing depend on the extent of available feedback signaling and characterize the relation. Our first contribution is a simpler argument, using linear programming (LP) duality, that the maximum routing throughput in the case of wireless unicast is the same as the rate achieved with coding. Using this, it is shown that backpressure routing achieves not only the optimal throughput among routing policies, but also more generally across policies that involve coding. However, backpressure policy performs a very dynamic routing of each packet through the network, and requires exchanging queue lengths from all neighbors at every step. This motivates an investigation into the fundamental limits of *static* routing policies which do not need extensive feedback signaling. We study this tradeoff and quantify the limits imposed by an appropriately defined notion of restricted feedback.

Consider the following extreme case to motivate the analysis (see Figure 3.1): There are m distributed agents with each agent holding a copy of a set of m distinct packets (which they all received from the source). With full coordination, they can schedule all the m packets in one transmission each by avoiding duplication. However, if we restrict their choice to be made in a distributed manner without coordination, the total number of distinct packets covered by a random choice at each relay approaches $1-(1-1/m)^m \to 1-e^{-1}$, thus leading to only 63% throughput. 100% throughput could be achieved by either (i) making coordinated choices among relay nodes through conferencing among all the relays, or (ii) letting each node send an independent random linear combination of all m packets. In other words, network coding seems to be essentially solving a distributed synchronization problem without the need for any feedback signaling.

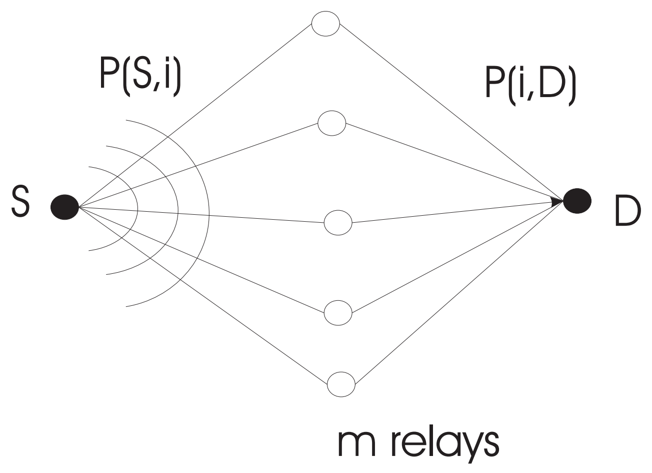


Figure 3.1: The relay network example.

The above synchronization constraint is modeled as the *feedback independent routing* (FIR) restriction (see Definition 8): informally, this says that the decision of whether a node chooses to forward a packet should not depend on erasures at other nodes. In all examples of policies that achieve the capacity without any coding, this information is implicitly utilized based on the feedback signaling. Under the FIR restriction, we show that the optimal policies are characterized as *tagging policies* where each packet being broadcast is assigned multiple next hops. *Tagging policies* can be considered a generalization of both flooding (where every broadcast packet is routed by everyone that receives it), and traditional routing (where each packet is routed only by a specific receiver chosen prior to the broadcast). When transmissions are subject to mutually independent losses at the receivers (but still possibly allowing for dependences of losses for packets transmitted by two different nodes), even when restricted to feedback independent routing, it is possible to achieve at least 63% of the capacity. In fact, as the capacity, $C^* \to 0$, the throughput achieved becomes 100% of C^* in the limit. For a general feed-forward network with $h+1$ hops in which all nodes are restricted to operate under a similar routing constraint, we show under a similar independence condition for link losses (but allowing dependence across different broadcasts) that the reduction in throughput is lower bounded as $f^h(C^*)$, where $f(x) = 1 - e^{-x}$. Thus, for a limited number of hops, and when the capacity is low to begin with, one can achieve close to optimal throughput without actually making dynamic routing choices. One might imagine that dependencies in link losses supply implicit information about other link losses, and therefore allow better throughput compared to independent losses. But this is not true, as we show a counterexample with dependent link losses for which static feedback independent routing capacity is arbitrarily bounded away from the network coding capacity even for a 2-hop network. One of the implications in this conclusion is that, when feedback is constrained, coding in the network could be unavoidable to achieve non-vanishing throughput.

3.2 Related Work

The merits of routing versus coding have been extensively studied in the context of wireless, both theoretically and by experiments. The following two lines of work deal with the ways in which local broadcast can be exploited: (i) By using coding: This has been investigated in [71, 73, 74]. The advantage of using network coding to exploit local broadcast is the lack of a need for sophisticated coordination and/or routing choices among the nodes. The price to pay for this is the decoding complexity. (ii) By making dynamic routing/flooding choices along with rich feedback signaling: This was the path taken in [72, 75, 76]. The maximum throughput in an information theoretic sense for the wireless erasure network (WEN) model was studied in [74] and was shown to be equal to the appropriate hypergraph min cut, C^*. In [72], for the unicast wireless setting, a flooding based policy with network wide instantaneous broadcast of the identity of each packet received at the destination was shown to achieve C^*. Using this, [72] makes an important observation that coding is in fact not necessary for achieving a throughput equal to the capacity in wireless unicast settings. In [76], backpressure routing (requiring rich feedback signaling to perform routing) was shown to be an optimal routing plus scheduling policy in a much more general context taking into account interference effects. In general (i.e. for multicast), [77] proposes the use of feedback together with network coding for online decoding.

3.3 The Wireless Erasure Network Model

Let $G = (V, E)$ be a directed graph. Let $\mathcal{N}(i) = \{j \in V : (i,j) \in E\}$. We consider a wireless network that operates on this graph over time $t \in \{0, 1, 2, \ldots\}$. For each t, a node can "broadcast" to its neighbors. The network is subject to probabilistic constraints on the successes of these broadcasts. Specifically, at any given time, t, for each $i \in V, Z \subseteq N(i)$, let $\chi(i, Z, t)$ denote a $\{0, 1\}$ random variable that represents the following:

$$\chi(i,Z,t) = \begin{cases} 1 & \text{, if broadcast from node } i \text{ at time } t \text{ is successful to } Z, \\ & \quad \text{and fails to } \mathcal{N}(i)/Z \\ 0 & \text{, otherwise} \end{cases}$$

The random variables χ can be arbitrarily correlated across the argument i, which allows for modeling arbitrary interference constraints, but we will assume that they are independent across t. For example, this could model the situation in the random access scheduling, where at each time slot, there is a probability that a given node actually transmits, with the transmission being successful if and only if no other node in its interference radius is simultaneously active. Note that, for Z, we have: $\sum_{Z \subseteq \mathcal{N}(i)} \chi(i, Z, t) = 1 \ \forall i, t$. We define: $c(i, Z) = E[\chi(i, Z, t)] \ \forall t$. Thus, we are given a network topology along with $c(i, Z)$ as the capacities. Our analysis will be impervious to the correlations across i, so we will not explicitly specify them. An interesting special case of the above model is to have link (i, j) successful with probability $p(i, j)$, with losses from i to each of the neighbors in $\mathcal{N}(i)$ being independent.

We then have:

$$c(i,Z) = \prod_{j \in Z} p(i,j) \prod_{j \in \mathcal{N}(i)/Z} (1 - p(i,j))$$

We consider a single unicast flow. However, the insights obtained are also general enough for multiple competing flows, with each flow assigned a fixed fraction of the link capacities and a comparison with intra-session network coding in such a scenario. Thus, without loss of generality we assume a source, S, and destination, D. The source has an infinite set of packets indexed over the integers, intended for replication at D. For any $v \in V$, let $\alpha_v(t)$ denote the number of distinct packets that were replicated at node v till time slot t.

Definition 3 (Routing policy, $\mathcal{P}$). *A routing policy $\mathcal{P}$ decides for each node $i \in V$, and time $t \in \{0, 1, \ldots\}$, a packet to be transmitted from among the $\alpha_i(t-1)$ choices in its possession.*

Definition 4 (Capacity). *The throughput of a policy $\mathcal{P}$ is defined as:*

$$C(\mathcal{P}) = \liminf_{t \geq 0} \frac{E[\alpha_D(t)]}{t}$$

The capacity is the highest possible throughput $C \triangleq \sup_{\mathcal{P}} C(\mathcal{P})$.

3.4 A Max Flow Characterization

A key observation is that any policy which uniquely routes each packet without keeping multiple copies can be represented by a valid flow on the throughput constrained graph. In the linear program defined below, each feasible solution represents a policy that routes a fraction proportional to $r(i,j,Z)$ of successful broadcasts from i to Z uniquely to $j \in Z$. The term $\sum_{\{Z \in \mathcal{N}(i): j \in Z\}} r(i,j,Z)$ represents the net flow from i to j. The optimum value of the linear program (LP) then represents the throughput achieved.

Definition 5 (F, the maxim flow value). *Let P denote the set of all $S-D$ paths in G. We define the F for a given broadcast capacitated graph as the* *optimum value of the following LP:*

$$F = \max \sum_{p \in P} x_p \tag{3.1}$$

$$\textit{Subject to:} \quad x_p \geq 0 \quad \forall \quad p \in P$$

$$r(i,j,Z) \geq 0 \ \ \forall \ \ \{(i,j,Z) : (i,j) \in E, j \in Z \subseteq \mathcal{N}(i)\}$$

$$\sum_{\{p \in P : (i,j) \in p\}} x_p - \sum_{\{Z \in \mathcal{N}(i) : j \in Z\}} r(i,j,Z) \leq 0 \ \ \forall (i,j) \in E$$

$$\sum_{j \in Z} r(i,j,Z) \leq c(i,Z) \ \ \forall \ \ \{(i,Z) : Z \subseteq \mathcal{N}(i)\}$$

Let x_p^*, $r^*(i,j,Z)$ denote the optimum solution to the above LP. It is straightforward to show that this capacity can be achieved by the following policy:

Definition 6 ($\mathcal{P}_{fs}$, the flow splitting policy)**.** *Any packet transmitted by node i, and received by the set $Z \subseteq \mathcal{N}(i)$ of its neighbors is "routed" uniquely to $j \in \mathcal{N}(i)$ with probability $\frac{r^*(i,j,Z)}{\sum_{k \in Z} r^*(i,k,Z)}$ (thus ensuring that at most one copy of each distinct packet is being transmitted at any point).*

We now consider the min cut appropriate for the model by considering the probability that at least one of the nodes across the cut receives a transmission.

Definition 7 (Minimum cut, C^*)**.** *A Cut is a disjoint partition of V into A and $\bar{A}$ with $S \in A$ and $D \in \bar{A}$. The capacity of the cut is then:*

$$C(A) = \sum_{i \in A, Z \subseteq \mathcal{N}(i), Z \cap \bar{A} \neq \phi} c(i,Z)$$

The minimum cut is: $C^* = \min_A C(A)$.

It is easy to argue that C^* is an upper bound on the throughput for any scheme even with coding, and was in fact shown to be equal to the information theoretic capacity of the WEN in [74]. Thus, $C \leq C^*$. Based on a duality argument analogous to the classical max flow min cut theorem, the following can be shown:

Theorem 15. $F = C^* = C$

Proof. Consider the dual program for the LP at Equation (3.1) with dual variables $b(i,Z)$ for each $i \in V, Z \subseteq \mathcal{N}(i)$ and $y(i,j)$ for each $(i,j) \in E$. We have:

$$DUAL^* = \min \sum_{i \in V, Z \subseteq \mathcal{N}(i)} c(i,Z)b(i,Z) \tag{3.2}$$

Such that:

$$\sum_{\{(i,j) \in p\}} y(i,j) \geq 1 \quad \forall \ p \in P \tag{3.3}$$

$$-y(i,j) + b(i,Z) \geq 0 \ \ \forall \{(i,j,Z) : (i,j) \in E, j \in Z \subseteq \mathcal{N}(i)\}$$

$$y(i,j) \geq 0 \ \ \forall (i,j) \in E$$

$$b(i,Z) \geq 0 \ \ \forall (i,Z) : i \in V, Z \subseteq \mathcal{N}(i) \tag{3.4}$$

Consider the mincut as written in Equation (7), and let $A^*, \bar{A}^*$ denote this cut. Let $y^*(i,j) = 1$ if $i \in A^*, j \in \bar{A}^*$ and 0 otherwise. Similarly, let $b^*(i,Z) = 1$ if $i \in A, Z \cap \bar{A}^* \neq \phi$ and 0 otherwise. Then, it can be verified that this defines a feasible solution to the dual LP above, and that $\sum_{i \in V, Z \subseteq \mathcal{N}(i)} c(i,Z)b^*(i,Z) = C^*$. Thus,

$$C^* \geq DUAL^* \tag{3.5}$$

Now consider an integral constrained version of the above dual:

$$DUAL_* = \min \sum_{i \in V, Z \subseteq \mathcal{N}(i)} c(i,Z)b(i,Z) \tag{3.6}$$

Such that:

$$\sum_{\{(i,j) \in p\}} y(i,j) \geq 1 \ \ \forall p \in P \tag{3.7}$$

$$-y(i,j) + b(i,Z) \geq 0 \ \ \forall \{(i,j,Z) : (i,j) \in E, j \in Z \subseteq \mathcal{N}(i)\} \tag{3.8}$$

$$y(i,j) \in \{0,1\} \ \ \forall (i,j) \in E$$

$$b(i,Z) \in \{0,1\} \ \ \forall (i,Z) : i \in V, Z \subseteq \mathcal{N}(i) \tag{3.9}$$

Let y_*, b_* define the optimal solution to the above integral constrained LP. Then, define $A_* = \{i \in V : \exists$ a path, p, from S to i such that $\sum_{\{(i,j) \in p\}} y_*(i,j) = 0\}$. Then, $D \notin A_*$ due to Equation (3.7), and thus A_* defines an (S,D) cut. Furthermore, from Equations (3.8) and (3.6), we get $C(A_*) = DUAL_*$. This implies that

$$C^* \leq DUAL_* \tag{3.10}$$

To summarize, we so far have:

$$F = DUAL^* \leq C \leq C^* \leq DUAL_* \tag{3.11}$$

If we are able to argue that the constrained LP indeed achieves the optimum (the details of this are given in Lemma 16), we would then have $DUAL^* = DUAL_*$, implying that all quantities in Equation (3.11) are the same. $\square$

Lemma 16. $DUAL^* = DUAL_*$

Proof. We use an analogous argument employed in showing the corresponding statement for the classical max flow min cut theorem. The argument considers a probability distribution on the set of all possible cuts and argues that the expected value of C^* thus obtained is no more than $DUAL^*$, which in turn implies that $DUAL_* \leq DUAL^*$, thus completing the proof. Consider the dual LP in Equation (3.2), and let $y^*(i,j), b^*(i,Z)$ denote the optimal solution which achieves $DUAL^*$. Consider a graph with edge lengths given by $y^*(i,j)$ and let $d(i) \triangleq$ length of the shortest path from S, with edge lengths given by $y^*(i,j)$. Let $\lambda \in (0,1)$ be chosen uniformly and define:

$$A^* = \{i \in V : d(i) \leq \lambda\}$$

This defines a cut with probability 1, since $d(S) \geq 1$ from Equation (3.3).

Then:

$$
\begin{aligned}
E[C(A^*)] &= E[\sum_{i\in A^*, Z\subseteq \mathcal{N}(i), Z\cap \bar{A}^*\neq \phi} c(i,Z)] \\
&= E[\sum_{i\in V, Z\subseteq \mathcal{N}(i)} c(i,Z)\mathbb{1}\{i\in A^*, Z\subseteq \mathcal{N}(i), Z\cap \bar{A}^*\neq \phi\}] \\
&= \sum_{i\in V, Z\subseteq \mathcal{N}(i)} c(i,Z)P\left(i\in A^*, j\in \bar{A}^* \text{ for some } j\in Z\right) \\
&= \sum_{i\in V, Z\subseteq \mathcal{N}(i)} c(i,Z)P\left(d(i)\leq \lambda, d(j)>\lambda \text{ for some } j\in Z\right) \\
&= \sum_{i\in V, Z\subseteq \mathcal{N}(i)} c(i,Z)(\max_{j\in Z} d(j)-d(i))_+ \quad (\because \lambda \text{ is uniform}) \\
&\leq \sum_{i\in V, Z\subseteq \mathcal{N}(i)} c(i,Z)(\max_{j\in Z} y^*(i,j)) \quad (\text{ traingle inequality }) \\
&\leq \sum_{i\in V, Z\subseteq \mathcal{N}(i)} c(i,Z)b^*(i,Z) \quad (\text{ from Equation (3.4)}) = DUAL^*
\end{aligned}
$$

□

Thus, coding in the network is not necessary to achieve the optimal throughput in unicast, assuming unconstrained feedback signaling. This fact was also argued by [72] in a closely related continuous time model (and conjectured and verified by simulation for the discrete time model that we consider). This was accomplished by considering a policy which involves flooding the network with each packet until at least one copy reaches the destination and subsequently using a network wide feedback signal to delete these copies every time the destination receives a new packet. It was shown that such a policy stabilizes the network for all rates below the C^* using Lyapunov stability argument. Lemma 17 shows that a distributed backpressure scheme which routes each packet to the least loaded neighbor (thus, avoiding multiple copies) also achieves the information theoretic min cut capacity, C^*. This is also suggested by Theorem 15 in conjunction with Neely's result of optimality of backpressure routing schemes in a context that involves power control and with a different interference model [76].

Lemma 17. *Consider a Markov chain defined on the network as follows: Each node has a queue of packets. New packets arrive to the queue at the source according to a Bernoulli process of rate λ. For any t, i, Z such that*

$\chi(i, Z, t) = 1$*, the backpressure policy routes a packet from node* i *to a node* j *such that queue size difference is maximized (subject to being positive). If* $\lambda < C^*$*, the Markov chain thus obtained is stable (positive recurrent).*

Proof. The proof follows along the lines of [78] using a quadratic Lyapunov function and Foster's condition to show that the Markov chain is positive recurrent. Packets are injected at the source node according to a Bernoulli i.i.d. process with mean $\lambda < C^*$. We shall use the potential, $V(t) = \sum_{i=1}^{n} q_i^2(t)$, where i represents an index over the n nodes in the network and $q_i(t)$ represents the number of packets that are held at node i. Let $\{\mathcal{F}_t\}_{t\geq 0}$ be a filtration adapted to the queue length process. By Foster's theorem (see [78] and the references therein), a sufficient condition for stability is to show that that the Lyapunov drift $E[V(t+1) - V(t)/\mathcal{F}_t] < 0$ when $V(t)$ is sufficiently large. Let $\mathbf{q}(t)$ denote the n dimensional row vector with $q_i(t)$ being the i^{th} element. Let $\mathbf{R}$ be the adjacency matrix of dimension $n \times L$ (where $L = |E|$, the number of links) for the given graph (i.e., the $(i, l)^{th}$ element, $r(i, l)$ is $1(-1)$ iff the link l starts(ends) at i, and 0 otherwise). Let $\mathbf{E}(t)$ denote the L dimensional indicator vector denoting the subset of links on which packets were routed by the backpressure policy, and let $\mathbf{A}(t)$ denote the arrival process indicator vector, i.e. the element of $\mathbf{A}(t)$ corresponding to the source is the Bernoulli random variable with mean λ and all other elements are 0. Then, the queue lengths evolve according to $\mathbf{q}(t+1) = \mathbf{q}(t) + \mathbf{R}\ \mathbf{E} + \mathbf{A}(t)$. Thus (with . denoting the usual dot product of vectors):

$$\begin{aligned} V(t+1) - V(t) &= \mathbf{q}(t+1).\mathbf{q}(t+1) - \mathbf{q}(t).\mathbf{q}(t) \\ &= (\mathbf{R}\ \mathbf{E}(t) + \mathbf{A}(t)).(2\mathbf{q}(t) + \mathbf{R}\mathbf{E}(t) + \mathbf{A}(t)) \\ &\leq (n+1)^2 + 2(\mathbf{q}(t).\mathbf{R}\mathbf{E}(t) + \mathbf{q}(t).\mathbf{A}(t)) \end{aligned}$$

Since the first term above is constant over time, showing that the second term has a large negative drift for large queue lengths is sufficient (since large potential implies large queue length under connectivity assumptions on the graph). Let $\mathbf{W}(t)$ denote the L dimensional vector where the l^{th} element denotes the queue length difference for the l^{th} link. Then, $\mathbf{q}(t).\mathbf{R}\mathbf{E}(t) = -\mathbf{W}(t).\mathbf{E}(t)$. Hence, we only need to argue that $E[-\mathbf{W}(t).\mathbf{E}(t) + \mathbf{q}(t).\mathbf{A}(t)/\mathcal{F}_t]$ has a negative drift. $E[\mathbf{q}(t).\mathbf{A}(t)/\mathcal{F}_t] = \mathbf{q}(t).\mathbf{A}$ where A is a vector where the element corresponding to the source is λ and all other elements 0. Since

$\lambda < C^*$, it follows from the max flow interpretation of Theorem 15 that there exist variables $r^*(i,j,Z)$ satisfying the constraints of the linear program given by Equation (3.1), and an $\epsilon > 0$ such that $\mathbf{q}(t).\mathbf{A} \leq (1-\epsilon)\mathbf{W}(t).\mathbf{f}^*$, where the component of $\mathbf{f}^*$ corresponding to link $l = (i,j)$ is $f_l^* = \sum_{\{Z \in \mathcal{N}(i): j \in Z\}} r^*(i,j,Z)$. Let $\mathbf{e}^*(t)$ denote the indicator vector of dimension L, for the routing selected by the flow splitting policy, $\mathcal{P}_{fs}$ given in Definition 6. Then, $\mathbf{W}(t).\mathbf{f}^* = E[\mathbf{W}(t).\mathbf{e}^*(t)/\mathcal{F}_t]$. Thus,

$$E[-\mathbf{W}(t).\mathbf{E}(t) + \mathbf{q}(t).\mathbf{A}(t)/\mathcal{F}_t] = E[-\mathbf{W}(t).\mathbf{E}(t) + (1-\epsilon)\mathbf{W}(t).\mathbf{e}^*(t)/\mathcal{F}_t]$$
$$= E[-\mathbf{W}(t).\mathbf{E}(t) + \mathbf{W}(t).\mathbf{e}^*(t)/\mathcal{F}_{t^+}/\mathcal{F}_t] - \epsilon E[\mathbf{W}(t).\mathbf{e}^*(t)/\mathcal{F}_t]$$

Here, $\mathcal{F}_{t^+}$ denotes the sigma algebra that contains information about the successes on links at time t in addition to the queue length process until time t, and thus, $\mathcal{F}_t \subseteq \mathcal{F}_{t^+} \subseteq \mathcal{F}_{t+1}$. The first term is non-positive because the backpressure policy minimizes $\mathbf{W}(t).\mathbf{E}(t)$ among all possible options conditioned on the information available on the link successes and the queue lengths (which is what conditioning on $\mathcal{F}_{t^+}$ denotes). The second term takes arbitrarily large negative values for large $V(t)$ for any fixed $\epsilon > 0$, and thus, we have the required negative drift. □

3.5 Formalizing a Notion of Restricted Feedback

While we have so far discussed schemes that can achieve the unicast capacity in a wireless network without the need for coding, employing coding in the network can achieve the capacity without using feedback. This calls for an understanding of the inherent limits to the achievable throughput when we restrict the exploitation of feedback in the choice of routing policies. The decision of whether to forward a packet further at a given node should ideally be made without considering the erasure events on other links (this is not the case with any of the routing schemes we have discussed so far which achieve the maximum throughput). In this context, we first fix the vector of rates that the one hop neighbors can support to the destination simultaneously and study how feedback constrained routing affects the overall throughput from the source. The most obvious visualization of this is a network in which the source and destination are assisted by relays which do not communicate within themselves (Figure 3.1).

Let $c(Z) = E[\chi(S, Z, t)]$ for $Z \subseteq [m]$. Let $p(S, i) = \sum_{Z \subseteq [m]: i \in Z} c(Z)$ denote the probability that the a packet transmitted from S is received at i successfully. We shall let $p(i, D)$ denote the long-term throughput that relay i can support to the sink D. Given a generalized routing/flooding policy, we shall use the $\{0, 1\}$ random variables, $r_i(p), r_i^*(p)$ which denote the following events:

1. $\{r_i(p) = 1\} \iff$ Packet transmitted in time slot p from the source (which shall henceforth be referred to as packet p) was received successfully by relay i. We assume that the source attempts broadcast of distinct packets at each time slot without any loss of generality (e.g., by employing appropriate source coding, or because it has an unlimited stream of useful packets).

2. $\{r_i^*(p) = 1\} \iff$ Packet p is routed to D via relay i. Note that: (1) $\{r_i^*(p) = 1\} \Rightarrow \{r_i(p) = 1\}$. (2) It is possible that $r_i^*(p) = 1$ for multiple i.

We will now explicitly describe what we mean by feedback independent routing.

Definition 8 (Feedback independent routing (FIR))**.** $\forall p \in \{0, 1, \ldots\}, \forall A, B \subseteq [m]$ *such that* $A \cap B = \phi$*, the given routing policy satisfies the* FIR *restriction if, conditioned on* $\{r_i(p) = 1 \;\; \forall \;\; i \in A\}$*, the following two collections of random variables:* $\{r_i^*(p)\}_{i \in A}$ *and* $\{r_i(p)\}_{i \in B}$*, are mutually independent.*

This condition is essentially equivalent to assuming a lack of feedback to the broadcasting node. Technically, lack of feedback is sufficient but not entirely necessary to satisfy this. While this distinction is subtle, it might be noted that one does not violate FIR by using rudimentary feedback for purposes other than routing. Nevertheless, source coding is a convenient way to completely eliminate feedback. As for the first hop nodes, we merely consider them as black-boxes that support some arbitrary vector of simultaneous rates to the destination. For a 2-hop relay network, these simultaneous rates could be achieved by employing any capacity achieving scheme for a single erasure channel, i.e. either by (1) feedback to the relays, or (2) forward erasure coding (FEC) at each relay.[1]

We have so far discussed three policies which achieve capacity without any coding: (1) the flow splitting policy $\mathcal{P}_{fs}$ in Definition 6, (2) the backpressure policy in Lemma 17 and (3) the policy of [72]; but they all involve a heavy interaction between feedback and routing, and hence fail to satisfy the given constraint.

3.6 Capacity under Feedback Independent Routing Constraint

Definition 9. *The capacity with feedback independent routing, C_R is defined as the maximum throughput as in Definition 4, restricted to policies $\mathcal{P}$ satisfying Definition 8.*

For FIR, one extreme is to tag each packet with a single relay (this is how routing is done in practice in the 802.11 protocol). This could be suboptimal because it does not exploit the local broadcast advantage. The other extreme is to flood every packet to all relays. This could be suboptimal when the relays do not have enough capacity to forward all packets they received to the destination. They will then have to make distributed decisions on which packets to forward from among the received packets, leading to redundant transmissions and a hence a decreased throughput. As we will show, when we restrict to FIR, the maximum throughput is achieved within a subclass of policies that we shall call the *tagging policies*. A tagging policy assigns to each packet a subset of relays, $Z \subseteq [m]$, which is independent of any feedback. We represent the fraction of packets that are tagged with Z as $t(Z)$. A relay i that receives a packet successfully routes the packet without dropping it if and only if $i \in Z$. These packets are retransmitted until the destination receives them. The capacity of such policies can be expressed via the linear program below, where each feasible solution corresponds to a specific tagging policy. The last constraint in the LP states that the arrival

[1]The key aspect that distinguishes such FEC from network coding is that in the case of FEC, the destination has to be able to decode the data being encoded by each relay independently from the transmissions received from that specific relay alone, whereas with network codes, the relay only needs to collect the packets from all relays and jointly decode them.

rate of packets to any relay's queue has to be less than its forwarding capacity to the destination.

Definition 10 (Tag capacity, C_T). *C_T is defined as the optimum value of the following LP with variables $t(Z) \geq 0$ where $Z \subseteq [m]$.*

$$C_T = \max \sum_{Z,Z' \subseteq [m] : Z \cap Z' \neq \phi} t(Z)c(Z') \tag{3.12}$$

$$\textit{Subject to: } \sum_{Z \subseteq [m]} t(Z) \leq 1; \tag{3.13}$$

$$p(S,i) \left(\sum_{Z \subseteq [m] : i \in Z} t(Z) \right) < p(i,D) \qquad\qquad \forall i \in [m] \tag{3.14}$$

All packets that reach a relay successfully *and* have the relay in the tag will eventually reach D because of the constraint in Equation (3.14) (which implies that the queue of packets at each relay is stable). Since the tags are chosen independent of the losses, the probability that a packet is successfully transmitted to some relay which is also included in its tag is $\sum_{Z \cap Z' \neq \phi} t(Z)c(Z')$, which can be readily shown to be equal to the throughput of the policy as per Definition 4.

Lemma 18. $C_R \geq C_T$

Lemma 18 holds due to the fact that any feasible solution to the LP gives us a tagging scheme whose throughput is equal to the value of the LP.

3.7 Optimality of Tagging Policies under FIR

Remarkably, these policies will now also be shown to be optimal in general under the FIR constraint, thus giving us an explicit characterization of C_R in the form of Theorem 19.

Theorem 19. *$C_R \leq C_T$, implying $C_T = C_R$*

Proof of Theorem 19. We look at any policy that satisfies FIR and show that it is possible to define appropriate variables that define a feasible solution to the LP in Definition 10, and for which the throughput is upperbounded by the objective function.

Consider any arbitrary policy. We will show that the expected number of distinct packets that reach the destination in k time slots cannot be more than kC_T as long as FIR in Definition 8 is satisfied, thus implying that the C_R is at most C_T. Define:

$$t(Z) = \frac{1}{k}\sum_{p=1}^{k} P\left(\frac{r_i^*(p) = 1 \ \forall\ i \in Z \text{ and } 0 \ \ \forall i \in [m]/Z}{r_i(p) = 1 \ \ \forall\ p \in [m]}\right) \tag{3.15}$$

where we use the standard convention of writing $\frac{P(A\cap B)}{P(B)}$ as $P(\frac{A}{B})$ for an event B with positive probability. It is easily verified that $t(Z) \geq 0$ and $\sum_{Z\subseteq[m]} t(Z) = 1$. We will now verify that the third constraint (Equation (3.14)) holds for any $i \in [m]$.

$$\begin{aligned}
&kp(i,D) \geq E[\sum_{p=1}^{k} \mathbb{1}\{r_i^*(p) = 1\}] \\
&= \sum_{p=1}^{k} P(r_i(p) = 1) P\left(\frac{r_i^*(p) = 1}{r_i(p) = 1}\right) \\
&= p(S,i) \sum_{p=1}^{k} P\left(\frac{r_i^*(p) = 1}{r_i(p) = 1}\right) \\
&= p(S,i) \sum_{p=1}^{k} P\left(\frac{r_i^*(p) = 1}{r_j(p) = 1 \ \forall j \in [m]}\right) \text{ (using def.8)} \\
&= p(S,i) \sum_{p=1}^{k} \sum_{Z\ni i} P\left(\frac{r_j^*(p) = 1 \ \forall\ j \in Z \text{ and } 0 \ \forall j \in [m]/Z}{r_j(p) = 1 \ \forall j \in [m]}\right) \\
&= kp(S,i) \sum_{Z\ni i} t(Z) \ \text{ (by definition of } t(Z) \text{ in Equation (3.15))}
\end{aligned}$$

We now calculate the number of distinct packets replicated at D in k time slots.

$$\begin{aligned}
&E[\alpha_D(k)] = \sum_{p=1}^{k} P(\{\bigcup_{i\in[m]} \{r_i^*(p) = 1\}\}) \\
&= \sum_{p=1}^{k} \sum_{Z\subseteq[m]} P\Big(r_i(p) = 1 \ \forall\ i \in Z \text{ and } 0 \ \ \forall i \in [m]/Z\Big) \times
\end{aligned}$$

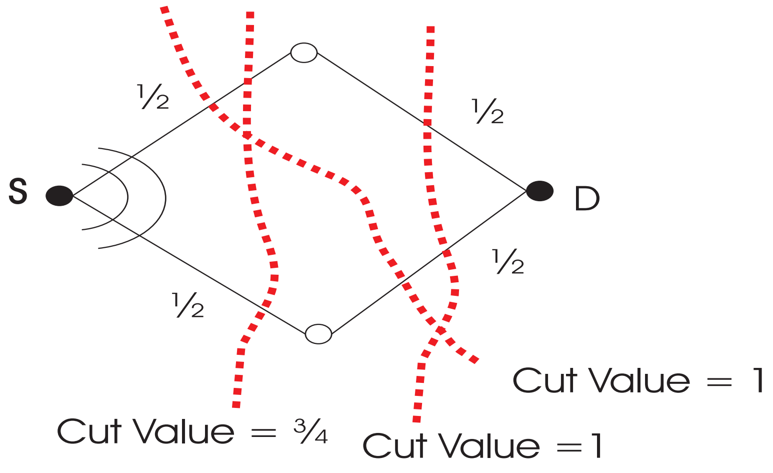
$$
\begin{aligned}
&P\Big(\frac{r_i^*(p)=1 \text{ for some } i\in[m]}{r_i(p)=1\ \forall\ i\in Z \text{ and } 0\ \ \forall i\in[m]/Z}\Big)\\
&=\sum_{p=1}^{k}\sum_{Z\subseteq[m]} c(Z)P\left(\frac{r_i^*(p)=1 \text{ for some } i\in Z}{r_i(p)=1\ \forall\ i\in Z \text{ and } 0\ \ \forall i\in[m]/Z}\right)\\
&\qquad(\because \forall i\in[m]/Z, \{r_i(p)=0\}\Rightarrow\{r_i^*(p)=0\})\\
&=\sum_{p=1}^{k}\sum_{Z\subseteq[m]} c(Z)P\left(\frac{r_i^*(p)=1 \text{ for some } i\in Z}{r_i(p)=1\ \forall\ i\in[m]}\right)\\
&\qquad\text{(using FIR Definition 8)}\\
&=\sum_{p=1}^{k}\sum_{Z} c(Z)\sum_{Z'\cap Z\neq\phi} P\left(\frac{r_i^*(p)=1\ \forall i\in Z' \text{ and } 0 \text{ else}}{r_i(p)=1\ \forall\ i\in[m]}\right)\\
&=k\sum_{Z\subseteq[m]} c(Z)\sum_{Z'\subseteq[m]:Z'\cap Z\neq\phi} t(Z)\\
&=k\sum_{Z,Z'\subseteq[m]:Z\cap Z'\neq\phi} t(Z)c(Z')
\end{aligned}
$$

□

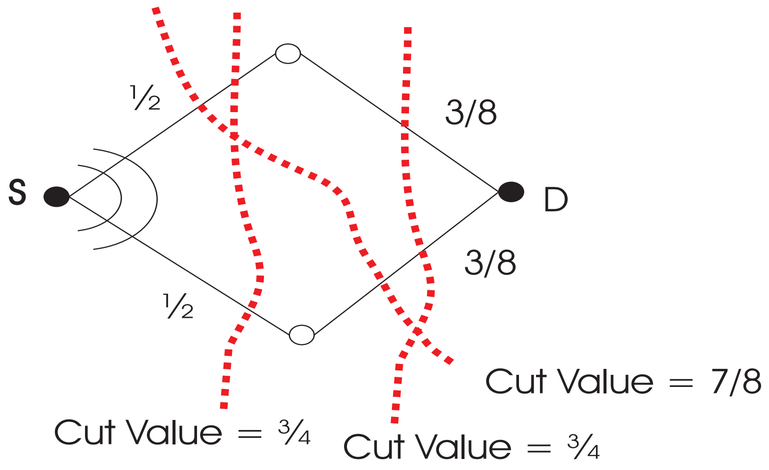
In general, it is not obvious if C_R is strictly less than the general min cut, C^*. Indeed, in many cases, these two quantities match, implying that in such cases, not only do we not need any coding, but the capacity can be achieved by optimized tagging schemes that satisfy FIR. For example, consider a network with two relays with success probabilities $p(S,1), p(S,2), p(1,D), p(2,D)$ where link losses for the same transmission are independent.

Example 1. *If* $p(S,1), p(S,2), p(1,D), p(2,D)$ *are all* $1/2$*, we have:* $C(\{1\}) = C(\{2\}) = C(\{1,2\}) = 1/4$*. From Figure 3.2(a),* $C^* = 3/4$ *and from the LP (Equation (3.12)), we can calculate that* $C_R = 3/4$*. In fact, a flooding policy achieves this rate of* $3/4$*. So there is no reduction in the capacity under FIR.*

Example 2. *Consider an example where the cuts are more finely balanced:* $p(S,1), p(S,2) = 1/2$*; and* $p(1,D), p(2,D) = 3/8$*. Again,* $C^* = 3/4$*, as explained in Figure 3.2(b). Set* $t(\{1\}) = x, t(\{2\}) = y, t(\{1,2\}) = z$*. Then:* $C_R = \ Max\ \ (\frac{1}{2}x + \frac{1}{2}y + \frac{3}{4}z)$ *Subject to:* $x, y, z \geq 0,\ \ x+y+z \leq 1,\ \ x+z \leq 3/4,\ \ y+z \leq 3/4$*. The optimal value can be verified to be* $5/8 < 3/4 = C^*$*. This optimum throughput of 5/8 under FIR is achieved by routing 25% of the packets exclusively to relay 1, 25% exclusively to relay 2 and the remaining 50% of the packets to both the relays.*



(a) Example 1



(b) Example 2

Figure 3.2: Examples.

3.8 Quantifying the Loss of Throughput under FIR

In this section, we will use the theory from previous sections to obtain results on the throughput attainable under FIR. We will first consider the case where the link losses from a given transmitter seen at its various receivers are independent. Note that this involves no assumptions on interference between two different transmissions. As a critical tool, we will use a *flooding policy*, which is defined below.

Definition 11 (Flooding policy, $\mathcal{P}_F$)**.** *Each relay blindly chooses each received packet from the source to be forwarded to the destination with probability* $\min(1, p(i,D)/p(S,i))$*. In other words, the relay effectively makes a uniformly random selection of a* $p(i,D)/p(S,i)$ *fraction of its received packets to be forwarded to the destination whenever* $p(i,D) \leq p(S,i)$*.*

We named the above policy as flooding in a general sense, because every packet that is successfully received is considered for being forwarded at any relay. It is possible that the total number of received packets at each relay could be more than the rate it can support to the destination, in which case $\mathcal{P}_F$ essentially makes a random selection of packets corresponding the maximum throughput it can support. The throughput of $\mathcal{P}_F$ can be calculated by evaluating the probability that a packet transmitted by the source in any given time slot reaches the destination along at least one of the m relays. Since these events are independent under our current hypothesis, we have Lemma 20, which is obtained from calculating the probability that any given packet reaches the destination via *at least* one of the relays.

Lemma 20.

$$C(\mathcal{P}_F) = 1 - \prod_{i \in [m]} (1 - \min(p(S,i), p(i,D)))$$

3.9 A Lower Bound for Independent Erasures

Using this flooding policy, we now prove a lower bound on C_R for a network with arbitrary fixed rates from the first hop forward. This argument explicitly bounds the loss of throughput because of the redundant transmissions arising out of making distributed decisions at each of the relay nodes with arbitrary min cuts when we have independence among link successes. The bound we provide also implies that when the capacity is low, the flooding is almost optimal for independent losses to the first hops. We will provide the proof on a 2-hop network for easier visualization, but the same claim holds for any fixed rates from the first hop to the destination.

Theorem 21. *Consider the relay network of Figure 3.3 with independent losses from the source to the relays.*

$$C_R \geq 1 - e^{-C^*}$$

Thus, $C_R/C^* \geq 1 - e^{-1}$*. It follows from this bound that, as* $C^* \to 0$*,* $C_R/C^* \to 1$

Proof. The argument is based on comparing the throughput of the flooding policy $C(\mathcal{P}_F)$ with C^*, since $\mathcal{P}_F$ is a policy that satisfies FIR. We will show that $C(\mathcal{P}_F) \geq 1 - e^{-C^*}$, which in turn implies the theorem since $\mathcal{P}_F$ clearly satisfies FIR (Definition 8).

Recall that:

$$C(\mathcal{P}_F) = 1 - \prod_{i \in [m]} (1 - \min(p(S,i), p(i,D)))$$

Applying the definition of the mincut (see Figure 3.3 for illustration), C^*, to the network under consideration, we see that

$$C^* = \min_{A \subseteq [m]} \left(1 - \prod_{i \in [m]/A} (1 - p(S,i)) + \sum_{i \in A} p(i,D) \right)$$

Let A^* be the arg min over $A \subseteq [m]$ for the above equation, so that:

$$C^* = 1 - \prod_{i \in [m]/A^*} (1 - p(S,i)) + \sum_{i \in A^*} p(i,D) \tag{3.16}$$

Consider any $i \in A^*$. By definition of A^*, we have:

$$1 - \prod_{j \in [m]/A^*} (1 - p(S,j)) + \sum_{j \in A^*} p(j,D)$$

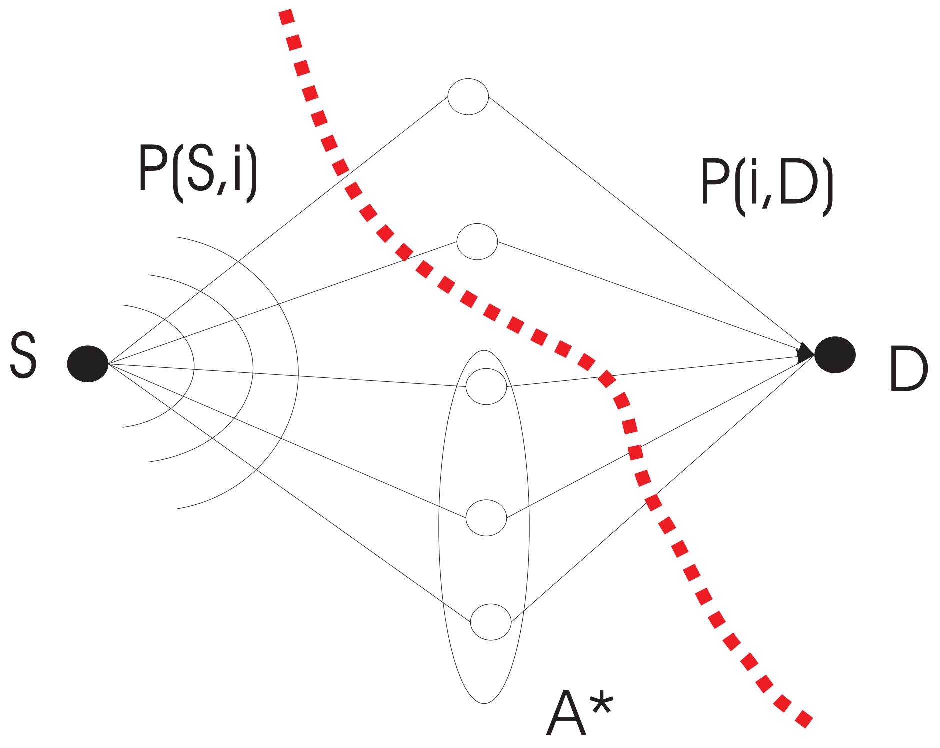


Figure 3.3: Illustration of a general min cut and the set defining A^* in this min cut.

$$\leq 1-(1-p(S,i))\prod_{j\in[m]/A^*}(1-p(S,j))+\sum_{j\in A^*}p(j,D)-p(i,D)$$

which implies:

$$p(i,D)\leq p(S,i)\prod_{j\in[m]/A^*}(1-p(S,j))\leq p(S,i) \tag{3.17}$$

Thus:

$$C(\mathcal{P}_F)=1-\prod_{i\in[m]}(1-\min(p(S,i),p(i,D)))$$
$$\geq 1-\prod_{i\in A^*}(1-\min(p(S,i),p(i,D))$$
$$=1-\prod_{i\in A^*}(1-p(i,D))$$
$$(\because\ p(i,D)\leq p(S,i)\ \ \forall\ i\in A^* \text{ from Equation (3.17)})$$
$$\geq 1-\prod_{i\in A^*}e^{-p(i,D)}\quad(\because\ 1-x\leq e^{-x}\ \ \forall\ x\geq 0)$$

$$
\begin{aligned}
&= 1 - e^{-\sum_{i \in A^*} p(i,D)} \\
&\geq 1 - e^{-C^*} \quad (\because \sum_{i \in A^*} p(i,D) \leq C^* \text{ from Equation (3.16)})
\end{aligned}
$$

□

The above analysis characterizes the degradation through the effects of feedback at a single node, given the rates achievable from the subsequent hops in any general network. We next build upon this argument to obtain a bound for arbitrary feed-forward networks where *every* node is restricted to static routing and without any assumption of achievable rates from the one hop neighbors.

Theorem 22. *For a general feed-forward network with $h+1$ hops, subject to feedback independent routing, the capacity is at least $f^h(C^*)$ where $f(x) = 1 - e^{-x}$ and C^* is the min cut capacity.*

Proof. The argument builds on the analysis in Theorem 21. Note that feed-forward networks can be reduced to general layered networks where only nodes at subsequent levels are connected, by introducing dummy nodes with unit capacity links. For such a layered network, we will adopt the convention that the sink is at level 0, and the source is at level $h+1$. We will define (i,j) to be the node indexed j at level i. In this notation, the source is assumed to be $(h+1,0)$, and the sink is $(0,0)$. Given any policy, $\mathcal{P}$, we define $\mathcal{P}(i,j)$ as the rate at which *distinct* packets are streaming to the sink through (i,j) . For example, $\mathcal{P}(h+1,0)$ is the throughput of the policy.

Consider ψ, the flooding policy satisfying FIR, and compare it with the splitting policy, $\mathcal{P}_{fs}$ in Definition 6: A node b queues the packets that it receives from node a in proportion to the rate $r^*(a,b) \doteq \sum_{Z \subseteq \mathcal{N}(a)} r^*(a,b,Z)$ with r^* as in Definition 6. Note that this is a static calculation and involves no feedback unlike $\mathcal{P}_{fs}$ or the backpressure policy, which requires queue length information to decide every routing step. For such a policy, we make the following claim, which can be shown using induction on i.

Claim 23. $\psi(i,j) \geq f^{i-1}(\mathcal{P}_{fs}(i,j))$

The claim follows easily for $i = 1$. Assume that it holds for some i. We will look at a node $(i+1,j)$ which is connected to $(i,j_1),\ldots,(i,j_k)$. Consider two different 2-hop networks both with $(i+1,j)$ as the source and

$(i, j_1), \ldots, (i, j_k)$ as relays with the same link characteristics as the feed-forward network for the first hop. To describe the second hop capacities, we will use the assumption that both $\mathcal{P}_{fs}$ and ψ choose the same fraction of the incoming capacities from various nodes for the forward throughput. Let α_t be the fraction of incoming rate that is chosen by node (i, j_t) for the throughput from $(i+1, j)$. For the second hop, the first and second networks have capacities $(\alpha_1\psi(i, j_1), \ldots, \alpha_k\psi(i, j_k))$ and $(\alpha_1\mathcal{P}_{fs}(i, j_1), \ldots, \alpha_k\mathcal{P}_{fs}(i, j_k))$ respectively. Let C_1^*, C_2^* be the respective C^*'s. Now, we can use Theorem 21 to argue that $\psi(i+1, j) \geq f(C_1^*)$. Using the induction hypothesis, we know that $\alpha_t\psi(i, j_t) \geq \alpha_t f^{i-1}(\mathcal{P}_{fs}(i, j_k))$. It can be shown under this condition (using Lemma 24) that $C_1^* \geq f^{i-1}(C_2^*)$, which implies that $\psi(i+1, j) \geq f(f^{i-1}(C_2^*)) = f^i(C_2^*) \geq f^i(\mathcal{P}_{fs}(i, j))$, thus completing the induction hypothesis. □

Lemma 24. *Following the notation used in Section 3.5, consider a two-hop relay network with parameters $c(Z)$ for the first hop and $r_i : i \in [m]$ as the rates for the second hop. Let the min cut of this network be C_1^*. Replace the second hop rates by $\alpha_i g(r_i/\alpha_i)$ where $g(x) = f^n(x)$ for some n with $f(x) = 1 - e^{-x}$. The new min cut $C_2^* \geq g(C_1^*)$.*

Proof. First note that for each $\alpha > 0$, for the given g, we have $\alpha_i g(r_i/\alpha) \geq g(r_i)$, which can again be shown using induction on the exponent of f corresponding to the given g. Therefore, it is sufficient to show that the network with second hop capacities, $g(r_i)$, has a mincut, $C \geq g(C_1^*)$. Let A^* be the set defining the min cut for the network defining C_1^* in the same sense as it was used in proof of Theorem 21. It can further be shown that $g(x_1 + \ldots + x_n) \leq g(x_1) + \ldots g(x_n)$ by using induction on the exponent of f in the representation of g. Using this fact (and denoting $\Theta = 1 - \prod_{i\in[m]/A^*}(1 - p(S, i))$ for convenience below), we then have: $C \geq \Theta + \sum_{i\in A^*} g(r_i) \geq g(\Theta) + \sum_{i\in A^*} g(r_i) \geq g(\Theta + \sum_{i\in A^*} r_i) = g(C_1^*)$ □

3.10 A Counterexample for Dependent Losses

From the previous section, one wonders if we are always assured of a constant factor reduction capacity in general, without either feedback or network coding on at least bounded diameter networks. Naively speaking, this seems plausible because correlations in the losses can only supply the relays with implicit information that might help them synchronize their routing more efficiently while satisfying FIR. This intuition is flawed, as the following counterexample shows that $C_R \to 0$ even though the capacity $C^* = 1$. This counterexample can be practically motivated as an extreme case of a situation where most of the time (with probability p to be explicated later) the broadcast channel is bad enough to be actually useful to a very small set of relays but occasionally experiences high strength, in which case most of the relays receive the broadcast. We make use of the characterization of C_R from Theorem 19 in showing this fact.

Definition 12 (Network, $\mathcal{N}$ of size m). *Let $\epsilon > 0 = \frac{1}{\sqrt{m}}$. When the source broadcasts, with probability ϵ, all relays receive the packet and with probability $1-\epsilon$, exactly one of the relays receives the packet.*

$$c(Z) = \begin{cases} \frac{1-\epsilon}{m} & \text{if } |Z| = 1 \\ \epsilon & \text{if } Z = [m] \\ 0 & \text{otherwise} \end{cases}$$

and for the relay to destination, we have:

$$p(i, D) = 1/m \quad \forall \quad i \in [m]$$

Theorem 25. *For the network $\mathcal{N}$, $C^* = 1$, but $\lim_{m\to\infty} C_R = 0$*

Proof. By computing the mincut, we have $C^* = 1$. We shall now derive an upper bound on the C_R which tends to 0 as $m \to \infty$ using Theorem 19. This implies the claim because of Theorem 19. Since $c(Z)$ depends only on $|Z|$, and the LP 3.12 is symmetric over i, one can set without loss of generality, $t(Z) = \phi(|Z|)$: $\sum_{Z,Z' \subseteq [m]: Z\cap Z' \neq \phi} t(Z)c(Z') = \sum_{k=1}^{m} \begin{pmatrix} m \\ k \end{pmatrix} \phi(k)(\sum_{Z':Z'\cap[k]\neq\phi} c(Z')) = \sum_{k=1}^{m} \phi(k) \begin{pmatrix} m \\ k \end{pmatrix} (\epsilon + (1-\epsilon)\frac{k}{m})$ The constraint in Equation (3.13) becomes:

$$\sum_{k=1}^{m} \begin{pmatrix} m \\ k \end{pmatrix} \phi(k) \leq 1 \tag{3.18}$$

As for the constraint in Equation (3.14), we have:

$$p(S,i) = \sum_{Z \subseteq [m]: i \in Z} c(Z) = \frac{1-\epsilon}{m} + \epsilon$$

$$\sum_{Z \subseteq [m]: i \in Z} t(Z) = \sum_{k=1}^{m} \binom{m-1}{k-1} \phi(k) = \frac{1}{m} \sum_{k=1}^{m} \binom{m}{k} k\phi(k)$$

Thus the second constraint becomes:

$$\sum_{k=1}^{m} \binom{m}{k} k\phi(k) \leq \frac{m}{1-\epsilon+m\epsilon} \tag{3.19}$$

Thus,

$$\begin{aligned}
C_T &= \sum_{k=1}^{m} \phi(k) \binom{m}{k} (\epsilon + (1-\epsilon)\frac{k}{m}) \\
&= \epsilon \left(\sum_{k=1}^{m} \binom{m}{k} \phi(k) \right) + \frac{1-\epsilon}{m} \left(\sum_{k=1}^{m} \binom{m}{k} k\phi(k) \right) \\
&\leq \epsilon + \frac{1-\epsilon}{1-\epsilon+m\epsilon} \quad (\text{ from Equations (3.18), (3.19)}) \\
&= o(1/\sqrt{m}) \quad (\text{ since } \epsilon = \frac{1}{\sqrt{m}})
\end{aligned}$$

□

3.11 Conclusion

While network coding is necessary to achieve the maximum throughput for multicast connections, this is not the case with wireless unicast. Rather, network coding is a convenient way to solve the distributed routing problem without having to depend on feedback signaling to make complicated routing choices for achieving the maximum throughput. In this context, we analyzed a relay network and quantitatively characterized the limitations of 'static' routing policies that operate in a feedback independent manner. Our characterization allows for explicitly identifying situations when there is no loss of throughput by restricting to such simple routing policies. Further, we show that the reduction in the throughput is controlled when the link losses for a given transmission are independent, and could even be minimal when the capacity is low, on a general feed-forward network. At worst, this is 63% and gets progressively close to 100%, as the capacity itself goes to 0. Thus, in such a situation, network coding delivers no benefit over simpler blind routing policies in the limit of unreliable communication. Nevertheless, highly correlated losses could lead to an unbounded loss even on a 2-hop network. In such a situation, network coding might be unavoidable if we need to be conservative with the feedback.

CHAPTER 4

OPTIMAL CONTROL OF A BROADCASTING SERVER

4.1 Introduction

Stochastic control problems on queues controlled by a broadcasting server are considered in this chapter. By "broadcasting server," what we mean is an abstraction in which each service clears all the customers present in the queue at once; i.e., the number of customers is reset to zero at any moment at which the broadcast server fires. More precisely, we consider a system with a dynamic audience interested in a common broadcast from a central server. This can be modeled as a queuing system in continuous time with customers (the audience) arriving according to a Poisson process. The server has the ability to service the audience with broadcasts separated by an exponentially distributed random duration, whose maximum rate is μ (and can be controlled to any value between 0 and μ). Whenever a broadcast is made, the entire audience present in the system at that instance is served (i.e., the total number of customers is reduced to 0 at that instance). There is a cost for holding customers in the system, which is specified by a cost rate function that depends on the number of customers in the system at any given time. The holding cost (rate) function could be monotone or even non-monotone but convex. Our aim is to minimize the infinite horizon discounted cost, and to understand the structure of the associated optimal policies. This optimization is also subject to constraints on the server, for which we consider two models, discussed next.

4.2 Contributions and Organization

In the first model, considered in Sections 4.4, each broadcast is charged a fixed non-negative instantaneous cost, in addition to the holding cost, which is accumulated continuously over time. This defines a Markov decision problem (MDP) where the state at any given time is the number of customers in the queue. In Section 4.4, we consider convex holding costs with a fixed broadcast cost. We show that the optimal control for the infinite horizon discounted problem is of the threshold type. In other words, operate the server at the maximum rate μ if the number of customers exceeds a given threshold, and at rate zero otherwise. This result complements the existing literature on batch processing queuing models that have typically only considered monotone holding costs. A more general problem is to consider multiple classes of customers (each belonging to a separate queue) where each service clears all customers of only one given class. In this case, the control problem includes the decision of which queue to serve at any given time.

In the second model, considered in Section 4.5, we replace the cost charged for each service with an online hard constraint on the number of broadcasts. To model this, we consider a "resource queue" associated with the broadcast server, with a fixed rate of arrivals, which is depleted corresponding to each service based on the rate at which the service is delivered. The constraint we consider restricts the ability to operate the server subject to having a positive resource balance. There is no explicit cost associated with the service. In contrast with the first model, where a control can be chosen at each instant of time, we now restrict the ability to choose the rate[1] at which the broadcast server fires to a single instance at the beginning of the interval between the broadcasts. This transforms the problem into an MDP where the state is the balance of the resource queue at the given time instance. For this model, we will only address the single queue case. In Section 4.6 we step back and address a general optimization problem that addresses a class of problems for which we have an online hard constraint on the control actions over time. We obtain a limiting result on the optimal value function and control where the discount factor is small. This result addresses a setting where the individual costs corresponding to each control action are negligible in comparison to the

[1]I.e., the parameter that defines the exponential distribution of the time taken for the broadcast server to fire.

long-term discounted cost. We then apply this result to the broadcast server problem under consideration in Section 4.5 and provide an explicit solution and numerical plots for linear holding costs.

4.3 Motivation

A few motivating scenarios for the general broadcast queuing model (i.e., each service clears the entire queue) and for the cost models we consider are given next before we proceed further.

Queuing systems have been studied under the related batch processing model. In this model, corresponding to a batch size of B, each service to the queue can clear a maximum of B customers. For broadcast, $B = \infty$. Deb and Serfozo [79] considered the stochastic control problem of a single queue under batch processing, and proved that the optimal control is of threshold type when the instantaneous cost function is monotone in the number of customers. There is extensive literature on the batch processing model for a single queue [80–84]; however, a common theme in all previous work is that the holding cost is generally monotone in the number of customers. On a related note, [85] proved that a threshold control is optimal for the standard queuing service model (i.e. each service clears one customer) with convex holding cost in the number of customers, and without any service cost.

Motivation for Non-Monotone Cost: A non-monotone cost model is relevant to a situation where the server has a strategic interest in preventing the audience interested in its broadcast from being as low as possible at all times. Such a possibility could arise in a peer to peer service model with selfish peers, where having too few interested peers on average incurs a high cost for the server because the server is itself predominantly only served by other selfish peers that are actively interested in its own service. On the other hand, keeping too many peers waiting also incurs a progressively higher cost beyond a certain point, because the server then risks being classified as a freeloader by its peers, leading to punishment from its frustrated peers in the form of degraded service to itself.

The general broadcast scheduling problem arises in applications where a central server has multiple pages with customer requests for each page arriving independently. Each service can satisfy all outstanding requests for any single page. The aim is usually to minimize either the average waiting time for the page requests, or to minimize the maximum waiting time. A large body of work in the database and algorithms literature has focused on scheduling for the broadcasting model (e.g [86–91]). The emphasis is on competitive analysis, in which oblivious online policies and optimal offline algorithms that have a precise knowledge of the future sample path of the customer arrivals are analyzed. Xia et al. [92] address the batch processing problem in multiple queues. A variant with constant service time was studied also in [93].

4.4 Broadcast Server with Fixed Service Cost

In this section, we consider a customer queue with arrivals defined by a Poisson process of rate λ. Whenever the queue is served, all customers in the queue exit at once. Without loss of generality, assume that $\lambda + \mu = 1$. Let $c : Z_+ \mapsto R_+$ denote a non-negative holding cost rate defined as a function of the number of customers in the queue, which we denote as x_t at time t. We assume that c is convex and also has an appropriate growth restriction to ensure that the infinite horizon discounted cost is well defined. Let c_{sw} denote the constant service cost associated with each broadcast. Assume that for a given time a control value, $0 \leq w \leq 1$, specifies the (exponential) rate of the broadcast as being $w\mu$. A useful way of interpreting this situation is to look at the rate μ process consistently at all times, and then actually utilize this *potential* broadcast opportunity with probability w when the control being applied is w, an equivalence which follows directly from the Poisson splitting property.

Now consider a rate 1 Poisson process obtained by adding the arrival process of rate λ and the potential departure process of rate μ. Let τ_n be the n^{th} transition of this net process and let x_n denote x_{τ_n} for simplicity. The discrete time jump Markov process obtained by sampling the system between these transitions has the following transition probabilities defined on Z_+ (this discrete time process is independent of the inter-event times): $p(y/x) = \lambda I\{y = x + 1\} + \mu(w(x)I\{y = 0\} + (1 - w(x))I\{y = x\}$, where w denotes the stationary, feedback control as a function of the current state.

The infinite horizon discounted cost to be minimized is:

$$E_x^w \int_0^\infty e^{-\alpha t} c(x_t)\, dt + \sum_{k=1}^{\infty} e^{-\alpha \tau_k} \chi\{x_{k-1} \neq 0 \text{ and } x_k = 0\} c_{sw}$$

where E_x^w denotes the expectation with the control w starting at $x_0 = x$, α is the discount factor and χ denotes the indicator function for the conditions given in its argument, and c_{sw} is the non-negative service cost associated with each broadcast. The above expression can be shown to be a constant factor of the equivalent cost on the discrete time process by invoking the independence between the inter-event times and the jump process dynamics [94]. This gives us an equivalent optimization on the discrete time Markov decision process defined above with the following objective function:

$$E_x^w \sum_{k=0}^{\infty} \beta^k \left(c(x_k) + c_s \chi\{x_{k-1} \neq 0, x_k = 0\} \right)$$

where $0 < \beta < 1$ is the discount factor and c_s (rather than c_{sw}) is the service cost for the equivalent discrete time problem. We shall also use a convention that $x_{-1} = 0$. Let w be a $[0,1]$ valued function on Z_+ denoting the control (i.e., $w(x)\mu$ is the rate of the broadcast server with x customers in the queue). Let U^w denote the value function corresponding to the control w:

$$U^w(x) = E_x^w \sum_{k=0}^{\infty} \beta^k \left(c(x_k) + c_s \chi\{x_{k-1} \neq 0, x_k = 0\} \right) \tag{4.1}$$

By a simple recursion argument, it can be shown that this value function satisfies a fixed point equation corresponding to the dynamic programming operator for U^w, given by

$$\mathcal{T}^w f(x) = c(x) + \beta(\lambda f(x+1) + \mu(w(x)(f(0) + c_s) + (1 - w(x))f(x)))$$

In other words,

$$U^w = \mathcal{T}^w U^w \tag{4.2}$$

Further, uniqueness of a solution to the above equation is implied by fixed point theorem for contractions in complete metric spaces, provided we assume the appropriate growth restrictions on $c(x)$ as in [85,94] (this is not restrictive, unless we need to model a situation with super exponential costs). Let $V(x)$ be the optimal value function, defined as the infimum over arbitrary control policies u, of the expected infinite horizon discounted cost starting at x :

$$V(x) = \inf_u E_x^u \sum_{k=0}^{\infty} \beta^k (c(x_k) + c_s \chi\{x_{k-1} \neq 0, x_k = 0\}) \tag{4.3}$$

The fixed point equation operator for V is:

$$\mathcal{T}f(x) = c(x) + \beta\{\lambda f(x+1) + \mu \min(f(x), f(0) + c_s)\} \tag{4.4}$$

V is the unique solution to:

$$V = \mathcal{T}V \tag{4.5}$$

Given the optimal V, an optimal control w would then be:

$$w(x) = \begin{cases} 1 & \text{if } V(x) > V(0) + c_s \\ 0 & \text{if } V(x) \leq V(0) + c_s \end{cases}$$

The main result we prove in this section is as follows:

Theorem 26. *The optimal control is given by the stationary state feedback control $w(x) = I\{x \geq l^*\}$ for a critical threshold l^*. Further,*

$$l^* = \min\{l : U_l(l) > U_l(0) + c_s\}$$

To denote w of the form $w(x) = I\{x \geq l\}$, we will from now on write it as w_l and the value function corresponding to it as U_l. It suffices to show that Equation (4.5) is satisfied for the operator T^w corresponding to w defined by a threshold control in order to show that it is an optimal control. First, we recall the definition of a quasiconvex (unimin) function:

Definition 13. *A function f on Z_+ is quasiconvex (unimin) if $f(x+1) - f(x) \geq 0$ for all $x > y$ whenever $f(y+1) - f(y) > 0$.*

A key element of the argument is to show that:

Theorem 27. *If $U_l(l-1) \leq U_l(0) + c_s < U_l(l)$, then U_l is quasiconvex.*

Proof. Implied by Lemmas 28, 30 and the hypothesis that $U_l(l-1) < U_l(l)$. □

Lemma 28. *If $U_l(l-1) \leq U_l(l)$, then $U_l(x)$ is quasiconvex on $0 \leq x \leq l-1$.*

Proof. Within this lemma, we will drop the subscript l and have the convention that U means U_l, and denote $\frac{\beta\lambda}{1-\beta\mu} = \gamma$ (note that $0 < \gamma < 1$). Further, we denote $c'(x) = c(x)/(1-\beta\mu)$. For $0 \leq x \leq l-1$, $U(x) = c(x) + \beta\lambda U(x+1) + \beta\mu U(x)$, which in turn implies:

$$\begin{aligned} U(x) &= c'(x) + \gamma U(x+1) \\ &= c'(x) + \gamma\{c'(x+1) + \gamma U(x+2)\} \\ &\dots \\ &= c'(x) + \gamma c'(x+1) + \gamma^2 c'(x+2) + \ldots + \gamma^{l-x-1} c'(l-1) + \gamma^{l-x} U(l) \end{aligned}$$

Let $\delta'(x) \triangleq c'(x+1) - c'(x)$. δ' is increasing since c' is convex. Also define

$$\Delta U(x) \triangleq U(x+1) - U(x)$$

Using the above definition, and from the fact that

$$U(l-1) = c'(l-1) + \gamma U(l)$$

one can verify the following relation for $0 \leq x \leq l-2$:

$$\Delta U(x) = \delta'(x) + \gamma\delta'(x+1) + \ldots + \gamma^{l-x-2}\delta'(l-2) + \gamma^{l-x-1}(U(l) - U(l-1))$$

A sufficient condition for $U(x)$ to be quasiconvex on $0 \leq x \leq l-1$ is the existence of a $\xi(x) > 0$ such that $\frac{\Delta U(x)}{\xi(x)}$ is increasing[2] for $0 \leq x \leq l-2$. We will now show this for the choice of $\xi(x) = 1 - \gamma^{l-x-1} > 0$ for $0 \leq x \leq l-2$. Since $\frac{\Delta U(x)}{\xi(x)}$ depends on the function δ' in a linear fashion, we just need to

[2]Throughout this chapter, 'increasing' and 'decreasing' mean 'non-decreasing' and 'non-increasing' respectively.

verify that $\frac{\Delta U(x)}{\xi(x)}$ is increasing when δ' is a constant, and when it is of the form $I\{x \geq b\}$. First if $\delta'(x) = a$ for any constant a, we have:

$$\begin{aligned}\frac{\Delta U(x)}{\xi(x)} &= a\frac{1+\gamma+\ldots+\gamma^{l-x-2}}{1-\gamma^{l-x-1}} + \frac{\gamma^{l-x-1}}{1-\gamma^{l-x-1}}(U(l)-U(l-1)) \\ &= \frac{a}{1-\gamma} + \frac{\gamma^{l-1}}{\gamma^{x}-\gamma^{l-1}}(U(l)-U(l-1))\end{aligned}$$

which is increasing in x since (1) $\gamma < 1$ and (2) $U(l) > U(l-1)$, by the hypothesis of the lemma. Next let $\delta'(x) = I\{x \geq b\}$. We then have (with the convention that if $b > l-2$, the appropriate terms below will be 0, and hence increasing by default):

$$\begin{aligned}\frac{\Delta U(x)}{\xi(x)} &= \frac{\gamma^{b-x}+\ldots+\gamma^{l-x-2}}{1-\gamma^{l-x-1}} + \frac{\gamma^{l-x-1}}{1-\gamma^{l-x-1}}(U(l)-U(l-1)) \\ &= \frac{\gamma^{b}+\ldots+\gamma^{l-2}}{\gamma^{x}-\gamma^{l-1}} + \frac{\gamma^{l-1}}{\gamma^{x}-\gamma^{l-1}}(U(l)-U(l-1))\end{aligned}$$

which is again increasing with x by the hypothesis of the lemma. □

Lemma 29. *$U_l(x)$ is quasiconvex for $x \geq l$.*

Proof. This can be proved by considering a coupled process and using an argument similar to that in [85]. Given an integer l, define a Markov process on Z_+^2 with the following transitions:

$$\begin{aligned}p((x',y')|(x,y)) = \lambda I\{x'=x+1, y'=y+1\}+ \\ \mu I\{x'=x(1-I\{x\geq l\}), y'=y(1-I\{y \geq l\})\}\end{aligned}$$

Then, corresponding to any initial state (x, y) the above coupled process has marginals which are identical to the individual processes. Although the condition (3) in [85] does not hold anymore, restrict attention to any $x \geq l$ and consider the process started in $(x, x+1)$. Then, $y_k - x_k$ takes values in $0, 1$ and is decreasing in k. Let E_x denote the expectation under this starting condition. Let

$$\tau = \min\{k \geq 0 : x_k = y_k\}$$

As in [85], let:

$$r(x) = \frac{U(x+1) - U(x)}{E_x \sum_{k=0}^{\tau-1} \beta^k}$$

On $x \geq l$, it can be shown that r is increasing, which implies U is quasiconvex on the same domain. For $x \geq l$:

$$U(x+1) - U(x) = E_x \sum_{k=0}^{\infty} \beta^k (c(y_k) - c(x_k)) = E_x \sum_{k=0}^{\tau-1} \beta^k \delta(x_k) \tag{4.6}$$

so that:

$$r(x) = \frac{E_x \sum_{k=0}^{\tau-1} \beta^k \delta(x_k)}{E_x \sum_{k=0}^{\tau-1} \beta^k}$$

Since δ is increasing and since r depends linearly on δ, it suffices to verify that it is increasing for constants and for functions of the form $I\{x \geq b\}$. Since r is a constant if δ is a constant or if $\delta(x) = I\{x \geq b\}$ where $b \leq x$ since $x \geq l$, we only have to check for $I\{x \geq b\}$ where $b > x$. Let

$$\sigma = \tau \wedge \min\{k : x_k = x+1\}$$

Then $r(x)$ can be compared favorably with $r(x+1)$ by writing (using $b > x$):

$$E_x \sum_{k=0}^{\tau-1} I\{x_k \geq b\} \beta^k = E_x[\beta^\sigma I\{\sigma < \tau\}] E_{x+1} \sum_{k=0}^{\tau-1} I\{x \geq b\} \beta^k$$

and

$$E_x \sum_{k=0}^{\tau-1} \beta^k = E_x \sum_{k=0}^{\sigma-1} \beta^k + E_x[\beta^\sigma I\{\sigma < \tau\}] E_{x+1} \sum_{k=0}^{\tau-1} \beta^k$$

$\square$

Lemma 30. *If $U_l(l-1) \leq U_l(0) + c_s < U_l(l)$, then $U_l(x)$ is increasing on $x \geq l$.*

Proof. We already know that U_l is quasiconvex for $l \leq x$ from Lemma 29. Hence, it is sufficient to show that $U_l(l+1) > U_l(l)$ to prove that it is increasing on $x \geq l$. For the rest of the proof in this Lemma, we will again implicitly drop the subscript l in U_l. Assume to the contrary that $U(l+1) \leq$

$U(l)$. Then:

$$\begin{aligned}U(l) &= c(l) + \beta\{\lambda U(l+1) + \mu(U(0) + c_s)\}\\ &\leq c(l) + \beta U(l)\, (\because U(0) + c_s < U(l), U(l+1) \leq U(l))\end{aligned}$$

$$\begin{aligned}U(l-1) &= c(l-1) + \beta\{\lambda U(l) + \mu U(l-1)\}\\ &\geq c(l-1) + \beta U(l-1) \quad (\because\; U(l) > U(l-1))\end{aligned}$$

The above two inequalities imply:

$$\begin{aligned}&\Delta U(l-1) \leq \delta(l-1) + \beta \Delta U(l-1)\\ &\Rightarrow 0 < (1-\beta)\Delta U(l-1) \leq \delta(l-1)\end{aligned}$$

Since δ is increasing, this also means that $\delta(l) > 0$. Then, again by a certain coupling argument similar to Lemma 29, we have $U(l+1) > U(l)$. More precisely, consider the coupled process of the proof of Lemma 29. For $x_0 = l$ and τ, the stopping time as defined in proof of Lemma 29, x_k is an increasing sequence for $0 \leq k \leq \tau - 1$. Hence, $\delta(x_k) \geq \delta(l) > 0$ for $0 \leq k \leq \tau - 1$. Thus, using Equation (4.6), $\Delta U(l) = E_{x_0=l} \sum_{k=0}^{\tau-1} \beta^k \delta(x_k) > 0$, which in turn implies that $U_l(x)$ is increasing on $x \geq l$. □

Lemma 31. *If $c(x)$ is convex, unless it is decreasing on all x, we have for some l large enough: $U_l(l) > U_l(0) + c_s$.*

Proof. Suppose $U_i(i) \leq U_i(0) + c_s$ for all $i < l$. For any $i < l$, consider a Markov decision problem where the only decision variable is at state $x = i$, with the rest of the control fixed to match the threshold $i+1$ control, w_{i+1}. Now apply policy iteration to the threshold i policy on the above MDP. Since $U_i(i) \leq U_i(0) + c_s$, an optimal control is to set $w(i) = 0$ (i.e., do not serve at $x = i$), and policy iteration results in w_{i+1}, the threshold-$i+1$ control. Hence, U_{i+1} is component wise less than U_i. Specifically, this means $U_i(0)$ is a decreasing sequence for $i \leq l$, implying that $U_l(0) \leq U_1(0)$. Since c is not decreasing and is convex, for some l large enough, we have $c(l) > U_1(0) + c_s$. For such an l, if we also have $U_i(i) \leq U_i(0) + c_s$ for all $i < l$, then $U_l(l) = c(l) + \beta\{\lambda U_l(l+1) + \mu(U_l(0) + c_s)\} > c(l) \geq U_1(0) + c_s \geq U_l(0) + c_s$. □

Theorem 32. *The optimal control is a threshold policy corresponding to the threshold l^* given by*

$$l^* = \min\{l : U_l(l) > U_l(0) + c_s\} \tag{4.7}$$

Proof. Suppose l^* is infinite. Then, the contrapositive of Lemma 31 implies that $c(x)$ is decreasing, in which case it is clear that an optimal policy is to never serve, which corresponds to a threshold $l^* = \infty$ optimal control. Thus, we now only need to argue about the case where l^* is finite. Now suppose

$$U_{l^*}(l^* - 1) > U_{l^*}(0) + c_s \tag{4.8}$$

Then consider the Markov decision subproblem where the the control is fixed to match the threshold l^* control for all x except for $x = l^* - 1$, which is the only decision variable. Then, by an application of policy iteration for this subproblem, we conclude that U_{l^*-1} strictly improves U_{l^*}. Now if the following is true:

$$U_{l^*-1}(l^* - 1) \leq U_{l^*-1}(0) + c_s \tag{4.9}$$

then we can again consider the Markov decision subproblem where the only decision variable is at $x = l^* - 1$ and everything else is fixed to match the threshold l^* control. Such consideration implies that U_{l^*} improves U_{l^*-1}, a contradiction to what we just concluded above. Hence Equation (4.9) must be false and

$$U_{l^*-1}(l^* - 1) > U_{l^*-1}(0) + c_s$$

which contradicts the definition of l^* in Equation (4.7). Hence, the assumption in Equation (4.8) is false and we conclude that:

$$U_{l^*}(l^* - 1) \leq U_{l^*}(0) + c_s$$

This means that l^* satisfies the hypothesis for Theorem 27 and hence is quasiconvex. This implies:

$$U_{l^*}(x) \begin{cases} \leq U_{l^*}(0) + c_s & \text{if } x \leq l^* - 1 \\ > U_{l^*}(0) + c_s & \text{if } x \geq l^* \end{cases}$$

Therefore, it also satisfies the fixed point equation corresponding to the optimal value function dynamic programming operator given in Equation (4.4). □

4.5 Broadcast Server with an Online Constraint

In this section, we are interested in a problem where there is no explicit cost per broadcast, but rather have a long-term constraint on the amount of resources that can be used by the server. As before, we consider a continuous time model. We then divide the continuous time into disjoint intervals separated by each broadcast. We restrict the opportunity to make a control decision to the beginning of each such time interval so as to formulate a discrete time Markov decision problem whose state is the balance of resources available for the server.

Let x_t denote the number of customers waiting in the queue at time t. Let t_i be the time instant at which the broadcast server fires for the i^{th} time. We define $[t_i, t_{i+1})$ as interval i. Note that $x_{t_i} = 0 \quad \forall i$. The process x_t is defined by a Poisson process of intensity λ_i during time interval i, where λ_i is a bounded i.i.d. random variable for each i. At the beginning of each time interval, the server makes a control decision $u_i \in [0, \mu]$ as the rate of the broadcast server. This determines the length of the corresponding i^{th} time interval to follow an exponential random variable with rate equal to u_i. Associated with the server, we have a "balance" queue, $b(i)$, which evolves every time interval with the following dynamics:

$$b(i+1) = b(i) + \mathbf{a}(i) - u_i$$

where $\mathbf{a}(i)$ is an i.i.d. (over i) bounded random variable, with mean equal to $\mathbf{E}[\mathbf{a}(i)] = a \geq 0$. The server gets to observe the arrival rate λ_i and the resource balance $b(i)$ prior to making the control decision during interval i. The cost of broadcast interval i is then defined to be $c_i \triangleq \frac{1}{t_{i+1}-t_i} \int_{t_i}^{t_{i+1}} c(x_t)dt$. Finally, for a discount factor $\beta > 0$, the total cost we wish to minimize is: $\sum_{i=0}^{\infty} e^{-\beta i} c_i$.

Given an initial balance, the objective is to minimize the total expected cost defined above subject to the hard constraint that $b(i) \geq 0 \quad, \forall i$. When

$\beta \to 0$, we provide a method to approximately solve for this problem by solving a more general problem with a similar constraint in the next section.

4.6 Generalized Online Knapsack Problems

In this section, we describe a general class of online stochastic control problems. We show that this class of problems includes online stochastic knapsack problems and some extensions, such as those faced by a bidder in repeated second-price auctions, and repeated generalized second-price auctions. In the problems considered, actions taken by a single decision maker generate utility by consuming a resource that depends on a random and partially observable environment. The objective of the decision maker is to take successive actions so as to maximize her infinite horizon discounted utility subject to a constraint of maintaining a positive balance of the resource at all times.

4.6.1 A Discrete Time Continuous State Markov Decision Model

Time. Time is discrete and indexed by $i = 0, 1, 2, \ldots$

Random environment. A random environment impacts the pay-offs and payments. This environment is assumed to be i.i.d. over time. At time i, the environment is described by a random variable $\xi(i)$ with values in $\mathbb{R}^n$. $\xi(i)$, $i = 0, 1, \ldots$ are independent and identically distributed. A *generic* random variable ξ represents the typical environment, i.e., it has the same distribution as $\xi(i)$ for any i. Let $\mathcal{F}$ be the sigma algebra generated by ξ. A part of the random environment is *observable.* Let $\mathcal{F}_0$ denote the sigma algebra (included in $\mathcal{F}$) that represents the observable part of the generic random environment.

Actions. At each time, the decision maker chooses an action u from a compact set U. The decision maker is able to partially observe the realization of the environment for this time period corresponding to the sigma algebra $\mathcal{F}_0$, and base her decision on this partial knowledge. Formally, the action of the decision maker can be represented by a random variable $\mathbf{u}$, that is measurable with respect to $\mathcal{F}_0$. As a special case, we can imagine that $\xi = (\xi_0, \xi_1)$, and that the decision maker knows ξ_0 before taking decision u. In such a case, $\mathcal{F}_0$ is the σ-algebra generated by random variable ξ_0.

Utility. Each action of the decision maker leads to an instantaneous utility which also depends on the realization of the random environment. It is represented by a measurable, continuous and bounded function, $g : U \times \mathbb{R}^n \to \mathbb{R}$. The net utility obtained is the infinite horizon discounted utility, $\sum_{i=0}^{\infty} e^{-\beta i} g(\mathbf{u}(i), \xi(i))$.

Resource constraint. The ability of the decision maker to take actions that maximize her utility is restricted by a resource constraint. At each time slot, the action taken by the decision maker, together with the realization of the random environment leads to a consumption of the resource, defined by a measurable, positive, bounded, continuous function given by $c : U \times \mathbb{R}^n \to \mathbb{R}_+$. It is also assumed that there exists an action which avoids consumption of the resource, i.e. $\exists\ \mathbf{0} \in U$ such that $c(\mathbf{0}, \xi) = 0\ \forall\ \xi$. Besides the consumption, the resource is also incremented by a fixed amount a in each time slot. The initial balance at time 0 is given by b. Let $\mathbf{b}(i)$ denote the balance of the resource available at the beginning of time slot i. The evolution of the resource balance can be written as:

$$\mathbf{b}(i+1) = \mathbf{b}(i) + a - c(\mathbf{u}(i), \xi(i)), \quad \mathbf{b}(0) = b \tag{4.10}$$

The decision maker is forbidden from taking any sequence of actions that lead to a negative balance of the resource at any point.

Objective. Consider a discrete time Markov decision process (MDP) on the continuous state space, $\mathbb{R}_+$ representing the resource balance. Let $\mathbf{u} \in \mathcal{F}_0$ denote a random variable $\mathbf{u}$ measurable in $\mathcal{F}_0$. Let $\mathcal{U}$ represent the collection of all admissible Markov policies (i.e. $\mathcal{U}$ represents all sequences of actions, defined by random variables $\mathbf{u}(i) \in \mathcal{F}_0$, which can be chosen after observing the corresponding states, $\mathbf{b}(i)$, with the additional restriction that any action $\mathbf{u}(i)$ which leads to a strictly positive probability on the event $\mathbf{b}(i+1) < 0$ is forbidden). Denote $\bar{g}(\mathbf{u}) \triangleq \mathbb{E}[g(\mathbf{u}, \xi)]$ and $\bar{c}(\mathbf{u}) \triangleq \mathbb{E}[c(\mathbf{u}, \xi)]$. Then, the value function with initial balance $\mathbf{b}(0) = b$ is given by:

$$v_\beta(b) = \sup_{\mathcal{U}} \left(\mathbb{E}_b \sum_{i=0}^{\infty} e^{-\beta i} \bar{g}(\mathbf{u}(i)) \right) \tag{4.11}$$

4.6.2 Examples

Example 1 – (Knapsack problem) Consider an online stochastic knapsack problem with a sequence of i.i.d. objects, each with an i.i.d. payoff represented by a generic random variable v and a weight represented by a generic random variable w. At time i, the decision maker needs to make a choice of whether to include the object in the knapsack or not with indicator variable in $u(i) \in U = \{0, 1\}$; $a = 0$; $\xi = (v, w) \in \mathbb{R}^2$, with $g(u, \xi) = uv$ and $c(u, \xi) = uw$. The random environment is also completely observable, with $\mathcal{F}_0 = \mathcal{F}$, i.e. $u(i)$ can be decided after observing both $w(i)$ and $v(i)$ and the remaining knapsack capacity, given by $b - \sum_{j=1}^{i-1} u(j)w(j)$. The MDP in this case with initial state b therefore corresponds to the optimization problem:

$$\sup \sum_{i=0}^{\infty} e^{-\beta i} \mathbb{E}[u(i)v(i)],$$
$$\text{s.t.} \sum_{i=0}^{\infty} u(i)w(i) \leq b.$$

Example 2 – (Repeated second-price auctions with budget constraints) The decision maker is a bidder participating in a sequence of second price auctions. The bidder wishes to optimize her bid based on a belief about the opponent bid modeled according to some probability distribution. The bidder assumes that her optimization horizon lasts for a large random duration modeled according to the memoryless exponential distribution with mean $1/\beta$, where $\beta > 0$ is a fixed parameter.

Random environment is modeled as i.i.d. with $\xi = (\mathbf{v}, \mathbf{b})$, where $\mathbf{v}$ represents the valuation of the object/service being auctioned in each time slot, and $\mathbf{b}$ denotes the competing bid. The *observable part* of the random environment for the bidder is the self valuation of the object, i.e. $\mathcal{F}_0 = \sigma(v)$, the sigma algebra generated by the random variable v.

Actions and Utility: For any given time, the action, $\mathbf{u}$, is the bid. The bidder wins the auction when $\mathbf{u}$ exceeds the competing bid, in which case, the utility derived is defined[3] as $g(\mathbf{u}, \xi) = \mathbb{1}_{\{\mathbf{u} > \mathbf{b}\}}(\mathbf{v} - \mathbf{b})$.

[3]In this case, the payment made is subtracted from the object's valuation to define the utility because the object's valuation is assumed to be measured in monetary units, and the value function represents the optimal "monetary surplus" that can be generated from a given initial balance.

Budget constraint: The bidder is assumed to have an *income* stream equivalent to a per unit time slot. The amount paid at the auction is given by $c(\mathbf{u}, \xi) = \mathbb{1}_{\{\mathbf{u}>\mathbf{b}\}}\mathbf{b}$.[4] The bidders are restricted to participate in the auctions subject to their ability to pay the maximum possible amount in any given auction.

The MDP in this case involves solving for optimal bidding strategies to maximize the infinite horizon discounted surplus monetary utility that can be generated with a given capital and a fixed income stream, which is part of our forthcoming paper [22].

4.6.3 Statement of Theorem and an Application to the Broadcast Server Problem

The theorem stated in this section provides a method to approximate the value function and the associated optimal control when the discount factor β is close to zero. This represents a situation where the discount factor is effective only over a large number of time slots. An alternate equivalent view is that when $\beta \to 0$, the value function is unbounded; therefore, this represents a situation where the magnitude of transactions in individual time slots is minuscule in comparison to the infinite horizon valuations. As far as the notation regarding balances and valuations is concerned, the following mnemonic is used: A capital letter variable indicates its corresponding small letter version scaled down by a factor of β. Therefore,

$$V_\beta(B) \triangleq \beta v_\beta(b) = \beta v_\beta(B/\beta)$$

where $v_\beta(b)$ is defined as the value function of the MDP in Equation (4.11).

Theorem 33. *Consider the MDP defined in Section 4.6.1. Let $\phi : \mathbb{R}_+ \mapsto \mathbb{R}$ be defined as:*

$$\phi(x) = ax + \sup_{u \in \mathcal{F}_0} (\bar{g}(u) - \bar{c}(u)x) \tag{4.12}$$

ϕ is a convex and Lipschitz function with a minimum denoted $\eta_* \triangleq \min_{x\geq 0} \phi(x)$. Let $f : \mathbb{R} \mapsto \mathbb{R}_+$ be an inverse function to ϕ defined as:$f(y) = \min\{x \geq 0 :$

[4]Continuity of c and g will hold in this case if we assume that the total possible bids are finite, to avoid unnecessary technicalities.

$\phi(x) = y\}$. *Then, for all* $B \geq 0$, $V(B) = \lim_{\beta \to 0} \beta v_\beta(B/\beta)$ *is well defined, and satisfies the ODE:*

$$\frac{dV}{dB} = f(V), \quad V(0) = \eta_* \tag{4.13}$$

We now apply the theorem to the broadcast control problem under consideration in Section 4.5 for computing the optimal policy in the limiting scenario where β is small. Let $c_h(x)$ denote a holding cost rate function associated with x customers in the queue waiting for service. To ease computations, we assume that $c_h(x) = x$ in the below calculations. For a generic interval that lasts for a duration T (where we count time from 0 at the beginning of the interval) and has customer arrivals of rate λ denoted by the Poisson process $\{N_t : t \geq 0\}$, we calculate the expected cost as (let t_j denote the arrival time of the j^{th} customer, with $t_0 = 0$ and $t_{k+1} = T$ below):

$$\begin{aligned}
&\frac{1}{T}\sum_{k=0}^{\infty} P[N_T = k]\,\mathbf{E}\left[\int_0^T c_h(N_t)dt|N_T = k\right] \\
=&\frac{1}{T}\sum_{k=0}^{\infty} P[N_T = k]\,\mathbf{E}\left[\sum_{j=0}^{k} c_h(j)(t_{j+1} - t_j)|N_T = k\right] \\
=&\sum_{k=0}^{\infty} P[N_T = k]\left(\sum_{j=0}^{k} \frac{c_h(j)}{k+1}\right) \\
=&\frac{1}{2}\,\mathbf{E}[N_T] = \frac{\lambda T}{2}
\end{aligned}$$

Let $u(\lambda, b)$ denote the control chosen as a function of λ for the customer arrival process during the given interval and resource balance b, as described in Section 4.5. Since $\mathbf{E}[T] = \frac{1}{u(\lambda,b)}$, the expected cost for the given interval can now be written as $\frac{\lambda}{2u(\lambda,b)}$. To apply Theorem 33, we need to consider the utility as the negative of the holding cost computed above. We therefore have:

$$\bar{g}(\mathbf{u}(\lambda, b)) = -\,\mathbf{E}\left[\frac{\lambda}{2u(\lambda, b)}\right]$$

The consumption of the resource when applying control u ($\leq \mu$) is given by:

$$\bar{c}(\mathbf{u}(\lambda, b)) = \mathbf{E}[\mathbf{u}(\lambda, b)]$$

In order to compute the ϕ function (Equation (4.12)), we need to evaluate:

$$\sup_{\mathbf{u}\in\mathcal{F}_0} (\bar{g}(\mathbf{u}) - \bar{c}(\mathbf{u})x) \tag{4.14}$$

$$= - \inf_{\mathbf{u}\in\mathcal{F}_0} \left(\mathbf{E}\left[\frac{\lambda}{2\mathbf{u}(\lambda, b)} + \mathbf{u}(\lambda, b)x \right] \right) \tag{4.15}$$

$$= - \mathbf{E}\left[\mathbb{1}_{\lambda<2x\mu^2}\sqrt{2\lambda x} + \mathbb{1}_{\lambda\geq 2x\mu^2}\left(\frac{\lambda}{2\mu} + \mu x \right) \right] \tag{4.16}$$

The maximizing argument above is given by $u^*(\lambda, b) = \min(\sqrt{\frac{\lambda}{2x}}, \mu)$ (where x can be interpreted as a function of b). More precisely, once we solve for the value function, $V(B)$, the optimal control as a function of the scaled balance $B = \beta b$ and customer arrival process intensity for the current interval, λ, is given by the above equation with $x = V'(\beta b)$.

To illustrate the remaining steps in the computation of $V(B)$ with simple equations, we will now assume that λ is a deterministic value and then write:

$$\phi(x) = \begin{cases} (a-\mu)x - \frac{\lambda}{2\mu} & \text{if } x \leq \frac{\lambda}{2\mu^2} \\ ax - \sqrt{2\lambda x} & \text{if } x \geq \frac{\lambda}{2\mu^2} \end{cases} \tag{4.17}$$

This gives:

$$x_* = \frac{\lambda}{2(\min(\mu, a))^2}$$

and

$$\eta_* = \phi(x_*) = \begin{cases} -\frac{\lambda}{2a} & \text{if } a \leq \mu \\ \frac{a\lambda}{2\mu^2} - \frac{\lambda}{\mu} & \text{otherwise} \end{cases} \tag{4.18}$$

By Theorem 33, the scaled gain (negative of the cost) value function $V_\beta(B) \triangleq \beta v_\beta(B/\beta)$ converges to a function, $V(B)$, which can be computed as the solution to the ODE $y' = f(y)$ with initial condition, $y(0) = \eta_*$, where f is the inverse function to ϕ, given by $f(y) \triangleq \inf\{x \geq 0 : \phi(x) = y\}$. These curves are now plotted numerically below for different income levels. We assume that $\lambda = 1$ with probability 1 and $\mu = 10$. Figures 4.1, 4.2, 4.3 show the plots for the ϕ function, the value function and the optimal control respectively as a function of the scaled balance for three different values of the arrival/income, a.

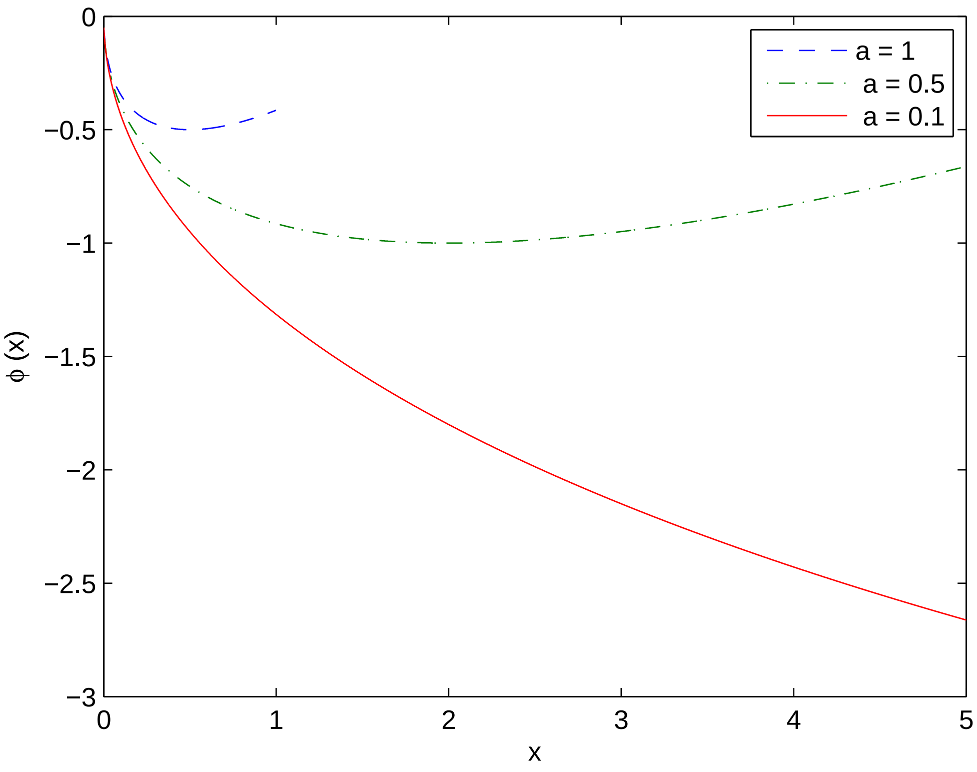


Figure 4.1: A plot of the ϕ function for three different values of a (arrival rate to the resource balance queue.

4.6.4 Proof of Theorem 33

In order to prove the theorem, the convergence of a variant of the problem, parametrized by *exit payoffs*, is first determined. The problem of interest is subsequently shown to converge to the variant with the right exit payoff, which characterizes the limiting value function claimed in the theorem.

4.6.4.1 Solution of Exit-Payoff Variants

For proof purposes, it is useful to introduce a variant of the given MDP, with identical state transitions as Equation (4.10), but without the non-negativity constraint on the actions. Instead, the variant is parametrized with an exit payoff, η, which defines the final utility obtained when the balance reaches a value less than or equal to zero for the first time. More precisely, let $\mathcal{U}'$ represent the collection of all admissible Markov policies.

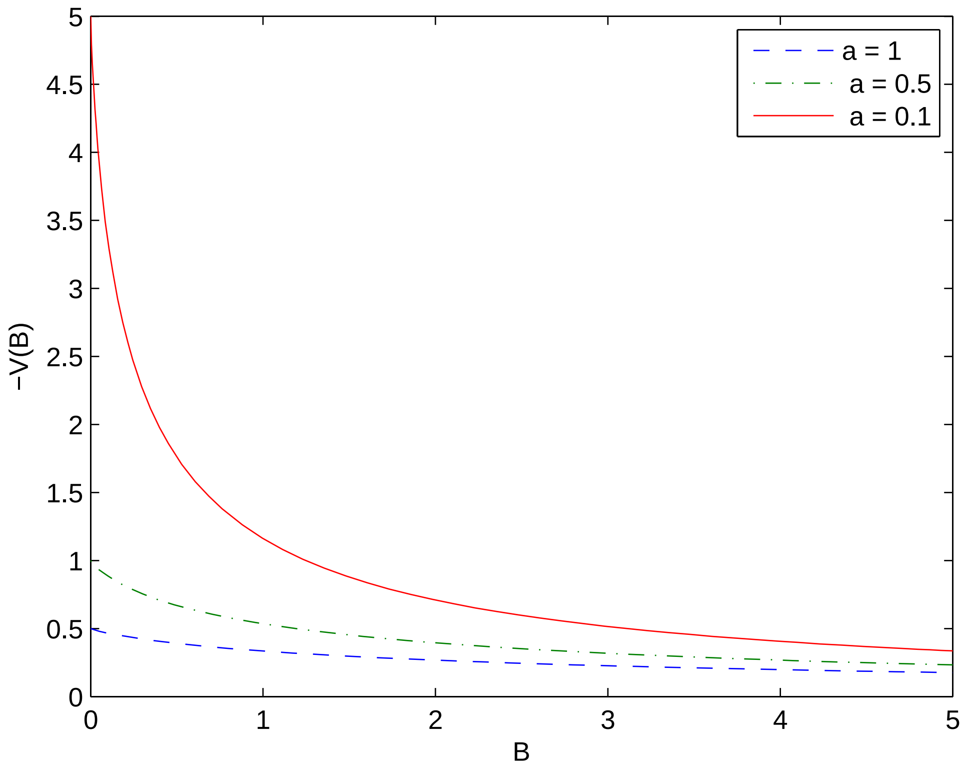


Figure 4.2: A plot of the infinite horizon optimal value function (cost) versus (scaled) balance for three different values of a (arrival rate to the resource balance queue).

Therefore, $\mathcal{U}'$ represents all sequences of actions, defined by random variables $\mathbf{u}(i) \in \mathcal{F}_0$, which can be chosen after observing the corresponding states, $\mathbf{b}(i)$ (but without the additional restriction on actions that could lead to $\mathbf{b}(i+1) < 0$). Let κ denote the following stopping time:

$$\kappa = \inf\{i \geq 0 : \mathbf{b}(i) \leq 0\}$$

The value function is given by:

$$j_\beta(b, \eta) \triangleq \sup_{\mathcal{U}'} \left(\mathbb{E}_b \sum_{i=0}^{\kappa} e^{-\beta i} \bar{g}(\mathbf{u}(i)) + e^{-\kappa} \eta \right)$$

Following prior convention, the scaled version of the value function is given by:

$$J_\beta(B, \eta) = \beta j_\beta(B/\beta, \eta/\beta)$$

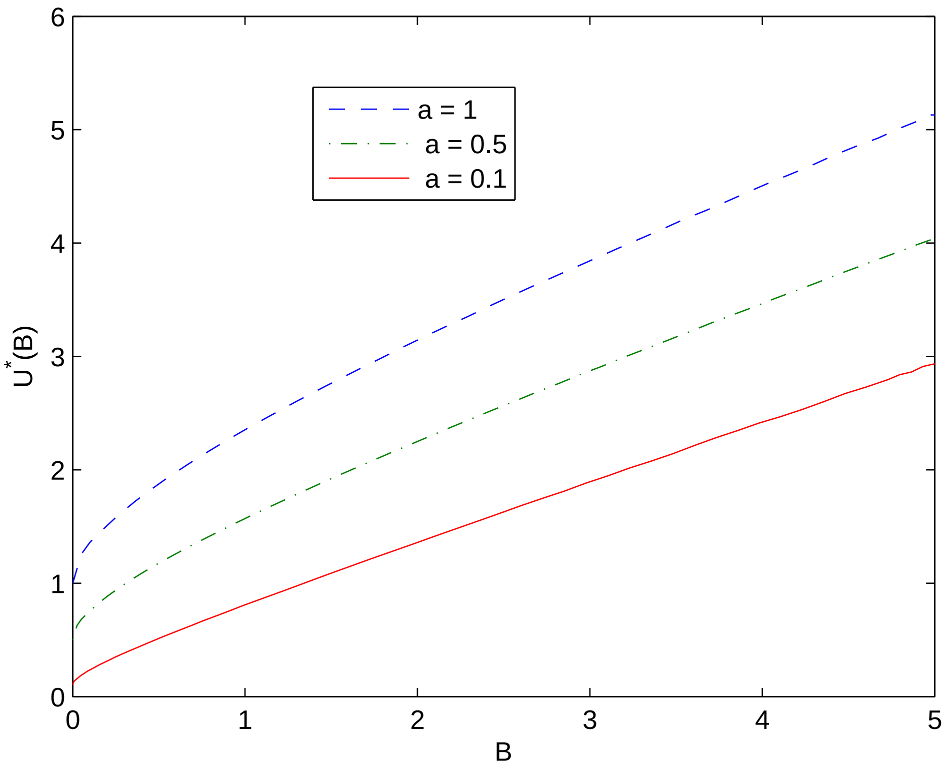


Figure 4.3: A plot of the optimal control as a function of (scaled) balance for three different values of a (arrival rate to the resource balance queue).

To illustrate the convergence of the above problem to its continuous time version, consider the following change of variables/notation: $t_i \triangleq \beta i, \Delta t_i \triangleq t_{i+1} - t_i = \beta$, $B(t_i) \triangleq \beta \mathbf{b}(i), \tau \triangleq \beta\kappa$ and $u(t_i) \triangleq \mathbf{u}(i)\ \forall i$. Using this change of notation:

$$J_\beta(B, \eta) = \sup_{\mathcal{U}'} \left(\mathbb{E}_b \sum_{t_i=0}^{\tau} e^{-t_i} \bar{g}(u(t_i))\Delta t_i + e^{-\tau}\eta \right) \tag{4.19}$$

where $\tau = \inf\{t_i \geq 0 : B(t_i) \leq 0\}$. The state transitions can be expressed as:

$$\Delta B(t_i) \triangleq B(t_{i+1}) - B(t_i) = (a(t_i) - \bar{c}(\mathbf{u}(t_i)))\Delta t$$

From standard results on convergence of discrete to continuous time control problems [95], the scaled value function, J, converges to the value function of a continuous time control problem. To specify this, let

$$\bar{\mathcal{U}} \triangleq \{u(t) \in \mathcal{F}_0 \ \forall \ t \geq 0\}$$

$$V(B,\eta) = \sup_{\bar{\mathcal{U}}} \left(\int_0^\tau e^{-t} \bar{g}(u(t))\, dt + e^{-\tau}\eta \right) \tag{4.20}$$

where $\tau = \inf\{t \geq 0 : B(t) = 0\}$, with state evolution:

$$\frac{dB(t)}{dt} = a - \bar{c}(u(t)), \forall t \geq 0, \quad B(0) = B$$

The above problem is a deterministic continuous time optimal control problem, whose value function has a sufficiency condition determined by its Hamilton-Jacobi-Bellman (HJB) equation [96], which represents an ODE with a boundary condition given by η. The boundary condition is $V(0,\eta) = \eta$, while the ODE is specified by simplifying the following dynamic programming relation for a small Δt:

$$V(B) \geq \sup_{u \in \mathcal{F}_0} \left(\bar{g}(u)\Delta t + e^{-\Delta t} \left(V(B) + \Delta t (a - \bar{c}(u)) V'(B) \right) \right)$$

This gives

$$V(B) = aV'(B) + \sup_{u \in \mathcal{F}_0} \left(\bar{g}(u) - \bar{c}(u)V'(B) \right) = \phi(V'(B))$$

Recall that $f : \mathbb{R}_+ \mapsto \mathbb{R}$, is an inverse to ϕ with $f(y) \triangleq \min\{x \geq 0 : \phi(x) = y\}$. ϕ is convex because it is the supremum over a class of linear functions. From the definition of η_* and the convexity of ϕ, it follows that ϕ is strictly decreasing on $[0, f(\eta_*)]$. From Lemma 40, ϕ is continuous,[5] implying that f is well defined and continuous on $[\eta_*, \eta_0]$ where $\eta_0 \triangleq \phi(0)$.[6] The ODE, $\frac{dV}{dB} = f(V), \quad V(0) = \eta$, therefore has a continuously differentiable solution. From the sufficiency condition of HJB, it is equal to $V(B,\eta)$, the optimal value function to the continuous time problem defined in Equation (4.20).

To summarize, we so far have:

Lemma 34. *Let $\eta \in [\eta_*, \eta_0]$. Then,* $\lim_{\beta \to 0} J_\beta(B,\eta) = V(B,\eta)$, *which satisfies the ODE: $\frac{dV}{dB} = f(V)$, with initial condition, $V(0) = \eta$*

[5]In fact, ϕ is also Lipschitz though this fact is not necessary for the proof.

[6]f is also Lipschitz on $[\eta, \eta_0] \ \ \forall \eta \in (\eta_*, \eta_0]$.

4.6.5 Characterizing the Exit Pay-off

The theorem states that $\lim_{\beta\to 0} V_\beta(B) = V(B,\eta_*)$. Lemma 36 and Lemma 38 together prove that $\lim_{\beta\to 0} V_\beta(0) = \eta_*$. This would imply: $\forall\ B \geq 0$, $\lim_{\beta\to 0} V_\beta(B) = \lim_{\beta\to 0} J_\beta(B,\eta_*) = V(B,\eta_*)$, proving the theorem.

Lemma 35. $\forall \eta, h > 0, B \geq 0$, $J_\beta(B+h,\eta) \geq V_\beta(B)$

Proof. Any optimal policy for $V_\beta(B)$ is also feasible for lower bounding $J_\beta(B+h,\eta)$ in the exit time variant, since the set of feasible policies is less constrained. Since the state transitions are identical in both variants, the stopping time is ∞ in the exit time variant, which implies that $V_\beta(B)$ is an achievable utility in the exit time variant starting from initial balance $B+h$ for any $h > 0$. $\square$

Lemma 36.

$$\limsup_{\beta\to 0} V_\beta(0) \leq \eta_*$$

Proof. Lemma 34, Lemma 35 and the fact that $V(B,\eta)$ is continuous in B for any $\eta \in [\eta_*,\eta_0]$ together imply Lemma 36. Specifically, letting $B = 0, \eta = \eta_*$ in Lemma 35 and letting $\beta \to 0$, we get $\forall h > 0, \quad \lim_{\beta\to 0} J_\beta(h,\eta_*) \geq \limsup_{\beta\to 0} V_\beta(0)$. From Lemma 34, we have $\lim_{\beta\to 0} J_\beta(h,\eta_*) = V(h,\eta_*)$, implying $\forall h > 0, \limsup_{\beta\to 0} V_\beta(0) \leq V(h,\eta_*)$, which implies the claim due to continuity of V. $\square$

An important precursor to Lemma 38 is proved next, for which the following definition is stated first.

Definition 14. *[ϵ-subdifferential] For $\epsilon \geq 0$, $c \in \mathbb{R}$ is called an ϵ-subdifferential to a function f at x_0 if $f(x) + \epsilon \geq f(x_0) + c(x - x_0)\ \forall\ x$. A 'subdifferential' refers to the 0-subdifferential.*

Lemma 37. *For any $\epsilon > 0$, $\exists\ \mathbf{u}^* \in \mathcal{F}_0$ such that $\bar{g}(\mathbf{u}^*) > \eta_* - \epsilon$ and $\bar{c}(\mathbf{u}^*) \leq a$.*

Proof. Let $x^* = \sup\{x \geq 0 : \phi(x) = \eta_*\}$ and $x_* = \inf\{x \geq 0 : \phi(x) = \eta_*\}$ (both of which are finite if $a > 0$, and when $a = 0$, we have $\eta_* = 0$, in which case the theorem is trivially true by choosing $\mathbf{u}^* = 0$). We first consider the case where $x_* = x^*$. For any $h > 0$, any subdifferential of ϕ at x^*+h is strictly positive because $\phi(x^*+h) > \phi(x^*)$. Let $\theta(h) > 0$ be small enough that any θ-subdifferential is also strictly positive for $\theta < \theta(h)$ (see Proposition 39). Let $\theta = \min(\epsilon/2, \theta(h))$. Let $\psi(\mathbf{u}, x) \triangleq \bar{g}(\mathbf{u}) + (a - \bar{c}(\mathbf{u}))x$. Choose $\mathbf{u}^*_+ \in \mathcal{F}_0$ such that $\psi(\mathbf{u}^*_+, x^*+h) > \phi(x^*+h) - \theta(h)$. Then, for any x, $\phi(x) \geq \psi(\mathbf{u}^*_+, x) = \psi(\mathbf{u}^*_+, x^*+h) + (a - \bar{c}(\mathbf{u}^*_+))(x - x^* - h) > \phi(x^*+h) + (a - \bar{c}(\mathbf{u}^*_+))(x - x^* - h) - \theta(h)$, implying that $a - \bar{c}(\mathbf{u}^*_+)$ is a $\theta(h)$-subdifferential to ϕ at x^*+h. Therefore, $\bar{c}(\mathbf{u}^*_+) \leq a$. Also, since $\eta_* < \phi(x^*+h) < \theta(h) + \psi(\mathbf{u}^*_+, x^*+h)$, we have:

$$\eta_* < \theta + \bar{g}(\mathbf{u}^*_+) + (a - \bar{c}(\mathbf{u}^*_+))(x^*+h) \tag{4.21}$$

Analogously, we can pick $\mathbf{u}^*_- \in \mathcal{F}_0$ such that $\bar{c}(\mathbf{u}^*_-) \geq a$ and

$$\eta_* < \theta + \bar{g}(\mathbf{u}^*_-) + (a - \bar{c}(\mathbf{u}^*_-))(x^*-h) \tag{4.22}$$

Now we can choose $\mathbf{u}^* = \delta \mathbf{u}^*_- + (1-\delta)\mathbf{u}^*_+$ where[7]

$$\delta = \begin{cases} 1 & \text{with probability } \alpha = \frac{a - \bar{c}(\mathbf{u}^*_+)}{\bar{c}(\mathbf{u}^*_-) - \bar{c}(\mathbf{u}^*_+)} \\ 0 & \text{with probability } 1-\alpha = \frac{\bar{c}(\mathbf{u}^*_-) - a}{\bar{c}(\mathbf{u}^*_-) - \bar{c}(\mathbf{u}^*_+)} \end{cases}$$

so that: $\bar{c}(\mathbf{u}^*) = a$. Using this, inequality (4.21) $\times (1-\alpha) +$ (4.22) $\times \alpha$ gives $\eta_* < \bar{g}(\mathbf{u}^*) + \theta + 2(a+C)h$. Now choose any $h < \frac{\epsilon}{4(a+C)}$ and since $\theta < \epsilon/2$, we have: $\eta_* < \bar{g}(\mathbf{u}^*) + \epsilon$.

It remains to argue for the case $x_* < x^*$. For given $h > 0$, we may pick $\mathbf{u}^*_+$ exactly as before because the subdifferential at x^*+h is again strictly positive. However, in this case, the only subdifferential for any $x_0 \in (x_*, x^*)$ is zero. Therefore, for any $h > 0$, $\exists\, \theta(h) > 0$ such that any $\theta-$subdifferential at x^*-h has to lie in $(-h, h)$. As before we can pick $\mathbf{u}^*_- \in \mathcal{F}_0$ such that Equation (4.22) holds and $a - \bar{c}(\mathbf{u}^*_-)$ is a subdifferential at x^*-h, implying that $a - \bar{c}(\mathbf{u}^*_-) \in (-h, h)$ (where we chose $\theta = \min(\epsilon/2, \theta(h))$). If $a \leq \bar{c}(\mathbf{u}^*_-)$, we can go through the same construction as before to obtain $\mathbf{u}^*$ for which $\bar{c}(\mathbf{u}^*) = a$ and $\eta_* < \bar{g}(\mathbf{u}^*) + \epsilon$. Now suppose $a > \bar{c}(\mathbf{u}^*_-)$. From Equation (4.22), we get $\eta_* < \theta + \bar{g}(\mathbf{u}^*_-) + h(x^*-h) > \bar{g}(\mathbf{u}^*_-) + \epsilon$ if we pick $h < \frac{\epsilon}{2x^*}$. Therefore, in this case, for $\mathbf{u}^* = \mathbf{u}^*_-$, we have $\bar{c}(\mathbf{u}^*) < a$ and $\bar{g}(\mathbf{u}^*) > \eta_* - \epsilon$, as required.

□

[7] if $\bar{c}(\mathbf{u}^*_+) = \bar{c}(\mathbf{u}^*_-) = a$, then we can pick any $\alpha \in [0,1]$.

Lemma 38.

$$\liminf_{\beta \to 0} V_\beta(0) \geq \eta_*$$

Proof. It is sufficient to show that there exists a feasible policy that can achieve a scaled infinite horizon discounted utility with a lower bound that is arbitrarily close to η_*. Let $u^* \in \mathcal{F}_0$ with $\bar{g}(u^*) = \eta_*$ and $a = \bar{c}(u^*)$. Consider a policy in which the decision maker chooses the random variable u^* in every time slot, i.e.:

$$U^* \triangleq \{u^*, u^*, \ldots\}$$

Therefore, the state transitions are defined by $b(i+1) = b(i) + w(i)$, where $w(1), w(2), \ldots$ are i.i.d. copies of their generic version, $w \triangleq a - c(u^*, \xi)$, which is bounded by $|w| \leq 1$ without loss of generality, and has zero drift: $\mathbb{E}[w] = 0$. Therefore, $b(i)$ represents a random walk with zero drift, which has a positive probability of falling below zero. This random walk has a strictly positive probability of going below zero. Therefore, $U^* \notin \mathcal{U}$. Consider a modified walk, for which, whenever $b(i) \leq 1$, we set $u'(j) = 0 \;\; \forall j = i, i+1, \ldots i + \frac{M}{a}$, where M is a constant that will be fixed. For all other j, $u'(j) = u^*$. Clearly, this is feasible for the original problem. Let x denote the utility achieved by this policy starting from $b(0) = M$. Let $\tau_M \triangleq \min\{i \geq 0 : \sum_{j=0}^{i} w(i) \leq -M + 1\}$. We have:

$$x = \mathbb{E}\left[\sum_{i=0}^{\tau} g(u^*, \xi) + e^{-\beta(\tau + \frac{M}{a})} x\right]$$

This can be simplified to give:

$$x = \left(\frac{\eta_*}{1 - e^{-\beta}}\right)\left(\frac{1 - \mathbb{E}[e^{-\beta(\tau_M + 1)}]}{1 - \mathbb{E}[e^{-\beta(\tau_M + M)}]}\right)$$

which implies:

$$V_\beta(0) \geq \beta e^{-\beta \frac{M}{a}} x = \eta_* e^{-\beta \frac{M}{a}} \left(\frac{\beta}{1 - e^{-\beta}}\right)\left(\frac{1 - \mathbb{E}[e^{-\beta(\tau_M + 1)}]}{1 - \mathbb{E}[e^{-\beta(\tau_M + M)}]}\right)$$

For any fixed M, as $\beta \to 0$, the lower bound above becomes arbitrarily close to $\eta_* \frac{\mathbb{E}[\tau_M]+1}{\mathbb{E}[\tau_M]+M}$. For large M, the expected hitting time for the asymmetric random walk with maximum step size equal to one is at most (stochastically bounded by) the expected hitting time of a symmetric random walk with step size equal to one, which scales as $\mathbb{E}[\tau_M] = \Omega(M^2)$. Therefore, the lower bound is arbitrarily close to η_*, proving the lemma. $\square$

Proposition 39. *Suppose all subdifferentials of a convex function, f, at x belong to $[d_l, d_h]$. Then, for any $\delta > 0$, $\exists\ \epsilon > 0$ small enough, such that any ϵ-subdifferential of f at x belongs to $(d_l - \delta, d_h + \delta)$.*

Proof. Draw two lines, L_1 and L_2 with slopes $d_l - \delta, d_h + \delta$ at x. Clearly, there exist $x_l < x$ and $x_h > x$ such that L_1 strictly dominates f in (x_l, x) and L_2 strictly dominates f in (x, x_h). Choose:

$$\epsilon^* = \min\left(\sup\{L_1(x) - f(x) : x \in (x_l, x)\}, \sup\{L_2(x) - f(x) : x \in (x, x_h)\}\right)$$

Then $\epsilon^* > 0$ and for any $\epsilon < \epsilon^*$, neither $d_l - \delta$ nor $d_h + \delta$ can be ϵ-subdifferentials, and therefore any ϵ-subdifferential of f has to lie in $(d_l - \delta, d_h + \delta)$. $\square$

Lemma 40. *ϕ is Lipschitz.*

Proof. Let $\psi(\mathbf{u}, x) \triangleq \bar{g}(\mathbf{u}) + (a - \bar{c}(\mathbf{u}))x$, so that $\phi(x) = \sup_{\mathbf{u} \in \mathcal{F}_0} \psi(\mathbf{u}, x)$. Let $|x - y| < \delta$. Let $\mathbf{u}^*(x), \mathbf{u}^*(y) \in \mathcal{F}_0$ such that $\psi(\mathbf{u}^*(x), x) \in (\phi(x) - \delta, \phi(x)]$ and $\psi(\mathbf{u}^*(y), y) \in (\phi(y) - \delta, \phi(y)]$ and assume $\psi(\mathbf{u}^*(x), x) \geq \psi(\mathbf{u}^*(y), y)$ without loss of generality. Then, $|\phi(x) - \phi(y)| \leq \delta + \psi(\mathbf{u}^*(x), x) - \psi(\mathbf{u}^*(y), y)$. Note that $\psi(\mathbf{u}^*(y), y) > \phi(y) - \delta \geq \bar{g}(\mathbf{u}^*(x)) + (a - \bar{c}(\mathbf{u}^*(x)))\, y - \delta \geq \psi(\mathbf{u}^*(x), x) - (1 + |a - C|)\delta$, where C is the bound on the consumption function. Therefore, $|\phi(x) - \phi(y)| < M\delta$, where $M = 2 + |a - C|$ is the Lipschitz constant. $\square$

4.7 Conclusion

In this chapter, we considered a queuing system for which each service clears the entire queue, which we called the broadcast server model. We considered control problems to minimize the discounted infinite horizon costs of holding customers in the system together with two types of constraints on the server. In the first type, each service is charged a fixed non-negative service cost. In the second type of constraint, we have an online running constraint on the total number of broadcasts. We formulated a general class of problems that have the second type of constraint in Section 4.6 called the generalized version of an online knapsack problem, and we derived a limiting result on the value function and the optimal control when the discount factor is effective over a large duration. This was then applied to the broadcast server problem with an online constraint to derive the approximately optimal control and value functions in the limiting case.

REFERENCES

[1] T. Cover and J. Thomas, *Elements of Information Theory.* John Wiley and Sons, 2006.

[2] D. Mackay, *Information Theory, Learning and Algorithms.* Cambridge University Press, 2003.

[3] R. Blahut, *Algebraic Codes for Data Transmission.* Cambridge University Press, 2003.

[4] J. Byers, M. Luby, M. Mitzenmacher, and A. Rege, "A digital fountain approach to reliable distribution of bulk data," in *Proc. ACM Special Interest Group Conference on Communications (SIGCOMM)*, Sep. 1998, pp. 56–67.

[5] J. Byers, M. Luby, and M. Mitzenmacher, "A digital fountain approach to asynchronous reliable multicast," *IEEE Journal on Selected Areas in Communication*, vol. 20, pp. 1528–1540, October 2002.

[6] M. Luby, "LT codes," in *Proc. IEEE Foundations of Computer Science (FOCS)*, 2002, pp. 271–280.

[7] A. Shokrollahi, "Raptor codes," *IEEE Transactions on Information Theory*, vol. 52, pp. 2551–2567, 2006.

[8] P. Maymounkov, "Online codes," New York University, Tech. Rep. TR2002-833, Nov. 2002.

[9] M. Luby, A. Shokrollahi, M. Watson, and T. Stockhammer, "Raptorq forward error correction scheme for object delivery," May 2011. [Online]. Available: http://tools.ietf.org/html/draft-ietf-rmt-bb-fec-raptorq-06

[10] "Raptor technology: Forward error correction," 2011. [Online]. Available: http://www.qualcomm.com/products_services/mobile_content_services/data_transfer_streaming/technology.html

[11] E. Arikan, "Some complexity results about packet radio networks," *IEEE Transactions on Information Theory*, vol. 30, pp. 681–685, 1984.

[12] P. Gupta and P. R. Kumar, "The capacity of wireless networks," *IEEE Transactions on Information Theory*, vol. 46, pp. 388–404, March 2000.

[13] R. Gummadi, K. Jung, D. Shah, and R. Sreenivas, "Feasible rate allocation in wireless networks," in *Proc. IEEE International Conference on Computer Communications*, 2008, pp. 995–1003.

[14] R. Gummadi, K. Jung, D. Shah, and R. Sreenivas, "Computing the capacity region of a wireless network," in *Proc. IEEE International Conference on Computer Communications*, 2009, pp. 1341 – 1349.

[15] R. Yeung, *Information Theory and Network Coding.* Springer, 2008.

[16] R. Ahlswede, N. Cai, S. Li, and R. Yeung, "Network information flow," *IEEE Transactions on Information Theory*, vol. 46, pp. 1204–1216, July 2000.

[17] R. Gummadi and R. Sreenivas, "Efficient repair for erasure codes," Under Review, 2011.

[18] R. Gummadi, A. Shokrollahi, and R. Sreenivas, "Broadcasting with side information," in *Proc. IEEE Information Theory Workshop (ITW)*, 2010, pp. 1–5.

[19] R. Gummadi and R. Sreenivas, "Relaying a fountain code across multiple nodes," in *Proc. IEEE Information Theory Workshop (ITW)*, May 2008, pp. 149–153.

[20] R. Gummadi, L. Massoulie, and R. Sreenivas, "The role of coding in local broadcast for wireless unicast," in *Proc. IEEE International Symposium on Network Coding (NETCOD '10)*, June 2010, pp. 1–6.

[21] R. Gummadi, "Optimal control of a broadcasting server," in *Proc. IEEE International Conference on Decision and Control (CDC)*, 2010, pp. 2634 – 2639.

[22] R. Gummadi, A. Proutiere, and P. Key, "Optimal bidding strategies and equilibria in repeated auctions with budget cosntraints," Working Paper, 2011.

[23] P. Elias, "Coding for two noisy channels," in *Information Theory, 3rd London Symposium*, 1955, pp. 61–76.

[24] F. Macwilliams and N. Sloane, *The Theory of Error-Correcting Codes.* North Holland, 1988.

[25] N. Alon and M. Luby, "A linear time erasure-resilient code with nearly optimal recovery," *IEEE Transactions on Information Theory*, vol. 42, pp. 1732–1736, Nov. 1996.

[26] M. Luby, M. Mitzenmacher, A. Shokrollahi, and D. Spielman, "Efficient erasure correcting codes," *IEEE Transactions on Information Theory*, vol. 47, p. 569, Feb. 2001.

[27] R. Gallager, *Low Density Parity-Check Codes.* MIT Press, 1963.

[28] D. J. C. MacKay and R. M. Neal, "Near Shannon limit performance of low density parity check codes," *Electronic Letters*, vol. 32, pp. 1645–1646, 1996.

[29] M. Luby, M. Mitzenmacher, A. Shokrollahi, and D. Spielman, "Analysis of low density codes and improved designs using irregular graphs," in *Proc. Annual ACM Symposium on Theory of Computing*, 1998, pp. 249–258.

[30] M. Luby, M. Mitzenmacher, A. Shokrollahi, and D. Spielman, "Improved low-density parity-check codes using irregular graphs and belief propagation," in *Proc. IEEE International Symposium on Information Theory*, 1998, p. 117.

[31] M. Luby, M. Mitzenmacher, A. Shokrollahi, and D. Spielman, "Analysis of low density codes and improved designs using irregular graphs," *IEEE Transactions on Information Theory*, vol. 47, pp. 671–686, Feb. 2000.

[32] T. Richardson and R. Urbanke, "Efficient encoding of LDPC codes," *IEEE Transactions on Information Theory*, vol. 47, pp. 638–656, Feb. 2001.

[33] A. Shokrollahi, "LDPC codes: An introduction," Digital Fountain, Tech. Rep., April 2003.

[34] M. Luby, M. Mitzenmacher, A. Shokrollahi, D. Spielman, and V. Stemann, "Practical loss-resilient codes," in *Proc. ACM symposium on Theory of computing (STOC)*, 1997, pp. 150–159.

[35] O. Etesami and A. Shokrollahi, "Raptor codes on binary memoryless symmetric channels," *IEEE Transactions on Information Theory*, vol. 52, pp. 2551–2567, June 2006.

[36] R. Gummadi and A. Shokrollahi, "Improving the error floor of raptor codes on binary symmetric channels," Unpublished, 2010.

[37] R. Karp, M. Luby, and A. Shokrollahi, "Finite length analysis of LT codes," in *Proc. IEEE International Symposium on Information Theory (ISIT)*, 2004, p. 39.

[38] B. Hajek, "Connections between network coding and stochastic network theory," 2006. [Online]. Available: http://www.ifp.illinois.edu/~hajek/Papers/HajekStochNets06.pdf

[39] R. Darling and J. Norris, "Structure of large random hypergraphs," *Annals of Applied Probability*, vol. 15, pp. 125–152, 2005.

[40] E. Maneva and A. Shokrollahi, "New model for rigorous analysis of LT codes," in *Proc. IEEE International Symposium on Information Theory (ISIT)*, 2006, pp. 2677–2679.

[41] S. Yekhanin, "Locally decodable codes," *Foundations and Trends in Theoretical Computer Science*, to appear.

[42] A. Dimakis, K. Ramachandran, Y. Wu, and C. Suh, "A survey on network codes for distributed storage," *Proceedings of the IEEE*, vol. 99, pp. 476–489, March 2011.

[43] A. Dimakis, "The coding for distributed storage wiki," 2011. [Online]. Available: http://tinyurl.com/storagecoding

[44] A. Jiang, "Network coding for joint storage and transmission with minimum cost," in *Proc. IEEE International Symposium on Information Theory*, 2006, pp. 1359–1363.

[45] A. G. Dimakis, P. B. Godfrey, Y. Wu, M. J. Wainwright, and K. Ramachandran, "Network coding for distributed storage systems," *IEEE Transactions on Information Theory*, vol. 56, pp. 4539–4551, Sep. 2010.

[46] Y. Wu, A. G. Dimakis, and K. Ramachandran, "Deterministic regenerating codes for distributed storage," in *Proc. Allerton Conference on Control Computing and Communications*, 2007.

[47] Y. Wu, "Existence and construction of capacity-achieving network codes for distributed storage," *IEEE Journal on Selected Areas in Communications*, vol. 28, pp. 277–288, Feb. 2010.

[48] Y. Wu, "Existence and construction of capacity-achieving network codes for distributed storage," in *Proc. IEEE International Symposium on Information Theory*, June 2009.

[49] Y. Wu and A. G. Dimakis, "Reducing repair traffic for erasure coding-based storage via interference alignment," in *Proc. IEEE International Symposium on Information Theory*, June 2009, pp. 2276–2280.

[50] D. Cullina, A. G. Dimakis, and T. Ho, "Searching for minimum storage regenerating codes," in *Proc. Allerton Conference on Control Computing and Communication*, Sep. 2009.

[51] K. V. Rashmi, N. B. Shah, P. V. Kumar, and K. Ramchandran, "Exact regenerating codes for distributed storage," in *Proc. Allerton Conference on Control Computing and Communication*, Sep. 2009.

[52] N. B. Shah, K. V. Rashmi, P. V. Kumar, and K. Ramchandran, "Explicit codes minimizing repair bandwidth for distributed storage," in *Proc. IEEE Information Theory Workshop*, Jan. 2010.

[53] C. Suh and K. Ramchandran, "Exact regeneration codes for distributed storage repair using interference alignment," in *Proc. IEEE International Symposium on Information Theory*, June 2010.

[54] Y. Wu, "A construction of systematic MDS codes with minimum repair bandwidth," *IEEE Transactions on Information Theory*, vol. 57, pp. 3738–3741, June 2011.

[55] A. Duminuco and E. Biersack, "A practical study of regenerating codes for peer-to-peer backup systems," in *Proc. International Conference on Distributed Computing Systems*, 2009, pp. 376–384.

[56] D. Papailiopoulos and A. G. Dimakis, "Connecting distributed storage codes and secure degrees-of-freedom of multiple access channels," in *Proc. Allerton Conference and Control Computing and Communications*, 2010.

[57] J. Li, S. Yang, X. Wang, and B. Li, "Tree-structured data regeneration in distributed storage systems with regenerating codes," in *Proc. IEEE International Conference on Computer Communications (INFOCOM)*, 2010.

[58] V. Cadambe, S. Jafar, and H. Maleki, "Distributed data storage with minimum storage regenerating codes – exact and functional repair are asymptotically equally efficient," in *Proc. IEEE International Workshop on Wireless Network Coding*, Apr. 2010.

[59] K. Rashmi, N. B. Shah, and P. V. Kumar, "Optimal exact-regenerating codes for distributed storage at the msr and mbr points via a product-matrix construction," 2010. [Online]. Available: http://arxiv.org/abs/1005.4178

[60] D. Sejdinovic, R. Piechocki, and A. Doufexi, "Fountain coding with decoder side information," in *Proc. IEEE International Conference on Communications (ICC)*, 2008, pp. 4477–4482.

[61] A. Hagedorn, S. Agarwal, D. Starobinski, and A. Trachtenberg, "Rateless coding with feedback," in *Proc. IEEE International Conference on Computer Communications (INFOCOM)*, April 2009.

[62] Z. Zhang, A. Deshpande, X. Ma, E. Thereska, and D. Narayanan, "Does erasure coding have a role to play in my data center?" Microsoft Research, Tech. Rep. MSR-TR-2010-52, May 2010.

[63] J. Plank, "Erasure codes for storage applications," in *Proc. 4th Usenix Conference on File Storage Technologies*, 2005.

[64] J. Metzner, "An improved broadcast retransmission protocol," *IEEE Transactions on Communications*, vol. 32, pp. 679–683, June 1984.

[65] Y. Birk and T. Kol, "Coding-on-demand by an informed source (iscod) for efficient broadcast of different supplemental data to caching clients," *IEEE Transactions on Information Theory*, vol. 52, pp. 2825–2830, June 2006.

[66] Z. Bar-Yossef, Y. Birk, T. Jayram, and T. Kol, "Index coding with side information," in *Proc. IEEE Conference on Foundations of Computer Science (FOCS)*, 2006.

[67] S. Sanghavi, "Intermediate performance of rateless codes," in *Proc. IEEE Information Theory Workshop (ITW)*, September 2007.

[68] C.-P. Chen and F. Qi, "The best bounds of the n-th harmonic number," *Global Journal of Mathematics and Mathematical Sciences*, vol. 1, pp. 41–49, March 2006.

[69] P. Pakzad, C. Fragouli, and A. Shokrollahi, "Coding schemes for line networks," in *Proc. IEEE International Symposium on Information Theory (ISIT)*, 2005.

[70] N. Harvey, D. Lun, and P. Maymounkov, "Methods for efficient network coding," in *Proc. Allerton Conference on Communications, Control and Computing*, 2006.

[71] S. Chachulski, M. Jennings, S. Katti, and D. Katabi, "Trading structure for randomness in wireless opportunistic routing," in *Proc. ACM Special Interest Group Coference on Commmunications (SIGCOMM)*, 2007.

[72] B. Smith and B. Hassibi, "Wireless erasure networks with feedback," 2008. [Online]. Available: http://arxiv.org/pdf/0804.4298v1

[73] D. Lun, M. Medard, R. Koetter, and M. Effros, "On coding for reliable communication over packet networks," *Physical Communication*, vol. 1, pp. 3–20, March 2008.

[74] A. Dana, R. Gowaikar, R. Palanki, and M. Effros, "Capacity of wireless erasure networks," *IEEE Transactions on Information Theory*, vol. 52, pp. 789–804, March 2006.

[75] S. Biswas and R. Morris, "Opportunistic routing in multi-hop wireless networks," in *Proc. ACM Special Interest Group Coference on Commmunications (SIGCOMM)*, 2005.

[76] M. Neely and R. Urgaonkar, "Optimal backpressure routing for wireless networks with multi-receiver diversity," *AD HOC NETWORKS (ELSEVIER)*, vol. 7, pp. 862–881, July 2009.

[77] J. Sundararajan, D. Shah, and M. Medard, "Feedback-based online network coding," April 2009. [Online]. Available: http://arxiv.org/abs/0904.1730

[78] L. Tassiulas and A. Ephremides, "Stability properties of constrained queueing systems and scheduling policies for maximum throughput in multihop radio networks," *IEEE Transactions on Automatic Control*, vol. 37, pp. 1936–1948, December 1992.

[79] R. Deb and R. Serfozo, "Optimal control of batch service queues," *Advances in Applied Probability*, vol. 5, pp. 340–361, 1973.

[80] S. Aalto, "Optimal control of batch service queues with compound poisson arrivals and finite service capacity," *Mathematical Methods of Operations Research*, vol. 48, pp. 317–335, 1998.

[81] S. Aalto, "Optimal control of batch service queues with finite service capacity and linear holding costs," *Mathematical Methods of Operations Research*, vol. 51, pp. 263–285, 2000.

[82] R. Deb, "Optimal control of bulk queues with compound poisson arrivals and batch service," *Opsearch*, vol. 21, pp. 227–245, 1984.

[83] H. Weiss, "Optimal control of batch service queues with non linear waiting costs," *Modeling and Simulation*, vol. 10, pp. 305–309, 1979.

[84] H. Weiss and S. Pliska, "Optimal policies for batch service queuing systems," *Opsearch*, vol. 19, pp. 12–22, 1981.

[85] B. Hajek, "Optimal control of two interacting service stations," *IEEE Transactions on Automatic Control*, vol. 29, pp. 491–499, June 1984.

[86] N. Vaidya and S. Hameed, "Scheduling data broadcasts in asymmetric communication environments," *Wireless Networks*, vol. 5-3, pp. 171–182, May 1999.

[87] M. Ammar and J. Wong, "The design of teletext broadcast cycles," *Performance Evaluation*, vol. 5, pp. 235–242, 1985.

[88] N. Bansal, M. Charikar, S. Khanna, and J. Naor, "Approximating the average response time in broadcast scheduling," in *Proc. Annual ACM-SIAM symposium on Discrete algorithms (SODA)*, 2005, pp. 215–221.

[89] A.Bar-Noy, R. Bhatia, J. Naor, and B. Schieber, “Minimizing service and operation costs of periodic scheduling,” *Mathematics of Operations Research*, vol. 27, pp. 518–544, 2002.

[90] J. Chang, T. Erlebach, R. Gailis, and S. Khuller, “Improved approximation algorithms for broadcast scheduling,” in *Proc. Annual ACM Symposium on Discrete Algorithms (SODA)*, 2008.

[91] C. Chekuri and B. Moseley, “Online scheduling to minimize the maximum delay factor,” in *Proc. Annual ACM Symposium on Discrete Algorithms (SODA)*, 2009.

[92] C. Xia, G. Michailidis, N. Bambos, and P. Glynn, “Optimal control of parallel queues with batch service,” *Probability in the Engineering and Informational Sciences*, vol. 16, pp. 289–307, 2002.

[93] C. Su, L. Tassiulas, and V. Tsotras, “Broadcast scheduling for inormation distribution,” *Wireless Networks*, vol. 5, pp. 137–147, March 1999.

[94] Z. Rosberg, P. Varaiya, and J. Walrand, “Optimal control of service in tandem queues,” *IEEE Transactions on Automatic Control*, vol. 27, June 1982.

[95] H. Kushner, “Numerical methods for stochastic control problems in continuous time,” *SIAM Journal on Control and Optimization*, vol. 28, pp. 999–1048, Sep. 1990.

[96] D. Bertsekas, *Dynamic Programming and Optimal Control.* Athena Scientific, 2007.