Iterative Decoding for Wireless Networks

Thesis by

Ravi Palanki

In Partial Fulfillment of the Requirements

for the Degree of

Doctor of Philosophy

California Institute of Technology

Pasadena, California

2004

(Submitted May 21, 2004)

To

my parents

and

Ramya

Acknowledgements

It takes a lot of good karma to have Prof. Robert McEliece as an adviser. His insightful thinking and his unbounded enthusiasm have always helped change his graduate students' lives for the better, and I am no exception to this rule. I will always be indebted to him for guiding me through four of the most memorable years of my life.

My heartfelt thanks also go to Dr. Jonathan Yedidia for offering me internships at Mitsubishi Electric Research Laboratories and for being my mentor ever since. Working with him and Dr. Marc Fossorier of the University of Hawaii was a truly rewarding experience.

I would like to thank Profs. Michelle Effros, Babak Hassibi, P. P. Vaidyanathan and Dr. Sam Dolinar of JPL for agreeing to be on my candidacy or defense committees and for being excellent teachers. Much of my knowledge of communications and information theory comes from them.

I am grateful to Caltech's administrative staff, especially Shirley Beatty, for their help in making my stay here extremely pleasant. I would also like to acknowledge generous funding from the National Science Foundation, Qualcomm Corp., Sony Corp., the Lee Center for Advanced Networking and Mitsubishi Electric Research Laboratories.

Finally, I would like to thank all my friends here at Caltech and elsewhere. Each one of them, in his or her own unique way, has left on me a lasting impression that can not be described in words.

Abstract

The invention of turbo codes and low density parity check (LDPC) codes has made it possible for us for design error correcting codes with low decoding complexity and rates close to channel capacity. However, such codes have been studied in detail only for the most basic communication system, in which a single transmitter sends data to a single receiver over a channel whose statistics are known to both the transmitter and the receiver. Such a simplistic model is not valid in the case of a wireless network, where multiple transmitters might want to communicate with multiple receivers at the same time over a channel which can vary rapidly.

While the design of efficient error correction codes for a general wireless network is an extremely hard problem, it should be possible to design such codes for several important special cases. This thesis takes a few steps in that direction. We analyze the performance of low density parity check codes under iterative decoding in certain simple networks and prove Shannon-theoretic results for more complex networks.

More specifically, we analyze the iterative decoding algorithm in two very important special cases: (a) when the transmitter and receiver have no prior knowledge of the channel and (b) when the channel is a multiple access channel. We also apply iterative decoding to some non-LDPC codes on the binary symmetric channel and the additive white Gaussian noise channel. Finally, we derive capacity results for a class of wireless multicast networks and a class of fading channels.

Contents

List of Figures

List of Tables

Chapter 1 Introduction

Digital communications technology has had an unrivaled impact on society over the course of the last few decades. With applications ranging from sophisticated military satellites and NASA's Mars Rovers to the ubiquitous Internet and cell phones used every day by billions of people, digital communications systems have altered almost every aspect of our lives. Yet, none of these systems would exist today were it not for Claude Shannon and his seminal paper, "The mathematical theory of communication" [71].

In this paper, originally published in 1948, Shannon gave the first quantitative definition of information, thereby creating the field of information theory. Shannon also proved that reliable communication of information over inherently unreliable channels is feasible. This means that even though a transmitter sends a message over a channel that corrupts or destroys part of the transmitted signal, the receiver can figure out precisely what message was sent. This counter-intuitive result, known as the "channel coding theorem," also tells us that such reliable communication is possible if and only if the information transmission rate R (measured in bits per channel use) is less than a threshold known as the "channel capacity" C (also measured in bits per channel use).

Shannon proposed the use of "channel codes" to construct such a reliable communication system. Let us suppose that a transmitter needs to send k bits of information to a receiver. If the transmitter sends these k bits directly over a noisy channel, some (or all) of these k bits would be corrupted by the time they reach the receiver. So the transmitter "encodes" the k bits into $n > k$ bits using a "channel code," thereby introducing $n - k$ redundant bits. It then transmits the n bits over the noisy chan-

nel. Even though these n bits are corrupted by the time they reach the receiver, the receiver can use the redundant bits to try to figure out (or "decode") the original k bits.

Shannon showed that as $k \to \infty$, with a judicious choice of the channel code, the receiver can almost surely infer what the transmitter sent. Moreover, the rate of the code $R \stackrel{\triangle}{=} k/n$ can be made arbitrarily close to the channel capacity C. Unfortunately, Shannon's choice for the channel code is impossible to implement in practical systems, because the computational complexity of the decoder is unacceptably high. This led to one of the most important problems in information theory viz., to find practical channel codes whose rates are close to the channel capacity.

This problem proved to be extremely hard to solve. Even though information theorists constructed a wide variety of practical channel codes, none of these codes had rates close to channel capacity. It was only in 1993 that Berrou, Glavieux and Thitimajshima [5] developed "turbo codes," the first practical codes that had rates close to capacity. This breakthrough revolutionized the field of channel coding and led to the development of low density parity check (LDPC) codes [22, 45, 66], which are state-of-the-art codes that have near-capacity performance on many important practical channels.

The rest of this chapter is devoted to brief descriptions of the channel coding theorem and LDPC codes. These descriptions are vital to understanding this thesis, for all results presented in this thesis are either channel capacity computations or analyses of the performance of LDPC codes on various channels. In Section 1.1, we define channel capacity and channel codes, and give examples of important channel models. In Section 1.2, we review low density parity check codes and the iterative algorithm used to decode them. In Section 1.3, we describe the motivation for this thesis and give a brief outline of the succeeding chapters.

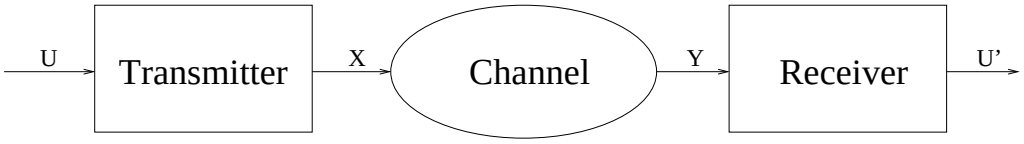


Figure 1.1: A canonical communication system.

1.1 Basics of channel coding

1.1.1 Channel capacity

A canonical model of a single transmitter, single receiver communication system, originally studied by Shannon, is shown in Figure 1.1. The objective is to send a sequence of symbols $\mathbf{U} \stackrel{\triangle}{=} (U_1, U_2, ..., U_k)$ across the noisy channel. To do this, the transmitter maps (or *encodes*) $\mathbf{U}$ to another sequence of bits $\mathbf{X} \stackrel{\triangle}{=} (X_1, X_2, ..., X_n)$, which it then transmits over the channel. The receiver sees a string of corrupted output symbols $\mathbf{Y} \stackrel{\triangle}{=} (Y_1, Y_2, ..., Y_n)$ where $\mathbf{Y}$ depends on $\mathbf{X}$ via a probability density function (pdf) $p_{\mathbf{Y}|\mathbf{X}}(\mathbf{y}|\mathbf{x})$. The receiver then estimates (or *decodes*) $\mathbf{U}$ based on $\mathbf{Y}$.

Definition 1.1 A channel is called *memoryless* if the channel output at any time instant depends only on the input at that time instant. Mathematically, this means that $p_{\mathbf{Y}|\mathbf{X}}(\mathbf{y}|\mathbf{x}) = \prod_{i=1}^{n} p_{Y|X}(y_i|x_i)$. In this case, the channel is completely described by its input and output alphabets, and the conditional pdf $p_{Y|X}(y|x)$ for one time instant.

If each X_i is chosen independently and identically distributed (i.i.d.) according to a pdf $p_X(x)$, then the Y_i's are also i.i.d., with the pdf $p_Y(y)$ given by

$$p_Y(y) = \int_X p_X(x) p_{Y|X}(y|x) dx \tag{1.1}$$

Definition 1.2 The *mutual information* between the random variables X and Y, denoted by $I(X;Y)$ is defined as

$$I(X;Y) = \int_{X,Y} p(x)p(y|x) \log_2 \left(\frac{p(y|x)}{p(y)} \right) dx \; dy \tag{1.2}$$

The mutual information between X and Y is a quantitative measure of what the knowledge of Y can tell us about X (and vice versa).

Definition 1.3 The *capacity* of a memoryless channel specified by $p_{Y|X}(y|x)$ is

$$C = \sup_{p_X(x)} I(X;Y) \tag{1.3}$$

Intuitively, C is the maximum of amount of information that can be learnt about X from Y and hence is a measure of the maximum rate R at which information can be reliably transmitted across the channel. Shannon rigorously showed this was the case [10, 71], i.e., error free transmission was possible at rates $R < C$ and impossible at rates $R > C$, where R and C are both measured in bits per channel use.

1.1.2 Channel models

In this thesis, we focus on *binary input symmetric channels* (BISCs) viz., channels with the input X chosen from a binary input alphabet and the output symmetric in the input. We interchangeably use the sets $\{0, 1\}$ and $\{+1, -1\}$ for the input alphabet with 0 mapping to $+1$ and 1 to -1. The symmetry condition implies that $p_{Y|X}(y|+1) = p_{Y|X}(-y|-1)$. We now present three of the most important BISCs, which arise in many communication systems.

Example 1.1 (The binary erasure channel (BEC)) This channel, with parameter p, has input alphabet $\{+1, -1\}$ and output alphabet $\{+1, 0, -1\}$. The output symbol 0 is also called an erasure. The output is equal to the input with probability

$1-p$ and is 0 with probability p. p is called the probability of erasure. This channel is arguably the simplest nontrivial channel model, and has capacity $1-p$.

Example 1.2 (The binary symmetric channel (BSC)) This is a binary-input, binary-output channel with parameter p. To view it as a BISC, it is convenient to let the input and output alphabets be $\{+1,-1\}$. Then the output is equal to the input with probability $1-p$, and is the negative of the input with probability p. p is called the crossover probability of the channel. The capacity of this channel is given by $1-H(p)$, where $H(p)$ is the entropy function $-p\log_2 p-(1-p)\log_2(1-p)$.

Example 1.3 (The binary input additive white Gaussian noise channel (BIAWGNC)) The binary input additive white Gaussian noise channel has inputs X restricted to inputs +1 and -1. The output Y is a random variable given by $Y=X+N$, where N is a Gaussian random variable with mean zero and variance σ^2. The capacity of the AWGNC is given by

$$C=1-\frac{1}{\sqrt{2\pi\sigma^2}}\int_{\mathbb{R}} H\left(\frac{1}{1+e^{2x/\sigma^2}}\right)e^{\frac{(x-1)^2}{2\sigma^2}}dx \tag{1.4}$$

where $H(\cdot)$ is again the entropy function. It is customary to express the capacity of a BIAWGNC in terms of its *signal-to-noise ratio* (SNR) defined as

$$\begin{aligned}\frac{E_s}{N_0} &= 10\log_{10}\left(\frac{1}{2\sigma^2}\right)\ \text{dB}\\ \frac{E_b}{N_0} &= 10\log_{10}\left(\frac{1}{2C\sigma^2}\right)\ \text{dB}\end{aligned} \tag{1.5}$$

Here E_s/N_0 denotes the SNR per transmitted bit and E_b/N_0 denotes the SNR per information bit. SNR is typically measured in *deciBels* (dB).

1.1.3 Channel codes

A channel code is defined as a mapping from a set of *messages* $\mathcal{M} = \{1, 2, ..., M\}$ to a set of *codewords*, which are vectors of length n over some alphabet $\mathcal{A}$. n is called the *blocklength* of the code and $\log_2 M$ its *dimension*.

The most common channel codes are *binary* codes in which the alphabet $\mathcal{A} = \{0, 1\}$ and $M = 2^k$ for some k. In such a case, we refer to the code as a (n, k) code. The *rate* R of the code is defined as k/n. A binary code can also be thought of as a mapping from $\{0, 1\}^k$ to $\{0, 1\}^n$. In other words, a channel code maps a k-bit message to an n-bit codeword. Most practical channel codes have $n > k$, which means that $n - k$ *redundant* bits are added to the message. It is this redundancy that helps in error correction.

Often, we need to impose additional structure on the codes to make them easier to design, analyze and implement. Most practical codes in use today are *linear* codes.

Definition 1.4 An (n, k) *linear code* over the binary field $GF(2)$ is a k-dimensional vector subspace of $GF(2)^n$.

Linear codes have several nice properties, for example, they look exactly the same around any codeword. That is, if $\mathcal{C}$ is a linear code and $\mathbf{c} \in \mathcal{C}$ is a codeword, then the set $\mathcal{C} - \mathbf{c}$ is identical to $\mathcal{C}$. Also, in order to describe a linear code, we don't have to list all its elements, but merely a basis. Such a description is called a *generator matrix* representation.

Definition 1.5 A *generator matrix* for an (n, k) linear code $\mathcal{C}$ is a $k \times n$ matrix G whose rows form a basis for $\mathcal{C}$. As $\mathbf{u}$ varies over the space $GF(2)^k$, $\mathbf{u}G$ varies over the set of codewords. Thus, the matrix G provides a simple encoding mechanism for the code.

Another useful representation of a linear code is a *parity-check matrix* representation.

Definition 1.6 A *parity-check matrix* (PCM) for an (n, k) linear code $\mathcal{C}$ is an $(n - k) \times n$ matrix H whose rows form a basis for the space of vectors orthogonal to $\mathcal{C}$. That is, H is a full rank matrix s.t. $H\mathbf{c} = \mathbf{0} \iff \mathbf{c} \in \mathcal{C}$.

The parity check matrix representation can be represented graphically using a bipartite graph called a *Tanner graph*. Each row i of the PCM corresponds to a *check node*, and each column j to a *variable node*. There is an edge between check node i and variable node j if and only if $H_{ij} = 1$.

Definition 1.7 The *Hamming distance* between two vectors is the number of components in which they differ. The *minimum distance* of a code is the smallest Hamming distance between two distinct codewords. For a linear code, this is the same as the least weight of any nonzero codeword.

1.1.4 Capacity achieving codes

Even for moderately large k, there are a very large number of channel codes. Finding *capacity achieving codes*, informally defined as codes with low probability of error and rates close to channel capacity, from this large set of codes seems like a computationally impossible task. However, Shannon showed that the *ensemble of random codes* can achieve capacity as $k \to \infty$ [10]. A random code is one in which each message is mapped to a codeword picked randomly according to a uniform distribution on $\{0, 1\}^n$. It can also be shown that the *ensemble of random linear codes* can achieve capacity on any BISC. In a random linear code, each of the n codeword bits is generated by taking a random linear combination of the k data bits. In other words, each element of the generator matrix G (or the parity check matrix H) is chosen i.i.d. according to the distribution $\Pr(0) = \Pr(1) = 1/2$.

The fact that a code constructed randomly can achieve capacity might lead one to believe that the channel coding problem is easy to solve. Unfortunately, random

codes and random linear codes can achieve capacity under *maximum-likelihood (ML) decoding*, an algorithm whose complexity grows exponentially in k. Given the received sequence $\mathbf{Y}$, the ML-decoder searches the entire codeword space (consisting of 2^k words) and outputs

$$\hat{\mathbf{x}} = \arg\max_{\mathbf{x}\in\mathcal{C}} p_{\mathbf{Y}|\mathbf{X}}(\mathbf{y}|\mathbf{x}). \tag{1.6}$$

Clearly, the ML-decoder is optimal in the sense that it minimizes the *word error rate* (WER), which is the probability $\hat{\mathbf{x}}$ is not the same as the transmitted codeword. ML-decoders are hard to implement for most codes since ML-decoding typically involves computing $p_{\mathbf{Y}|\mathbf{X}}(\mathbf{y}|\mathbf{x})$ for all 2^k codewords. It must be noted here that there are classes of codes which can be ML-decoded in polynomial time. An example would be convolutional codes [17], whose ML-decoding algorithm is the Viterbi algorithm [81] whose complexity is linear in the length of the code. However, practical convolutional codes are not capacity achieving.

1.2 Low density parity check codes

The high complexity associated with ML-decoding of random linear codes creates the need for capacity-achieving codes with efficient decoding algorithms. Such codes could not be found despite the invention of a large number of code families [53, 82] with efficient decoding algorithms. The situation changed in 1993 with invention of turbo codes [5], which led to the design of capacity-achieving *low density parity check* (LDPC) codes. We will now describe the structure of these codes and their decoding algorithms.

As their name suggests, low density parity check codes, originally invented by Gallager [22], have *sparse* parity check matrices. Informally, this means that the number of ones in any row or column of the parity check matrix is small. An LDPC code has $O(1)$ ones per row in contrast to a typical random linear code, which has

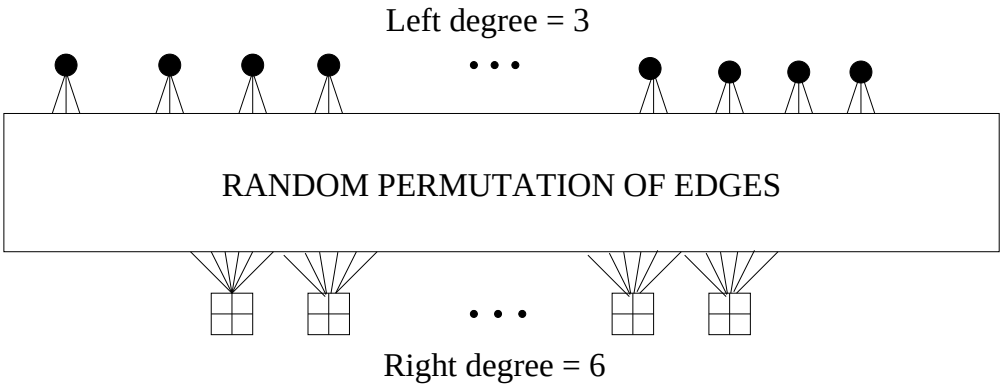


Figure 1.2: Tanner graph of $(3, 6)$-regular LDPC code.

around $n/2$ or $O(n)$ ones per row. This means that each variable node in the Tanner graph of an LDPC code is connected to $O(1)$ check nodes, and each check node is connected to $O(1)$ variable nodes.

The simplest kind of LDPC codes are *regular* LDPC codes. In a (d_v, d_c)-regular LDPC code, each variable node is connected to exactly d_v check nodes and each check node is connected to exactly d_c variable nodes. The connections between the variable nodes and check nodes are generally chosen at random. For large n, the rate of such a code is $1 - d_v/d_c$. Figure 1.2 shows the Tanner graph of a $(3, 6)$-regular LDPC code.

1.2.1 Degree distributions

A more general class of LDPC codes are *irregular* LDPC codes. In an irregular LDPC code, different nodes may have different connectivities. For example, half the variable nodes may be connected to two check nodes each, a quarter to three check nodes each, and the remaining quarter to eight check nodes. The *variable node degree distribution* of such a code is specified by the polynomial $\nu(x) = 0.5x^2 + 0.25x^3 + 0.25x^8$. The *check node degree distribution* $\mu(x)$ is similarly defined. For a regular LDPC code,

$\nu(x) = x^{d_v}$ and $\mu(x) = x^{d_c}$.

We can also define the *edge degree distributions* $\lambda(x)$ and $\rho(x)$.

$$\begin{aligned} \lambda(x) &= \frac{\nu^{'}(x)}{\nu^{'}(1)} \\ \rho(x) &= \frac{\mu^{'}(x)}{\mu^{'}(1)} \end{aligned} \tag{1.7}$$

For the (3,6)-LDPC code, $\lambda(x) = x^2$ and $\rho(x) = x^5$. $\lambda(x) = x^2$ means that an edge has two neighboring edges at the variable node side, i.e., two edges connected to the same variable node. Similarly, $\rho(x) = x^5$ means that five neighboring edges at the check node side. For an irregular LDPC code, the polynomials $\lambda(x)$ and $\rho(x)$ specify probability distributions on edge connectivities. Edge distribution polynomials are more useful in analyzing code performance than node distribution polynomials. This is because of the nature of the algorithm used to decode LDPC codes.

1.2.2 Iterative decoding

The iterative decoding algorithm used to decode LDPC codes, called the *sum-product algorithm*, is a completely distributed algorithm with each node acting as an independent entity communicating with other nodes through the edges. The message sent by a variable node to a check node is its estimate of its own value. Typically the messages are sent in *log-likelihood ratio* (LLR) form. The LLR of any bit is defined as $\log(\Pr(\text{bit} = 0)/\Pr(\text{bit} = 1))$. The message sent by a check node to a variable node is the check node's estimate of the variable node's value.

At a variable node of degree j, if l_1, l_2, $\ldots$, l_{j-1} denote the incoming LLRs along $j-1$ edges, and l_0 the LLR corresponding to the channel evidence, then the outgoing LLR l_{out} along the jth edge is merely the *maximum a posteriori* (MAP) estimate of the underlying binary random variable given j independent estimates of it, and is

given by

$$l_{\text{out}} = l_0 + \sum_{i=1}^{j-1} l_i. \tag{1.8}$$

At a check node, the situation is similar, though the update rule is more complicated. If l_1, l_2, ..., l_{k-1} denote the incoming LLR's at a check node of degree k, then the outgoing LLR l_k along the kth edge corresponds to the pdf of the binary sum of $j-1$ independent random variables, and works out to be

$$\tanh(l_{\text{out}}/2) = \prod_{i=1}^{k-1} \tanh(l_i/2). \tag{1.9}$$

(For a derivation of eqs. (1.8) and (1.9), see [66, Section 3.2].)

Given these update rules, we only need a schedule for updating the various messages to complete the description of the decoding algorithm, but this schedule varies from code to code, and sometimes there are many reasonable schedules even for a single code. There is one canonical schedule, however, which is to update all variable nodes together, followed by all check nodes, followed again by the variable nodes etc. In practice, for this algorithm to work well, a Tanner graph should have few short cycles. This is true in the case of LDPC codes, but is not true for general linear codes.

We do not describe the theory behind the sum-product algorithm, for it is beyond the scope of this thesis. However, we must mention that it has been studied extensively in the literature. For example, McEliece et al. showed that the sum-product algorithm was an instance [3, 54] of a more general algorithm known as *belief propagation* [63], which is widely used in the artificial intelligence community. Luby et al. analyzed the performance of LDPC codes at infinite blocklengths on the BEC [45]. This analysis, known as *density evolution*, was later extended to other channels by Richardson et al. [66]. More recently, Yedidia et al. explored the connections between belief propagation and free energy approximations in statistical physics [85].

1.2.3 Capacity achieving distributions

Since a message is passed along each edge in every iteration of the sum-product algorithm, the complexity of decoding (per iteration) is proportional to the number of edges in the Tanner graph. Since the graph is sparse, the number of edges and thus the decoding complexity (per iteration) grow linearly in n. This allows for the decoding of codes with very high blocklengths. Therefore, all that remains to be done is the design of LDPC codes which can achieve capacity under iterative decoding.

It can be shown using density evolution analysis [45, 66] that such design is indeed feasible. The proof consists of three main steps. Firstly, it can be shown that the performance of an ensemble of LDPC codes depends only on its degree distribution in the limit of infinite blocklength. This is not hard to understand because the connections between the variable nodes and check nodes are chosen at random. The second step is show that an ensemble has a *threshold* channel parameter. For example, codes designed using the $(3, 6)$-regular distribution have a threshold of $p = 0.429$ on the BEC. This means that the average *bit error rate* of the code ensemble approaches zero as $n \to \infty$ when the codes are used on a BEC with probability of erasure less than 0.429. For comparison, a capacity-achieving random linear code under ML-decoding has a threshold of $p = 0.5$. This means that the $(3, 6)$-code ensemble is not capacity achieving.

The third and hardest part of the proof is demonstrating the existence of degree distributions with thresholds approaching channel capacity. Luby et al. proved that this is the case for any BEC [45], i.e., at any given rate R, there exists a sequence of irregular degree distributions with thresholds approaching $1 - R$. However, such a result exists only for the BEC, probably because density evolution analysis is much easier for the BEC than it is for general channels. However, it is widely believed that capacity achieving distributions can be designed for other channels as well. For example, Chung et al. designed rate-1/2 degree distributions whose thresholds are

within 0.0045 dB of channel capacity on the BIAWGNC [8]. While these codes are not strictly capacity-achieving, their thresholds are close enough to channel capacity for all practical purposes.

While density evolution allows us to design asymptotically good degree distributions, it does not guarantee that these codes work well at finite block lengths. However, extensive simulation studies have shown that LDPC codes have very good performance in the intermediate blocklength ($n \approx 1000$) to long blocklength ($n \approx 100000$) range [9]. They perform rather poorly in the short block length ($n < 1000$) range. However, since iterative decoding is extremely fast, it is possible to use long codes in practice. For example, Flarion Technologies designed an LDPC code with $n = 8192$ and rate 1/2 that has a WER of 10^{-10} at under 1.5 dB from capacity on the BIAWGNC. The hardware implementation of their decoder can support data rates up to 10 Gbps [18].

1.3 Thesis outline

The results stated in Section 1.2.3 might lead us to the conclusion that the problem of transmitting data reliably and efficiently over noisy channels has been solved. Such a conclusion would not be untrue for the communication system shown in Figure 1.1 where a single transmitter transmits data to a single receiver over a channel whose statistics are known a priori to both the transmitter and the receiver. However, not all communications systems can be characterized by such a simple model.

Figure 1.3 shows an example of a communication *network* which is more general than the system shown in Figure 1.1. In a network, there are multiple nodes, each of which can be a transmitter or receiver or both. Each node might want to transmit information to one or more receivers, and each node might want to receive information from one or more transmitters. The channel between any two nodes may vary with

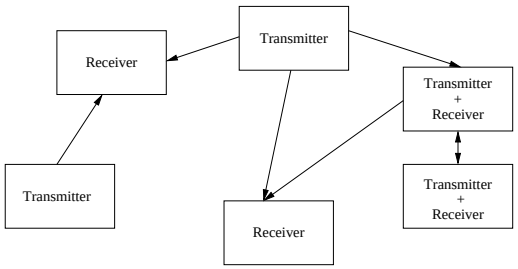


Figure 1.3: A general communication system.

time, and the channel statistics may not be known to the nodes. Moreover, if the network is *wireless*, transmissions from multiple senders can *interfere* at a receiver.

Since many contemporary communication systems are in fact networks, we would like to design efficient error correction codes for general networks. Unfortunately, this is very hard for two reasons. Firstly, the capacity region of a general network is not known [10, Chapter 14]. Secondly, even in the special cases where the capacity region is known, it is not clear that efficient capacity achieving codes exist. In this thesis, we take a few steps towards the design of efficient network codes. Our approach is twofold: in cases where the capacity region of the network is known, we study the performance of LDPC-like codes under iterative decoding. In cases where the capacity region is unknown, we try to compute the capacity region.

The rest of the thesis is organized as follows. In Chapter 2, we study the performance of *rateless codes* on the BSC and the BIAWGNC. These codes are designed to be used on time varying channels where the channel statistics are not known a priori to the transmitter and the receiver. An ideal rateless code should work well on any channel, no matter how good or bad the channel is. We show that this is the case for a class of codes called *raptor codes*.

In Chapter 3, we design codes for *multiple access channels* (MACs), where multiple senders wish to send information to a single receiver. The transmissions from the

senders interfere with each other at the receiver. We describe how the sum-product algorithm can be adapted to handle such interference, and how LDPC codes can be designed for a MAC.

In Section 1.2.3, we mentioned that LDPC codes perform poorly at short block-lengths. In Chapter 4, we design iterative decoding algorithms for a class of non-sparse codes called *Euclidean geometry* (EG) codes. We show that some short EG codes exhibit excellent performance under these algorithms. More generally, we show that it is possible to decode non-LDPC codes using iterative decoding.

In Chapter 5, we study a class of wireless relay networks for which the capacity region was previously unknown. In these networks, a single source wants to transmit information to a single receiver. All the other nodes act as *relays*, which aid in the communication between the transmitter and the receiver. We derive the capacity of such a network under certain conditions.

In Chapter 6, we study *fading channels*, which are very common in wireless networks. We study the capacity-achieving distributions of certain *block fading* channels, and show that the capacity achieving distributions of all block fading channels have some common theoretical properties.

Finally, we summarize our results and list some open problems in Chapter 7.

Chapter 2 Rateless codes on noisy channels

In this chapter, we consider iterative decoding for channels when the transmitter and receiver have no prior knowledge of the channel statistics. We give a brief description of *rateless codes* and go on to study the performance of two classes of rateless codes (LT and raptor codes) on noisy channels such as the BSC and the AWGNC. We find that raptor codes outperform LT codes, and have good performance on a wide variety of channels.

2.1 Introduction

Recent advances in coding theory, especially the invention of regular [22] and irregular [45] low density parity check (LDPC) codes, have shown that very efficient error correction schemes are possible. LDPC codes, decoded using the belief propagation algorithm, can achieve capacity on the binary erasure channel (BEC) [45, 57] and achieve rates very close to capacity on other channels such as the binary symmetric channel (BSC) and the additive white Gaussian noise channel (AWGNC)[8]. Because of this, one could say that the problem of reliable communication over many practical channels has been solved. However, such a statement comes with a caveat: both the transmitter and the receiver must know the exact channel statistics a priori. While this assumption is valid in many important cases, it is clearly not true in many other equally important cases. For example, on the Internet (which is modeled as a BEC), the probability p that a given packet is dropped varies with time, depending on traffic conditions in the network. A code designed for a good channel (low p) would result in

decoding failure when used over a bad channel (high p). Conversely, a code designed for a bad channel would result in unnecessary packet transmissions when used over a good channel.

This problem can be solved using *rateless codes*. Instead of encoding the k information bits to a pre-determined number of bits using a block code, the transmitter encodes them into a potentially infinite stream of bits and then starts transmitting them. Once the receiver gets a sufficient number of symbols from the output stream, it decodes the original k bits. The number of symbols required for successful decoding depends on the quality of the channel. If decoding fails, the receiver can pick up a few more output symbols and attempt decoding again. This process can be repeated until successful decoding. The receiver can then tell the transmitter over a feedback channel to stop any further transmission.

The use of such an *incremental redundancy* scheme is not new to coding theorists. In 1974, Mandelbaum [50] proposed puncturing a low rate block code to build such a system. First the information bits are encoded using a low rate block code. The resulting codeword is then punctured suitably and transmitted over the channel. At the receiver the punctured bits are treated as erasures. If the receiver fails to decode using just the received bits, then some of the punctured bits are transmitted. This process is repeated till every bit of the low rate codeword has been transmitted. If the decoder still fails, the transmitter begins to retransmit bits till successful decoding. It is easy to see such a system is indeed a rateless code, since the encoder ends up transmitting a different number of bits depending on the quality of the channel. Moreover, if the block code is a random (or random linear) block code, then the rateless code approaches the Shannon limit on every binary input symmetric channel (BISC) as the rate of the block code approaches zero. Thus such a scheme is optimal in the information theoretic sense.

Unfortunately, it does not work as well with practical codes. Mandelbaum origi-

nally used Reed-Solomon codes for this purpose and other authors have investigated the use of punctured low rate convolutional [27] and turbo [39] codes. In addition to many code-dependent problems, all these schemes share a few common problems. Firstly, the performance of the rateless code is highly sensitive to the performance of the low rate block code, i.e., a slightly sub-optimal block code can result in a highly sub-optimal rateless code. Secondly, the rateless code has very high decoding complexity, even on a good channel. This is because on any channel, the decoder is decoding the same low rate code, but with varying channel information. The complexity of such a decoding scheme grows at least as $O(k/R)$ where R is the rate of the low rate code.

In a recent landmark paper, Luby [43] circumvented these problems by designing rateless codes which are not obtained by puncturing standard block codes. These codes, known as *Luby Transform* (LT) codes, are low density generator matrix codes which are decoded using the same message passing decoding algorithm (belief propagation) that is used to decode LDPC codes. Also, just like LDPC codes, LT codes achieve capacity on every BEC. Unfortunately, LT codes also share the *error floor* problem endemic to capacity achieving LDPC codes. Shokrollahi [73] showed that this problem can be solved using raptor codes, which are LT codes combined with outer LDPC codes. These codes have no noticeable error floors on the BEC. However, their rate is slightly bounded away from capacity.

The aim of this chapter is to study the performance of LT and raptor codes on channels other than the BEC. Since LDPC codes designed for the BEC perform fairly well on other channels, one might conjecture that such a result holds for LT codes as well. We test this conjecture for LT codes using simulation studies and density evolution [66].

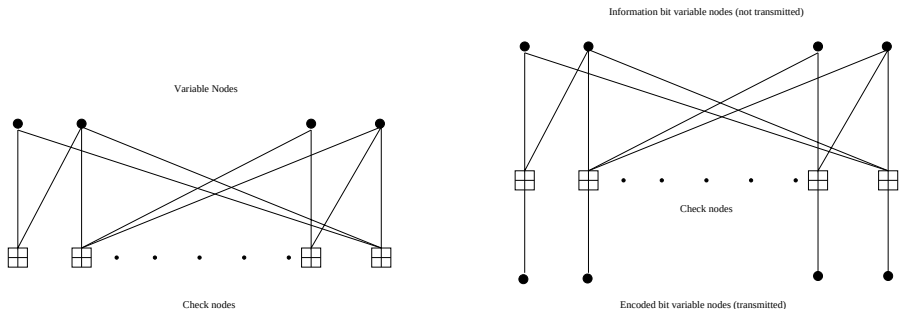
2.2 Luby transform codes

The operation of an LT encoder is very easy to describe. From k given information bits, it generates an infinite stream of encoded bits, with each such encoded bit generated as follows:

1. Pick a degree d at random according to a distribution $\mu(d)$.
2. Choose uniformly at random d distinct input bits.
3. The encoded bit's value is the XOR-sum of these d bit values.

The encoded bit is then transmitted over a noisy channel, and the decoder receives a corrupted version of this bit. Here we make the non-trivial assumption that the encoder and decoder are completely synchronized and share a common random number generator, i.e., the decoder knows which d bits are used to generate any given encoded bit, but not their values. On the Internet, this sort of synchronization is easily achieved because every packet has an uncorrupted packet number. More complicated schemes are required on other channels; here we shall just assume some such scheme exists and works perfectly in the system we're studying. In other words, the decoder can reconstruct the LT code's Tanner graph without error.

Having done that, the decoder runs a belief propagation algorithm on this Tanner graph. The message passing rules are straightforward and resemble those of an LDPC decoder. However there is one important difference: the Tanner graph of an LDPC code contains only one kind of variable node (Figure 2.1(a)), while that of an LT code contains two kinds of variable nodes (Figure 2.1(b)) These are the information bit variable nodes which are not transmitted (and hence have no channel evidence) and the encoded bit variable nodes which are transmitted over the channel.



(a) (b)

Figure 2.1: Tanner graph of (a) LDPC code (b) LT code.

2.2.1 The robust soliton distribution

Clearly, for large block lengths, the performance of such a system depends mostly on the degree distribution μ. Luby uses the Robust Soliton (RS) distribution, which in turn is based on the ideal soliton distribution, defined as follows:

$$\begin{aligned} \rho(1) &= 1/k \\ \rho(i) &= 1/i(i-1)\ \forall i \in \{2,3,...,k\} \end{aligned} \tag{2.1}$$

While the ideal soliton distribution is optimal in some ways (cf. [43]), it performs rather poorly in practice. However, it can modified slightly to yield the robust soliton distribution $RS(k,c,\delta)$. Let $R \stackrel{\triangle}{=} c \cdot \ln(k/\delta)\sqrt{k}$ for some suitable constant $c > 0$. Define

$$\tau(i) = \begin{cases} R/ik & \text{for } i = 1,\ldots,k/R-1 \\ R\ln(R/\delta)/k & \text{for } i = k/R \\ 0 & \text{for } i = k/R+1,\ldots,k \end{cases} \tag{2.2}$$

Now add $\tau(.)$ to the ideal soliton distribution $\rho(.)$ and normalize to obtain the robust soliton distribution:

$$\mu(i) = (\rho(i) + \tau(i))/\beta \tag{2.3}$$

where β is the normalization constant chosen to ensure that μ is a probability distri-

bution.

Luby's analysis and simulation studies show that this distribution performs very well on the erasure channel. The only disadvantage is the decoding complexity grows as $O(k \ln k)$, but it turns out that such a growth in complexity is in fact necessary to achieve capacity [73]. However, slightly sub-optimal codes called raptor codes, can be designed with decoding complexity $O(k)$ [73]. On the BEC, theoretical analysis of the performance of LT codes and raptor codes is feasible, and both codes have been shown to have excellent performance. In fact, raptor codes are currently being used by Digital Fountain, a Silicon Valley based company, to provide fast and reliable transfer of large files over the Internet.

On other channels such as the BSC and the AWGNC, there have been no studies in the literature on the use of LT and raptor codes, despite the existence of many potential applications, e.g., transfer of large files over a wireless link, multicast over a wireless channel. We hope to fill this void by presenting some simulation results and some theoretical analysis (density evolution). In this chapter, we focus on the BSC and the AWGNC, but we believe that our results can be extended to time varying and fading channels.

2.3 LT codes on noisy channels

When the receiver tries decoding after picking up a finite number n of symbols from the infinite stream sent out by the transmitter, it is in effect trying to decode an (n, k) code, with a non-zero rate $R = k/n$. As R decreases, the decoding complexity goes up and the probability of decoding error goes down. In this chapter, we have studied the variation of bit error rate (BER) and word error rate (WER) with the rate of the code on a given channel.

In Figure 2.2, we show some results for LT codes on a BSC with 11% bit flip

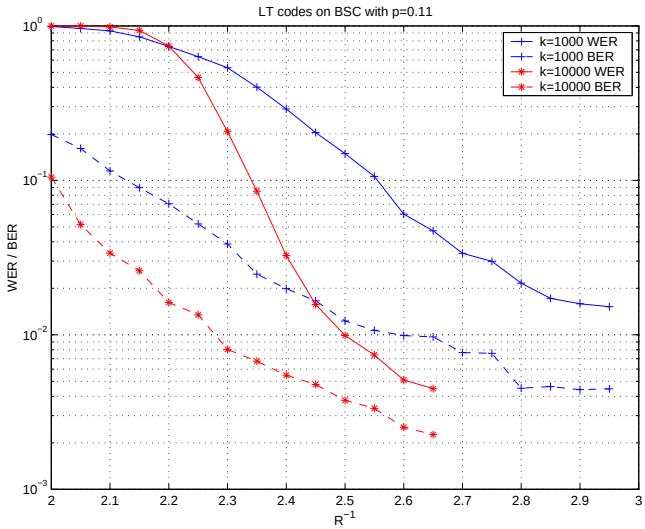


Figure 2.2: The performance of LT codes generated using the $RS(k, 0.01, 0.5)$ distribution on a BSC with $p = 0.11$.

probability. We mention that the results are similar in nature on other BSCs and other AWGNCs as well. In this figure, we plot R^{-1} on the x-axis and BER/WER on the y-axis. The receiver buffers up kR^{-1} bits before it starts decoding the LT code using belief propagation. On a BSC with 11% bit flip probability, the Shannon limit is $R^{-1} = 2$, i.e., a little over $2k$ bits should suffice for reliable decoding in the large k limit. We see from the figure that an LT code with $k = 10000$ drawn using the RS(10000,0.1,0.5) distribution can achieve a WER of 10^{-2} at $R^{-1} = 2.5$ (or $n = 25000$). While this may suffice for certain applications, neither a 25% overhead nor a WER of 10^{-2} is particularly impressive. Moreover, the WER and BER curves bottom out into an error floor, and achieving very small WERs without huge overheads in nearly impossible. Going to higher block lengths is also not practical because of the $O(k \ln k)$ complexity.

2.3.1 Error floors

The error floor problem is not confined to LT codes generated using a robust soliton distribution. Codes generated using distributions optimized by Shokrollahi for the

BEC[1][73] also exhibit similar behaviour. The main advantage of these distributions is that the average number of edges per node remains constant with increasing k, which means the decoding complexity grows only as $O(k)$. On the minus side, there will be a small fraction of information bit nodes that are not connected to any check node. This means that even as k goes to infinity, the bit error rate does not go to zero and consequently, the word error rate is always one.

In this chapter, we discuss the performance of one particular distribution from [73]:

$$\begin{aligned}\mu(x) &= 0.007969x + 0.493570x^2 + 0.166220x^3 \\ &\quad +0.072646x^4 + 0.082558x^5 + 0.056058x^8 + 0.037229x^9 \\ &\quad +0.055590x^{19} + 0.025023x^{65} + 0.0003135x^{66} \end{aligned} \tag{2.4}$$

Shown in Figure 2.3 is the performance of codes generated using the distribution in equation (2.4) at lengths 1000, 10000 and infinity. The performance at length infinity is computed using density evolution [66]. Again, we observe fairly bad error floors, even in the infinite blocklength limit.

We must mention that these error floors are not just due to the presence of information bit variable nodes not connected to any check nodes. For example, when $R^{-1} = 3.00$, only a very small fraction of variable nodes (2.25×10^{-8}) are unconnected, while density evolution predicts a much larger bit error rate (1.75×10^{-4}). This can be attributed in part to the fact that there are variable nodes which are connected to a relatively small number of output nodes and hence are always unreliable.

[1]Note that these distributions were not designed to be used in LT codes, but in raptor codes. See Section 2.4 for a description of raptor codes.

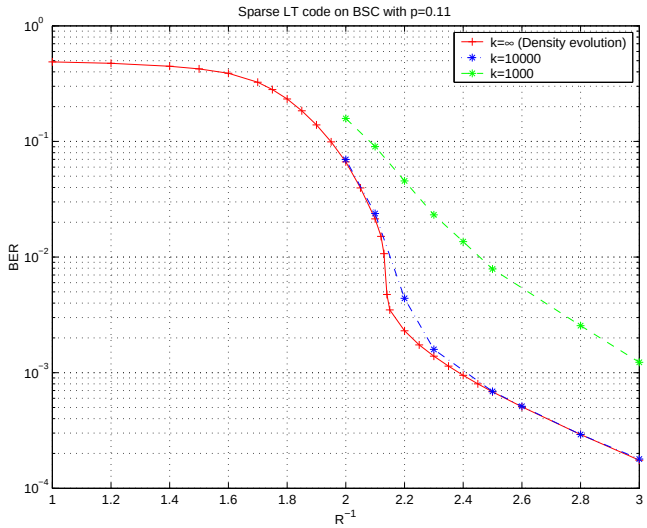


Figure 2.3: Performance of LT codes generated using right distribution given in equation (2.4) on BSC with $p = 0.11$.

2.4 Raptor codes

The error floors exhibited by LT codes suggest the use of an outer code. Indeed this is what Shokrollahi does in the case of the BEC [73, 74] where he introduces[2] the idea of raptor codes, which are LT codes combined with outer codes. Typically these outer codes are high rate LDPC codes. In this chapter, we use the distribution in equation (2.4) for the inner LT code. For the outer LDPC code, we follow Shokrollahi [73] and use a left regular distribution (node degree 4 for all nodes) and right Poisson (check nodes chosen randomly with a uniform distribution).

The encoder for such a raptor code works as follows: the k input bits are first encoded into k' bits to form a codeword of the outer LDPC code. These k' bits are then encoded into an infinite stream of bits using the rateless LT code. The decoder picks up a sufficient number (n) of output symbols, constructs a Tanner graph that incorporates both the outer LDPC code and the inner LT code, and decodes using belief propagation on this Tanner graph.

[2]We must mention here that Maymounkov [52] independently proposed the idea of using an outer code.

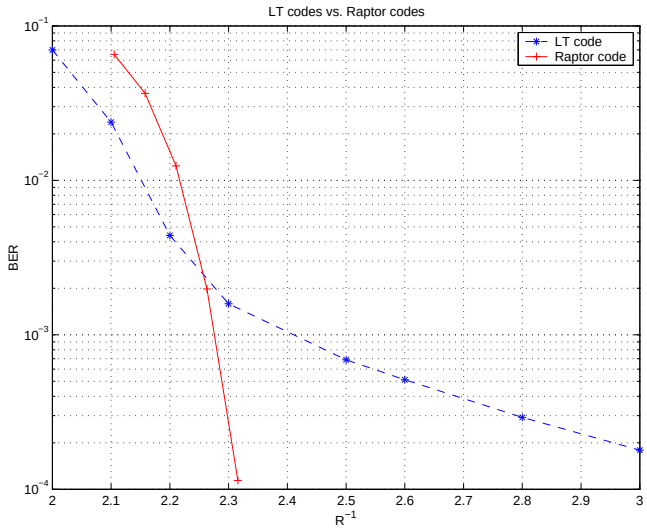
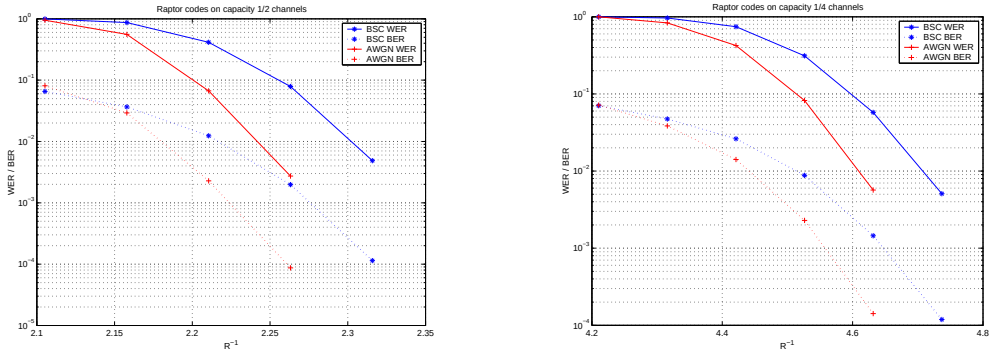


Figure 2.4: Comparing LT codes with raptor codes on a BSC with p=0.11. The LT code has $k = 10000$ and is generated using the distribution in equation (2.4). The raptor code has $k = 9500$ and has two components: an outer rate-0.95 LDPC code and an inner rateless LT code generated using the distribution in equation (2.4).

Simulation studies, such as the one shown in Figure 2.4, clearly indicate the superiority of raptor codes. Figure 2.4 shows a comparison between LT codes and raptor codes on a BSC with bit flip probability 0.11. The LT code has $k = 10000$ and is generated using the distribution in equation (2.4). The raptor code has $k = 9500$ and uses an outer LDPC code of rate 0.95 to get $k' = 10000$ encoded bits. These bits are then encoded using an inner LT code, again generated using the distribution in equation (2.4). Figure 2.4 clearly shows the advantage of using the outer high rate code.

Raptor codes not only beat LT codes comprehensively, but also have near-optimal performance on a wide variety of channels as shown in Figure 2.5, which shows the performance of the aforementioned raptor code on four different channels. On each of these channels, the raptor code has a waterfall region close to the Shannon capacity, with no noticeable error floors. Of course, this does not rule out error floors at lower WERs.

Another indicator of the performance of a rateless code on any given channel is the



(a) (b)

Figure 2.5: Performance of raptor code with k=9500 and k'=10000 on different channels: (a) BSC with $p = 0.11$ and AWGNC with $E_s/N_0 = -2.83dB$. Both channels have capacity 0.5. (b) on BSC with $p = 0.2145$ and AWGNC with $E_s/N_0 = -6.81dB$. Both channels have capacity 0.25

number of bits required for successful decoding. We must note here this indicator not only depends on the code, but also on the number of decoding attempts made by the receiver. For example, the decoder could attempt decoding each time it receives a new noisy bit. While such a decoder would be optimal in terms of number of bits required for successful decoding, it would have prohibitively high decoding complexity. A more practical decoder would wait for more bits to come in before decoding. Such a decoder would have a vastly lower complexity at the expense of slightly larger number of bits received. Note that there is no need for such a tradeoff in the case of the BEC. This is because the decoder fails when the Tanner graph is reduced to a *stopping set* [13]. After new bits are received, further decoding can be done on the stopping set instead of the original Tanner graph. Such a scheme is not appplicable to noisy channels where the decoder must start over every time new bits are received.

Figure 2.6 shows a histogram of the number of noisy bits needed for decoding the previously described raptor code with $k = 9500$. We observe that the expected number of noisy bits required for successful decoding (20737) is fairly close to the Shannon limit (19000).

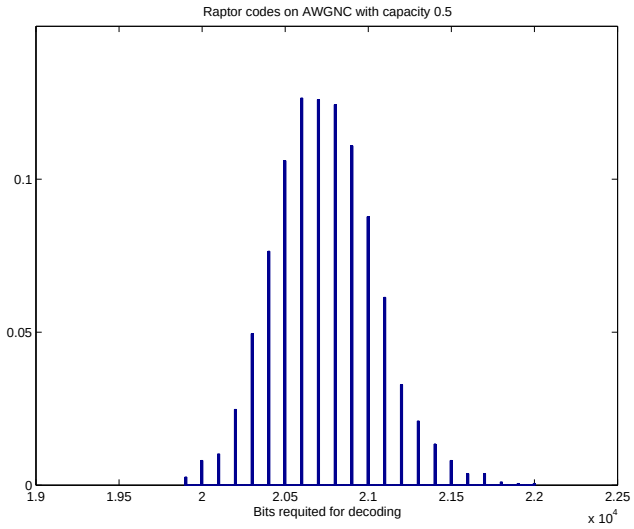


Figure 2.6: Histogram of number of bits required for successful decoding of raptor code with $k = 9500$ on an AWGNC with $E_s/N_0 = -2.83dB$. The capacity of this channel is 0.5. The receiver first attempts decoding after receiving 19000 noisy bits (Shannon limit). Whenever decoding fails, the receiver waits for another 100 bits before attempting to decode again.

2.5 Conclusion

We have conducted simulation studies and density evolution analysis of rateless codes on channels such as the BSC and the AWGNC. We found that raptor codes have excellent performance on such channels, while the performance of LT codes is not as good. These results suggest that raptor codes are ideal for use in data transfer protocols on noisy channels. A similar observation has already been made on the BEC [73] and consequently, commercial applications that use raptor codes on the Internet are already in the market.

We must point out here that we have not explicitly designed any forward error correction based data transfer scheme for noisy channels. We have only shown that raptor codes are likely to outperform any other known class of rateless codes in such a scheme. Therefore, a natural direction for future work is the design of a raptor code based protocol and a study of its performance on relevant noisy channels, such as fading channels.

Chapter 3 Graph-based codes for synchronous multiple access channels

In this chapter, we discuss a general algorithm for using LDPC-like codes on a synchronous multiple access channel (MAC). We then introduce a code design procedure known as graph-splitting, and show that codes designed using this technique achieve capacity on the binary adder channel (BAC) without using timesharing. Finally, we present simulation results for the noisy binary adder channel, qualitatively analyze these results and argue that LDPC-like codes perform well on multiple access channels.

3.1 Introduction to multiple access channels

A multiple access channel (MAC) [10, Section 14.3] is defined as a channel in which two or more senders send information to the same receiver. Examples include a satellite receiver with many independent ground stations or a cellular base station receiving inputs from many cell phones. In these channels, the senders must not only contend with the receiver noise, but also interference from each other. In mathematical terms, a discrete memoryless MAC is defined as a channel that takes in n inputs $x_1 \in \mathcal{X}_1, x_2 \in \mathcal{X}_2, \ldots, x_n \in \mathcal{X}_n$ and produces an output $y \in \mathcal{Y}$ according to a probability transition matrix $p(y|x_1, x_2, \ldots, x_n)$.

Several information theoretic results are known about the MAC, the most important one being about its capacity. The capacity region of a two-user MAC $(\mathcal{X}_1 \times \mathcal{X}_2, p(y|x_1, x_2), \mathcal{Y})$ is the closure of the convex hull of all rate pairs (R_1, R_2)

satisfying

$$\begin{aligned} R_1 &< I(X_1; Y|X_2), \\ R_2 &< I(X_2; Y|X_1), \\ R_1 + R_2 &< I(X_1, X_2; Y) \end{aligned} \tag{3.1}$$

for some product distribution $p_1(x_1)p_2(x_2)$ on $\mathcal{X}_1 \times \mathcal{X}_2$. A detailed proof of this result is found in [10, Section 14.3]. The key point to note here is that the input distribution should be a product distribution. This reflects the fact that the inputs X_1 and X_2 come from different users.

The MACs we consider in this chapter are synchronous, i.e., all users share a common clock. In a synchronous MAC, we can assume that the codeword length and codeword boundaries are the same for every user. We must mention the capacity region of an asynchronous MAC is the same as that of the corresponding synchronous MAC [11]; however, the coding scheme that achieves capacity is much simpler in the synchronous case.

3.2 Decoding LDPC codes on a MAC

In this section, we discuss a general algorithm for using low density parity check (LDPC) codes on a binary input two-user MAC. Assume that User 1 and User 2 encode their respective data independently using two distinct LDPC codes with same blocklength n and rates R_1 and R_2. At the channel output, we get a sequence of symbols which is a probabilistic function of the two transmitted codewords. Based on the received channel symbol, we can compute channel information by using Bayes' rule.

$$p_{ch}(x_1, x_2|y) = \frac{p_1(x_1)p_2(x_2)\ p(y|x_1, x_2)}{p(y)} \propto p(y|x_1, x_2) \tag{3.2}$$

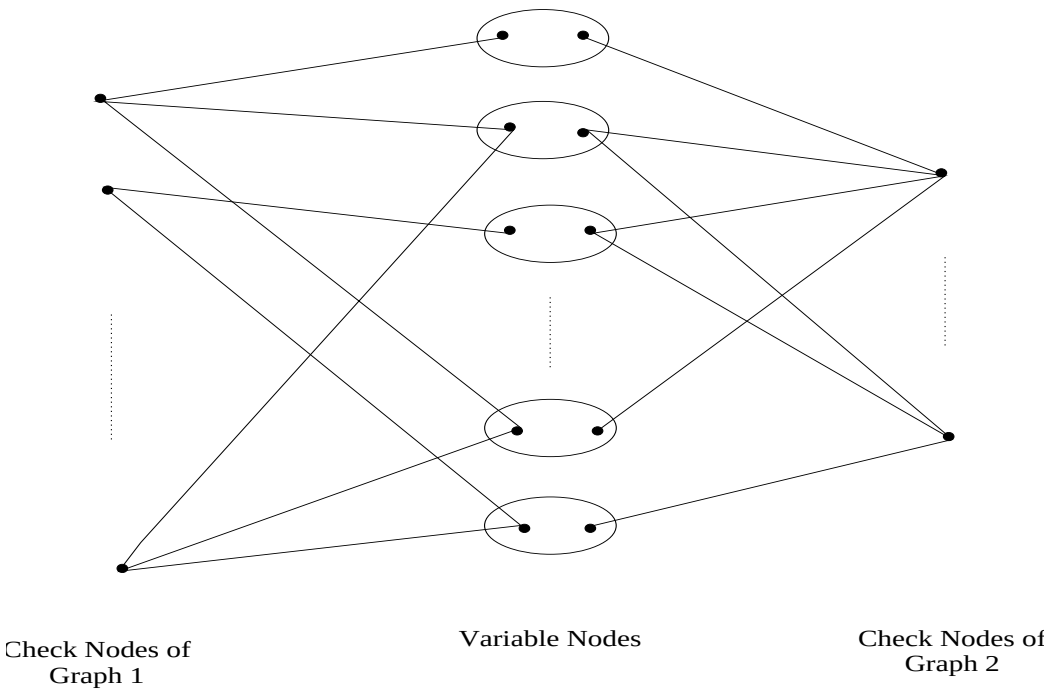


Figure 3.1: LDPC codes on a MAC: Each user encodes his data independently using an LDPC code, but the channel gives an output dependent on the bits from both users.

where $x_1, x_2 \in \{0, 1\}$ and $y \in \mathcal{Y}$. Note that we have a joint distribution on the possible channel inputs.

The belief propagation decoding algorithm proceeds as follows. At a check node (of either Graph 1 or Graph 2), the update rule is the same as in a single user LDPC code. The incoming bits to any check node should sum to zero, therefore the outgoing message along any edge is the convolution of the messages along all other incoming edges. Since the pmfs get convolved, their Fourier transforms get multiplied and therefore the outgoing message along any edge i is given by

$$P_{i_{out}}(x) = \prod_{j \neq i} P_{j_{in}}(x) \tag{3.3}$$

where P is the Fourier transform of the distribution p. Note that $(P(0), P(1)) = (p(0) + p(1), p(0) - p(1)) = (1, p(0) - p(1))$

At the variable nodes, each incoming message along the edges of Graph 1 is an independent estimate of the x_1. Similarly, each incoming message along the edges of Graph 2 is an independent estimate of x_2. Moreover, if the two graphs are also designed randomly, then with high probability, the messages coming in from both the graphs are also independent of each other. In addition to these messages, we also have the joint channel information. Therefore, by the usual belief propagation rule, we get the outgoing joint message along an edge i of Graph 1 to be

$$p_{i_{out}}(x_1, x_2) \propto p_{ch}(x_1, x_2) \prod_{j_1 \neq i} p_{j_1}(x_1) \prod_{j_2} p_{j_2}(x_2) \tag{3.4}$$

Thus, the joint message passed along an edge of Graph 1 depends on the channel information, all the incoming messages along the edges of Graph 2 connected to the corresponding variable node, and the incoming messages along all the edges of Graph 1 (except the edge along which the outgoing message is passed) connected to the same variable node. However, we do not want to pass this joint distribution along Graph 1 since we just need to pass a message about x_1, so we marginalize the joint distribution to get

$$p_{i_{out}}(x_1) \propto \prod_{j_1 \neq i} p_{j_1}(x_1) \sum_{x_2=0}^{1} \left(p_{ch}(x_1, x_2) \prod_{j_2} p_{j_2}(x_2) \right) \tag{3.5}$$

Now we define the updated channel information $p_{ch}^{'}$ as follows:

$$p_{1_{ch}}^{'}(x_1) \propto \sum_{x_2=0}^{1} \left(p_{ch}(x_1, x_2) \prod_{j_2} p_{j_2}(x_2) \right) \tag{3.6}$$

Note that $p_{ch}^{'}$ does not depend on the edge i along which the message is to be passed. In this way, it is similar to channel information in a single user LDPC code. However, the updated channel information depends on the messages being passed from Graph 2.

After computing the updated channel information, we pass messages as in a single-user LDPC code. The message along edge i is

$$p_{i_{out}}(x_1) \propto p^{'}_{1_{ch}}(x_1) \prod_{j_1 \neq i} p_{j_{1_{in}}}(x_1) \tag{3.7}$$

Similar update rules apply to Graph 2; using the incoming messages from Graph 1 and the joint channel information, we compute the updated channel information for Decoder 2.

$$p^{'}_{2_{ch}}(x_2) \propto \sum_{x_1=0}^{1} \left(p_{ch}(x_1, x_2) \prod_{j_1} p_{j_1}(x_1) \right) \tag{3.8}$$

Then we pass messages on Graph 2 similar to a usual single-user LDPC code, i.e., in a manner similar to that in equation 3.7.

$$p_{i_{out}}(x_2) \propto p^{'}_{2_{ch}}(x_2) \prod_{j_2 \neq i} p_{j_{2_{in}}}(x_2) \tag{3.9}$$

We repeat the above iteration steps till a criterion for stopping is met. For example, the algorithm can be stopped when all the parity checks are satisfied or after a fixed number of iterations.

Thus the decoder for the MAC is the same as two single-user LDPC decoders with each decoder updating the effective channel information for the other code at each iteration. The MAC is very similar to the turbo decoder used to decode concatenated codes; just like the MAC decoder, a turbo decoder is comprised of two or more simple component decoders operating in tandem [5].

3.3 The binary adder channel

One of the simplest examples of a MAC is the binary adder channel (BAC). This channel has binary inputs $x_1, x_2 \in \{0, 1\}$ and a ternary output $y = x_1 + x_2$ where the

addition is over the real field. There is no ambiguity in (x_1, x_2) if $y = 0$ or if $y = 2$. However $y = 1$ can result from either $(x_1, x_2) = (0, 1)$ or $(x_1, x_2) = (1, 0)$. Thus the inputs are determined for two of the possible output values, but are ambiguous for the third. This is why the BAC is sometimes referred to as the binary erasure multiple access channel.

The capacity of the BAC can be computed very easily from equations (3.1). It turns out the capacity of the BAC [10] is given by

$$R_1 < 1,\ R_2 < 1,\ R_1 + R_2 < 1.5 \tag{3.10}$$

Therefore it is possible to achieve a joint rate of 1.5 bits per channel use. We now consider our algorithm on the BAC. Suppose User 1 encodes his data with an LDPC code of rate R_1 and User 2 does the same with a code of rate R_2. Let n be the blocklength of both the codes. Since both bitstreams are independent, the weak law of large numbers tells us that with high probability, the channel input would have an equal number (i.e., $n/4$) of the four possible bit pairs $(0, 0), (0, 1), (1, 0)$ and $(1, 1)$. Since $(0, 1)$ and $(1, 0)$ result in the same output $Y = 1$, at the channel output we would see $n/4$ 0s, $n/2$ 1s, and $n/4$ 2s. To either decoder, this looks like $n/4$ 0s, $n/4$ 1s and $n/2$ erasures, i.e., like the output of a erasure channel with probability of erasure $1/2$. However, the erasures input to both the decoders are dependent and decoding an erasure in one user's codeword would automatically decode the corresponding erasure in the other user's codeword.

The belief propagation algorithm used by either user is the same as that used to decode a usual erasure correcting LDPC code. In an erasure correcting code, if an erasure is decoded to a 0 (or 1), then it always stays a 0 (or 1), i.e., no mistakes are ever made by the decoder. Therefore, the decoding algorithm on the BEC is: (1) at an erased variable node, if at least one incoming message is a non-erasure, decode the

erasure to that incoming non-erasure, (2) at a check node, if all except one incoming messages are non-erasures, then decode that erasure to be the sum over $GF(2)$ of all the other incoming messages. If there are more than one incoming erasures, then the check node just passes back erasures.

Our algorithm for the BAC works exactly the same way, except that the two LDPC decoders interact. The algorithm proceeds as follows: Do one iteration on each graph, thereby correcting some erasures on the input to Graph 1 and some other erasures on the input to Graph 2. Now erasures corrected on one graph are corrected on the other also. This corresponds to the "update channel information" step of the algorithm for the general MAC. Repeat this procedure till there are no erasures left or till no further erasures can be corrected.

3.4 Design of LDPC codes for the BAC

All that remains to be done is to find good LPDC codes for both the users. For this purpose, we introduce a graph-splitting design. We start with a good erasure correcting code of rate $R < 1/2$ having a Tanner graph $\mathcal{G}$. We first split the set of check nodes $\mathcal{C}$ into two disjoint sets $\mathcal{C}_1$ and $\mathcal{C}_2$ having c_1 and c_2 check nodes respectively. We now split the graph $\mathcal{G}$ into two disjoint graphs $\mathcal{G}_1$ and $\mathcal{G}_2$ where $\mathcal{G}_1$ is the set of all edges connected to the check nodes in $\mathcal{C}_1$ and $\mathcal{G}_2$ is the set of all edges connected to the check nodes in $\mathcal{C}_2$. Now let $\mathcal{G}_1$ be the Tanner graph of User 1's LDPC code and $\mathcal{G}_2$ be the Tanner graph of User 2's LDPC code. The rates of the codes are given by

$$R_1 = 1 - c_1/n, \;\; R_2 = 1 - c_2/n \tag{3.11}$$

$$R_1 + R_2 = 2 - (c_1 + c_2)/n = 1 + (1 - c/n) = 1 + R \tag{3.12}$$

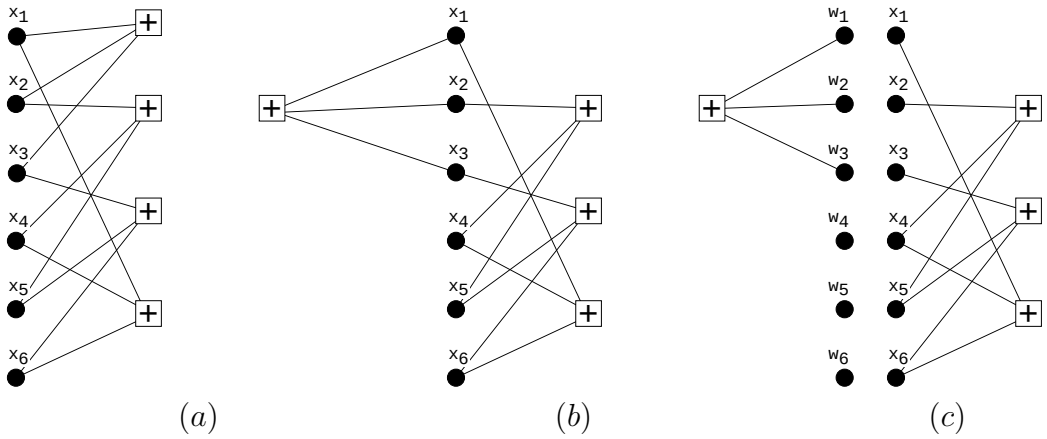
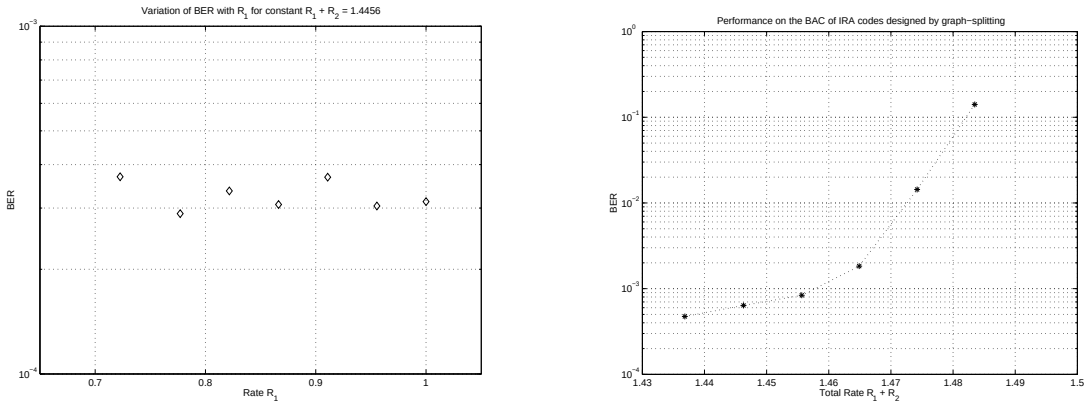


Figure 3.2: The graph-splitting technique: (a) Start with the Tanner graph of a single user LDPC code (b) Split the graph into two subgraphs containing disjoint sets of check nodes (c) The two subgraphs are now the the Tanner graphs for the two users.

Thus, splitting a rate-R graph gives a joint graph of rate $1 + R$. Therefore, if we split a "good" rate-1/2 erasure code, we get a joint rate of 1.5, which is the capacity of the BAC. What remains to be shown, of course, is that this graph-splitting design works. This is done by density evolution [66] analysis of the joint decoder.

3.4.1 Density evolution on the BAC

Suppose the probability that an erasure is passed along an edge of a graph to a check node is x. The message passed out of a check node is also an erasure when at least one of the other messages is an erasure. This occurs with probability $y = 1 - \rho(1 - x)$, where ρ is the right degree sequence polynomial of the graph and equals x^{a-1} in the case of a right-regular graph. Now at the variable nodes, an erasure is passed out when all the incoming messages from both graphs and the channel are erasures. This occurs with probability $\lambda(y)$, where λ is the connectivity polynomial describing the probability distribution on the number of edges in both graphs that an edge is connected to. But this is nothing but the left-degree sequence λ of the original graph



(a) (b)

Figure 3.3: Performance of graph-split codes: (a) Variation of BER with R_1 for constant overall rate R (b) Variation of BER with total rate R at rates close to capacity.

$\mathcal{G}$. Thus the density evolution goes as follows:

$$x \longrightarrow 1 - \rho(1 - x) \quad \text{at the check nodes.} \tag{3.13}$$

$$x \longrightarrow 0.5\lambda(x) \quad \text{at the variable nodes.} \tag{3.14}$$

$$x \longrightarrow 0.5\lambda(1 - \rho(1 - x)) \text{ in one iteration.} \tag{3.15}$$

Note that this is the exactly the same as the density evolution of the probability of erasure when the code with Tanner graph $\mathcal{G}$ is used on a binary erasure channel (BEC) with probability of erasure 0.5. Therefore, if we design a capacity achieving rate-1/2 code on the BEC and split its graph, then we get a pair of capacity achieving codes on the BAC. Recent research on graph-based codes has showed that such erasure correcting codes indeed exist. Starting with the codes of Luby et al. [45], various linear-time decodable capacity-achieving graph-based codes have been constructed. These include Shokrollahi's right-regular LDPC codes [72] and irregular repeat-accumulate (IRA) codes introduced by Jin, Khandekar and McEliece [33]. All the above codes essentially use belief-propagation decoding. Using techniques similar to density evolution [66], it has been shown (by the respective authors) that these

codes achieve capacity on the BEC. Therefore, we can state the following result: Codes designed by graph-splitting achieve capacity on the binary adder channel.

We used irregular repeat accumulate (IRA) codes [33] to test our graph-splitting code design and decoding algorithm on the BAC. IRA codes are very similar to LDPC codes, and have the extra advantage that they can be encoded in linear time. We used codes of length 10000 and constant right-degree 5 IRA codes. The simulations confirm the fact that the final probability of error does not depend on the individual R_1 and R_2, but only on the joint rate $R_1 + R_2$, i.e., it depends only on the graph $\mathcal{G}$ and not the way it is split into subgraphs (see Figure 3.3a). A graph showing the performance of the codes at joint rates close to capacity is also shown (see Figure 3.3b).

3.5 The noisy binary adder channel

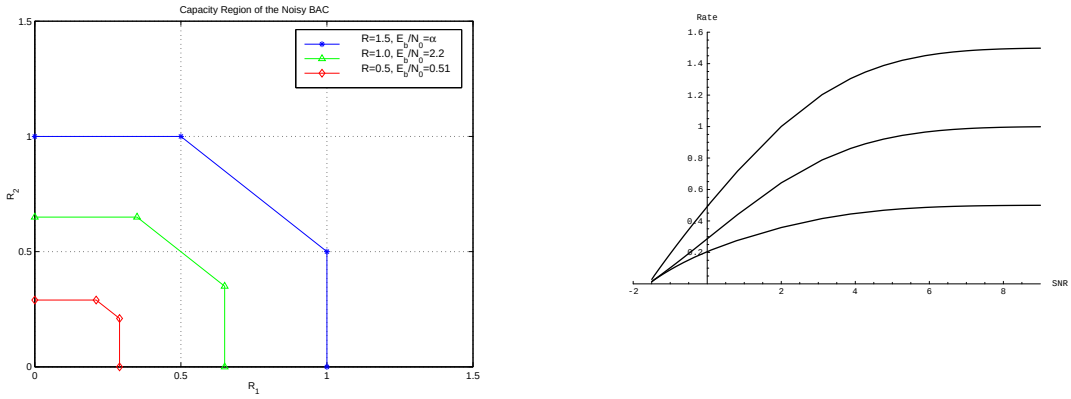
We now consider the noisy binary adder channel (noisy BAC) whose output y is given by

$$y = x_1 + x_2 + n \tag{3.16}$$

where $x_1, x_2 \in \{-1, +1\}$ are the inputs from the users and n is a zero-mean Gaussian random variable with variance σ^2. Thus, in addition to the interference between the two users, noise is also present at the receiver. The variation of the capacity region of the noisy BAC with $E_b/N_0 = 10\log_{10}(1/(R_1 + R_2)\sigma^2)$ is shown in Figure 3.4.

We now study the performance of LDPC-like codes on this channel. Our first choice for the two users' codes would be those designed by graph-splitting, since they work well on the BAC. However, it turns out that these codes do not work at all on the noisy BAC. More precisely, we can show that the bit error probability is bounded away from zero for any noise variance $\sigma^2 > 0$.

The proof relies on the fact that there are "unprotected variable nodes" that result



(a) (b)

Figure 3.4: Capacity region of the noisy BAC: (a) The capacity region of the noisy BAC for three values of E_b/N_0 (b) Plot showing the maximum achievable rate for a given E_b/N_0 and the R_1, R_2 coordinates of an extremal point of the capacity region at that E_b/N_0.

from the graph-splitting design. By splitting any graph $\mathcal{G}$, there would be a non-zero fraction f of nodes for which all of the connected edges are in one of the subgraphs, say $\mathcal{G}_2$ (see Figure 3.2). For these nodes, only x_2 (viz., the node with the edges connected) can be corrected because only it participates in the belief propagation process. Now even if we assume x_2 is perfectly decoded to $+1$ or -1 (which certainly need not be the case), the resulting effective single user channel for User 1 is $y = x_1 + n$. Among the unprotected bits of User 1, there will be a non-zero fraction $Q(\sigma^{-1})$ of errors. On the whole there will at least be a non-zero fraction of errors $fQ(\sigma^{-1}) > 0$ of errors in User 1's bit stream. Therefore graph-splitting in its present form does work for the noisy BAC. A simple extension of this argument will show that this is the case for any noisy MAC, i.e., a MAC in which the output is not a deterministic function of both the inputs.

The next choice would be independently designed single-user LDPC codes with parameters $(\lambda_1(x), \rho_1(x))$ and $(\lambda_2(x), \rho_2(x))$. Assuming the coefficients of x^0 and x^1 in $\lambda_1(x)$ and $\lambda_2(x)$ are zero (an essential condition for a good single-user LDPC code), we can show (proof omitted) using density evolution techniques that these codes do not

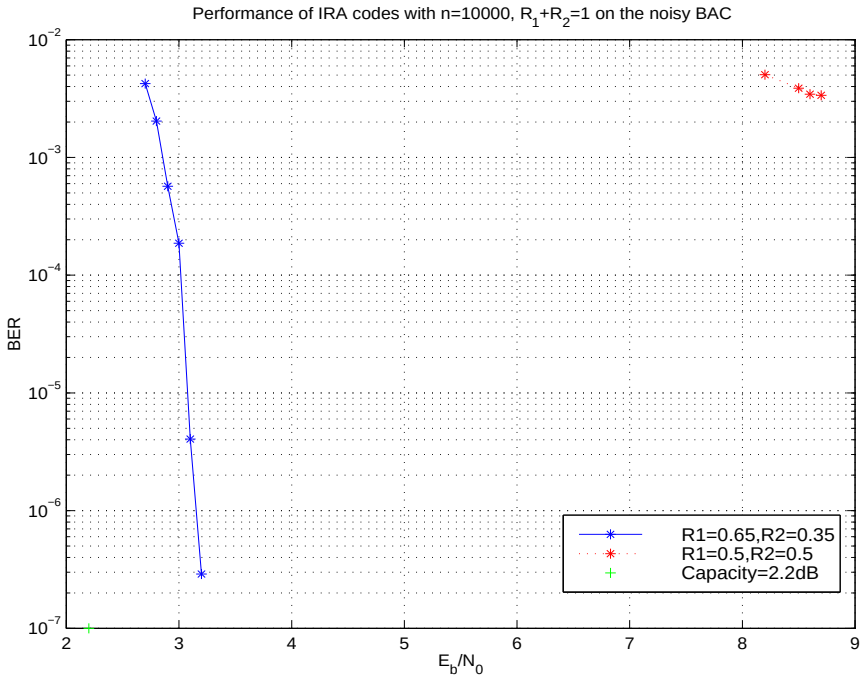


Figure 3.5: Performance of IRA Codes on the noisy BAC at $R_1 + R_2 = 1$.

achieve capacity on the BAC, except in the extreme case when $(R_1, R_2) = (1.0, 0.5)$ or $(0.5, 1.0)$. Therefore, we would expect something similar in the case of the noisy BAC as well.

We again used IRA codes to test this hypothesis. SNR vs. BER performance curves for total rate $R = 1$ are shown in Figure 3.5. As we can see, the independently designed codes perform well when the rates R_1 and R_2 are close to the extremal points of the capacity region and poorly when the rates R_1 and R_2 are equal.

The above result can be explained qualitatively as follows. An extremal point of any MAC is the rate pair $(I(X_1; Y), I(X_2; Y|X_1))$ (or the similar pair with X_1 and X_2 interchanged). This shows that X_1 can be decoded independently first, because the capacity of the single-user channel that User 1 sees is $I(X_1; Y)$. Based on the decoded X_1, we can decode X_2 because the capacity of the single-user channel that User 2 sees now is $I(X_2; Y|X_1)$, since X_1 has already been decoded. Therefore the joint decoder can be replaced by two single-user decoders, one operating after the

other. This method of decoding is known as successive cancellation or onion peeling [67]. Only the extremal points of the rate region of the MAC can be decoded by successive cancellation.

In the case of the independently designed LDPC codes on the noisy BAC, the extremal points can be decoded by using two single-user LDPC decoders. Since single-user LDPC codes are known (though not provably) to do extremely well, we expect good performance for these codes at the extremal point rates of the MAC as well. Note that we did not actually use a successive cancellation decoder in our simulations. Instead, we used the joint decoder described in Section 3.2. Since the joint decoder performs at least as well as the successive cancellation decoder, the performance for the extremal point is good. There is no reason however to believe that the same will be true for the non-extremal points. This explains the performance curves in Figure 3.5.

By timesharing the extremal rate pairs, we can achieve any rate pair in the capacity region. This shows that LDPC-like codes exhibit good performance for any rate-pair on the noisy BAC. Also note that all the above arguments apply to any MAC and not just the noisy BAC.

3.6 Conclusion

We studied a coding scheme that graph-based codes on synchronous MACs and saw how the decoder is equivalent to two or more single-user decoders updating each other's channel information. Our analysis and simulation studies showed that LDPC codes designed for single user channels perform well on MACs only at rates close to the extremal points of the capacity region of the MAC. Of course, this can be used in combination with timesharing to get good performance at any point in the achievable rate region of the MAC. If one were to insist on designing codes that do

not use timesharing, then special design techniques must be used. Graph-splitting is one such technique; however its applicability is limited to the binary adder channel. Other authors have designed codes for the noisy BAC [4]. However, a specialized system designed by such techniques offers only marginal gains over a system that timeshares single user codes with the extremal rates.

Chapter 4 Iterative decoding of multi-step majority logic decodable codes

In this chapter, we investigate the performance of iterative decoding algorithms for multi-step majority logic decodable (MSMLD) codes of intermediate length. We introduce a new bit-flipping algorithm that is able to decode these codes nearly as well as a maximum likelihood decoder on the binary symmetric channel. MSMLD codes decoded using bit-flipping algorithms can out-perform comparable BCH codes decoded using standard algebraic decoding algorithms, at least for high bit flip rates (or low and moderate signal to noise ratios).

4.1 Introduction

Recently, iterative decoding algorithms for low density parity check (LDPC) codes have received a great deal of attention. In [22, 23], a (J, L) LDPC code is defined as an (N, K, d) linear block code whose M x N parity check matrix H has J ones per column and L ones per row, where J and L are relatively small numbers. For large N, it is quite easy to avoid the occurrence of two check sums intersecting on more than one position when constructing H. In that case, the check sums are called orthogonal and the Tanner graph representation [80] of H has girth at least six. This is often considered to be an important feature for good performance of iterative decoding [48, 49].

In [36, 47, 83] it was shown that iterative decoding of one-step majority logic decodable codes also performed very well; indeed often better than for ordinary LDPC codes of similar blocklength and rate for lengths up to a few thousand bits. Despite

the fact that the parity check matrix of these codes has a higher density of ones than that of the original LDPC codes, the geometric structure guarantees a girth of six. Perhaps even more importantly, the matrix H used for decoding is highly redundant, i.e., $M > N - K$, and this feature seems to significantly help iterative decoding algorithms.

In this chapter, we investigate iterative decoding of multi-step majority logic decodable (MSMLD) codes for transmission over a binary symmetric channel (BSC). With the use of redundant H matrices, these codes have already been shown to perform relatively well on the additive white Gaussian noise (AWGN) channel [37, 42, 46, 79, 84]. However, whereas on the AWGN channel the performance of iterative decoding does not approach that of maximum likelihood decoding (MLD), we find that on the BSC, fast and low complexity bit flipping (BF) algorithms can achieve near MLD performance.

The chapter is organized as follows. After a brief review of MSMLD codes in Section 4.2, an improved version of the Gallager's bit flipping algorithm B is presented and analyzed in Section 4.3. Different decoding approaches exploiting the structure of MSMLD codes are proposed in Section 4.4. and simulation results are reported in Section 4.5. Possible extensions to iterative decoding of these codes for the AWGN channel are discussed in Section 4.6 and concluding remarks are finally given in Section 4.7.

4.2 A brief review of multi-step majority logic decodable codes

The most famous MSMLD codes are the Reed-Muller (RM) codes introduced in [56]. Motivated by the efficient multi-step majority logic decoding algorithm proposed in [65], several other classes of MSMLD codes were developed in the 1960's and 70's.

Many of these are based on constructions derived from finite geometries [6, 41, 55, 64]. Unfortunately, the minimum distance d of these codes does not compare favorably to that of their counterpart BCH codes. Consequently, when decoded using a t-bounded distance decoding (t-BDD) algorithm (i.e., when decoded up to the guaranteed error correcting capability t of the code), they are outperformed by BCH codes also decoded by a t-BDD algorithm.

One-step majority logic decodable codes can also be viewed as a special class of LDPC codes with orthogonal check sums. For example, a one step majority logic decodable Euclidean geometry (EG) code of length $N = 2^{ms} - 1$ over the finite field $GF(2^s)$ is also an LDPC code with

$$\begin{aligned} J &= \frac{2^{ms}-1}{2^s-1} - 1, \\ L &= 2^s. \end{aligned}$$

Iterative decoding of these codes has been shown to perform very well and most importantly for the BSC, is able to correctly decode many error patterns with considerably more than t errors.

The main feature in the construction of one-step majority logic decodable codes is the same as that of LDPC codes, that is the fact that each bit can be estimated by J check sums orthogonal on it. In constructing a μ-step majority logic decodable code, this principle is generalized into μ steps as follows: at step i, $1 \leq i \leq \mu$, the modulo-2 sum of K_i bits is estimated by J_i check sums orthogonal on these K_i positions, with $K_\mu = 1$, $K_i < K_{i-1}$, and $J_i \geq d - 1$. While μ-step majority logic decoding directly follows this construction method, its extension to an iterative decoding method is not straightforward for $\mu \geq 2$ because any graphical representation of H necessarily contains many four-cycles corresponding to check sums intersecting on K_1 positions.

In the following, we consider the family of μ-step majority logic decodable EG

codes since the same developments apply to other families of majority logic decodable codes.

4.3 Three-state decoding algorithm

In [22, 23], Gallager proposed two different BF algorithms. These algorithms are designed for LDPC codes with few check sums of low weight orthogonal on each bit and therefore, careful attention must be paid to the introduction of correlations in the iterative process. In particular, in Gallager's bit-flipping algorithms, he takes care that the "message" from a bit to its neighboring check should not directly depend on the message sent by that check back to the bit and vice versa. In our case, because of the very large number of check sums intersecting on each bit, we can neglect that refinement with negligible performance degradation, and obtain the following algorithm, which simplifies Gallager's algorithm B:

- For each check sum m and for each bit n in check sum m, compute the modulo-2 sum σ_{mn} of the initial value of bit n and of the other bit values computed at iteration-$(i-1)$.

- For each bit n, determine the number N_u of unsatisfied check sums σ_{mn} intersecting on it. If N_u is larger than some predetermined threshold b_1, invert the original received bit n, otherwise keep this value.

The use of a single threshold b_1 implies that bits with very different values N_u are viewed with the same reliability at the next iteration. While for the codes considered in [22, 23], N_u can take only a few different values, this is no longer the case for the codes considered in this chapter. It seems reasonable to try to reflect the differing reliablities of the bits in our algorithm. Consequently, we propose to modify the algorithm described above into the following "three-state" algorithm, which also allows

bits to be erased and check sums to be de-activated.

- For each check sum m and for each bit n in check sum m, compute the modulo-2 sum σ_{mn} of the initial value of bit n and of the other bit values computed at iteration-$(i-1)$. If any of these bits is erased, the check sum is de-activated.

- For each bit n, determine the number N_{ua} of unsatisfied activated check sums σ_{mn} intersecting on it.

  If $N_{ua} \geq b_1$, invert the original received bit n.

  If $b_1 > N_{ua} \geq b_2$, erase bit n.

  Otherwise keep the original received bit n.

Empirically, we find that the three-state algorithm performs best when the thresholds b_1 and b_2 are functions of the iteration number. Unfortunately, there are many ways to do this, and we only could roughly optimize to find the best schedules, but fortunately the performance seems to be a rather insensitive function of the choice made. For our schedules, we typically chose to begin at the first iteration with b_1 equal to the maximum possible number of unsatisfied checks J, and with $b_2 \approx b_1 - J/15$, and then to decrease b_1 and b_2 by the same small fixed integer (say one to five) at each iteration, continuing to decrease their values until they reach zero.

The proposed three-state approach can also be applied in a straightforward way to Gallager's original algorithm B. In fact, for a theoretical analysis, only this version is meaningful since the simplified algorithm introduces correlation and it is not known how to handle correlated values in the analysis of an iterative decoding algorithm in general. In that case, the three-state algorithm becomes a generalized version of the algorithm described in [66, Example 5], where $b_2 = b_1 - 1$. Consequently, if we assume the graph representation of the code is a tree, the same approach as in [66] can be used to analyze the three-state algorithm.

4.4 Decoding approaches

4.4.1 Fixed cost approaches

Direct approach

A μ-step majority logic decodable EG code can be represented by its M x N incidence matrix H in which rows represent μ-flats and columns points, with $h_{ij} = 1$ if the j-th point belongs to the i-th μ-flat.

A straightforward approach is to run the BF algorithm based on H. This matrix will be plagued by many four-cycles, but fortunately it can also be made very redundant with $M >> N$, and the weight of each row of the parity check matrix need not be too high. Furthermore, by exploiting the cyclic structure of the code, a very balanced graph is obtained so that the same speed of convergence can be expected in all parts of the graph.

Multi-step approach

In [84], a general method was presented for modifying the parity check matrix of a code to make it more suitable for iterative message-passing algorithms. Using this method on a two-step majority logic decodable EG code, one obtains a new parity check matrix whose graphical representation contains no four-cycles. It is a (M_1+M_2) x $(N_1 + N_2)$ matrix

$$H = \begin{bmatrix} A & B \\ D & C \end{bmatrix}, \tag{4.1}$$

in which the M_1 and M_2 rows represent the plane constraints and line constraints, respectively, and the N_1 and N_2 columns represent the points and lines, respectively. As a result, $M_2 = N_2$ and C represents the identity matrix, A is the all-0 matrix while

the remaining matrices B and D are free of four-cycles (and so is H). Generalization of (4.1) to μ-step majority logic decodable EG codes is straightforward.

Decoding based on (4.1) can be realized in at least two ways. First the BF algorithm can be run on H with the N_2 nodes corresponding to the lines initialized without a-priori knowledge. The drawback of this approach is that nodes with no a priori information from the channel directly exchange highly unreliable information with each other.

To overcome this problem, H can be modified so that each row of B has weight one. If a plane is composed of l lines, this corresponds to duplicating each plane l times and viewing it as the union of one line and of the points composing the remaining $l-1$ lines. As a result, nodes without a-priori information no longer directly exchange information, but the graph representation of the resulting matrix A now contains many four-cycles. The BF algorithm can then be decomposed in two steps based on the following scheduling: in step-1, only the top part $[AB]$ of H is used to estimate the N_2 lines, while in step-2, the bottom part $[CD]$ is used to estimate the N_1 points. We notice that this scheduling "mimics" two-step majority logic decoding and can be easily generalized to μ steps for μ-step majority logic decodable codes.

Decomposable approach

By their construction, several μ-step majority logic decodable codes have a decomposable structure. For example, Reed-Muller (RM) codes can be constructed by the $|u|u \oplus v|$ construction or the iterative squaring construction [19]. For simplicity, we consider the $|u|u \oplus v|$ construction in the following. If C_1 and C_2 are two codes with parity check matrices H_1 and H_2, respectively, then $C = |C_1|C_1 \oplus C_2|$ has parity check

matrix

$$H = \begin{bmatrix} H_2 & H_2 \\ H_1 & 0 \end{bmatrix}. \tag{4.2}$$

Following the approach described in [20, 30], two stage decoding based on (4.2) is performed as follows. Assuming the received sequence corresponding to the codeword $|u_1|u_1 \oplus u_2|$ is $y = |y_1|y_2|$, first $y_1 \oplus y_2$ is decoded by a BF algorithm based on H_2 to estimate $\hat{u}_2$. Then $|y_1|y_2 \oplus \hat{u}_2|$ is decoded by the three-state BF algorithm of Section 4.3 based on H_1 to estimate $\hat{u}_1$. At the initialization of this second decoding stage, the values which coincide in y_1 and $y_2 \oplus \hat{u}_2$ are conserved, while the other values are erased.

4.4.2 Variable cost approach

The matrix H used for decoding is generally highly redundant, so that $M >> N$. If a sufficient number of check sums is used, then the BF algorithm converges rapidly to its final solution while if not enough check sums are used, the BF algorithm generally never converges to a codeword. In this latter case, a decoding failure is detected.

This observation suggests a "call by the need" algorithm in which, for $M_a < M_b < \cdots < M$, M_a check sums are initially used for N_a iterations. If the algorithm converges to a codeword, correct decoding is assumed; otherwise, the algorithm is reinitialized (not continued) and performed based on M_b check sums during N_b iterations. This process is repeated until either a codeword is found, or all M check sums have been used without success, in which case the decoding fails.

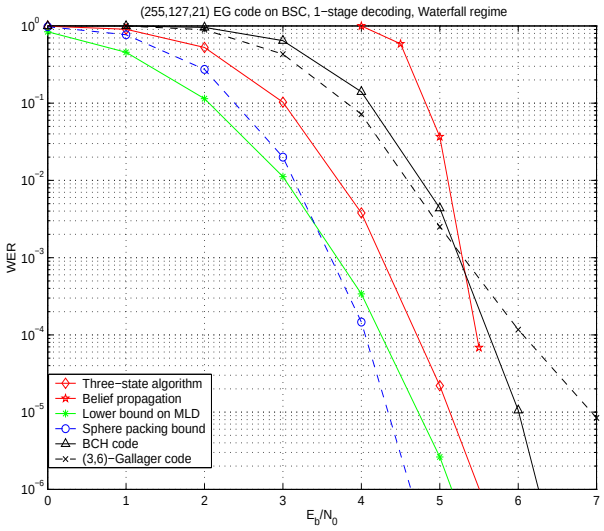


Figure 4.1: BF decoding of the (255,127,21) EG code; (a) low SNR regime.

4.5 Simulation results

We assume a BSC obtained from BPSK signaling, so that for a code of rate R, we have $p_0 = \mathrm{Q}\left(\sqrt{RE_b/N_0}\right)$, where E_b/N_0 is the signal to noise ratio (SNR) per information bit.

4.5.1 (255,127,21) EG code

In Figure 4.1, the simulated error performance of three-state BF decoding of the (255,127,21) EG code with the direct approach of Section 4.4.1 is compared to t-BDD of its (255,123,39) BCH code counterpart as well as its $(3, 6)$ Gallager LDPC code counterpart. This EG code corresponds to a $\mu = 2$ Euclidean geometry with 255 points and 5355 planes, so we can construct a parity check matrix H with 5355 rows and 255 columns. We observe that three-state BF decoding of the EG code not only outperforms its counterparts at the SNR values represented, but also remains quite close to the sphere packing bound (SPB), also represented in Figure 4.1. In fact, a lower bound on the MLD failure rate for this code was computed by checking

whether the decoding errors were also MLD errors (with unbiased recording of the ties). This bound is represented in Figure 4.1. One can see that the performance of the three-state BF algorithm must be very close (within a few tenths of a dB) of MLD performance. The error performance of the standard sum-product or belief propagation (BP) algorithm, initialized with the crossover probability p_0 of the BSC is also shown in Figure 4.1. The reasons for the much worse performance of BP at low SNR's are elaborated in Section 4.6.

We also mention that the advantage of the three-state BF algorithm over Gallager's algorithm B is a reduction factor that ranges between two and five in the number of errors. This gain is small, but remains non-negligible in approaching MLD performance, especially since the three-state algorithm is not much harder to implement than Gallager's algorithm B.

Since this code is two-step majority logic decodable, the two-step approach of Section 4.4.1 was also implemented. The decomposition of [84] gives an 32,130 x 5610 matrix. Each row corresponding to one of the of 26,775 plane constraints in this matrix has to be duplicated four times to have weight one in the B-part of (4.1). The final matrix H given by (4.1) becomes an 112,455 x 5610 matrix. Unfortunately, despite the large increase in complexity, only a tiny improvement was obtained by this approach. One explanation is that the multi-step approach can be viewed as a particular scheduling of the direct approach in which hidden nodes are introduced as intermediary states. As a result, the information initially available is used in successive steps rather than at once as in the direct approach. In the case of a binary erasure channel (BEC), the increased number of constraints and erasures associated with the multi-step approach helps in improving the decoding as information can only be improved [84]. However, for the BSC (or other channels introducing errors), erroneous decisions can propagate through the hidden nodes so that using all available information at once in a suboptimum way becomes as good as using it partially in

a more optimum (but still suboptimum after iteration-1) way. The only advantage of the multi-step approach is its guarantee to perform no worse than t-BDD since its first iteration can be made equivalent to multi-step majority logic decoding with $b_1 = b_2 = \lceil J/2 \rceil$.

In Figure 4.2, we plot the performance of the three-state BF decoding algorithm for the (255,127,21) EG code into the very high SNR, or low decoding failure, regime. These plots actually show the performance of the two-step algorithm described above, but as mentioned already, the difference in performance between the direct 3-state algorithm and the more complex two-step algorithm is tiny. At all word error rates (WERs) down to 10^{-20}, this difference is less than 0.1 dB.

To obtain these performance curves, we randomly generated random errors of fixed weight w, $w > t$ and for each weight w, evaluated the corresponding error performance $P_s(w)$. The overall error performance P_s was then obtained by the average

$$P_s = \sum_{w=t+1}^{N} P_s(w) \binom{N}{w} p_0^w (1-p_0)^{N-w}. \tag{4.3}$$

The results are reported in Figure 4.3. Since for WERs larger than 10^{-6}, no reliable evaluation of $P_s(w)$ is possible, we computed: (a) an upper bound on (4.3) by assuming the same $P_s(w_{min})$ as the smallest simulated for weights $w^{'}$, $t < w^{'} < w_{min}$; (b) a lower bound on (4.3) by assuming $P_s(w^{'}) = 0$ for weights $w^{'}$, $t < w^{'} < w_{min}$; and (c) an approximation by extrapolating $P_s(w^{'})$ for weights $w^{'}$, $t < w^{'} < w_{min}$. A pessimistic lower bound on MLD was also obtained from the lower bound on P_s. From Figure 4.2, we conclude that the three-state BF for the (255,127,21) EG code outperforms t-BDD of its BCH counterpart down to a WER of about 10^{-13} for the two-step approach (and 10^{-12} for the direct approach).

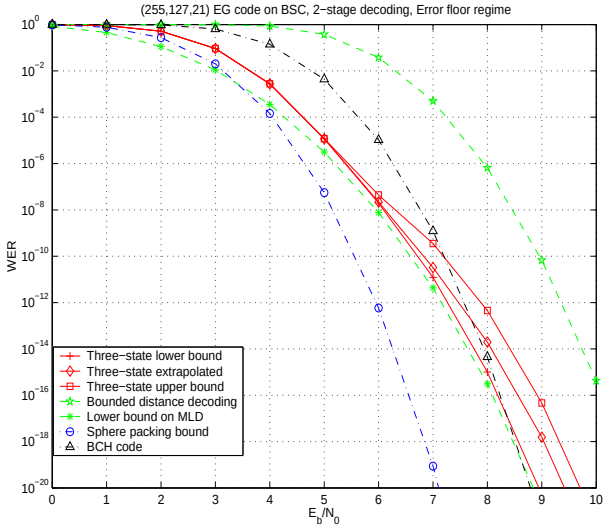


Figure 4.2: BF decoding of the (255,127,21) EG code; (b) high SNR regime.

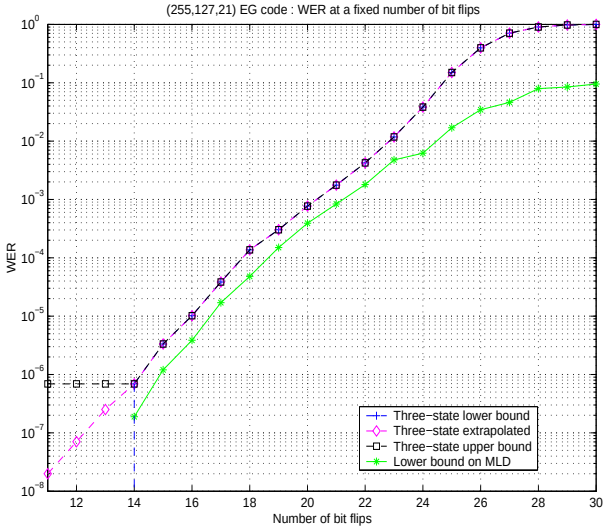


Figure 4.3: BF decoding of the (255,127,21) EG code for fixed number of errors.

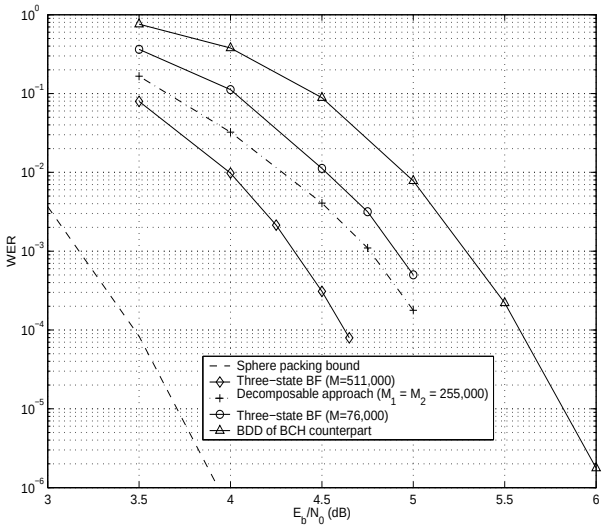


Figure 4.4: BF decoding of the (511,256,31) EG (or RM) code.

4.5.2 (511,256,31) EG (RM) code

Figure 4.4 depicts the performance of three-state BF decoding of the (511,256,31) EG (or RM) code with the direct approach of Section 4.4.1 and the decomposable approach of Section 4.4.1 based on the $|u|u \oplus v|$ construction. For comparison, the SPB and t-BDD of the counterpart (511,250,63) BCH code have also been represented.

For the direct approach, $M = 76,650$ and $M = 511,000$ have been considered (corresponding to 150 and 1000 different cyclic shifts of weight-32 codewords of the dual code, respectively). Also the progressive method was used to speed up each decoding. In both case, we chose five different sizes of the set of check sums used, namely, $M_a = 5110$; $M_b = 12,775$; $M_c = 22,550$; and $M_d = 51,000$. For each size, at most 10 iterations were performed. The value b_1 was set to the maximum number of unsatisfied check sums at each initial iteration and decreased by one (or a small number) at each subsequent iteration while we chose $b_2 = b_1 - 20$. Again these values were not thoroughly optimized so that additional secondary gains should be achievable.

The application of the progressive method is validated by the fact that for $M =$

$76,650$, no undetected error was recorded at all simulated SNR values. For $M = 511,000$, at the SNR value of 4.5 dB, about 10% of the errors were undetected (all of them occurring when all check sums were considered) and at this SNR value, one out of the 100 errors recorded was recognized as an MLD error. At lower SNR values, no undetected errors and no MLD errors were recorded. While a reasonably good error performance is achieved, we are clearly not able to obtain a tight bound on MLD performance. Because the three-state BF algorithm has a very low word error rate even for error patterns with a number of bit flips far beyond the guaranteed error-correcting capability t of the code, we are also not able to meaningfully repeat the analysis of the very high SNR regime. We also observe that despite the fact that the minimum distance of this code is about half of that of its BCH counterpart, iterative BF decoding of this EG code can easily outperform t-BDD of its BCH counterpart and approaches relatively closely the SPB at the WERs represented in Figure 4.4.

The decomposable approach of Section 4.4.1 was also tried with C_1 and C_2 being the (255,163,15) and (255,93,31) RM codes, respectively (resulting in a (510,256,30) code). At each stage, at most $M_1 = M_2 = 255,000$ check sums were considered. Again the progressive approach was used with all previous sizes of check sum sets divided by two. When decoded separately with $M = 255,000$, about 95% of the errors are undetectable errors, and about 40% of the errors are MLD errors for the (255,163,15) RM code. For the (255,93,31) RM code, about 80-90% of the errors are undetectable errors while about 10% are MLD errors. However, despite these near MLD individual performances, the resulting two-stage decoding is not as good as expected. This is mostly due to the dominance of undetected errors at stage-1 in conjunction with the suboptimality of this approach (a slight improvement can be obtained by choosing $M_2 > M_1$ since the performance of stage-1 dominates the overall error performance). Hence, the applications of the techniques developed in [14, 15, 16, 78] to iterative decoding should provide interesting error performance improvements.

At a given code rate, as N increases, the weight of the rows of the parity check matrix H also increases for the class of MSMLD codes. This causes the number of redundant rows in H to grow to a very large number if near MLD peformance is required, as is already apparent for the results we present for the (511,256,31) code. Consequently, this approach does not seem to scale up very well with N despite the fact that iterative decoding is used. This is not totally surprising, as in general, the decoding complexity of MLD increases exponentially with N.

4.6 Extension to iterative decoding for the AWGN channel

A very natural extension of these results is to replace the BSC by an AWGN channel. Although as already stated in the introduction, relatively good results for iterative decoding of MSMLD codes have been previously reported for the AWGN channel, all these results fall short of near MLD. The main reason we believe is the large dynamical range taken by the a-posteriori values evaluated after few iterations due to the large correlation propagated by feedback (note that in the BF algorithms, the values at the bit nodes are always the same at the beginning of each iteration). As a result, there is no longer much difference between soft information and hard information with erasure. Indeed, the same conclusions also hold for BP decoding over the BSC, although in that case, no significant degradation can be expected at high enough SNR, as observed in Figure 4.1.

Using a heuristic extension of the decomposition proposed in [28], the a-posteriori information L_{i+1} evaluated at iteration-$(i+1)$ can be represented as the sum of the a-priori information L_0 and a function of approximated extrinsic information values $\tilde{L}_i^e$ derived (and observable) at iteration-i. In graphs with cycles, $\tilde{L}_i^e$ can be viewed as the sum of the true extrinsic information L_i^e and additional correlated values L_i^c,

so that

$$\begin{aligned} L_{i+1} &= L_0 + f(\tilde{L}^e) \\ \text{with} \quad \tilde{L}_i^e &= L_i^e + L_i^c, \end{aligned}$$

Consequently, the influence of correlation can be reduced by modifying the function $f()$ in several ways $g()$ such as scaling ($f \circ g = \alpha f$, $0 < \alpha \leq 1$), off-setting ($f \circ g = \text{sgn}(f) \max\{|f| - \beta, 0\}$), damping ($f \circ g = \alpha f_i + (1-\alpha) f_{i-1}$, $0 < \alpha \leq 1$), or clipping ($f \circ g = \text{sgn}(f) \min\{|f|, C\}$). However, these modifications affect both L_i^e and L_i^c while hypothetically, it would be desirable to reduce L_i^c only. This is indeed a much difficult task as we have direct access to $\tilde{L}_i^e$ only. For example, all best approaches used to iteratively decode the (255,127,21) EG code over the AWGN channel felt short of MLD by about 0.8 dB.

4.7 Conclusion

In this chapter, we have shown that iterative BF algorithms can achieve near MLD of intermediate length MSMLD codes despite the presence of four-cycles in their graph representation. This drawback is overcome by the very large number of redundant low weight check sums. The most straightforward parity check matrix representation of these codes in conjunction with a "call by the need" decoding seems to provide the best compromise between error performance and decoding complexity.

In principle, the three-state BF decoding approach could be applied to any other intermediate length linear code. One "merely" needs to find a sufficient number of redundant low weight codewords in the dual code to construct a useful parity check matrix H. Unfortunately, this does not appear to be an easy task for codes that are not as nicely structured as the families of codes considered in this chapter [7, 77].

Chapter 5 On the capacity of wireless erasure relay networks

In this chapter we determine the capacity of a certain class of wireless erasure relay networks. We define the "cut-capacity" for erasure networks with broadcast at transmission and no interference at reception. We show that with this definition, a max-flow min-cut result holds for the capacity of these networks.

5.1 Introduction

Determining the capacity for a general multi-terminal network has been a long-standing open problem. A general outer bound, based on splitting the network into two subsets in all possible ways, has been proposed in [10]. This outer bound is easily derived: for any division of the network into two parts (also called a "cut"), the amount of information that can be sent from the source side to the destination is less than the "cut-capacity," i.e., the sum-capacity of the links connecting the two subsets assuming full co-operation between all the nodes on the source side and full co-operation between all the nodes on the destination side.

Unfortunately, this bound is not tight even for some simple networks. However, this outer bound can be achieved for a *wireline network* with a single source and a single destination. As the name suggests, transmissions along any given link in a wireline network do not affect transmissions on any other link. In a single source, single destination network, only one node has information to transmit and only one node wants to receive that information. For such a wireline network, the maximum amount of information that can be sent from the source to the destination is the

minimum of the cut-capacities of all cuts separating the source from the destination [2, 35, 40]. In a wireline network, the capacity of any cut is just the the sum of the capacities of the links in that cut, as there is no interference. Therefore, this capacity result is similar to the max-flow min-cut theorem for fluid flows on graphs, and hence results such as this are referred to as max-flow min-cut results.

Note that the analogy between fluid flow and information transfer does not extend too far; for example, a max-flow min-cut style result holds for wireline multicast problems in which a single source wants to transmit the same information to multiple destinations [2, 35, 40]. In the case of fluids, even the problem statement "how much of the same fluid can the source send to multiple destinations?" does not make sense.

The goal of this chapter is to show that a max-flow min-cut result holds for a class of wireless networks. In these networks, we assume the transmission from each node must be a broadcast, i.e., the information sent out by a node along all links connected to it must be the same. This is an accurate model of transmission in a wireless system. However, for reception we assume a model without interference, i.e., messages coming in to a node from different incoming transmitters do not affect each other. Note that this is not true in general for a wireless system; however, this can be realized through some time/frequency/code division multiple access scheme.

Next we assume that each link is modeled as an memoryless erasure channel with no delay. Moreover, the erasures on any links are independent of the erasures on all other links. Finally, we assume that side-information regarding erasure locations on each link is available to the destination. This assumption is based on the premise that the network actually operates on long packets. Moreover, each link is a packet erasure channel, i.e., each packet is either received exactly or dropped completely. Now we can assume that the information about the erasures on every link is sent to the receiver either as a header to other packets or encoded separately into a few packets. If the packets are sufficiently long, the overhead of transmitting the erasure information will

be negligible when compared to the length of the packet. This justifies our channel model of a bit erasure channel on each link with side information to the destination about all erasures on all links. The results for this channel go through for the packet erasure channel as well.

We shall prove a max-flow min-cut type capacity result for these wireless erasure relay networks under the assumptions mentioned above. Although this chapter only concerns itself with the single source, single destination scenario, similar results will go through for some multicast settings also. Many results parallel to those of [35] for this wireless setting can be found in [12]. The rest of the chapter is organized as follows: We introduce the model and the problem statement in Section 5.2 and state our main result, which we prove in Sections 5.3 and 5.4. In Section 5.5, we have a concluding discussion where we state without proof some more results related to this problem.

5.2 Model, definitions and main result

5.2.1 Network model

Let the relay network be modeled by a directed acyclic graph $G = (V, E)$ where $V = \{v_1, \ldots, v_{|V|}\}$ is the set of vertices and $E \subset V \times V$ is the set of directed edges. Each edge $(v_i, v_j) \in E$ represents a memoryless erasure channel from v_i to v_j with erasure probability $\epsilon_{i,j}$ associated with it. All channels are assumed independent and operate without delay. Let $s \in V$ be the source node that wishes to transmit a message to the destination node $d(\neq s) \in V$. Without loss of generality, let $s = v_1$ and $d = v_{|V|}$. All nodes in $V - \{s, d\}$ are relay nodes and are used only to aid communication from s to d. Define $\Gamma_O(v_i) = \{(v_i, v_j)|(v_i, v_j) \in E\}$ (i.e., the set of all edges leaving v_i) and $\Gamma_I(v_i) = \{(v_j, v_i)|(v_j, v_i) \in E\}$ (i.e., the set of all edges coming in to v_i).

We now describe the transmission and reception models. We incorporate broadcast in our network by insisting that vertex v_i transmit the same bits on all outbound edges, i.e., all edges in $\Gamma_O(i)$ carry the same bit sequence X_i. On each edge (v_i, v_j), X_i is distorted since some bits may get erased. The received signal on edge (v_i, v_j) is a string of bits and erasures, denoted by $Y_{i,j}$. For reception, we assume that v_i receives the symbols from each edge in $\Gamma_I(i)$ without interference, i.e., for every v_j such that $(v_j, v_i) \in \Gamma_I(i)$, $Y_{j,i}$ is received at v_i. Define $Y_i = (Y_{j,i}, (v_j, v_i) \in \Gamma_I(i))$.

Clearly, the bit sequence X_s sent out by the source will be a function of the message M that it wants to transmit. At any intermediate vertex i, the outgoing message X_i sent out by the vertex will be a function of the incoming information Y_i. At the destination d, the decoded message M' is a function of the received sequence Y_d and the erasure locations on each link (which we assume are known perfectly to the destination). The choice of all these functions defines the coding scheme.

5.2.2 Capacity

We now state our main result. Define an $s - d$ cut as a partition of the vertex set V into two subsets V_s and $V_d = V - V_s$ such that $s \in V_s$ and $d \in V_d$. Clearly, an $s - d$ cut is determined simply by V_s. For the $s - d$ cut given by V_s, let the cutset $E(V_s)$ be the set of edges defined as the set of edges going from V_s to $V - V_s$, i.e.,

$$E(V_s) \stackrel{\triangle}{=} \{(v_i, v_j) | (v_i, v_j) \in E, v_i \in V_s, v_j \in V_d\}$$

Define $W(V_s)$, the *cut-capacity* or the *value* of an $s - d$ cut given by V_s as

$$W(V_s) \stackrel{\triangle}{=} \sum_{i \in V_s} \left(1 - \prod_{j:(v_i,v_j) \in E(V_s)} \epsilon_{i,j} \right) \tag{5.1}$$

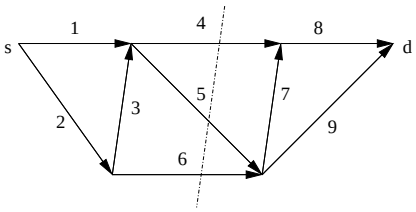


Figure 5.1: Example of cut-capacity: The value of this cut is $(1 - \epsilon_4\epsilon_5) + (1 - \epsilon_6)$, where ϵ_e is the probability of erasure along edge e.

Theorem 5.1 *The capacity of the erasure relay network described above is given by the minimum cut-capacity.*

$$C = \min_{V_s} W(V_s) \tag{5.2}$$

where V_s determines an $s - d$ cut.

5.3 Achievability

5.3.1 Network operation

In this section, we use a random coding argument to show that all rates $R < C$ are achievable. Assume that s has to transmit one message out of 2^{nR} possible messages in n channel uses. Let $\Omega = \{1, 2, \ldots, \lceil 2^{nR} \rceil\}$. Let $\mathcal{X}^n : \Omega \to \{0,1\}^n$ be an encoding function where each $\mathcal{X}^n(i)$ is chosen randomly and independently assuming a uniform distribution on $\{0,1\}^n$. $\mathcal{X}^n(1), \mathcal{X}^n(2), \ldots, \mathcal{X}^n(\lceil 2^{nR} \rceil)$ are called the codewords and together form the codebook $\mathcal{C}$. If s wishes to transmit message k, then the bit sequences X_i transmitted by the intermediate nodes will also be functions of k and hence are denoted $X_i(k)$. Similarly, the received sequences Y_i and $Y_{i,j}$ are also functions of k and hence can be written as $Y_i(k)$ and $Y_{i,j}(k)$.

We now describe how the sequence $X_i(k)$ is chosen. Assume that the channel random variable on edge (v_i, v_j) is $s_{i,j}$. This is a vector in $\{0,1\}^n$ where 1 represents an erasure. Each entry is chosen independently from a Bernoulli distribution

of probability $\epsilon_{i,j}$. The received symbol on edge (v_i, v_j), viz. $Y_{i,j}(k)$, is obtained by erasing those bits of $X_i(k)$ where $s_{i,j}$ is 1. Clearly, $Y_{i,j}(k) \in \{0,1,e\}^n$ where e represents erasures. Since v_i has $|\Gamma_I(v_i)|$ incoming edges, the received symbol is $Y_i(k) \in \{\{0,1,e\}^n\}^{|\Gamma_I(v_i)|}$. For each vertex $v_i \in V - \{s,d\}$, define an encoding function $f_i : \{\{0,1,e\}^n\}^{|\Gamma_I(v_i)|} \rightarrow \{0,1\}^n$. For any $Y_i(k)$ described above, let $f_i(Y_i(k))$ be chosen randomly and independently assuming a uniform distribution on $\{0,1\}^n$. Let $X_i(k) = f_i(Y_i(k))$. (For uniformity of notation, we define $Y_1(k) = k$ so that $X_1(k) = f_1(Y_1(k)) = \mathcal{X}^n(k)$. With this notation, f_1 is chosen in exactly the same manner as the rest of the f_i's. The operation of the network is thus completely determined.

5.3.2 Decoder

We now describe the decoder $\mathcal{D}_1$. Let $y(k) \in \{\{0,1,e\}^n\}^{|\Gamma_I(d)|}$ be what d has received when s wants to transmit message k. We assume that d knows all the functions f_i as well as the codebook $\mathcal{C}$. In addition, the locations of the erasures on each link are available to d as side information, i.e., it knows the values of vectors $s_{i,j}$ exactly. With this, d is in a position to calculate all $X_i(l)$, $Y_i(l)$ and $Y_{i,j}(l)$ for any message $l \in \Omega$ and that instantiation of the network (i.e., those erasure locations). It can then compare the computed $Y_{|V|}(l)$ with what it has received viz. $y(k)$. If there exists a unique $r \in \Omega$ such that $y(k) = Y_{|V|}(r)$, r is declared to be the transmitted message. Otherwise, an error is declared. Note that $y(k) = Y_{|V|}(k)$ will always hold. Therefore, an error is declared only when $\exists r \neq k$ such that $Y_{|V|}(r) = Y_{|V|}(k) = y(k)$. Let the decoding function be denoted by $g_1 : \{\{0,1,e\}^n\}^{|\Gamma_I(d)|} \rightarrow \Omega \cup \{\mathcal{E}\}$ where $\mathcal{E}$ denotes an error.

5.3.3 Probability of error

To analyze the probability of error, we first describe a slightly different decoder $\mathcal{D}_2$. Consider the following typical sets of erasure locations, defined for each vertex v_i that has outgoing edges.

$$A_{\delta_i}^{(n)}(i) \stackrel{\triangle}{=} \{(s_{i,j}, (v_i, v_j) \in \Gamma_O(i))|(s_{i,j}, (v_i, v_j) \in \Gamma_O(i)) \text{ are jointly } \delta_i\text{-typical}\}$$

Decoder $\mathcal{D}_2$ knows all the $s_{i,j}$'s from the network. If any set of $s_{i,j}$'s of the form $(s_{i,j}, (v_i, v_j) \in \Gamma_O(i))$ does not belong to the corresponding typical set $A_{\delta_i}^{(n)}(i)$, an error is declared. Otherwise, a decoding operation identical to that of decoder $\mathcal{D}_1$ is performed. Denote this decoding function by g_2. Clearly, decoding errors made by $\mathcal{D}_1$ are a subset of those made by $\mathcal{D}_2$. Hence the rate achievable by $\mathcal{D}_2$ is also achievable by $\mathcal{D}_1$.

We now analyze the probability of error with $\mathcal{D}_2$. Define the probability that the decoder makes $\mathcal{D}_2$ makes an error when message k is sent:

$$\lambda_k = P(g_2(Y_{|V|}(k)) = \mathcal{E}) \tag{5.3}$$

The average probability of error for a given codebook is now given by

$$P_e^{(n)} = \frac{1}{|\Omega|} \sum_{k=1}^{|\Omega|} \lambda_k \tag{5.4}$$

The probability of error averaged over all codebooks is defined as

$$P(\mathcal{E}) = \sum_{\mathcal{C}} P(\mathcal{C}) P_e^{(n)}(\mathcal{C}) \tag{5.5}$$

Because of the symmetry of the code construction, we have

$$P(\mathcal{E}) = P(\mathcal{E}|k=1) \tag{5.6}$$

i.e., the average probability of error given that the first codeword was sent. Let T be the event that for all i such that $\Gamma_I(i) \neq \emptyset$ we have $(s_{i,j}, (v_i, v_j) \in \Gamma_O(i)) \in A_{\delta_i}^{(n)}(i)$. By making n sufficiently large we have $P(A_{\delta_i}^{(n)}(i)) = (1-\delta_i)$ which can be made arbitrarily close to 1. Now

$$P(T) = \prod_{i:\Gamma_O(i)\neq\emptyset} P(A_{\delta_i}^{(n)}(i)) \tag{5.7}$$

which can also be made arbitrarily close to 1. Let $P(T) = 1-\Delta$. If T does not occur decoder $\mathcal{D}_2$ declares an error. If T occurs, we have a decoding error only if $\exists l \neq 1$ such that $Y_{|V|}(1) = Y_{|V|}(l) = y(1)$. Define the event E_l as follows.

$$E_l = \{Y_{|V|}(1) = Y_{|V|}(l)\} \tag{5.8}$$

i.e., the received messages are identical for messages 1 and l.

$$\begin{aligned}
P(\mathcal{E}) &= P(\mathcal{E}|k=1) \\
&= P(\bigcup_{l=2}^{|\Omega|} E_l) \\
&= P(\bigcup_{l=2}^{|\Omega|} E_l|T^c)P(T^c) + P(\bigcup_{l=2}^{|\Omega|} E_l|T)P(T) \\
&\leq P(T^c) + P(\bigcup_{l=2}^{|\Omega|} E_l|T)P(T) \\
&\leq P(T^c) + P(T)\sum_{l=2}^{|\Omega|} P(E_l|T)
\end{aligned} \tag{5.9}$$

We now consider the event $\{E_l|T\} = \{Y_{|V|}(1) = Y_{|V|}(l)|T\}$. For $l \neq 1$, $Y_1(1) \neq$

$Y_1(l)$ holds trivially.

$$
\begin{aligned}
& P(E_l|T) \\
= \;& P(Y_{|V|}(1) = Y_{|V|}(l), Y_1(1) \neq Y_1(l)|T) \\
= \;& \sum_{V_s \text{ is a cut}} P(Y_i(1) \neq Y_i(l), Y_j(1) = Y_j(l), v_i \in V_s, v_j \in V_d|T) \\
\leq \;& \sum_{V_s \text{ is a cut}} P(Y_i(1) \neq Y_i(l), Y_j(1) = Y_j(l), (v_i, v_j) \in E(V_s)|T) \\
\leq \;& \sum_{V_s \text{ is a cut}} P(Y_i(1) \neq Y_i(l), Y_{i,j}(1) = Y_{i,j}(l), (v_i, v_j) \in E(V_s)|T) \\
= \;& \sum_{V_s \text{ is a cut}} \prod_{i:(v_i,v_j)\in E(V_s)} P(Y_i(1) \neq Y_i(l), Y_{i,j}(1) = Y_{i,j}(l), \\
& \qquad\qquad j \text{ such that } (v_i, v_j) \in E(V_s)|T) \\
= \;& \sum_{V_s \text{ is a cut}} \prod_{i:(v_i,v_j)\in E(V_s)} P(Y_i(1) \neq Y_i(l)|T) \cdot P(Y_{i,j}(1) = Y_{i,j}(l), \\
& \qquad\qquad j \text{ such that } (v_i, v_j) \in E(V_s)|T, Y_i(1) \neq Y_i(l)) \\
\leq \;& \sum_{V_s \text{ is a cut}} \prod_{i:(v_i,v_j)\in E(V_s)} P(Y_{i,j}(1) = Y_{i,j}(l), \\
& \qquad\qquad j \text{ such that } (v_i, v_j) \in E(V_s)|T, Y_i(1) \neq Y_i(l)) \quad (5.10)
\end{aligned}
$$

For a fixed i and V_s, consider the event

$$F_i(V_s) = \{Y_{i,j}(1) = Y_{i,j}(l), \forall j \text{ such that } (v_i, v_j) \in E(V_s)|T, Y_i(1) \neq Y_i(l)\} \quad (5.11)$$

Since $Y_i(1) \neq Y_i(l)$, we know that $X_i(1)$ and $X_i(l)$ will be chosen independently from a uniform distribution on $\{0,1\}^n$. The probability that $X_i(1)$ and $X_i(l)$ differ in at most α places is given by $2^{-(n-\alpha)}$. For a fixed i we will have $Y_{i,j}(1) = Y_{i,j}(l)$ for every j such that $(v_i, v_j) \in E(V_s)$ only if all the bits where $X_i(1)$ and $X_i(l)$ differ get erased on all these edges. For a fixed i, since T occurs, we know that the number of bits that are erased on every j such that $(v_i, v_j) \in E(V_s)$ is close to its expected value,

i.e., of the order of $n \cdot d(V_s, i)$ where

$$d(V_s, i) = \prod_{(v_i, v_j) \in E(V_s)} \epsilon_{i,j}. \tag{5.12}$$

Let this range around $d(V_s, i)$ that is required for typicality according to the definition of set $A_{\delta_i}^{(n)}(i)$ be given by $(d(V_s, i) - d_1(V_s, i), d(V_s, i) + d_2(V_s, i))$. Note that $d_1(V_s, i)$ and $d_2(V_s, i)$ depend on δ_i and $\epsilon_{i,j}$s and are independent of n. Therefore the occurrence of T ensures that at most $n(d(V_s, i) + d_2(V_s, i))$ bits will be erased on all the edges of interest. Also, $d_2(V_s, i)$ and $d_1(V_s, i)$ go to 0 as δ_i goes to 0 provided we let n grow without bound. Note also that $d(V_s, i)$ is exactly the probability of a bit getting erased on all edges $(v_i, v_j) \in E(V_s)$.

Let t_m for $m = 1, 2, \ldots, N$ denote all the distinct values that the vectors $s_{i,j}$ can take such that T occurs. Clearly, the t_ms represent disjoint events and $P(T) = \sum_{m=1}^{N} P(t_m)$. The probability of event $F_i(V_s)$ can now be calculated.

$$\begin{aligned}
P(F_i(V_s)) &= P(Y_{i,j}(1) = Y_{i,j}(l), (v_i, v_j) \in E(V_s) | T, Y_i(1) \neq Y_i(l)) \\
&= \frac{P(Y_{i,j}(1) = Y_{i,j}(l), (v_i, v_j) \in E(V_s), T, Y_i(1) \neq Y_i(l))}{P(T, Y_i(1) \neq Y_i(l))} \\
&= \frac{\sum_{m=1}^{N} P(Y_{i,j}(1) = Y_{i,j}(l), (v_i, v_j) \in E(V_s), t_m, Y_i(1) \neq Y_i(l))}{P(T, Y_i(1) \neq Y_i(l))} \\
&\leq 2^{-n(1-d(V_s,i)-d_2(V_s,i))} \frac{\sum_{m=1}^{N} P(t_m, Y_i(1) \neq Y_i(l))}{P(T, Y_i(1) \neq Y_i(l))} \\
&= 2^{-n(1-d(V_s,i)-d_2(V_s,i))}
\end{aligned} \tag{5.13}$$

Substituting this in the expression in (5.10) we get

$$\begin{aligned} P(E_l|T) &\leq \sum_{V_s \text{ is a cut}} \prod_{i:(v_i,v_j)\in E(V_s)} 2^{-n(1-d(V_s,i)-d_2(V_s,i))} \\ &= \sum_{V_s \text{ is a cut}} 2^{-n\sum_{i:(v_i,v_j)\in E(V_s)}(1-d(V_s,i)-d_2(V_s,i))} \\ &= \sum_{V_s \text{ is a cut}} 2^{-n(W(V_s)-d_2(V_s))} \end{aligned} \tag{5.14}$$

where

$$d_2(V_s) = \sum_{i:(v_i,v_j)\in E(V_s)} d_2(V_s, i). \tag{5.15}$$

and $W(V_s)$ is the value of the cut as defined earlier.

Note that this upper bound on $P(E_l|T)$ is independent of l. Substituting this back in (5.9) we get

$$\begin{aligned} P(\mathcal{E}) &\leq P(T^c) + P(T)(2^{nR}-1) \sum_{V_s \text{ is a cut}} 2^{-n(W(V_s)-d_2(V_s))} \\ &\leq P(T^c) + P(T)2^{nR} \sum_{V_s \text{ is a cut}} 2^{-n(W(V_s)-d_2(V_s))} \\ &= \Delta + (1-\Delta) \sum_{V_s \text{ is a cut}} 2^{n(R-W(V_s)+d_2(V_s))} \end{aligned} \tag{5.16}$$

We know that as n grows, Δ as well as $d_2(V_s)$ can be made arbitrarily close to 0. Therefore, if $R < W(V_s)$ for every cut V_s in the network, we can make $P(\mathcal{E})$ arbitrarily small by letting n grow without bound. Thus the rate

$$R < \min_{V_s} W(V_s)$$

is achievable.

5.4 Converse

Consider an $s-d$ cut given by V_s. Recall that $E(V_s)$ is the set of edges going from V_s to $V-V_s$. We now define quantities $X(V_s)$ and $Y(V_s)$ for this $s-d$ cut.

$$\begin{aligned} X(V_s) &= \{X_i | (v_i, v_j) \in E(V_s)\} \\ Y(V_s) &= \{Y_{i,j} | (v_i, v_j) \in E(V_s)\} \end{aligned} \tag{5.17}$$

It is easy to see that $P(Y_{|V|}|X_1, X(V_s)) = P(Y_{|V|}|X(V_s))$. Therefore $X_1 - X(V_s) - Y_{|V|}$ is a Markov chain. Similarly, $X(V_s) - Y(V_s) - Y_{|V|}$ is a Markov chain. We now have the following sequence of inequalities as a consequence of the data processing inequality [10].

$$\begin{aligned} nR &= I(X_1; Y_{|V|}) \\ &\leq I(X(V_s); Y_{|V|}) \\ &\leq I(X(V_s); Y(V_s)) \end{aligned} \tag{5.18}$$

We now evaluate the quantity $I(X(V_s); Y(V_s)) = H(Y(V_s)) - H(Y(V_s)|X(V_s))$.

$$\begin{aligned} H(Y(V_s)|X(V_s)) &= H((s_{i,j}, (v_i, v_j) \in E(V_s))) \\ &= \sum_{(i,j):(v_i,v_j)\in E(V_s)} H(s_{i,j}) \\ &= n \sum_{(i,j):(v_i,v_j)\in E(V_s)} H(\epsilon_{i,j}) \end{aligned} \tag{5.19}$$

$$\begin{aligned} H(Y(V_s)) &\leq \sum_{i:(v_i,v_j)\in E(V_s)} H(Y_{i,j}, j \text{ such that } (v_i, v_j) \in E(V_s)) \\ &= n \sum_{i:(v_i,v_j)\in E(V_s)} \left(\left(1 - \prod_{j:(v_i,v_j)\in E(V_s)} \epsilon_{i,j} \right) H(p_i) + \sum_{j:(v_i,v_j)\in E(V_s)} H(\epsilon_{i,j}) \right) \end{aligned}$$

where X_i is a string of bits satisfying a Bernoulli distribution with parameter p_i. Clearly $p_i = \frac{1}{2}$ maximizes $H(Y(V_s))$. Therefore we have

$$\begin{aligned} I(X(V_s); Y(V_s)) &\leq n \sum_{i:(v_i,v_j)\in E(V_s)} \left(1 - \prod_{j:(v_i,v_j)\in E(V_s)} \epsilon_{i,j}\right) \\ &= nW(V_s) \end{aligned} \tag{5.20}$$

We have thus shown that

$$R \leq W(V_s)$$

for all cuts V_s, i.e.,

$$R \leq \min_{V_s} W(V_s)$$

5.5 Conclusion

We have generalized some of the capacity results that hold for wireline networks to a certain class of wireless erasure relay networks. The method we employ to reach capacity makes it unnecessary for intermediate nodes to decode and then do channel or network coding separately. Thus, it takes care of the channel coding and network coding aspects in one shot, much like the randomized network coding approach of Ho et al. [31, 32].

In [12] we show that if we restrict ourselves to randomly chosen linear functions at each node, we can still reach capacity. Furthermore, several of the multicast results of [35] also go through for the class of wireless erasure relay networks that we have defined. However, our techniques may not be applicable to other wireless networks. While randomized coding will give us an achievability region, this will not be the capacity region in general. It will be interesting to see if capacity results can be obtained for other types of wireless networks, i.e., networks involving channels other than the erasure channel.

Chapter 6 On the capacity achieving distributions of some vector channels

We study the capacity achieving distributions of some channels with vector inputs, such as the vector Gaussian channel and block fading channels. Using the theory of holomorphic functions of several complex variables, we show that in many cases the capacity achieving random variable will be singular, i.e., it is supported by a set that contains no open set. Roughly stated, this means that a channel with an n-dimensional input has a capacity achieving random variable with dimension $(n-1)$ or lower. This somewhat strange result is a generalization of similar results known in the scalar case [1, 26, 34, 70, 76]. As a corollary, we prove that for a single antenna Rayleigh block fading channel [51] the capacity achieving random variable is the product of a discrete real variable and an isotropically distributed unit vector. We also prove a general discreteness result that holds for any memoryless or block fading channel under certain input constraints. This result suggests, but does not prove, that any non-trivial power constrained memoryless or block fading channel has a capacity achieving random variable that is the product of a discrete real variable and an independent isotropically distributed unit vector.

6.1 Introduction

The problem of finding the capacity of a peak and average-power limited Gaussian channel was first solved by Smith [76], who showed that the capacity of such a channel is achieved by a discrete variable taking on a finite number of values. This surprising result was generalized by Shamai and Bar-David [70] to the quadrature Gaussian

channel with peak and average power constraints. For this case, the capacity achieving random variable takes on a finite number of amplitude values, but has continuous uniform phase independent of the amplitude. Geometrically, the support of the distribution is a finite number of concentric circles centered at the origin. More recently, Abou-Faycal, Trott and Shamai [1] showed that the capacity achieving distribution of a power-constrained discrete-time memoryless Rayleigh fading channel is discrete with a finite number of mass points. Similar results are also known for the Rician fading channel [26] and the non-coherent Gaussian channel [34].

In this chapter, we try to generalize these results to higher dimensional problems. The rest of the chapter is organized as follows. In Section 6.2, we develop the mathematical tools that are necessary for proving our results. We first present the Kuhn-Tucker optimization condition that any capacity achieving random variable must satisfy. We then derive an identity theorem for holomorphic functions of several variables. These two results will then be applied jointly to several channels in later sections.

In Section 6.3, we consider a power constrained vector Gaussian channel composed of n component scalar channels. We show that if the input of this vector channel is constrained to be in a given set $A \subset \mathbb{R}^n$ such that $\mathbb{R}^n \backslash A$ has non-zero Lebesgue measure, then the support of the capacity achieving random variable must not be a superset of any open set in $\mathbb{R}^n$ (or any infinite sequence of points that "resembles" an open set in $\mathbb{R}^n$). Roughly speaking, this means that the "dimension" of the support of the capacity achieving random variable must not exceed $(n-1)$. We then derive several previously known results, including Smith's discreteness result [76] and the well-known waterfilling rule [10], as special cases of this general result. In addition, we extend Shamai et al's result on the quadrature Gaussian channel [70] to higher dimensional spherically symmetric Gaussian channels.

In Section 6.4, we apply the same techniques to the single antenna Rayleigh block

fading channel. This channel is specified by $y_t = h_t x_t + w_t$, where x_t is the input at time t, y_t is the output, w_t is additive complex white Gaussian noise, and h_t is a fade with a complex Gaussian distribution. Moreover, the fade h_t remains constant for T channel uses, before changing to an value drawn independently at random from the complex Gaussian distribution. Marzetta and Hochwald [51] showed that the capacity achieving random variable for this channel is the product of an amplitude (a scalar real number) and an isotropically distributed unit vector which is independent of the amplitude. They also conjectured that the amplitude takes on only a discrete number of values. This conjecture was proved in [1] for the case when $T = 1$, i.e., when the channel is memoryless. In this chapter, we shall prove the conjecture for all $T > 1$.

In Section 6.5, we prove a result that holds for any block fading channel. The result states that if the channel input X was constrained to satisfy $E_X||x||^{2+\epsilon} < a$, then for any $\epsilon > 0$, the capacity achieving random variable X is of the form $X = RU$ where R takes on a finite number of real values and U is an independent isotropically distributed unit vector. This result is true for any distribution that the fade random variable h_t may have and for any fade block length T. We believe that a similar result holds even when $\epsilon = 0$ (power constraint) for any non-trivial fading channel, though we are not able to prove it.

6.2 Mathematical background

6.2.1 Kuhn-Tucker conditions

In this section, we present the required mathematical background and derive preliminary results that we shall use in later sections to prove our main theorems. The derivation of these results closely parallels the corresponding derivations in [1, 26, 34, 70, 76]. The results themselves are purely mathematical in nature, with little or no informa-

tion theoretic meaning. However, when applied to the problem of finding channel capacity, they throw light on the strange structure of capacity achieving distributions.

Before we proceed to the derivation, we must mention that we use the vector Gaussian channel as an example in our derivation. However, the derivation is not specific to the vector Gaussian channel, so we keep our notation as general as possible. Let $X \in \mathbb{R}^n$ be the input random variable, $Y \in \mathbb{R}^n$ be the output random variable and $p(y|x)$ be the channel transition probabilities. In the case of the vector Gaussian channel $x = (x_1, \ldots, x_n)$ is the input and $y = (y_1, \ldots, y_n) = (x_1 + w_1, \ldots, x_n + w_n)$ is the output where the w_i's are independent Gaussian random variables with variance σ_i^2 respectively, i.e.,

$$p(y|x) = \prod_{i=1}^{n} \frac{1}{\sqrt{2\pi\sigma_i^2}} e^{\frac{-(y_i - x_i)^2}{2\sigma_i^2}} \tag{6.1}$$

The marginal output density induced by an input distribution $F(x)$ is

$$p(y; F) = \int p(y|x) dF(x) \tag{6.2}$$

Let $\mathcal{F}$ be the set of distributions F that meet the input constraints (a) X is restricted to be in a given set $A \subset \mathbb{R}^n$ almost surely (b) The overall average power is limited: $\int ||x||^2 dF(x) \leq a$. The capacity of this constrained channel is

$$C = \sup_{F \in \mathcal{F}} \int D(p(y|x) \;||\; p(y; F)) dF(x) \tag{6.3}$$

where $D(a(y)||b(y))$ is the Kullback-Leibler distance given by

$$D(a||b) = \int a(y) \ln \left[\frac{a(y)}{b(y)}\right] dy \tag{6.4}$$

A necessary and sufficient condition for a random variable X^* with distribution F^* to achieve the capacity C is $\exists \beta \geq 0$ such that

$$D(p(y|x) \mid\mid p(y;F^*)) - (C + \beta(||x||^2 - a)) \leq 0 \tag{6.5}$$

for all x, with equality if x is in E_0, the support of X. This is the well known Kuhn-Tucker optimization condition. The derivation of inequality (6.5) is almost identical to the derivation of the corresponding conditions in [1, 70, 76] and hence is omitted for the sake of brevity.

Next, we define the LHS of inequality (6.5) to be $f(x_1, x_2, \ldots, x_n)$. In the Gaussian case,

$$\begin{aligned} f(x_1, \ldots, x_n) = -\int_{-\infty}^{\infty} \cdots \int_{-\infty}^{\infty} \prod_{i=1}^{n} \frac{1}{\sqrt{2\pi\sigma_i^2}} e^{\frac{-(y_i - x_i)^2}{2\sigma_i^2}} \ln p(y;F^*)\, dy_1 \ldots dy_n \\ - \sum_{i=1}^{n} \frac{1}{2} \ln(2\pi e \sigma_i^2) - C - \beta(\sum_{i=1}^{n} x_i^2 - a) \end{aligned} \tag{6.6}$$

Inequality (6.5) states that $f(x) \leq 0$, with equality when x is in E_0, the support of the optimizing random variable. We define an extension of f to $\mathbb{C}^n$ by just replacing the $x_i \in \mathbb{R}$ with $z_i \in \mathbb{C}$. In other words,

$$\begin{aligned} f(z_1, \ldots, z_n) = -\int_{-\infty}^{\infty} \cdots \int_{-\infty}^{\infty} \prod_{i=1}^{n} \frac{1}{\sqrt{2\pi\sigma_i^2}} e^{\frac{-(y_i - z_i)^2}{2\sigma_i^2}} \ln p(y;F^*)\, dy_1 \ldots dy_n \\ - \sum_{i=1}^{n} \frac{1}{2} \ln(2\pi e \sigma_i^2) - C - \beta(\sum_{i=1}^{n} z_i^2 - a) \end{aligned} \tag{6.7}$$

6.2.2 Holomorphic functions

It is easy to see that the $f(z_1, \ldots, z_n)$ defined this way is a holomorphic function of n complex variables [25] with domain $\mathbb{C}^n$. Moreover, E_0 is a subset of the zero set of the holomorphic function f, which we'll denote by $\mathcal{Z}(f)$.

Let us take a look at the topological properties of $\mathcal{Z}(f)$. In the single variable case, $\mathcal{Z}(f)$ not having an accumulation point is a necessary and sufficient condition for it to be the zero set of an analytic function which is not identically zero. Unfortunately, there is no nice characterization of zero sets of holomorphic functions of several variables. The main result in this regard roughly states that $\mathcal{Z}(f)$ is locally a complex manifold of dimension $(n-1)$, except on an exceptional set of lower dimension. Stronger versions of this result are highly technical and are not significant improvements [38]. Therefore, we'll consider a weaker result which permits a reasonable, though not complete, description of what $\mathcal{Z}(f)$ should not be. We provide a simple proof for this result, which is a generalization of the identity theorem used in [1, 26, 34, 70, 76].

Theorem 6.1 *Let f be an holomorphic function defined in an open domain $D \subset \mathbb{C}^n$. If there exists a compact set $S \subset D$ such that $\exists^\infty z_1 \exists^\infty z_2 \ldots \exists^\infty z_n$[1] where $(z_1, \ldots, z_n) \in S$ and $f(z_1, \ldots, z_n) = 0$, then $f(z_1, \ldots, z_n) = 0$ throughout the domain D.*

Corollary 6.2 *If $\exists$ an open set $A \subset \mathbb{R}^n$ such that $f(z) = 0\ \forall z \in A$, then $f(z) = 0\ \forall z \in D$.*

Proof. We prove the theorem for the case $n = 2$. The proof is essentially the same for higher n. Suppose we have a compact set S with infinitely many z_1 at each of which there are infinitely many z_2 such that the points (z_1, z_2) are in S and $f(z_1, z_2) = 0$. Pick any point z_1 and consider the "marginal function" $f_{z_1}(z_2) \stackrel{\triangle}{=} f(z_1, z_2)$. This is a holomorphic function in z_2 [25] and it is equal to zero at infinitely many z_2. Since S is compact, the intersection of S with the corresponding complex plane $(Z_1 = z_1)$ is also compact [75]. Hence the sequence of the infinitely many z_2 has an accumulation point (Bolzano-Weierstrass theorem). We now have a holomorphic function (f_{z_1}) of a

[1] $\exists^\infty$ is the "there exist infinitely many" quantifier. $\exists^\infty x : S(x)$ would mean that there are infinitely many values of x where statement $S(x)$ is true.

single complex variable (z_2) vanishing at a sequence of infinitely many points and the accumulation point. By the identity theorem [69], $f_{z_1}(z_2) = 0$ for all z_2 in the domain of definition of f. Thus we now have infinitely many marginal functions vanishing for all z_2. Now look at the marginal functions $f_{z_2}(z_1)$ for any z_2. It vanishes at infinitely many z_1 in a compact set and hence is identically zero by the same argument. Hence $f_{z_2}(z_1) = 0 \ \forall (z_1, z_2) \in D$ which proves the theorem.

6.3 The vector Gaussian channel

We shall now apply Theorem 6.1 to the vector Gaussian channel to say something about its capacity achieving distribution. Suppose the support set E_0 contains an open set. Then by Theorem 6.1, $f(z) = 0 \ \forall z \in \mathbb{C}^n$, i.e.,

$$\int_{-\infty}^{\infty} \cdots \int_{-\infty}^{\infty} \prod_{i=1}^{n} \frac{1}{\sqrt{2\pi\sigma_i^2}} e^{\frac{-(y_i - z_i)^2}{2\sigma_i^2}} \ln p(y; F^*) \ dy_1 \ldots dy_n$$
$$+ \sum_{i=1}^{n} \frac{1}{2} \ln(2\pi e \sigma_i^2) + C + \beta(\sum_{i=1}^{n} z_i^2 - a) = 0 \ \forall (z_1, \ldots, z_n) \in \mathbb{C}^n \tag{6.8}$$

This condition can be used to solve for the output density $p(y; F^*)$. It turns out that the only possible solution is

$$p(y; F^*) = \prod_{i=1}^{n} \frac{1}{\sqrt{2\pi\tau^2}} e^{-\frac{y_i^2}{2\tau^2}}$$
$$\tau^2 = (a + \sum_{i=1}^{n} \sigma_i^2)/n \tag{6.9}$$

where $1/2\tau^2$ equals the Lagrange multiplier β. It is easy to see that the $p(y; F^*)$ in equation (6.9) is in fact a solution to equation (6.8). Uniqueness of this solution can be established using the fact that the integral in equation (6.8) is an invertible transform, a result which follows from the invertibility of the n-dimensional Laplace transform.

Note that the density $p(y; F^*)$ is that of n i.i.d Gaussian random variables with zero mean and variance τ^2. Thus if the input has support on any open set (or any sequence "resembling" an open set) in $\mathbb{R}^n$, then the output density has to be that of n i.i.d Gaussians. We apply this result to several cases.

Unconstrained: Suppose x can take any value in $\mathbb{R}^n$ and the only constraint is that on the overall average power. The only input random variable that achieves $p(y; F^*)$ is independent Gaussians with variance $(\tau^2 - \sigma_i^2)$. Since each of these variances has to be non-negative, we get $\tau^2 \geq \sigma_{max}^2$, where σ_{max}^2 is the largest of the noise variances. This imposes a constraint on the input power:

$$a \geq \sum_{i=1}^{n} (\sigma_{\max}^2 - \sigma_i^2) \tag{6.10}$$

This is a necessary (and sufficient) condition for E_0 to be the superset of an open set. If it is satisfied, then the capacity achieving input density is the product of n independent Gaussians with the variances given by $P_i = \tau^2 - \sigma_i^2$. This is the well-known waterfilling rule [10]. If equation (6.10) is not satisfied, then our result tells us that E_0 can't contain an open set. This is indeed true because at power levels which do not satisfy equation (6.10), the waterfilling rule allocates zero power to some of the channels. In other words E_0 is a hyper-plane of lower dimension in $\mathbb{R}^n$.

General constraints: Now we impose the additional constraint that the input is restricted to be in A ($E_0 \subset A \neq \mathbb{R}^n$) where $\mathbb{R}^n \backslash A$ has non-zero Lebesgue measure. With this constraint, it is no longer possible to achieve the $p(y; F^*)$ in equation (6.9), since Gaussian input distributions on each of the component channels are not possible. This implies that inequality (6.5) can't be satisfied with equality everywhere. This leads us to state our first result.

Theorem 6.3 *If the input of the vector Gaussian channel is restricted to be in a set A such that $\mathbb{R}^n \backslash A$ has non-zero Lebesgue measure, then the support E_0 of the capacity*

achieving distribution must be a subset of the zero set of a non-zero holomorphic function. In particular, E_0 can't contain any open set in $\mathbb{R}^n$.

Computing the optimizing input distribution and the capacity is however a hard problem for general A. One special case is Smith's result for the scalar Gaussian channel where $n = 1$, $A = [-L, L]$ where we see that E_0 must be a finite set of points.

Rectangular constraints: If the constraints are of the form $|x_i| < A_i$. it is easy to see that the vector channel can be decoupled into n independent scalar channels. This is because for any joint density $p_{X_1,\ldots,X_n}(x_1, \ldots, x_n)$, the product of the marginals $\prod p_{X_i}(x_i)$ is also a valid density and it results in higher output entropy since $h(Y_1, \ldots, Y_n) \leq \sum h(Y_i)$. Thus the capacity achieving density is the product of densities on each of the component channels. By Smith's result, the optimizing random variables on each channel have support only on a finite number of points. Therefore we conclude that E_0 for the vector channel is also a finite set of points.

Spherically symmetric constraints: Suppose the input is constrained to be in the ball $||x|| \leq a$ and the noise is also spherically symmetric, i.e., $\sigma_i^2 = \sigma^2 \ \forall i$. Then by spherical symmetry and the fact that mutual information is concave in the distribution, it follows that E_0 is spherically symmetric. Since the necessary condition for Theorem 6.1 should not be satisfied, E_0 has to be a finite number of spheres of the form $||x|| = r$. This is a generalization of the Shamai-Bar-David result for the quadrature Gaussian channel, where the support set is a finite number of circles.

6.4 The Rayleigh block fading channel

As described in [51], the single antenna Rayleigh block fading channel takes in T complex numbers as the input, multiplies all of them by the same fade h distributed

$\mathcal{CN}(0,1)$ and adds a noise vector, the components of which are drawn i.i.d $\mathcal{CN}(0,1)$.

$$v_t = hu_t + w_t \; t = 1, \ldots, T \tag{6.11}$$

where $(u_1, \ldots, u_T)$ is the input vector, h is the fade and $(w_1, \ldots, w_T)$ is the noise vector and $(v_1, \ldots, v_T)$ is the output vector. Note that the input and the output are in $\mathbb{R}^{2T}$ (or $\mathbb{C}^T$).

Marzetta and Hochwald [51] use the spherical symmetry of the problem to show that the capacity is achieved by a random variable which is the product of a real scalar amplitude R and an independent isotropically distributed unit vector. They go on to conjecture (mainly based on numerical optimization) that R takes on only a discrete number of values. We will now prove this conjecture by contradiction. Let us assume r took on an infinite number of values in any compact interval (which is of course true if the random variable R has support on a continuous interval). Then the support E_0 of the capacity achieving distribution satisfies the necessary condition for Theorem 6.1. Therefore $f(z) = 0 \; \forall z \in \mathbb{C}^{2T}$ where $f(z)$ the LHS of inequality (6.5) suitably extended to the complex domain.

In particular, $f(z) = 0$ when $z = (x, 0, \ldots, 0) = u$ with $x \in \mathbb{R}$. Let us look closely at $f(z)$ in this case. Here information is only transmitted at $t = 1$ on a scalar Rayleigh fading channel. At time instances $t = 2, \ldots, T$, only Gaussian noise is received. Hence

$$p(v|u) = \frac{1}{\pi(x^2+1)} \exp\left[\frac{-|v_1|^2}{x^2+1}\right] \prod_{t=2}^{T} \frac{1}{\pi} \exp\left(-|v_t|^2\right) \tag{6.12}$$

The Kuhn-Tucker condition for this special case states

$$\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\ldots\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} p(v|u) \ln\left[p(v|u)/p(v)\right] \, dv_{1_R} dv_{1_I} \ldots dv_{T_R} dv_{T_I}$$
$$-(C + \beta(x^2 - a)) = 0 \; \forall x \tag{6.13}$$

where v_{t_R}, v_{t_I} are the real and imaginary parts respectively of v_t. Here we have used $||u||^2 = x^2$.

We now need to solve equation (6.13) for an output distribution $p(v)$. The Marzetta-Hochwald result tells us that the optimizing input density $p(u)$ is isotropically distributed. As a consequence the output density $p(v)$ is also isotropically distributed, i.e., the probability is just a function of the amplitude.

$$p(v_1, \ldots, v_T) = g(||v||^2) = g(|v_1|^2 + \cdots + |v_T|^2) \tag{6.14}$$

where g is a strictly positive function of a nonnegative real variable.

We observe that every term in equation (6.13) has a dependence only on $|v_t|^2$. Therefore, we use the change of variables $y_t = |v_t|^2, \theta_t = \tan^{-1}(v_{t_I}/v_{t_R})$ to obtain

$$\begin{aligned} \int_0^\infty \ldots \int_0^\infty p(y|u) \ln \left[p(y|u)/p(y)\right] \, dy_1 \ldots dy_T \\ -(C + \beta(x^2 - a)) = 0 \end{aligned} \tag{6.15}$$

where

$$p(y|u) = \frac{1}{(1+x^2)} \exp\left[\frac{-y_1}{1+x^2}\right] \prod_{t=2}^{T} \exp\left(-y_t\right) \tag{6.16}$$

$$p(y) = \pi g(y_1 + \cdots + y_T) \tag{6.17}$$

Integrating out the $p(y|u) \ln p(y|u)$ term in equation (6.15) yields

$$\begin{aligned} \int_0^\infty \ldots \int_0^\infty p(y|u) \ln p(y) dy_1 \ldots dy_T \\ +C + T + \ln(1+x^2) + \beta(x^2 - a) = 0 \end{aligned} \tag{6.18}$$

Define the marginal function $f(y_1)$ which integrates out the variables $y_2, \ldots, y_T$ in

equation (6.18).

$$f(y_1) = \int_0^\infty \dots \int_0^\infty \exp(-(y_2 + \dots + y_T)) \ln(\pi g(y_1 + \dots + y_T)) dy_2 \dots dy_T \quad (6.19)$$

equation (6.18) now reduces to

$$\int_0^\infty \frac{1}{(1+x^2)} \exp\left[\frac{-y_1}{1+x^2}\right] f(y_1) dy_1 + \alpha + \ln(1+x^2) + \beta x^2 = 0 \quad (6.20)$$

where $\alpha = C + T - \beta a$. This is exactly the same form as the equation for the memoryless Rayleigh fading channel [1]. So, for the sake of brevity, we state without proof the solution of equation (6.20) derived in [1].

$$f(y_1) = \ln K - \ln y_1 - \beta y_1 \quad (6.21)$$

It is easy to see that the above $f(y_1)$ is indeed a solution of equation (6.20). Uniqueness follows from the uniqueness of the Laplace transform. Substituting back in equation (6.19) and multiplying both sides by $\exp(-y_1)$, we get

$$\begin{aligned} \int_0^\infty \dots \int_0^\infty \exp(-(y_1 + y_2 + \dots + y_T)) \ln(\pi g(y_1 + \dots + y_T)) dy_2 \dots dy_T \\ = \exp(-y_1)(\ln K - \ln y_1 - \beta y_1) \end{aligned} \quad (6.22)$$

Define the RHS of equation (6.22) to be $Q(y_1)$ and the integrand in the LHS to be $q(y_1 + y_2 + \dots + y_n)$. In other words,

$$\int_0^\infty \dots \int_0^\infty q(y_1 + y_2 + \dots + y_T) dy_2 \dots dy_T = Q(y_1) \quad (6.23)$$

We now want to solve for q in terms of the known Q. For this we start by replacing

$y_2 + y_1$ by y_2. This gives us

$$\int_{\infty}^{y_1} \int_{0}^{\infty} \dots \int_{0}^{\infty} q(y_2 + \dots + y_T) dy_2 \dots dy_T = Q(y_1) \tag{6.24}$$

Using the fundamental theorem of calculus, we get

$$\int_{0}^{\infty} \dots \int_{0}^{\infty} q(y_1 + y_3 + \dots + y_T) dy_3 \dots dy_T = (-1) \frac{dQ(y_1)}{dy_1} \tag{6.25}$$

Repeating this procedure a further $T - 2$ times gives us

$$q(y_1) = (-1)^{T-1} \frac{d^{T-1} Q(y_1)}{dy_1^{T-1}} \tag{6.26}$$

Substituting the $q(y_1)$ and $Q(y_1)$ defined in equation (6.22), we get

$$q(y_1) = (-1)^{T-1} \frac{d^{T-1}}{dy_1^{T-1}} [\ln K - \beta y_1 - \ln y_1] = \exp(-y_1) \left[\frac{(T-1)! + o(1)}{y_1^{T-1}} \right] \tag{6.27}$$

with $o(1)$ being used to denote a term that approaches zero as $y \to 0$. This means the output distribution $p(y_1, \dots, y_T)$ has to be of the form

$$p(y_1, \dots, y_T) = \exp \left[\frac{(T-1)! + o(1)}{(y_1 + \dots + y_T)^{T-1}} \right] \tag{6.28}$$

This cannot be a probability density, because

$$\int_{0}^{\infty} \dots \int_{0}^{\infty} p(y_1, \dots, y_T) dy_1 \dots dy_T = \infty \tag{6.29}$$

no matter what the $o(1)$ term is. Thus our assumption that the amplitude R has support on an infinite number of points on any compact interval leads to an invalid output distribution. This gives us our second result:

Theorem 6.4 *The capacity achieving random variable for a Rayleigh block fading*

channel is of the form $X = RU$ where U is an isotropically distributed unit vector and R takes on a discrete number of real values such that in any compact interval, it takes on only a finite number of values.

It may be possible to prove that R takes on only a finite number of values (which is the case when $T = 1$), but that is beyond the scope of this chapter.

6.5 General block fading channels

In this section, we will first consider a general memoryless fading channel and try to describe the structure of the capacity achieving distribution. The fading channel is of the form

$$Y = HX + W \tag{6.30}$$

where $Y, H, X, W \in \mathbb{C}$. Here X is the input variable, Y is the output variable, W is the additive noise distributed $\mathcal{CN}(0, 1)$ and H the fade distributed $p_H(h)$. The only constraint imposed on the fade is that the average fade power $E_H(h^2)$ be finite. On the input variable X, we impose a "higher moment constraint" by forcing X to satisfy

$$E_X(|x|^{2+\epsilon}) \leq a \tag{6.31}$$

The case $\epsilon = 0$ corresponds to the usual power condition. We will prove that for any $\epsilon > 0$, there is a capacity achieving distribution of the form $X = Re^{j\Phi}$ where R is a real variable taking on a finite number of values and Φ is an independent variable distributed uniformly on $[0, 2\pi]$. Such distributions are called discrete amplitude uniform independent phase (DAUIP) distributions.

The UIP part easily follows from the circular symmetry of the problem. Suppose $X_0 = Re^{j\Phi_0}$ is a capacity achieving random variable resulting in mutual information I_0. Consider a new random variable $X_1 = X_0 e^{j\Theta}$ where Θ is uniformly distributed.

Conditioned on Θ, X_1 results in the same mutual information, i.e., $I(X_1; Y|\Theta) = I_0$. Now by the concavity of mutual information in the input distribution, it follows that X_1 achieves mutual information $I_1 \geq I_0$. Clearly, X_1 has UIP since $\Theta + \Phi$ is independent of R and uniformly distributed.

The DA part is harder to prove and we have to use the method developed in the previous sections. Firstly, we will state the Kuhn-Tucker condition.

$$\exists \beta \geq 0 \ : \ D(p(y|x) \ || \ p(y; F^*)) - (C + \beta(|x|^{2+\epsilon} - a)) \leq 0 \tag{6.32}$$

with equality when x is the support E_0 of the capacity achieving distribution.

In the next step, we see that if R takes on an infinite number of values in any compact interval, then the necessary conditions for Theorem 6.1 are satisfied. (We have to be slightly careful here since the function we are dealing with is not analytic at $z_1 = 0$ or $z_2 = 0$, but that is not a major problem). Therefore inequality (6.32) holds with equality everywhere. In particular we can pick some sequence x_i with $|x_i| \to \infty$ where

$$D(p(y|x_i) \ || \ p(y; F^*)) = C + \beta(|x_i|^{2+\epsilon} - a) \tag{6.33}$$

Note that this holds even in the case when R has support on an infinite sequence of points with only a finite number of points in any compact interval. This means that there is a sequence $r_i \to \infty$ in the support of R and hence equation (6.33) holds for some sequence x_i with $|x_i| \to \infty$.

In the last step, we upper-bound the LHS of equation (6.33) by a function that grows as $|x_i|^2$ thereby proving that equation (6.33) can't hold for $|x_i| \to \infty$. In the

derivation, we use the notation $g(y) = (1/\pi)e^{-|y|^2}$ for convenience.

$$\begin{aligned}
D(p(y|x)||p(y)) &= D(E_H p(y|x,h)||E_X E_H p(y|x,h)) \\
&= D(E_H p(y|x,h)||E_H E_X p(y|x,h)) && (6.34) \\
&\leq E_H D(p(y|x,h)||E_X p(y|x,h)) && (6.35) \\
&= E_H \int g(y-xh) \ln \left[\frac{g(y-xh)}{E_X g(y-xh)} \right] dy \\
&= E_H \left[-\ln \pi e + \int g(y-xh)(-\ln E_X g(y-xh)) dy \right] \\
&\leq E_H \left[-\ln \pi e + \int g(y-xh) E_X(-\ln g(y-xh)) dy \right] && (6.36) \\
&= E_H \left[-1 - \ln \pi + \int g(y-xh) E_X(|y-xh|^2 + \ln \pi) dy \right] \\
&= E_H \left[-1 + \int g(y-xh)(|y|^2 + |h|^2 E_X|x|^2) dy \right] && (6.37) \\
&= E_H \left[-1 + |h|^2 E_X|x|^2 + \int g(y)|y+xh|^2 dy \right] \\
&= E_H \left[|h|^2 E_X(|x|^2) + |h|^2|x|^2 \right] \\
&= (E_H|h|^2)(|x|^2 + E_X|x|^2) && (6.38)
\end{aligned}$$

E_H and E_X can be interchanged in equation (6.34) because X and H are independent. Inequality (6.35) comes from the convexity of $D(.||.)$ in its inputs [10], while inequality (6.36) comes from the convexity of $-\ln(.)$. X having uniform phase implies $E_X(x) = 0$, a fact used in equation (6.37). $E_H|h|^2$ is finite by assumption and $E_X|x|^2$ is also finite because $E_X|x|^{2+\epsilon}$ is.

The upper bound (6.38) implies that inequality (6.33) can't be satisfied as $|x| \to \infty$ for any $\epsilon > 0$ for any positive β. This leads to our third main result.

Theorem 6.5 *Consider any memoryless fading channel with finite fade second moment ($E_H|h|^2 < \infty$). Let there be an input higher moment constraint of the form $E_X|x|^{2+\epsilon} \leq a$. Then $\forall \epsilon > 0$ the capacity achieving random variable is of the form*

$X = Re^{j\Phi}$ where Φ is uniformly distributed on $[0, 2\pi]$ and R takes on a finite number of real values.

Note that $\beta = 0$ corresponds to the unphysical situation where the capacity is independent of the input higher moment constraint. This means that the capacity at any $E_X|x|^{2+\epsilon} > a$ equals the capacity at $E_X|x|^{2+\epsilon} = a$. This can be shown to be impossible for all non-trivial fading channels by constructing a sequence of distributions with increasing mutual information. Indeed the construction given in [1] suffices.

The fact that the capacity achieving R has finite support for all $\epsilon > 0$ doesn't prove that is the case when $\epsilon = 0$. Indeed, one counterexample is the Gaussian channel (constant fade) where the capacity achieving X has support on the entire complex plane. However we believe that the Gaussian case is the only fading channel where such a continuous-discrete transition occurs at $\epsilon = 0$. In other words, we believe that all non-constant fading channels have DAUIP capacity achieving distributions. The results on the Rayleigh fading [1], Rician fading [26] and non-coherent Gaussian [34] channels support this conjecture.

If this conjecture were true, then finding the capacity achieving distribution would require numerical optimization of a discrete set of parameters, as opposed to a continuous set of parameters. This should lead to faster computation and better precision (see [1, 70, 76] and [68] for computation details). Even if the conjecture were not true, there may be something to be gained in the way of complexity by computing the capacities with a higher moment constraint instead of the power constraint. Because $E_X|x|^{2+\epsilon}$ is continuous in ϵ, the capacity computed this way by using finite dimensional optimization can be reasonably close to the actual capacity if ϵ is sufficiently small.

We can extend the results of Theorem 6.5 to all block fading channels. These are channels where the fade remains constant for T channel uses, then changes to

an independent value drawn according to some probability density $p_H(h)$. We can extend Theorem 6.5 to such channels easily by using steps similar to (6.34)-(6.38).

Theorem 6.6 *Consider any block fading channel with finite fade second moment ($E_H|h|^2 < \infty$). Let there be an input constraint of the form $E_X||x||^{2+\epsilon} \leq a$. Then $\forall \epsilon > 0$ the capacity achieving random variable is of the form $X = RU$ where U is isotropically distributed and R takes on a finite number of real values.*

Again we conjecture that a similar result holds even when $\epsilon = 0$, except in the case when the fade is constant (spherically symmetric vector Gaussian channel). Theorem 6.4 about the Raleigh block fading channel partly supports this conjecture.

6.6 Conclusion

We have studied several vector channels and found that the capacity achieving distribution is singular in almost all of these channels. In other words, a channel with an n-dimensional vector input generally has a capacity achieving distribution with lower dimension. A classic example is the power constrained vector Gaussian channel on which the waterfilling rule allots zero power to some of the component channels. This means that the capacity achieving random variable can be specified by $(n-1)$ or fewer parameters. Other previously known examples include Smith's result for a peak power limited scalar (1-dimensional) Gaussian channel [76], which states the capacity achieving distribution has support on a finite number of points, a 0-dimensional set. Similarly the peak-power limited quadrature Gaussian channel [70], with a 2-dimensional input, has a capacity achieving distribution supported by a finite number of circles, a set with dimension 1. In this chapter, we showed that these results are instances of a general singularity result (Theorem 6.3) that holds for almost all vector Gaussian channels.

We also showed that similar singularity results hold for block fading channels as well. This singular nature, when combined with channel specific information, often reduces the computation of the capacity achieving distribution to the optimization of a finite (or at least discrete) number of parameters. As an example, we used our singularity results and spherical symmetry to show that the capacity achieving random variable for the Rayleigh block fading channel [51] is the product of a discrete real amplitude and an independent isotropically distributed unit vector.

We hope that the techniques we developed will be useful on channels other than the ones described in this chapter. In particular, we hope that they can be used to prove that the capacity achieving random variable on any block fading channel is the product of a discrete real amplitude and an independent isotropically distributed unit vector. This conjecture, if true, would encompass many previously known results [1, 26, 34] on this subject.

Chapter 7 Conclusion

In the preceding chapters, we have studied several important problems related to the design of efficient error correction codes for wireless networks. In this chapter, we present a brief summary of our results and some open problems.

In Chapter 2, we studied two classes of rateless codes viz., LT codes and raptor codes on channels such as the BSC and the AWGNC. We found that raptor codes outperform LT codes and have good performance on a wide variety of channels. However, there is room for minor improvements. For example, we could use raptor codes optimized for the AWGNC instead of those optimized for the BEC. Moreover, we did not design an explicit data transfer protocol that uses raptor codes. Designing such a protocol and analyzing its performance on important wireless channels would be a logical direction for future research.

Our results in Chapter 3 are similar in nature to those in Chapter 2. While error correcting codes designed for the single user BEC work fairly well on MACs, it is possible to have explicit code constructions that offer marginal improvements. We provided such a construction in the case of the BAC. Like in the case of rateless codes, we have assumed synchronization between all nodes. Designing a protocol that achieves this is a non-trivial task.

In Chapter 4, we studied iterative decoding algorithms for a class of non-LDPC codes called Euclidean geometry (EG) codes. We found that the use of highly redundant parity check matrices makes it possible to obtain near-optimal performance for some EG codes. It would be interesting to see if similar techniques can be applied to other classes of codes, such as Reed-Solomon codes.

In Chapter 5, we designed a distributed network code that achieves capacity for

a class of wireless erasure multicast networks. However, our results are not easily extended to non-erasure networks. One goal of future research would be to find the capacity of other wireless networks.

In Chapter 6, we conjectured that all non-trivial block fading channels have singular capacity achieving distributions and proved several results that support this conjecture. A proof of this conjecture would be desirable.

The list of open problems given above is by no means exhaustive. There are a large number of other important problems in the design of efficient codes for wireless networks. Solutions to these problems are likely to have significant impact on future communications systems. We hope that this thesis will be prove to be useful in solving a few of these open problems.

Bibliography

[1] I. C. Abou-Faycal, M. D. Trott and S. Shamai, "The capacity of discrete-time memoryless Rayleigh-fading channels," *IEEE Trans. Inform. Theory,* vol. 47, pp. 1290-1301, May 2001.

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

[3] S. Aji and R. J. McEliece, "The generalized distributive law," *IEEE Trans. Inform. Theory*, vol. 32, no. 1, pp. 325-343, Mar 2000.

[4] A. Amraoui, S. Dusad, R. Urbanke, "Achieving general points in the 2-user Gaussian MAC without time-sharing or rate-splitting by means of iterative coding," *Proc. Intl. Symp. Inform. Theory,* Jul 2002.

[5] C. Berrou, A. Glavieux and P. Thitimajshima, "Near Shannon limit error-correcting coding and decoding," *Proc. Intl. Conf. Comm.,* pp. 1064-1070, May 1993.

[6] I. F. Blake and R. C. Mullin, *The mathematical theory of coding,* Academic Press, 1975.

[7] A. Canteaut and F. Chabaud, "A new algorithm for finding minimum-weight words in a linear code: Application to McEliece's cryptosystem and to narrow-sense BCH codes of length 511," *IEEE Trans. Inform. Theory*, vol. 44, pp. 367-378, Jan 1998.

[8] S.-Y. Chung, G. Forney, T. Richardson and R. Urbanke, "On the Design of low-density parity check codes within 0.0045 db of the Shannon Limit," *IEEE Comm. Letters,* vol. 5, pp. 58-60, Feb 2001.

[9] S.-Y. Chung, R. Urbanke and T. J. Richardson, "Analysis of sum-product decoding of low-density parity-check codes using a Gaussian approximation," *IEEE Trans. Inform. Theory*, vol. 47, pp. 657-670, Feb. 2001.

[10] T. M. Cover and J. A. Thomas, *Elements of information theory,* Wiley, 1991.

[11] T. M. Cover, R. J. McEliece and E. C. Posner, "Asynchronous multiple-access channel capacity," *IEEE Trans. Inform. Theory* vol. 27, pp. 409-413, Jul 1981.

[12] A. F. Dana, R. Gowaikar, R. Palanki, B. Hassibi and M. Effros, "On the capacity of wireless erasure networks," *submitted to IEEE Trans. Inform. Theory.*

[13] C. Di, D. Proietti, E. Teletar, T. Richardson and R. Urbanke, "Finite-length analysis of low density parity-check codes on the binary erasure channel," *IEEE Trans. Inform. Theory,* vol. 48, pp. 1570-1579, Jun 2002.

[14] I. Dumer and K. Shabunov, "Recursive decoding of Reed-Muller codes," *Proc. Intl. Symp. Inform. Theory,* Jun 2000.

[15] I. Dumer and K. Shabunov, "Near-optimum decoding for subcodes of Reed-Muller Codes," *Proc. Intl. Symp. Inform. Theory,* Jun 2001.

[16] I. Dumer and K. Shabunov, "Recursive and permutation decoding for Reed-Muller codes," *Proc. Intl. Symp. Inform. Theory,* Jun 2002.

[17] P. Elias, "Coding for noisy channels," *IRE Conv. Record*, Part 4, pp. 37-37, Mar 1955.

[18] Flarion Technologies, "Vector LDPC coding solution," *Technical data sheet.* Available online at `http://www.flarion.com/products/vector.asp`

[19] G. D. Forney Jr., "Coset codes II: Binary lattices and related codes," *IEEE Trans. Inform. Theory,* vol. 34, pp. 1152-1187, Sep 1988.

[20] M. Fossorier and S. Lin, "Generalized coset decoding," *IEEE Trans. Comm.*, vol. 45, pp. 393-395, Apr 1997.

[21] M. Fossorier, R. Palanki and J. S. Yedidia, "Iterative decoding of multi-step majority logic decodable codes," *Proc. 3rd Intl. Symp. on Turbo Codes and Related Topics,* Sep 2003.

[22] R. G. Gallager, *Low-density parity-check codes,* M.I.T. Press, 1963.

[23] R. G. Gallager, "Low-density parity-check codes," *IRE Trans. Inform. Theory*, vol. 8, pp. 21-28, Jan 1968.

[24] R. Gowaikar, A. F .Dana, R. Palanki, B. Hassibi and M. Effros, "On the capacity of wireless erasure relay networks," *accepted for presentation at ISIT 2004.*

[25] R. C. Gunning and H. Rossi, *Analytic functions of several complex variables,* Prentice-Hall, 1965.

[26] M. C. Gursoy, H. V. Poor and S. Verdu, "On the capacity-achieving distribution of the noncoherent Rician fading channel," *Canadian Workshop on Inform. Theory,* May 2003.

[27] J. Hagenauer, "Rate-compatible punctured convolutional codes and their applications," *IEEE Trans. Comm.,* vol. 36, pp. 389-400, Apr 1988.

[28] J. Hagenauer, E. Offer and L. Papke, "Iterative decoding of block and convolutional codes," *IEEE Trans. Inform. Theory*, vol. 42, pp. 429-445, Mar 1997.

[29] J. Harel, R. J. McEliece and R. Palanki, "Poset belief propagation - experimental results," *Proc. Intl. Symp. Inform. Theory,* Jun 2003.

[30] F. Hemmati, "Closest coset decoding of $|u|u+v|$ codes," *IEEE Jour. Select. Areas Commun.*, vol. 7, pp. 982-988, Aug 1989.

[31] T. Ho, R. Koetter, M. Medard, D. R. Karger and M. Effros, "The benefits of coding over routing in a randomized Setting," *Proc. Intl. Symp. Inform. Theory,* Jul 2003.

[32] T. Ho, M. Medard, J. Shi, M. Effros and D. R. Karger, "On randomized network coding," *Proc. 41st Allerton Annual Conference on Communication, Control, and Computing,* Oct 2003.

[33] H. Jin, A. Khandekar and R. J. McEliece, "Irregular repeat-accumulate codes," *Proc. 2nd Intl. Symp. on Turbo Codes and related topics,* Sep 2000.

[34] M. Katz and S. Shamai, "On the capacity-achieving distribution of the discrete-time non-coherent additive white Gaussian noise channel," *Proc. Intl. Symp. Inform. Theory*, Jul 2002.

[35] R. Koetter and M. Medard, "An algebraic approach to network coding," *IEEE/ACM Trans. Networking*, Feb 2003.

[36] Y. Kou, S. Lin and M. Fossorier, "Low density parity check codes based on finite geometries: A rediscovery and new results," *IEEE Trans. Inform. Theory*, vol. 47, pp. 2711-2736, Nov 2001.

[37] Y. Kou, J. Xu, H. Tang, S. Lin and K. Abdel-Ghaffar, "On circulant low density parity check codes," *Proc. Intl. Symp. Inform. Theory*, Jun 2002.

[38] S. G. Krantz, *Function theory of several complex variables,* Wiley, 1982.

[39] C. F. Leanderson, G. Caire and O. Edfors, "On the performance of incremental redundancy schemes with turbo codes," *Proc. Radiovetenskap och Kommunikation 2002,* pp. 57-61, Jun 2002.

[40] S. -Y. R. Li, R. W. Yeung and N. Cai, "Linear network coding," *IEEE Trans. Inform. Theory*, vol. 49, no. 2, pp. 371-381, Feb 2003.

[41] S. Lin and D. J. Costello, Jr., *Error control coding: fundamentals and applications,* Prentice-Hall, 1983.

[42] S. Lin, H. Tang and Y. Kou, "On a class of finite geometry low density parity check codes," *Proc. Intl. Symp. Inform. Theory*, Jun 2001.

[43] M. Luby, "LT- codes," *Proc. 43rd Annual IEEE Symp. on the Foundations of Comp. Science*, pp. 271-280, Nov 2002.

[44] M. Luby, M. Mitzenmacher, A. Shokrollahi and D. Spielman, "Improved low-density parity check codes using irregular graphs," *IEEE Trans. Inform. Theory,* vol. 47, pp. 585-598, Feb 2001.

[45] M. Luby, M. Mitzenmacher, A. Shokrollahi, D. Spielman and V. Stemann, "Practical loss-resilient codes," *Proc. 29th ACM Symp. on the Theory of Computing,* pp. 150-159, May 1997.

[46] R. Lucas, M. Bossert and M. Breitbach, "On iterative soft-decision decoding of linear binary block codes and product codes," *IEEE Jour. Select. Areas Commun.*, vol. 16, pp. 276-298, Feb 1998.

[47] R. Lucas, M. Fossorier, Y. Kou and S. Lin, "Iterative decoding of one-step majority logic decodable codes based on belief propagation," *IEEE Trans. Commun.*, vol. 48, pp. 931-937, Jun 2000.

[48] D. J. C. MacKay, "Good error-correcting codes based on very sparse matrices," *IEEE Trans. Inform. Theory*, vol. 45, pp. 399-431, Mar 1999.

[49] D. J. C. MacKay and R. M. Neal, "Near Shannon limit performance of low density parity check codes," *Electron. Let.*, vol. 32, pp. 1645-1646, Aug 1996.

[50] D. M. Mandelbaum, "An adaptive-feedback coding scheme using incremental redundancy," *IEEE Trans. Inform. Theory*, vol. 20, pp. 388-389, May 1974.

[51] T. L. Marzetta, B. M. Hochwald, "Capacity of a mobile multiple-antenna communication link in Rayleigh flat fading," *IEEE Trans. Inform. Theory,* vol. 45, pp. 139-157, Jan 1999.

[52] P. Maymounkov, "Online codes," NYU Technical Report TR2003-883, Nov 2002. Available online at `http://www.scs.cs.nyu.edu/~petar`

[53] R. J. McEliece, *The theory of information and coding,* Cambridge University Press, 2002.

[54] R. J. McEliece, D. J. C. MacKay and J.-F. Cheng, "Turbo decoding as an instance of Pearl's 'belief propagation' algorithm," *IEEE J. Selected Areas in Comm.,* vol. 16, no. 2, pp. 140-152, Feb 1998.

[55] F. J. MacWilliams and N. J. A. Sloane, *The theory of error-correcting codes,* North-Holland Mathematical Library, 1977.

[56] D. E. Muller, "Application of Boolean algebra to switching circuit design and to error detection," *IRE Trans. Electron. Comput.*, vol. 3, pp. 6-12, Jan 1954.

[57] P. Oswald and A. Shokrollahi, "Capacity-achieving sequences for the erasure channel," *IEEE Trans. Inform. Theory,* vol. 48, pp. 3017-3028, Dec 2002.

[58] R. Palanki, "On the capacity achieving distributions of some fading channels," *Proc. 40th annual Allerton conference*, Oct 2002.

[59] R. Palanki, M. Fossorier and J. S. Yedidia, "Iterative decoding of multi-step majority logic decodable codes," *submitted to IEEE Trans. Comm.*

[60] R. Palanki, A. Khandekar and R. J. McEliece, "Graph-based codes for synchronous multiple access channels," *Proc. 39th Allerton Conf. on Communication, Control and Computing,* Oct 2001.

[61] R. Palanki and J. S. Yedidia, "Rateless codes on noisy channels," *Proc. Conf. Inform. Sciences and Systems*, Mar 2004.

[62] R. Palanki and J. S. Yedidia, "Rateless codes on noisy channels," *accepted for presention at ISIT 2004.*

[63] J. Pearl, *Probabilistic reasoning in intelligent systems,* Morgan Kauffman, 1988.

[64] W. W. Peterson and E. J Weldon Jr., *Error-correcting codes,* M.I.T. Press, 1972.

[65] I. S. Reed, "A class of multiple-error-correcting codes and the decoding scheme," *IRE Trans. Inform. Theory*, vol. 4, pp. 38-49, Sep 1954.

[66] T. Richardson and R. Urbanke, "The capacity of low-density parity check codes under message-passing decoding," *IEEE Trans. Inform. Theory*, vol. 47, pp. 599-618, Feb 2001.

[67] B. Rimoldi, "Generalized time sharing: A low-complexity capacity-achieving multiple-access technique," *IEEE Trans. Inform. Theory,* vol.47, pp. 2432-2442, Sep 2001.

[68] K. Rose, "A mapping approach to rate-distortion computation and analysis," *IEEE Trans. Inform. Theory,* vol. 40, pp. 1939-1952, Nov 1994.

[69] W. Rudin, *Real and complex analysis,* McGraw-Hill, 1987.

[70] S. Shamai and I. Bar-David, "The capacity of average and peak-power limited quadrature Gaussian channels," *IEEE Trans. Inform. Theory,* vol. 41, pp. 1060-1071, Jul 1995.

[71] C. E. Shannon, "The mathematical theory of communication," *Bell System Technical Journal*, vol. 27, pp. 379-423 and pp. 623-656, Jul and Oct 1948.

[72] A. Shokrollahi, "New sequences of linear time erasure codes approaching channel capacity," *Proc 1999 ISITA,* pp. 65-76, Nov 1999.

[73] A. Shokrollahi, "Raptor codes," *accepted for presentation at ISIT 2004.* Available online at `http://www.inference.phy.cam.ac.uk/mackay/DFountain.html`

[74] A. Shokrollahi, S. Lassen and M. Luby, "Multi-stage code generator and decoder for communication systems," U.S. Patent application number 20030058958, Dec 2001.

[75] G. F. Simmons, *Introduction to topology and modern analysis,* McGraw-Hill, 1963.

[76] J. G. Smith, "The information capacity of amplitude and variance-constrained scalar Gaussian channels," *Inform. Contr.,* vol. 18, pp. 203-219, Apr 1971.

[77] J. Stern, "A method for finding codewords of small weight," *Lecture Notes in Computer Science*, vol. 388, pp 106-113, Springer Verlag, 1989.

[78] D. Stojanovic, M. Fossorier and S. Lin, "Iterative multi-stage maximum likelihood decoding of multi-level concatenated codes," *Proc. Workshop on Coding and Cryptography*, Jan 1999.

[79] H. Tang, J. Xu, Y. Kou, S. Lin and K. Abdel-Ghaffar, "On algebraic construction of Gallager low density parity check codes," *Proc. Intl. Symp. Inform. Theory*, Jun 2002.

[80] R. M. Tanner, "A recursive approach to low complexity codes," *IEEE Trans. Inform. Theory*, vol. 27, pp. 533-547, Sep 1981.

[81] A. J. Viterbi, "Error bounds for convolutional codes and an asymptotically optimal decoding algrotihm," *IEEE Trans. Inform. Theory*, vol. IT-13, pp. 260-269, Apr 1967.

[82] S. B. Wicker, *Error control systems for digital communication and storage,* Prentice-Hall, 1995.

[83] K. Yamaguchi, H. Iizuka, E. Nomura and H. Imai, "Variable threshold soft decision decoding," *IEICE Trans. Elect. and Comm.*, vol. 72, pp. 65-74, Sep 1989.

[84] J. S. Yedidia, J. Chen and M. Fossorier, "Generating code representations suitable for belief propagation decoding," *Proc. 40th Annual Allerton Conf.,* Oct 2002.

[85] J. S. Yedidia, W. T. Freeman and Y. Weiss, "Constructing free energy approximations and generalized belief propagation algorithms," *MERL technical report,* Aug 2002. Available online at `http://www.merl.com/papers/TR2004-040/`