The Pennsylvania State University
The Graduate School

LOW-DENSITY PARITY-CHECK CODES: ASYMPTOTIC BEHAVIOR AND ZETA FUNCTIONS

A Dissertation in
Mathematics
by
Min Lu

Submitted in Partial Fulfillment
of the Requirements
for the Degree of

Doctor of Philosophy

August 2009

The dissertation of Min Lu was reviewed and approved* by the following:

Wen-Ching Winnie Li
Professor of Mathematics
Dissertation Advisor, Chair of Committee

John J. Metzner
Professor of Computer Science and Engineering

David J. Miller
Professor of Electrical Engineering

Gary Mullen
Head of the Department of Mathematics

*Signatures are on file in the Graduate School.

Abstract

A LDPC code is binary linear code equipped with a sparse parity-check matrix. A bipartite graph, called Tanner graph, can be associated to a parity-check matrix. Then message passing iterative decoding (MPID) algorithms can be applied on the Tanner graph to do fast efficient decoding. Richardson and Urbanke proposed a scheme, called density evolution, to analyze the performance of MPID algorithms. Using density evolution we shall report new approaches to designing asymptotically good LDPC code ensembles over binary erasure channel (BEC). We then answer an open question raised by Shokrollahi and Oswald during their study on capacity-achieving sequences over BEC. Next, in order to understand MPID algorithms better we bring up the discussion of pseudo-codewords, which are regarded as the cause of decoding errors. We shall present two rational functions whose monomials in the Taylor expansion enumerate all the pseudo-codewords of a general binary parity-check code.

Table of Contents

List of Figures

List of Tables

Acknowledgments

I am lucky to work under the supervision of Wen-Ching Winnie Li. As the PhD advisor Winnie has been a great teacher both in the research related topics and in life. Her encouragement in open-mindedness and in independent thinking is undoubtably invaluable in these years. I have learned a lot from her skills in communicating with different people too. I really hope that we will have chances to work together in the future.

As a teaching assistant for 7 years, I am grateful to James Sellers, the director of undergraduate studies, and the staff members, who have made teaching in Penn State an interesting and enjoyable experience.

Many friends have made these years fun and I have learned more or less from them. Ziming Zhuang is a nice partner in photography and road travelling. The experience with him in the national parks across the countries has been precious memory. A special thanks goes to my friends Guangri Xue and Ji Xuan, whose encouragement and support greatly helped me to eventually step out of the darkest time in 2004. Without them I might have dropped the PhD program and there wouldn't have been this thesis. Thanks, Guangri and Ji.

My parents have always been important to me. I want to thank them for the support in every important decision I have made.

Of course I must thank the most important person, my fiancee Yiying Hong, for the love and all the wonderful things we have been through together. Alaska, Grand Canyon, Yosemite, weekend home poker games, there are, and will be, much more on the list. I love you, Yiying.

Chapter 1

Introduction

Traditionally, mathematical coding theory focused on finding codes with large minimal Hamming distance given the code sizes. There have been some well-known inequalities describing the tradeoff between the minimal distance and the code size, e.g. Gilbert-Vashamov bound. People then tried to find efficient decoding algorithms that took advantage of the structures of the codes designed. However, it might take quite some time to find such an efficient decoding algorithm. The invention of turbo codes in 1993 [4] was a milestone in coding theory that brought the power of decoding algorithms to people's attention. And it led people to review the code design process from a new perspective. From this "modern perspective" the workflow is reversed: given a fast efficient decoding algorithm, people want to find good codes under the decoding scheme. This thesis follows the "modern perspective" and deals with the low-density parity-check (LDPC) codes proposed by Gallager in 1962 [8]. Gallager designed for LDPC codes a decoding algorithm whose computational complexity was beyond the limits of computers at that time. Therefore they were forgotten for quite a long time until rediscovered in 1990's[19]. Since then research on LDPC codes has been a hot area in modern coding theory.

A LDPC code is a binary linear code equipped with a sparse parity-check matrix H. To each parity-check matrix we can associate a bipartite graph called Tanner graph, whose two sets of vertices are called check nodes, corresponding to the columns of H, and bit nodes, corresponding to rows of H, respectively. There are several efficient decoding schemes that can be applied on Tanner graphs. In such schemes messages are passed back and forth between the two sets of verticies of

the Tanner graphs. These verticies try to make the best estimations based on the received information and then broadcast the estimations to their neighbors. These decoding algorithms converge fast thanks to the sparseness of the parity check matrices. Richardson and Urbanke [23] proposed a valuable scheme, called density evolution, to analyze these message passing iterative decoding (MPID) algorithms in the asymptotic sense. They pointed out how the asymptotic behavior of a MPID algorithm related to the degree information of the underlying Tanner graph and therefore to the code ensemble defined by the graph. Following their steps the first contribution of this thesis is a new numerical approach, based on curve fitting, designed in chapter 3 to find asymptotically good LDPC code ensembles, which yields better results than those existed in literatures, under the simplest communication channel: the binary erasure channel (BEC).

It is interesting to notice that the numerical methods, including those documented in literatures, tend to favor graphs whose check nodes have nearly the same degree. An explanation was given by Shokrollahi and Oswald [21] who took a study on sequences of code ensembles, instead of individual code ensembles. Roughly speaking, a capacity-achieving sequence of code ensembles is one whose performance under a MPID algorithm converge to the limit determined by Shannon capacity. Shokrollahi and Oswald proposed a systematic way to construct capacity-achieving sequences under BEC. They also defined two quantities to measure the convergence speeds of the capacity-achieving sequences. At the end of their paper they asked a question about the convergence speed of a special sequence. The answer to this question is provided in chapter 4.

We have seen evidences that MPID algorithms can yield performance close to the theoretical limit. Now we are wondering what is the cause of decoding errors. Intuitively, since a MPID algorithm operates locally on the Tanner graph (recall that each node receives messages only from the neighbors and sends them only to the neighbors) it can't distinguish the underlying graph from other graphs with the same local structure. So it is natural to consider finite unramified covers of a Tanner graph, which are all locally the same. Let $\tilde{T}$ be a m-fold unramified cover of a Tanner graph T and $\tilde{C}$ be the code defined by $\tilde{T}$. The code length of $\tilde{C}$ will be m times the code length of C because each vertex in T is lifted up to m vertices

in $\tilde{T}$. For a codeword

$$\mathbf{c} = (c_{(1,1)}, \ldots, c_{(1,m)}; c_{(2,1)}, \ldots, c_{(2,m)}; \ldots; c_{(n,1)}, \ldots, c_{(n,m)})$$

of $\tilde{C}$, where n is the code length of C, the unscaled pseudo-codeword $\mathbf{p}(\mathbf{c})$ corresponding to $\mathbf{c}$ is defined as

$$\mathbf{p}(\mathbf{c}) = (p_1, \ldots, p_n),$$

where $p_i = \sum_{k=1}^{m} c_{(i,k)}$. Since the (unscaled) pseudo-codewords come from codewords on graph covers, which are locally same as the Tanner graph, they compete with the real codewords of C during the decoding process. Therefore it is important to understand pseudo-codewords.

Koetter and Vontobel [11] defined the smallest cone containing all pseudo-codewords, called the fundamental cone, by a set of inequalities related to the parity-check matrix. Two characterizations for pseudo-codewords were given in [10] by Koetter-Li-Vontobel-Walker. The first says that pseudo-codewords are lattice points in the fundamental cone that are congruent to codewords modulo 2. And the second wraps the pseudo-codewords in the Taylor expansion of a rational function emerged from a graph. If C is a cycle code, which means that all the bit nodes have degree 2, the pseudo-codewords are captured exactly by the monomials in the Taylor expansion of the rational function. If C is a general code not all monomials of the rational function correspond to the pseudo-codewords. A special property must be satisfied. So a question was raised in [10]: given a general C does there exist a rational function whose monomials correspond to the pseudo-codewords? Chapter 5 is dedicated to answer this question. We will briefly report the main result in the following.

Given two vectors $\mathbf{u} = (u_1, \ldots, u_n)$ and $\mathbf{p} = (p_1, \ldots, p_n)$ in $\mathbb{R}^n$, denote by $\mathbf{u}^{\mathbf{p}}$ the monomial $u_1^{p_1} \cdots u_n^{p_n}$. Then for a general binary parity-check code C of length n the generating function of the pseudo-codewords of C can be written as

$$Z_C(u_1, ..., u_n) = \sum_{\mathbf{p}=(p_1,...,p_n) \text{ pseudo-codeword of } C} \mathbf{u}^{\mathbf{p}}.$$

Using the first characterization of pseudo-codewords mentioned above, we shall

show that

Theorem 1.1. *Let C be an LDPC code of length n with parity check matrix H and fundamental cone $\mathcal{K}(H)$. Then its zeta function is a rational function of the form*

$$Z_C(u_1, ..., u_n) = \frac{\sigma_\Pi(\mathbf{u})}{\prod_{1 \le i \le d}(1 - \mathbf{u}^{\mathbf{w}_i})},$$

where $\mathbf{w}_1, ..., \mathbf{w}_d$ are pseudo-codewords which generate the cone, namely,

$$\mathcal{K}(H) = \{\lambda_1 \mathbf{w}_1 + \cdots + \lambda_d \mathbf{w}_d \ : \ \lambda_i \ge 0 \text{ for } i = 1, ..., d\},$$

and $\sigma_\Pi(\mathbf{u}) = \sum_{\mathbf{p}} \mathbf{u}^{\mathbf{p}}$ is the sum over pseudo-codewords $\mathbf{p}$ in the fundamental parallelopiped

$$\Pi = \{\lambda_1 \mathbf{w}_1 + \cdots + \lambda_d \mathbf{w}_d : 0 \le \lambda_i < 1, i = 1, ..., d\}.$$

The latter part of chapter 5 tries to find the generators $\mathbf{w}_i$'s of the fundamental cone for some special parity-check matrices without running any algorithms.

The rest of thesis is organized as follows. Chapter 2 reviews the backgrounds for LDPC codes and density evolution especially over BEC. In chapter 3 we reports the new approaches to find asymptotically good LDPC code ensembles over BEC. We then try to answer the open question raised by Shokrollahi and Oswald in chapter 4. Chapter 5, the most important part, is devoted to the discussion of pseudo-codewords and presents two rational functions that enumerates the pseudo-codewords.

Chapter 2

LDPC Codes and Density Evolution

2.1 LDPC Codes and Tanner graphs

In 1962, Gallager invented the low-density parity-check (LDPC) code in his thesis [8] together with some iterative decoding algorithms. This class of new codes, however, was neglected for a long time until it was rediscovered in 1990's [19][26]. A Low-Density Parity-Check (LDPC) code $\mathcal{C}$ is a binary linear code defined by a $m \times n$ sparse parity check matrix H, i.e. $\mathcal{C} = \{x \in \mathbb{F}_2^n | Hx^T = 0\}$. To each parity check matrix H we associate a bipartite graph G called Tanner graph with two vertex sets V an C, where V is the set of n bits in a codeword and C is the set of m parity check equations. The elements in V are called bit nodes, or message nodes, and those in C are called check nodes. There is an edge connecting a bit node v and a check node c if and only if the bit v shows up in the parity check equation c. Thus a codeword is an assignment of values 0 or 1 to the bit nodes of the Tanner graph such that at each check node its neighbors sum to 0. Note that a code may be defined by different parity check matrices. Therefore the Tanner graph of a code is not unique. In a Tanner graph, if all the bit nodes have degree d_v and all the check nodes have degree d_c the corresponding code is called a (d_v, d_c) regular LDPC code. Otherwise it is said to be irregular.

2.2 Message Passing Iterative Decoding (MPID) Algorithm

The advantage of LDPC codes is that there are several fast and effective decoding algorithms that can be performed on the Tanner graph. The most frequently used algorithms are all based on a message passing scheme, e.g. belief propagation, sum-product or min-sum algorithm. The following is a description of the belief propagation algorithm.

The likelihood of a binary random variable x is defined as $\frac{P(x=0)}{P(x=1)}$, denoted by $L(x)$. Similarly, the conditional likelihood of x given y is $\frac{P(x=0|y)}{P(x=1|y)}$, denoted by $L(x|y)$. To make computation easier, the message (belief) being passed is log-likelihood, i.e. $\log L(x)$ or $\log L(x|y)$.

When a corrupted codeword is received, each bit node computes the log-likelihood based on the channel and the received value, which measures whether the value at that bit should be a 0 or 1. Then the messages (log-likelihoods) are sent to the adjacent check nodes along the edges. According to the updating rule on check nodes, each check node computes the new messages based on incoming ones and then sends them back to its neighboring bit nodes. This finishes one round of iteration. When the bit nodes receive the messages from the neighboring check nodes they make decisions of their values. If the decisions of the bit nodes happen to form a codeword, the iteration stops. Otherwise, the bit nodes send messages to their neighbors based on the updating rule on bit nodes and a new round of iteration starts. The messages are sent back and forth along the edges until a codeword is found. The updating rules can be formulated as follows.

The message sent from the bit node v to an adjacent check node c at round l is:

$$m_{vc}^{(l)} = \begin{cases} m_v & \text{if } l = 0, \\ m_v + \sum_{c' \in C_v - \{c\}} m_{c'v}^{(l-1)} & \text{if } l \geq 1. \end{cases}$$

The message sent from the check node c to an adjacent bit node v at round l is:

$$m_{cv}^{(l)} = \log \frac{1 + \prod_{v' \in V_c - \{v\}} \tanh(m_{v'c}^{(l)}/2)}{1 + \prod_{v' \in V_c - \{v\}} \tanh(m_{v'c}^{(l)}/2)}.$$

Here C_v is the set of check nodes incident to bit node v, and V_c is the set of bit nodes incident to check node c. Note that the outgoing message from c to v (resp. from v to c) uses all incoming messages from the neighbors of c (resp. v) excluding v (resp. c).

The updating rules assume that the incoming messages are independent. This assumption holds only if the associated Tanner graph is a tree. The numerical results, however, show that the algorithm is still effective even if the Tanner graph contains cycles, in which case the independence assumption is violated.

2.3 Density Evolution and Threshold

We are interested in asymptotic properties of LDPC codes. Instead of looking at a particular code, we consider ensembles of codes. For instance, an ensemble $\mathcal{C}^n(d_v, d_c)$ consists of all (d_v, d_c)- regular codes of block length n with uniform distribution. Richardson and Urbanke [23] showed that the performance of codes in the ensemble $\mathcal{C}^n(d_v, d_c)$ is concentrated around the expected, or average, performance. In other words, probabilistically speaking, there is no big difference between the performance of codes in an ensemble.

In the same theorem, Richardson and Urbanke [23] also showed that when the block length n is large enough the performance of an arbitrary (d_v, d_c)-regular code is very close to that of a (d_v, d_c)-regular code whose Tanner graph is a tree. Therefore when we do the asymptotic analysis for an ensemble we can safely assume that we are treating a Tanner graph that is a tree. Thus the independence assumption holds.

This remarkable theorem is listed below. Interested readers can refer to [23] (Theorem 2, page 613) for the proof and other details.

Theorem 2.1. *(Richardson-Urbanke) Over the probability space of all graphs $\mathcal{C}^n(d_v, d_c)$ and channel realizations let Z be the number of incorrect messages among all nd_v bit-to-check node messages passed at iteration l. Let p be the expected number of incorrect messages passed along an edge with a tree-like directed neighborhood of depth at least $2l$ at the lth iteration. Then, there exist positive constants $\beta = \beta(d_v, d_c, l)$ and $\gamma = \gamma(d_v, d_c, l)$ such that*

[Concentration Around Expected Value] For any $\epsilon > 0$ we have

$$Pr\{|Z - \mathbb{E}[Z]| > nd_v\epsilon/2\} \leq 2e^{-\beta\epsilon^2 n}.$$

[Convergence to Cycle-Free Case] For any $\epsilon > 0$ and $n > \frac{2\gamma}{\epsilon}$ we have

$$|\mathbb{E}[Z] - nd_v p| < nd_v\epsilon/2.$$

[Concentration Around Cycle-Free Case] For any $\epsilon > 0$ and $n > \frac{2\gamma}{\epsilon}$ we have

$$Pr\{|Z - nd_v p| > nd_v\epsilon\} \leq 2e^{-\beta\epsilon^2 n}.$$

Based on the argument of concentration around expected value in the above theorem we shall not distinguish between a (d_v, d_c) ensemble and a (d_v, d_c)-regular code in what follows.

Density evolution is a powerful tool developed by Richardson and Urbanke in [23] to do the asymptotic analysis. It tracks the probability distribution of the messages after each round of iteration in the MPID algorithm. And we can obtain a lot of useful information from this probability distribution.

2.3.1 BEC with Belief Propagation

A binary erasure channel (BEC) has input alphabet $\mathcal{X} = \{0, 1\}$ and output alphabet $\mathcal{Y} = \{0, 1, e\}$, where e stands for erasure. The transition probability is as follows:

$$P(b|a) = \begin{cases} 1 - \epsilon & \text{if } b = a, \\ \epsilon & \text{if } b = e, \\ 0 & \text{otherwise.} \end{cases}$$

2.3.1.1 Decoding algorithm

The message alphabet in belief propagation over BEC is a set of 2 symbols, $+\infty$ and 0. Assuming that a all-one codeword is sent we describe the decoding process as follows.

If a bit receives an erasure, then the log likelihood is 0. If a bit receives 1, then the log likelihood is $+\infty$. A check node c sends $+\infty$ to a bit node v if and only if all the messages from adjacent bit nodes excluding v are $+\infty$, else it sends 0. A bit node v sends $+\infty$ to a check node c if and only if v is not erased or at least one of the check nodes adjacent to v other than c sends $+\infty$, otherwise it sends 0. The decoding is successful if all the bit nodes receive the message 1 at some round.

2.3.1.2 Analysis

Let $p_0^{(0)}$ be the average probability that the message sent from a bit node to a check node at time zero is equal to 0. Let $p_0^{(l)}$ denote the corresponding probability at iteration l and $q_0^{(l)}$ denote the the average probability of messages sent from a check node to a bit node being 0 in the lth iteration.

For a (d_v, d_c)-regular LDPC code, we have

$$p_0^{(l+1)} = p_0^{(0)} {q_0^{(l)}}^{d_v - 1}$$

and

$$q_0^{(l)} = 1 - (1 - p_0^{(l)})^{d_c - 1}.$$

Combining these two, we get

$$p_0^{(l+1)} = p_0^{(0)} (1 - (1 - p_0^{(l)})^{d_c - 1})^{d_v - 1}.$$

This is a recursion for $p_0^{(l)}$. Suppose that we have a (3,6)-regular LDPC code and that $\epsilon = p_0^{(0)} = 0.4$. We can compute $p_0^{(1)}, p_0^{(2)}, \ldots$ to be $0.3402, 0.3062, 0.2818, 0.2617, 0.2438, 0.2266, \ldots$. If, however, we start with $p_0^{(0)} = 0.45$, then we get $0.4058, 0.3858, 0.3748, 0.3681,\ 0.3639, 0.3612, \ldots$. The former sequence tends to 0, whereas the latter tends to 0.3554. Therefore the first will lead to successful decoding but the second fails. Further experiment shows that the cutoff in behavior is at $p_0^{(0)} = 0.42944$. This value is called the **threshold**.

2.3.2 BSC with Gallager's Decoding Algorithm A

A binary symmetric channel (BSC) has input alphabet $\mathcal{X} = \{0, 1\}$ and output alphabet $\mathcal{Y} = \{0, 1\}$. The transition probability is

$$P(b|a) = \begin{cases} 1 - \epsilon & \text{if } b = a, \\ \epsilon & \text{if } b \neq a. \end{cases}$$

The error probability ϵ is also called the crossover probability.

For convenience, in this subsection we assume that the bits sent over channels are -1 or 1 and that a all-one codeword is transmitted over the channel.

2.3.2.1 Decoding algorithm

We describe Gallager's decoding algorithm A in this subsection. Let m_v be the received value at bit node v. The updating rules are as follows.

The message sent from the bit node v to check node c at round l is

$$m_{vc}^{(l)} = \begin{cases} m_v & \text{if } l = 0, \\ -m_v & \text{if } l \geq 1 \text{ and } m_{c'v}^{(l-1)} = -m_v \text{ for all } c' \in C_v - \{c\}, \\ m_v & \text{otherwise.} \end{cases}$$

The message sent from the check nodes c to the bit node v at round l is

$$m_{cv}^{l} = \prod_{v' \in V_c - \{v\}} m_{v'c}^{(l)} \quad .$$

Here, as before, C_v is the set of check nodes incident to bit node v, and V_c is the set of bit nodes incident to check node c. Consequently the message alphabet in this decoding algorithm is $\{-1, 1\}$, which is the same as the information alphabet sent over the channel. So this is actually a hard decoding scheme. The decoding algorithm successfully exits if all the bit nodes receive one at certain round. Otherwise it quits when the messages received at the bit nodes form a codeword, which results in a decoding error.

2.3.2.2 Analysis

For $m \in \{-1, 1\}$, let $p_m^{(l)}$ (resp. $q_m^{(l)}$) denote the average probability that the message sent from a bit node to a check node (resp. from a check node to a bit node) in the lth iteration is m. Then $p_1^{(0)} = 1 - \epsilon$ and $p_{-1}^{(0)} = \epsilon$.

It follows from the decoding algorithm that

$$\begin{aligned}&(q_{-1}^{(l)},\, q_1^{(l)})\\ &= (\frac{1-(1-2p_{-1}^{l-1})^{d_c-1}}{2},\, \frac{1+(1-2p_{-1}^{(l-1)})^{d_c-1}}{2})\end{aligned}$$

and

$$\begin{aligned}&(p_{-1}^{(l)},\, p_1^{(l)})\\ &= (p_1^{(0)}(q_{-1}^{(l)})^{d_v-1} + p_{-1}^{(0)}(1-(q_1^{(l)})^{d_v-1}),\, p_{-1}^{(0)}(q_1^{(l)})^{d_v-1} + p_1^{(0)}(1-(q_{-1}^{(l)})^{d_v-1})).\end{aligned}$$

Putting together, we obtain

$$\begin{aligned}p_{-1}^{(l)} = p_{-1}^{(0)} &- p_{-1}^{(0)}(\frac{1+(1-2p_{-1}^{(l-1)})^{d_c-1}}{2})^{d_v-1}\\ &+ (1-p_{-1}^{(0)})(\frac{1-(1-2p_{-1}^{(l-1)})^{d_c-1}}{2})^{d_v-1}.\end{aligned}$$

This shows that $p_{-1}^{(l)}$ is an increasing function in $p_{-1}^{(0)}$. The threshold for Gallager's decoding algorithm A is

$$\epsilon^* = sup\{p_{-1}^{(0)} | \lim_{l\to\infty} p_{-1}^{(l)} = 0\}.$$

Thus, for all $p_{-1}^{(0)} < \epsilon^*$, $\lim_{l\to\infty} p_{-1}^{(l)} = 0$, hence the information can be correctly decoded using this algorithm.

Table 2.1 compares the thresholds and the maximum tolerable crossover probabilities ϵ_{opt} for some regular codes. Here ϵ_{opt} is obtained by solving the equation $r = 1 - H(\epsilon_{opt})$ where r is the rate of the regular code and $H(\cdot)$ is the entropy function.

d_v	d_c	Rate	ϵ^*	ϵ_{opt}
3	6	0.5	0.04	0.11
4	8	0.5	0.047	0.11
5	10	0.5	0.027	0.11
3	5	0.4	0.061	0.146
4	6	0.333	0.066	0.174
3	4	0.25	0.106	0.215

Table 2.1. Thresholds of Gallager's Decoding Algorithm A and the maximum tolerable error probability under BSC[23]

Chapter 3

Approaches to Find Good Degree Distributions of LDPC Codes Over BEC

3.1 Introduction

We have shown two examples of density evolution analysis of regular codes over different channels in last chapter. The analysis can be extended to irregular codes too. For instance, we shall consider irregular codes over BEC with belief propagation decoding algorithm in the following analysis. Let

$$\lambda(x) = \sum_{i=2}^{d_v^{max}} \lambda_i x^{i-1} \text{ and } \rho(x) = \sum_{i=2}^{d_c^{max}} \rho_i x^{i-1},$$

where d_v^{max} and d_c^{max} are the maximum degree of the bit nodes and check nodes respectively and λ_i(resp. ρ_i) is the fraction of edges incident to bit nodes (resp. check nodes) with degree i. Assuming that the rows in the parity-check matrix are linearly independent we can write the code rate in terms of $\lambda(x)$ and $\rho(x)$ as

$$r = 1 - \frac{\int_0^1 \rho(x)\mathrm{d}x}{\int_0^1 \lambda(x)\mathrm{d}x}.$$

Indeed, assume that the parity-check matrix has dimension m by n then we have $r = \frac{n-m}{n} = 1 - \frac{m}{n}$. Let E be the number of edges in the Tanner graph. Then

$$\begin{aligned} E\int_0^1 \rho(x)\mathrm{d}x &= \sum_{i=2}^{d_c^{max}} \frac{E\rho_i}{i} \\ &= \text{number of check nodes} \\ &= m \end{aligned}$$

Similarly,

$$\begin{aligned} E\int_0^1 \lambda(x)\mathrm{d}x &= \sum_{i=2}^{d_v^{max}} \frac{E\lambda_i}{i} \\ &= \text{number of bit nodes} \\ &= n \end{aligned}$$

Therefore

$$r = 1 - \frac{m}{n} = 1 - \frac{\int_0^1 \rho(x)\mathrm{d}x}{\int_0^1 \lambda(x)\mathrm{d}x}.$$

Assume that the erasure probability is δ. Denote by p_0^l (resp. q_0^l) the expected probability that the message sent from a bit node (resp. check node) to a check node (resp. bit node) is 0 at lth iteration. Then we have

$$p_0^{l+1} = \delta\lambda(q_0^l)$$

and

$$q_0^l = 1 - \rho(1 - p_0^l).$$

Putting them together, we get

$$p_0^{l+1} = \delta\lambda(1 - \rho(1 - p_0^l)).$$

We say that a degree distribution (λ, ρ) affords δ if the decoding is successful with the initial erasure probability δ. Then (λ, ρ) affords δ if and only if $\lim_{l\to\infty} p_0^l = 0$,

which is equivalent to that

$$\delta\lambda(1-\rho(1-x)) < x, \quad x \in (0,\delta] \tag{3.1}$$

(i.e. 0 is the only fixed point of the function $f(x) = \delta\lambda(1-\rho(1-x))$ over $[0,\delta]$). This result was also derived from another perspective in [18].

The supremum of δ that a degree distribution (λ,ρ) can afford is called the threshold corresponding to (λ,ρ), denoted by δ^*. From the affordability condition (3.1) we can see that

$$\delta^* <= \frac{x}{\lambda(1-\rho(1-x))}, \quad x \in (0,\delta].$$

Then the maximality of δ^* indicates that

$$\delta^* = \inf_{x\in(0,1]} \frac{x}{\lambda(1-\rho(1-x))}.$$

Many researchers realized that the irregular codes had much better performance than regular codes ([15],[22],[17]). Furthermore, the performance of irregular codes can be extremely close to the limit determined by Shannon capacity ([6],[22]).

The question considered in the following is that given the maximum left (bit node) degree d_v, the maximum right (check node) degree d_c and the code rate r how to find degree distributions $\lambda(x)$ and $\rho(x)$ with threshold as large as possible over BEC under the belief propagation algorithm.

3.2 Two-stage methods

3.2.1 Linear programming approach

This approach was first suggested in [18] and [15].Recall that a degree distribution (λ,ρ) affords δ if

$$\delta\lambda(1-\rho(1-x)) < x, \quad x \in (0,\delta]. \tag{3.2}$$

Given δ, let $\{x_i\}_{i=1}^N$ be a set of N points chosen from $(0,\delta]$. Typically we can choose $x_i = \frac{i\delta}{N}$. We require that

$$\delta\lambda(1-\rho(1-x_i)) < x_i \quad i = 1, \ldots, N.$$

Given $\rho(x)$ and δ, all of the N inequalities above are linear in $\lambda_2, \ldots, \lambda_{d_v}$. Therefore we can set up the following linear program LP1.

LP1:

$$\begin{aligned}
&\text{minimize} && 1 \\
&\text{s.t} && \sum_{i=2}^{d_v} \lambda_i = 1 \\
& && 0 \le \lambda_i \le 1 \quad i = 2, \ldots, d_v \\
& && \sum_{i=2}^{d_v} \frac{\lambda_i}{i} = \frac{\int_0^1 \rho(x)\mathrm{d}x}{1-r}, \quad \text{(rate constraint)} \\
& && \delta\lambda(1-\rho(1-x_i)) < x_i, \quad i = 1, \ldots, N
\end{aligned}$$

Thus a binary-search algorithm can be used to optimize $\lambda(x)$ given $\rho(x)$.

1. Assign an initial value to δ
2. Solve LP1. If a feasible solution exists then increase the value of δ, otherwise decrease the value.
3. If the increment of δ is small enough then stop, otherwise go to step 2

If, instead, $\lambda(x)$ is given a similar idea can be used to optimize $\rho(x)$. Observe that (3.2) is equivalent to

$$\rho(1-\delta\lambda(x)) > 1-x. \tag{3.3}$$

Indeed, if we let $t = 1-\rho(1-x)$ then $x = 1-\rho^{-1}(1-t)$ since monotonicity of $\rho(x)$ guarantees the existence of $\rho^{-1}(x)$ on $(0,1)$. So (3.2) can be written as

$$\delta\lambda(t) < 1-\rho^{-1}(1-t), \text{ i.e. } \rho^{-1}(1-t) < 1-\delta\lambda(t).$$

Applying ρ to the two sides and using the monotonicity of $\rho(x)$ again we get (3.3). So we can set up a similar linear program and use a binary search to find a good

δ^*	$\lambda(x)$ and $\rho(x)$	$\delta^*/0.5$
0.4845	$\lambda(x) = 0.25594x + 0.37910x^2 + 0.127423x^{10} + 0.237538x^{11}$ $\rho(x) = x^6$	0.9690
0.4834	$\lambda(x) = 0.25800x + 0.32950x^2 + 0.01056x^4 + 0.07637x^7 + 0.32557x^8$ $\rho(x) = 0.47013x^5 + 0.41733x^6 + 0.11253x^{20}$	0.9668
0.4886	$\lambda(x) = 0.11286x + 0.06153x^2 + 0.12936x^3 + 0.25559x^7 + 0.24226x^8 + 0.02550x^{24} + 0.17290x^{25}$ $\rho(x) = 0.17263x^5 + 0.28658x^6 + 0.14805x^{28} + 0.39274x^{29}$	0.9772

Table 3.1. Degree distributions for rate 0.5 codes with linear programming approach [25]

$\rho(x)$ given $\lambda(x)$.

Thus the algorithm to find good (λ, ρ) can be described as follows. We start with a fixed $\rho(x)$ and solve the corresponding linear program several times trying different values of δ to find an optimized $\lambda(x)$ corresponding to $\rho(x)$. Fixing this $\lambda(x)$ leads to an optimized $\rho(x)$ corresponding to it. Applying these two steps back and forth we will finally get a good degree distribution (together with a good threshold). Some numerical results([25]) are presented in Table 3.1.

One disadvantage of this approach is that the integral $\int_0^1 \rho$, hence $\int_0^1 \lambda$, is fixed throughout the process. Therefore various values of $\int_0^1 \rho$ ($\int_0^1 \lambda$) have to be tried to get good results.

3.2.2 Soft-max and Curve-fitting approach

3.2.2.1 Soft-max

The linear programming approaches have another disadvantage. Even if looking for a good λ(resp. ρ) given ρ(resp. λ) we have to solve LP1 several times trying different values of δ. The soft-max and the curve-fitting approach, instead, can optimize λ(resp. ρ) given ρ(resp. λ) by solving a (nonlinear) program once.

We define the soft maximum([12]) of a set of real numbers $\{z_i\}$,denoted by $\max^{(K)}\{z_i\}$, as

$$\max^{(K)}\{z_i\} = \frac{1}{K}\log\left(\sum_i \exp(Kz_i)\right).$$

Clearly $\max^{(K)}\{z_i\} \geq \max\{z_i\}$ and it can be shown that

$$\max\{z_i\} = \lim_{K\to\infty} \max^{(K)}\{z_i\}.$$

Recall that the threshold δ^* corresponding to (λ, ρ) is

$$\delta^* = \inf_{x\in(0,1]} \frac{x}{\lambda(1-\rho(1-x))}, \text{or} \quad \frac{1}{\delta^*} = \sup_{x\in(0,1]} \frac{\lambda(1-\rho(1-x))}{x}.$$

Given $\rho(x)$, we choose N points $\{x_i\}_{i=1}^N$ from $(0,1)$ and evaluate $y_i := \frac{\lambda(1-\rho(1-x_i))}{x_i}$. Besides, we also want to throw in the limit when x tends to 0. So let $y_0 := \lim_{x\to 0} \frac{\lambda(1-\rho(1-x))}{x} = \rho'(1)\lambda_2$. Note that each of $y_i (i = 0, \ldots, N)$ is linear in $\lambda_2, \ldots, \lambda_{d_v}$ and $\max^{(K)}\{y_i\}$ is a nonlinear function with the same independent variables. We want to minimize the maximum of $\{y_i\}_{i=0}^N$ so we can set up the following nonlinear program, NP1, with linear constraints:

NP1:

$$\begin{array}{lll} \underset{\{\lambda_i\}}{\text{minimize}} & \max^{(K)}\{y_i\} & \\ \text{s.t} & \lambda_i \geq 0 \quad \text{for} \quad 2 \leq i \leq d_v \\ & \sum_{i=2}^{d_v} \lambda_i = 1 \\ & \sum_{j=2}^{d_v} \lambda_j/j = \frac{\int_0^1 \rho}{1-r} \end{array}$$

In NP1, although the soft-max converges to max as $K \to \infty$ a too large K results in overflow of floating-point numbers in the computation. In Matlab the maximum floating-point number is $1.7977e+308$. In Table 3.2, $K = 200$ is used.

If, instead, $\lambda(x)$ is given and $\rho(x)$ is to be optimized the idea above does

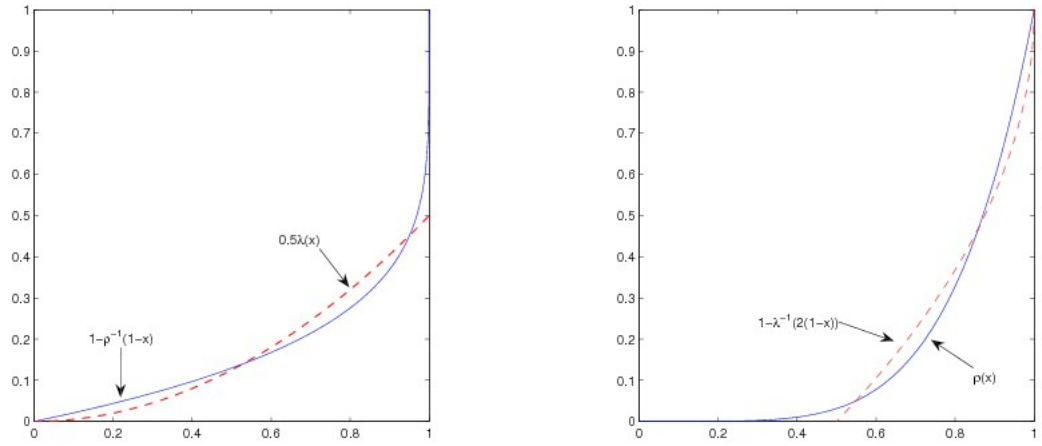

Figure 3.1. Graphs of C_1, C_2 (left) and C_1', C_2' where $\lambda(x) = x^3, \rho(x) = x^6$ and code rate is 0.5

not work well. In this case each y_i is a nonlinear function in $\rho_2, \ldots, \rho_{d_c}$ and the execution of the algorithm is slow. A solution is proposed in the next subsection.

3.2.2.2 Curve-fitting approach

Suppose that now we want to find a good $\rho(x)$ given $\lambda(x)$. The idea is based on the following observation. First note that

$$\delta^* = \inf_{x\in(0,1]} \frac{x}{\lambda(1-\rho(1-x))} = \inf_{x\in(0,1]} \frac{1-\rho^{-1}(1-x)}{\lambda(x)}.$$

If the threshold of (λ, ρ) ensemble is close to the theoretical upper bound $1 - r$, then the two curves $C_1 : y = 1 - \rho^{-1}(1 - x)$ and $C_2 : y = (1 - r)\lambda(x)$ should be "close" to each other. Let C_1' and C_2' be two curves that are symmetric about $x + y = 1$ to C_1 and C_2 respectively. Then $C_1' : y = \rho(x)$ and $C_2' : y = 1 - \lambda^{-1}(\frac{1-x}{1-r})$ should be "close" to each other too. Fig. 3.1 shows graphs of C_1, C_2, C_1' and C_2' for $\lambda = x^3$ and $\rho = x^6$.

So the idea is to, given $\lambda(x)$, sample points on C_2' and then find the $\rho(x)$ that best fit those points in certain sense of closeness. The points on C_2', (x_i, y_i), can

be easily generated since a parametric equation of C_2' is

$$\begin{cases} x = 1 - (1-r)\lambda(t) \\ y = 1 - t \end{cases} \qquad t \in [0,1].$$

We hope to minimize $\max\{1-\lambda^{-1}(\frac{1-x}{1-r})-\rho(x)\}$, or approximately $\max_i\{y_i-\rho(x_i)\}$. Again we can approximate this discrete max function by the soft-max. Therefore the following nonlinear program NP2 can be set up.

NP2:

minimize	$\max^{(K)}\{y_i - \rho(x_i)\}$
s.t	$\rho_i \geq 0 \quad \text{for} \quad 2 \leq i \leq d_c$
	$\sum_{i=2}^{d_c} \rho_i = 1$
	$\sum_{j=2}^{d_c} \frac{\rho_j}{j} = (1-r)\int_0^1 \lambda$

Thus, in order to find good pair of (λ, ρ), we fix a proper K and start from a $\lambda(x)$ (or $\rho(x)$),then solve NP1 and NP2 back and forth.

Note that the soft-max approach in 3.2.2.1 also falls in curve-fitting category. In NP1 the objective function is

$$\max^{(K)}\{y_i\},$$

where

$$y_i = \frac{\lambda(1-\rho(1-x_i))}{x_i} = \frac{\lambda(\tilde{x}_i)}{1-\rho^{-1}(1-\tilde{x}_i)}, \tilde{x}_i = 1-\rho(1-x_i)$$

Thus we are approximating $1-\rho^{-1}(1-x)$ with $(1-r)\lambda(x)$ in NP1 using a different measure of closeness.

Some good codes with rate 0.5 found by curve-fitting (soft-max) approach are presented in Table 3.2.

δ^*	$\lambda(x)$ and $\rho(x)$	$\delta^*/0.5$
0.4939	$\lambda(x) = 0.2995x + 0.1604x^2 + 0.0745x^4 + 0.1459x^5 + 0.0362x^6 + 0.2835x^{19}$ $\rho(x) = 0.3304x^6 + 0.6696x^7$	0.9878
0.4953	$\lambda(x) = 0.2902x + 0.1401x^2 + 0.0623x^3 + 0.0251x^4 + 0.0543x^5 + 0.0746x^6 + 0.0643x^7 + 0.0255x^8 + 0.0836x^{23} + 0.1776x^{24}$ $\rho(x) = 0.1064x^6 + 0.8936x^7$	0.9906
0.4957	$\lambda(x) = 0.271x + .1283x^2 + .0845x^3 + 0.0115x^5 + 0..0903x^6 + 0.1195x^7 + 0.0102x^8 + .2846x^{29}$ $\rho(x) = 0.6328x^7 + 0.3672x^8$	0.9914

Table 3.2. Degree distributions for rate 0.5 codes with curve-fitting approach (K=200)

3.2.3 Some discussions

There is a disadvantage in all these two-stage approaches. We start from a λ (or ρ), then $\int \lambda$ and $\int \rho$ are not changed for all λ and ρ produced in the algorithm. So various values of $\int_0^1 \rho$(or $\int_0^1 \lambda$) have to be tried to find good results. In [24] the author showed that if the gap between the threshold of (λ, ρ) code and the Shannon limit $1 - r$ tends to zero then the average right degree $a_r = \frac{1}{\int \rho}$ must go to infinity. But if the maximum right degree d_c is fixed,which is the case we are considering now, it is not true that the integral of the optimum ρ must be very close to zero. Actually the claim in [24] indicated that to get close to the Shannon limit d_c must be large. So we have no idea what value of $\int \rho$ leads to best result if d_c is fixed. Some ideas, however, in [2] can be borrowed to find the following lower and upper bound of $\int_0^1 \rho$.

Proposition 3.1. *If the* (λ, ρ) *code affords* δ*, then*

$$\int \rho \geq \frac{1}{\frac{1}{\lambda_2 \delta} + 1}.$$

Proof.

$$\begin{aligned}\int \rho(\rho'(1)+1) &= (\sum \frac{\rho_i}{i})(1+\sum (i-1)\rho_i) \\ &= (\sum \frac{\rho_i}{i})(\sum i\rho_i) \\ &\geq 1,\end{aligned}$$

where Cauchy-Schwartz inequality is used in the last step.
Thus the desired result follows from

$$\frac{1-\int \rho}{\int \rho} \leq \rho'(1) \leq \frac{1}{\lambda_2 \delta}.$$

(The second inequality follows from the stability condition) □

Proposition 3.2. *Assume that the* (λ, ρ) *code has rate* r *and affords* δ*. Let* $\epsilon = 1 - \frac{\delta}{1-r}$*. If* $\epsilon < \frac{3r}{3+3r}$ *then the equation*

$$f(x) := -\delta x^2 + \left[(\frac{6}{1-r} - 2)\delta - 1\right] x - 2 = 0$$

has only one real root α *in* $[0,1]$*. And*

$$\int \rho \leq (1-r)\frac{\alpha+2}{6}.$$

Proof. First,

$$\int \lambda = \sum_i \frac{\lambda_i}{i} \leq \frac{\lambda_2}{2} + \frac{1-\lambda_2}{3} = \frac{\lambda_2 + 2}{6}$$

Another inequality to be used is

$$\rho'(1) \geq \frac{1-\int \rho}{\int \rho},$$

which was shown in proposition 3.1.

Now,

$$\begin{aligned}\delta \le \frac{1}{\lambda_2 \rho'(1)} &\le \frac{1}{\lambda_2 \frac{1-\int \rho}{\int \rho}} \\ &= \frac{1}{\lambda_2 \frac{1-(1-r)\int \lambda}{(1-r)\int \lambda}} \\ &\le \frac{1}{\lambda_2 \frac{1-(1-r)\frac{\lambda_2+2}{6}}{(1-r)\frac{\lambda_2+2}{6}}} \\ &= \frac{(1-r)(\lambda_2+2)}{\lambda_2\left[6-(1-r)(\lambda_2+2)\right]}\end{aligned}$$

Rewriting the above inequality, we get an inequality about λ_2.

$$-\delta\lambda_2^2 + \left[(\frac{6}{1-r} - 2)\delta - 1\right]\lambda_2 - 2 \le 0, \text{ i.e. } f(\lambda_2) \le 0.$$

$f(0) = -2 < 0$. If $\epsilon < \frac{3r}{3+3r}$ it is easy check that $f(1) > 0$. So there is a unique zero,α, of $f(x)$ in $[0, 1]$. Hence $\lambda_2 \le \alpha$ and $\int \rho \le (1-r)\frac{\alpha+2}{6}$. □

Example. We want to find good (λ, ρ) of rate 0.5 with threshold at least 0.49. Then $\epsilon = 1 - \frac{0.49}{0.5} = 0.02 < \frac{3(0.5)}{3+3(0.5)} = 1/3$. The equation $-0.49x^2 + 3.9x - 2 = 0$ has a unique real root $\alpha = 0.551$. By proposition 1,

$$\int \rho \le 0.5(2.551)/6 = 0.2126$$

So we only need to try those ρ such that $\int \rho \le 0.2126$ or, equivalently,those λ such that $\int \lambda \le 0.4252$.

Actually, if $d_c = 20, 25, 30$ respectively the optimum $\rho(x)$ found with curve-fitting approach (Table 3.2) has $\int \rho = 0.1309, 0.1269, 0.1199$ respectively.

3.3 joint design methods (Differential Evolution)

The authors of [25] proposed a scheme to optimize (λ, ρ) using differential evolution, which is an algorithm to attack general nonlinear program. λ and ρ are

δ^*	$\lambda(x)$ and $\rho(x)$	$\delta^*/0.5$
0.4939	$\lambda(x) = 0.29730x + 0.17495x^2 + 0.24419x^5 + 0.28353x^{19}$ $\rho(x) = 0.33181x^6 + 0.66818x^7$	0.9878
0.4948	$\lambda(x) = 0.27692x + 0.20256x^2 + 0.26207x^6 + 0.25843x^{24}$ $\rho(x) = 0.10531x^6 + 0.89468x^7$	0.9896
0.4955	$\lambda(x) = 0.26328x + .18020x^2 + .27000x^6 + .28649x^{29}$ $\rho(x) = .6328x^7 + .3672x^8$	0.9910

Table 3.3. Degree distributions for rate 0.5 codes with differential evolution [25]

optimized simultaneously during the execution of the algorithm, which is an advantage over the approaches we discussed in the previous sections. Same as the approaches discussed in previous sections, this method also involves discretization of x in $(0, 1]$. Some good codes found by this approach are listed in Table 3.3.

3.4 Comparison of different approaches

In this section we will discuss the the advantages and disadvantages of the approaches we mentioned in the previous sections and also some noteworthy points during the implementation of the algorithms.

All these three approaches discretize x in the interval $(0, 1]$. In the linear programming approach and the differential evolution approach the sample points show in the constraints. The discretized constraints $\delta(1 - \rho(1 - x_i)) < x_i$ approximate the constraint $\delta(1 - \rho(1 - x)) < x$. So the more the sample points are, the more constraints we put on these optimization problems and therefore the better approximation we obtain. On the other hand, the discretization shows in the objective function in curve-fitting approach, where we use the soft-max of a set of function values to approximate the maximum of the function. In practice the number of sample points will not change the results significantly if it is large enough (say, more than 100).

There is another parameter K used in the soft-max in the curving-fitting approach. The larger K is, the better the soft-max approximates the maximum. But

a too large K will result in numbers beyond the largest floating point number that a computer can represent. $K = 200$ is used in table 3.2.

One main disadvantage of the linear programming approach and the curving-fitting approach, as mentioned in section 3.2.3, is that once an initial $\lambda(x)$ (or $\rho(x)$) is chosen the integrals over $[0, 1]$ of all the degree distribution polynomials produced will remain the same during the searching process. So we have to feed initial $\lambda(x)$ (or $\rho(x)$) with various integrals to the algorithms to find a good degree distribution pair. The differential evolution approach does not have this drawback.

All of these three approaches yield threshold within 3% of the theoretical limit. The curve-fitting approach and the differential evolution approach are slightly better, which are within 2% of the theoretical limit. The linear programming approach is the easiest to implement while the differential evolution approach is the hardest. The curve-fitting approach is a trade-off between the optimality of results and the implementation difficulty. Users can choose an approach according to their needs and preferences.

Chapter 4

Capacity-Achieving Sequences for Binary Erasure Channel

4.1 Introduction

Given a code of rate r over BEC, $1-r$ is the upper bound for the erasure probability such that a decoding algorithm is successful. In the last chapter we studied numerically how to find a degree distribution (λ, ρ) whose corresponding threshold is as close to $1-r$ as possible. Interestingly, many good codes we found are, or close to, right-regular (all the check nodes have the same degree). To explain this phenomenon, Oswald and Shokrollahi carried a systematic study of capacity-achieving sequences and proposed two different measures about the convergence speed. We shall introduce their work in this chapter and answer a question proposed by Oswald and Shokrollahi in their paper [21].

Recall that the belief propagation decoding over BEC is successful on a Tanner graph with degree distribution (λ, ρ) and initial erasure probability δ if

$$\delta\lambda(1-\rho(1-x)) < x, x \in (0, \delta).$$

Given a degree distribution (λ, ρ) let $\delta(\lambda, \rho)$ be the threshold, i.e. the supremum of all δ that satisfy the above inequality. A sequence of degree distribution (λ^n, ρ^n) is said to capacity-achieving if the corresponding thresholds $\delta^n = \delta(\lambda^n, \rho^n)$ converges to the upper bound $1-r$ as n tends to infinity. Several capacity-achieving sequences

can been found in the literature. For instance, in Tornado sequence [16] (a.k.a. heavy tail/Poisson sequence in [24]) $\lambda^n(x)$ is the normalized initial segment of $-\log(1-x)$ such that $\lambda^n(1) = 1$ and $\rho^n(x)$ is the initial segment of $e^{\alpha(x-1)}$, where the constant α is chosen to meet the designing rate r. To be more specific,

$$\lambda^n(x) = \frac{1}{H(n)} \sum_{i=1}^{n} \frac{x^i}{i}, \text{ where } H(n) = \sum_{i=1}^{n} \frac{1}{i},$$

and

$$\rho^n(x) = e^{-\alpha} \sum_{i=1}^{n} \frac{\alpha^i x^i}{i!}.$$

Another capacity-achieving sequence, right-regular sequence, was documented in [24]. Now $\rho^n(x) = x^n$ and $\lambda^n(x)$ is the normalized initial segment of $(1-x)^{1/n}$. Specifically, for a fixed integer $\nu > 1$ let $N = \nu^n$ and define

$$\lambda^n(x) = \frac{1}{1 - n(N+1)\binom{\frac{1}{n}}{N+1}(-1)^N} \sum_{i=1}^{N} \binom{\frac{1}{n}}{i} (-1)^{i+1} x^i,$$

where the fractional binomial coefficients $\binom{\alpha}{N} = \frac{\alpha(\alpha-1)\cdots(\alpha-N+1)}{N!}$.

Oswald and Shokrollahi observed that in both cases people chose a function $f(x)$ (or the initial segment of the Taylor expansion) as $\lambda(x)$ (resp. $\rho(x)$) and let $\rho(x)$ (resp. $\lambda(x)$) be a polynomial constructed from $1 - f^{-1}(1-x)$. So they proposed the idea to construct a capacity-achieving sequence as follows. Define an operator Γ on functions as

$$\Gamma f(x) := 1 - f^{-1}(1-x)$$

and let

$$\mathcal{P} := \{f(x) = \sum_{k=1}^{\infty} f_k x^k, x \in [0,1] | f_k \geq 0, f(1) = 1\}.$$

The candidates of $\rho(x)$ are normalized truncations of functions in the set

$$\mathcal{A} := \{f \in \mathcal{P} \,|\, \Gamma f \in \mathcal{P}\}.$$

And then $\lambda(x)$ are constructed from $1 - f^{-1}(1-x)$ accordingly such that the code

rate meets the specification. Interested readers can refer to [21] for details.

4.2 Convergence speed of capacity-achieving sequences

Let

$$\epsilon(\lambda, \rho) = 1 - \frac{\delta(\lambda, \rho)}{1 - r}.$$

It was shown in [24][1] that

$$\epsilon \geq r^{a_r}, \text{ where } a_r = (\int_0^1 \rho)^{-1}. \tag{4.1}$$

Oswald and Shokrollahi [21] defined two quantities μ and Δ below to measure the convergence speed of a capacity-achieving sequence $(\lambda^{(n)}, \rho^{(n)})$:

$$\mu = \limsup_{n \to \infty} \frac{a_r^{(n)} \log r}{\log \epsilon(\lambda^{(n)}, \rho^{(n)})},$$

$$\Delta = \limsup_{n \to \infty} \frac{\epsilon(\lambda^{(n)}, \rho^{(n)})}{r^{a_r^{(n)}}}.$$

A sequence is called *asymptotically quasi-optimal* with constant μ if the first limit defined above exists. And it is called *asymptotically optimal* with constant Δ if the second limit exists.

By (4.1), it is easy to see that $\mu \geq 1$ and $\Delta \geq 1$. Moreover, if $\Delta < \infty$ then $\mu = 1$, i.e. asymptotically optimal sequences with any Δ are asymptotically quasi-optimal with constant 1.

The following theorem in [21] shows that right-regular sequence is asymptotically optimal, which explains why it is better than many other sequences and why optimal degree distributions found in numerical experiments tend to be right-regular.

Theorem 4.1. *(Oswald-Shokrollahi) Let (λ^n, ρ^n) be the right-regular sequence of degree distributions of rate $r \in (0, 1)$ obtained from the family $f^n(x) = x^n \in \mathcal{A}$.*

This sequence is asymptotically optimal with constant

$$\Delta = e^{\gamma},$$

where $\gamma = 0.57721566...$ *is the Euler constant.*

Oswald and Shokrollahi ([21]) also studied capacity-achieving sequences constructed from $f^{(n)} := \phi(x^n)$, where $\phi(x) \in \mathcal{A}$. They proved the following theorem.

Theorem 4.2. *(Oswald-Shokrollahi) If a sequence* (λ^n, ρ^n) *of rate* $r \in (0,1)$ *is generated by* $\phi(x^n)$*, where* $\phi \in \mathcal{A}$*, then*

$$\mu \leq \mu^{\phi} := (\int_0^1 \frac{\phi(x)}{x})^{-1}.$$

And they asked a natural question that whether the inequality is tight. We shall show in the following theorem ([14]) that the answer is yes under some condition and no otherwise.

Theorem 4.3. *If a sequence* (λ^n, ρ^n) *of rate* $r \in (0,1)$ *is generated by* $\phi(x^n)$*, where* $\phi(x) = \sum_{i \geq 1} \phi_i x^i \in \mathcal{A}$*, and if* i_0 *is the smallest integer such that* $\phi_{i_0} > 0$*, then*

$$\mu \geq max\{1, \frac{\mu^{\phi}}{i_0}\}.$$

Proof. For the most part of the proof, the dependencies on n are suppressed to simplify the notations. Thus $a_r^{(n)}$, $\epsilon(\lambda^{(n)}, \rho^{(n)})$ and $\delta(\lambda^{(n)}, \rho^{(n)})$ will be abbreviated by a_r, ϵ and δ respectively.

It was shown in [24][1] that

$$\epsilon = 1 - \frac{\delta}{1-r} \geq R(1-\delta) = a_r \int_0^{1-\delta} \rho,$$

where

$$R(x) = \frac{\int_0^x \rho}{\int_0^1 \rho}.$$

Applying the above to $\rho^{(n)} = \phi(x^n)$, we have

$$\begin{aligned}\frac{a_r \log r}{\log \epsilon} &\geq \frac{a_r \log r}{\log a_r \int_0^{1-\delta} \phi(x^n)} \\ &= \frac{n\frac{a_r}{n} \log r}{\log n + \log \frac{a_r}{n} + \log \int_0^{1-\delta} \phi(x^n)} \\ &= \frac{\frac{a_r}{n} \log r}{\frac{\log n}{n} + \frac{\log \frac{a_r}{n}}{n} + \frac{\log \frac{1}{n}}{n} + \frac{1}{n} \log \int_0^{(1-\delta)^n} \frac{\phi(t)}{t} t^{1/n} \mathrm{d}t}.\end{aligned}$$

Now let $n \to \infty$. Oswald and Shokrollahi showed in [21] that $\lim_{n\to\infty} \frac{a_r}{n} = \mu^\phi$. So the first three terms in the denominator of last expression above vanish. Therefore,

$$\mu \geq \lim_{n\to\infty} \frac{\mu^\phi \log r}{\frac{1}{n} \log \int_0^{(1-\delta)^n} \frac{\phi(t)}{t}}.$$

Since $\phi(x) \in \mathcal{A}$ the series $\phi(x) = \sum_{i\geq 1} \phi_i x^i$ converges for every $x \in [0,1]$ and all the ϕ_i are nonnegative. Therefore the convergence is uniform. If i_0 is the smallest integer such that $\phi_{i_0} > 0$, then

$$\begin{aligned}\int_0^{(1-\delta)^n} \frac{\phi(t)}{t} &= \int_0^{(1-\delta)^n} \sum_{i\geq i_0} \phi_i t^{i-1} \\ &= \sum_{i\geq i_0} \frac{1}{i} \phi_i ((1-\delta)^n)^i \\ &= (1-\delta)^{ni_0} (\frac{\phi_{i_0}}{i_0} + \frac{\phi_{i_0+1}}{i_0+1}(1-\delta)^n + \ldots).\end{aligned}$$

So,

$$\begin{aligned}\mu &\geq \lim_{n\to\infty} \frac{\mu^\phi \log r}{i_0 \log(1-\delta) + \frac{1}{n} \log(\frac{\phi_{i_0}}{i_0} + \frac{\phi_{i_0+1}}{i_0+1}(1-\delta)^n + \ldots)} \\ &= \frac{\mu^\phi}{i_0}\end{aligned}$$

since $1-\delta \to r$ as $n \to \infty$. Because we also have $\mu \geq 1$ the theorem is proved. □

A corollary follows immediately.

Corollary 4.1. *If a sequence (λ^n, ρ^n) of rate $r \in (0,1)$ is generated by $\phi(x^n)$,*

where $\phi(x) = \sum_{i \geq 1} \phi_i x^i \in \mathcal{A}$*, and if* $\phi_1 > 0$*, then*

$$\mu = \mu^{\phi} := (\int_0^1 \frac{\phi(x)}{x})^{-1}.$$

If $\phi_1 = 0$ consider the right-regular sequence constructed from $\rho^{(n)}(x) = x^n$ as in [24][21]. It was shown that for this sequence $\Delta = e^{\gamma}$, where $\gamma = 0.57721\ldots$ is the Euler constant, which implies that $\mu = 1$. Now let $\phi(x) = x^2$. Clearly the sequence constructed from $\phi(x^n) = x^{2n}$ is a subsequence of the right-regular sequence. Therefore the μ for this subsequence must be 1 as well. But the upper bound in Theorem 4.2 is $\mu^{\phi} = (\int_0^1 \frac{\phi(x)}{x})^{-1} = (\int_0^1 x)^{-1} = 2$, which obviously is not tight.

4.3 Conclusion

We reviewed the capacity-achieving sequences in this chapter and answered an open question raised by Shokrollahi and Oswald in [21]. For a sequence constructed from the sequence $\phi(x^n)$, where ϕ is a function satisfying special conditions, we can completely determine the convergence speed if ϕ satisfies an additional minor condition. There are still interesting open questions related to capacity-achieving sequences. For instance, are there any other capacity-achieving sequences (λ^n, ρ^n) of given code rate R for which $\lim_{x \to \infty} \Delta(\lambda^n, \rho^n) < e^{\gamma}$? In other words, is the right-regular sequence optimal in some sense? Interested readers can refer to [21] for more open questions.

Chapter 5

Enumerating Pseudo-codewords in Fundamental Cones

In this chapter we will discuss the performance of the message-passing iterative decoding (MPID) algorithms. Recall that MPID algorithms operate locally on Tanner graphs. Each node receives information from and sends information to its neighbors only. This feature makes the MPID algorithms fast but also results in non-optimality of the solutions. To further study the effects of these locally operating algorithms we need the following definition [10].

Definition 5.1. *An finite, unramified cover, or, simply, a cover of a graph* $G = (V, E)$ *is a graph* $\tilde{G} = (\tilde{V}, \tilde{E})$ *together with a projection* $\pi : \tilde{V} \to V$ *which maps the adjacent vertices of* $\tilde{G}$ *to adjacent vertices of* G *such that for each vertex* $v \in V$ *and each* $\tilde{v} \in \pi^{-1}(v)$*, the neighborhood* $\partial(\tilde{v})$ *of* $\tilde{v}$ *is mapped bijectively to* $\partial(v)$*. A cover is call an* M*-cover, where* M *is a positive integer, if* $|\pi^{-1}(v)| = M$ *for every vertex* v *in* V*.*

So the MPID algorithms can't distinguish a Tanner graph from its finite unramified covers, which have the same local properties. This can also be viewed as the competitions between codewords from the Tanner graph and those from all the finite covers. Therefore pseudo-codewords are defined as follows [10].

Definition 5.2. *Let* $C \subseteq \mathbb{F}_2^n$ *be a binary linear code of length* n *with Tanner graph* T *and let*

$$\tilde{\mathbf{a}} = (a_{(1,1)} : \ldots : a_{(1,M)}, \ldots, a_{(n,1)} : \ldots : a_{(n,M)})$$

be a codeword in the code $\tilde{C}$ *corresponding to some* M*-cover* $\tilde{T}$ *of* T*. The unscaled pseudo-codewords corresponding to* $\tilde{\mathbf{a}}$ *is the vector* $\mathbf{p}(\tilde{\mathbf{a}}) := (p_1, \ldots, p_n)$ *where* $p_i = \sum_{k=1}^{M} a_{(i,k)}$*. The normalize pseudo-codeword corresponding to* $\tilde{\mathbf{a}}$ *is the vector* $\omega(\tilde{\mathbf{a}}) = \frac{1}{M}\mathbf{p}(\tilde{\mathbf{a}})$*.*

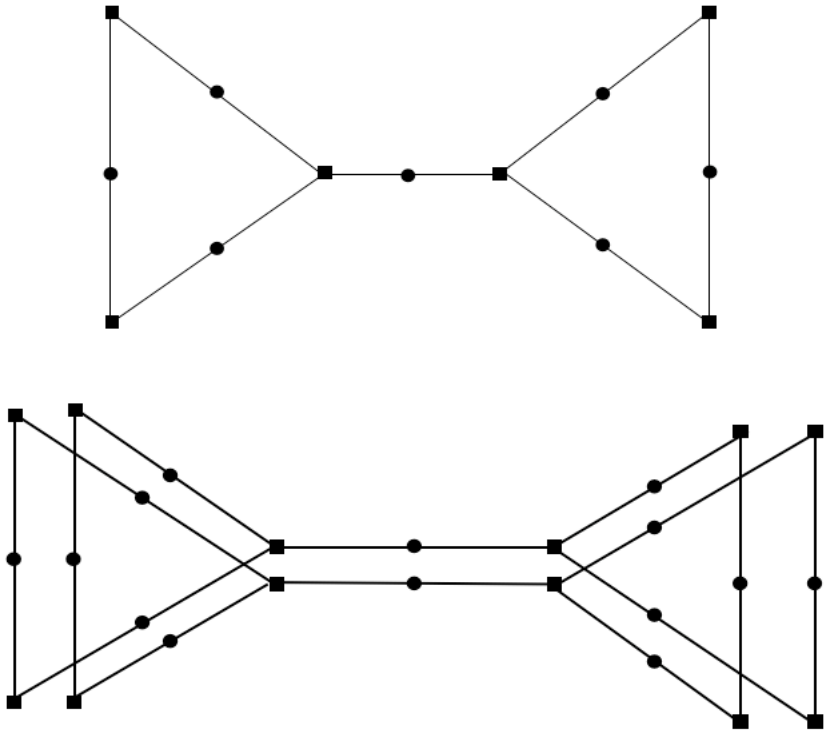

Figure 5.1. A Tanner graph G and a 2-cover of G, where the dots and squares are bit nodes and check nodes respectively.

Figure 5.1 shows a Tanner graph G together with a 2-cover of it.

To understand pseudo-codewords, Koetter and Vontobel proposed the concept of fundamental cone [11], which is the smallest cone containing all the pseudo-codewords and is defined by a set of inequalities [10].

Definition 5.3. *Let* $H = (h_{ij})$ *be an* $m \times n$ *parity-check matrix. The fundamental cone* $\mathcal{K}(H)$ *of* H *is the set of vectors* $\mathbf{v} = (v_1, \ldots, v_n) \in \mathbb{R}^n$ *such that, for all*

$1 \leq j \leq n$ *and* $1 \leq i \leq m$, *we have*

$$v_j \geq 0$$

and

$$\sum_{j' \neq j} h_{ij'} v'_j \geq h_{ij} v_j.$$

5.1 Pseudo-codewords and Edge Zeta Functions

Given a parity-check matrix H (hence a Tanner graph), it was shown in [10] that there existed a rational function whose monomials capture the information of all pseudo-codewords. If the code defined by H is a cycle code, meaning that each bit node has degree two, then the monomials exactly correspond to pseudo-codewords. If the code is general then pseudo-codewords are captured by some monomials satisfying special conditions. We will introduce this result in this section.

Let X be an undirected graph with edge set $\{e_1, \ldots, e_n\}$. For each path $\Gamma = (e_{i_1}, \ldots, e_{i_k})$, define the monomial of Γ to be

$$g(\Gamma) := u_{i_1} \cdots u_{i_k},$$

where u_{i_j}'s are variables.

A path $\Gamma = (e_{i_1}, \ldots, e_{i_k})$ is called a cycle if it starts and ends at the same vertex. A cycle is said to be backtrackless if $e_{i_j} \neq e_{i_{j+1}}$ for $j = 1, \ldots, k-1$. And it is said to be tailless if $e_{i_1} \neq e_{i_k}$. Moreover a cycle Γ is said to primitive if there is no cycle Ω on X such that $\Gamma = \Omega^r$ for some $r \geq 2$, i.e. such that Γ is obtained by following Ω r times. We say that a cycle $\Delta = (e_{j_1}, \ldots, e_{j_k})$ is equivalent to Γ if there exists an integer t such that $e_{i_s} = e_{j_{s+t}}$ for all $s = 1, \ldots, k$, where the sum in subindices are taken modulo k.

The edge zeta function of X is defined by

$$Z_X(u_1, \ldots, u_n) = \prod_{[\Gamma]} (1 - g(\Gamma))^{-1},$$

where $[\Gamma]$ runs though all equivalent classes of backtrackless, tailless and primitive

cycles in X.

Stark and Terras showed in [27] that the above edge zeta function is a rational function. Let $\tilde{X}$ be a directed graph derived from X by adding another n edges which are in the opposite direction as the original ones. The directed edge matrix of $\tilde{X}$ is the $2n \times 2n$ matrix $M = (m_{ij})$, where

$$m_{ij} = \begin{cases} 1 & \text{if } (e_i, e_j) \text{ is a backtrackless path,} \\ 0 & \text{otherwise.} \end{cases}$$

Theorem 5.1. *(Stark-Terras)*

$$Z_X(u_1, \ldots, u_n) = \frac{1}{det(I - MU)} = \frac{1}{det(I - UM)},$$

where I is the $2n \times 2n$ identity matrix and $U = diag(u_1, \ldots, u_n, u_1, \ldots, u_n)$.

If C is a cycle code defined by parity-check matrix H, then each bit node can be viewed as an edge connecting the two check nodes adjacent to it. The normal graph $N(H)$ is defined as a graph whose vertices are the check nodes in the Tanner graph and whose edges correspond to the bit nodes. Then the following theorem [9] [10] built the connection between pseudo-codewords and the edge zeta function of the normal graph.

Theorem 5.2. *(Koetter-Li-Vontobel-Walker)*

Let C be a cycle code defined by a parity-check matrix H having normal graph $N := N(H)$. Then $(p_1, \ldots, p_n)$ is an unscaled pseudo-codeword if and only if the monomial $u_1^{p_1} u_2^{p_2} \cdots u_n^{p_n}$ occurs in the power series expansion of $Z_N(u_1, \ldots, u_n)$ with nonzero coefficient.

For a general LDPC code there is a similar result in [10]. But now only some monomials in the edge zeta function satisfying a special condition correspond to pseudo-codewords.

Theorem 5.3. *(Koetter-Li-Vontobel-Walker) Let C be an LDPC code of length n with parity-check matrix H and Tanner graph T. Assume that every bit node has even degree. Then $(p_1, \ldots, p_n)$ is an unscaled pseudo-codeword if and only if in the*

edge zeta function Z_T there occurs a monomial in which the degree of the variables representing the edges incident to the ith bit node is p_i for $1 \leq i \leq n$.

5.2 Pseudo-codewords of General LDPC codes

In theorem 5.3, although the edge zeta function captures all the pseudo-codewords there are redundant monomials that do not correspond to pseudo-codewords. In this section we want to find a rational function such that there is a one-to-one correspondence between the pseudo-codewords and the monomials in the power series expansion of this function. Actually we shall prove that, given an LDPC code C with parity-check matrix H,

$$Z_C(u_1, \ldots, u_n) = \sum_{\mathbf{p}=(p_1,\ldots,p_n) \text{ pseudo-codeword of } C} \mathbf{u}^{\mathbf{p}}$$

is a rational function, where $\mathbf{u}^{\mathbf{p}} := u_1^{p_1} \cdots u_n^{p_n}$. Moreover, we shall give a simpler rational function which also enumerates the pseudo-codewords of C, but allows the coefficients of the monomials to be greater than 1. This work can be found in [13].

5.2.1 Preliminaries

In this section we will review some known results on the structure of cones, which will be used in the proof of the main theorem. We follow the terminologies in [3].

A **pointed cone** $\mathcal{K} \subseteq \mathbb{R}^d$ is a set of the form

$$\mathcal{K} = \{\mathbf{v} + \lambda_1 \mathbf{w}_1 + \lambda_2 \mathbf{w}_2 + \cdots + \lambda_m \mathbf{w}_m : \lambda_i \geq 0, i = 1, \ldots, m\},$$

where $\mathbf{v}, \mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_m \in \mathbb{R}^d$ are such that there exists a hyperplane H for which $H \cap \mathcal{K} = \{\mathbf{v}\}$, in other words, $\mathcal{K} \backslash \{\mathbf{v}\}$ lies strictly on one side of H. The vector $\mathbf{v}$ is called the **apex** of $\mathcal{K}$, and $\mathbf{w}_k$ is a generator of $\mathcal{K}$ if $\mathbf{w}_k = \sum_{i=1}^m \lambda_i \mathbf{w}_i (\lambda_i \geq 0)$ implies that $\lambda_i = 0$ for all $i \neq k, 1 \leq i \leq m$. The cone is **rational** if $\mathbf{v}, \mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_m \in \mathbb{Q}^d$, in which case we may choose $\mathbf{w}_1, \ldots, \mathbf{w}_m \in \mathbb{Z}^d$ by clearing the denominators. The **dimension** of $\mathcal{K}$ is the dimension of the affine space spanned by $\mathcal{K}$; if $\mathcal{K}$ is of

dimension d, we call it a d-cone. The d-cone $\mathcal{K}$ is **simplicial** if $\mathcal{K}$ has precisely d linearly independent generators.

A collection T of simplicial d-cones is a **triangulation** of the d-cone $\mathcal{K}$ if it satisfies:
(1) $\mathcal{K} = \bigcup_{\mathcal{S} \in T} \mathcal{S}$; and
(2) For any $\mathcal{S}_1, \mathcal{S}_2 \in T$, $\mathcal{S}_1 \cap \mathcal{S}_2$ is a face common to both $\mathcal{S}_1$ and $\mathcal{S}_2$.

We say that $\mathcal{K}$ can be **triangulated using no new generators** if there exists a triangulation T such that the generators of any $\mathcal{S} \in T$ are generators of $\mathcal{K}$. The following is an important theorem on triangulating cones.

Theorem 5.4. *([3], p. 20) Any pointed cone can be triangulated into simplicial cones using no new generators.*

The next theorem shows the "double-description" property of cones. In other words, every cone can be defined by either a set of generators or a set of linear inequalities.

Theorem 5.5. *([28], p. 30) A Cone $\mathcal{K} \subseteq \mathbb{R}^d$ is a finitely generated combination of vectors if and only if it is a finite intersection of closed linear halfspaces.*

Our fundamental cone $\mathcal{K}(H)$ is defined by inequalities derived from the check equations as follows. The ith check equation $\sum_j h_{ij} x_j = 0$ gives rise to n inequalities

$$\sum_{j \neq l} h_{ij} x_j \geq h_{il} x_l, \quad \text{for } 1 \leq l \leq n,$$

and the cone consists of points $(x_1, ..., x_n) \in \mathbb{R}^n$ satisfying all $x_j \geq 0$ and all the inequalities above. By Theorem 5.5 it is a pointed cone with apex at the origin. Its properties are summarized in

Theorem 5.6. *The fundamental cone $\mathcal{K}(H)$ is a pointed rational cone with apex at the origin, it is generated by finitely many pseudo-codewords, and it can be triangulated into simplicial cones using no new generators.*

Proof. The generators of the fundamental cone lie on its 1-dimensional boundaries, which are the intersections of closed linear half-spaces defined by linear

equations with integral coefficients, hence the cone is rational. Moreover, each 1-dimensional boundary contains pseudo-codewords, for instance, the lattice points with all even components; choose as generators a nonzero pseudo-codeword from each 1-dimensional boundary. The third assertion follows from Theorem 5.4. □

5.2.2 Generating Functions of Rational Cones

The generating function of a subset $S \subset \mathbb{R}^d$ is the sum of monomials recording the lattice points in S:

$$\sigma_S(\mathbf{z}) = \sigma_S(z_1, z_2, \ldots, z_d) := \sum_{\mathbf{m} \in S \cap \mathbb{Z}^d} \mathbf{z}^{\mathbf{m}}.$$

The following theorem describes how to enumerate the lattice points in a simplicial cone.

Theorem 5.7. *([3], p. 62) Suppose*

$$\mathcal{K} := \{\lambda_1 \mathbf{w}_1 + \lambda_2 \mathbf{w}_2 + \cdots + \lambda_d \mathbf{w}_d : \lambda_i \geq 0, i = 1, \ldots, d\}$$

is a simplicial rational d-cone, where $\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_d \in \mathbb{Z}^d$. *Then for* $\mathbf{v} \in \mathbb{R}^d$, *the generating function* $\sigma_{\mathbf{v}+\mathcal{K}}$ *of the shifted cone* $\mathbf{v} + \mathcal{K}$ *is the rational function*

$$\sigma_{\mathbf{v}+\mathcal{K}}(\mathbf{z}) = \frac{\sigma_{\mathbf{v}+\Pi}(\mathbf{z})}{(1 - \mathbf{z}^{\mathbf{w}_1})(1 - \mathbf{z}^{\mathbf{w}_2}) \cdots (1 - \mathbf{z}^{\mathbf{w}_d})},$$

where Π *is the (half-open) fundamental parallelepiped of* $\mathcal{K}$*:*

$$\Pi := \{\lambda_1 \mathbf{w}_1 + \lambda_2 \mathbf{w}_2 + \cdots + \lambda_d \mathbf{w}_d : 0 \leq \lambda_1, \lambda_2, \ldots, \lambda_d < 1\}.$$

By Theorem 5.4 and the fact that the intersection of simplicial cones in a triangulation is again a simplicial cone, the following consequence is evident.

Corollary 5.1. *([3], p. 63) The generating function of a finitely generated pointed rational cone is a rational function.*

Here is a simple example of enumerating lattice points in a two-dimensional cone.

Example 4.1 ([3], Example 3.4, p. 60) Consider the two-dimensional cone

$$\mathcal{K} := \{\lambda_1(1,1) + \lambda_2(-2,3) : \lambda_1, \lambda_2 \geq 0\} \subset \mathbb{R}^2$$

as shown in Fig. 5.2. Its fundamental parallelogram Π is

$$\Pi := \{\lambda_1(1,1) + \lambda_2(-2,3) : 0 \leq \lambda_1, \lambda_2 < 1\} \subset \mathbb{R}^2\}.$$

The cone $\mathcal{K}$ can be exactly covered by the translations of Π by nonnegative linear combinations of $(1,1)$ and $(-2,3)$. The lattice points in Π are $(0,0)$,$(0,1)$,$(0,2)$,$(-1,2)$ and $(-1,3)$. Hence

$$\sigma_{\mathcal{K}}(x,y) = \frac{1 + y + y^2 + x^{-1}y^2 + x^{-1}y^3}{(1-xy)(1-x^{-2}y^3)}.$$

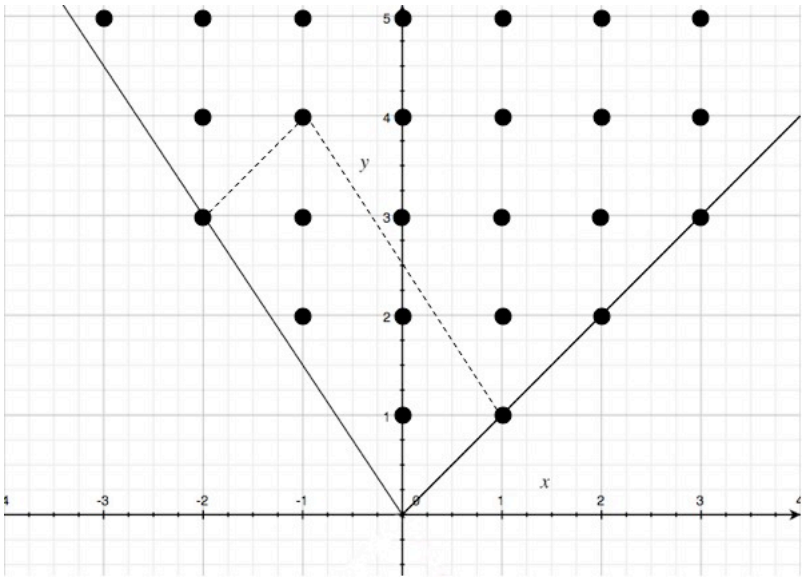

Figure 5.2. The cone with generators (1,1) and (-2,3)

5.2.3 Generating Functions of Pseudo-codewords

We will use the following characterization of pseudo-codewords given in [10].

Theorem 5.8. *(Koetter-Li-Vontobel-Walker) Let* $\mathbf{p} = (p_1, \ldots, p_n)$ *be a vector with integral components. Then the following two statements are equivalent:*
(1) $\mathbf{p}$ *is an unscaled pseudo-codeword of* C*;*
(2) $\mathbf{p} \in \mathcal{K}(H)$ *and* $H\mathbf{p}^t \equiv \mathbf{0} \pmod 2$.

This theorem shows that pseudo-codewords of C are just the lattice points in $\mathcal{K}(H)$ congruent to codewords modulo 2. We prove the first main result of this section by adapting the proof of Theorem 5.7.

Theorem 5.9. *The zeta function*

$$Z_C(u_1, ..., u_n) = \sum_{\mathbf{p}=(p_1,...,p_n)\ \text{pseudo-codeword of}\ C} \mathbf{u}^{\mathbf{p}}$$

is rational.

Proof. By Theorem 5.6 the fundamental cone is generated by finitely many pseudo-codewords. We only need to show that the generating function of pseudo-codewords in a simplicial rational cone is rational.

Assume that a simplicial cone $\mathcal{K}$ is generated by the pseudo-codewords $\mathbf{w}_1$,$\mathbf{w}_2$, ..., $\mathbf{w}_d$ and that all generators have even components. (Multiply them by 2 if necessary). We claim that every pseudo-codeword $\mathbf{p}$ in $\mathcal{K}$ can be uniquely written as

$$\mathbf{p} = \mathbf{p_0} + \lambda_1\mathbf{w}_1 + \lambda_2\mathbf{w}_2 + \cdots + \lambda_d\mathbf{w}_d \tag{5.1}$$

for some pseudo-codeword $\mathbf{p_0}$ in the fundamental parallelepiped Π of $\mathcal{K}$ and some non-negative integers $\lambda_1, \lambda_2, \ldots, \lambda_d$. Indeed, it follows from the definition of a simplicial cone given in §4.1 that we may write

$$\mathbf{p} = \alpha_1\mathbf{w}_1 + \alpha_2\mathbf{w}_2 + \cdots + \alpha_d\mathbf{w}_d$$

uniquely for some nonnegative real numbers $\alpha_1, \alpha_2, \ldots, \alpha_d$. Let $\lfloor x \rfloor$ and $\{x\}$ denote the integral part and fractional part of a non-negative real number x, respectively. Then

$$\mathbf{p} = \mathbf{p_0} + \lfloor\alpha_1\rfloor\mathbf{w}_1 + \lfloor\alpha_2\rfloor\mathbf{w}_2 + \cdots + \lfloor\alpha_d\rfloor\mathbf{w}_d,$$

where $\mathbf{p_0} = \{\alpha_1\}\mathbf{w}_1 + \{\alpha_2\}\mathbf{w}_2 + \cdots + \{\alpha_d\}\mathbf{w}_d$ is a lattice point since it is the difference of $\mathbf{p}$ and a lattice point. Clearly $\mathbf{p_0}$ lies in the fundamental parallelepiped Π since $0 \leq \{\alpha_i\} < 1$. Again, this representation is unique. Because all the generators have even components, $\mathbf{p}$ is congruent to $\mathbf{p_0}$ modulo 2. Therefore, by Theorem 5.8, $\mathbf{p_0}$ is a pseudo-codeword because $\mathbf{p}$ is. This proves the claim with $\lambda_i = \lfloor \alpha_i \rfloor$ for $i = 1, 2, \ldots, d$. The above unique representation of each pseudo-codeword in $\mathcal{K}$ allows us to express the generating function $\sigma_{\mathcal{K}}(\mathbf{z})$ of the pseudo-codewords in a simplicial (rational) cone $\mathcal{K}$ as

$$\begin{aligned}
\sigma_{\mathcal{K}}(\mathbf{u}) &:= \sum_{\mathbf{p} \ \text{ pseudo-codeword in } \mathcal{K}} \mathbf{u}^{\mathbf{p}} \\
&= \Big(\sum_{\mathbf{p_0} \ \text{pseudo-codeword in } \Pi} \mathbf{u}^{\mathbf{p_0}} \Big) \Big(\sum_{\lambda_1 \geq 0} \mathbf{u}^{\lambda_1 \mathbf{w}_1} \Big) \cdots \Big(\sum_{\lambda_d \geq 0} \mathbf{u}^{\lambda_d \mathbf{w}_d} \Big) \\
&= \frac{\sigma_{\Pi}(\mathbf{u})}{(1 - \mathbf{u}^{\mathbf{w}_1})(1 - \mathbf{u}^{\mathbf{w}_2}) \cdots (1 - \mathbf{u}^{\mathbf{w}_d})},
\end{aligned}$$

where $\sigma_{\Pi}(\mathbf{u}) = \sum_{\mathbf{p_0}} \mathbf{u}^{\mathbf{p_0}}$ is a finite polynomial summing over the pseudo-codewords in the fundamental parallelepiped Π. Therefore $\sigma_{\mathcal{K}}(\mathbf{u})$ is a rational function.

By Theorem 5.6, the fundamental cone can be triangulated into simplicial cones. Since the intersections of simplicial cones in a triangulation are again simplicial, the generating function of the pseudo-codewords in $\mathcal{K}(H)$, which is Z_C, can be obtained using inclusion-exclusion. It is rational because it is the sum and difference of rational functions. □

From the proof we can see that if we do not require that each pseudo-codeword appears exactly once in the generating function, that is, the coefficient of each monomial equal to 1, we do not have to decompose the fundamental cone into simplicial cones, which really saves a lot of time.

5.2.4 An Example

Consider the parity-check matrix

$$H = \begin{pmatrix} 1 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 \end{pmatrix}.$$

It defines the code

$$C(H) = \{(0,0,0,0),(1,0,1,0),(1,1,0,1),(0,1,1,1)\},$$

and the fundamental cone $\mathcal{K}(H)$ is generated by the following 5 vectors with even components:

$$\mathbf{w}_1 = (2,2,0,2),$$

$$\mathbf{w}_2 = (2,4,2,0),$$

$$\mathbf{w}_3 = (2,0,2,0),$$

$$\mathbf{w}_4 = (2,0,2,4),$$

$$\mathbf{w}_5 = (0,2,2,2).$$

(These generators are found by the software package POLYMAKE.) A triangulation of $\mathcal{K}(H)$ is

$$T(\mathcal{K}(H)) = \{\mathcal{S}_1, \mathcal{S}_2\},$$

where

$$\mathcal{S}_1 = \text{cone generated by}\{\mathbf{w}_1, \mathbf{w}_2, \mathbf{w}_3, \mathbf{w}_5\}$$

and

$$\mathcal{S}_2 = \text{cone generated by}\{\mathbf{w}_1, \mathbf{w}_3, \mathbf{w}_4, \mathbf{w}_5\}.$$

Let $\sigma_S(\mathbf{u})$ be the generating function of pseudo-codewords in the region S and Π_C be the fundamental parallelepiped of the cone C. then

$$\begin{aligned}\sigma_{\mathcal{K}(H)}(\mathbf{u}) &= \frac{\sigma_{\Pi_{\mathcal{S}_1}}(\mathbf{u})}{(1-\mathbf{u}^{\mathbf{w}_1})(1-\mathbf{u}^{\mathbf{w}_2})(1-\mathbf{u}^{\mathbf{w}_3})(1-\mathbf{u}^{\mathbf{w}_5})} \\ &\quad + \frac{\sigma_{\Pi_{\mathcal{S}_2}}(\mathbf{u})}{(1-\mathbf{u}^{\mathbf{w}_1})(1-\mathbf{u}^{\mathbf{w}_3})(1-\mathbf{u}^{\mathbf{w}_4})(1-\mathbf{u}^{\mathbf{w}_5})} \\ &\quad - \frac{\sigma_{\Pi_{\mathcal{S}_1\cap\mathcal{S}_2}}(\mathbf{u})}{(1-\mathbf{u}^{\mathbf{w}_1})(1-\mathbf{u}^{\mathbf{w}_3})(1-\mathbf{u}^{\mathbf{w}_5})},\end{aligned}$$

where

$$\begin{aligned}\sigma_{\Pi_{\mathcal{S}_1}}(\mathbf{u}) &= 1+u_1u_3+u_2u_3u_4+u_1u_2u_4+u_1u_2^2u_3+u_1u_2u_3^2u_4+u_1^2u_2u_3u_4\\ &\quad +u_1u_2^2u_3u_4^2+u_1^2u_2^2u_3^2+u_1u_2^3u_3^2u_4+u_1^2u_2^3u_3u_4+u_1^2u_2^2u_3^2u_4^2\\ &\quad +u_1^2u_2^3u_3^3u_4+u_1^3u_2^3u_3^2u_4+u_1^2u_2^4u_3^2u_4^2+u_1^3u_2^4u_3^3u_4^2,\end{aligned}$$

$$\begin{aligned}\sigma_{\Pi_{\mathcal{S}_2}}(\mathbf{u}) &= 1+u_1u_3+u_1u_2u_4+u_2u_3u_4+u_1u_3u_4^2\\ &\quad +u_1u_2u_3^2u_4+u_1^2u_2u_3u_4+u_1u_2^2u_3u_4^2+u_1^2u_3^2u_4^2\\ &\quad +u_1^2u_1u_3u_4^3+u_1u_2u_3^2u_4^3+u_1^2u_2^2u_3^2u_4^2\\ &\quad +u_1^2u_2u_3^3u_4^3+u_1^3u_2u_3^2u_4^3+u_1^2u_2^2u_3^2u_4^4+u_1^3u_2^2u_3^3u_4^4,\end{aligned}$$

and

$$\begin{aligned}\sigma_{\Pi_{\mathcal{S}_1\cap\mathcal{S}_2}}(\mathbf{u}) &= 1+u_1u_3+u_1u_2u_4+u_2u_3u_4\\ &\quad +u_1u_2u_3^2u_4+u_1^2u_2u_3u_4+u_1u_2^2u_3u_4^2+u_1^2u_2^2u_3^2u_4^2.\end{aligned}$$

The initial terms in the Taylor series of $Z_{C(H)}(\mathbf{u})$ are

$$\begin{aligned}Z_{C(H)}(\mathbf{u}) &= 1+u_1u_3+u_1u_2u_4+u_2u_3u_4+u_1u_2^2u_3+u_1u_3u_4^2+u_1^2u_3^2\\ &\quad +u_1^2u_2u_3u_4+u_1u_2u_3^2u_4+u_1u_2^2u_3u_4^2+u_1^2u_2^2u_3^2+u_1^2u_2^2u_4^2\\ &\quad +u_1^2u_3^2u_4^2+u_2^2u_3^2u_4^2+u_1^3u_3^3\\ &\quad +u_1^2u_2^3u_3u_4+u_1^3u_2u_3^2u_4+u_1u_2^3u_3^2u_4+u_1^2u_2u_3^3u_4+u_1^2u_2u_3u_4^3+u_1u_2u_3^2u_4^3\\ &\quad +u_1^2u_2^4u_3^2+u_1^2u_3^2u_4^4+u_1^3u_2^2u_3^3+u_1^3u_3^3u_4^2+u_1^3u_2^2u_3u_4^2+u_1u_2^2u_3^3u_4^2\\ &\quad +u_1^2u_2^2u_3^2u_4^2+u_1^4u_3^4+\ldots\end{aligned}$$

5.2.5 Another Zeta Function Enumerating Pseudo-codewords

For a general LDPC code, it is usually complicated to triangulate the fundamental cone, especially when the number of generators of the cone is much greater than the dimension of the cone. So it is desirable to find another rational function that not only records the pseudo-codewords but also has a simpler form. The proof of

Theorem 5.9 shows that if we only want to enumerate pseudo-codewords and have no restrictions on the coefficients of the monomials, then the rational function in the following theorem will serve our purpose.

Theorem 5.10. *Suppose that the fundamental cone $\mathcal{K}(H)$ of C is generated by the pseudo-codewords $\mathbf{w}_1, \ldots, \mathbf{w}_m$ with even components. Then the rational function*

$$Z'_C(u_1, \ldots, u_n) = \frac{\sigma_\Pi(u_1, \ldots, u_n)}{(1 - \mathbf{u}^{\mathbf{w}_1}) \cdots (1 - \mathbf{u}^{\mathbf{w}_m})}$$

enumerates all the pseudo-codewords of C, where $\sigma_\Pi(u_1, \ldots, u_n)$ enumerates the pseudo-codewords in

$$\Pi = \{\lambda_1 \mathbf{w}_1 + \cdots + \lambda_m \mathbf{w}_m : 0 \leq \lambda_i < 1 \text{ for } 1 \leq i \leq m\}.$$

This is because (5.1) still holds, although the expression may not be unique. In fact, the number of different representations of $\mathbf{p}$ is the coefficient of $\mathbf{u}^{\mathbf{p}}$ in Z'_C.
Example 4.2 Consider again the code studied in §5.2.4. We compute $Z'_{C(H)}$. There are 52 pseudo-codewords in $\Pi = \{\lambda_1 \mathbf{w}_1 + \cdots + \lambda_5 \mathbf{w}_5 : 0 \leq \lambda_i < 1 \text{ for } 1 \leq i \leq 5\}$, and

$$\begin{aligned} \sigma_\Pi(\mathbf{u}) &= 1 + u_1u_3 + u_2u_3u_4 + u_1u_2u_4 + u_1u_3u_4^2 + \cdots \\ &\quad + u_1^5u_2^6u_3^5u_4^4 + u_1^5u_2^6u_3^5u_4^6 + u_1^6u_2^5u_3^5u_4^5 + u_1^6u_2^6u_3^6u_4^6. \end{aligned}$$

Thus,

$$\begin{aligned} Z'_{C(H)}(\mathbf{u}) &= \frac{\sigma_\Pi(\mathbf{u})}{(1 - \mathbf{u}^{\mathbf{w}_1}) \cdots (1 - \mathbf{u}^{\mathbf{w}_5})} \\ &= 1 + u_1u_3 + u_1u_2u_4 + u_2u_3u_4 + u_1u_2^2u_3 + u_1u_3u_4^2 + u_1^2u_3^2 + \cdots \\ &+ 2u_1^2u_2^3u_3^3u_4^3 + 2u_1^4u_2^3u_3^3u_4^3 + 2u_1^3u_2^3u_3^4u_4^3 + \cdots \\ &+ 3u_1^4u_2^5u_3^3u_4^3 + 3u_1^5u_2^5u_3^4u_4^3 + 3u_1^4u_2^5u_3^5u_4^3 + \cdots . \end{aligned}$$

The monomials occurring in $Z_{C(H)}$ agree with those in $Z'_{C(H)}$. Listed above are the lowest degree monomials where the coefficients are more than 1.

5.3 Generators of Some Fundamental Cones

In both Theorem 5.9 and Theorem 5.10 the numerators are the generating functions of pseudo-codewords in a finite region, which are relatively easy to find. And the denominators, which record information of generators of the fundamental cones, describe how they will grow along different directions to reach all other pseudo-codewords. Therefore it is important to find these generators in order to compute the zeta functions. Given a cone defined by a set of inequalities, there are some classic algorithms, e.g. Fourier-Motzkin elimination (page 37-39, [28]), double description method ([20][7]) and Chernikov algorithm ([5]), to find the generators. But all these algorithms are slow when the parity-check matrix is large. For some fundamental cones with special structures, it is possible to find the generators in $O(1)$. We shall study three kinds of special cones in this section.

5.3.1 $H = (1, 1, \ldots, 1)$

Assume that the parity check matrix H has only one row of length N, whose entries are all ones. Let

$$S = \{(1, 1, 0, 0, \ldots, 0) \text{ and all its permutations}\}.$$

In other words, S consists of all vectors that have two and only two ones and zeros elsewhere. We shall show that S generates $\mathcal{K}(H)$. The proof is two-fold and will be an immediate conclusion from the following two lemmas.

Lemma 5.1. *Each $v \in S$ is an generator of $\mathcal{K}(H)$.*

Proof. We only need to prove the lemma for $v = (1, 1, 0, \ldots, 0)$.

If $v = \lambda\mathbf{a} + (1-\lambda)\mathbf{b}$ for some $\mathbf{a}, \mathbf{b} \in \mathcal{K}(H)$ and some $0 \leq \lambda \leq 1$, then we have

$$\begin{cases} \lambda a_1 + (1-\lambda)b_1 = 1, \\ \lambda a_2 + (1-\lambda)b_2 = 1, \\ \lambda a_3 + (1-\lambda)b_3 = 0, \\ \cdots \\ \lambda a_N + (1-\lambda)b_N = 0. \end{cases} \tag{5.2}$$

Since $a_i \geq 0, b_i \geq 0, 0 \leq \lambda \leq 1$, the 3rd to the Nth equations in (5.2) imply that $a_i = b_i = 0$ for any $3 \leq i \leq N$. Therefore

$$\mathbf{a} = (a_1, a_2, 0, \ldots, 0)$$

$$\mathbf{b} = (b_1, b_2, 0, \ldots, 0).$$

By the definition of fundamental cone we also know that $A\mathbf{a} \geq 0, A\mathbf{b} \geq 0$, where

$$A = \begin{pmatrix} -1 & 1 & 1 & \cdots & 1 \\ 1 & -1 & 1 & \cdots & 1 \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ 1 & 1 & \cdots & 1 & -1 \end{pmatrix}. \tag{5.3}$$

Thus $a_1 = a_2$ and $b_1 = b_2$, i.e. $\mathbf{a} = c_1\mathbf{v}$ for some $c_1 \geq 0$ and $\mathbf{b} = c_2\mathbf{v}$ for some $c_2 \geq 0$. This shows that $\mathbf{v}$ is an generator of $\mathcal{K}(H)$. □

Lemma 5.2. *Every* $\mathbf{p} = (p_1, p_2, \ldots, p_N) \in \mathcal{K}(H)$ *is a nonnegative linear combination of the vectors in* S.

Proof. We shall only prove the case for rational $\mathbf{p}$. Then by choosing a rational sequence and taking the limit the conclusion is clearly true for irrational $\mathbf{p}$. To prove for rational $\mathbf{p}$ it suffices to consider lattice point $\mathbf{p}$.

Assume that $p_i \geq p_j \geq p_k$ for any $k \neq i, j$, i.e. p_i and p_j are two largest entries in $\mathbf{p}$. Let $\mathbf{v}_{ij}$ be a vector in S whose only two nonzero entries are ones at the ith

and jth position respectively and

$$\mathbf{p}' = \mathbf{p} - \mathbf{v}_{ij} = (\cdots, p_i - 1, \cdots, p_j - 1, \cdots). \tag{5.4}$$

We want to show that $\mathbf{p}'$ will be still in $\mathcal{K}(H)$ with one exception, in which case $\mathbf{p}$ is still a nonnegative linear combination of the vectors in S. We shall study the following two cases.

(1) $p'_i = p_i - 1$ is still the largest entry in $\mathbf{p}'$, i.e. $p'_i \geq p'_l$ for $1 \leq l \leq N$.

We have

$$\sum_{l \neq i} p'_l \geq p'_i \tag{5.5}$$

since $\sum_{l \neq i} p_l \geq p_i$ and we simply subtract 1 from both side to get (5.5). Since p'_i is the largest entry in $\mathbf{p}$ the other inequalities that define $\mathcal{K}(H)$ will be satisfied. So in this case $\mathbf{p}' \in \mathcal{K}(H)$.

(2) $p'_k > p'_i$ for some $k \neq i$. Note that $k \neq j$ since $p'_i \geq p'_j$. So $p'_k = p_k$. Therefore we have $p_k > p'_i = p_i - 1 \geq p_k - 1$, which implies that $p_i = p_k$. So the inequality

$$\sum_{l \neq k} p'_l = \sum_{l \neq k,i,j} p_l + (p_i - 1) + (p_j - 1) \geq p'_k = p_k$$

will be satisfied unless $p_l = 0, (l \neq i, j, k)$ and $p_i = p_j = p_k = 1$, in which case

$$\mathbf{p} = \frac{1}{2}(\mathbf{v}_{ij} + \mathbf{v}_{ik} + \mathbf{v}_{jk}). \tag{5.6}$$

Now we have a new vector $\mathbf{p}' \in \mathcal{K}(H)$. Let $\mathbf{p}'$ be $\mathbf{p}$ in (5.4) and repeat this process until we have (5.6) for some i, j, k. So $\mathbf{p}$ is a nonnegative linear combination of vectors in S. □

Theorem 5.11. *S generates $\mathcal{K}(H)$.*

Proof. This is an immediate result from Lemma 5.1 and Lemma 5.2. □

5.3.2 H consisting of all permutations of the first row

Without loss of generality, assume that the first row of H is

$$(\underbrace{1,1,\ldots,1}_{M},\underbrace{0,0,\ldots,0}_{N}), N>0.$$

The other rows of H are all permutations of the first row. Then a vector $\mathbf{p} = (p_1, p_2, \ldots, p_{N+M}) \in \mathcal{K}(H)$ if and only if

$$\sum_{l=1}^{M-1} p_{k_l} \geq p_{k_M} \tag{5.7}$$

for any M entries $p_{k_1}, \ldots, p_{k_M}$ of $\mathbf{p}$. Let S be the set of the following vectors and all their permutations.

$$\begin{aligned}
\mathbf{v}_{1,1} = \cdots = \mathbf{v}_{1,N} &= (\underbrace{1,1,\ldots,1}_{N+2},\underbrace{0,0,\ldots,0}_{M-2})\\
\mathbf{v}_{2,1} &= (\underbrace{1,1,\ldots,1}_{N+2},2,\underbrace{0,0,\ldots,0}_{M-3})\\
\mathbf{v}_{2,2} &= (\underbrace{1,1,\ldots,1}_{N+1},2,2,\underbrace{0,0,\ldots,0}_{M-3})\\
&\cdots\\
\mathbf{v}_{2,N} &= (1,1,1,\underbrace{2,2,\ldots,2}_{N},\underbrace{0,0,\ldots,0}_{M-3})\\
\mathbf{v}_{3,1} &= (\underbrace{1,1,\ldots,1}_{N+3},3,\underbrace{0,0,\ldots,0}_{M-4})\\
\mathbf{v}_{3,2} &= (\underbrace{1,1,\ldots,1}_{N+2},3,3,\underbrace{0,0,\ldots,0}_{M-4})\\
&\cdots\\
\mathbf{v}_{3,N} &= (1,1,1,1,\underbrace{3,3,\ldots,3}_{N},\underbrace{0,0,\ldots,0}_{M-4})\\
&\cdots\\
\mathbf{v}_{M-1,1} &= (\underbrace{1,1,\ldots,1}_{M+N-1},M-1)\\
\mathbf{v}_{M-1,2} &= (\underbrace{1,1,\ldots,1}_{M+N-2},M-1,M-1)\\
&\cdots
\end{aligned}$$

$$\mathbf{v}_{M-1,N} = (\underbrace{1,1,\ldots,1}_{M},\underbrace{M-1,M-1,\ldots,M-1}_{N})$$

We shall show that S contains all generators of $\mathcal{K}(H)$.
The following lemma is helpful to characterize vectors in $\mathcal{K}(H)$.It shows that we only need to check one inequality to determine whether $\mathbf{p}$ is in $\mathcal{K}(H)$ or not.

Lemma 5.3. *Assume that* $\mathbf{p} = (p_1, p_2, \ldots, p_{M+N}) \in \mathbb{R}^{M+N}_{\geq 0}$ *and that* $p_1 \leq p_2 \leq \cdots \leq p_{M+N}$. *Then* $\mathbf{p} \in \mathcal{K}(H)$ *if and only if*

$$\sum_{i=1}^{M-1} p_i \geq p_{M+N}.$$

Proof. If $\mathbf{p} \in \mathcal{K}(H)$, then $\sum_{i=1}^{M-1} p_i \geq p_{M+N}$ by the definition of fundamental cone.

Now let's assume that $\sum_{i=1}^{M-1} p_i \geq p_{M+N}$.We want to show that (5.7) holds for any M entries of $\mathbf{p}$.

Indeed, if p_{M+N} is on the left hand side of (5.7) the inequality is clearly true since p_{M+N} is the maximum entry in $\mathbf{p}$. If p_{M+N} is on the right hand side of (5.7), then for any $k_1, k_2, \ldots, k_{M-1}$ we have

$$\sum_{l=1}^{M-1} p_{k_l} \geq \sum_{l=1}^{M-1} p_l \geq p_{M+N}.$$

If p_{M+N} doesn't show up in (5.7), then

$$\sum_{l=1}^{M-1} p_{k_l} \geq p_{M+N} \geq p_{k_M}.$$

Thus, (5.7) holds for any M entries of $\mathbf{p}$. □

The following lemma shows the first fold of theorem 5.12.

Lemma 5.4. *Each* $\mathbf{v} \in S$ *is a generator of* $\mathcal{K}(H)$.

Proof. We only need to prove the lemma for $\mathbf{v}_{ij}$. The conclusion for their permutations follows from the symmetry of H.

Each $\mathbf{v}_{ij}$ has the form

$$\mathbf{v}_{ij} = (\underbrace{1, 1, \ldots, 1}_{N+i-(j-1)}, \underbrace{i, i, \ldots, i}_{j}, \underbrace{0, 0, \ldots, 0}_{M-i-1}).$$

If $\mathbf{v}_{ij} = \lambda\mathbf{a} + (1-\lambda)\mathbf{b}$ for some $\mathbf{a}, \mathbf{b} \in \mathcal{K}(H)$ and some $0 \leq \lambda \leq 1$, then

$$\begin{cases} \lambda a_1 + (1-\lambda)b_1 = 1, \\ \ldots \\ \lambda a_{N+i-(j-1)} + (1-\lambda)b_{N+i-(j-1)} = 1, \\ \lambda a_{N+i-(j-2)} + (1-\lambda)b_{N+i-(j-2)} = i, \\ \ldots \\ \lambda a_{N+i+1} + (1-\lambda)b_{N+i+1} = i, \\ \lambda a_{N+i+2} + (1-\lambda)b_{N+i+2} = 0, \\ \ldots \\ \lambda a_{N+M} + (1-\lambda)b_{N+M} = 0 \end{cases} \tag{5.8}$$

The equations above can be divided into 3 sets, which have 1,i and 0 on the right hand side respectively. Let's call these 3 sets $EQS1$,$EQS2$ and $EQS3$. And we have $|EQS1| = N + i - (j-1), |EQS2| = j$ and $|EQS3| = M - i - 1$. Since $a_k \geq 0, b_k \geq 0$ for $1 \leq k \leq N + M$ and $0 \leq \lambda \leq 1$ the equations in $EQS3$ imply that

$$a_k = b_k = 0 \text{ for } N + i + 1 \leq k \leq N + M. \tag{5.9}$$

If we subtract any equation in $EQS2$ from the sum of i arbitrarily chosen equations in $EQS1$ ($1 \leq j \leq N$ so $|EQS1| > i$) we will have

$$\lambda(\sum_{l=1}^{i} a_{k_l} - a_k) + (1-\lambda)(\sum_{l=1}^{i} b_{k_l} - b_k) = 0, \text{ for any } N + i - (j-2) \leq k \leq N + i + 1. \tag{5.10}$$

$\mathbf{a} = (a_1, \ldots, a_{N+i+1}, \underbrace{0, \ldots, 0}_{M-i-1}) \in \mathcal{K}(H)$ so by Lemma 5.3

$$\sum_{l=1}^{i} a_{k_l} + \sum_{l=1}^{M-i-1} 0 \geq a_k, \text{ i.e. } \sum_{l=1}^{i} a_{k_l} - a_k \geq 0.$$

Similarly,

$$\sum_{l=1}^{i} b_{k_l} - b_k \geq 0.$$

Together with $0 \leq \lambda \leq 1$, we conclude from (5.10) that

$$\sum_{l=1}^{i} a_{k_l} - a_k = 0 \text{ and } \sum_{l=1}^{i} b_{k_l} - b_k = 0. \tag{5.11}$$

Because $N + i - (j-2) \leq k \leq N + i + 1$ is arbitrary, if we fix $k_1, \ldots, k_i$ in (5.11) we will have

$$a_{N+i-(j-2)} = \cdots = a_{N+i+1} \text{ and } b_{N+i-(j-2)} = \cdots = b_{N+i+1}. \tag{5.12}$$

If we fix k and $k_1, \ldots, k_{i-1}$ in (5.11) and allow k_i to change we will have

$$a_1 = \cdots = a_{N+i-(j-1)} \text{ and } b_1 = \cdots = b_{N+i-(j-1)}. \tag{5.13}$$

Putting (5.9),(5.12) and (5.13) together, we find that $\mathbf{a}$ and $\mathbf{b}$ are both multiples of $\mathbf{v}_{ij}$, which completes the proof. □

Let $\#supp(\mathbf{p})$ be the number of nonzero entries of $\mathbf{p}$. We have the following lemma.

Lemma 5.5. *Every vector* $\mathbf{p}$ *with* $\#supp(\mathbf{p}) = K + 1 \geq N + 2$ *and in the form*

$$\mathbf{p} = (\underbrace{0, \ldots, 0}_{M+N-(K+1)}, \underbrace{1, \ldots, 1}_{(K+1)-L}, \underbrace{K-N, K-N, \ldots, K-N}_{L})$$

is a nonnegative linear combination of vectors in S.

Proof. If $L \leq N$, then $\mathbf{p}$ is a vector in S. If $L = N + 1$, then $\mathbf{p}$ is the sum of the

following $K-N$ vectors,each of which is a permutation of $v_{1,1}$ in S:

$$\begin{aligned}\mathbf{p} =&(\underbrace{0,\ldots,0}_{M+N-(K+1)} ,\underbrace{1,0,\ldots,0}_{K-N},\underbrace{1,1,\ldots,1}_{N+1})\\ &+(\underbrace{0,\ldots,0}_{M+N-(K+1)} ,\underbrace{0,1,\ldots,0}_{K-N},\underbrace{1,1,\ldots,1}_{N+1})\\ &+\cdots\\ &+(\underbrace{0,\ldots,0}_{M+N-(K+1)} ,\underbrace{0,0,\ldots,1}_{K-N},\underbrace{1,1,\ldots,1}_{N+1}).\end{aligned}$$

If $L \geq N+2$, first note that any vector $\mathbf{v} = (0,\ldots,0,1,1,\ldots,1)$ with $\#supp(\mathbf{v}) \geq N+2$ and its permutations are nonnegative linear combinations of vectors in S. Indeed, if $\#supp(\mathbf{v}) = N+i+1 = N+2$ then $\mathbf{v} = \mathbf{v}_{11} \in S$. If $\#supp(\mathbf{v}) = N+i+1 > N+2$ then $\mathbf{v}$ is the sum of some permutations of $\mathbf{v}_{i1} \in S$ divided by a positive constant:

$$\begin{aligned}\mathbf{v} =&\frac{1}{2i-1}(\underbrace{0,\ldots,0}_{M-i-1},1,1,\ldots,1,i)\\ &+\frac{1}{2i-1}(\underbrace{0,\ldots,0}_{M-i-1},1,1,\ldots,i,1)\\ &+\cdots\\ &+\frac{1}{2i-1}(\underbrace{0,\ldots,0}_{M-i-1},i,1,\ldots,1,1)\end{aligned}$$

Now $\mathbf{p}$ will be the sum of following $K-N$ vectors:

$$\begin{aligned}
\mathbf{p} =&(\underbrace{0,\ldots,0}_{M+N-(K+1)} ,\underbrace{1,0,\ldots,0}_{K+1-L},\underbrace{1,1,\ldots,1}_{L})\\
&+(\underbrace{0,\ldots,0}_{M+N-(K+1)} ,\underbrace{0,1,\ldots,0}_{K+1-L},\underbrace{1,1,\ldots,1}_{L})\\
&+\cdots\\
&+(\underbrace{0,\ldots,0}_{M+N-(K+1)} ,\underbrace{0,0,\ldots,1}_{K+1-L},\underbrace{1,1,\ldots,1}_{L})\\
&+(\underbrace{0,\ldots,0}_{M+N-(K+1)} ,\underbrace{0,0,\ldots,0}_{K+1-L},\underbrace{1,1,\ldots,1}_{L})\\
&+(L-(N+1))(\underbrace{0,\ldots,0}_{M+N-(K+1)} ,\underbrace{0,0,\ldots,0}_{K+1-L},\underbrace{1,1,\ldots,1}_{L}).
\end{aligned}$$

Observe that each vector occurring in $\mathbf{p}$ is a nonnegative linear combination of vectors in S, hence so is $\mathbf{p}$. □

The following lemma is the second fold of theorem 5.12.

Lemma 5.6. *Every $\mathbf{p} = (p_1, p_2, \ldots, p_{M+N}) \in \mathcal{K}(H)$ is a nonnegative linear combination of the vectors in S.*

Proof. Similar to the proof of Lemma 5.2, we can just assume that $\mathbf{p}$ is a lattice point.

First notice that if a nonzero vector $\mathbf{p} \in \mathcal{K}(H)$ we must have $\#supp(\mathbf{p}) \geq N+2$. If $\#supp(\mathbf{p}) \leq N+1$ then there are at least $M-1$ zero entries in $\mathbf{p}$. Then the inequality in Lemma 5.3 is not satisfied, which implies that $\mathbf{p}$ is not in $\mathcal{K}(H)$.

If $\mathbf{p} \in \mathcal{K}(H)$, then any permutation of $\mathbf{p}$ is also in $\mathcal{K}(H)$ because H consists of all permutations of its first row. So without loss of generality we may assume that $p_1 \leq p_2 \leq \cdots \leq p_{M+N}$ if $\mathbf{p} = (p_1, p_2, \ldots, p_{M+N})$. We want to use induction on $\#supp(\mathbf{p})$.

If $\#supp(\mathbf{p}) = N+2$, then $p_1 = p_2 = \cdots = p_{M-2} = 0$. By Lemma 5.3 we have $p_{M-1} = p_{M+N}$, which means that $p_{M-1} = p_M = \cdots = p_{M+N}$. So $\mathbf{p}$ is just a multiple of $\mathbf{v}_{11} \in S$.

Assume that when $\mathbf{p} \in \mathcal{K}(H)$ and $\#supp(\mathbf{p}) = K$ $\mathbf{p}$ is a nonnegative linear combination of vectors in S. Now let's study the case when $\#supp(\mathbf{p}) = K+1$.

To facilitate our proof we let

$$\Delta(\mathbf{p}) = (\Delta_1, \Delta_2, \ldots, \Delta_{M+N}),$$

where

$$\Delta_1 = p_1 \text{ and } \Delta_i = p_i - p_{i-1} \text{ for } 2 \leq i \leq M+N.$$

Because $\#supp(\mathbf{p}) = K+1$ we have

$$\Delta_1 = \cdots = \Delta_{M+N-(K+1)} = 0.$$

Now let

$$\tilde{\mathbf{p}} = (K-N-1)\mathbf{p}.$$

Then

$$\Delta(\tilde{\mathbf{p}}) = (\tilde{\Delta}_1, \ldots, \tilde{\Delta}_{M+N}),$$

where $\tilde{\Delta}_i = (K-N-1)\Delta_i$ for $1 \leq i \leq M+N$. Note that each $\tilde{\Delta}_i$ is a multiple of $K-N-1$.

The idea is to successively subtract a vector, which is a nonnegative linear combination of vectors in S, from $\tilde{\mathbf{p}}$. And we want to show that after finite step we will get a vector in $\mathcal{K}(H)$ with $\#supp = K$ or some vector that is a nonnegative linear combination of vectors in S.

If there are L identical maximum entries in $\tilde{\mathbf{p}}$, i.e. $\tilde{p}_{M+N-(L-1)} = \tilde{p}_{M+N-(L-2)} = \cdots = \tilde{p}_{M+N}$, then let

$$\begin{aligned} \tilde{\mathbf{p}}' &= \tilde{\mathbf{p}} - \mathbf{q}_L \\ &= \tilde{\mathbf{p}} - (\underbrace{0, \ldots, 0}_{M+N-(K+1)}, \underbrace{1, \ldots, 1}_{(K+1)-L}, \underbrace{K-N, \ldots, K-N}_{L}). \end{aligned} \tag{5.14}$$

By Lemma 5.5, $\mathbf{q}_L$ is a nonnegative linear combination of vectors in S.

Then we have

$$\tilde{\Delta}'_i = \tilde{\Delta}_i = 0, \text{ for } 1 \leq i \leq M+N-(K+1),$$

$$\tilde{\Delta}'_{M+N-K} = \tilde{\Delta}_{M+N-K} - 1,$$

$$\tilde{\Delta}'_i = \tilde{\Delta}_i, \text{ for } M+N-(K-1) \le i \le M+N-L,$$

$$\tilde{\Delta}'_{M+N-L+1} = \tilde{\Delta}_{M+N-L+1} - (K-N-1),$$

and

$$\tilde{\Delta}'_i = \tilde{\Delta}_i = 0, \text{ for } M+N-L+2 \le i \le M+N.$$

In other words,

$$\tilde{\Delta}' = (\underbrace{0,\ldots,0}_{M+N-(K+1)}, \tilde{\Delta}_{M+N-K}-1, \tilde{\Delta}_{M+N-K+1}, \ldots, \tilde{\Delta}_{M+N-L}, \tilde{\Delta}_{M+N-L+1}-(K-N-1), 0,\ldots,0).$$

We shall show that $\tilde{\mathbf{p}}' \in \mathcal{K}(H)$.

Indeed,

$$\tilde{\mathbf{p}}' = (\underbrace{0,\ldots,0}_{M+N-(K+1)}, \underbrace{\tilde{p}_{M+N-K}-1,\ldots,\tilde{p}_{M+N-L}-1}_{(K+1)-L}, \tilde{p}_{M+N-L+1}-(K-N-1),\ldots, \tilde{p}_{M+N}-(K-N-1)).$$

If $M+N-L \ge M-1$, by Lemma 5.3 $\tilde{\mathbf{p}}' \in \mathcal{K}(H)$.

If $M+N-L < M-1$, because $\tilde{p}_{M+N-L+1} = \tilde{p}_{M+N}$ we still have $\tilde{\mathbf{p}}' \in \mathcal{K}(H)$ by Lemma 5.3.

Now if $\tilde{\Delta}'_{M+N-K} > 0$ and $\tilde{\Delta}'_{M+N-L+1} > 0$ we let $\tilde{\mathbf{p}}'$ be $\tilde{\mathbf{p}}$ and repeat (5.14) until $\tilde{\Delta}'_{M+N-K} = 0$ or $\tilde{\Delta}'_{M+N-L+1} = 0$. This is guaranteed because each $\tilde{\Delta}'_i$ is a multiple of $K-N-1$ and every time $\tilde{\Delta}'_{M+N-K}$ and $\tilde{\Delta}'_{M+N-L+1}$ decrease by 1 and $K-N-1$ respectively.

If $\tilde{\Delta}'_{M+N-K} = 0$ then $\#supp(\tilde{\mathbf{p}}') = K$. By induction assumption the lemma is proved.

If $\tilde{\Delta}'_{M+N-L+1} = 0$ then we let $\tilde{\mathbf{p}}'$ be $\tilde{\mathbf{p}}$ and let

$$\begin{aligned}\tilde{\mathbf{p}}' &= \tilde{\mathbf{p}} - \mathbf{q}_{L+1} \\ &= \tilde{\mathbf{p}} - (\underbrace{0,\ldots,0}_{M+N-(K+1)}, \underbrace{1,\ldots,1}_{K-L}, \underbrace{K-N,\ldots,K-N}_{L+1}).\end{aligned}$$

Now we have

$$\tilde{\Delta}' = (\underbrace{0,\ldots,0}_{M+N-(K+1)}, \tilde{\Delta}_{M+N-K}-1, \tilde{\Delta}_{M+N-K+1}, \ldots, \tilde{\Delta}_{M+N-L}, \tilde{\Delta}_{M+N-L}-(K-N-1),$$
$$0,\ldots,0).$$

Repeat this process until $\tilde{\Delta}'_{M+N-K}=0$ or $\tilde{\Delta}'_{M+N-L}=0$. If $\tilde{\Delta}'_{M+N-K}=0$ we have a vector with $\#supp = K$ and the lemma is proved by induction assumption. If $\tilde{\Delta}'_{M+N-L}=0$ we let $\tilde{\mathbf{p}}'$ be $\tilde{\mathbf{p}}$ and subtract $\mathbf{q}_{L+2}$ from it.

So we will eventually have $\tilde{\Delta}'_{M+N-K}=0$ or $\tilde{\Delta}'_{M+N-K}$ is the only nonzero entry in $\tilde{\Delta}'$. In the former case we have a vector with $\#supp = K$ and in the latter case $\tilde{\mathbf{p}}'$ is a multiple of $\mathbf{v}_{11} \in S$. So the lemma is proved. □

Theorem 5.12. *S generates $\mathcal{K}(H)$.*

Proof. It is an immediate conclusion from Lemma 5.4 and Lemma 5.6. □

5.3.3 $H = (G\,G)$ and open problems

Now let's consider a parity-check matrix H that is the concatenation of two identical matrices G. We pose the following conjecture about the structure of the fundamental cone $\mathcal{K}(H)$.

Conjecture 5.1. *Assume that S is a set of generators of the fundamental cone $\mathcal{K}(G)$ of an $m \times n$ parity-check matrix G. Let $\mathbf{e_i} = (0,\ldots,0,\underset{ith}{1},0,\ldots,0) \in \mathbb{R}^n$. Then a vector $\mathbf{v}$ is a generator of $\mathcal{K}(H)$, where $H = (G\,G)$, if and only if it satisfies one of the following two conditions.*

1. *$\mathbf{v} = (v_1, v_2, \ldots, v_n, v_{n+1}, \ldots, v_{2n})$ such that $(v_1+v_{n+1}, \ldots, v_i+v_{n+i}, \ldots, v_n+v_{2n}) \in S$ and $v_i v_{n+i} = 0$ for $1 \le i \le n$;*
2. *$\mathbf{v} = (\mathbf{e_i}, \mathbf{e_i})$ for some $1 \le i \le n$.*

We have studied the fundamental cones of three different types of parity-check matrices. In all cases the parity-check matrices are obtained by applying a subgroup of the permutation group of n elements, where n is the code length, to the

first rows of the matrices. And in the searching process for generators of fundamental cones we implicitly used the group symmetry at some step. So we are naturally led to the following questions:

1. What are the generators of the fundamental cone if the parity-check matrix is obtained by applying some other permutation subgroup to the first row?

2. How is the symmetry on the rows of the parity-check matrix reflected on the generators of the fundamental cone?

3. If there is no symmetry on the rows of the parity-check matrix, can its fundamental cone be approximated by those with symmetries?

Bibliography

[1] O. Barak, D. Burshtein, and M. Feder, *Bounds on achievable rates of ldpc codes used over the binary erasure channel*, IEEE Transactions on Information Theory **50** (2004), no. 10, 2483–2492.

[2] L. Bazzi, T. J. Richardson, and R. Urbanke, *Exact thresholds and optimal codes for the binary-symmetric channel and gallager's decoding algorithm a.*, IEEE Transactions on Information Theory **50** (2004), no. 9, 2010–2021.

[3] M. Beck and S. Robins, *Computing the continuous discretely*, Springer, 2007.

[4] C. Berrou, A. Glavieux, and P. Thitimajshima, *Near shannon limit error-correcting coding and decoding*, Proc. Int. Communications Conf. (ICC'93, Geneva, Switzerland), 1993, pp. 1064–1070.

[5] N. V. Chernikova, *An algorithm for finding a general formula for nonnegative solutions of system of linear inequalities*, U.S.S.R. Computational Mathematics and Mathematical Physics **5** (1965), 228–233.

[6] S.-Y. Chung, G. D. Forney, T. J. Richardson, and R. Urbanke, *On the design of low-density parity-check codes within 0.0045 db of the shannon limit*, IEEE Commun. Lett. **5** (2001), no. 2, 58–60.

[7] K. Fuduka and A. Prodon, *Double description method revisited*, Selected papers from the 8th Franco-Japanese and 4th Franco-Chinese Conference on Combinatorics and Computer Science, Springer-Verlag, 1996, pp. 91–111.

[8] R. G. Gallager, *Low-density parity-check codes*, Ph.D. thesis, M.I.T., 1962.

[9] Ralf Koetter, Wen-Ching W. Li, Pascal O. Vontobel, and Judy L. Walker, *Pseudo-codewords of cycle codes via zeta functions*, Proc. IEEE Inform. Theory Workshop (San Antonio, TX, USA), 2004, pp. 7–12.

[10] ———, *Characterizations of pseudo-codewords of (low-density) parity-check codes*, Advances in Mathematics **213** (2007), 205–229.

[11] Ralf Koetter and Pascal O. Vontobel, *Graph covers and iterative decoding of finite-length codes*, Proc. 3rd Intern. Conf. on Turbo Codes and Related Topics (Brest, France), 2003, pp. 75–82.

[12] ———, *Towards low-complexity linear-programming decoding*, CoRR **abs/cs/0602088** (2006).

[13] Wen-Ching W. Li, Min Lu, and Chenying Wang, *Recent developments in low-density parity-check codes*, Proceedings of the International Workshop on Coding and Cryptography 2009, Lecture Notes in Computer Science 5557, Springer-Verlag, 2009, pp. 107–123.

[14] Min Lu, *On convergence speed of capacity-achieving sequences for erasure channel*, IEEE Transactions on Information Theory **54** (2008), no. 4, 1793–1794.

[15] M. Luby, M. Mitzenmacher, A. Shokrollah, and D. Spielman, *Analysis of low density codes and improved designs using irregular graphs*, In Proceedings of the 30th annual ACM Symposium on Theory of Computing, 1998, pp. 249–258.

[16] M. Luby, M. Mitzenmacher, A. Shokrollahi, and D. Spielman, *Efficient erasure correcting codes*, IEEE Transactions on Information Theory **47(2)** (2001), 569–584.

[17] ———, *Improved low-density parity-check codes using irregular graphs*, IEEE Transactions on Information Theory **47** (2001), no. 2, 585–598.

[18] M. Luby, M. Mitzenmacher, A. Shokrollahi, D. Spielman, and V. Stemann, *Practical loss-resilient codes*, Proceedings of the 29th annual ACM Symposium on Theory of Computing, 1997, pp. 150–159.

[19] D. J. C. MacKay, *Good error-correcting codes based on very sparse matrices*, IEEE Transactions on Information Theory **45** (1999), no. 2, 399–431.

[20] T. S. Motzkin, H. Raiffa, G. L. Thompson, and R. M. Thrall, *The double description method*, Contributions to the Theory of Games II (H. W. Kuhn and A. W. Tucker, eds.), Ann. of Math. Stud., vol. 8, Princeton University Press, 1953, pp. 51–73.

[21] P. Oswald and A. Shokrollahi, *Capacity-achieving sequences for the erasure channel*, IEEE Transactions on Information Theory **48** (2002), no. 12, 3017–3028.

[22] T. J. Richardson, A. Shokrollahi, and R. Urbanke, *Design of capacity-approaching irregular low-density parity-check codes*, IEEE Transactions on Information Theory **47** (2001), no. 2, 619–637.

[23] T. J. Richardson and R. Urbanke, *The capacity of low-density parity-check codes under message-passing decoding*, IEEE Transactions on Information Theory **47** (2001), no. 2, 599–618.

[24] A. Shokrollahi, *New sequences of linear time erasure codes approaching the channel capacity*, Proceedings of the 13th International Symposium on Applied Algebra, Algebraic Algorithms, and Error-Correcting Codes (M. Fossorier, H. Imai, S. Lin, and A. Poli, eds.), Lecture Notes in Computer Science, no. 1719, 1999, pp. 65–76.

[25] A. Shokrollahi and R. Storn, *Design of efficient erasure codes with differential evolution*, Proc. ISIT'00, 2000, p. 5.

[26] M. Sipser and D. Spielman, *Expander codes*, IEEE Trans. Inform. Theory **42** (1996), no. 6, part 1, 1710–1722. MR 98d:94031

[27] H. M. Stark and A. A. Terras, *Zeta functions of finite graphs and coverings*, Adv. Math. **121** (1996), no. 1, 124–165.

[28] Günter. M. Ziegler, *Lectures on polytopes*, Springer-Verlag, 1995.

Vita

Min Lu was born in 1979 in Fuzhou, the capital of Fujian province of China. He graduated from Fuzhou No. 2 middle school in 1997 and was admitted into Shanghai Jiaotong university, where he received the bachelor's degree in mathematics in 2001. Between July 2001 and May 2002, he was working as a software engineer at Shanghai Kingstar Computer Company (acquired by Sungard in 2007). Since August 2002 he has been working under the supervision of Dr. Wen-Ching Winnie Li towards the doctorate degree at the department of mathematics in the Pennsylvania State University (University Park).