On Constructing Low-Density Parity-Check Codes

by

Xudong Ma

A thesis
presented to the University of Waterloo
in fulfilment of the
thesis requirement for the degree of
Doctor of Philosophy
in
Electrical and Computer Engineering

Waterloo, Ontario, Canada, 2007

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Abstract

This thesis focuses on designing Low-Density Parity-Check (LDPC) codes for forward-error-correction. The target application is real-time multimedia communications over packet networks. We investigate two code design issues, which are important in the target application scenarios, designing LDPC codes with low decoding latency, and constructing capacity-approaching LDPC codes with very low error probabilities.

On designing LDPC codes with low decoding latency, we present a framework for optimizing the code parameters so that the decoding can be fulfilled after only a small number of iterative decoding iterations. The brute force approach for such optimization is numerical intractable, because it involves a difficult discrete optimization programming. In this thesis, we show an asymptotic approximation to the number of decoding iterations. Based on this asymptotic approximation, we propose an approximate optimization framework for finding near-optimal code parameters, so that the number of decoding iterations is minimized. The approximate optimization approach is numerically tractable. Numerical results confirm that the proposed optimization approach has excellent numerical properties, and codes with excellent performance in terms of number of decoding iterations can be obtained. Our results show that the numbers of decoding iterations of the codes by the proposed design approach can be as small as one-fifth of the numbers of decoding iterations of some previously well-known codes. The numerical results also show that the proposed asymptotic approximation is generally tight for even non-extremely limiting cases.

On constructing capacity-approaching LDPC codes with very low error probabilities, we propose a new LDPC code construction scheme based on 2-lifts. Based on stopping set distribution analysis, we propose design criteria for the resulting codes to have very low error floors. High error floors are the main problems of previously constructed capacity-approaching codes, which prevent them from achieving very low error probabilities. Numerical results confirm that codes with very low error floors can be obtained by the proposed code construction scheme and the design criteria. Compared with the codes by the previous standard construction schemes, which have error floors at the levels of 10^{-3} to 10^{-4}, the codes by the proposed approach do not have observable error floors at the levels higher than 10^{-7}. The error floors of the codes by the proposed approach are also significantly lower compared with the codes by the previous approaches to constructing codes with low error floors.

Acknowledgments

First of all, I would like to thank my thesis supervisor, Prof. En-hui Yang, for the supports, and advices he has provided throughout my time as his student. I would also like to express my gratitude toward the other members of my committee, Prof. Frank Kschischang, Prof. Amir Khandani, Prof. Oussama Damen, and Prof. Bruce Richter, for helpful discussions, valuable suggestions and criticisms.

I have spent the last seven years in University of Waterloo, first working toward the M.A.Sc. degree, and later the Ph.D. degree. In this process, I have been fortunate enough to meet many people and benefited from interacting with them in terms of expanding my knowledge, skills, and experience. I wish to take this opportunity to thank all of them.

Finally, I wish to express my sincere gratitude to my family. This work would not have been possible without their supports.

Dedication

This thesis is dedicated to my family for their supports and encouragements.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Low-Density Parity-Check (LDPC) codes have attracted a lot of attention in recent years. The codes have several properties, which make them favorable choices for real-time and high-throughput communications. First, the codes are capacity-approaching. Second, the codes can be efficiently decoded by parallel iterative decoding algorithms with low latency.

This thesis focuses on designing LDPC codes for forward-error-correction. The target application scenarios are real-time multimedia communications over packet networks, which are next-generation cutting-edge technologies. Several code design issues, which are crucial in the target applications, are investigated. We propose novel code construction and design approaches, which are enabling technologies in the application scenarios.

1.1 Real-Time Multimedia Communications Over Packet Networks

During recent years, there have been increasing demands for real-time multimedia communications over packet networks. One typical application is multimedia conferencing, which is increasingly adopted in education, health-care, business management etc. Other applications include Voice over IP, and digital video surveillance. However, achieving real-time communications is a very challenging problem, certain crucial enabling technologies are not available until recent years.

Compared with non real-time communication systems, real-time multimedia communication systems present the following challenges.

- First, latency requirements in real-time communication systems are much more stringent. While in a non real-time communication system, the latency for signal processing, data compression, and error correction can be as large as several seconds, the latency in the counter-part real-time systems is usually within several milliseconds. In the state-of-the-art video conferencing and videophone systems, the latency is usually required to be no larger than 50 milliseconds, and is desired to be within 10 milliseconds. Failing to meet the latency requirements usually results in noticeable delays, which makes the conversations impossible. For example, in the H.264 codec developed by W&W communication Inc. for real-time communications, the latency is within two milliseconds, where parallel processing and high-speed processing units are massively adopted to accelerate the encoding and decoding speeds [42].

- Second, the transmission of compressed images and videos is less tolerant of transmission errors because of error propagation. Therefore, the error probability requirements are much more stringent.

1.2 System Description

In practical packet network communication systems, the transmission is impaired by packet loss. The packet loss can be either caused by network congestion, or by packet dropping at relay nodes, where check-sum errors are detected. In order to compensate for lost packets, forward-error-correction schemes are usually adopted. The forward-error-correction schemes introduce redundancy to the transmitted signals. Therefore, at the receiver end, the transmitted messages can be recovered, even when only a fraction of the total transmitted packets is received.

The block diagram of the considered forward-error-correction scheme is shown in Fig. 1.1. In this scheme, compressed multimedia signals are first encoded by LDPC encoding. The coded bits are then grouped into packets. We call this process *packetization*. The packets are transmitted through packet networks. At the receiver end, the received bits are de-packetized, which is the inverse process of packetization. Finally, the transmitted messages are recovered from the received bits by LDPC decoding. Here, we assume that the packetization scheme is a randomized scheme, that is, for each transmitted codeword, the bits in the codeword are randomly permuted and divided into different groups, where each group corresponds to one packet. Hence, analogue to the *random coding argument*, the performance of

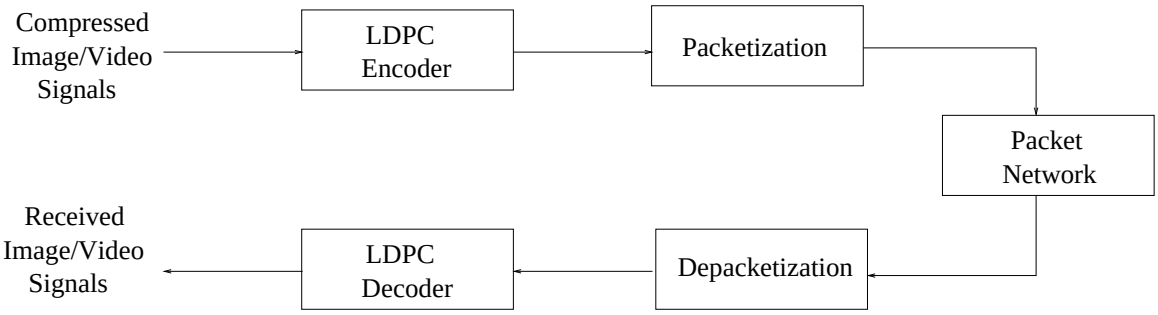


Figure 1.1: A forward-error-correction scheme with LDPC codes and interleaving

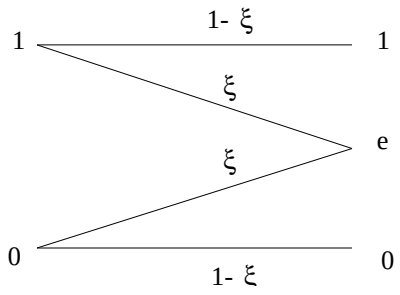


Figure 1.2: Binary Erasure Channel

the considered system corresponds to the average performance of all packetization schemes.

Because of the randomized packetization scheme, the received signals at the input ends of LDPC decoders can be considered as LDPC codewords with randomly missing bits. Therefore, the data transmission processes from the output ends of LDPC encoders to the input ends of LDPC decoders can be modeled as Binary Erasure Channels (BEC). An example of BEC is shown in Fig. 1.2. The channel takes binary inputs and outputs 0, 1, or e, where e stands for a missing bit. The channel parameter ξ is the probability that a transmitted bit is lost. The channel outputs a transmitted bit correctly with probability $1 - \xi$, and outputs e with probability ξ. We call the missing bits erasures. It can be checked that the capacity is $C = 1 - \xi$.

1.3 Problem Statement

In order to meet the above stringent requirements on decoding latency and error probabilities, we consider the following design issues for LDPC codes.

- We consider the problem of optimizing code parameters (more precisely, the degree distributions), so that the code can be successfully decoded after only a small number of parallel iterative decoding iterations. We say that a code has *fast decoding speeds*, if the code have the above property. We argue that the problem is crucial because fast decoding speeds are equivalent to low decoding latency, if the decoding is implemented by highly parallel hardware implementations, which is necessary for real-time communication scenarios. It should be noted that we give less considerations to total decoding complexity, because total decoding complexity is a secondary design objective compared with decoding latency, which must be designed to meet certain requirements.
- We consider the problem of designing LDPC codes with low error floors. High error floors are the main problems of previously constructed capacity-approaching LDPC codes, which prevent them from achieving very low error probabilities.

1.4 Summary of Contributions

We present an asymptotic approximation to the number of decoding iterations for capacity-approaching low-density LDPC codes. We also show by numerical examples that this approximation is tight for even non-extremely limiting cases. Therefore, the approximation is tight for most practical code design scenarios, because practical capacity-approaching codes always have low densities.

We present an approximate optimization framework for designing LDPC codes with fast decoding speeds. The framework is based on the above asymptotic approximation to the number of decoding iterations. The direct optimization of the code parameters is numerical intractable, because it involves a difficult discrete optimization programming. The proposed approximate optimization approach is numerical tractable.

We propose a new capacity-approaching LDPC code construction scheme based on a series of random 2-lifts. We propose design criteria for the above 2-lift based codes so that small stopping sets are largely avoided in the corresponding Tanner

graphs. According to previous research, high error floors are mainly the results of the small stopping sets. Numerical results confirm that the resulting codes have significantly lower error floors, compared with previously constructed capacity-approaching codes in the literature.

1.5 Organization of this Thesis

This thesis is organized as follows. In Chapter 2, we provide a brief review of LDPC codes, including the construction schemes, the decoding algorithms, and the state-of-the-art performance analysis methods. In Chapter 3, we present our approximate optimization approach to designing LDPC codes with fast decoding speeds. In Chapter 4, we discuss the proposed code construction scheme by 2-lifts. Conclusions will be provided in Chapter 5. We will discuss possible future research in Chapter 6.

Chapter 2

Introduction to Low-Density Parity-Check Codes

In this chapter, we will provide a review of LDPC codes. In Section 2.1, we will discuss the Tanner graph representations, which are widely adopted graphical models for LDPC codes. In Section 2.2, we will review the Richardson-Urbanke code ensembles. We will review factor graphs in Section 2.3. In Section 2.4, we will discuss the message-passing algorithm, which is a parallel algorithm for marginal distribution calculation. In Section 2.5, we will review the message-passing decoding algorithm for LDPC codes, which is a special case of the general message-passing algorithm.

In this chapter, we will also review the state-of-the-art LDPC design and performance analysis methods. These methods can be divided into two main categories: the asymptotic methods, and the finite-length methods. Each category of methods has its strength and weakness. The two categories of methods complement each other, and are both adopted in practical code design. The asymptotic methods investigate the limiting behaviors of the decoding algorithms as the blocklengths go to infinity. These asymptotic results provide insights on LDPC codes with long blocklengths. One of the most important asymptotic methods is *Density Evolution*, which will be reviewed in Sections 2.6, 2.7, and 2.8.

The finite-length methods, on the other hand, investigate the error probability performance of codes with fixed blocklengths. The methods relate the error probabilities to codeword-like structures, such as stopping sets, trapping sets, and pseudo-codewords. We will present a brief introduction on stopping sets in Section 2.9. More detailed in-context exposition on stopping set based analysis will be provided in Chapter 4.

2.1 Tanner Graph

Low-Density Parity-Check codes were invented by Gallager in 1960s [13]. Unfortunately, the codes were largely forgotten in the following three decades, because the corresponding decoding is difficult to be implemented in hardware at that time. However, LDPC codes become popular again in the first decade of the 21st century. The reasons are two-fold. First, recent theoretical results have shown that LDPC codes can approach Shannon capacity limit [23], [7]. Second, the hardware implementation of the decoding algorithms are not difficult anymore nowadays due to progress in electrical circuit technologies. The codes have already found many applications in deep-space communications, satellite communications and wireless communications.

Low-Density Parity-Check codes are linear block codes with sparse parity-check matrices. More precisely, a linear block code $\mathcal{C}$ is a set of binary vectors $\boldsymbol{x}$, such that $\boldsymbol{x} \in \mathcal{C}$ if and only if $\boldsymbol{H}\boldsymbol{x} = \mathbf{0}$, where $\boldsymbol{H}$ is a given matrix, $\mathbf{0}$ is the all zero vector and all elements of $\boldsymbol{x}$ and $\boldsymbol{H}$ are in $GF(2)$ (the Galois field of order two). A Low-Density Parity-Check code is a linear block code, where each row and column of the matrix $\boldsymbol{H}$ contains only a small number of non-zero elements.

Tanner graphs, which were invented by Tanner in 1980s [38], are widely adopted graphical representations for LDPC codes. A Tanner graph is a bipartite graph with one partition consisting of all variable nodes and the other partition consisting of all check nodes. Each variable node represents one codeword bit and each check node represents one parity-check constraint. There exists an edge between a variable node and a check node if and only if the corresponding bit is involved in the parity-check constraint.

An example of a Tanner graph is shown in Fig. 2.1. The corresponding parity-check matrix is shown as follows:

$$\boldsymbol{H} = \begin{bmatrix} 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 1 \end{bmatrix} \tag{2.1}$$

In Tanner graph representation, the variable nodes are usually drawn as circles, and the check nodes are usually drawn as squares.

2.2 Richardson-Urbanke Ensemble

The design and analysis of LDPC codes are usually based on ensembles. An LDPC code ensemble is a set of codes with certain properties. Usually, it is easy to evaluate

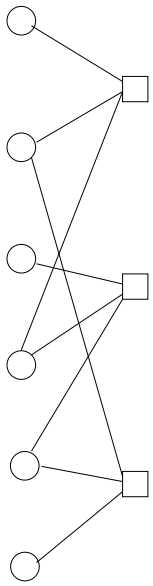


Figure 2.1: The Tanner graph representation

the average performance of a code ensemble or the performance of the majority of codes in a code ensemble, while it is very difficult to determine the performance of one particular code. In the first step of a typical LDPC code design procedure, a code ensemble is found, so that the majority of codes in the code ensemble have satisfactory performance. An LDPC code is usually randomly constructed by uniformly choosing one code from the code ensemble. Note that after the code is chosen, it is fixed in practical systems, despite that the code is randomly generated and the analysis is ensemble based.

A Richardson-Urbanke LDPC code ensemble $\mathcal{C}(\lambda, \rho, n)$ is specified by three parameters: a left degree distribution $\lambda(x)$, a right degree distribution $\rho(x)$, and a blocklength n [31]. The left and right degree distributions are polynomials

$$\lambda(x) = \sum_{i=2}^{d_v} \lambda_i x^{i-1}, \;\; \rho(x) = \sum_{j=2}^{d_c} \rho_j x^{j-1}. \tag{2.2}$$

An LDPC code is in the code ensemble $\mathcal{C}(\lambda, \rho, n)$, if and only if,

- the blocklength of the code is n;
- in the corresponding Tanner graph, the fraction of edges that are connected to variable nodes with degree i is λ_i; and

- in the corresponding Tanner graph, the fraction of edges that are connected to check nodes with degree j is ρ_j.

The Richardson-Urbanke ensembles are also called *standard* or *unstructured* code ensembles in the literature.

2.3 Factor Graphs

Factor graphs represent global product functions and illustrate how the global functions can be factored into local functions. Since such global functions and decompositions are frequently observed in statistical physics, machine learning and artificial intelligence, factor graphs have become a widely-adopted technical language in these areas. We refer interested readers to [18] for a more complete survey on the topic.

In this thesis, we use factor graphs to represent global probability distribution functions, which can be factored into conditional probability functions. Factor graphs provide a convenient technical language to describe the message-passing algorithm, which will be discussed in Section 2.4.

A factor graph is a bipartite graph with one partition consisting of function nodes and the other partition consisting of variable nodes. Each function node represents one local function. Each variable node represents one variable. The global function is the product of all the local functions. Edges indicate the involvement between functions and variables; that is, there is an edge between one variable node and one function node if and only if the variable is an argument of the local function.

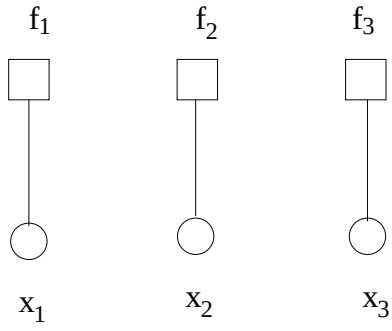


Figure 2.2: The factor graph

An example of a factor graph is shown in Fig. 2.2. We use circles to represent variable nodes and squares to represent function nodes. The factor graph represents the probability distribution function of a binary sequence $X_1X_2X_3$ with length three. The random variables X_i, $1 \leq i \leq 3$, are independent and identically distributed with a probability distribution $p(x)$. The global function is

$$f_{X_1X_2X_3}(x_1, x_2, x_3) = f_{X_1}(x_1)f_{X_2}(x_2)f_{X_3}(x_3), \tag{2.3}$$

where, $f_{X_i}(\cdot) = p(\cdot)$, for $i = 1, 2, 3$, are the local functions.

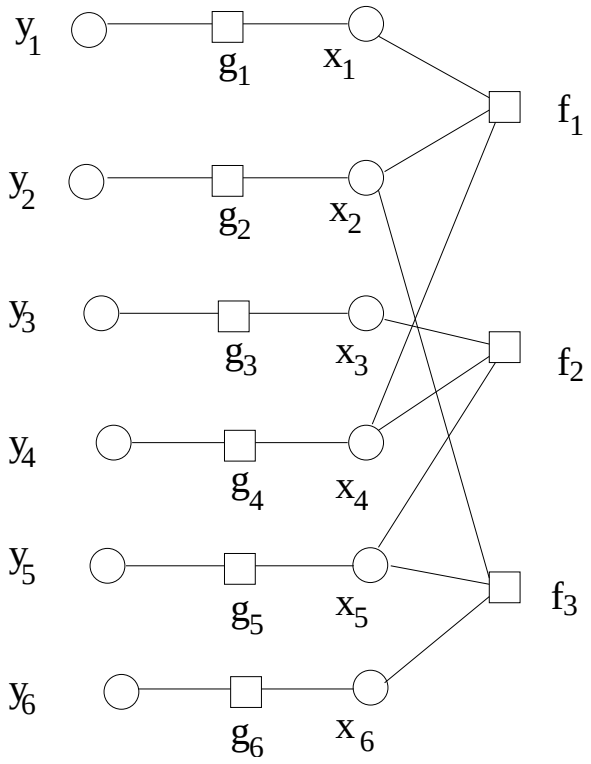


Figure 2.3: An example of factor graph representations of probability distributions in LDPC decoding

In this thesis, we use factor graphs to represent probability distributions in the decoding of LDPC codes. Let $\boldsymbol{x}_1^n = \{x_1, \ldots, x_n\}$ denote the transmitted codeword. Let $\boldsymbol{y}_1^n = \{y_1, \ldots, y_n\}$ be the received channel outputs. Assume the channel is memoryless. Define functions f_j as,

$$f_j(\boldsymbol{x}_1^n) = \begin{cases} 1, & \text{if } j^{th} \text{ parity check is satisfied} \\ 0, & \text{otherwise} \end{cases} \tag{2.4}$$

Define functions $g_i(x_i, y_i)$ as the conditional probability that y_i is the channel output, given that x_i is transmitted. That is,

$$g(x_i, y_i) = \mathbb{P}(y_i \text{ received } | x_i \text{ transmitted }). \tag{2.5}$$

The probability density function $p(\boldsymbol{x}_1^n, \boldsymbol{y}_1^n)$ can be factored as,

$$p(\boldsymbol{x}_1^n, \boldsymbol{y}_1^n) = \alpha \left(\prod_{j=1}^{m} f_j(\boldsymbol{x}_1^n) \right) \left(\prod_{i=1}^{n} g(x_i, y_i) \right), \tag{2.6}$$

where α is a normalization constant, and m is the number of parity-checks. An example of such a factor graph for LDPC decoding is shown in Fig. 2.3.

2.4 Message-Passing Algorithm

The message-passing algorithm is a parallel algorithm for marginal distribution calculation. The algorithm is widely adopted in signal processing, machine learning, artificial intelligence etc. For example, the BCJR algorithm and Kalman filtering can all be considered as variations of the message-passing algorithms. In this section, we present a brief introduction on the message-passing algorithms in the prospective of LDPC decoding. We refer interested readers to [18] and references therein for more general discussion.

The marginal distributions can be exactly calculated by the message-passing algorithm in polynomial time, if the global probability distribution functions can be represented by acyclic or tree-structured factor graphs, that is, the factor graphs do not contain cycles. For the decoding of LDPC codes, the factor graphs are usually cyclic. In such cases, the calculation of exact marginal probability distributions is usually NP-hard. Message-passing algorithms on cyclic factor graphs are called *loopy propagations* in machine learning. The loopy propagation message-passing algorithm calculates the messages using the same set of rules of the message-passing algorithms on tree-structured factor graphs. The calculated results are only approximate values of marginal distributions. However, this algorithm usually has satisfactory performance for LDPC decoding. A review of rigorous analysis on this loopy propagation message-passing algorithm will be provided in Sections 2.6, 2.7, and 2.8. Throughout this thesis, we use the term *message-passing algorithm* to denote both the standard message-passing algorithm and loopy propagation message-passing algorithm, when no ambiguity arises.

In this section, we will first provide a review of the message-passing algorithm on tree-structured factor graphs. We consider a factor graph with n variable nodes and m function nodes. The global probability distribution function f can be factored as follows:

$$f(x_1, \ldots, x_n) = \prod_{j=1}^{m} f_j(\mathcal{X}_j), \tag{2.7}$$

where $\mathcal{X}_j$ denotes the subset of variables, which are arguments of the j-th function $f_j(\cdot)$. The marginal distribution with respect to a variable X_i is,

$$f_{X_i}(s) = \sum_{x_1} \sum_{x_2} \cdots \sum_{x_{i-1}} \sum_{x_{i+1}} \cdots \sum_{x_n} f(x_1, \ldots, x_{i-1}, s, x_{i+1}, \ldots, x_n). \tag{2.8}$$

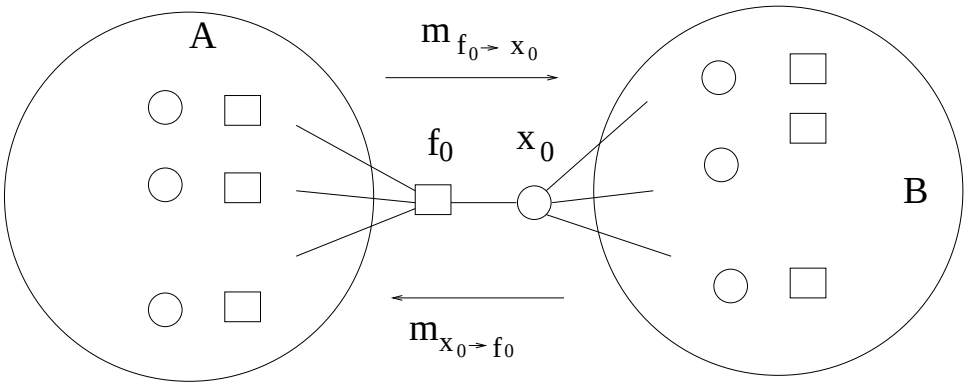


Figure 2.4: The interpretation of messages

In the message-passing algorithm, we define messages on all edges. The messages are partial calculation results for marginal distribution calculations. Consider a function node f_0 and a neighboring variable node x_0. If we remove the edge between the nodes f_0 and x_0, the factor graph will become a graph consisting of two unconnected smaller graphs. Consider the graph which contains the function node f_0. Denote the set of function nodes and variable nodes in the graph except the function node f_0 by A. Similarly, denote the set of function nodes and variable nodes in the other graph except the variable node x_0 by B. Denote the product of all local functions in the set A and the function f_0 as $f_A(\cdot)$, while denote the product of all local functions in the set B as $f_B(\cdot)$.

The message on the edge between f_0 and x_0, with a direction from the function node f_0 to the variable node x_0, is defined as,

$$m_{f_0 \to x_0}(s) = \sum_{x_1} \sum_{x_2} \cdots \sum_{x_k} f_A|_{x_0=s}(x_0, x_1, \ldots, x_k), \tag{2.9}$$

where $\{x_1, \ldots, x_k\}$ denotes the set of variable nodes in the set A. The message on the edge between f_0 and x_0, with a direction from the variable node x_0 to the function node f_0, is defined as,

$$m_{x_0 \to f_0}(s) = \sum_{y_1} \sum_{y_2} \cdots \sum_{y_l} f_B|_{x_0=s}(x_0, y_1, \ldots, y_l), \tag{2.10}$$

where $\{y_1, \ldots, y_l\}$ denotes the set of variable nodes in the set B. Each message is a function with a domain identical to the alphabet of the variable x_0. The definition of messages is illustrated in Fig. 2.4.

The message-passing algorithm computes messages in a recursive manner. For a message with a direction from a variable node x_0 to a function node f_0, the message $m_{x_0 \to f_0}(s)$ can be recursively calculated from the messages $m_{f_i \to x_0}(s)$, $i = 1, \ldots, k$, where $f_0, f_1, \ldots, f_k$ are all the neighboring functions nodes of the variable node x_0. It can be checked that

$$m_{x_0 \to f_0}(s) = \prod_{i=1}^{k} m_{f_i \to x_0}(s). \tag{2.11}$$

The message update rule is depicted in Fig. 2.5. Similarly, for a message with a direction from a function node f_0 to a variable node x_0, the message $m_{f_0 \to x_0}(s)$ can be recursively calculated from the messages $m_{x_i \to f_0}$, $i = 1, \ldots, k$, where $x_0, x_1, \ldots, x_k$ are all the neighboring variable nodes of the function node f_0. It can be checked that

$$m_{f_0 \to x_0}(s) = \sum_{x_1, \ldots, x_k} \left(f_0|_{x_0=s}(x_0, x_1, \ldots, x_k) \prod_{i=1}^{k} m_{x_i \to f_0}(x_i) \right). \tag{2.12}$$

The message update rule is depicted in Fig. 2.6. The calculating of all messages can be decomposed into calculating messages with a direction from a leaf of the factor graph in the above recursive manner. The latter ones can be easily calculated directly.

The marginal distributions of each variable node can be determined from messages. Let x_0 be a variable node. Assume that there are k neighboring function

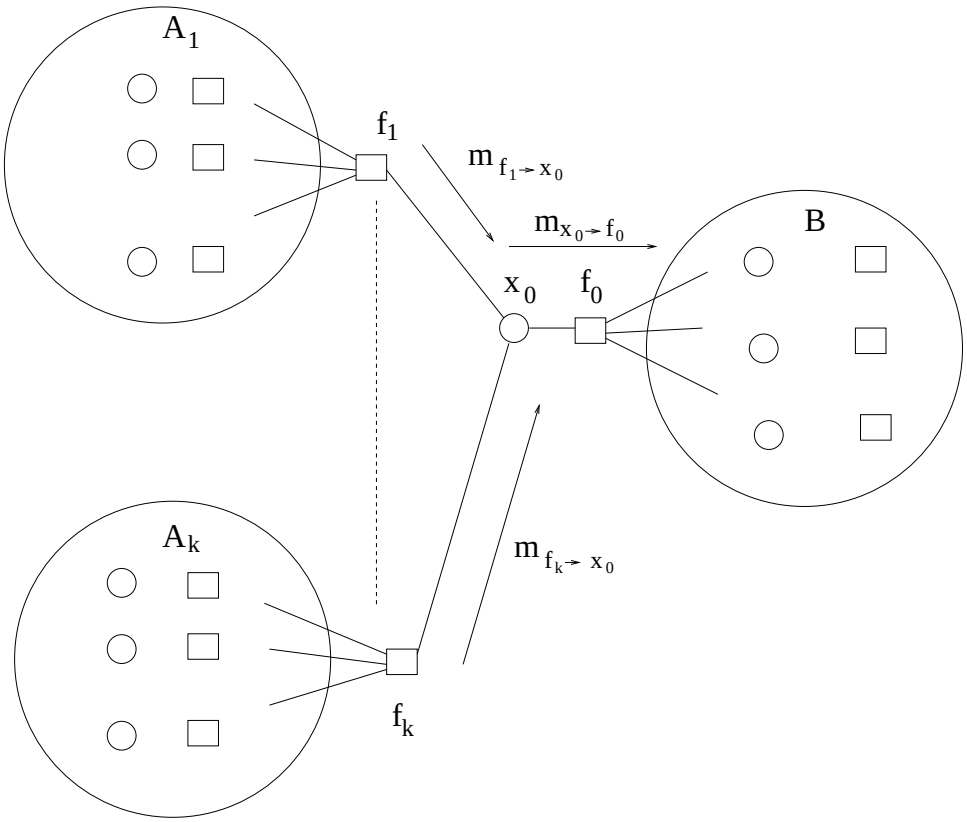


Figure 2.5: The messages update rule for a message with a direction from a variable node to a function node

nodes of x_0. It can be checked that

$$\begin{aligned} f_{X_i}(s) &= \sum_{x_1}\sum_{x_2}\cdots\sum_{x_{i-1}}\sum_{x_{i+1}}\cdots\sum_{x_n} f(x_1,\ldots,x_{i-1},s,x_{i+1},\ldots,x_n) \\ &= \prod_{i=1}^{k} m_{f_i\to x_0}(s). \end{aligned} \tag{2.13}$$

The marginal distribution calculation from messages is depicted in Fig. 2.7.

2.5 Message-Passing Decoding for LDPC Codes

Consider factor graphs for LDPC code decoding, such as the one shown in Fig. 2.3. If the factor graph is acyclic, then the marginal distribution for each codeword

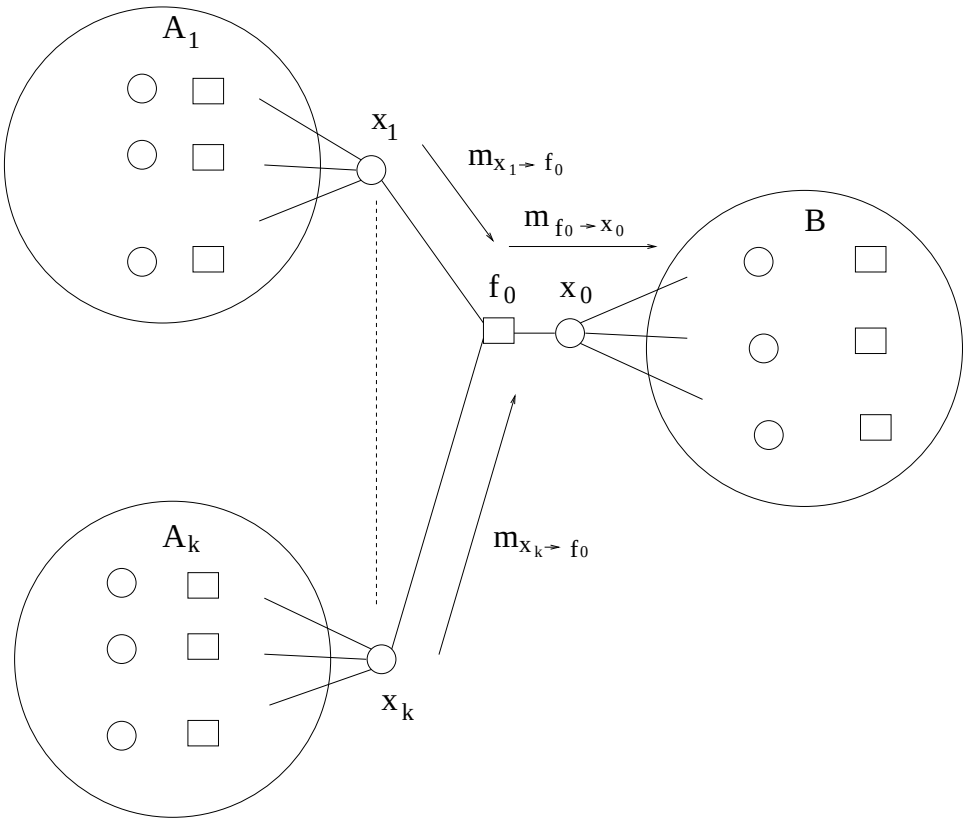


Figure 2.6: The messages update rule for a message with a direction from a function node to a variable node

bit can be calculated by the standard message-passing algorithm. For a variable x_0, the bit-wise MAP (Maximum a Posterior) decoding algorithm outputs 1 if the marginal distribution $\mathbb{P}(x_0 = 1|\boldsymbol{y}_1^n) \geq \mathbb{P}(x_0 = 0|\boldsymbol{y}_1^n)$, and 0 otherwise. Generally speaking, factor graphs for practical LDPC codes are not trees. In these cases, a parallel message-passing decoding procedure with a so called *flooding schedule* is adopted. The decoding algorithm is an instance of loopy propagation.

Assume that the LDPC code has a blocklength n, and m parity-checks. Denote the variable nodes corresponding to codeword bits by x_i, $i = 1 \ldots, n$. Denote the function nodes corresponding to parity-check constraints by f_j, $j = 1, \ldots, m$. Denote the function nodes corresponding to channel conditional probabilities by g_i, $i = 1, \ldots, n$. The message-passing decoding algorithm for the LDPC code usually follows the following flooding schedule for message updating. At each iteration, the algorithm first updates all messages directed from variable nodes x_i to function

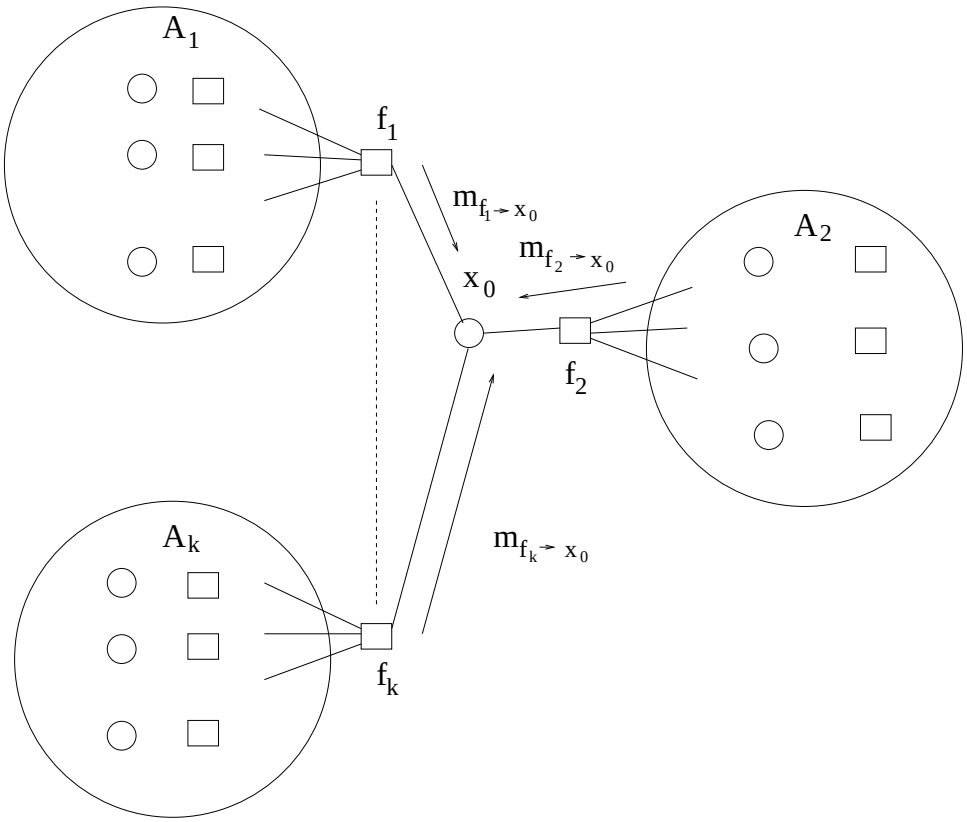


Figure 2.7: Calculating marginal distributions from messages

nodes f_j, and then updates all messages directed from function nodes f_j to variable nodes x_i. The messages directed from function nodes g_i to variable nodes x_i are fixed during all iterations. The messages directed from variable nodes x_i to function nodes g_i are not calculated, because the decoding decisions are independent of these messages. The initial values of messages directed from function nodes f_j to variable nodes x_i are all set to be one. That is, for a message directed from a function node f_j to a variable node x_i,

$$m_{f_j \to x_i}(0) = m_{f_j \to x_i}(1) = 1. \tag{2.14}$$

The initial values of messages directed from function nodes g_i to variable nodes x_i are calculated according to the channel conditional probability distribution. That is, for a message directed from a function node g_i to a variable node x_i,

$$m_{g_i \to x_i}(0) = \mathbb{P}\left(y_i \text{ is received } | 0 \text{ is transmitted}\right) \tag{2.15}$$

$$m_{g_i \to x_i}(1) = \mathbb{P}\left(y_i \text{ is received } | 1 \text{ is transmitted}\right) \tag{2.16}$$

During each iteration, the message updating rule for a message directed from a variable node x_0 to a function node f_0 is as follows:

$$m_{x_0 \to f_0}(s) = m_{g_i \to x_0}(s) \prod_{i=1}^{k} m_{f_i \to x_0}(s), \tag{2.17}$$

where, $f_0, f_1, \ldots, f_k$ are all neighboring function nodes of x_0 that correspond to parity checks. Similarly, the message updating rule for a message directed from a function node f_0 to a variable node x_0 is as follows:

$$m_{f_0 \to x_0}(s) = \sum_{x_1} \sum_{x_2} \cdots \sum_{x_k} \left(f_0(x_0 = s, x_1, \ldots, x_k) \prod_{i=1}^{k} m_{x_i \to f_0}(x_i) \right), \tag{2.18}$$

where, $x_0, x_1, \ldots, x_k$ are all neighboring variable nodes of f_0.

For the special case of binary LDPC decoding, there exist more simple formula for the message updating rules in Eqns. 2.17, and 2.18. We define the log-likelihood ratios for messages as follows,

$$\Lambda_{f_0 \to x_0} = \ln \left(\frac{m_{f_0 \to x_0}(0)}{m_{f_0 \to x_0}(1)} \right) = \log_e \left(\frac{m_{f_0 \to x_0}(0)}{m_{f_0 \to x_0}(1)} \right), \tag{2.19}$$

$$\Lambda_{x_0 \to f_0} = \ln \left(\frac{m_{x_0 \to f_0}(0)}{m_{x_0 \to f_0}(1)} \right) = \log_e \left(\frac{m_{x_0 \to f_0}(0)}{m_{x_0 \to f_0}(1)} \right), \tag{2.20}$$

$$\Lambda_{g_0 \to x_0} = \ln \left(\frac{m_{g_0 \to x_0}(0)}{m_{g_0 \to x_0}(1)} \right) = \log_e \left(\frac{m_{g_0 \to x_0}(0)}{m_{g_0 \to x_0}(1)} \right), \tag{2.21}$$

where, $m_{f_0 \to x_0}$, $m_{x_0 \to f_0}$, $m_{g_0 \to x_0}$ are messages. The hyperbolic trigonometric functions are defined as,

$$\sinh(x) = \frac{e^x - e^{-x}}{2}, \tag{2.22}$$

$$\cosh(x) = \frac{e^x + e^{-x}}{2}, \tag{2.23}$$

$$\tanh(x) = \frac{\sinh(x)}{\cosh(x)} = \frac{e^x - e^{-x}}{e^x + e^{-x}} = \frac{e^{2x} - 1}{e^{2x} + 1}. \tag{2.24}$$

The message updating rules can be simplified into the following well known *sum law* and tanh *law* (see for example [40]). Sketch proofs are provided in order to make this chapter self-contained.

Proposition 2.5.1 *(sum law) For a message directed from a variable node x_0 to a function node f_0, the message update rule can be written as*

$$\Lambda_{x_0 \to f_0} = \Lambda_{g_0 \to x_0} + \sum_{j=1}^{k} \Lambda_{f_i \to x_0}, \tag{2.25}$$

where, g_0 is the neighboring function node that corresponds to the channel conditional probability function, $f_0, f_1, \ldots, f_k$ are all the neighboring function nodes of x_0 that correspond to parity checks.

Proof: By the definition of log-likelihood ratios. ∎

Proposition 2.5.2 *(*tanh *law) For a message directed from a function node f_0 to a variable node x_0, the message update rule can be written as*

$$\tanh\left(\frac{\Lambda_{f_0 \to x_0}}{2}\right) = \prod_{i=1}^{k} \tanh\left(\frac{\Lambda_{x_i \to f_0}}{2}\right), \tag{2.26}$$

where, $x_0, x_1, \ldots, x_k$ are all the neighboring variable nodes of f_0.

Proof: By the definition of the function f_0, we have

$$m_{f_0 \to x_0}(0) = \sum_{x_1,\ldots,x_k : x_1+\ldots+x_k=0} \left(\prod_{i=1}^{k} m_{x_i \to f_0}(x_i)\right), \tag{2.27}$$

$$m_{f_0 \to x_0}(1) = \sum_{x_1,\ldots,x_k : x_1+\ldots+x_k=1} \left(\prod_{i=1}^{k} m_{x_i \to f_0}(x_i)\right). \tag{2.28}$$

Therefore,

$$m_{f_0 \to x_0}(0) - m_{f_0 \to x_0}(1) = \prod_{i=1}^{k} \left(m_{x_i \to f_0}(0) - m_{x_i \to f_0}(1)\right), \tag{2.29}$$

$$m_{f_0 \to x_0}(0) + m_{f_0 \to x_0}(1) = \prod_{i=1}^{k} \left(m_{x_i \to f_0}(0) + m_{x_i \to f_0}(1)\right). \tag{2.30}$$

By the definition of the tanh function,

$$\begin{aligned}\tanh\left(\frac{\Lambda_{f_0\to x_0}}{2}\right) &= \frac{\frac{m_{f_0\to x_0}(0)}{m_{f_0\to x_0}(1)}-1}{\frac{m_{f_0\to x_0}(0)}{m_{f_0\to x_0}(1)}+1} = \frac{m_{f_0\to x_0}(0)-m_{f_0\to x_0}(1)}{m_{f_0\to x_0}(0)+m_{f_0\to x_0}(1)} \\ &= \prod_{i=1}^{k}\frac{m_{x_i\to f_0}(0)-m_{x_i\to f_0}(1)}{m_{x_i\to f_0}(0)+m_{x_i\to f_0}(1)} \\ &= \prod_{i=1}^{k}\tanh\left(\frac{\Lambda_{x_i\to f_0}}{2}\right) \end{aligned} \tag{2.31}$$

The proposition is proven. ∎

For the special cases of BECs, the message update rules can be further simplified. In this case, the log-likelihood ratios for messages can only take three values: $+\infty$, $-\infty$, and 0. The message update rules in Eqns. 2.25, 2.26 can be simplified as follows:

$$\Lambda_{x_0\to f_0} = \begin{cases} +\infty, & \text{one of } \Lambda_{f_i\to x_0} \text{ and } \Lambda_{g_0\to x_0} \text{ is equal to } +\infty \\ -\infty, & \text{one of } \Lambda_{f_i\to x_0} \text{ and } \Lambda_{g_0\to x_0} \text{ is equal to } -\infty \\ 0, & \text{all of } \Lambda_{f_i\to x_0} \text{ and } \Lambda_{g_0\to x_0} \text{ are equal to 0} \end{cases} \tag{2.32}$$

$$\Lambda_{f_0\to x_0} = \begin{cases} +\infty, & \text{all } \Lambda_{x_i\to f_0}\neq 0,\ \text{number of } -\infty \text{ in } \Lambda_{x_i\to f_0} \text{ is even} \\ -\infty, & \text{all } \Lambda_{x_i\to f_0}\neq 0,\ \text{number of } -\infty \text{in } \Lambda_{x_i\to f_0} \text{ is odd} \\ 0, & \text{one of } \Lambda_{x_i\to f_0}=0 \end{cases} \tag{2.33}$$

for each log-likelihood ratios $\Lambda_{x_0\to f_0}$, and $\Lambda_{f_0\to x_0}$ during one iteration. It can be checked that, in this special case, the message-passing algorithm is equivalent to a parallel version of the *edge deletion algorithm* proposed by Luby [23].

2.6 Density Evolution

Density Evolution is an asymptotic method for calculating the limiting numbers of incorrectly decoded bits, after certain decoding iterations, as the blocklengths go to infinity [31]. The computed numbers of incorrectly decoded bits are accurate approximations for these of LDPC codes with long blocklengths.

The theoretical foundations of Density Evolution are concentration Theorems [31]. The concentration Theorems show that, after a given number of decoding

iterations, the fractions of incorrectly decoded bits converge to their limiting values with probabilities approaching one, as the blocklengths go to infinity, where the code is uniformly chosen from a given code ensemble at random. The Theorems are proven for the special case of regular LDPC codes [31]. It can be checked that the Theorems can be trivially generalized to the cases of irregular LDPC codes.

One consequence of the concentration Theorems is an efficient approach to approximately calculating the decoding bit error probabilities. The decoding bit error probabilities are approximated by their limiting values, which can be calculated recursively. This approach of approximately calculating the decoding bit error probabilities is called *Density Evolution*.

The concentration Theorems also imply that the performance of the majority of codes in the code ensemble is close to the ensemble average performance. For a properly designed code ensemble, majority of the codes in the ensemble are codes with satisfactory performance. This analysis justifies the ensemble based approaches to designing fixed codes in practical systems.

For many classes of channels, there exist ordering of the channels, such that the channel with order α_1 can be considered as a physically degraded channel of the channel with order α_2, if $\alpha_1 < \alpha_2$. Examples include Binary Input Additive White Gaussian Noise (BIAWGN) channels, Binary Symmetric Channels (BSC), and BECs. It has been shown that [31], for many classes of channels with such ordering, there exists a critical value $\alpha(\lambda, \rho)$ for each fixed pair of degree distributions $\lambda(x)$, $\rho(x)$, such that

- the ensemble average bit error probabilities of $\mathcal{C}(\lambda, \rho, n)$ approach zero for all channels with order $\alpha > \alpha(\lambda, \rho)$ as the blocklength n go to infinity, and
- the ensemble average bit error probabilities of $\mathcal{C}(\lambda, \rho, n)$ are bounded away from zero for all channel with order $\alpha \leq \alpha(\lambda, \rho)$.

This critical value $\alpha(\lambda, \rho)$ is a very important parameter for a LDPC code ensemble. It is called a *threshold* for the pair of degree distributions $\lambda(x), \rho(x)$ and can be calculated by Density Evolution.

2.7 Density Evolution for BIAWGN Channels

In this section, we will review the Density Evolution for BIAWGN channels. A BIAWGN channel is shown in Fig. 2.8. The input signal X takes values 1 or -1.

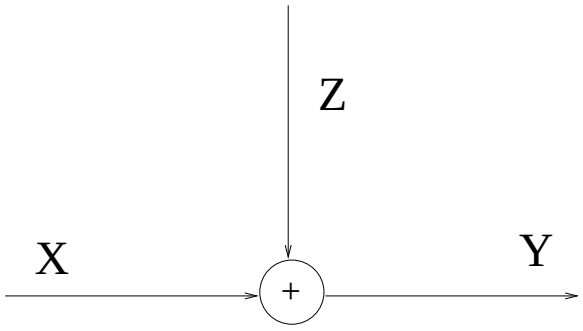


Figure 2.8: Binary input additive white Gaussian noise channels.

The noise Z is white with Gaussian distribution $N(0, \sigma_n^2)$. The received signal is $Y = X + Z$.

We will use the following notation.

- Let g_i, f_j, x_i, $\Lambda_{g_i \to x_i}$, $\Lambda_{x_i \to f_j}$, $\Lambda_{f_j \to x_i}$ be defined as in Section 2.5.
- Let $h^l_{g_i \to x_i}$, $h^l_{x_i \to f_j}$, $h^l_{f_j \to x_i}$ denote the probability density functions of the log-likelihood ratios $\Lambda_{g_i \to x_i}$, $\Lambda_{x_i \to f_j}$, $\Lambda_{f_j \to x_i}$, after l decoding iterations.
- Let $h^l_{g \to x}$, $h^l_{x \to f}$, $h^l_{f \to x}$ denote the probability distributions of messages averaged over all variable nodes and check nodes, after l decoding iterations. It is clear that $h^l_{g \to x}$ corresponds to messages from g_i to x_i, $h^l_{x \to f}$ corresponds to messages from x_i to f_j, and $h^l_{f \to x}$ corresponds to messages from f_j to x_i.

Because BIAWGN channels are symmetric and the LDPC codes are linear block codes, the error probabilities are independent of the transmitted codewords. Hence, in order to simplify the discussions, we can assume that the transmitted codeword is the all zero codeword. We call a message $m_{f_j \to x_i}$ *incorrect* if $\Lambda_{f_j \to x_i} < 0$. The fraction of incorrect messages, after l decoding iterations, is

$$\int_{-\infty}^{0} h^l_{f \to x}(x) dx. \tag{2.34}$$

Therefore, the performance of the code can be determined if we can calculate the distribution $h^l_{f \to x}(x)$.

However, calculating the distribution $h^l_{f \to x}$ or even the asymptotic limit is quite difficult, because $h^l_{f \to x}$ is a function and the computation is in an infinite dimensional space. One potential solution is based on Monte Carlo methods, which have

high computational complexities. There are alternative computationally efficient approaches based on approximating the distribution $h^l_{f\to x}$ by parametric probability distributions, for example, the Gaussian approximation approach [6] and the semi-Gaussian approach [1]. We will review the Gaussian approximation approach to illustrate the basic principle. In Section 2.7.1, we will discuss the so called *symmetric conditions*, which can be used to simplify the computation in the Gaussian approximation approach. The Gaussian approximation method will be reviewed in Section 2.7.2.

2.7.1 Symmetry

In this section, we will show that in Density Evolution for BIAWGN channels the distribution functions are all *symmetric functions*. These results were first proven in [31]. However, our presentation here is slightly different.

Definition: A probability density function $h(x)$ for a log-likelihood ratio $\Lambda_{f\to x}$ is said to be symmetric if the following equation holds,

$$h(-x) = e^{-x}h(x). \tag{2.35}$$

We will need the following notation. Assume X is a random variable. The probability density function of X is $h(x)$. Assume that $h(x)$ is a symmetric function. Define a random variable Z as,

$$Z = -\ln(\tanh(|X/2|)). \tag{2.36}$$

Define the functions F_h^+ and F_h^- as follows:

$$F_h^+(z) = \mathbb{P}(Z \le z, X > 0), \tag{2.37}$$

$$F_h^-(z) = \mathbb{P}(Z \le z, X < 0). \tag{2.38}$$

Define $f_h^+(z)$ and $f_h^-(z)$ as

$$f_h^+(z) = \frac{d}{dz}\left(F_h^+(z)\right), \tag{2.39}$$

$$f_h^-(z) = \frac{d}{dz}\left(F_h^-(z)\right). \tag{2.40}$$

In the sequel, we will use X, Z to denote the random variables, and x, z to denote their realizations.

Proposition 2.7.1 *The function in Eqn, 2.36 is not a one-to-one mapping. Therefore, the function is not invertible. However, there exist the following partial inverse functions (inverse functions by restricting the domain).*

$$X = \begin{cases} -\ln\tanh(Z/2), & \text{if } X > 0 \\ \ln\tanh(Z/2), & \text{if } X < 0 \end{cases} \tag{2.41}$$

Lemma 2.7.2 *There exist the following formula for the functions $f_h^+(z)$, and $f_h^-(z)$.*

$$f_h^+(z) = \frac{h(x)}{\sinh(z)} = \frac{1}{\sinh(z)} h(-\ln\tanh(z/2)), \tag{2.42}$$

where $z = -\ln\tanh(x/2)$, $x > 0$,

$$f_h^-(z) = \frac{h(x)}{\sinh(z)} = \frac{1}{\sinh(z)} h(\ln\tanh(z/2)), \tag{2.43}$$

where $z = -\ln\tanh(-x/2)$, $x < 0$.

Proof: If $x > 0$, $z = -\ln(\tanh(x/2))$, and we have,

$$x = \ln\left(\frac{e^z + 1}{e^z - 1}\right) = -\ln(\tanh(z/2)). \tag{2.44}$$

Define two variables x' and z', such that $x' > 0$, x', z' are functions of each other, and satisfy the following equations,

$$x' = -\ln(\tanh(z'/2)), \tag{2.45}$$

$$z' = -\ln(\tanh(x'/2)). \tag{2.46}$$

By the standard formula for transformations of variables,

$$\begin{aligned} f_h^+(z) &= \left|\left(\left.\frac{dx'}{dz'}\right|_{z'=z}\right)\right| h(x) = \left|\frac{-1}{\sinh(z)}\right| h(x) \\ &= \frac{1}{\sinh(z)} h(-\ln\tanh(z/2)). \end{aligned} \tag{2.47}$$

Similarly, if $x < 0$, $z = -\ln(\tanh(-x/2))$, and we have,

$$x = \ln\left(\frac{e^z - 1}{e^z + 1}\right) = \ln(\tanh(z/2)). \tag{2.48}$$

Define two variables x' and z', such that $x' < 0$, x', z' are functions of each other, and satisfy the following equations,

$$x' = \ln(\tanh(z'/2)), \tag{2.49}$$

$$z' = -\ln(\tanh(-x'/2)). \tag{2.50}$$

By the standard formula for transformations of variables,

$$\begin{aligned} f_h^-(z) &= \left| \left(\left. \frac{dx'}{dz'} \right|_{z'=z} \right) \right| h(x) = \left| \frac{1}{\sinh(z)} \right| h(x) \\ &= \frac{1}{\sinh(z)} h(\ln \tanh(z/2)) \end{aligned} \tag{2.51}$$

∎

Lemma 2.7.3 *Let $h(x)$ be a probability density function. Let the functions $f_h^+(z)$ and $f_h^-(z)$ be defined as in Eqn. 2.39, and 2.40. Then, the density function $h(x)$ is symmetric, if and only if, the functions $f_h^+(z)$ and $f_h^-(z)$ satisfy*

$$\tanh(z/2) f_h^+(z) = f_h^-(z), \quad \textit{for } z > 0. \tag{2.52}$$

Proof: The direction $\Rightarrow$:

$$\begin{aligned} f_h^-(z) &= \frac{1}{\sinh(z)} h(\ln \tanh(z/2)) \\ &\overset{(a)}{=} \frac{1}{\sinh(z)} h(-\ln \tanh(z/2)) \tanh(z/2) \\ &= \tanh(z/2) f_h^+(z), \end{aligned} \tag{2.53}$$

where (a) follows from the fact that $h(x)$ is a symmetric function.

The direction $\Leftarrow$:

In the case that $x > 0$, let us define

$$z = -\ln(\tanh(|x/2|)). \tag{2.54}$$

We have

$$\begin{aligned} h(-x) &\stackrel{(a)}{=} f_h^-(z)\sinh(z) \\ &\stackrel{(b)}{=} f_h^+(z)\sinh(z)\tanh(z/2) \\ &\stackrel{(c)}{=} h(x)\tanh(z/2) \\ &= e^{-x}h(x), \end{aligned} \tag{2.55}$$

where, (a) and (c) follow from Lemma 2.7.2, and (b) follows from the hypothesis.

Similarly, we can show that the $h(-x) = e^{-x}h(x)$, for $x < 0$. ∎

Lemma 2.7.4 *Let $h(x)$, $h_1(x)$, and $h_2(x)$ be probability density functions, such that $h(x) = h_1(x) * h_2(x)$, where $*$ stands for convolution. Suppose that the functions $h_1(x)$ and $h_2(x)$ are symmetric. Then the function $h(x)$ is also symmetric.*

Proof: By the definition of convolution

$$h(x) = \int_{-\infty}^{\infty} h_1(x-y)h_2(y)dy \tag{2.56}$$

We have

$$\begin{aligned} h(-x) &= \int_{-\infty}^{\infty} h_1(-x-y)h_2(y)dy \\ &= \int_{-\infty}^{\infty} h_1(x+y)h_2(y)\exp(-x-y)dy \\ &\stackrel{(a)}{=} \int_{-\infty}^{\infty} h_1(x-z)h_2(-z)\exp(-x+z)dz \\ &= \int_{-\infty}^{\infty} h_1(x-z)h_2(z)\exp(-z)\exp(-x+z)dz \\ &= e^{-x}\int_{-\infty}^{\infty} h_1(x-z)h_2(z)dz = e^{-x}h(x), \end{aligned} \tag{2.57}$$

where (a) follows from the change of variable $y = -z$. ∎

Lemma 2.7.5 *Let x_1, x_2 be two independent random variables. Assume the probability density function of x_1 is $h_1(x_1)$, and the probability density function of x_2 is $h_2(x_2)$. Define x as a random variable such that $\tanh(x/2) = \tanh(x_1/2)\tanh(x_2/2)$. Denote the probability density function of x by $h(x)$. If $h_1(x)$ and $h_2(x)$ are symmetric, then $h(x)$ is symmetric.*

Proof: It can be checked that

$$f_h^+(z) = f_{h_1}^+(z) * f_{h_2}^+(z) + f_{h_1}^-(z) * f_{h_2}^-(z), \tag{2.58}$$

$$f_h^-(z) = f_{h_1}^+(z) * f_{h_2}^-(z) + f_{h_1}^-(z) * f_{h_2}^+(z). \tag{2.59}$$

Because $h_1(x)$ and $h_2(x)$ are symmetric, we have

$$f_{h_1}^-(z) + f_{h_1}^+(z) = \tanh(z/2) f_{h_1}^+(z) + f_{h_1}^+(z) = \frac{2e^z}{e^z+1} f_{h_1}^+(z), \tag{2.60}$$

$$f_{h_1}^-(z) - f_{h_1}^+(z) = \tanh(z/2) f_{h_1}^+(z) - f_{h_1}^+(z) = \frac{-2}{e^z+1} f_{h_1}^+(z), \tag{2.61}$$

$$f_{h_2}^-(z) + f_{h_2}^+(z) = \tanh(z/2) f_{h_2}^+(z) + f_{h_2}^+(z) = \frac{2e^z}{e^z+1} f_{h_2}^+(z), \tag{2.62}$$

$$f_{h_2}^-(z) - f_{h_2}^+(z) = \tanh(z/2) f_{h_2}^+(z) - f_{h_2}^+(z) = \frac{-2}{e^z+1} f_{h_2}^+(z). \tag{2.63}$$

Consequently, we have,

$$\begin{aligned}
\big(f_{h_1}^-(z) + f_{h_1}^+(z)\big) * \big(f_{h_2}^-(z) + f_{h_2}^+(z)\big) &= \int_{-\infty}^{\infty} \frac{2e^t}{e^t+1} f_{h_1}^+(t) \frac{2e^{(z-t)}}{e^{(z-t)}+1} f_{h_2}^+(z-t)dt \\
&= \int_{-\infty}^{\infty} \frac{4e^z}{(e^t+1)(e^{(z-t)}+1)} f_{h_1}^+(t) f_{h_2}^+(z-t)dt \\
&= e^z \int_{-\infty}^{\infty} \frac{4 f_{h_1}^+(t) f_{h_2}^+(z-t)}{(e^t+1)(e^{(z-t)}+1)} dt,
\end{aligned} \tag{2.64}$$

$$\begin{aligned}
\big(f_{h_1}^-(z) - f_{h_1}^+(z)\big) * \big(f_{h_2}^-(z) - f_{h_2}^+(z)\big) &= \int_{-\infty}^{\infty} \frac{-2}{e^t+1} f_{h_1}^+(t) \frac{-2}{e^{(z-t)}+1} f_{h_2}^+(z-t)dt \\
&= \int_{-\infty}^{\infty} \frac{4}{(e^t+1)(e^{(z-t)}+1)} f_{h_1}^+(t) f_{h_2}^+(z-t)dt \\
&= \int_{-\infty}^{\infty} \frac{4 f_{h_1}^+(t) f_{h_2}^+(z-t)}{(e^t+1)(e^{(z-t)}+1)} dt.
\end{aligned} \tag{2.65}$$

Note that

$$f_h^+(z) = \frac{1}{2}\left((f_{h_1}^-(z) + f_{h_2}^+(z)) * (f_{h_2}^-(z) + f_{h_2}^+(z)) + (f_{h_1}^-(z) - f_{h_1}^+(z)) * (f_{h_2}^-(z) - f_{h_2}^+(z))\right)$$
$$= \frac{e^z+1}{2}\int_{-\infty}^{\infty} \frac{4f_{h_1}^+(t)f_{h_2}^+(z-t)}{(e^t+1)(e^{(z-t)}+1)}dt, \tag{2.66}$$

$$f_h^-(z) = \frac{1}{2}\left((f_{h_1}^-(z) + f_{h_1}^+(z)) * (f_{h_2}^-(z) + f_{h_2}^+(z)) - (f_{h_1}^-(z) - f_{h_1}^+(z)) * (f_{h_2}^-(z) - f_{h_2}^+(z))\right)$$
$$= \frac{e^z-1}{2}\int_{-\infty}^{\infty} \frac{4f^+(t)g^+(z-t)}{(e^t+1)(e^{(z-t)}+1)}dt. \tag{2.67}$$

Therefore,

$$f_h^-(z) = \frac{e^z-1}{e^z+1}f_h^+(z) = \tanh(z/2)f_h^+(z) \tag{2.68}$$

The Lemma follows from Lemma 2.7.3. ∎

Theorem 2.7.6 *In the message-passing decoding, if the probability density functions $h_{g\to x}^l$, $i \geq 1$, are symmetric, and the probability density function $h_{x\to f}^l$ is symmetric, then the probability density functions $h_{f\to x}^{l+1}$, and $h_{x\to f}^{l+1}$ are also symmetric.*

Proof: The theorem follows from Lemma 2.7.4, and Lemma 2.7.5. ∎

Theorem 2.7.7 *For BIAWGN channels, the probability density functions $h_{g\to x}^1$, $h_{x\to f}^1$ are symmetric. Furthermore, all density functions $h_{x\to f}^l$, $h_{f\to x}^l$, $l \geq 2$, are symmetric.*

Proof: We will first show that $h_{g\to x}^l$, $l \geq 0$, are symmetric. Let x denote the transmitted signal, y denote the received signal, and σ_n denote the noise variance. By the definition of BIAWGN channels,

$$\mathbb{P}(y|x=-1) = \frac{1}{\sqrt{2\pi}\sigma_n}\exp\left\{-\frac{(y+1)^2}{2\sigma_n^2}\right\}, \tag{2.69}$$

$$\mathbb{P}(y|x=1) = \frac{1}{\sqrt{2\pi}\sigma_n}\exp\left\{-\frac{(y-1)^2}{2\sigma_n^2}\right\}. \tag{2.70}$$

By the definition of log-likelihood ratios,

$$\Lambda_{g\to x} = \ln\left(\frac{\mathbb{P}(y|x=-1)}{\mathbb{P}(y|x=1)}\right) = \frac{-2y}{\sigma_n^2}. \tag{2.71}$$

By the assumption that the all-zero codeword is transmitted, $\Lambda_{g\to x}$ is Gaussian distributed,

$$h^l_{g\to x} = N\left(2/\sigma_n^2, 4/\sigma_n^2\right). \tag{2.72}$$

It can be checked that the above normal distribution is symmetric.

By definition, $h^1_{f\to x} = \delta(x)$. Hence, the density function $h^1_{x\to f}$ is symmetric and equal to $h^1_{g\to x}$. The theorem follows from Theorem 2.7.6. ∎

2.7.2 Gaussian Approximation

As mentioned earlier, the computation of the density functions $h^l_{f\to x}$, and $h^l_{x\to f}$ is in infinite dimensional spaces. The computational complexities of Monte-Carlo approaches are too high, which make these approaches unpractical.

Density Evolution with Gaussian approximation is a reduced-complexity scheme on approximately calculating the probability density functions. In such schemes, the distributions $h^l_{f\to x}$ and $h^l_{x\to f}$ are approximated by Gaussian distributions. Since a Gaussian distribution can be specified by its two parameters, the recursive calculation of distributions in infinite dimensional spaces can be reduced to the recursively calculation of the distribution parameters in finite dimensional spaces.

A first thought on this problem may suggest that two parameters need to be recursively calculated for each distribution. A careful examination shows that the calculation on one parameter for each distribution is sufficient. This is because the distributions are symmetric. If a Gaussian distribution $h(x) = N(m, \sigma^2)$ is symmetric, then $\sigma^2 = 2m$. The distribution can be uniquely determined by its mean value.

Let us denote the mean of the log-likelihood ratios from check nodes to variables nodes at l-th iteration by $\mu^l_{c\to v}$. For an irregular LDPC code with the left degree sequence $\lambda(x) = \sum_i \lambda_i x^{i-1}$ and right degree sequence $\rho(x) = \sum_j \rho_j x^{i-1}$. The mean $\mu^l_{c\to v}$ can be calculated recursively as follows [6],

$$\mu^{l+1}_{c\to v} = \sum_{j=2}^{d_c} \rho_j \phi^{-1}\left(1 - \left[1 - \sum_{i=2}^{d_c} \lambda_i \phi\left(\mu_{g\to v} + (i-1)\mu^l_{c\to v}\right)\right]^{j-1}\right), \tag{2.73}$$

where, d_v and d_c are the maximal left and right degrees, $\mu_{g\to v}$ is the mean of the log-likelihood ratios of messages from channel output nodes to variable nodes, and the function $\phi(x)$ is defined as,

$$\phi(x) = \begin{cases} 1 - \frac{1}{\sqrt{2\pi x}} \int_{-\infty}^{\infty} \tanh(\frac{u}{2}) e^{-\frac{(u-x)^2}{4x}} du, & x > 0 \\ 1, & x = 0 \end{cases} \tag{2.74}$$

2.8 Density Evolution for BECs

In the message-passing decoding for BECs, the log-likelihood ratios $\Lambda_{c\to v}$ and $\Lambda_{v\to c}$ can only take values ∞, $-\infty$ or 0. A message is said to be an erasure, if the corresponding log-likelihood ratio is 0. Let us consider the irregular LDPC codes with the left degree sequence $\lambda(x) = \sum_i \lambda_i x^{i-1}$ and right degree sequence $\rho(x) = \sum_j \rho_j x^{j-1}$. Let $p^{(l)}_{e,c\to v}$ ($p^{(l)}_{e,v\to c}$) denote the probability that a message from a check node to a variable node (a variable node to a check node) is erasure along a randomly chosen edge during the l-th iteration. The erasure probability $p^{(l)}_{e,c\to v}$ can be calculated as,

$$\begin{aligned} p^{(l)}_{e,c\to v} =& \sum_{j=2}^{d_c} \mathbb{P}(c \text{ has degree } j) \mathbb{P}\left(m^{(l)}_{c\to v} \text{ is erasure} \mid \text{the degree of } c \text{ is } j\right) \\ =& \sum_{j=2}^{d_c} \rho_j \left\{1 - [1 - p^{(l)}_{e,v\to c}]^{j-1}\right\} = 1 - \sum_{j=2}^{d_c} \rho_j \left[1 - p^{(l)}_{e,v\to c}\right]^{j-1}. \end{aligned} \tag{2.75}$$

The erasure probability $p^{(l)}_{e,v\to c}$ can be calculated as,

$$\begin{aligned} p^{(l)}_{e,v\to c} =& \sum_{i=2}^{d_v} \mathbb{P}(v \text{ has degree } i) \mathbb{P}(m_{v\to c} \text{ is erasure} \mid \text{the degree of } v \text{ is } i) \\ =& \xi \sum_{i=2}^{d_v} \lambda_i \left[p^{(l-1)}_{e,c\to v}\right]^{i-1}, \end{aligned} \tag{2.76}$$

where ξ is the channel parameter.

In summary,

$$p^{(l)}_{e,v\to c} = \xi \sum_{i=2}^{d_v} \lambda_i \left\{1 - \sum_{j=2}^{d_c} \rho_j \left[1 - p^{(l-1)}_{e,v\to c}\right]^{j-1}\right\}^{i-1} = \xi\lambda\left(1 - \rho\left(1 - p^{i-1}_{e,v\to c}\right)\right) \tag{2.77}$$

The code can be successfully decoded with high probability if and only if

$$\xi\lambda\left(1-\rho\left(1-x\right)\right) < x, \tag{2.78}$$

for all $0 < x \leq \xi$.

2.9 Finite-Length Analysis

Density Evolution is a very powerful asymptotic analysis method for LDPC codes. However, the method has certain limitations. In particular, the error probabilities of finite-length codes can not be determined by the asymptotic analysis (although the limiting error probabilities as the block length goes to infinity can be determined by Density Evolution).

The finite-length methods denote several recently proposed approaches on estimating the error probabilities of finite-length codes. These finite-length methods relate the error probabilities to codeword-like structures, such as stopping sets, trapping sets, and pseudo-codewords. Therefore, error probabilities of finite-length codes can be determined by these analysis methods. Finite-length analysis methods are extremely powerful in dealing with *error-floor problems*, which will be discussed extensively in Chapter 4.

Since the first finite-length analysis method was proposed by Di, Proietti, Telatar, Richardson, and Urbanke [10], many similar analysis methods have been proposed and some of them are quite involved. It is not realistic to present all these methods in detail. We refer interested readers to [12] [27] [33] for the discussions on trapping sets and pseudo-codewords; and [4] [11] [9] [21] [20] [29] for the corresponding weight distribution calculations.

In Chapter 4, we will present a stopping set analysis of the proposed code construction scheme. The definition of stopping sets is given as follows. More detailed discussions will be presented in the context.

Definition: A nonempty variable node set $\mathcal{S}$ is called a *stopping set* if every check node in the Tanner graph is not singly connected to the variable nodes in $\mathcal{S}$. We define the weight of a stopping set $\mathcal{S}$ to be its cardinality.

Chapter 3

Designing LDPC Codes with Fast Decoding Speeds

In this chapter, we consider the problem of designing LDPC codes with fast message-passing decoding speeds. By *fast decoding speeds*, we mean that the bit error probabilities drop significantly after only a small number of decoding iterations in message-passing decoding. Such codes are desirable, because they have less decoding computational complexities and decoding delay.

A direct approach to finding LDPC codes with fast decoding speeds is solving a constrained optimization problem. The optimization variables are the code parameters. The objective function is the number of decoding iterations, which can be approximately calculated by Density Evolution. The constraints include the fixed code rate, the condition ensuring that the code can be successfully decoded, and the valid ranges of the code parameters. However, the direct approach is difficult analytically, because the objective function is discrete and not differentiable. All local searching based optimization algorithms fail, because local searching is not efficient on high-dimensional discrete functions. The global optimization algorithms are available only for optimization problems with certain structures, which are unfortunately not present in our optimization problem.

In this chapter, we propose an alternative solution to the problem. We present an asymptotic approximation to the number of decoding iterations. The asymptotic approximation is differentiable. The asymptotic approximation results in an approximate optimization approach to designing codes with good decoding speed. Instead of minimizing the number of decoding iterations directly, the approximate optimization approach minimizes the asymptotic approximation. Numerical results

confirm that the number of decoding iterations and its asymptotic approximation are consistent.

One essential ingredient of proving the asymptotic approximation is the satisfaction of the *flatness condition*. We prove that *density-efficient* and *capacity-approaching* LDPC codes satisfy a so called *flatness condition*. By *capacity-approaching*, we mean that the code rate is close to the channel capacity. By *density-efficient*, we mean that the density of the parity-check matrix is low. In this chapter, we only consider codes with efficiently low parity-check matrix density and low maximal left and right degrees. The codes with high parity-check matrix density or high maximal degrees are not practical in implementations.

The proposed asymptotic approximation provides a closed-form relation between the number of decoding iterations and the code parameters. This closed-form relation make it possible for further in-depth discussions on decoding speeds. Based on this closed-form relation, we investigate the conditions for the optimal degree distributions in the sense of decoding speeds. Based on numerical observations, we conjecture that the optimal codes, in terms of decoding speeds, are right-concentrated. Here, *right-concentrated* means that the degrees of check nodes concentrate around the average right degree. Using the asymptotic approximation, we show that the conjecture is true for certain special cases.

There is much research in the literature related to LDPC decoding complexity. Tradeoffs between density and performance have been investigated in [13], [17], [34], and [35]. Optimization frameworks for decoding complexity are also discussed in [37] [41]. However, it should be noted that, in this chapter, the considered problem is significantly different from these in the previous discussions. We consider minimizing the number of decoding iterations as the sole design objective, which is crucial in the target application scenarios. To our best knowledge, this problem has not been considered previously in the literature.

The remainder of this chapter is organized as follows. In Section 3.1, we present preliminary materials, which will be used in the later discussions in this chapter. In Section 3.2, we discuss the flatness condition and asymptotic approximation to the number of decoding iterations. In Section 3.3, we discuss the proposed approximate optimization approach for finding codes with good decoding speeds. In Section 3.4, we present numerical examples of designing codes with fast decoding speeds. In Section 3.5, we discuss the conditions for optimal degree distributions in the sense of decoding speeds. Some conclusions for this chapter are presented in Section 3.6.

3.1 Preliminary

In this chapter, we consider randomly generated LDPC codes. A Richardson-Urbanke ensemble $\mathcal{C}(\lambda, \rho, n)$ has three parameters: the left degree distribution $\lambda(x)$, the right degree distribution $\rho(x)$, and the codeword length n. The Tanner graph of the random code is generated by growing edges from variable and check nodes according to the degree distributions. If the codeword length is sufficiently long, the design code rate is

$$R = 1 - \frac{\int_0^1 \rho(x)dx}{\int_0^1 \lambda(x)dx}. \tag{3.1}$$

In the case of BECs, the message erasure probability $P_e^{(l)}$ after the l-th iteration can be approximately calculated by Density Evolution as in Eqn, 2.76. Asymptotically with the codeword length, the code can be successfully decoded with high probability if and only if

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

Previous research has shown that LDPC codes with bounded degrees can not achieve the channel capacity. There exists a tradeoff between the parity-check matrix density and the achievable rate [13] [34]. In the case of BEC, Shokrollahi [35] showed the following bound for achievable rate:

$$(1 - \xi)^a \leq \frac{\Delta R}{\xi + \Delta R}, \tag{3.3}$$

where, a is the average right degree,

$$1/a = \int_0^1 \rho(x)dx, \tag{3.4}$$

and ΔR is the gap between the achievable rate to the channel capacity. This lower bound of the gap to the capacity decreases exponentially as the average degree increases. In the same paper, Shokrollahi also showed that this bound is tight. That is, there exist codes with an exponentially decreasing gap to the capacity with respect to linearly growing average degrees.

3.2 Flatness Condition and Asymptotic Approximation to the Number of Iterations

In this section, we prove that density-efficient capacity-approaching LDPC codes satisfy a necessary condition - the flatness condition. Based on the flatness condition, we further derive an asymptotic approximation to the number of decoding iterations.

3.2.1 Notation and Definition

Consider a BEC with channel parameter ξ. Consider a LDPC code with left degree distribution $\lambda(x)$ and right degree distribution $\rho(x)$. Denote the average right degree by a. Define $b = 1/a$. Let R denote the code rate. The gap between the capacity and the code rate is $\Delta R = C - R$. We define a function $B(\Delta R, b, x)$ as follows:

$$B(\Delta R, b, x) = \sqrt{\frac{2\Delta R}{(\xi + \Delta R)(1-\xi)\rho(1-x)}}. \tag{3.5}$$

We define the *decoding convergence time* T_η to be the maximal l such that the message erasure probability is greater than the probability level η after l decoding iterations. For any left and right degree distributions $\lambda(x)$ and $\rho(x)$, we define

$$F(\lambda, \rho, \eta) \triangleq \int_\eta^\xi \frac{dx}{x - \xi\lambda(1-\rho(1-x))}. \tag{3.6}$$

Throughout Section 3.2, the derivatives are taken with respect to x.

3.2.2 Main Theorems

Let us consider a BEC with channel parameter ξ. Consider a sequence of degree distribution pairs:

$$(\lambda_1, \rho_1), \cdots, (\lambda_n, \rho_n), \cdots \tag{3.7}$$

where,

$$\lambda_n(x) = \sum_i \lambda_{ni} x^{i-1}, \tag{3.8}$$

$$\rho_n(x) = \sum_j \rho_{nj} x^{j-1}, \tag{3.9}$$

are respectively the left and right degree distributions satisfying the successfully decoding condition $\xi\lambda_n(1-\rho_n(1-x)) < x$, for all $x \in (0, \xi]$. Each pair of degree distributions defines a code ensemble. Let a_n denote the average right degree for the n-th code. Define $b_n = 1/a_n$. Let R_n denote the rate of the n-th code. Let ΔR_n denote the gap between the capacity and the code rate for the n-th code, $\Delta R_n = C - R$. Further assume that the maximal degrees of λ_n and ρ_n are upper bounded by $k_v a_n$ and $k_c a_n$ respectively, where k_v and k_c are constants. Then we have the following theorem.

Theorem 3.2.1 *If b_n is strictly decreasing with $\lim_{n\to\infty} b_n = 0$, and ΔR_n is strictly decreasing with*

$$\lim_{n\to\infty} \frac{\Delta R_n}{(b_n)^{15}} \to 0. \tag{3.10}$$

Then, the derivative of $\xi\lambda_n(1-\rho_n(1-x))$ with respect to x converges to 1 uniformly with respect to x in the interval $(0, \xi]$, as $n \to \infty$.

Proof: The proof is in Appendix 3.A.

According to this theorem, the function $\xi\lambda(1-\rho(1-x))$ becomes flat as the code rate approaches the channel capacity and the parity-check matrix density remains sufficiently low. Based on this conclusion, we prove the following theorem, which shows an asymptotic approximation to the decoding convergence time T_η.

Theorem 3.2.2 *Let η be a fixed probability level, $0 < \eta < \xi$. Suppose that b_n is strictly decreasing with $\lim_{n\to\infty} b_n = 0$, and ΔR_n is strictly decreasing with*

$$\lim_{n\to\infty} \frac{\Delta R_n}{(b_n)^{15}} \to 0. \tag{3.11}$$

Then, as $n \to \infty$, the ratio

$$F(\lambda_n, \rho_n, \eta)/T_\eta \tag{3.12}$$

goes to 1.

Proof: The proof is in Appendix 3.B.

Remark In the above theorems, we assume that ΔR_n decrease polynomially with respect to b_n. According to Section 3.1, in the most efficient tradeoffs, ΔR_n decreases exponentially with respect to b_n. Therefore, the conditions 3.10, 3.11 are not stringent.

3.3 Approximate Optimization

A brute force approach to finding the LDPC code with minimal decoding convergence time T_η and a fixed gap to the capacity is solving the following constrained optimization problem:

$$\min T_\eta(\lambda, \rho, \xi) \tag{3.13}$$

subject to

$$\xi\lambda(1-\rho(1-x)) < x, \text{ for } x \in (0, \xi] \tag{3.14}$$

$$\frac{\int_0^1 \rho(x)dx}{\int_0^1 \lambda(x)dx} = 1 - C + \Delta R \tag{3.15}$$

$$\sum_i \lambda_i = 1, \ \sum_j \rho_j = 1, \ \lambda_i \geq 0, \text{ and } \rho_j \geq 0 \tag{3.16}$$

$$\lambda_i = 0 \text{ for all } i > d_v, \quad \rho_j = 0 \text{ for all } j > d_c \tag{3.17}$$

In the above, C is the channel capacity, and $0 < \Delta R < C$ is a fixed gap to the channel capacity. The condition in Eqn. 3.14 is the successful decoding condition. The condition in Eqn. 3.15 imposes that the code rate is $C - \Delta R$. The constraints in Eqn. 3.16 come from the probability nature of degree distributions. The conditions in Eqn. 3.17 impose that the maximal left degree is at most d_v, and the maximal right degree is at most d_c.

However, the above optimization problem is not tractable. The objective function T_η is not differentiable. This brings in convergence problems for optimization algorithms. To get around these difficulties, at this point, we invoke Theorem 3.2.2 and replace T_η by $F(\lambda, \rho, \eta)$. This yields the following approximate optimization problem:

$$\min F(\lambda, \rho, \eta) \tag{3.18}$$

subject to

$$\xi\lambda(1-\rho(1-x)) < x, \quad \text{for } x \in (0, \xi] \tag{3.19}$$

$$\frac{\int_0^1 \rho(x)dx}{\int_0^1 \lambda(x)dx} = 1 - C + \Delta R \tag{3.20}$$

$$\sum_i \lambda_i = 1, \ \sum_j \rho_j = 1, \ \lambda_i \geq 0, \text{ and } \rho_j \geq 0 \tag{3.21}$$

$$\lambda_i = 0 \text{ for all } i > d_v, \ \rho_j = 0 \text{ for all } j > d_c \tag{3.22}$$

Because of the nice numerical properties of the objective function, the approximate optimization problem can be efficiently solved by most local searching based algorithms. We use a sequential quadratic programming algorithm [3] to solve the optimization. Another alternative optimization algorithm with similar performance is also adopted. In this alternative optimization algorithm, the program in Eqns. 3.18, 3.19, 3.20, 3.21, 3.22 is solved by iteratively fixing one of the degree distributions $\lambda(x)$ and $\rho(x)$ and optimizing the other degree distribution. The numerical results will be presented in Section 3.4.

3.4 Numerical Result

In this section, we present numerical examples of designing codes with fast decoding speeds for different rates, and channel parameters. The purpose of these numerical examples is to confirm our theoretical discussions. In all the numerical examples, the probability level η is chosen to be 10^{-3}.

Example 1: We design an LDPC code for a BEC with parameter $\xi = 0.48$. The code rate is 0.48. The left and right degree distributions are as follows:

$$\lambda(x) = 0.1863x + 0.4143x^2 + 0.0512x^8 + 0.3482x^{15} \tag{3.23}$$

$$\rho(x) = 0.5330x^6 + 0.4670x^7 \tag{3.24}$$

The function $\xi\lambda(1-\rho(1-x))$ is shown in Fig. 3.1 as the solid line. The dashed line shows the straight line $y = x$. The two functions are also shown in Fig. 3.2 on a log scale. The decoding convergence time T_η is 47. The asymptotic approximation $F(\lambda, \rho, \eta) = 47.9400$. As predicted by the theoretical discussions, $T_\eta \approx F(\lambda, \rho, \eta)$. We also construct practical codes according to the designed degree distributions. The codeword length is $32k$ bits (i.e., the codeword length is $32 \times 1024 = 32768$

bits). The simulation results on bit erasure probabilities are shown in Fig. 3.3. The bit erasure probabilities after different numbers of iterations are shown as the dash-dot curve. The bit erasure probabilities approximately calculated by density evolution are shown as the solid curve. The results show that the approximate bit error probability calculation by density evolution is accurate. The decoding convergence time is 44 for the practical code, which has been accurately predicted by density evolution.

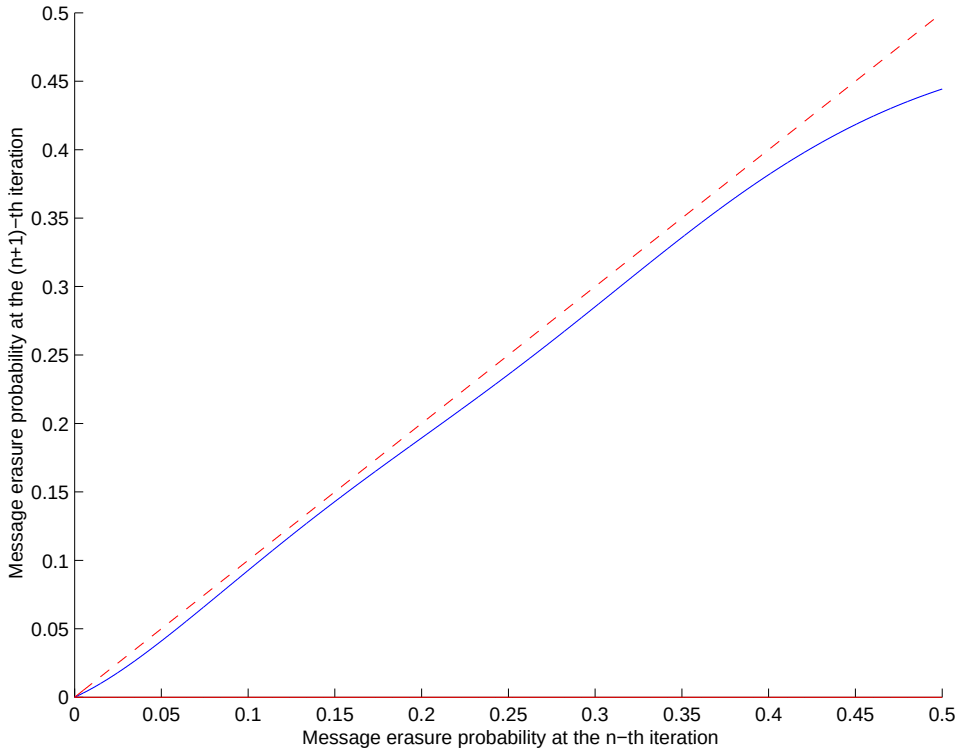


Figure 3.1: The function $\xi\lambda(1-\rho(1-x))$ in the Example 1

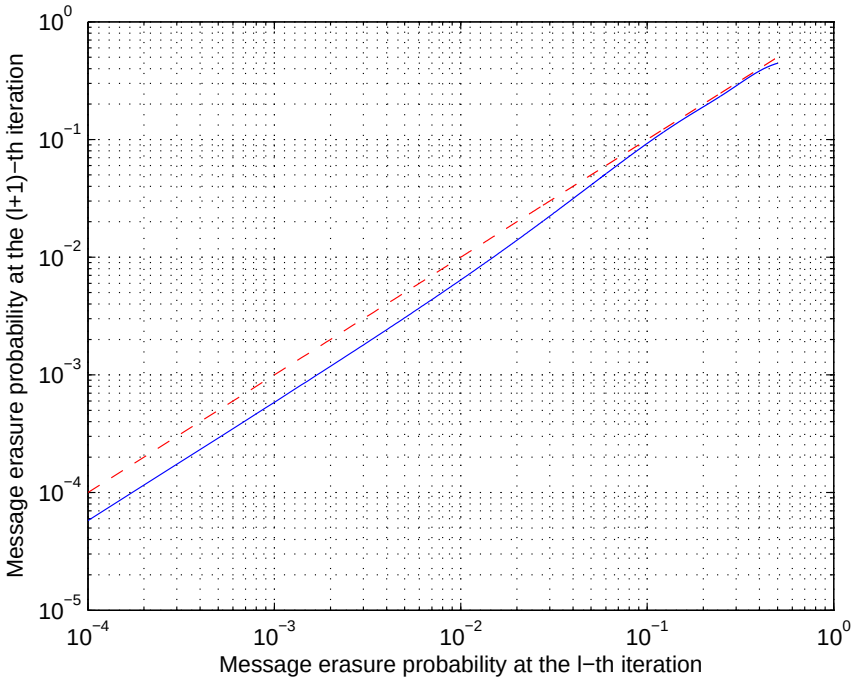


Figure 3.2: The function $\xi\lambda(1-\rho(1-x))$ in the Example 1

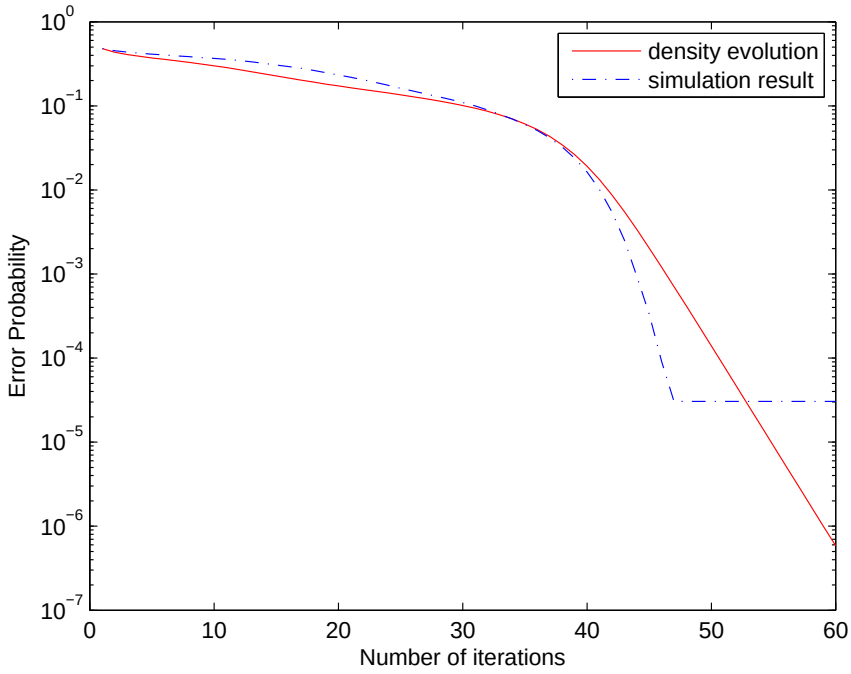


Figure 3.3: The bit erasure probabilities in Example 1, the X-axis shows the number of iterations, the Y-axis shows the bit erasure probabilities at each iteration.

Example 2: We design an LDPC code for the BEC with parameter $\xi = 0.48$. The code rate is 0.5. The left and right degree distributions are as follows:

$$\lambda(x) = 0.2452x + 0.2982x^2 + 0.1112x^5 + 0.3454x^{15} \tag{3.25}$$

$$\rho(x) = 0.3398x^6 + 0.6602x^7 \tag{3.26}$$

The function $\xi\lambda(1-\rho(1-x))$ is shown in Fig. 3.4 as the solid line. The dash line shows the straight line $y = x$. The two functions are also shown in Fig. 3.5 on a log scale. The decoding convergence time $T_\eta = 107$. The asymptotic approximation $F(\lambda, \rho, \eta) = 108.7363$. $T_\eta \approx F(\lambda, \rho, \eta)$. We construct practical codes according to the above degree distributions. The codeword length is $32k$ bits. The simulation results on bit erasure probabilities after different numbers of iterations are shown in Fig. 3.6 as the dash-dot curve. The bit erasure probabilities approximately calculated by density evolution are shown as the solid curve. The decoding convergence time for the practical code is 100.

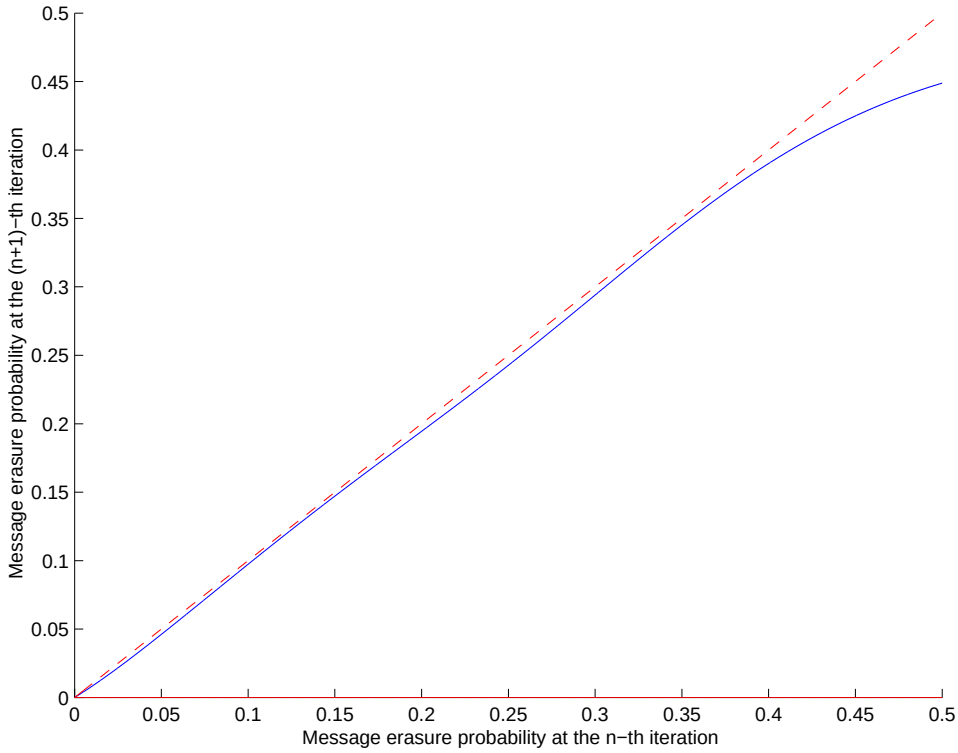


Figure 3.4: The function $\xi\lambda(1-\rho(1-x))$ in the Example 2

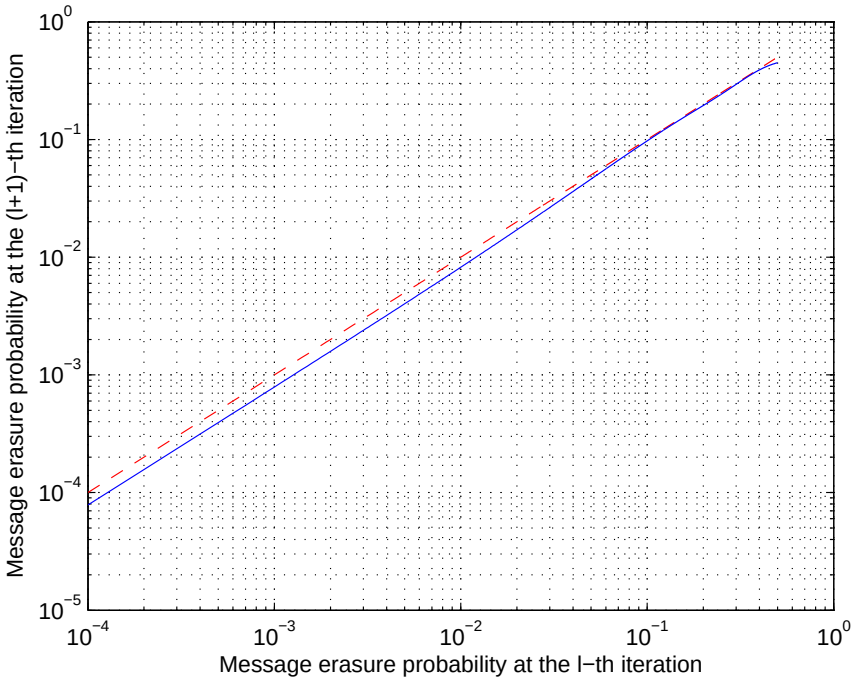


Figure 3.5: The function $\xi\lambda(1-\rho(1-x))$ in the Example 2

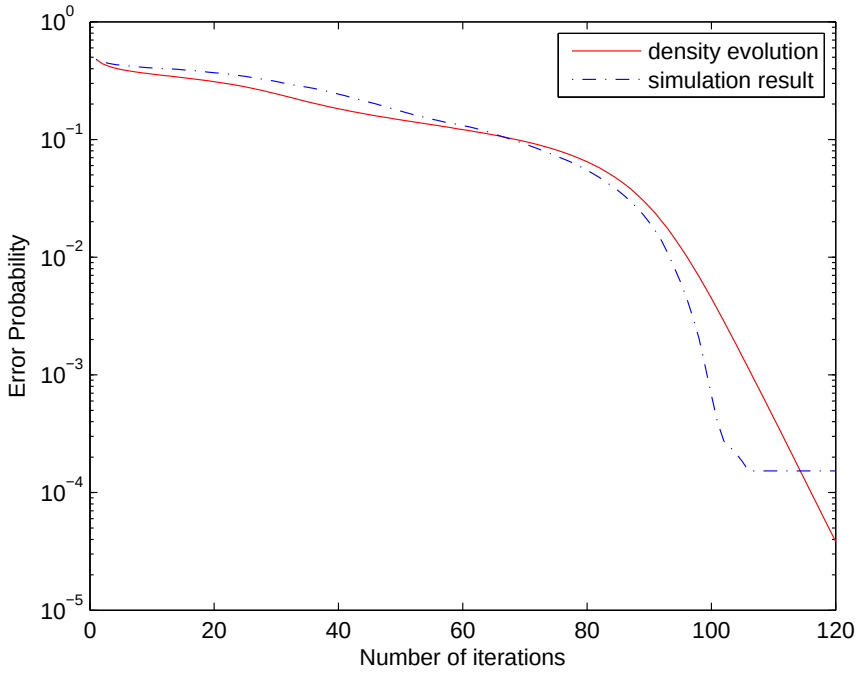


Figure 3.6: The message erasure probabilities in the Example 2, the X-axis shows the number of iterations, the Y-axis shows the message erasure probabilities at each iteration.

Example 3: In this numerical example, we compare the code ensembles with fast decoding speeds by the proposed approach with certain previously well-known code ensembles. The purpose of this numerical example is to show that capacity-approaching codes can have significantly different performance in terms of decoding speeds. The considered channel is a BEC with channel parameter $\xi = 0.46$. A code ensemble with fast decoding speeds is designed by the proposed approach. The rate is 0.5018. The left and right degree distributions are as follows:

$$\lambda(x) = 0.1819x + 0.4101x^2 + 0.0152x^7 + 0.3928x^{15} \tag{3.27}$$

$$\rho(x) = 0.0891x^6 + 0.9109x^7 \tag{3.28}$$

We compare the codes designed by the proposed approach with the *Heavy-tail/Poisson* code ensembles, which are previously well-known capacity-approaching

ensembles [23] [36]. The code ensemble has rate 0.5017. The left and right degree distributions are as follows [36] :

$$\begin{aligned}\lambda(x) =& 0.3014x + 0.1507x^2 + 0.1005x^3 + 0.0753x^4 + 0.0604x^5 + \\ & 0.0502x^6 + 0.0431x^7 + 0.0377x^8 + 0.0335x^9 + 0.0301x^{10} + \\ & 0.0274x^{11} + 0.0251x^{12} + 0.0232x^{13} + 0.0215x^{14} + 0.0201x^{15}\end{aligned} \tag{3.29}$$

$$\begin{aligned}\rho(x) =& 0.0060x + 0.0213x^2 + 0.0502x^3 + 0.0887x^4 + 0.1255x^5 + \\ & 0.1479x^6 + 0.1495x^7 + 0.1321x^8 + 0.1039x^9 + 0.0735x^{10} + \\ & 0.0472x^{11} + 0.0278x^{12} + 0.0151x^{13} + 0.0077x^{14} + 0.0036x^{15}\end{aligned} \tag{3.30}$$

The decoding convergence time T_η by density evolution is 47 for the code by the proposed approach and 263 for the Heavy-tail/Poisson codes.

We construct practical codes from the two code ensembles with various block-lengths. We depict the bit error probabilities of the codes in Figs. 3.7, 3.8, 3.9, 3.10, where the two codes in each figure have the same block length. The decoding convergence times for practical codes in the code ensemble with degree distributions in Eqns 3.27 and 3.28 are approximately 40. The decoding convergence times for practical codes in the Heavy-tail/Poisson code ensemble are approximately $120 - 160$. The numerical results show that the codes by the proposed design approach have significantly faster decoding speeds compared with the codes from the Heavy-tail/Poisson ensemble. This example shows that capacity-approaching codes can have significantly different performance in terms of decoding speeds.

In this numerical example, the decoding convergence times for practical codes from Heavy-tail/Poisson ensembles are not accurately predicted by the density evolution approach. We believe that this is because density evolution is more accurate for estimating bit error probabilities after a small number of decoding iterations and less accurate for estimating bit error probabilities after a large number of decoding iterations. In our design framework, we consider density evolution as a reliable method for estimating bit error probabilities, because we only consider codes with fast decoding speeds. In other words, the decoding convergence times of the codes with fast decoding speeds can be accurately estimated by density evolution, because their bit error probabilities approach zero quickly after only small numbers of decoding iterations.

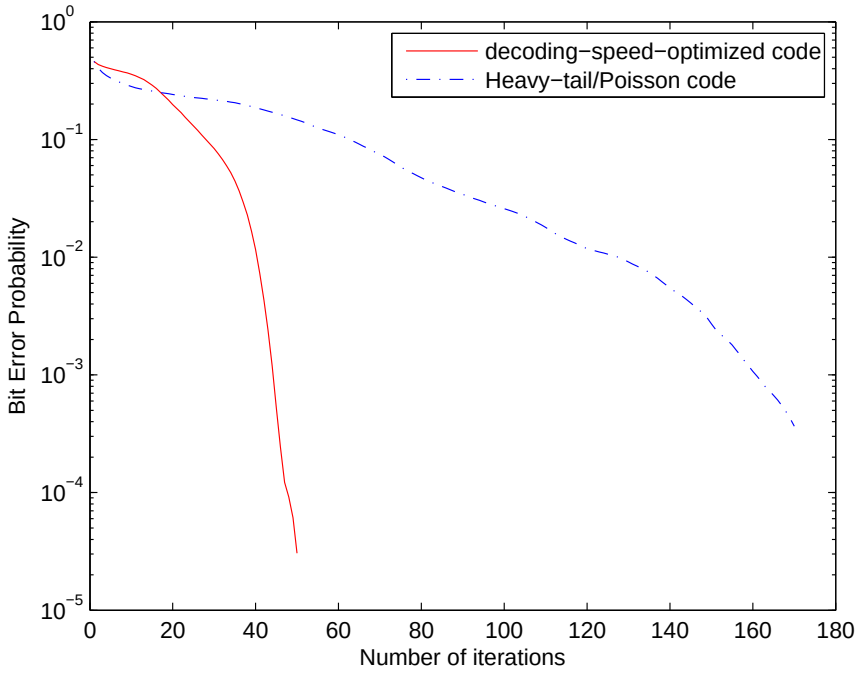


Figure 3.7: The bit error probabilities of two codes in Example 3, the X-axis shows the number of iterations, the Y-axis shows the bit error probabilities at each iteration. The blocklengths are $32k$.

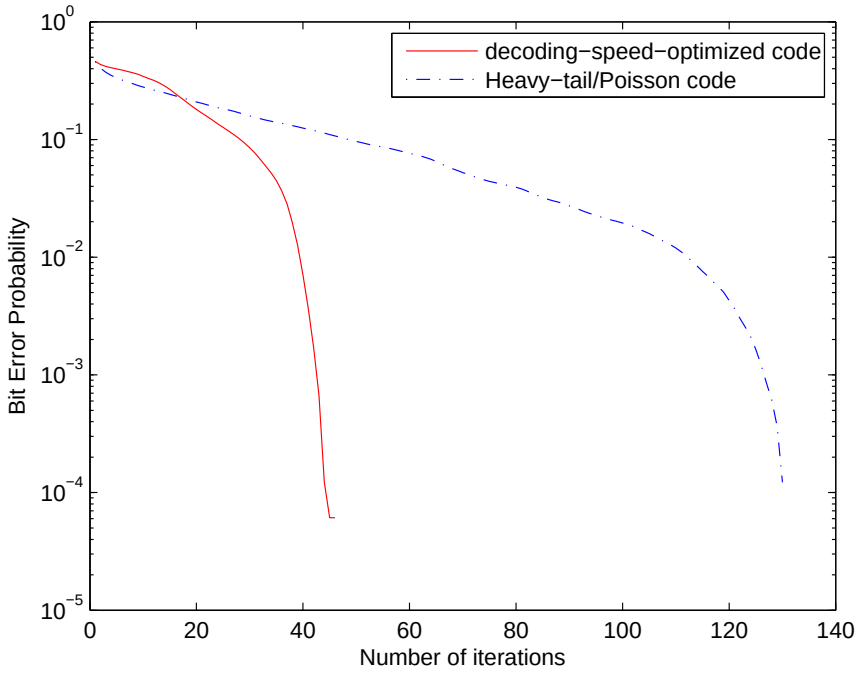


Figure 3.8: The bit error probabilities of two codes in Example 3, the X-axis shows the number of iterations, the Y-axis shows the bit error probabilities at each iteration. The blocklengths are $16k$.

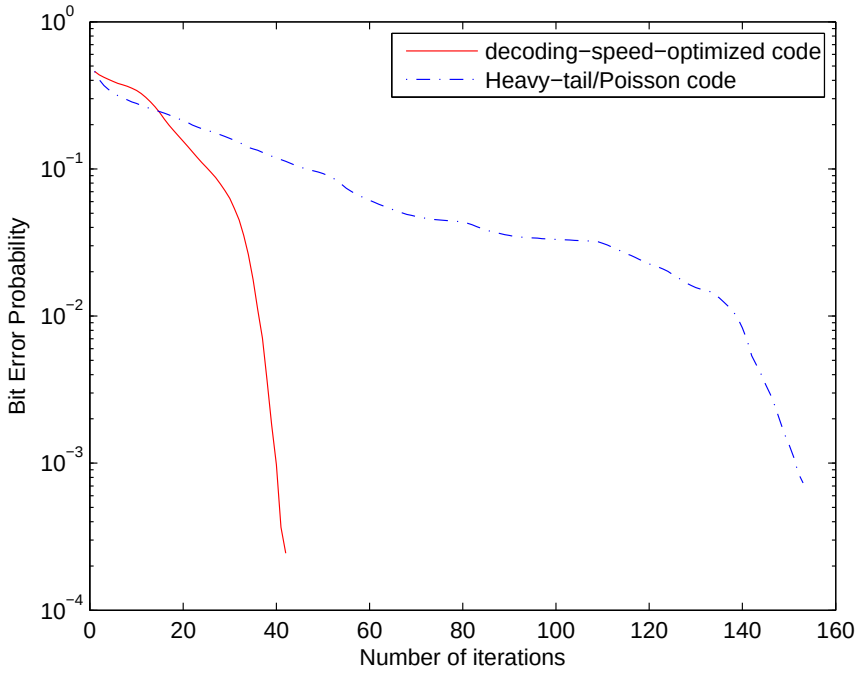


Figure 3.9: The bit error probabilities of two codes in Example 3, the X-axis shows the number of iterations, the Y-axis shows the bit error probabilities at each iteration. The blocklengths are $8k$.

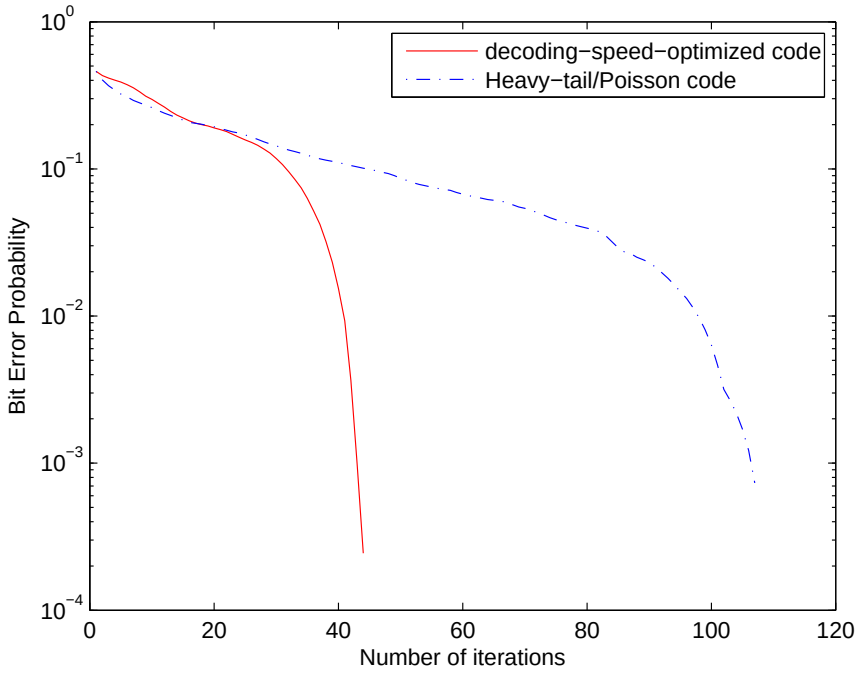


Figure 3.10: The bit error probabilities of two codes in Example 3, the X-axis shows the number of iterations, the Y-axis shows the bit error probabilities at each iteration. The blocklengths are $4k$.

Some more examples of designing codes with fast decoding speeds are summarized in Tables 3.1, 3.2, 3.3, 3.4, 3.5, 3.6, 3.7, 3.8, 3.9. For a BEC with channel parameter $\xi = 0.5$, we design three code ensembles with rates 0.46, 0.47, and 0.48. The left degree distributions are shown in Table 3.1. The right degree distributions are shown in Table 3.2. The decoding convergence time and the asymptotic approximation are shown in Table 3.3. For a BEC with channel parameter $\xi = 0.75$, we design three code ensembles with rates 0.21, 0.22, and 0.23. The left degree distributions are shown in Table 3.4. The right degree distributions are shown in Table 3.5. The decoding convergence time and the asymptotic approximation are shown in Table 3.6. For a BEC with channel parameter $\xi = 0.25$, we design three code ensembles with rates 0.71, 0.72, and 0.73. The left degree distributions are shown in Table 3.7. The right degree distributions are shown in Table 3.8. The decoding convergence time and the asymptotic approximation are shown in Table 3.9. These numerical examples show that the asymptotic approximation is generally tight for practical non-extremely limiting cases.

degree	rate=0.46	rate=0.47	rate=0.48
2	0.1951	0.2209	0.2307
3	0.4215	0.3683	0.3008
4	0	0	0
5	0	0	0
6	0	0	0.1422
7	0	0	0
8	0	0.0875	0
9	0	0	0
10	0.0410	0	0
11	0.0347	0	0
12	0	0	0
13	0	0	0
14	0	0	0
15	0	0	0
16	0.3076	0.3233	0.3263

Table 3.1: the left degree distributions for the design examples where $\xi = 0.5$.

degree	rate=0.46	rate=0.47	rate=0.48
2	0	0	0
3	0	0	0
4	0	0	0
5	0	0	0
6	0	0	0
7	1	0.8462	0.8632
8	0	0.1538	0.0509
9	0	0	0
10	0	0	0
11	0	0	0
12	0	0	0
13	0	0	0
14	0	0	0
15	0	0	0
16	0	0	0.0858

Table 3.2: the right degree distributions for the design examples where $\xi = 0.5$.

rate	T_η	$F(\lambda, \rho, \eta)$
rate=0.46	48	50.80
rate=0.47	67	69.63
rate=0.48	114	117.89

Table 3.3: the numbers of decoding iterations and the asymptotic approximations for the design examples where $\xi = 0.5$.

degree	rate=0.21	rate=0.22	rate=0.23
2	0.2762	0.1743	0.3344
3	0.4461	0.4579	0.3058
4	0	0	0
5	0	0.0601	0
6	0	0.1809	0
7	0	0	0.1252
8	0	0	0
9	0	0	0
10	0	0	0
11	0	0.1268	0
12	0	0	0
13	0.1645	0	0
14	0.1133	0	0
15	0	0	0
16	0	0	0.2346

Table 3.4: the left degree distributions for the design examples where $\xi = 0.75$.

degree	rate=0.21	rate=0.22	rate=0.23
2	0	0	0
3	0	0.2470	0
4	0.9585	0.5308	0.6462
5	0	0	0.3538
6	0	0	0
7	0	0	0
8	0	0	0
9	0	0	0
10	0	0	0
11	0	0	0
12	0.0219	0	0
13	0.0196	0	0
14	0	0	0
15	0	0	0
16	0	0.2222	0

Table 3.5: the right degree distributions for the design examples where $\xi = 0.75$.

rate	T_η	$F(\lambda, \rho, \eta)$
rate=0.21	66	69.48
rate=0.22	114	117.37
rate=0.23	147	151.63

Table 3.6: the numbers of decoding iterations and the asymptotic approximations for the design examples where $\xi = 0.75$.

degree	rate=0.71	rate=0.72	rate=0.73
2	0.0847	0.1211	0.2915
3	0.5058	0.4414	0.0100
4	0	0	0.3869
5	0	0	0
6	0	0	0
7	0	0	0
8	0	0	0
9	0	0	0
10	0	0	0
11	0	0	0
12	0	0	0
13	0	0	0
14	0	0	0
15	0	0	0
16	0.4094	0.4374	0.3116

Table 3.7: the left degree distributions for the design examples where $\xi = 0.25$.

degree	rate=0.71	rate=0.72	rate=0.73
2	0	0	0
3	0	0	0
4	0	0	0
5	0	0	0
6	0	0	0
7	0	0	0
8	0	0	0
9	0	0	0
10	0	0	0
11	0	0	0
12	0	0	0
13	0	0	0.2431
14	0.4068	0	0.5178
15	0.5932	0.7957	0.2391
16	0	0.2043	0

Table 3.8: the right degree distributions for the design examples where $\xi = 0.25$.

rate	T_η	$F(\lambda, \rho, \eta)$
rate=0.71	25	26.63
rate=0.72	34	35.30
rate=0.73	87	98.46

Table 3.9: the numbers of decoding iterations and the asymptotic approximations for the design examples where $\xi = 0.25$.

3.5 Right-Concentrated Degree Distribution

In this section, we discuss the conditions for the optimal degree distributions in the sense of convergence speeds. In the literature, majority of the previously reported degree distributions with satisfactory performance are right-concentrated. In our numerical experiments, we find that optimal degree distributions in the sense of convergence speeds tend to be right-concentrated. Therefore, we conjecture that right-concentrated degree distributions are optimal or near-optimal in the sense of decoding convergence speeds. In this section, we show that the conjecture is true for certain special cases.

3.5.1 Low-Erasure-Probability-Region Convergence Speed Analysis

In practical applications, the probability level η is generally small. The number of decoding iterations while the erasure probability is in a low erasure probability region may constitute a large fraction of the decoding convergence time. In other words, the decoding speed in the low erasure probability region dominates the overall decoding speed.

We show in this section that if the average right degree is fixed, then the right-concentrated degree distributions have optimal decoding speed in the low erasure probability region.

Note that the relation between $P_e^{(l)}$ and $P_e^{(l+1)}$ can be also written as follows:

$$y = \rho(1 - P_e^{(l)}) \tag{3.31}$$

$$y = 1 - \lambda^{-1}(P_e^{(l+1)}/\xi) \tag{3.32}$$

where, y is an auxiliary variable. The iterative process of $P_e^{(l)}$ is illustrated in Fig. 3.11. To increase the decoding speed, we need to move the curve $\rho(1-x)$ upward and the curve $1-\lambda^{-1}(x/\xi)$ to the left. Moving the curve $\rho(1-x)$ upward is equivalent to making the function $\rho(1-x)$ larger. Moving the curve $1-\lambda^{-1}(x/\xi)$ to the left is equivalent to making the function $\lambda(1-x)$ smaller. To have fast decoding speed in the low message erasure probability region, we need to make $\rho(x)$ large for x near 1 and $\lambda(x)$ small for x near 0.

We need the following auxiliary lemma for proving the main theorem in this section.

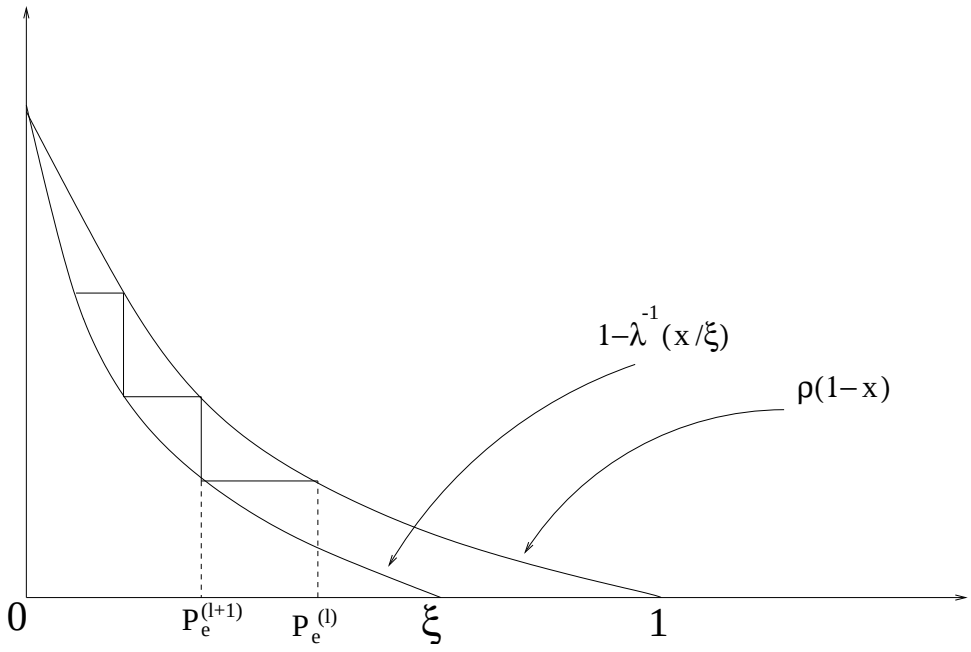


Figure 3.11: The iteration process of erasure probabilities

Lemma 3.5.1 *Let $\gamma(x)$ be a degree distribution with average degree a and maximal degree $d \geq 4$. Assume $\gamma_{i-1} > 0$ and $\gamma_{i+1} > 0$, for $2 < i < d$. Construct another degree distribution $\hat{\gamma}(x)$ with the same average and maximal degrees as follows:*

$$\hat{\gamma}(x) = \gamma(x) + \beta x^{i+1} - \left\{ \frac{(i-1)}{2i} \beta x^i + \frac{(i+1)}{2i} \beta x^{i+2} \right\} \tag{3.33}$$

where β is a sufficiently small positive real number such that $\hat{\gamma}(x)$ is well defined. Then, we have

$$\gamma(x) < \hat{\gamma}(x), \quad \textit{for} \ \ \frac{i-1}{i+1} < x < 1 \tag{3.34}$$

$$\gamma(x) > \hat{\gamma}(x), \quad \textit{for} \ \ 0 < x < \frac{i-1}{i+1} \tag{3.35}$$

Proof: The proof is in Appendix 3.C.

Theorem 3.5.2 *Let d be a positive integer. Let x be an arbitrary real number, $x > 1 - \frac{2}{d+1}$. Then the degree distribution with average degree a and maximal degree d which maximizes the function $\gamma(x)$ is*

$$\gamma(x) = \gamma_i x^{i-1} + \gamma_{i+1} x^i \tag{3.36}$$

where $i = \lfloor a \rfloor$ is the largest integer smaller than a.

Proof: The proof is in Appendix 3.D.

The above Theorem implies that the degree distributions with fast decoding speed at the low erasure probability region are right-concentrated.

3.5.2 Asymptotic Approximation Based Analysis

In this section, we discuss the condition for the optimal degree distributions in the approximate constrained optimization problem in Eqns. 3.18, 3.19, 3.20, 3.21, 3.22. We have the following theorem, which shows that the optimal degree distributions are right-concentrated in certain special cases.

Theorem 3.5.3 *Let (λ^*, ρ^*) be the solution of the constrained optimization problem in Eqns. 3.18, 3.19, 3.20, 3.21, 3.22. Further assume that $\xi \leq 1 - e^{-2/d_c}$, then ρ_j^* are non-zero only for two j's.*

Proof: The proof is in Appendix 3.E.

3.6 Conclusion

In this chapter, we present a framework for designing LDPC codes with fast decoding speeds. Both the theoretical discussions and numerical results show that density-efficient capacity-approaching codes satisfy the flatness condition. Asymptotically, the decoding convergence time T_η can be approximated by the function $F(\lambda, \rho, \eta)$. The asymptotic approximation is generally tight for practical scenarios. We conjecture that the optimal degree distributions in the sense of decoding speeds are right-concentrated. We prove that the conjecture is true for certain special cases.

3.A The Proof of Theorem 3.2.1

Proof: In the proof of the theorem, we need the following technical lemmas. The proofs of these lemmas can be found in Appendix 3.F to 3.M.

Proposition 3.A.1

$$\rho(1-x) \leq (-1)[\rho(1-x)]' \leq \frac{k_c a}{1-\xi}\rho(1-x), \quad \textit{for any } x \in (0,\xi] \tag{3.37}$$

Proposition 3.A.2

$$[\rho(1-x)]'' \leq (-1)\frac{k_c a}{1-\xi}[\rho(1-x)]', \quad \textit{for any } x \in (0,\xi]$$

Proof: Similar to that of Proposition 3.A.1. ∎

Lemma 3.A.3

$$\int_0^1 \rho(1-x)dx - \int_0^{\xi} \left[1-\lambda^{-1}(x/\xi)\right] dx = \frac{b\Delta R}{\xi + \Delta R} \tag{3.38}$$

where $\lambda^{-1}(\cdot)$ denotes the inverse function of $\lambda(x)$ throughout this chapter.

Lemma 3.A.4

$$\frac{1-\lambda^{-1}(x/\xi)}{\rho(1-x)} \geq 1 - \sqrt{k_c}B(\Delta R, b, x), \quad \textit{for any } x \in (0,\xi) \tag{3.39}$$

Lemma 3.A.5

$$|[\lambda(1-\rho(1-x))]''| \leq \frac{k_v^2 k_c^2 \rho(1-x)^2 a^4}{(1-\xi)^2} + \frac{k_v k_c^2 \rho(1-x) a^3}{(1-\xi)^2} \tag{3.40}$$

Lemma 3.A.6 *Let $b^5 + b^2 < \xi$ and $b^5 < x < \xi - b^2$. Then,*

$$\frac{[1-\lambda^{-1}(x/\xi)]'}{[\rho(1-x)]'} \geq 1 + \frac{-k_c b}{2(1-\xi)} - \sqrt{k_c} B(\Delta R, b, x) a^2 \tag{3.41}$$

$$\frac{[1-\lambda^{-1}(x/\xi)]'}{[\rho(1-x)]'} \leq 1 + \sqrt{k_c} B(\Delta R, b, x) a^5 + \frac{k_c b^4}{2(1-\xi)} \left(1 + \frac{b^5}{1-\xi}\right)^{k_c a} \tag{3.42}$$

Lemma 3.A.7 *Let $x_0 \in (0, \xi)$ and $x_1 = \xi\lambda(1-\rho(1-x_0))$. Then,*

$$[\xi\lambda(1-\rho(1-x_0))]' \leq \frac{[\rho(1-x_i)]'}{[1-\lambda^{-1}(x_i/\xi)]'} \tag{3.43}$$

for $i = 0, 1$.

Lemma 3.A.8 *For $x \in (0, \xi)$,*

$$x - \xi\lambda(1-\rho(1-x)) \leq \sqrt{\frac{-2b\Delta R}{(\xi + \Delta R)[\rho(1-x)]'}} \tag{3.44}$$

Lemma 3.A.9 *If $b < (1-\xi)/k_c$, then*

$$\rho(1 - \xi + b^2) \leq \frac{2k_c(1-\xi)\Delta R}{(\xi + \Delta R)(1 - \xi - k_c b^2)} \tag{3.45}$$

The rest of the proof is divided into three steps.

Step I: We will define a partition of the interval $(0, \xi]$.

For any n, we partition the interval $(0, \xi]$ into three subintervals $(0, \zeta_0]$, $(\zeta_0, \zeta_1]$, and $(\zeta_1, \xi]$, where

$$\zeta_0 = 2b_n^5 \tag{3.46}$$

$$\zeta_1 = 1 - \rho_n^{-1}\left((b_n)^5\right) \tag{3.47}$$

we claim that the partition is well-defined for sufficiently large n. That is, $\zeta_0 < \zeta_1$ for sufficiently large n. Note that

$$\rho_n(1-\zeta_0) = \rho_n\left(1 - 2(b_n)^5\right) = \sum_{j=2}^{k_c a_n} \rho_{nj}\left(1 - 2(b_n)^5\right)^{j-1} \tag{3.48}$$

Lower bounding $[1-2(b_n)^5]^{j-1}$ by $[1-2(b_n)^5]^{k_c a_n - 1}$, we have

$$\rho_n(1-\zeta_0) \geq \left[1-2(b_n)^5\right]^{k_c a_n - 1} \tag{3.49}$$

This lower bound of $\rho_n(1-\zeta_0)$ goes to 1, as n goes to infinity. Hence

$$\rho_n(1-\zeta_0) \to 1 \tag{3.50}$$

On the other hand,

$$\rho_n(1-\zeta_1) = (b_n)^5 \to 0 \tag{3.51}$$

Since $\rho_n(1-x)$ is a monotonously decreasing function, we conclude that $\zeta_0 < \zeta_1$ for sufficiently large n.

We claim that $\zeta_1 < \xi - (b_n)^2$ for sufficiently large n. According to Lemma 3.A.9,

$$\rho_n\left(1-\xi+(b_n)^2\right) = o\left((b_n)^{15}\right) \tag{3.52}$$

while by definition

$$\rho_n(1-\zeta_1) = (b_n)^5 \tag{3.53}$$

Hence $\zeta_1 < \xi - (b_n)^2$. The claim is proven.

Step II: In this step, we show that the derivative of the function $\xi\lambda_n(1-\rho_n(1-x))$ converges to 1 uniformly in the subinterval (ζ_0, ζ_1). That is, for any $\epsilon > 0$, there exists an N, such that for any $n \geq N$, $x \in (\zeta_0, \zeta_1)$,

$$\left|[\xi\lambda_n(1-\rho_n(1-x))]' - 1\right| \leq \epsilon. \tag{3.54}$$

We will show that the function $[\xi\lambda_n(1-\rho_n(1-x))]'$ is upper bounded and this upper bound goes to 1 uniformly as n goes to infinity. According to Lemma 3.A.7,

$$[\xi\lambda_n(1-\rho_n(1-x))]' \leq \frac{[\rho_n(1-x)]'}{[1-\lambda_n^{-1}(x/\xi)]'} \tag{3.55}$$

According to lemma 3.A.6,

$$\frac{[1-\lambda_n^{-1}(x/\xi)]'}{[\rho_n(1-x)]'} \geq 1 - \frac{k_c b_n}{2(1-\xi)} - \sqrt{k_c} B(\Delta R_n, b_n, x)(a_n)^2 \tag{3.56}$$

Also note that

$$\Delta R_n = o\left((b_n)^{15}\right) \tag{3.57}$$

$$B(\Delta R_n, b_n, x) = o\left((b_n)^5\right) \tag{3.58}$$

We conclude that

$$\frac{[1-\lambda_n^{-1}(x/\xi)]'}{[\rho_n(1-x)]'} \geq 1 + O(b_n) \tag{3.59}$$

Therefore, the function $[\xi\lambda_n(1-\rho_n(1-x))]'$ is upper bounded,

$$[\xi\lambda_n(1-\rho_n(1-x))]' \leq \frac{1}{1+O(b_n)} \tag{3.60}$$

This upper bound goes to 1 as n goes to infinity.

We claim that for $x \in (\zeta_0, \zeta_1]$,

$$x - \xi\lambda_n(1-\rho_n(1-x)) = o\left((b_n)^{5.5}\right) \tag{3.61}$$

Hence for $x \in (\zeta_0, \zeta_1]$,

$$\xi\lambda_n(1-\rho_n(1-x)) > (b_n)^5 \tag{3.62}$$

for sufficiently large n. Denote $\xi\lambda_n(1-\rho_n(1-x))$ by y. According to Lemma 3.A.8,

$$\Delta x \;=\; x - y \leq \sqrt{\frac{-2b_n\Delta R_n}{(\xi+\Delta R_n)[\rho(1-x)]'}} \tag{3.63}$$

Bounding $[\rho_n(1-x)]'$ as in Proposition 3.A.1, we have

$$\Delta x \;=\; x - y \leq \sqrt{\frac{2b_n\Delta R_n}{(\xi+\Delta R_n)[\rho(1-x)]}} \tag{3.64}$$

Note that

$$\rho_n(1-x) \geq \rho_n(1-\zeta_1) = (b_n)^5 \tag{3.65}$$

$$\Delta R_n = o\left((b_n)^{15}\right) \tag{3.66}$$

We have

$$\Delta x = o\left((b_n)^{5.5}\right) \tag{3.67}$$

Therefor for sufficiently large n,

$$\xi\lambda_n(1-\rho(1-x)) = x - \Delta x \geq \zeta_0 - \Delta x > (b_n)^5 \tag{3.68}$$

We will show that the function $[\xi\lambda_n(1-\rho_n(1-x))]'$ is also lower bounded and this lower bound converges to 1 as n goes to infinity. Note that

$$\begin{aligned}[\xi\lambda_n(1-\rho_n(1-x))]' =& \frac{[\rho_n(1-x)]'}{[1-\lambda_n^{-1}(y/\xi)]'} \\ =& \left\{\frac{[\rho_n(1-y)]'}{[1-\lambda_n^{-1}(y/\xi)]'}\right\}\left\{\frac{[\rho_n(1-x)]'}{[\rho_n(1-y)]'}\right\} \end{aligned} \tag{3.69}$$

where $y = \xi\lambda_n(1-\rho_n(1-x))$. For sufficiently large n, we bound the second term as follows:

$$\frac{[\rho_n(1-x)]'}{[\rho_n(1-y)]'} \geq \left(\frac{1-x}{1-y}\right)^{k_c a_n - 1} \geq \left(\frac{1-x}{1-x+b_n^{5.5}}\right)^{k_c a_n - 1} \tag{3.70}$$

We have the following lower bound for $[\xi\lambda_n(1-\rho_n(1-x))]'$

$$[\xi\lambda_n(1-\rho_n(1-x))] \geq \frac{[\rho_n(1-y)]'}{[1-\lambda_n^{-1}(y/\xi)]'}\left(\frac{1-x}{1-x+b_n^{5.5}}\right)^{k_c a_n - 1} \tag{3.71}$$

Since $y \geq (b_n)^5$, according to Lemma 3.A.6 we have,

$$\frac{[\rho_n(1-y)]'}{[1-\lambda_n^{-1}(y/\xi)]'} \to 1 \text{ as } n \to \infty \tag{3.72}$$

Also

$$\left(\frac{1-x}{1-x+b_n^{5.5}}\right)^{k_c a_n - 1} \to 1 \tag{3.73}$$

as $n \to \infty$. We conclude that this lower bound for $[\xi\lambda_n(1-\rho_n(1-x))]'$ converges to 1 as n goes to infinity.

From the above, we conclude that $[\xi\lambda_n(1-\rho_n(x))]'$ converges to 1 uniformly for $x \in (\zeta_0, \zeta_1]$ as n goes to infinity.

Step III: In this step, we show that the function $[\xi\lambda_n(1-\rho_n(1-x))]'$ also converges to 1 uniformly in the subintervals $(0, \zeta_0)$ and $(\zeta_1, \xi]$.

For $x \in (0, \zeta_0)$, according to Lemma 3.A.5,

$$|[\xi\lambda_n(1-\rho_n(1-x))]''| \leq \frac{k_v^2 k_c^2 \left[\rho_n(1-x)\right]^2 a_n^4}{(1-\xi)^2} + \frac{k_v k_c^2 \rho_n(1-x) a_n^3}{1-\xi} \leq O(a_n^4) \tag{3.74}$$

while the length of this interval is $2b_n^5$. Hence $[\xi\lambda_n(1-\rho_n(1-x))]'$ converges to 1 uniformly for $x \in (0, \zeta_0)$.

For $x \in (\zeta_1, \xi)$, according to Lemma 3.A.5,

$$|[\xi\lambda_n(1-\rho_n(1-x))]''| \leq \frac{k_v^2 k_c^2 \left[\rho_n(1-x)\right]^2 a_n^4}{(1-\xi)^2} + \frac{k_v k_c^2 \rho_n(1-x) a_n^3}{1-\xi} = O(b_n^2); \quad (3.75)$$

while the length of this interval is bounded by 1. Hence $[\xi\lambda_n(1-\rho_n(1-x))]'$ also converges to 1 uniformly in the interval (ζ_2, ξ). The theorem is proven. ∎

3.B The Proof of Theorem 3.2.2

Proof: According to Theorem 3.2.1, the derivative of $\xi\lambda_n(1-\rho_n(1-x))$ converges uniformly to 1 for $x \in (0, \xi]$ as n goes to infinity. Thus, there exists an ϵ_n such that

$$\left|[x - \xi\lambda_n(1-\rho_n(1-x))]'\right| \leq \epsilon_n, \quad (3.76)$$

for $x \in (0, \xi)$, and $\epsilon_n \to 0$ as $n \to \infty$. We define

$$y_{k+1} = \xi\lambda_n(1-\rho_n(1-y_k)) \quad (3.77)$$

for $k = 0, 1, 2, \ldots$, with $y_0 = \xi$. Then, the decoding convergence time

$$T_\eta = \max\{k : y_k \geq \eta\}. \quad (3.78)$$

We can upper bound the function $F(\lambda_n, \rho_n, \eta)$ as follows,

$$\begin{aligned} F(\lambda_n, \rho_n, \eta) &= \int_\eta^\xi \frac{dx}{x - \xi\lambda_n(1-\rho_n(1-x))} \\ &\overset{(a)}{\leq} \sum_{k=1}^{T_\eta+1} \int_{y_k}^{y_{k-1}} \frac{dx}{x - \xi\lambda_n(1-\rho_n(1-x))} \\ &\overset{(b)}{=} \sum_{k=1}^{T_\eta+1} \frac{y_{k-1} - y_k}{\zeta_k - \xi\lambda_n(1-\rho_n(1-\zeta_k))} \\ &\overset{(c)}{\leq} \sum_{k=1}^{T_\eta+1} \frac{y_{k-1} - y_k}{(1-\epsilon_n)(y_{k-1} - y_k)} = \frac{T_\eta + 1}{(1-\epsilon_n)} \end{aligned} \quad (3.79)$$

In the above inequalities, (a) follows from the fact that the integral on the right hand side is on a larger interval; (b) follows from the Mean-Value Theorem, where ζ_k is a real number between y_k and y_{k-1}. The inequality (c) holds, because

$\zeta_k - \xi\lambda_n(1-\rho_n(1-\zeta_k))$ can be bounded as follows by using the Mean-Value Theorem,

$$\begin{aligned}
\zeta_k - \xi\lambda_n(1-\rho_n(1-\zeta_k)) &= y_{k-1} - \xi\lambda_n(1-\rho_n(1-y_{k-1})) \\
&\quad + (\zeta_k - y_{k-1})\left.\frac{d}{dx}\right|_{x=\kappa_k}[x - \xi\lambda_n(1-\rho_n(1-x))] \\
&= y_{k-1} - y_k + (\zeta_k - y_{k-1})\left.\frac{d}{dx}\right|_{x=\kappa_k}[x - \xi\lambda_n(1-\rho_n(1-x))] \\
&\geq y_{k-1} - y_k - (\zeta_k - y_{k-1})\epsilon_n \\
&\geq (1-\epsilon_n)(y_{k-1} - y_k) \qquad (3.80)
\end{aligned}$$

where κ_k is a real number between ζ_k and y_{k-1}.

Similarly, we can also lower bound the function $F(\lambda_n, \rho_n, \eta)$ as follows,

$$\begin{aligned}
F(\lambda_n, \rho_n, \eta) &= \int_\eta^\xi \frac{dx}{x - \xi\lambda_n(1-\rho_n(1-x))} \\
&\geq \sum_{k=1}^{T_\eta} \int_{y_k}^{y_{k-1}} \frac{dx}{x - \xi\lambda_n(1-\rho_n(1-x))} \\
&= \sum_{k=1}^{T_\eta} \frac{y_{k-1} - y_k}{\zeta_k - \xi\lambda_n(1-\rho_n(1-\zeta_k))} \\
&\geq \sum_{k=1}^{T_\eta} \frac{y_{k-1} - y_k}{(1+\epsilon_n)(y_{k-1} - y_k)} = \frac{T_\eta}{1+\epsilon_n} \qquad (3.81)
\end{aligned}$$

Combining the two equations 3.79, and 3.81, we have

$$\frac{1}{1+\epsilon_n} \leq \frac{F(\lambda_n, \rho_n, \eta)}{T_\eta} \leq \frac{1}{1-\epsilon_n}\left(1 + \frac{1}{T_\eta}\right) \qquad (3.82)$$

It is easy to show that $T_\eta \to \infty$, as $n \to \infty$. The theorem follows. ∎

3.C The Proof of Lemma 3.5.1

Proof: We can check that $\hat{\gamma}(x)$ is a valid degree distribution, $\sum_i \hat{\gamma}_i = 1$ and the average degree is a, $\int_0^1 \hat{\gamma}(x) = 1/a$.

Note that $\frac{i-1}{i+1}$ and 1 are two roots of the polynomial

$$x - \frac{(i+1)}{2i}x^2 - \frac{(i-1)}{2i} = \frac{\hat{\gamma}(x) - \gamma(x)}{x^i} \tag{3.83}$$

Hence, for $0 < x < \frac{i-1}{i+1}$,

$$\hat{\gamma}(x) < \gamma(x), \tag{3.84}$$

for $\frac{i-1}{i+1} < x < 1$,

$$\hat{\gamma}(x) > \gamma(x). \tag{3.85}$$

∎

3.D The Proof of Theorem 3.5.2

Proof: We prove the theorem by contradiction. Assume that the degree distribution $\gamma(x)$ is nonzero for more than three indices or is nonzero for two non-consecutive indices. Then, either one of the following two cases happens.

Case 1: There exist three consecutive indices $i-1$, i, $i+1$ such that γ_{i-1}, γ_i, and γ_{i+1} are nonzero.

Note that

$$x > 1 - \frac{2}{d+2} > \frac{i-1}{i+1} \tag{3.86}$$

According to Lemma 3.5.1, we can constructed another degree distribution $\hat{\gamma}(x)$ such that $\hat{\gamma}(x) > \gamma(x)$. This contradict to the hypothesis that $\gamma(x)$ is optimal.

Case 2: There exist positive integers i and j such that γ_i, γ_j are nonzero, $i < i+1 < j$, and $\gamma_k = 0$ for any k, $i < k < j$.

The conditions of Lemma 3.5.1 is also satisfied in this case. Define $\gamma^k(x)$ for each k, $i < k < j$, as:

$$\gamma^k(x) = x^{k-1} - \frac{(k-1)}{2k}x^{k-2} - \frac{(k+1)}{2k}x^k \tag{3.87}$$

We can find real numbers $\alpha_{i+1}, \alpha_{i+2}, \cdots, \alpha_{j-1}$ such that $\alpha_k > 0$, for $i < k < j$, and the polynomial

$$\hat{\gamma}(x) = \gamma(x) + \sum_{k=i+1}^{j-1} \alpha_k \gamma^k(x) \tag{3.88}$$

is a valid degree distribution. For example, we can set $\alpha_k = k\beta$, for $i < k < j$, where β is an arbitrary real number. The degree distribution $\hat{\gamma}(x)$ has average right degree a. Since $\gamma^k(x) > 0$ for each k, we have

$$\hat{\gamma}(x) > \gamma(x) \tag{3.89}$$

This contradicts to the hypothesis that $\gamma(x)$ is optimal.

The theorem follows from the discussions in the two cases. ∎

3.E The Proof of Theorem 3.5.3

Proof: The optimal ρ_j^* must also be the solution to the following constrained optimization problem with λ_i being fixed and equal to λ_i^*.

$$\min \int_{\eta}^{\xi} \frac{dx}{x - \xi\lambda^*(1-\rho(1-x))} \tag{3.90}$$

subject to

$$\xi\lambda^*(1-\rho(1-x)) < x, \quad \text{for any } x \in (0,\xi] \tag{3.91}$$

$$\frac{\int_0^1 \rho(x)dx}{\int_0^1 \lambda^*(x)dx} = 1 - C + \Delta R \tag{3.92}$$

$$\sum_j \rho_j = 1, \;\; \rho_j \geq 0 \tag{3.93}$$

$$\rho_j = 0, \text{ for all } j > d_c \tag{3.94}$$

According to the Karush-Kuhn-Tucker condition [5], ρ_j^* satisfy the following equation, for all j with nonzero ρ_j^*,

$$\frac{\partial}{\partial\rho_j}\bigg|_{\rho=\rho^*} F(\lambda^*,\rho,\eta) + \alpha \frac{\partial}{\partial\rho_j}\bigg|_{\rho=\rho^*} \left[\frac{\int_0^1 \rho(x)dx}{\int_0^1 \lambda^*(x)dx}\right] + \beta\left[\frac{\partial}{\partial\rho_j}\bigg|_{\rho=\rho^*} \sum_j \rho_j\right] = 0, \tag{3.95}$$

where α, β are constants. This is equivalent to

$$j \left.\frac{\partial}{\partial \rho_j}\right|_{\rho=\rho^*} F(\lambda^*, \rho, \eta) + \frac{\alpha}{\int_0^1 \lambda^*(x)dx} + \beta j = 0 \tag{3.96}$$

After finding the derivative of $F(\lambda, \rho, \eta)$, we can rewrite the above equation as follows:

$$\int_\eta^\xi g(x) j (1-x)^{j-1} dx = \frac{\alpha}{\int_0^1 \lambda^*(x)dx} + \beta j \tag{3.97}$$

where

$$g(x) = \frac{\xi \sum_i \left\{ \lambda_i^*(i-1) \left[1 - \sum_j \rho_j^* (1-x)^{j-1} \right]^{i-2} \right\}}{\left\{ x - \xi \sum_i \left[\lambda_i^* \left(1 - \sum_j \rho_j^* (1-x)^{j-1} \right)^{i-1} \right] \right\}^2} \tag{3.98}$$

It can be checked that, in the interval $(0, 1 - e^{-2/d_c}]$, $j(1-x)^{j-1}$ is a concave function with respect to j. Hence $\int_\eta^\xi g(x) j (1-x)^{j-1} dx$ is also concave with respect to j. There exist at most two j's which satisfy Eqn. 3.97. The theorem follows. ∎

3.F The proof of Proposition 3.A.1

Proof: The lower bound of $(-1)[\rho(1-x)]'$ follows from the fact that

$$\begin{aligned}
(-1)[\rho(1-x)]' &= \sum_{j=2}^{k_c a} \rho_j (j-1)(1-x)^{j-2} \\
&\geq \sum_{j=2}^{k_c a} \rho_j (1-x)^{j-2} \\
&\geq \sum_{j=2}^{k_c a} \rho_j (1-x)^{j-1} = \rho(1-x)
\end{aligned} \tag{3.99}$$

Note that the maximal right degree is bounded by $k_c a$. The upper bound follows from

$$\begin{aligned}(-1)[\rho(1-x)]' &= \sum_{j=2}^{k_c a} \rho_j (j-1)(1-x)^{j-2} \\ &\leq (k_c a) \sum_{j=2}^{k_c a} \rho_j (1-x)^{j-2} \\ &\leq \frac{k_c a}{1-x} \sum_{j=2}^{k_c a} \rho_j (1-x)^{j-1} \\ &\leq \frac{k_c a}{1-\xi} \sum_{j=2}^{k_c a} \rho_j (1-x)^{j-1} \\ &= \frac{k_c a}{1-\xi} \rho(1-x) \end{aligned} \tag{3.100}$$

∎

3.G The proof of Lemma 3.A.3

Proof: Using the change of variables $x = 1 - v$, we have

$$\int_0^1 \rho(1-x)dx = \int_0^1 \rho(v)dv = b \tag{3.101}$$

Using the change of variables $x = \xi\lambda(u)$, we have

$$\int_0^{\xi} \lambda^{-1}(x/\xi)dx = \int_0^1 \xi u \lambda'(u)du = \xi - \frac{b\xi}{\xi + \Delta R} \tag{3.102}$$

The lemma follows. ∎

3.H The Proof of Lemma 3.A.4

Proof: Let us denote x by x_0 for convenience and define

$$c \triangleq 1 - \frac{1 - \lambda^{-1}(x/\xi)}{\rho(1-x)} \tag{3.103}$$

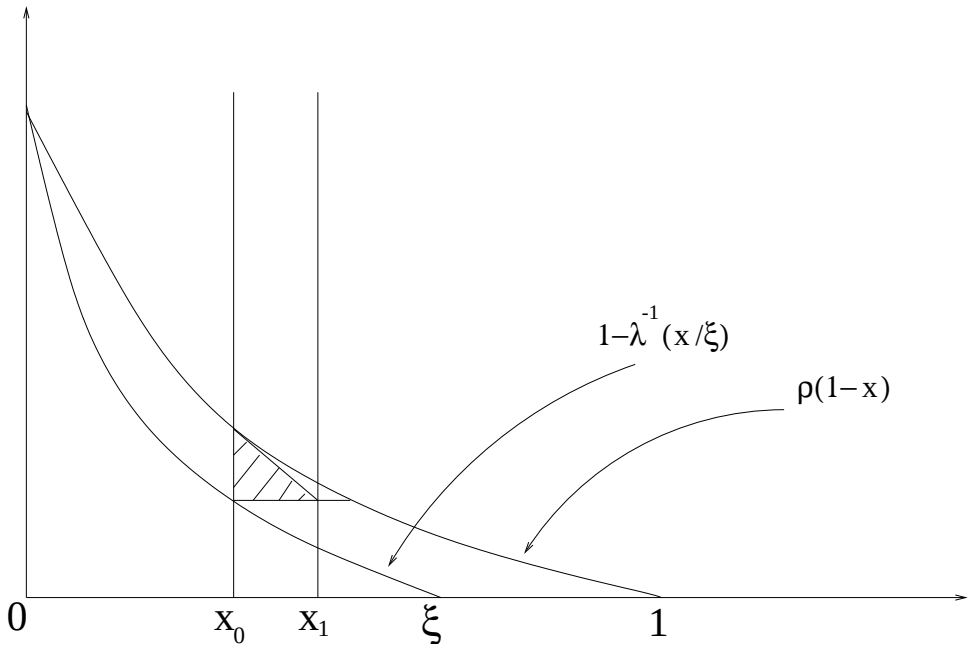


Figure 3.12: The geometrical interpretation of x_0 and x_1

$$x_1 \triangleq x_0 - \frac{c\rho(1-x_0)}{[\rho(1-x_0)]'} \tag{3.104}$$

The geometric meaning of x_1 is shown in Fig. 3.12. The number x_1 is the x coordinate of the intersection point of the horizontal line $y = 1 - \lambda^{-1}(x_0/\xi)$ and the straight line tangent to the curve $y = \rho(1-x)$ at the point $(x_0, \rho(1-x_0))$.

The shadowed region in Fig. 3.12 is smaller than the shadowed region in Fig. 3.13. The area of the shadowed region in Fig. 3.12 is

$$\frac{(-1)c^2\left[\rho(1-x_0)\right]^2}{2[\rho(1-x_0)]'} \tag{3.105}$$

The area of the shadowed region in Fig. 3.13 is

$$\int_0^1 \rho(1-x)dx - \int_0^\xi \left[1-\lambda^{-1}(x/\xi)\right] dx = \frac{b\Delta R}{\xi + \Delta R} \tag{3.106}$$

Hence

$$c^2 \leq \frac{-2b\Delta R[\rho(1-x_0)]'}{(\xi+\Delta R)\left[\rho(1-x_0)\right]^2} \tag{3.107}$$

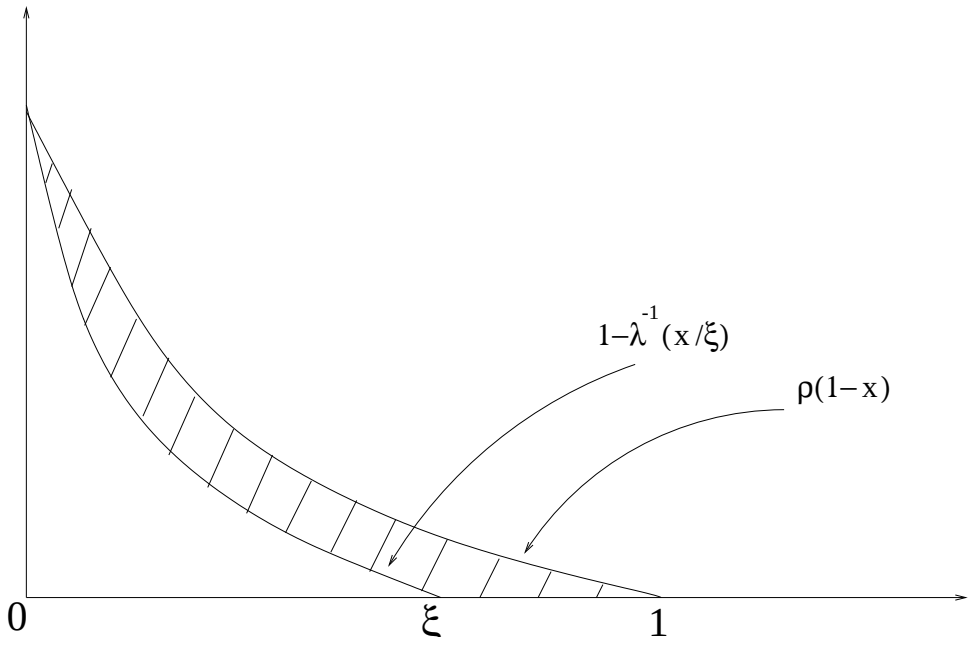


Figure 3.13: The geometrical interpretation

We further bound $(-1)[\rho(1-x_0)]'$ as in Proposition 3.A.1. This gives us the following bound

$$c \leq \sqrt{\frac{2k_c\Delta R}{(\xi+\Delta R)(1-\xi)\rho(1-x_0)}} \tag{3.108}$$

The lemma follows. ∎

3.I The Proof of Lemma 3.A.5

Proof: We can write $[\lambda(1-\rho(1-x))]''$ as follows:

$$[\lambda(1-\rho(1-x))]'' = \left[\sum_{i=3}^{k_v a}\lambda_i(i-1)(i-2)\left[1-\sum_j \rho_j(1-x)^{j-1}\right]^{i-3}\right]\left[\sum_j \rho_j(j-1)(1-x)^{j-2}\right]^2 - \left[\sum_i \lambda_i(i-1)\times\left[1-\sum_j \rho_j(1-x)^{j-1}\right]^{i-2}\right]\left[\sum_{j=3}^{k_c a}\rho_j(j-1)(j-2)(1-x)^{j-3}\right] \tag{3.109}$$

Hence, we have the following bound:

$$\begin{aligned}|[\lambda(1-\rho(1-x))]''| \leq & \left[\sum_{i=3}^{k_v a}\lambda_i(i-1)(i-2)\left[1-\sum_j \rho_j(1-x)^{j-1}\right]^{i-3}\right]\times \\ & \left[\sum_j \rho_j(j-1)(1-x)^{j-2}\right]^2 + \\ & \left[\sum_i \lambda_i(i-1)\times\left[1-\sum_j \rho_j(1-x)^{j-1}\right]^{i-2}\right]\times \\ & \left[\sum_{j=3}^{k_c a}\rho_j(j-1)(j-2)(1-x)^{j-3}\right]\end{aligned} \tag{3.110}$$

Applying the following bounding,

$$(i-1) < k_v a, \tag{3.111}$$

$$(i-2) < k_v a, \tag{3.112}$$

$$(j-1) < k_c a, \tag{3.113}$$

$$(j-2) < k_c a, \tag{3.114}$$

we have

$$
\begin{aligned}
&|[\lambda(1-\rho(1-x))]''| \leq k_v^2 k_c^2 a^4 \times \\
&\left[\sum_{i=3}^{k_v a} \lambda_i \left[1-\sum_j \rho_j (1-x)^{j-1}\right]^{i-3}\right] \left[\sum_j \rho_j (1-x)^{j-2}\right]^2 \\
&+k_v k_c^2 a^3 \left[\sum_i \lambda_i \left[1-\sum_j \rho_j (1-x)^{j-1}\right]^{i-2}\right] \times \\
&\left[\sum_{j=3}^{k_c a} \rho_j (1-x)^{j-3}\right] \qquad (3.115)
\end{aligned}
$$

Note that

$$\left[\sum_{i=3}^{k_v a} \lambda_i \left[1-\sum_j \rho_j (1-x)^{j-1}\right]^{i-3}\right] \leq 1 \qquad (3.116)$$

$$\left[\sum_i \lambda_i \left[1-\sum_j \rho_j (1-x)^{j-1}\right]^{i-2}\right] \leq 1 \qquad (3.117)$$

we have

$$
\begin{aligned}
&|[\lambda(1-\rho(1-x))]''| \leq k_v^2 k_c^2 a^4 \left[\sum_j \rho_j (1-x)^{j-2}\right]^2 + \\
&k_v k_c^2 a^3 \left[\sum_{j=3}^{k_c a} \rho_j (1-x)^{j-3}\right] \qquad (3.118)
\end{aligned}
$$

Further applying the following upper bounds for $\sum_j \rho_j (1-x)^{j-2}$ and $\sum_{j=3}^{k_c a} \rho_j (1-x)^{j-3}$,

$$\sum_j \rho_j (1-x)^{j-2} = \frac{\sum_j \rho_j (1-x)^{j-1}}{1-x} \leq \frac{\rho(1-x)}{1-\xi}, \qquad (3.119)$$

$$\sum_{j=3}^{k_c a} \rho_j (1-x)^{j-3} \leq \frac{\sum_{j=2}^{k_c a} \rho_j (1-x)^{j-1}}{(1-x)^2} \leq \frac{\rho(1-x)}{(1-\xi)^2}, \qquad (3.120)$$

we have,

$$|[\lambda(1-\rho(1-x))]''| \leq \frac{k_v^2 k_c^2 \rho(1-x)^2 a^4}{(1-\xi)^2} + \frac{k_v k_c^2 \rho(1-x) a^3}{(1-\xi)^2}. \tag{3.121}$$

∎

3.J The Proof of Lemma 3.A.6

Proof: Let $x_0 \in (b^5, \xi - b^2)$. For all $x \in (0, \xi)$, $x \neq x_0$, we have the following Taylor series expansion,

$$\begin{aligned} &1-\lambda^{-1}(x/\xi)-\rho(1-x) = 1-\lambda^{-1}(x_0/\xi)-\rho(1-x_0)+ \\ &\left\{\left[1-\lambda^{-1}(x_0/\xi)\right]' - [\rho(1-x_0)]'\right\}(x-x_0)+ \\ &\left\{\left[1-\lambda^{-1}(\zeta/\xi)\right]'' - [\rho(1-\zeta)]''\right\}\frac{(x-x_0)^2}{2}, \end{aligned} \tag{3.122}$$

where ζ is a real number between x_0 and x. According to the hypotheses, $\xi\lambda(1-\rho(1-x)) < x$. This implies $1-\lambda^{-1}(x/\xi)-\rho(1-x) < 0$, and

$$\begin{aligned} &1-\lambda^{-1}(x_0/\xi)-\rho(1-x_0)+ \\ &\left\{\left[1-\lambda^{-1}(x_0/\xi)\right]' - [\rho(1-x_0)]'\right\}(x-x_0)+ \\ &\left\{\left[1-\lambda^{-1}(\zeta/\xi)\right]'' - [\rho(1-\zeta)]''\right\}\frac{(x-x_0)^2}{2} < 0. \end{aligned} \tag{3.123}$$

Note that $[1-\lambda^{-1}(\zeta/\xi)]'' > 0$. Therefore, we have the following more convenient inequality:

$$\begin{aligned} &1-\lambda^{-1}(x_0/\xi)-\rho(1-x_0)+ \\ &\left\{\left[1-\lambda^{-1}(x_0/\xi)\right]' - [\rho(1-x_0)]'\right\}(x-x_0) \\ &-[\rho(1-\zeta)]''\frac{(x-x_0)^2}{2} < 0. \end{aligned} \tag{3.124}$$

Setting $x = x_0 + b^2$ in the above inequality and applying the fact that $[\rho(1-x_0)]' < 0$, with a little algebra, we have,

$$\left\{\frac{[1-\lambda^{-1}(x_0/\xi)]'}{[\rho(1-x_0)]'} - 1\right\} \geq \frac{b^2[\rho(1-\zeta)]''}{2[\rho(1-x_0]'} + \frac{[\lambda^{-1}(x_0/\xi)+\rho(1-x_0)-1]}{b^2[\rho(1-x_0)]'}. \tag{3.125}$$

To prove the lower bound in the lemma, we will bound the two terms in the right hand side of Eqn. 3.125 separately.

We bound the first term as follows. Because $[\rho(1-x)]''$ is a monotonous decreasing function,

$$[\rho(1-\zeta)]'' \leq [\rho(1-x_0)]'', \tag{3.126}$$

which, combined with Proposition 3.A.2, implies

$$[\rho(1-\zeta)]'' \leq \frac{(-1)k_c a}{(1-\xi)}[\rho(1-x_0)]'. \tag{3.127}$$

Thus, the first term in the right hand side of Eqn. 3.125 can be lower bounded as,

$$\frac{b^2[\rho(1-\zeta)]''}{2[\rho(1-x_0]'} \geq \frac{-k_c b}{2(1-\xi)}. \tag{3.128}$$

We bound the second term in the right hand side of Eqn. 3.125 as follows. The second term in the right hand side of Eqn. 3.125 can be rewritten as,

$$\begin{aligned}&\frac{\rho(1-x_0)-1+\lambda^{-1}(x_0/\xi)}{b^2[\rho(1-x_0)]'} = \\ &(-a^2)\left\{\frac{\rho(1-x_0)-1+\lambda^{-1}(x_0/\xi)}{\rho(1-x_0)}\right\}\left\{\frac{\rho(1-x_0)}{(-1)[\rho(1-x_0)]'}\right\}.\end{aligned} \tag{3.129}$$

According to Lemma 3.A.4,

$$\frac{\rho(1-x_0)-1+\lambda^{-1}(x_0/\xi)}{\rho(1-x_0)} \leq \sqrt{k_c}B(\Delta R, b, x_0). \tag{3.130}$$

According to Proposition 3.A.1

$$\left\{\frac{\rho(1-x_0)}{-[\rho(1-x_0)]'}\right\} \leq 1. \tag{3.131}$$

Hence, the second term in the right hand side of Eqn. 3.125 can be lower bounded as,

$$\frac{\rho(1-x_0)-1+\lambda^{-1}(x_0/\xi)}{[\rho(1-x_0)]'b^2} \geq -\sqrt{k_c}B(\Delta R, b, x_0)a^2. \tag{3.132}$$

Substituting Eqns. 3.132 and 3.128 into Eqn. 3.125 gives the lower bound in the lemma.

In the following, we will prove the upper bound in the lemma. Setting $x = x_0 - b^5$ in Eqn. 3.124, with a little algebra, we have

$$\frac{[1-\lambda^{-1}(x_0/\xi)]'}{[\rho(1-x_0)]'} - 1 \leq \frac{1-\lambda^{-1}(x_0/\xi)-\rho(1-x_0)}{b^5[\rho(1-x_0)]'} + \frac{-b^5[\rho(1-\zeta)]''}{2[\rho(1-x_0)]'}. \tag{3.133}$$

Due to Lemma 3.A.4 and Proposition 3.A.1, the first term at the right hand side of Eqn. 3.133 can be bounded as,

$$\begin{aligned}\frac{1-\lambda^{-1}(x_0/\xi)-\rho(1-x_0)}{[\rho(1-x_0)]'b^5} \leq & \frac{a^5\sqrt{k_c}B(\Delta R,b,x)\rho(1-x_0)}{(-1)[\rho(1-x_0)]'} \\ \leq & \sqrt{k_c}B(\Delta R,b,x)a^5. \end{aligned} \tag{3.134}$$

Note that the maximal right degree is bounded by $k_c a$. Hence

$$\begin{aligned}[\rho(1-\zeta)]'' \leq & [\rho(1-x_0)]'' \left[\frac{1-\zeta}{1-x_0}\right]^{k_c a-2} \\ \leq & [\rho(1-x_0)]'' \left[\frac{1-x_0+b^5}{1-x_0}\right]^{k_c a-2} \\ \leq & [\rho(1-x_0)]'' \left[1+\frac{b^5}{1-\xi}\right]^{k_c a}. \end{aligned} \tag{3.135}$$

By bounding $[\rho(1-\zeta)]''$ as in Proposition 3.A.2, the second term at the right hand side of Eqn. 3.133 can be bounded by

$$(-1)\frac{[\rho(1-\zeta)]''b^5}{2[\rho(1-x_0)]'} \leq \frac{k_c b^4}{2(1-\xi)}\left(1+\frac{b^5}{1-\xi}\right)^{k_c a}. \tag{3.136}$$

Substituting Eqns. 3.134 and 3.136 into Eqn. 3.133 gives the upper bound in the lemma. ∎

3.K The Proof of Lemma 3.A.7

Proof: Notice that

$$[\xi\lambda(1-\rho(1-x_0))]' = \frac{[\rho(1-x_0)]'}{[1-\lambda^{-1}(x_1/\xi)]'}. \tag{3.137}$$

Because $[\rho(1-x)]'$ and $[1-\lambda^{-1}(x/\xi)]'$ are monotonously increasing,

$$\frac{[\rho(1-x_1)]'}{[1-\lambda^{-1}(x_1/\xi)]'} \geq \frac{[\rho(1-x_0)]'}{[1-\lambda^{-1}(x_1/\xi)]'}, \tag{3.138}$$

$$\frac{[\rho(1-x_0)]'}{[1-\lambda^{-1}(x_0/\xi)]'} \geq \frac{[\rho(1-x_0)]'}{[1-\lambda^{-1}(x_1/\xi)]'}, \tag{3.139}$$

The Lemma follows. ∎

3.L The Proof of Lemma 3.A.8

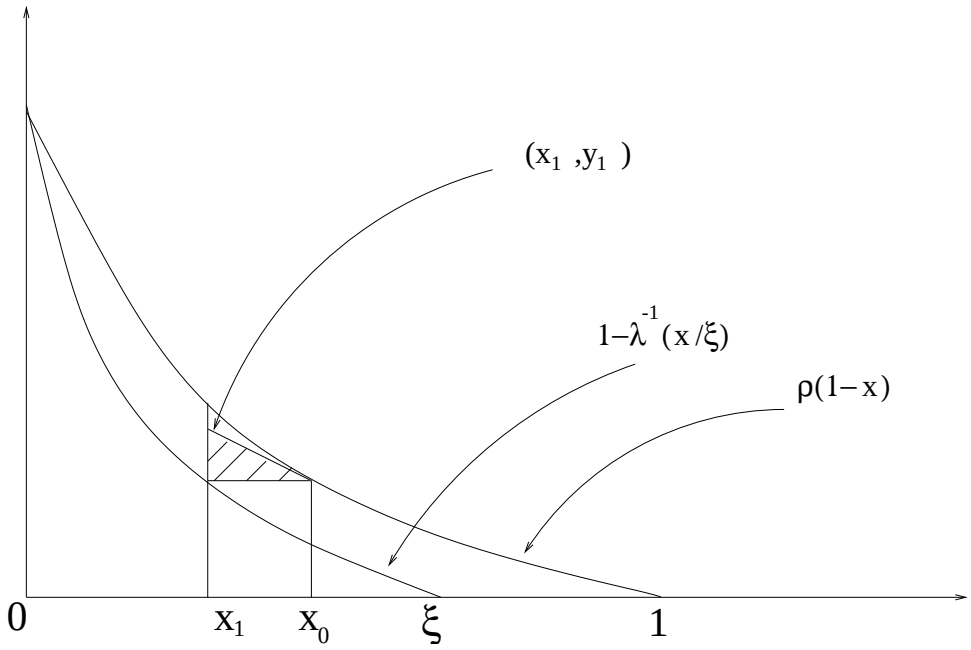


Figure 3.14: The geometrical interpretation of x_0 and x_1.

Proof: Denote x by x_0 for convenience. Define

$$x_1 = \xi\lambda(1-\rho(1-x_0)), \tag{3.140}$$

$$y_0 = \rho(1 - x_0), \tag{3.141}$$

$$y_1 = y_0 - (x_0 - x_1)[\rho(1 - x_0)]'. \tag{3.142}$$

The geometric meaning of x_0, x_1, y_0, and y_1 is shown in Fig. 3.14. The point (x_1, y_1) is the intersection of the vertical straight line $x = x_1$ and the straight line tangent to the curve $\rho(1 - x)$ at the point (x_0, y_0).

The area of the shadowed region in Fig. 3.14 is

$$\frac{-1}{2}[\rho(1 - x_0)]'(x_0 - x_1)^2. \tag{3.143}$$

The shadowed region in Fig. 3.14 is smaller than the shadowed region in Fig. 3.13,

$$\frac{-1}{2}[\rho(1 - x_0)]'(x_0 - x_1)^2 \leq \frac{b\Delta R}{\xi + \Delta R}. \tag{3.144}$$

The lemma follows. ∎

3.M The Proof of Lemma 3.A.9

Proof: Define

$$x_0 = \xi - b^2, \tag{3.145}$$

$$y_0 = \rho(1 - x_0), \tag{3.146}$$

$$x_1 = x_0 + \frac{y_0}{(-1)[\rho(1 - x_0)]'}. \tag{3.147}$$

Because $b < (1 - \xi)/k_c$,

$$\begin{aligned} x_1 =& x_0 + \frac{y_0}{(-1)[\rho(1 - x_0)]'} \geq x_0 + \frac{(1 - \xi)y_0 b}{k_c \rho(1 - x_0)} \\ \geq& x_0 + \frac{(1 - \xi)b}{k_c} > \xi. \end{aligned} \tag{3.148}$$

The geometric meaning of x_1 is shown in Fig. 3.15. The point $(x_1, 0)$ is the intersection of the x-axis and the straight line tangent to the curve $y = \rho(1 - x)$ at the point (x_0, y_0).

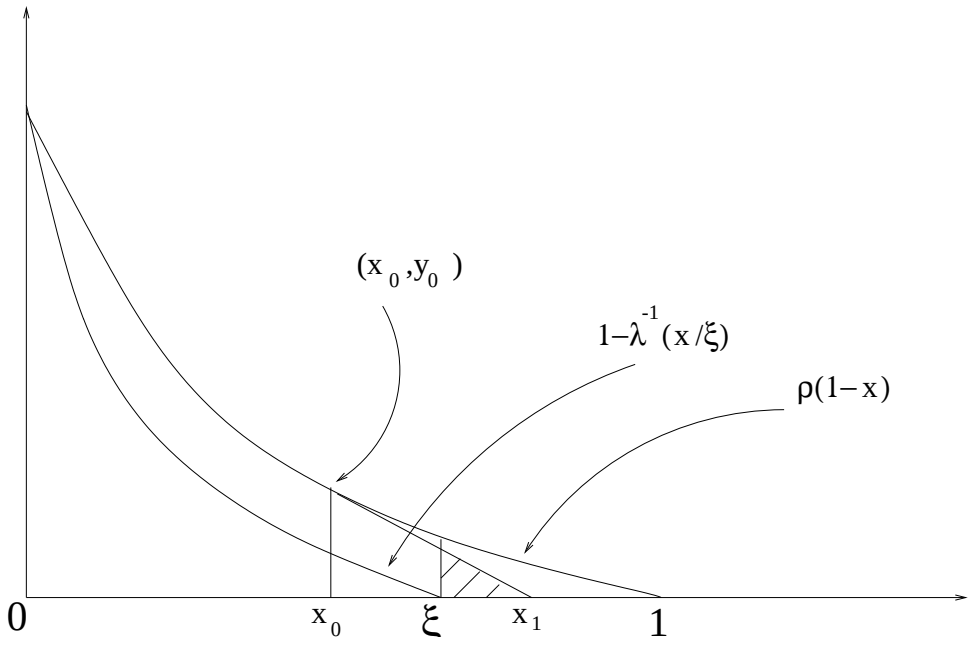


Figure 3.15: The geometrical interpretation of x_1

The shadowed region in Fig. 3.15 is smaller than the shadowed region in Fig. 3.13. For the area of the shadowed triangle region in Fig. 3.15, the width is $x_1 - \xi$, the height is $(x_1 - \xi)y_0/x_1 - x_0$, and the area is

$$\frac{y_0(x_1 - \xi)^2}{2(x_1 - x_0)} \tag{3.149}$$

Because this area is monotonously increase with respect to x_1, applying the bound for x_1 in Eqn. 3.148, we have the following lower bound of the area

$$\frac{y_0 b}{2k_c(1-\xi)} \left(1 - \xi - k_c b\right)^2 . \tag{3.150}$$

This lower bound is less than the area of the shadowed region in Fig. 3.13

$$\frac{y_0 b}{2k_c(1-\xi)} \left(1 - \xi - k_c b\right)^2 \leq \frac{b\Delta R}{\xi + \Delta R}. \tag{3.151}$$

The lemma follows. ∎

Chapter 4

Constructing LDPC Codes by 2-Lifts

In this chapter, we propose a new code construction scheme based on random 2-lifts. The motivation is to construct Low-Density Parity-Check codes with low error floors.

All previously constructed LDPC codes exhibit error floor phenomenon under message-passing decoding. That is, the bit error probabilities decrease very slowly as the signal-to-noise ratio increase. The error floors are caused by small subgraphs with certain structures in the corresponding Tanner graphs [33], [27], [10], [12]. In the special case of BEC, the error floors are caused by small stopping sets.

In order to understand and predict the error probability performance of LDPC codes, previous research has presented several analysis on stopping set distributions and codeword distributions [4], [29], [20], [21], [13], [11], [9]. It has been shown that capacity-approaching codes over BECs by the standard construction have low-weight codewords with high probabilities [11], [9]. The contributions of low-weight codewords to the error probabilities have also been calculated [29], [11]. For a BEC with channel parameter ϵ, it has been shown that [11, Corollary 2],

$$\mathbb{E}[P_B] \geq C_0, \qquad \mathbb{E}[P_b] \geq \frac{C_1}{n}, \tag{4.1}$$

for sufficiently large n, where

- P_B denotes the block error probability,
- P_b denotes the bit error probability,

- $\mathbb{E}[\cdot]$ denotes the expectation with respect to the randomness in the code construction,
- n denotes the blocklength, and
- C_0, C_1 are constants, C_0, C_1 only depend on $\lambda(x)$, $\rho(x)$ and ϵ.

The right hand sides of Eqn. (4.1) are the contributions of codewords with weights less than $\nu^{\mathrm{BP}} n$, where ν^{BP} only depends on ϵ, $\lambda(x), \rho(x)$ (note that codewords are a special class of stopping sets, more precisely, supports of codewords are stopping sets). The analysis is consistent with the previous numerical observations that capacity-approaching codes by the standard construction have high error floors.

Based on the above analysis, it has been conjectured that capacity-approaching and achieving low error probabilities in the error floor regions are conflicting design objectives [11], [9]. However, to the best of our knowledge, the conjecture is neither proved nor disproved up-to-date. Nevertheless, finding code ensembles with both capacity-approaching performance and low error floors has both theoretical and practical significance.

In this chapter, we propose a new LDPC code construction scheme based on a series of random 2-lifts. The resulting code ensembles are expurgated code ensembles of the standard LDPC code ensembles. We present an analysis of stopping set distributions of the resulting codes in terms of graph expansion properties. Our result states that low-weight stopping sets in the resulting codes are typically the results of stopping sets with weak graph expansion properties in the base graphs. Based on this analysis, we propose design criteria for the proposed 2-lift based codes, such that small stopping sets are largely avoided, and as a consequence, the resulting codes have low error floors. According to the design criteria, small stopping sets with weak graph expansion properties should be avoided in the base graphs. More detailed explanations will be provided in Section 4.3.

We present numerical results on proposed capacity-approaching LDPC codes over BECs. The numerical results show that the resulting codes by the proposed construction have significantly lower error floors compared with the codes by the standard construction.

There are also other related previous works on constructing LDPC codes by lifts and expansion properties of lift graphs. A code construction scheme was proposed in [39], where the resulting Tanner graphs are the N-lifts of small sized *protographs*. The major difference between this work and the previous works on lift based constructions lies in the different design methodologies. More precisely, the

main contribution of this chapter is the design criteria. For general discussions on graph expansions, 2-lifts etc., we refer interested readers to [14], [2] and references therein.

The error floor problem has attracted much attention previously. The previous research on this problem mainly focused on girth-conditioning algorithms and algebraic LDPC code constructions. The girth-conditioning approach constructs the LDPC codes in a way so that the Tanner graphs do not contain short cycles, or equivalently, the codes have high girths, where the girth of a graph is defined to be the length of the shortest cycle in the graph. This approach is based on a heuristic argument that the message-passing algorithms are exact on acyclic graphs. Its effects on alleviating the error floor problem is empirically observed. However, the improvement is marginal. The algebraic LDPC code construction approach brings significant improvement in terms of error floor. However, this class of codes usually can not come close to Shannon capacity limits. We refer interested readers to [28] and references therein for the related discussions.

The rest of this chapter is organized as follows. In Section 4.1, we introduce notation and backgrounds. The proposed code construction scheme is shown in Section 4.2. In Section 4.3, we present the analysis on stopping set distributions. Based on the analysis, a set of design criteria is presented. The proof of the main Theorems is shown in Sections 4.4, and 4.5. We present numerical results in Section 4.6. Conclusions are given in Section 4.7.

4.1 Notation and Preliminary

We will use the following widely adopted graph-theoretical notation throughout this chapter. Given a graph G, we use the notation $G = (V, E)$ to denote that V is the set of all vertices and E is the set of all edges. We use (u, v) to denote the *undirected* edge between two vertices u and v. For a set of vertices $\mathcal{S}$ in a graph $G = (V, E)$, we denote the neighborhood of $\mathcal{S}$ by $N(\mathcal{S})$, $N(\mathcal{S}) = \{u : (u, v) \in E, v \in \mathcal{S}\}$. We use $|\mathcal{S}|$ to denote the cardinality of a set $\mathcal{S}$. We say that a graph $G' = (V', E')$ is a subgraph of $G = (V, E)$ induced by V', if $V' \subset V$ and E' is the set of edges in E joining two vertices in V'.

We will need the following notation on graph expansion properties in our later discussions. Let G denote a Tanner graph.

Definition: We say that a variable node set $\mathcal{S}$ in G has β vertex expansion property, if $|N(\mathcal{S})| \geq \beta|\mathcal{S}|$.

Definition: Let $\mathcal{S}$ denote a variable node set in G. Let $\widetilde{N}(B)$ denote the set of check nodes connected to nodes in a variable node set B at least twice. We say that $\mathcal{S}$ has (η, γ) uniform-edge-distribution property if for each subset $B \subset \mathcal{S}$ with $|B| \leq \eta|\mathcal{S}|$, we have

$$|\widetilde{N}(B)| \leq \gamma|B|. \tag{4.2}$$

Definition: Given a graph $G = (V, E)$, a graph $\widehat{G} = (\widehat{V}, \widehat{E})$ is said to be an N-lift of G if the following conditions hold.

- The vertex set $\widehat{V}$ is the union of N disjoint vertex sets $V_1, \ldots, V_N$, where $V_1, \ldots, V_N$ have the same cardinality as V. That is, $\widehat{V} = V_1 \cup \cdots \cup V_N$, and $|V| = |V_1| = \cdots = |V_N|$.
- There exists a covering map $\pi : \widehat{V} \to V$. That is, π is a surjective map such that the cardinality of $\pi^{-1}(v)$ is independent of v.
- For any undirected edge (u, v) in the graph G, if a vertex $u_1 \in \pi^{-1}(u) \subset \widehat{V}$, then u_1 is connected to exactly one vertex in $\pi^{-1}(v)$.

We call the graph G the base graph of $\widehat{G}$. In the following, we say that a graph $\widehat{G}$ is a uniform random 2-lift of a graph G, if $\widehat{G}$ is randomly chosen as one of the 2-lifts of G with equal probability.

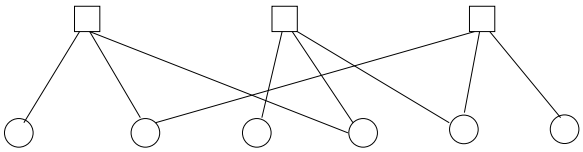
An N-lift can also be visualized as an operation of *copying-N-times and edge-permutating*. By "edge-permutating", we mean the operation of deleting edges (u_1, v_1), (u_2, v_2), and adding edges (u_1, v_2), (u_2, v_1), where u_1, u_2 are the two copies of the same vertex u, and v_1, v_2 are the two copies of the same vertex v. A graph $\widehat{G}$ is an N-lift of a graph G, if $\widehat{G}$ can be obtained from G by copying-N-times and edge-permutating any number of times.

Some examples of 2-lift graphs are shown in Fig. 4.1. The 2-lifts in the subgraphs 4.1b, 4.1c are both the 2-lifts of the base graph in Fig. 4.1a. The 2-lift graph in Fig. 4.1b is obtained by copying the base graph two times without any edge permutation. The 2-lift graph in Fig. 4.1c is obtained by copying the base graph two times and permuting one pair of edges.

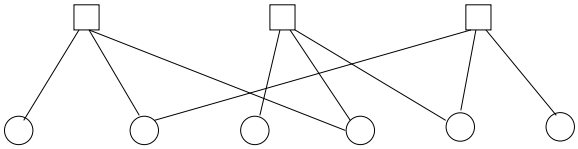
We use $H_e(x)$ to denote the binary entropy function with base e,

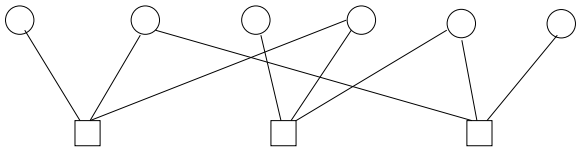
$$H_e(x) = -x \log_e(x) - (1 - x) \log_e(1 - x). \tag{4.3}$$

Similarly, $H_2(x)$ denotes the binary entropy function with base 2. We denote the set difference of two sets A, and B as $A \backslash B$, $A \backslash B = \{u | u \in A, u \notin B\}$. Throughout this chapter, we assume all graphs are simple graphs; that is, the graphs do not contain loop and parallel edges.

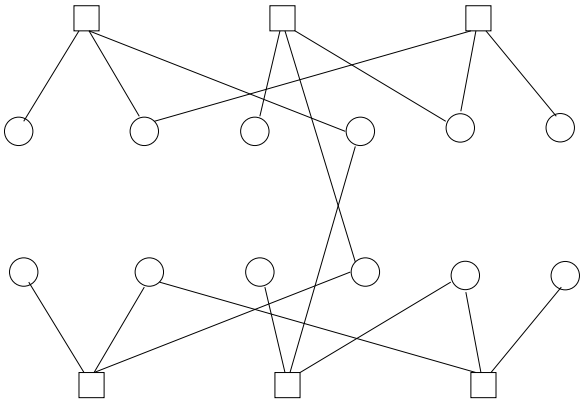


(a) the base graph





(b) a 2-lift without edge permutation



(c) a 2-lift with one edge permutation

Figure 4.1: Examples of N-lift

4.2 Code Construction Scheme

We propose the following LDPC code construction scheme. A small sized Tanner graph is first constructed by computational searching or algebraic construction. We denote this Tanner graph by G_0 and call it the *protograph*. The Tanner graph of the resulting code is obtained by recursively applying the uniformly random 2-lift procedure several times. That is, we construct a sequence of Tanner graphs $G_1, \ldots, G_k, \ldots, G_M$, such that G_k is a uniform random 2-lift of G_{k-1}, for $k = 1, \ldots, M$. The graph G_M is the Tanner graph of the resulting code.

4.3 Stopping Set Distribution Analysis

In this section, we will present an analysis on stopping set distributions. We will need the following notation.

- As with Section 4.2, we denote the sequence of Tanner graphs obtained in the code construction scheme by $G_0, G_1, \ldots, G_M$.
- Denote the covering map from G_k to G_{k-1} by $\pi_k(\cdot)$.
- Denote the number of variable nodes in the protograph by n_0.
- Denote the minimal weight of stopping sets in the protograph by d_0. That is, the cardinalities of all stopping sets in G_0 are greater than or equal to d_0, and there exists one stopping set with cardinality d_0.
- Denote the collection of stopping sets in G_k with cardinality w by $\mathcal{A}(G_k, w)$.
- For a stopping set $\mathcal{S}$ in the graph G_k, $k \in \{0, \ldots, M-1\}$, let $\mathcal{B}(\mathcal{S}, \theta)$ denote the collection of stopping sets $\widehat{\mathcal{S}}$ in G_{k+1} with $|\widehat{\mathcal{S}}| = (1+\theta)|\mathcal{S}|$, and $\pi_{k+1}(\widehat{\mathcal{S}}) = \mathcal{S}$, where $\pi_{k+1}(\widehat{S})$ is equal to $\{\pi_{k+1}(u) : u \in \widehat{S}\}$ by definition.

Proposition 4.3.1 *If $\widehat{\mathcal{S}}$ is a stopping set in a 2-lift G_k, then the set $\mathcal{S} = \pi(\widehat{\mathcal{S}})$ is a stopping set in the base graph G_{k-1}.*

Corollary 4.3.2 *The minimal weight of stopping sets in G_M is greater than or equal to d_0.*

Proposition 4.3.3 *Let $\mathcal{S}$ be a variable node set in G_{k-1}, and $\widehat{\mathcal{S}}$ be the variable node set $\widehat{\mathcal{S}} = \{u : \pi_k(u) \in \mathcal{S}\}$ in G_k. If $\mathcal{S}$ is a stopping set, then $\widehat{\mathcal{S}}$ is a stopping set in G_k.*

By Proposition 4.3.1 and the linearity of expectation, we have the following recursive formula for the expected stopping set distribution:

$$\mathbb{E}\left[|\mathcal{A}(G_{k+1}, w)||G_k\right] = \sum_{\substack{\mathcal{S}', w', \\ \mathcal{S}' \in \mathcal{A}(G_k, w'), \\ w/2 \leq w' \leq w}} \mathbb{E}\left[|\mathcal{B}(\mathcal{S}', w/w' - 1)||G_k\right]. \tag{4.4}$$

As discussed in the previous sections, in order for the resulting codes to have low error floors, $\mathbb{E}\left[|\mathcal{A}(G_{k+1}, w)|| G_k\right]$ should be minimized for small cardinalities w. However, due to difficulties in exactly calculating the left hand side of Eqn. (4.4), we will investigate $\mathbb{E}\left[|\mathcal{B}(\mathcal{S}', \theta||G_k\right]$ for different types of stopping sets $\mathcal{S}'$ instead. It is clear that a stopping set $\mathcal{S}'$ is problematic if $\mathbb{E}\left[|\mathcal{B}(\mathcal{S}', \theta||G_k\right]$ takes large values for $\theta \approx 0$, because $\mathcal{S}'$ introduces small stopping sets in the lift graph with large probabilities.

In order to gain insights into the properties of the function $\mathbb{E}\left[|\mathcal{B}(\mathcal{S}', \theta||G_k\right]$, it is worthwhile to consider the following example. Let us consider a stopping set $\mathcal{S}$, such that all variable nodes in $\mathcal{S}$ have degree two, and the variable nodes in $\mathcal{S}$ and their neighboring check nodes form a single cycle. It can be checked that for any stopping set $\widehat{\mathcal{S}}$ in G_{k+1} with $\pi_{k+1}(\widehat{\mathcal{S}}) = \mathcal{S}$, the variable nodes in $\widehat{\mathcal{S}}$ and their neighboring check nodes form either a single cycle or two cycles with the same length. Hence,

$$\mathbb{E}\left[|\mathcal{B}(\mathcal{S}, \theta||G_k\right] = \begin{cases} 1, & \theta = 1, \\ 1, & \theta = 0, \\ 0, & \text{otherwise.} \end{cases} \tag{4.5}$$

The important insight we obtain from the example is that the function $\mathbb{E}\left[|\mathcal{B}(\mathcal{S}, \theta||G_k\right]$ is generally not continuous (or can be approximated by a continuous function) with respect to θ. Therefore, it is unlikely that there exists a general analytic formula for $\mathbb{E}\left[|\mathcal{B}(\mathcal{S}, \theta||G_k\right]$. However, we will show an insightful bound for $\mathbb{E}\left[|\mathcal{B}(\mathcal{S}, \theta||G_k\right]$, which is tight for certain special cases.

Subsequently, we say that a variable node set $\mathcal{S}$ has strong graph expansion properties, if the set $\mathcal{S}$ satisfies the following Condition (A) with a large β and a small γ.

Condition A: Let β, γ, η, $\overline{d}$ be real numbers, $0 < \beta$, $0 < \eta < 1$, and $\overline{d} > 0$. We say that a set $\mathcal{S}$ in a Tanner graph G satisfies Condition (A) with respect to β, γ, η, $\overline{d}$, if the following conditions hold:

- the set $\mathcal{S}$ has β vertex expansion property;
- the set $\mathcal{S}$ has (η, γ) uniform-edge-distribution property; and
- the neighboring check nodes $N(\mathcal{S})$ of $\mathcal{S}$ in the graph G have an average degree less than $\overline{d}$ in the subgraph induced by $\mathcal{S} \cup N(\mathcal{S})$.

Theorem 4.3.4 *Let $\mathcal{S}$ be a stopping set in the Tanner graph G_k. Assume that the set $\mathcal{S}$ satisfies Condition (A) with respect to β, η, γ, $\overline{d}$. Let θ be a real number such that $\theta|\mathcal{S}|$ is an integer, $\theta \leq \eta$. Then, the conditional expectation $\mathbb{E}\left[|\mathcal{B}(\mathcal{S}, \theta)||G_k\right]$ is upper bounded as follows.*

$$\mathbb{E}\left[|\mathcal{B}(\mathcal{S}, \theta)||G_k\right] \leq \exp\left\{\left[H_e(\theta) + (1-\theta)\log_e 2 - 2(\beta - \gamma\theta)\left(\frac{1}{2}\right)^{(\beta\overline{d}/(\beta-\gamma\theta))}\right]|\mathcal{S}|\right\}. \tag{4.6}$$

Proof: The proof will be provided in Section 4.4.

Theorem 4.3.5 *Let $\mathcal{S}$ be a stopping set in the Tanner graph G_k. Assume that the set $\mathcal{S}$ satisfies Condition (A) with respect to β, η, γ, $\overline{d}$. Let θ_0 be a positive real number, such that $\theta_0 \leq \eta$, and for all $0 < \theta \leq \theta_0$,*

$$H_e(\theta) + (1-\theta)\log_e 2 - 2(\beta - \gamma\theta)\left(\frac{1}{2}\right)^{(\beta\overline{d}/(\beta-\gamma\theta))} \leq -\Delta, \tag{4.7}$$

where Δ is a positive real number. Then, with high probability, there is no stopping set $\widehat{\mathcal{S}}$ with $\pi_{k+1}(\widehat{\mathcal{S}}) = \mathcal{S}$, $|\widehat{\mathcal{S}}| \leq (1+\theta_0)|\mathcal{S}|$ in G_{k+1}, if $|\mathcal{S}|$ is sufficiently large.

Proof: The proof will be provided in Section 4.5.

Consider a stopping set $\mathcal{S}$ with strong graph expansion properties. According to Theorem 4.3.4,

$$\begin{aligned}\mathbb{E}\left[|\mathcal{B}(\mathcal{S}, \theta)||G_k\right] &\leq \exp\left\{\left[H_e(\theta) + (1-\theta)\log_e 2 - 2(\beta - \gamma\theta)\left(\frac{1}{2}\right)^{(\beta\overline{d}/(\beta-\gamma\theta))}\right]|\mathcal{S}|\right\} \\ &\approx 0\end{aligned} \tag{4.8}$$

for small θ. According to Theorem 4.3.5, with high probability, there is no stopping set $\widehat{\mathcal{S}}$ in G_{k+1}, $\pi_{k+1}(\widehat{\mathcal{S}}) = \mathcal{S}$, with cardinality less than $(1+\theta_0)|\mathcal{S}|$. The above discussion implies that stopping sets with strong graph expansion properties are less problematic in terms of introducing low-weight stopping sets.

Based on the above discussion, we propose the following design criteria for designing codes with low error floors:

- the protograph should have a large minimal weight of stopping sets;
- the protograph should not contain low-weight stopping sets with weak graph expansion properties.

4.4 Proof of Theorem 4.3.4

We first consider a randomly generated variable node set $\widehat{\mathcal{S}}$ in the graph G_{k+1} chosen from all variable node sets A, satisfying $\pi_{k+1}(A) = \mathcal{S}$, and $|A| = (1+\theta)|\mathcal{S}|$ with equal probability. Let E denote the random event that the set $\widehat{\mathcal{S}}$ is a stopping set. We can bound the probability of this random event as follows.

$$\begin{aligned}
\mathbb{P}\{E\} &\overset{(a)}{=} \sum_{G} \mathbb{P}(G_{k+1} = G)\mathbb{P}(E|G_{k+1} = G) \\
&\overset{(b)}{=} \sum_{G} \mathbb{P}(G_{k+1} = G)\frac{n_1(G)}{n_2(G)} \\
&\overset{(c)}{\geq} \sum_{G} \mathbb{P}(G_{k+1} = G)\frac{n_1(G)}{\exp\{[H_e(\theta) + \log_e(2)(1-\theta)]|\mathcal{S}|\}} \\
&= \exp\{-|\mathcal{S}|H_e(\theta) - (\log_e 2)(1-\theta)|\mathcal{S}|\}\,\mathbb{E}\left[|\mathcal{B}(\mathcal{S},\theta)||G_k\right] \qquad (4.9)
\end{aligned}$$

In the above inequality, (a) follows from the total probability theorem, where G denotes the realization of the random graph G_{k+1}; (b) follows from a simple counting argument on the discrete probability space where $n_1(G)$ denotes the number of stopping sets A in G with $\pi_{k+1}(A) = \mathcal{S}$, $|A| = (1+\theta)|\mathcal{S}|$, and where $n_2(G)$ denotes the number of variable node sets A in G with $\pi_{k+1}(A) = \mathcal{S}$, $|A| = (1+\theta)|\mathcal{S}|$; (c) follows from the fact that the number of variable node sets A, $\pi_{k+1}(A) = \mathcal{S}$ with cardinality $(1+\theta)|\mathcal{S}|$ is

$$\binom{|\mathcal{S}|}{\theta|\mathcal{S}|} 2^{(1-\theta)|\mathcal{S}|} \qquad (4.10)$$

and the Stirling approximation [8, page 284]

$$\frac{2^{nH_2(k/n)}}{n+1} \leq \binom{n}{k} \leq 2^{nH_2(k/n)} \tag{4.11}$$

Given the variable node set $\widehat{\mathcal{S}}$, let $\mathcal{S}_1$ denote the set of variable nodes u in $\mathcal{S}$ with $\pi^{-1}(u) \subset \widehat{\mathcal{S}}$. Let $\mathcal{S}_0 = \mathcal{S} \backslash \mathcal{S}_1$. It is clear that $|\mathcal{S}_1| = \theta|\mathcal{S}|$, $|\mathcal{S}_0| = (1-\theta)|\mathcal{S}|$. Let the set $\mathcal{C}$ denote the set of check nodes c such that c is in $N(\mathcal{S})$, and c is connected to $\mathcal{S}_1$ no more than once. We further consider an arbitrary chosen check node set $\widehat{\mathcal{C}}$ in G_{k+1} such that $\pi_{k+1}(\widehat{\mathcal{C}}) = \mathcal{C}$, $|\widehat{\mathcal{C}}| = |\mathcal{C}|$. It is clear that for each pair of check nodes $c_1, c_2 \in \pi_{k+1}^{-1}(c)$ with $c \in \mathcal{C}$, exactly one of c_1, c_2 is in $\widehat{\mathcal{C}}$. By the assumptions,

$$|\widehat{\mathcal{C}}| \geq (\beta - \gamma\theta)|\mathcal{S}|. \tag{4.12}$$

Let m_j denote the number of check nodes $c \in \widehat{\mathcal{C}}$ such that $\pi_{k+1}(c)$ is not connected to any variable node in $\mathcal{S}_1$, and $\pi_{k+1}(c)$ has degree j in the subgraph induced by variable node set $\mathcal{S}$ and its neighboring check node set $N(\mathcal{S})$. Let n_j denote the number of check nodes $c \in \widehat{\mathcal{C}}$ such that $\pi_{k+1}(c)$ is connected to exactly one variable node in $\mathcal{S}_1$, and $\pi_{k+1}(c)$ has degree j in the subgraph induced by variable node set $\mathcal{S}$ and its neighboring check node set $N(\mathcal{S})$.

The probability that the set $\widehat{\mathcal{S}}$ is a stopping set can be upper bounded as follows.

$$\begin{aligned}
\mathbb{P}\{E\} &\overset{(a)}{\leq} \prod_j \left\{1 - j\left(\frac{1}{2}\right)^j\right\}^{m_j} \prod_j \left\{1 - \left(\frac{1}{2}\right)^{j-1}\right\}^{n_j} \\
&= \exp\left\{\sum_j m_j \log_e\left[1 - j\left(\frac{1}{2}\right)^j\right]\right\} \exp\left\{\sum_j n_j \log_e\left[1 - \left(\frac{1}{2}\right)^{j-1}\right]\right\} \\
&\overset{(b)}{\leq} \exp\left\{(-1)\sum_j m_j j\left(\frac{1}{2}\right)^j\right\} \exp\left\{(-1)\sum_j n_j \left(\frac{1}{2}\right)^{j-1}\right\} \\
&\overset{(c)}{\leq} \exp\left\{(-2)\sum_j m_j \left(\frac{1}{2}\right)^j\right\} \exp\left\{(-2)\sum_j n_j \left(\frac{1}{2}\right)^j\right\} \\
&\overset{(d)}{\leq} \exp\left\{(-2)\left(\sum_j m_j + n_j\right)\left(\frac{1}{2}\right)^{(\beta\bar{d}/(\beta-\gamma\theta))}\right\} \\
&\overset{(e)}{\leq} \exp\left\{(-2)(\beta-\gamma\theta)|\mathcal{S}|\left(\frac{1}{2}\right)^{\beta\bar{d}/(\beta-\gamma\theta)}\right\}
\end{aligned} \tag{4.13}$$

The right hand side of (a) is the probability that the check node set $\widehat{\mathcal{C}}$ does not contain any check node that is connected to $\widehat{\mathcal{S}}$ exactly once. The inequality (b) follows from the inequality $\log_e(1-x) \leq -x$, for $0 < x < 1$. The inequality (c) follows from the fact that the degree j is always greater than or equal to two. The inequality (d) follows from the convexity of exponential functions and the fact that the average of degree j is less than $\beta\overline{d}/(\beta - \gamma\theta)$. The inequality (e) follows from the inequalities (4.12).

The Theorem follows from the inequalities (4.9), and (4.13).

4.5 Proof of Theorem 4.3.5

Let $\mathcal{M}$ denote the number of stopping sets in G_{k+1}, such that $\pi_{k+1}(\widehat{\mathcal{S}}) = \mathcal{S}$, $|\widehat{\mathcal{S}}| \leq (1+\theta_0)|\mathcal{S}|$. Let E denote the random event that there exists a $\widehat{\mathcal{S}}$, such that $\pi_{k+1}(\widehat{\mathcal{S}}) = \mathcal{S}$, $|\widehat{\mathcal{S}}| \leq (1+\theta_0)|\mathcal{S}|$.

According to the assumptions and Theorem 4.3.4, we have

$$\begin{aligned}
\mathbb{E}[\mathcal{M}|G_k] &= \sum_{i=0}^{i<\theta_0|\mathcal{S}|} \mathbb{E}\left[\mathcal{B}\left(\mathcal{S}, i/|\mathcal{S}|\right)|G_k\right] \\
&\leq \sum_{i=0}^{i<\theta_0|\mathcal{S}|} \exp\left\{-\Delta|\mathcal{S}|\right\} \\
&\leq (\theta_0|\mathcal{S}|+1)\exp\left\{-\Delta|\mathcal{S}|\right\}. \qquad (4.14)
\end{aligned}$$

By the Markov inequality, we have

$$\mathbb{P}[E] \leq \mathbb{E}[\mathcal{M}|G_k] \leq (\theta_0|\mathcal{S}|+1)\exp\left\{-\Delta|\mathcal{S}|\right\}. \qquad (4.15)$$

This upper bound of $\mathbb{P}[E]$ is close to one for sufficiently large $|\mathcal{S}|$. The theorem follows.

4.6 Numerical Results

In this section, we present simulation results for the proposed code construction scheme. We construct capacity-approaching LDPC codes with varying rates, block-lengths, and degree distributions by the proposed code construction scheme. The protographs are obtained by exhaustive searching, such that low-weight stopping

sets with weak graph expansion properties are avoided. We compare the codes by the proposed construction scheme with random codes by the standard construction and high-girth codes by the PEG construction schemes [15], [16].

We depict the block error probabilities of various LDPC codes with degree distributions in Table 4.1 in Figs. 4.2, 4.4, 4.6. The bit error probabilities are shown in Figs. 4.3, 4.5, 4.7. The rates of the codes are 0.5. The blocklengths are $4k$ bits, $8k$ bits, and $16k$ bits. The protograph is shown in Table 4.5. The degree distributions are obtained by the numerical optimization algorithm in Chapter 3.

We depict the block error probabilities of various LDPC codes with degree distributions in Table 4.2 in Figs. 4.8, 4.10, 4.12. The bit error probabilities are shown in Figs. 4.9, 4.11, 4.13. The blocklengths are $4k$ bits, $8k$ bits, and $16k$ bits. The degree distributions are obtained by LdpcOpt, an on-line degree distribution optimizer for LDPC code ensembles [19]. The rates of the random codes by the standard construction and high-girth codes are 0.5. The rates of the 2-lift based codes are 0.4688. The code rate is slightly lower than the design rate, due to that the coefficients of the degree distributions are rounded to integers in the protograph construction.

We depict the block error probabilities of various LDPC codes with degree distributions in Table 4.3 in Figs. 4.14, 4.16, 4.18. The bit error probabilities are shown in Figs. 4.15, 4.17, 4.19. The blocklengths are $4k$ bits, $8k$ bits, and $16k$ bits. The degree distributions are obtained by LdpcOpt. The rates of the random codes by the standard construction and high-girth codes are 0.75. The rates of the 2-lift based codes are 0.7188.

We depict the block error probabilities of various LDPC codes with degree distributions in Table 4.4 in Figs. 4.20, 4.22, 4.24. The bit error probabilities are shown in Figs. 4.21, 4.23, 4.25. The rates of the codes are 0.25. The blocklengths are $4k$ bits, $8k$ bits, and $16k$ bits. The degree distributions are obtained by the numerical optimization algorithm in Chapter 3.

In all the figures, the error probabilities of the codes by the proposed construction are represented by solid curves, the error probabilities of the codes by the standard construction are represented by solid curves marked with triangles, the error probabilities of the high-girth codes are represented by solid curves marked with circles. Due to the numerical difficulties in accurately estimating very low error probabilities, error probabilities for certain channel parameters are not shown in the figures.

From the above numerical results, we can see that the capacity-approaching codes by the standard construction have high error floors as predicted theoretically.

The error floors can be lowered by girth-conditioning. However, the improvement is marginal. Both the code by the standard construction and high-girth codes exhibit high error floors. The error floors in block error probabilities are usually higher than 10^{-5}, and the error floors in bit error probabilities are usually higher than 10^{-7}. On the other hand, 2-lift based codes do not exhibit any error floor higher than 10^{-5} in block error probability and 10^{-7} in bit error probability. The codes by the proposed construction schemes significantly outperform the other codes in the error floor regions (in some cases, also in waterfall regions). The proposed code construction scheme provides a promising approaching to designing LDPC codes with very low error probabilities.

left distribution		right distribution	
degree	coefficient	degree	coefficient
2	0.189633	2	0
3	0.480613	3	0
4	0	4	0
5	0	5	0
6	0	6	0
7	0	7	0.869345
8	0	8	0.130655
9	0	9	0
10	0	10	0
11	0	11	0
12	0.103197	12	0
13	0.226557	13	0
14	0	14	0
15	0	15	0
16	0	16	0

Table 4.1: The degree distributions for one code ensemble with rate 0.5. The degree distributions are obtained by numerical optimization methods in Chapter 3.

left distribution		right distribution	
degree	coefficient	degree	coefficient
2	0.285486	2	0
3	0.313850	3	0
4	0	4	0
5	0	5	0
6	0	6	0
7	0	7	1
8	0.199606	8	0
9	0	9	0
10	0	10	0
11	0	11	0
12	0	12	0
13	0	13	0
14	0	14	0
15	0.201058	15	0
16	0	16	0

Table 4.2: The degree distributions for one code ensemble with rate 0.5. The degree distributions are obtained by LdpcOpt [19].

left distribution		right distribution	
degree	coefficient	degree	coefficient
2	0.133279	2	0
3	0.513769	3	0
4	0	4	0
5	0	5	0
6	0	6	0
7	0	7	0
8	0	8	0
9	0	9	0
10	0	10	0
11	0	11	0
12	0.252823	12	0
13	0.100129	13	0
14	0	14	0
15	0	15	1
16	0	16	0

Table 4.3: The degree distributions for one code ensemble with rate 0.75. The degree distributions are obtained by LdpcOpt [19].

left distribution		right distribution	
degree	coefficient	degree	coefficient
2	0.223581	2	0
3	0.422349	3	0
4	0	4	0.487896
5	0	5	0.422673
6	0	6	0
7	0	7	0
8	0.129514	8	0
9	0	9	0
10	0	10	0
11	0	11	0
12	0	12	0
13	0	13	0
14	0	14	0
15	0	15	0
16	0.224556	16	0.089431

Table 4.4: The degree distributions for one code ensemble with rate 0.25. The degree distributions are obtained by numerical optimization methods in Chapter 3.

$$\left[\begin{array}{cccccccccccccccc}
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\
0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 \\
1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \\
1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 \\
1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
1 & 1 & 1 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
1 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 0 \\
1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 0 \\
0 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1
\end{array}\right]$$

Table 4.5: The parity-check matrix corresponds to a protograph obtained by computational searching. The rows correspond to variable nodes. The columns correspond to parity-check equations.

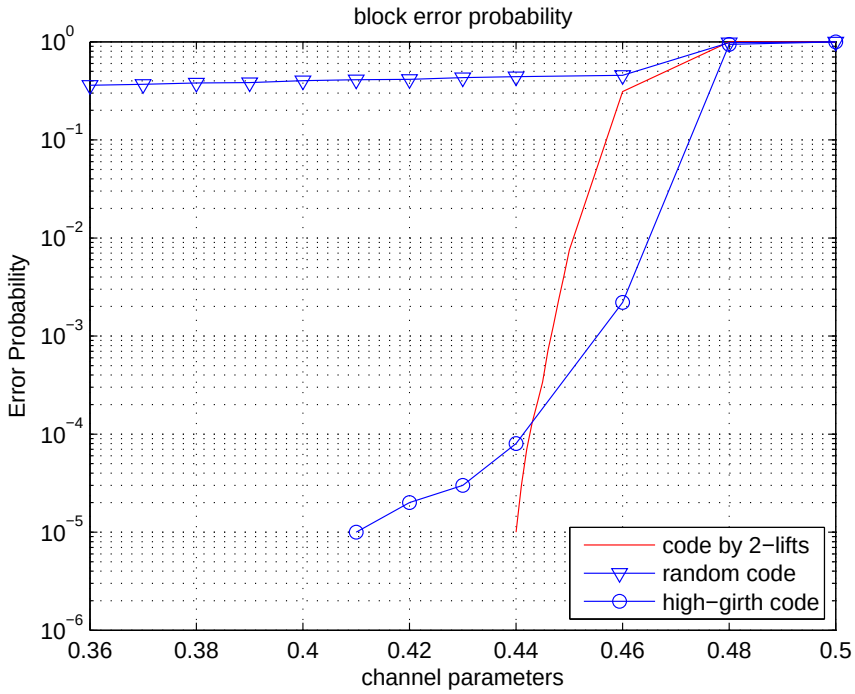


Figure 4.2: Block error probabilities of various LDPC codes. The code rate is 0.5. The block length is $16k$ bits. The numbers on X-axis represent varying channel parameters. The numbers on Y-axis represent the error probabilities.

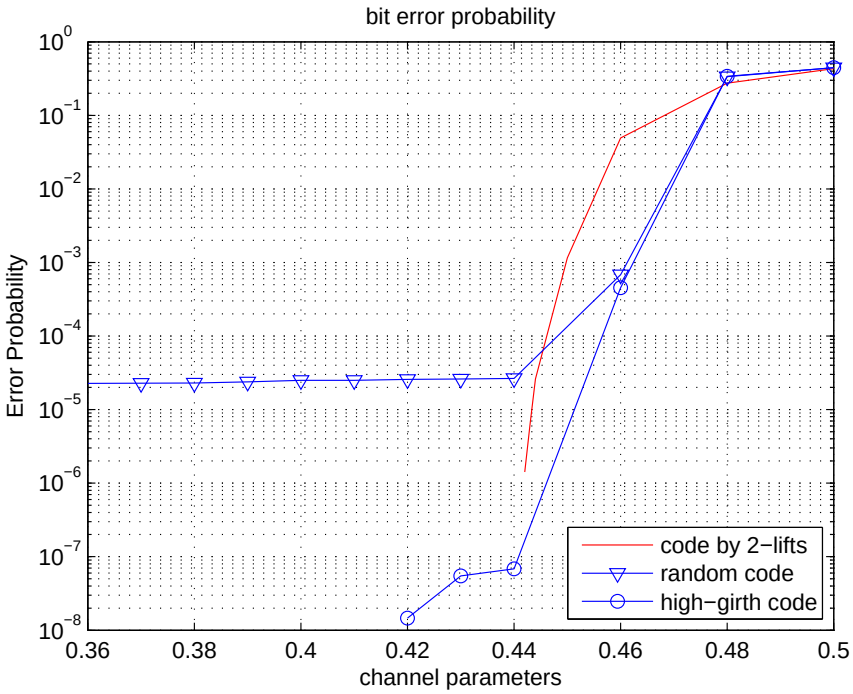


Figure 4.3: Bit error probabilities of various LDPC codes. The code rate is 0.5. The block length is $16k$ bits. The numbers on X-axis represent varying channel parameters. The numbers on Y-axis represent the error probabilities.

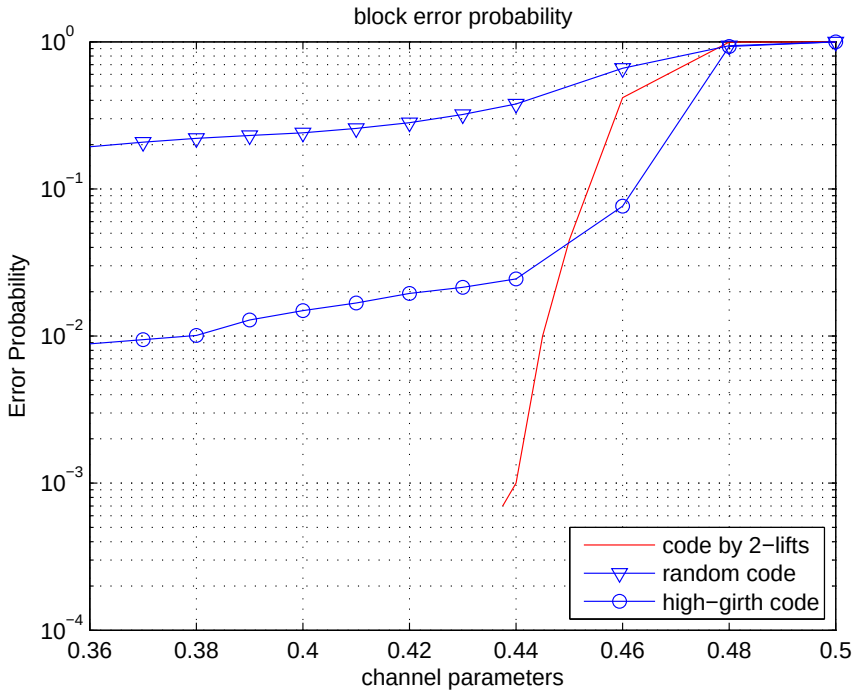


Figure 4.4: Block error probabilities of various LDPC codes. The code rate is 0.5. The block length is $8k$ bits. The numbers on X-axis represent varying channel parameters. The numbers on Y-axis represent the error probabilities.

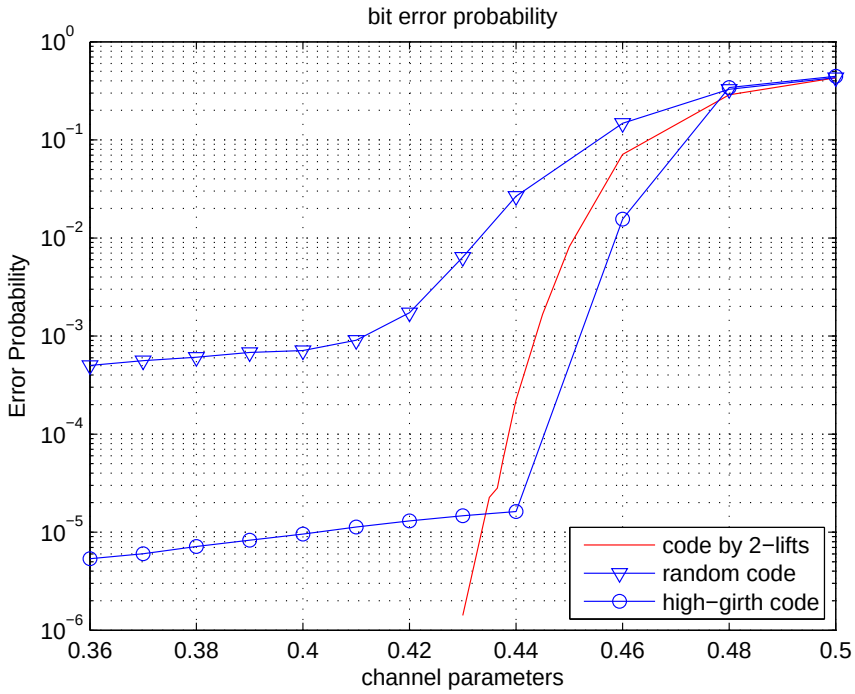


Figure 4.5: Bit error probabilities of various LDPC codes. The code rate is 0.5. The block length is $8k$ bits. The numbers on X-axis represent varying channel parameters. The numbers on Y-axis represent the error probabilities.

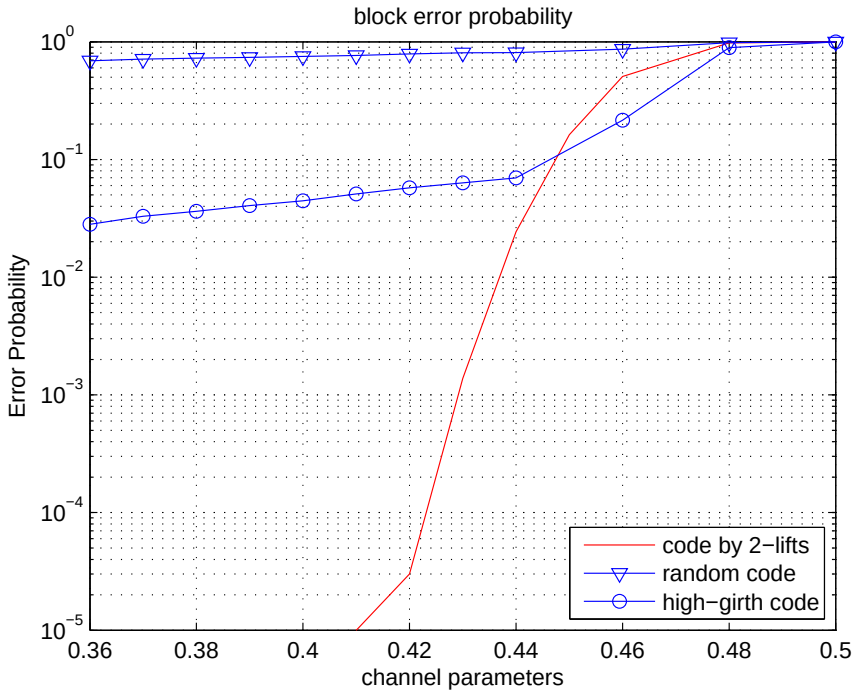


Figure 4.6: Block error probabilities of various LDPC codes. The code rate is 0.5. The block length is $4k$ bits. The numbers on X-axis represent varying channel parameters. The numbers on Y-axis represent the error probabilities.

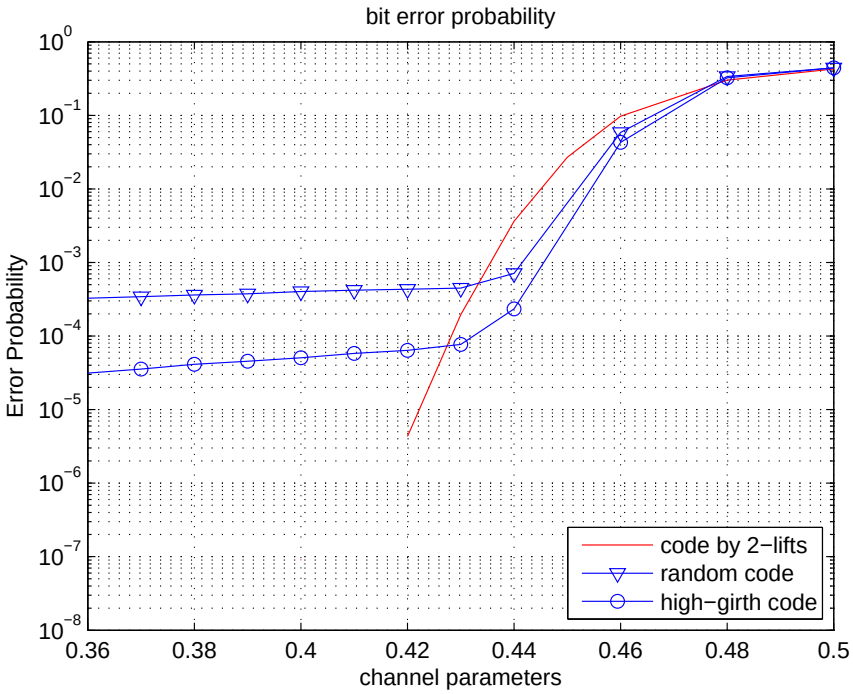


Figure 4.7: Bit error probabilities of various LDPC codes. The code rate is 0.5. The block length is $4k$ bits. The numbers on X-axis represent varying channel parameters. The numbers on Y-axis represent the error probabilities.

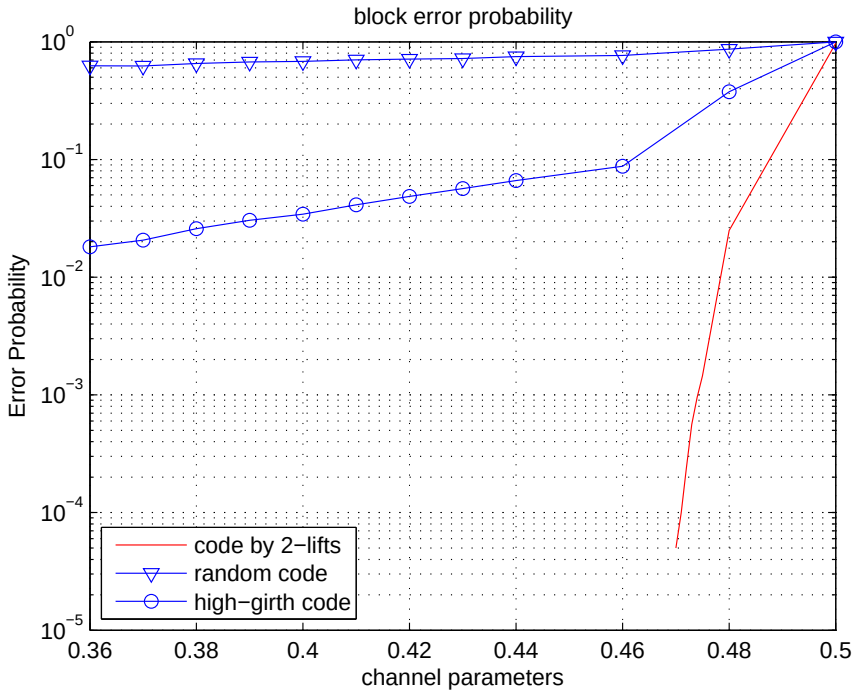


Figure 4.8: Block error probabilities of various LDPC codes. The code rate is 0.5. The block length is $16k$ bits. The numbers on X-axis represent varying channel parameters. The numbers on Y-axis represent the error probabilities.

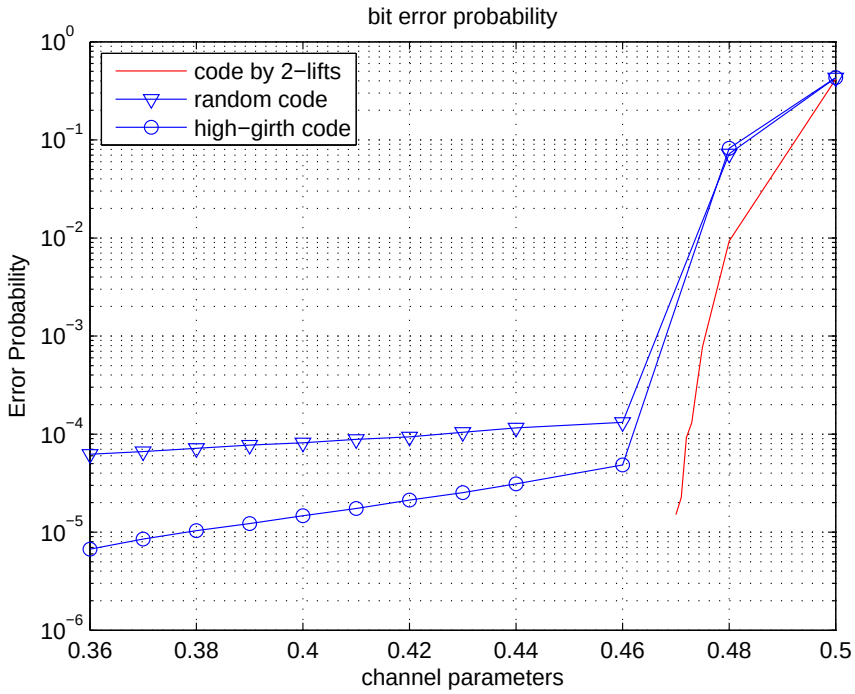


Figure 4.9: Bit error probabilities of various LDPC codes. The code rate is 0.5. The block length is $16k$ bits. The numbers on X-axis represent varying channel parameters. The numbers on Y-axis represent the error probabilities.

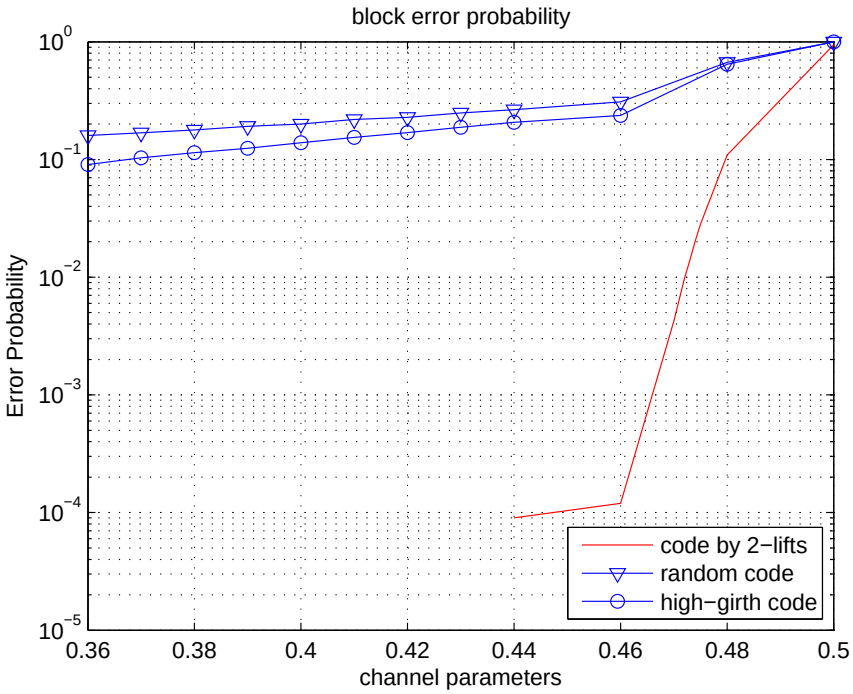


Figure 4.10: Block error probabilities of various LDPC codes. The code rate is 0.5. The block length is $8k$ bits. The numbers on X-axis represent varying channel parameters. The numbers on Y-axis represent the error probabilities.

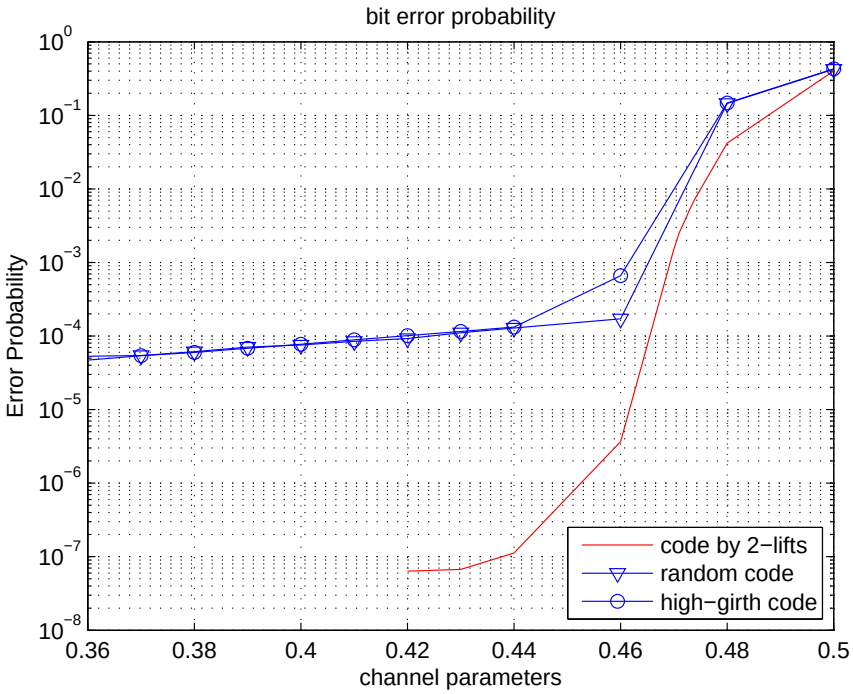


Figure 4.11: Bit error probabilities of various LDPC codes. The code rate is 0.5. The block length is $8k$ bits. The numbers on X-axis represent varying channel parameters. The numbers on Y-axis represent the error probabilities.

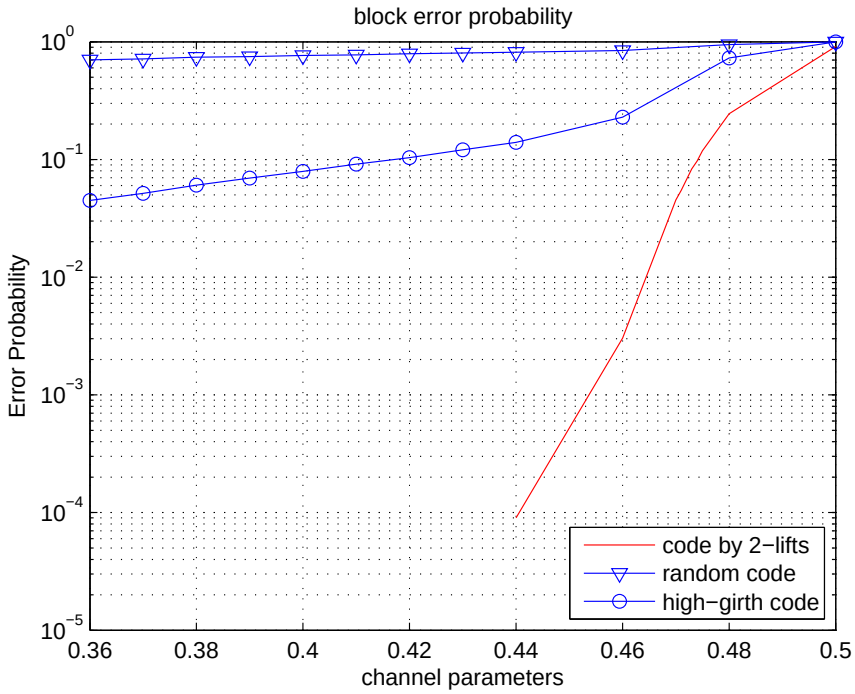


Figure 4.12: Block error probabilities of various LDPC codes. The code rate is 0.5. The block length is $4k$ bits. The numbers on X-axis represent varying channel parameters. The numbers on Y-axis represent the error probabilities.

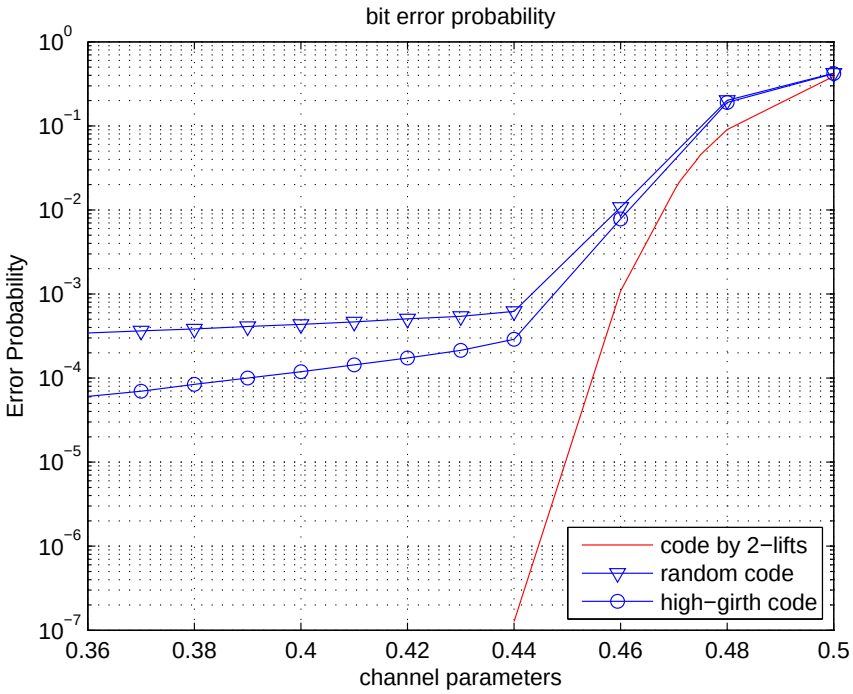


Figure 4.13: Bit error probabilities of various LDPC codes. The code rate is 0.5. The block length is $4k$ bits. The numbers on X-axis represent varying channel parameters. The numbers on Y-axis represent the error probabilities.

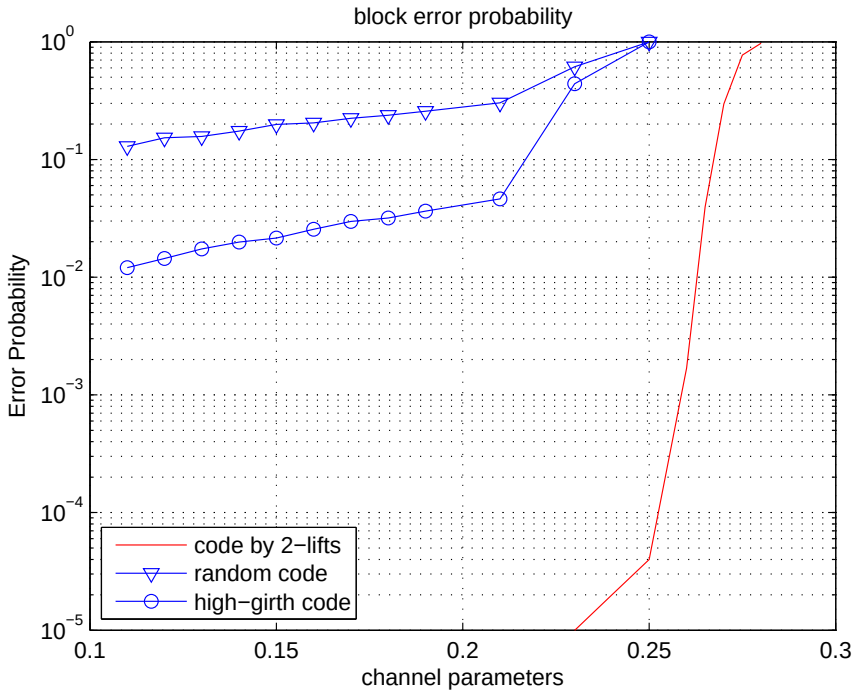


Figure 4.14: Block error probabilities of various LDPC codes. The code rate is 0.75. The block length is $16k$ bits. The numbers on X-axis represent varying channel parameters. The numbers on Y-axis represent the error probabilities.

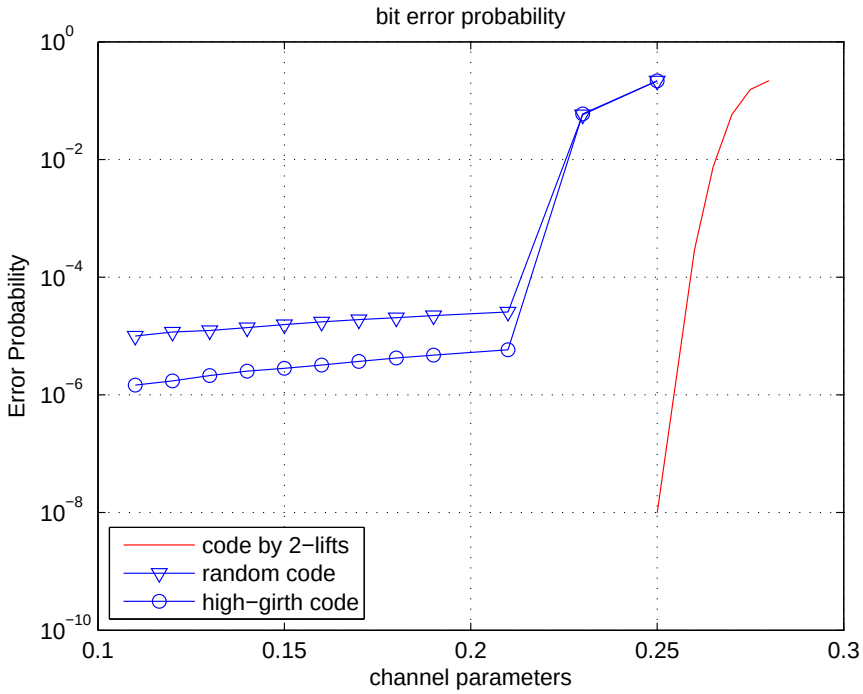


Figure 4.15: Bit error probabilities of various LDPC codes. The code rate is 0.75. The block length is $16k$ bits. The numbers on X-axis represent varying channel parameters. The numbers on Y-axis represent the error probabilities.

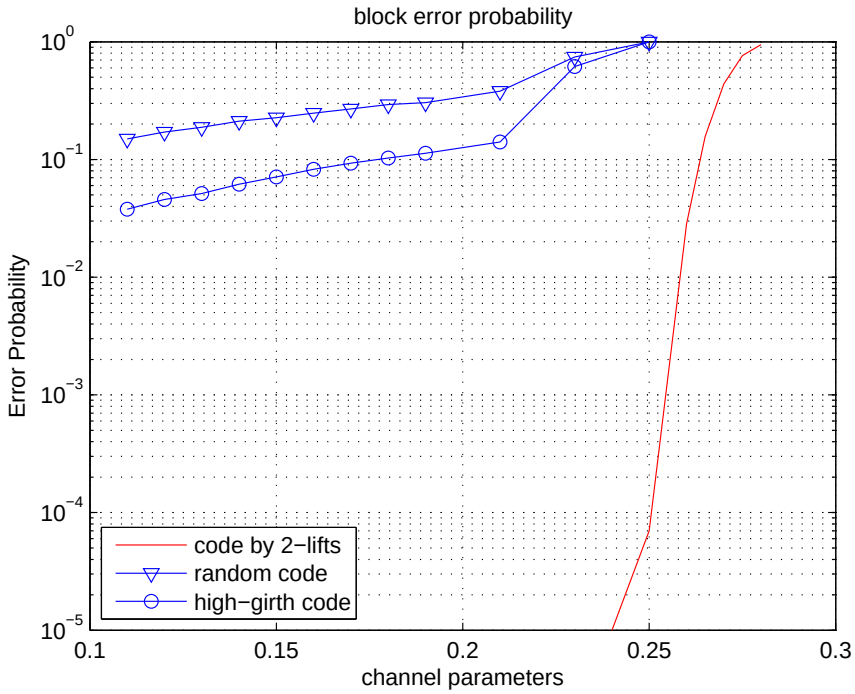


Figure 4.16: Block error probabilities of various LDPC codes. The code rate is 0.75. The block length is $8k$ bits. The numbers on X-axis represent varying channel parameters. The numbers on Y-axis represent the error probabilities.

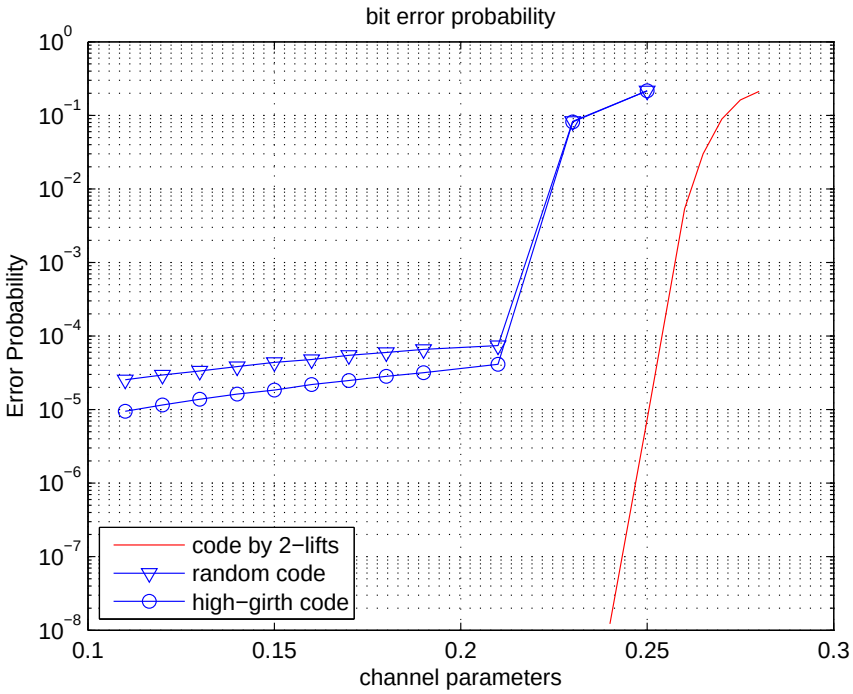


Figure 4.17: Bit error probabilities of various LDPC codes. The code rate is 0.75. The block length is $8k$ bits. The numbers on X-axis represent varying channel parameters. The numbers on Y-axis represent the error probabilities.

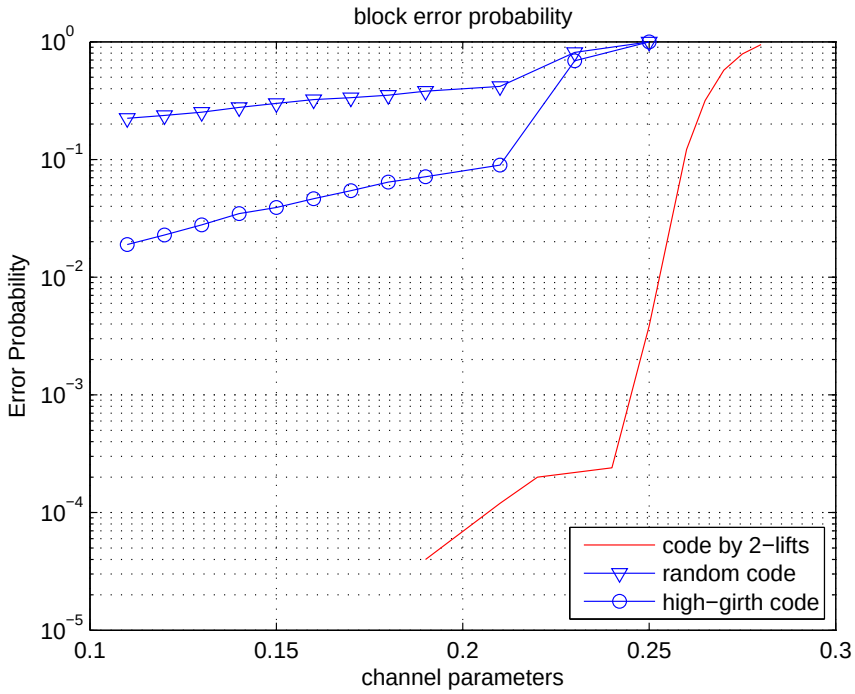


Figure 4.18: Block error probabilities of various LDPC codes. The code rate is 0.75. The block length is $4k$ bits. The numbers on X-axis represent varying channel parameters. The numbers on Y-axis represent the error probabilities.

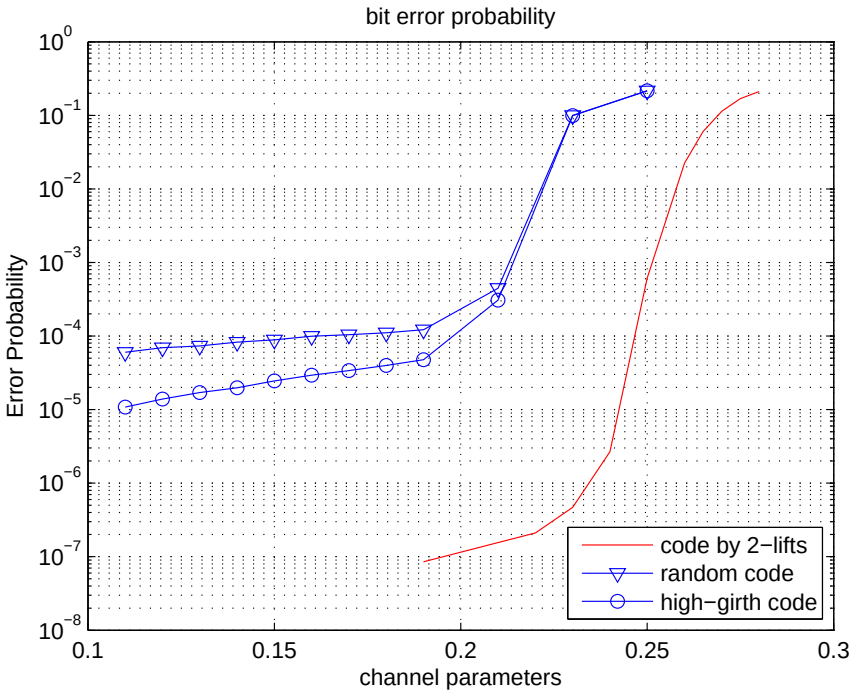


Figure 4.19: Bit error probabilities of various LDPC codes. The code rate is 0.75. The block length is $4k$ bits. The numbers on X-axis represent varying channel parameters. The numbers on Y-axis represent the error probabilities.

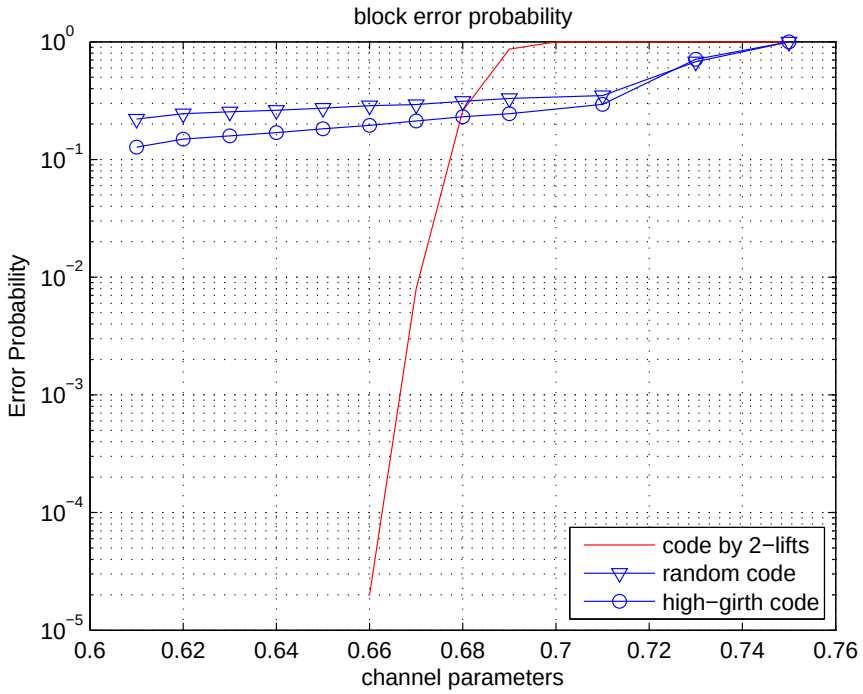


Figure 4.20: Block error probabilities of various LDPC codes. The code rate is 0.25. The block length is $16k$ bits. The numbers on X-axis represent varying channel parameters. The numbers on Y-axis represent the error probabilities.

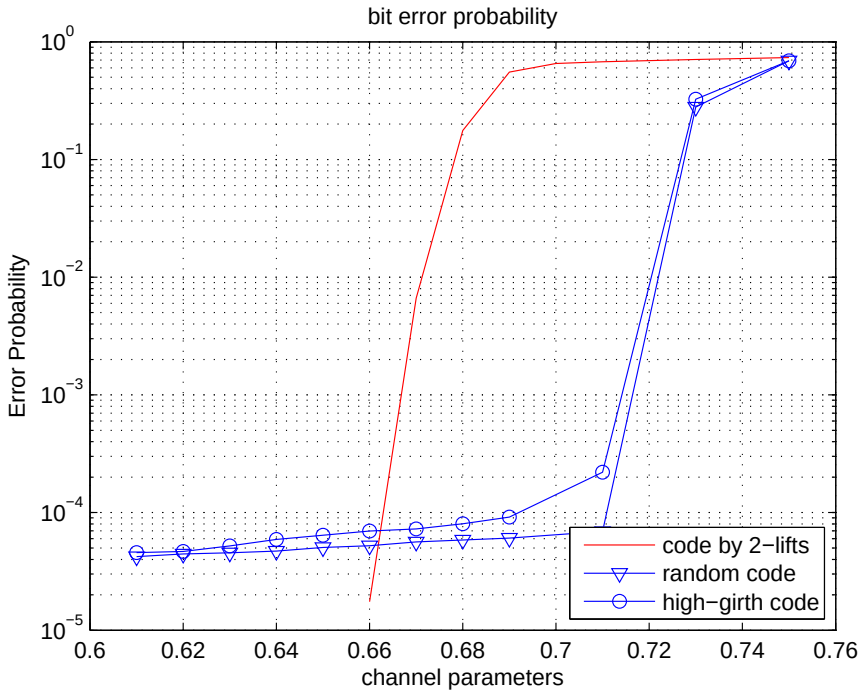


Figure 4.21: Bit error probabilities of various LDPC codes. The code rate is 0.25. The block length is $16k$ bits. The numbers on X-axis represent varying channel parameters. The numbers on Y-axis represent the error probabilities.

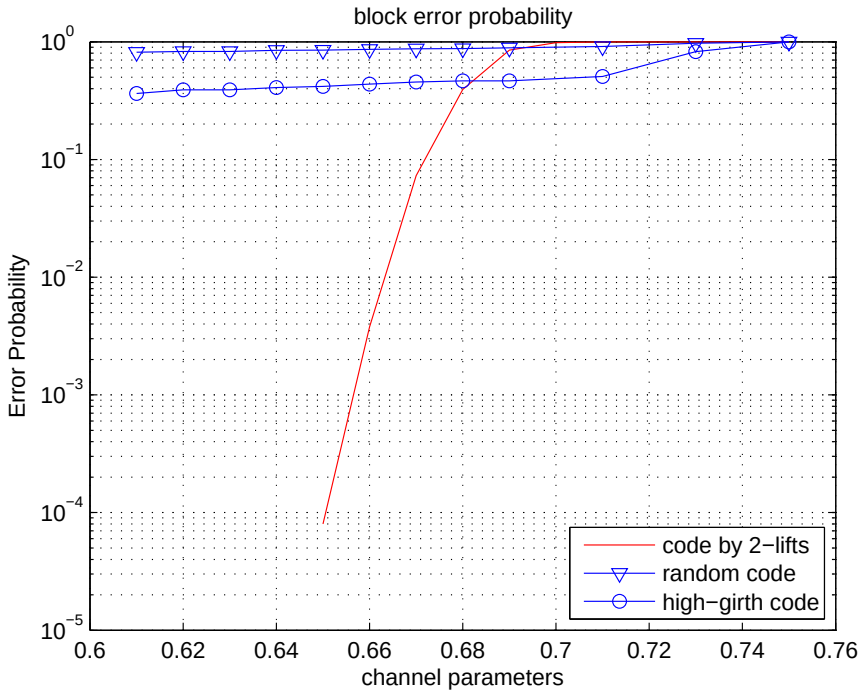


Figure 4.22: Block error probabilities of various LDPC codes. The code rate is 0.25. The block length is $8k$ bits. The numbers on X-axis represent varying channel parameters. The numbers on Y-axis represent the error probabilities.

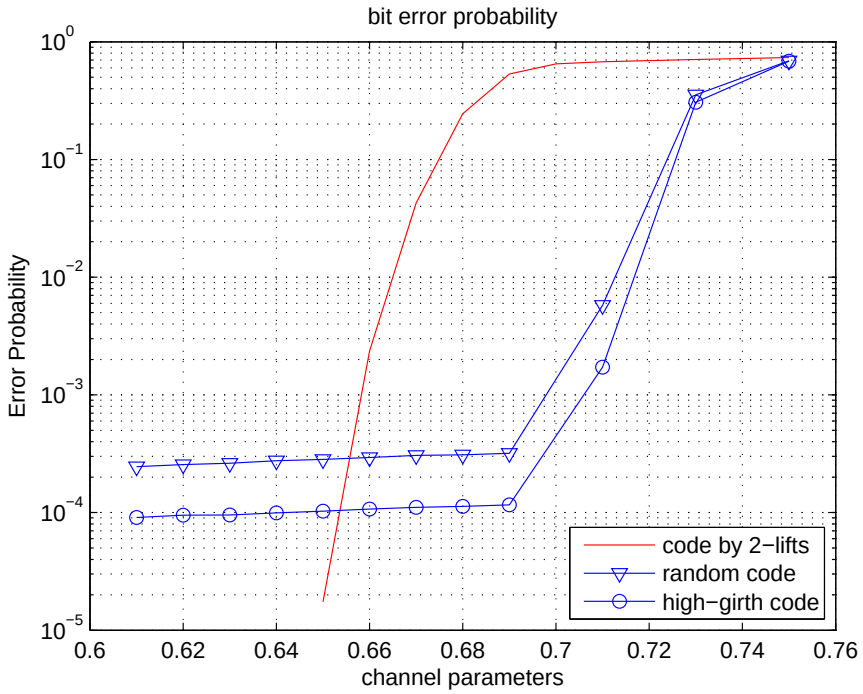


Figure 4.23: Bit error probabilities of various LDPC codes. The code rate is 0.25. The block length is $8k$ bits. The numbers on X-axis represent varying channel parameters. The numbers on Y-axis represent the error probabilities.

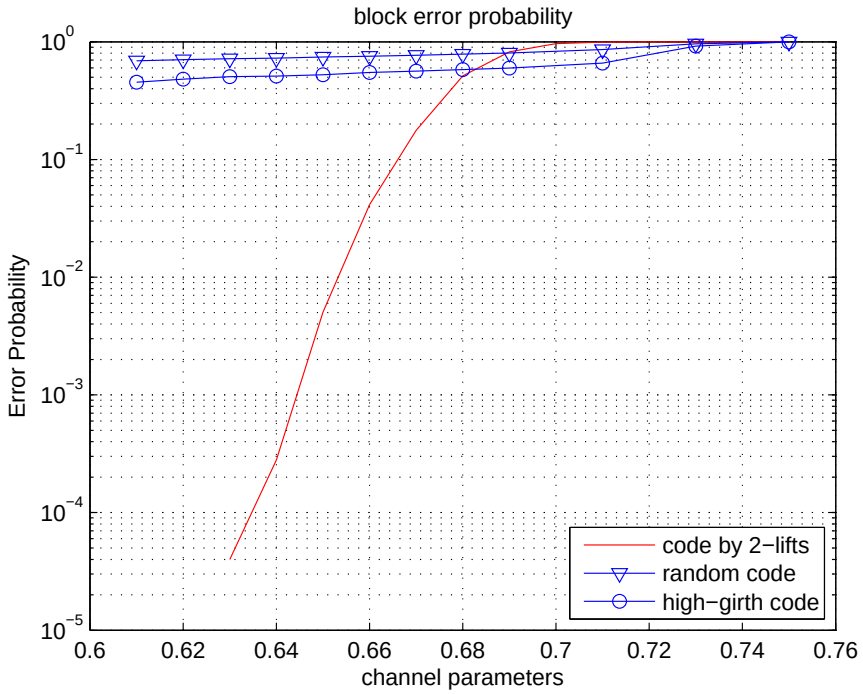


Figure 4.24: Block error probabilities of various LDPC codes. The code rate is 0.25. The block length is $4k$ bits. The numbers on X-axis represent varying channel parameters. The numbers on Y-axis represent the error probabilities.

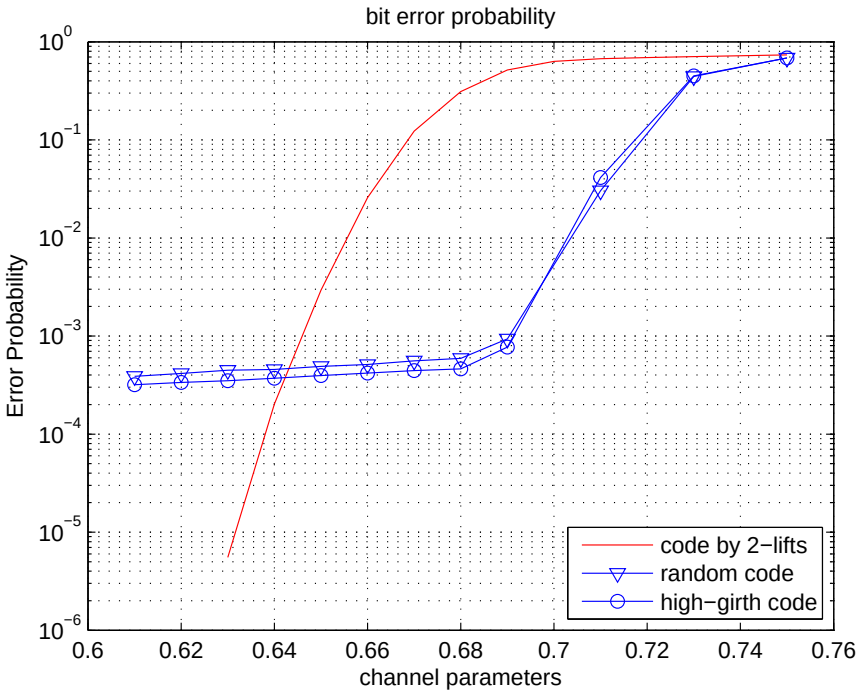


Figure 4.25: Bit error probabilities of various LDPC codes. The code rate is 0.25. The block length is $4k$ bits. The numbers on X-axis represent varying channel parameters. The numbers on Y-axis represent the error probabilities.

4.7 Conclusion

In this chapter, we propose a new construction scheme for LDPC codes based on a series of random 2-lifts. We propose design criteria for the resulting codes to have low error floors. The design criteria state that stopping sets with weak graph expansion properties should be avoided in the base graphs. Numerical results show that the codes by the proposed construction have significantly lower error floors compared with codes by the standard construction and the high-girth codes, while at the same time, maintain capacity-approaching performance.

Chapter 5

Conclusion

LDPC codes have received much attention in recent years. The advantages of the codes include capacity-approaching performance, and highly-efficient parallel decoding algorithms. LDPC codes are enabling technology for many new applications. However, a lot of design issues still need to be addressed, so that, optimality in terms of many aspects are achieved.

In this thesis, we consider the problem of designing LDPC codes for real-time multimedia communications. We investigate two code design issues: designing LDPC codes with fast decoding speeds, and constructing LDPC codes by 2-lifts with the motivation of designing LDPC codes with low error floors.

For the issue of designing LDPC codes with fast decoding speeds, we present a tractable optimization framework based on an asymptotic approximation. Numerical results confirm that the asymptotic approximation is tight for even non-extremely-limiting cases. The optimized codes by the proposed approach have significantly faster decoding speeds compared with un-optimized codes.

For the issue of constructing LDPC codes by 2-lifts, we present an analysis on the distributions of low-weight stopping sets. Based on the analysis, design criteria for designing codes with low error floors are presented. Numerical results confirm that the resulting codes do not have visible error floors.

Chapter 6

Future Research

This thesis focuses on designing LDPC code for real-time multimedia communications over packet networks. However, the proposed code design principle and construction schemes can be further generalized to other application scenarios. In addition, there are still many design issues needed to be addressed in the future. Some promising future research directions are listed as follows:

- The design of LDPC codes with fast decoding speeds for other important channel can be investigated. The asymptotic approximation for the number of decoding iterations can be generalized to the cases of other channels. For example, the case of binary-input memoryless symmetric channels has been considered in [37].
- The design of 2-lift based codes with low error floors for other channels can be investigated.
- The design of 2-lift based codes with other hardware implementation considerations can be investigated. The examples of the hardware implementation considerations include power consumption, wiring complexity etc.

Bibliography

[1] M. Ardakani, and F.R. Kschischang, "A more Accurate One-dimensional Analysis and Design of Irregular LDPC Codes", *IEEE Transactions on Communications*, Vol. 52, No. 12, pp. 2106 - 2114, December 2004

[2] Y. Bilu, and N. Linial, "Lifts, Discrepancy and Nearly Optimal Spectral Gaps", *Combinatorica*, to appear.

[3] P.T. Boggs, and J. W. Tolle, "Sequential quadratic programming," *Acta Numerica*, 1995, 1-51.

[4] D. Burshtein, and G. Miller, "Asymptotic enumeration methods for analyzing LDPC codes," *IEEE Trans. on Inform. Theory*, vol. 50, no. 6, pp. 1115-1131, June 2004.

[5] E.K.P. Chong, and S.H. Zak, *An introduction to optimization,* John Wiley & Sons, Inc., Second Edition, 2001.

[6] S.Y. Chung, T. Richardson and R. Urbanke, "Analysis of sum-product decoding of low-density parity-check codes using a Gaussian approximation", *IEEE Transactions on Information Theory*, Vol 47, No. 2, pp. 657-670, Februry 2001.

[7] S. Chung, G. Forney Jr., T.J. Richardson and R. Urbanke, "On the design of low-density parity-check codes within 0.0045 dB of the Shannon limit", *IEEE Communication Letters*, Vol. 5, No. 2, pp. 58-60, Feb. 2001.

[8] T.M. cover, and J.A. Thomas, *Elements of Information Theory*, John Wiley, 1991.

[9] C. Di, R. Urbanke, and T. Richardson, "Weight distributions: how deviant can you be?" *Proc. IEEE Symp. on Inform. Theory*, p. 50, 2001.

[10] C. Di, D. Proietti, I. Telatar, T.J. Richardson, and R. Urbanke, "Finite-Length Analysis of Low-Density Parity-Check Codes on the Binary Erasure Channel", *IEEE Trans. on Inform. Theory*, vol. 48, no. 6, pp. 1570-1579, June 2002.

[11] C. Di, T.J. Richardson, and R.L. Urbanke, "Weight Distribution of Low-Density Parity-Check Codes", *IEEE Trans. on Inform. Theory*, vol. 52, no. 11, p.p. 4839- 4855, Nov. 2006.

[12] G.D. Forney, R. Koetter, F.R. Kschischang, and A. Reznik, "On the Effective Weights of Pseudocodewords for Codes Defined on Graphs with Cycles", *Codes, Systems and Graphical Models*, pp. 102-112, New York: Springer-Verlag 2001.

[13] R.G. Gallager, *"Low-density parity-check codes"*, MIT Press, 1963.

[14] S. Hoory, N. Linial, and A. Wigderson, "Expander Graphs and Their Applications", *Bulletin AMS*, vol. 49, no.4, pp. 439-561, October 2006.

[15] X.Y. Hu, E. Eleftheriou, and D.M. Arnold, "Progressive edge-growth Tanner graph", IEEE Global Telecommunications Conference, 2001.

[16] X.Y. Hu, E. Eleftheriou, and D.M. Arnold, "Irregular progressive edge-growth Tanner graph", *Proceedings of IEEE International Symposium on Information Theory*, 2002.

[17] C. Hsu and A. Anastasopoulos, "Capacity-achieving Codes for Noisy Channels with Bounded Graphical Complexity and Maximum Likelihood Decoding", manuscript.

[18] F.R. Kschischang, B.J. Frey, and H. Loeliger, "Factor graphs and the sum-product algorithm", *IEEE Transactions on Information Theory*, Vol. 47, No. 2, pp. 498-519, February 2001.

[19] LdpcOpt, an on-line degree distribution optimizer for LDPC code ensembles, http://lthcwww.epfl.ch/research/ldpcopt/index.php

[20] S. Litsyn and V. Shevelev, "On ensembles of low-density parity-check codes: Asymptotic distance distributions", *IEEE Trans. on Inform. Theory*, vol. 48, no.4, pp. 887-908, April 2002.

[21] S.L. Litsyn, and V. Shevelev, "Distance distribution in ensembles of irregular low-density parity-check codes", *IEEE Trans. on Inform. Theory*, vol. 49, no. 12, pp. 3140-3159, December 2003.

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

[23] M. Luby, M. Mitzenmacher, A. Shokrollahi, D. Spielman, "Efficient erasure correction codes", *IEEE Transactions on Information Theory*, Vol. 47, No. 2, pp. 569-584, Feb., 2001.

[24] M. Luby, M. Mitzenmacher, M. Shokrollahi and D. Spielman, "Improved low-density parity-check Codes using irregular graphs", *IEEE Transactions on Information Theory*, Vol. 47, No. 2, pp. 585-598, Feb. 2001.

[25] X. Ma, E.-H. Yang, "Low-Density Parity-Check codes with fast decoding convergence speed", p. 277, *Proceeding of IEEE International Symposium on Information Theory*, Chicago, 2004.

[26] X. Ma, E.-H. Yang, "Constructing LDPC Codes by 2-Lifts", Proceedings of the 2007 IEEE International Symposium on Information Theory (ISIT 2007), Nice, France, June 24-29, 2007.

[27] D.J.C. Mackay, and M.S. Postol, "Weaknesses of Margulis and Ramanujan-Margulis Low-Density Parity-Check Codes", *Electronic Notes in Theoretical Computer Science*, vol. 74, pp. 1-8, Oct. 2003.

[28] J. Moura, J. Lu, and H. Zhang, "Structured low-density parity-check codes", *IEEE Signal Processing Magazine*, Vol. 21, No. 1, pp. 42- 55, January 2004.

[29] A. Orlitsky, K. Viswanathan, and J. Zhang, "Stopping set distribution of LDPC code ensembles", *IEEE Trans. on Inform. Theory*, vol. 51, no. 3, pp. 929-953, March 2005.

[30] P. Oswald, A. Shokrollahi, "Capacity-achieving sequences for the erasure channel", *IEEE Transactions on Information Theory*, Vol. 48, No. 12, pp. 3017-3028, December 2002.

[31] T. Richardson and R. Urbanke, "The capacity of low-density parity-check codes under message-passing decoding", *IEEE Transaction on Information Theory*, Vol.47, No. 2, pp. 599-618, Feb. 2001.

[32] T. Richardson, M. Shokrollahi and R. Urbanke, "Design of capacity-approaching irregular low-density parity-check codes", *IEEE Transactions on Information Theory*, Vol. 47, No. 2, pp. 619-637, February 2001.

[33] T.J. Richardson, "Error Floors of LDPC Codes", *Proc. Allerton Conference on Communication, Control, and Computing*, pp. 1426-1435, 2003.

[34] I. Sason and R. Urbanke, "Parity-check density versus performance of binary linear block codes over memoryless symmetric channels", *IEEE Transactions on Information Theory*, vol. 49, No. 7, pp. 1611 - 1635, July 2003.

[35] M.A. Shokrollahi, "New sequences of linear time erasure codes approaching the channel capacity", *Proceedings of AAECC-13, number 1719 of Lecture Notes in computer Science*, pp. 65-76, 1999.

[36] M.A. Shokrollahi, "Capacity-achieving sequences", *Codes, Systems, and Graphical Models, number 123 of IMA volumes in Mathematics and its Applications*, pp. 153-166, 2000.

[37] B. Smith, M. Ardakani, W. Yu, and F.R. Kschischang, "Design of Low-Density Parity-Check Codes with Optimized Complexity-Rate Tradeoff", manuscript.

[38] R.M. Tanner, "A recursive approach to low-complexity codes", *IEEE Transactions on Information Theory*, Vol. 27, No. 5, pp. 533-547, September 1981.

[39] J. Thorpe, K. Andrews, and S. Dolinar, "Methodologies for Designing LDPC Codes Using Protographs and Circulants", *Proc. IEEE Symp. on Inform. Theory*, p. 238, 2004.

[40] S. Wicker, and S. Kim, *Fundamentals of Codes, Graphs, and Iterative Decoding*, Kluwer Academic Publishers, 2003.

[41] W. Yu, M. Ardakani, B. Smith, and F.R. Kschischang, "Complexity-optimized LDPC Codes for Gallager Decoding Algorithm B", *Proceedings of IEEE International Symposium on Information Theory*, Adelaide, Australia, September 2005.

[42] The web page of W&W Communications Inc., http://www.wwcoms.com/