Design of Polar Code Lattices of Moderate Dimension

Ning Liu

Graduate School of Advanced Science and Technology
Japan Advanced Institute of Science and Technology
(Information Science)

March, 2022

Master's Thesis

Design of Polar Code Lattices of Moderate Dimension

1910238 Ning Liu

Graduate School of Advanced Science and Technology
Japan Advanced Institute of Science and Technology
(Information Science)

March, 2022

Abstract

In 2008, Arikan first proposed the concept of channel polarization, then channel polarization was explained in more detail, and a new encoding method was given. Polar codes are the only known channel coding method that can be strictly proven to "reach" channel capacity when the code length goes to infinity under successive cancellation (SC) decoding. Thus, polar codes have attracted the attention of academia and industry in the past decade.

Lattice codes are linear codes defined over Euclidean space and also have attracted attention in recent years. In wireless communications, lattice codes can be used to perform shaping technique efficiently to obtain about 1.53 dB shaping gain on the additive white Gaussian noise (AWGN) channel channel. In addition, lattices are integral to certain Gaussian network coding approaches, including compute-forward relaying and integer-forcing MIMO.

In this thesis, we propose a design of polar code lattices using Construction D of moderate dimension. Construction D forms lattices from binary codes and allows decoding lattices using binary code decoder. In the design, we use the dimension N and the target decoding error probability P_{trgt} as parameters. Furthermore, we use the explicit finite-length properties of the polar code to select the code rates of the Construction D component codes. Under SC decoding, over the (AWGN) channel, instead of using the approximation of the Bhattacharyya parameter, we use density evolution to select the information bit positions that allows obtaining the probability distribution for each position. Then, choose code rates for the component codes that satisfy the equal error probability rule. For polar codes with successive cancellation decoding, we propose a function ρ defined as the greatest code rate with error rate under P_{trgt}; ρ may be found by density evolution efficiently. Under successive cancellation list (SCL) decoding and ordered statistic decoding (OSD), since density evolution is not practical, function ρ can be obtained by Monte Carlo simulation.

Dimension $N = 128$ polar code lattices are given as design example. From the simulation results, under SC decoding with complexity $O(\mathrm{N} \log N)$, polar code lattice comes within 1 dB of the best-known BCH code lattice. Under SCL decoding with L = 128 and CRC-6, polar code lattice comes within 0.2 dB of the BCH code lattice. SCL decoding with list size L, complexity scales as $O(\mathrm{LN}\log N)$. For polar code lattice under OSD, there is 1 dB gap between BCH code lattice. OSD decoding has significantly higher complexity. The complexity of order-l OSD is proportional to $\sum_{i=0}^{l}\binom{K}{i}$.

Keywords: Channel polarization, binary polar code, successive cancellation decoding, polar code lattice, Construction D, AWGN channel

Acknowledgments

My deepest gratitude goes foremost to my supervisor, Professor Brian M. Kurkoski for his patient guidance, instructive suggestions and constant encouragements throughout my research. When I joined as a young and stupid research student, I took his lecture, which gave me an incredible journey to coding theory. I could never imagine that those complicated contents could be explained so simply and clearly by him. He is always responsible to any questions and gives as detailed explanations as possible. Brian is not only a knowledgeable professor from whom I really learned a lot but also a wonderful mentor in my research and personal life. Not a single line in this thesis would exist without him.

Last but not the least, my gratitude also extends to my beloved grandparents, Mingyuan Liu and Shuhua Yang, and my parents who have been assisting, supporting and caring for me all of my life.

Contents

List of Figures

List of Tables

Acronyms

AMGN	Additive Mod-2 Gaussian Noise
AWGN	Additive White Gaussian Noise
BEC	Binary erasure channel
BER	Bit Error Rate
BI-AWGN	Binary-input Additive White Gaussian Noise
BI-DMC	Binary-input Discrete Memoryless Channel
BPSK	Binary Phase Shift Keying
BSC	Binary symmetric channel
CRC	Cyclic Redundancy Check
IoT	Internet of Things
LLR	Log Likelihood Ratio
ML	Maximum-likelihood
OSD	Ordered Statistic Decoding
SC	Successive Cancellation

SCF	Successive Cancellation Flip
SCL	Successive Cancellation List
SCS	Successive Cancellation Stack
SNR	Signal-to-Noise Ratio
WER	Word Error Rate

Notations

E_b/N_0	Energy per bit to noise power spectral density ratio
N	Block-length
$\mathbb{R}^{\mathrm{N}}$	N-dimensional real field
$\mathbf{G}_N$	Polar code generator matrix
$\mathbf{G}_\Lambda$	Lattice generator matrix
$\mathbf{u}$	Input sequence
$\mathbf{x}$	Codeword
$\mathbf{y}$	Received sequence
$\mathbf{z}$	Channel noise
$\mathcal{F}$	Set of frozen bits
$\mathcal{I}$	Set of information bits
$\mathcal{P}(N, K, \mathcal{F})$	Binary polar code
u_1^K	Abbreviation of a row vector $u_1, u_1, u_2, \ldots, u_K$

Chapter 1

Introduction

1.1 Background and Motivation

In recent years, the number of wireless devices has increased dramatically and requires higher and higher data rates. Reliable low-latency communications are needed to handle the demands of real-time control applications, which require low-latency codes so that devices can respond rapidly. As a case in point, doctors need real-time feedback from medical devices during operations. Unmanned vehicles need to react quickly to road conditions, etc. For the Internet of Things (IoT), a new category of reliable low-latency communications for small amounts of data is desired.

Shannon pointed out in the noisy channel coding theorem that there exist codes that can reach the Shannon limit [1]. Polar code is a new type of channel coding based on the channel polarization theorem proposed by E. Arikan in 2008 [2], and is the first structured channel coding method that can be strictly proven to "reach" the symmetric capacity of a binary-input discrete memoryless channel (BI-DMC) using a low-complexity successive cancellation (SC) decoding when the code length goes to infinity. In the past decade, polar codes have attracted the attention of academia and industry, such that polar codes were chosen as a channel coding scheme of the 5th generation wireless systems (5G) standardization process of the 3rd generation partnership project (3GPP). Polar codes have been widely studied and applied in various fields with satisfying prospects for development.

Lattices are sphere bound-achieving codes. In communications applications, lattices can be used to perform shaping efficiently to obtain about 1.53 dB of shaping gain on the AWGN channel. In addition, lattices are integral to certain Gaussian network coding approaches, including compute-forward relaying [3] and integer-forcing MIMO [4]. In the classical point-to-point channel, compared with random Gaussian codes, lattices provide a low-complexity solution. In modulation for real communication channel, compared with binary coding schemes, the noise and transmission functions are based on the real field $\mathbb{R}^{\mathrm{N}}$ but not binary. This is the other reason why we choose lattices.

In 2019, Liu et al. proposed a new class of lattices constructed from polar codes [5]. They constructed polar lattices to approach the capacity of the power-constrained Gaussian channel, following the multilevel approach of Forney et al. The construction is based on a combination of channel polarization and source polarization by constructing capacity-achieving polar codes on each level. The component polar codes are shown to be naturally nested, thereby, fulfilling the requirement of the multilevel lattice construction. They proved that the polar lattices are AWGN-good and used block length $N=$ 1024 as an example. It is feasible to consider using channel capacity as a guide to design high dimension lattices, but this breaks down when the dimension is small to medium. Questions about the best way to design finite-dimensional polar code lattices are still an open problem.

Construction D produces lattices from two or more component linear binary codes and can be decoded through a multilevel decoding procedure [6] [7]. An important aspect is that Construction D decoding uses the decoder for the binary codes. Polar codes decoders are well-understood and SC decoder can be used to reduce the complexity. Construction D lattices allow designing using decoder error rates for component codes under the equal error probability rule [8]. In small or medium-dimension lattices, it is more reasonable to use decoder error rates in the design than using channel capacity. In addition, polar codes also have flexibility in rate selection. In this sense, we can easily get code rates of polar codes of different block lengths, which is important for good lattice design. The code rates of the component codes can be obtained by density evolution which is introduced in Section 3.2.1. Polar code lattices are attractive because lattices inherit the good properties of component polar codes.

Very long block-length ($N>$ 1000) and short block-length ($N\leq$ 24) lattices are well-studied. However, far too little attention has been paid to short block-length ($N= 64, 128, \dots, 512$) lattices not only for polar code lattices but also for any lattices. It is becoming increasingly difficult to ignore the performance of short bolck length lattices.

The originality of this work is to design short block-length lattices, that is block-length $N = 64, 128, \dots, 512$, based on binary polar codes using construction D. Because binary codes are already well understood, binary polar codes have good coding gain. Moreover, codes based on lattices have the potential to achieve reliable communications with high bandwidth efficiency without sacrificing decoding complexity. Compared with conventional QAM, lattices have higher shaping gain, and therefore are more power efficient.

1.2 Contributions

In this work, we contribute a design technique for polar code lattices of moderate dimensions and use block length $N= 128$ as an example. In the design, we use Construction D to form lattices from nested binary polar codes. Furthermore, we have solved the challenge of selecting rates of the component codes that give the best lattice properties. We use the explicit finite-length code properties in the design rather than using the capacity. For a given block length N, we define a function ρ as the code rate such that its decoder achieves a given target error rate of the channel noise σ^2. It allows us to design polar code lattices systematically and efficiently. Under SC decoding, the ρ function can be found efficiently by using density evolution. The function ρ, based on finite-length code properties, has an S-shape which is characteristic of a channel capacity curve. Then, polar lattices can be designed by these properties of binary polar codes.

We published the paper "Design of polar code lattices of finite dimension" at the 2021 IEEE International Symposium on Information Theory. This thesis gives some more detailed explanations in that paper.

1.3 Thesis Organization

The remaining part of this thesis proceeds as follows:

- In Chapter 2, we provide the system model and defined some symbols we used in this thesis. Then we give a introduction of AWGN channel and the SNR definition. Additionally, we give performance measurements that we use in Sec. 5.4 to explain how we evaluate the performance of a code.

- In Chapter 3, we start from explaining the idea of channel polarization. Then we going to introduce binary polar codes including the construction, encoding method and a series of decoders. Density evolution is introduced in Sec. 3.2.1, and the distribution curves for all positions for the $N = 8$ polar code is given as an example of density evolution. To make the explanation of decoders clearly, we give an example of SC decoding and SC decoding tree in Sec. 3.3.1. At the end of the chapter, we provide an OSD example.

- In Chapter 4, we first introduce lattices and then propose the Construction D lattice, which is the main contribution to this work. In Sec.4.2.2, the introduction of encoding and decoding Construction D lattices is given. Then, the multi-level construction is also given to explain how we construct lattices from nested binary codes. At the end of this chapter, we introduce some Construction D lattices design rules that are relevant to our design method.

- In Chapter 5, we introduce equal error probability rule and the ρ function that we use in the design of polar code lattices. Under SC decoding, the ρ function is found by density evolution efficiently. Under SCL decoding and OSD, the ρ function can be obtained by Monte Carlo simulation. Then we give a design example to introduce our design methods more detail. Finally, the simulation results for polar code lattices are shown in Sec.5.4.

- In Chapter 6, we summarize our work and describe future works. In the future work section, we give some ideas which may improve the performance of the code.

Chapter 2

Preliminaries

2.1 System Model

Fig. 2.1 shows the model of a communication system. It shows the process of transmitting data through a noisy channel. Clearly, it consists of five parts: an information source, an encoder, a channel, a decoder, and an information sink.

In this work, we mainly talk about binary polar codes consisting of K information bits and $N - K$ frozen bits and will be introduced in Sec. 3.1. For a given block length N and dimension K ($K \leq N$), an information sequence $\mathbf{u}$ is defined as:

$$\mathbf{u} = (u_1, u_2, u_3, \ldots, u_K) = u_1^K.$$

The code rate R can be obtained by:

$$R = \frac{K}{N}.$$

Then the polar coder encoder produces a codeword $\mathbf{x}$:

$$\mathbf{x} = (x_1, x_2, x_3, \ldots, x_N) = x_1^N$$

as the input of a noisy channel. After $\mathbf{x}$ is transmitted through a channel, a sequence $\mathbf{y}$ can be defined as the codeword $\mathbf{x}$ plus the channel noise $\mathbf{z}$:

$$\mathbf{y} = \mathbf{x} + \mathbf{z}.$$

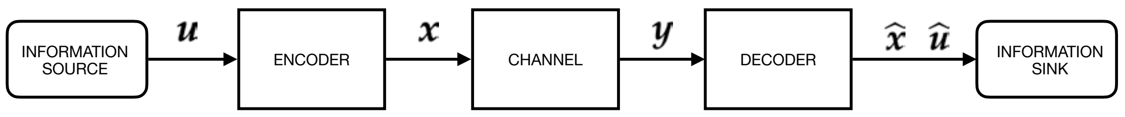


Figure 2.1: Model of a communication system with encoder, channel and decoder.

The decoder receives this sequence $\mathbf{y}$, and then gives an estimation of the information source as the output $\widehat{\mathbf{u}}$, or an estimate of the transmitted codeword $\widehat{\mathbf{x}}$:

$$\widehat{\mathbf{u}} = (\widehat{u}_1, \widehat{u}_2, \widehat{u}_3, \ldots, \widehat{u}_K) = \widehat{u}_1^K$$
$$\widehat{\mathbf{x}} = (\widehat{x}_1, \widehat{x}_2, \widehat{x}_3, \ldots, \widehat{x}_K) = \widehat{x}_1^N$$

2.2 Additive White Gaussian Noise Channel

Additive White Gaussian Noise (AWGN) channel is a basic noisy channel model that is often used to mimic the effect of many random processes that occur in nature. The model does not take into account fading, frequency selectivity, etc. Without considering other phenomena, it produces simple and tractable mathematical models that are useful for gaining insight into the fundamental behavior of the system.

The AWGN channel is represented by a sequence of outputs $\mathbf{y}$. As we mentioned in Section 2.1, $\mathbf{y}$ is the sum of the channel input $\mathbf{x}$ and noise $\mathbf{z}$ where $\mathbf{z}$ is independent and identically distributed (i.i.d.) and drawn from a zero-mean normal distribution with a variance of $\sigma^2 = \frac{N_0}{2}$ where N_0 is the noise power. The distribution of $\mathbf{z}$ can be represented as $\mathcal{N}(0, \sigma^2)$.

For each $x \in \mathcal{X}$, the probability of transmitting symbol $\mathbf{x}$ is $p_{\mathrm{X}}(x)$. E_s refers the average transmit power that is given by:

$$E_s = \sum_{x \in \mathcal{X}} p_{\mathrm{X}}(x) x^2 \tag{2.1}$$

E_b/N_0 is a normalized signal-to-noise ratio (SNR) measure, also known as the "SNR per bit" in digital communication or data transmission. E_b is defined as signal energy per bit and can be calculated as:

$$E_b = \frac{E_s}{R}$$

where R is the code rate. It is especially useful when comparing the bit error rate (BER) performance of different digital modulation schemes without taking bandwidth into account.

For the Gaussian channel, the average input power constraint P on N inputs x_1^N is given as:

$$\frac{1}{N}\sum_{i=1}^{N} x_i^2 \leq P. \tag{2.2}$$

The input x_i is assumed to be zero mean, so $\mathbf{y}$ is also zero mean. As we mentioned before, the variance of z_i is σ^2. The SNR can be represented as:

$$\text{SNR} = \frac{P}{\sigma^2} \tag{2.3}$$

In this thesis, SNR is defined as

$$\text{SNR} = 1/\sigma^2.$$

Then:

$$\text{SNR dB} = 10\log_{10}\text{SNR} = 10\log_{10}\frac{1}{\sigma^2}. \tag{2.4}$$

Due to the Gaussian distribution of noise $\mathbf{z}$, the probability of x and y is given by:

$$p_{\text{Y}|\text{X}}(y \mid x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-(y-x)^2/2\sigma^2} \tag{2.5}$$

For a binary-input AWGN (BI-AWGN) channel or a binary phase-shift keying (BPSK) channel. It has input $x \in \{+1, -1\}$ with probability distribution $p_X(x) = \left[\frac{1}{2}, \frac{1}{2}\right]$. The average transmit power E_s obtained by Eqn.2.1 is 1. The conditional channel output distribution is:

$$p_{\text{Y}|\text{X}}(y \mid -1) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-(y+1)^2/2\sigma^2}$$
$$p_{\text{Y}|\text{X}}(y \mid +1) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-(y-1)^2/2\sigma^2}$$

that can be used to calculate LLR in Section 3.3.

2.3 Performance Measurements

Word-error rate (WER) and bit-error rate (BER) are two measures that can help assess a decoder's performance.

For a transmitted codeword $\mathbf{x}$, assume that the decoder can give an estimation as $\widehat{\mathbf{x}}$. WER is the probability of word error which means $\widehat{\mathbf{x}} \neq \mathbf{x}$. BER is the probability of bit error which means $\widehat{x}_i \neq x_i$.

In a one-time decoding process, two cases of the decoder are shown as follows:

- If $\widehat{\mathbf{x}} = \mathbf{x}$, that means the decoder successfully decoded the input sequence. Here, both the BER and WER are 0; non-error occurred.
- If $\widehat{\mathbf{x}} \neq \mathbf{x}$, , that means the decoder failed to decode. In this case, the WER is 1. The range of BER is $0 < \text{BER} \leq 1$.

Example of BER and WER

Transmitted a 4-bit sequence $\mathbf{x} = \begin{bmatrix} 1 & 1 & 0 & 0 \end{bmatrix}$ once.

- If the decoder gives an estimation as $\widehat{\mathbf{x}} = \begin{bmatrix} 1 & 1 & 0 & 0 \end{bmatrix}$, then it means no error occurs, BER$= 0$ and WER$= 0$;
- If the decoder gives an estimation as $\widehat{\mathbf{x}} = \begin{bmatrix} 1 & 1 & 1 & 1 \end{bmatrix}$, then it is clear $\widehat{x}_3 \neq x_3$ and $\widehat{x}_4 \neq x_4$ thus, $\widehat{\mathbf{x}} \neq \mathbf{x}$. BER$= 2/4$ and WER$= 1$.

In this work, we obtain WER and BER by Monte Carlo simulations. We set a term *count* which refers to the total number of codewords in the simulation. Then WER and BER can be defined by:

$$\text{BER} = \frac{\text{Number of Bit Errors}}{\text{Total Number of Bits}} = \frac{\text{Number of Bit Errors}}{count \times N}$$

$$\text{WER} = \frac{\text{Number of Word Errors}}{\text{Total Number of Words}} = \frac{\text{Number of Word Errors}}{count}$$

It is clear that under the same SNR, the code with lower WER (or BER) gives better performance. In this work, we mainly use WER to evaluate the performance of a code. The simulation result will be given in Sec. 5.4.

Chapter 3

Binary Polar Codes

In this chapter, we introduce polar codes including encoding and decoding schemes. In the encoding part, we mainly explain the structure of polar code and methods to select information bits and frozen bits. In the decoding part, we introduce several decoding algorithms including successive cancellation (SC) decoding, successive cancellation list (SCL) decoding and ordered statistic decoding (OSD).

3.1 Introduction to Polar Codes

In 2008, Arikan first proposed the concept of channel polarization [2]. Then, channel polarization was explained in more detail, and a new encoding method was given.

Due to the channel polarization, a polar code of block length N is divided to two types of bits, information bits and frozen bits. For $n \geq 1$ and a $\mathcal{P}(N, K, \mathcal{F})$ binary polar code, $N = 2^n$ is the code length in bits, K is the number of elements in the information bit indices $\mathcal{I}$ and $\mathcal{F}$ is an index vector of frozen bits. Clearly, the length of $\mathcal{F}$ is $N - K$, the code rate is $R = K/N$. Different from other linear block codes, the input $\mathbf{u}$ has N elements that encompasses both information bits and frozen bits. Frozen bits are set to a fixed value. For the information set $\mathcal{I}$ and frozen set $\mathcal{F}$, it has:

$$\begin{gathered} \mathcal{F} \subset \{1, 2, \dots, N\}, \quad \mathcal{I} \subset \{1, 2, \dots, N\} \quad \text{and} \\ \mathcal{I} \cup \mathcal{F} = \{1, 2, \dots, N\}, \quad \mathcal{I} \cap \mathcal{F} = \emptyset. \end{gathered} \tag{3.1}$$

Fig. 3.1 shows the effective channel of polar codes. Let $W : \mathcal{X} \to \mathcal{Y}$ be a BI-DMC with transition probabilities: $W(y \mid x)$ where the input alphabet is $x \in \mathcal{X}$ and the output alphabet is $y \in \mathcal{Y}$.

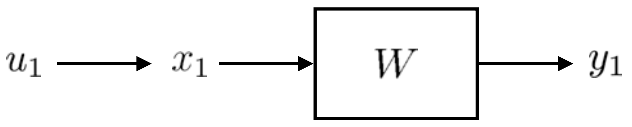


Figure 3.1: Effective channel.

For a given BI-DMC W, the symmetric capacity $I(W)$ is the maximum rate for reliable transmission with uniform input distribution through the channel W that can be defined as:

$$I(W) \triangleq \sum_{y \in \mathcal{Y}} \sum_{x \in \mathcal{X}} \frac{1}{2} W(y \mid x) \log \frac{W(y \mid x)}{\frac{1}{2} W(y \mid 0) + \frac{1}{2} W(y \mid 1)}$$

Additionally, for channels with input-output symmetry, the capacity is given by

$$C(W) \triangleq I(X;Y)$$

with the input X is uniformly distributed over $\{0,1\}$. Using base-2 logarithms, $0 \leq C(W) \leq 1$.

When W is used only once to transmit a bit $x \in \{0,1\}$, the Bhattacharyya parameter gives an upper bound on the probability of ML decision error:

$$Z(W) \triangleq \sum_{y \in \mathcal{Y}} \sqrt{W(y \mid 0) W(y \mid 1)} \tag{3.2}$$

From Eqn. 3.2, it is clear that $Z(W)$ takes value in $[0,1]$ where 0 means totally reliable and 1 means totally unreliable, so that $Z(W)$ can be used as measures of channel reliability.

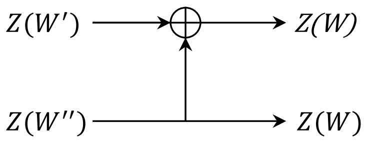


Figure 3.2: Bhattacharyya parameter of W_2.

In Arikan's paper [2], as Fig. 3.2 shows, for two synthetic channels, the Bhattacharyya parameter of the "upper channel" is denoted as $Z(W')$ and for the "lower

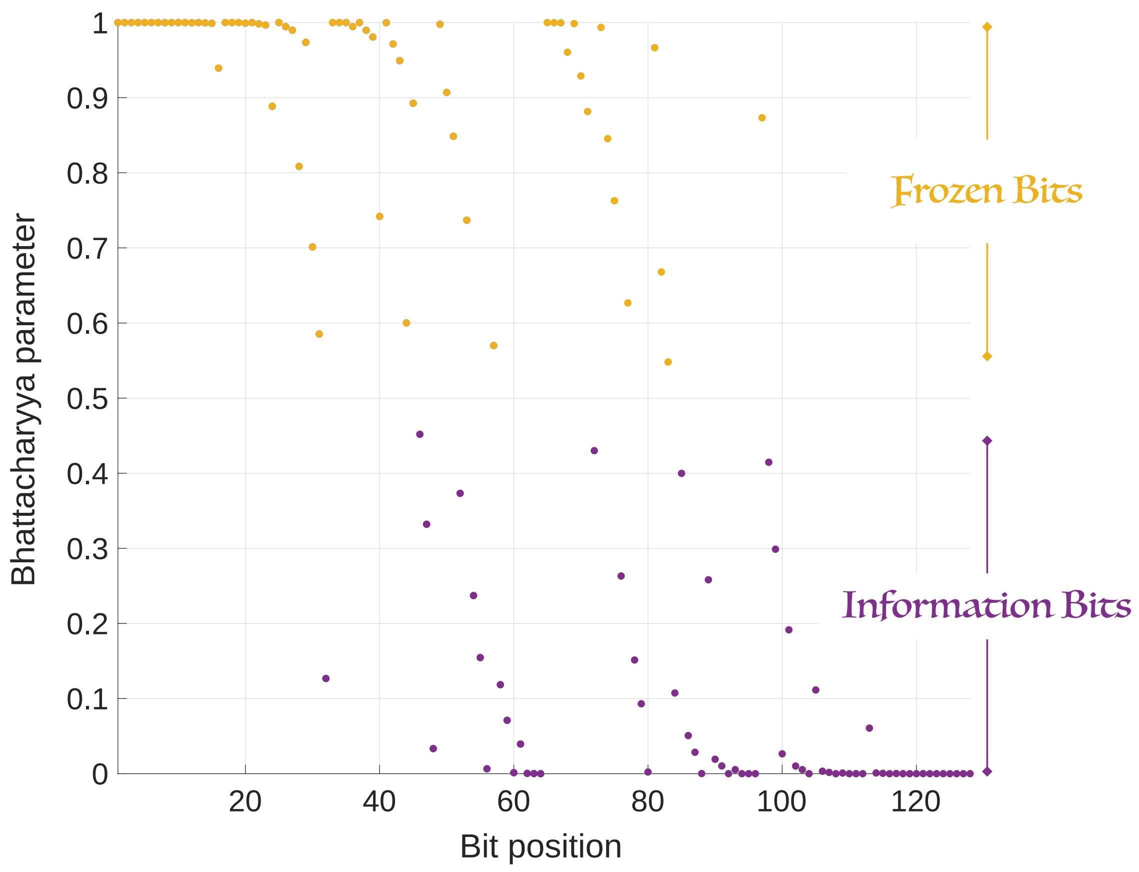


Figure 3.3: Information bits selection of N=128 polar code over BEC $\epsilon = 0.5$.

channel" is $Z\left(W''\right)$. For some set of binary-input channels, it has:

$$\begin{aligned} Z\left(W''\right) &= Z(W)^2 \\ Z\left(W'\right) &\leq 2Z(W) - Z(W)^2. \end{aligned} \tag{3.3}$$

equality holds if and only if W is a binary erasure channel (BEC).

For a BEC, given the code length N=128 and erasure probability $\epsilon = 0.5$, the the Bhattacharyya parameter for each position can be calculated. As we mentioned in Eqn. 3.2, we can we can design a rate 1/2 polar code by freezing 64 positions with greater $Z(W)$ represented by yellow points as Fig. 3.3 shows.

3.1.1 Channel Polarization

Channel polarization is divided into two phases: channel combining and channel splitting.

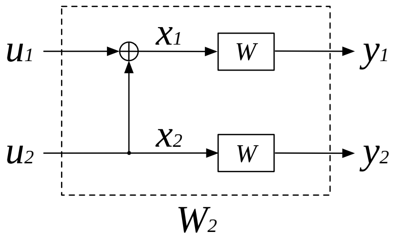


Figure 3.4: W_2 channel model.

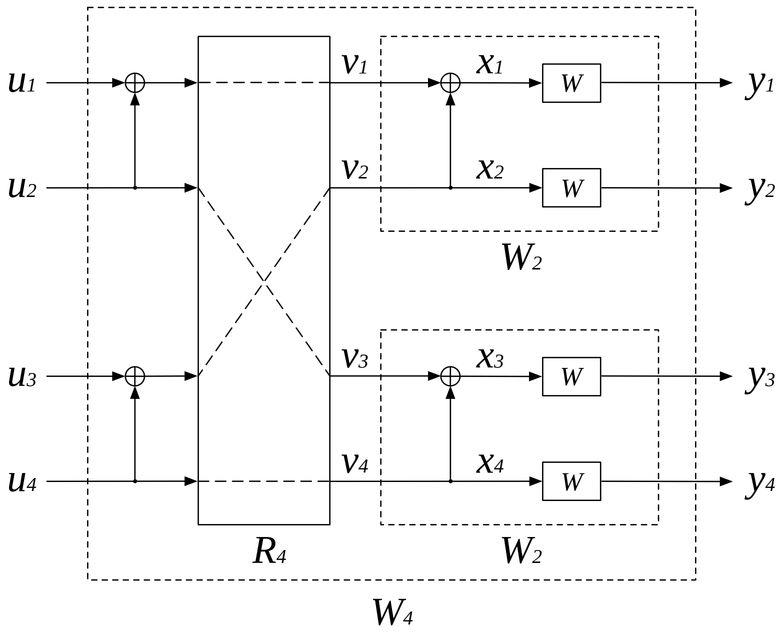


Figure 3.5: W_4 channel model.

Channel Combining

Fig. 3.4 shows the W_2 channel model which have inputs u_1, u_2 and outputs y_1, y_2. It is a combination of two copies of W channels. The input of the first channel is $u_1 \oplus u_2$ and the input of the second channel is u_2. The transition probabilities of channel $W_2 : \mathcal{X}^2 \to \mathcal{Y}^2$ can be defined as:

$$W_2\left(y_1, y_2 \mid u_1, u_2\right) = W\left(y_1 \mid u_1 \oplus u_2\right) W\left(y_2 \mid u_2\right).$$

As Fig. 3.5 shows, W_4 channel can be obtained by two copies of W_2 channel with inputs $u_1^4 = (u_1, u_2, u_3, u_4)$ and outputs $y_1^4 = (y_1, y_2, y_3, y_4)$. R_4 reverses input combinations $u_1 \oplus u_2, u_2, u_3 \oplus u_4, u_4$ of W_4 channel to input vectors $v_1^4 = (u_1 \oplus u_2, u_3 \oplus u_4, u_2, u_4)$ of W_2 channels. The transition probabilities of channel $W_4 : \mathcal{X}^4 \to \mathcal{Y}^4$

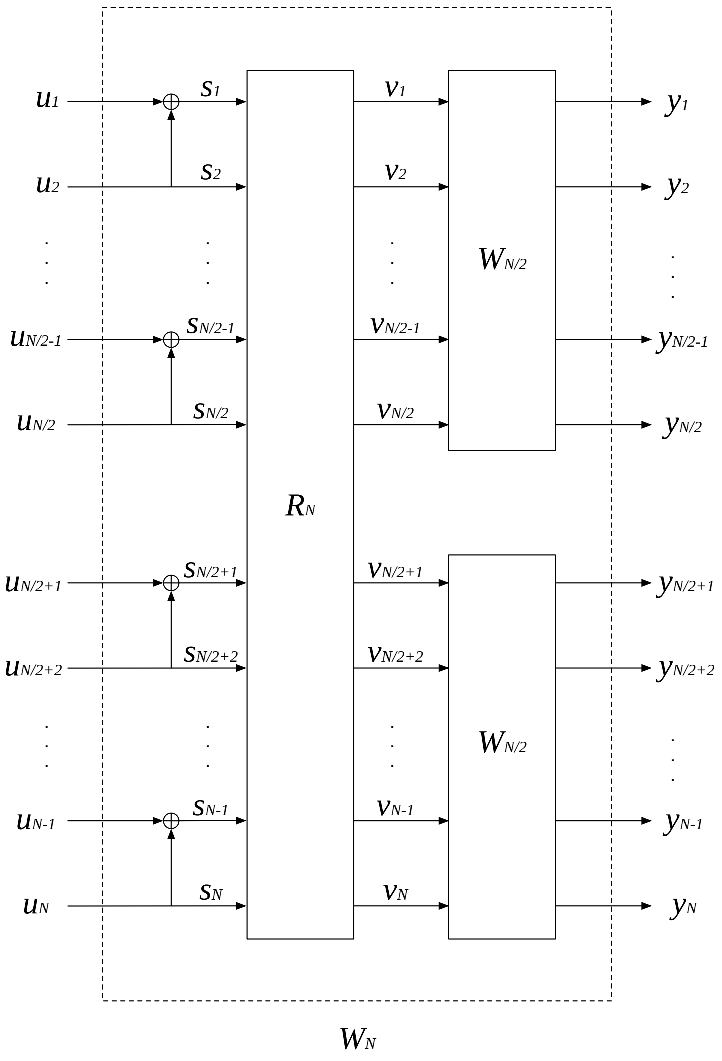


Figure 3.6: W_n channel model.

are:

$$W_4\left(y_1^4 \mid u_1^4\right) = W_2\left(y_1^2 \mid u_1 \oplus u_2, u_3 \oplus u_4\right) W_2\left(y_3^4 \mid u_2, u_4\right).$$

Similarly, the W_N channel can be obtained by two copies of $W_{2/N}$ channel, shown in Fig. 3.6. $W_N\left(y_1^N \mid x_1^N\right) = \prod_{i=1}^N W\left(y_i \mid x_i\right)$ denotes the channel corresponding to N uses of the channel W where $N = 2^n, n \geq 0$, thus, $W_N : \mathcal{X}^N \rightarrow \mathcal{Y}^N$. The construction of polar codes combines BI-DMCs recursively to form a vector channel W_N, which is called channel combining.

Channel Splitting

After we obtained the vector channel W_N out of a series of BI-DMCs, the next step of channel polarization is called channel splitting. Channel splitting is to split the vector channel W_N back into a set of N binary-input coordinate channels $W_N^{(i)}: \mathcal{X} \rightarrow \mathcal{Y}^N \times \mathcal{X}^{i-1}, 1 \leq i \leq N$. The transition probability of the $W_N^{(i)}$ is defined as:

$$W_N^{(i)}\left(y_1^N, u_1^{i-1} \mid u_i\right) \triangleq \sum_{u_{i+1}^N \in \mathcal{X}^{N-i}} \frac{1}{2^{N-1}} W_N\left(y_1^N \mid u_1^N\right)$$

where $\left(y_1^N, u_1^{i-1} \mid u_i\right)$ denotes that y_1^N, u_1^{i-1} are inputs of $W_N^{(i)}$ and u_i is the output of $W_N^{(i)}$. Then, for any $n \geq 0, N = 2^n, 1 \leq i \leq N$, the transition probabilities of odd-order and even-order splitting sub-channels can be obtained, respectively, by two recursive equations shown as [2, Proposition 3]:

$$\begin{aligned} & W_{2N}^{(2i-1)}\left(y_1^{2N}, u_1^{2i-2} \mid u_{2i-1}\right) \\ & = \sum_{u_{2i}} \frac{1}{2} W_N^{(i)}\left(y_1^N, u_{1,o}^{2i-2} \oplus u_{1,e}^{2i-2} \mid u_{2i-1} \oplus u_{2i}\right) \cdot W_N^{(i)}\left(y_{N+1}^{2N}, u_{1,e}^{2i-2} \mid u_{2i}\right) \end{aligned} \tag{3.4}$$

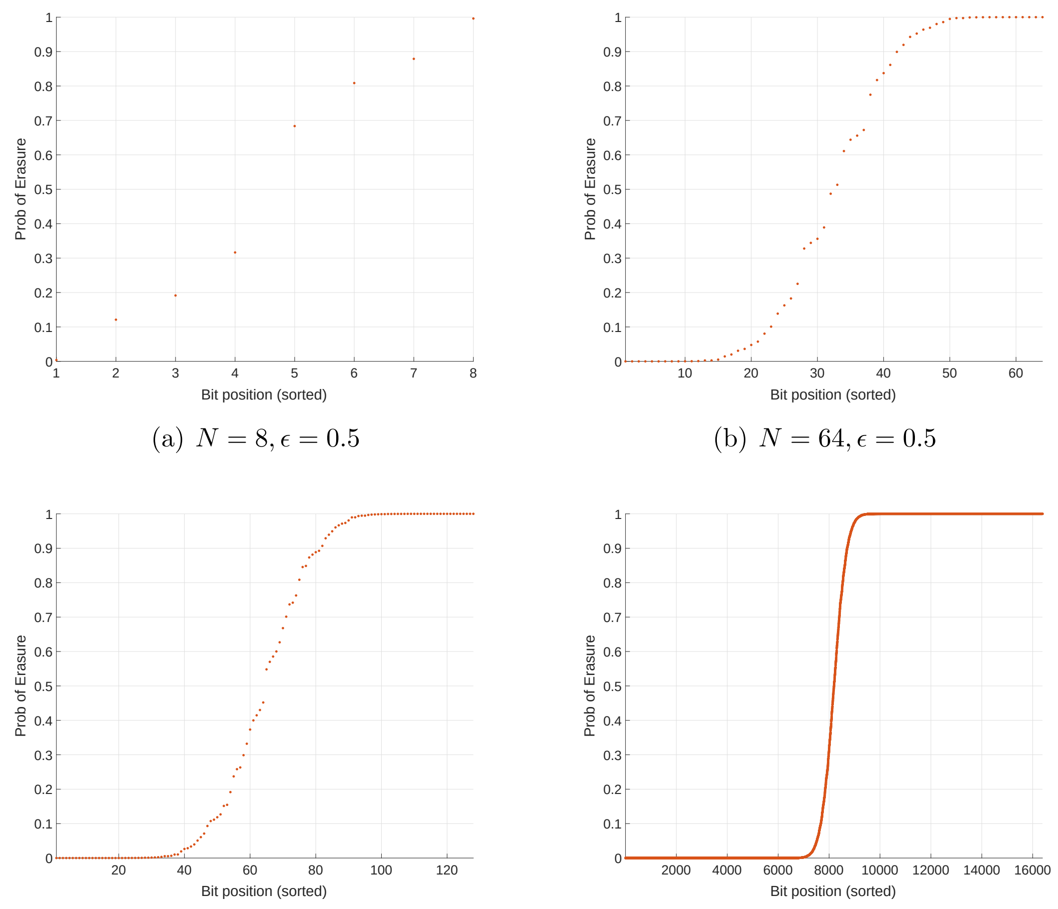
and

$$\begin{aligned} & W_{2N}^{(2i)}\left(y_1^{2N}, u_1^{2i-1} \mid u_{2i}\right) \\ & = \frac{1}{2} W_N^{(i)}\left(y_1^N, u_{1,o}^{2i-2} \oplus u_{1,e}^{2i-2} \mid u_{2i-1} \oplus u_{2i}\right) \cdot W_N^{(i)}\left(y_{N+1}^{2N}, u_{1,e}^{2i-2} \mid u_{2i}\right). \end{aligned} \tag{3.5}$$

By combining and spliting to BI-DMCs, the capacity of each sub-channel tends to be polarized. As the code length N increases, the sub-channel tends to be more polarized, as shown in Fig. 3.7. The bit positions have been sorted according to increasing probability of erasure.

The point of channel polarization can be explained more detail. In Fig. 3.4, let W be BEC with the erasure probability ϵ; that is, for each W: the probability that X is successfully transmitted is $1-\epsilon$, and when X failed to transmit, the probability is ϵ. By Eqn. 3.4 and 3.5, we obtain the two W channels as $W_2^{(1)}$ and $W_2^{(2)}$:

$$\begin{aligned} W_2^{(1)} &= W\left(y_1, y_2 \mid u_1\right) \\ W_2^{(2)} &= W\left(y_1, y_2, u_1 \mid u_2\right) \end{aligned} \tag{3.6}$$



(a) $N = 8, \epsilon = 0.5$ (b) $N = 64, \epsilon = 0.5$

(c) $N = 512, \epsilon = 0.5$ (d) $N = 16384, \epsilon = 0.5$

Figure 3.7: Probability of erasure for each bit position for $N = 8, 64, 512, 16384$.

From Eqn. 3.6, it is clear that y_1, y_2 should be used to decode u_1. The decoding table can be shown as:

y_1	y_2	$\widehat{u}_1$
$u_1 \oplus u_2$	u_2	u_1
?	u_2	?
$u_1 \oplus u_2$	?	?
?	?	?

(3.7)

where $y_1 = u_1 \oplus u_2, y_2 = u_2$. From the table, only if both y_1 and y_2 are known, u_1 can be decoded successfully. Those probabilities are summarized below:

$$W(y_1, y_2 \mid u_1) = \begin{cases} (1-\epsilon)^2 & \text{with} \quad (u_1 \oplus u_2, u_2) \\ \epsilon(1-\epsilon) & \text{with} \quad (\ ?\ ,\ u_2\ ) \\ (1-\epsilon)\epsilon & \text{with} \quad (u_1 \oplus u_2,\ ?\ ) \\ \epsilon^2 & \text{with} \quad (\ ?\ ,\ ?\ ) \end{cases} \tag{3.8}$$

The probability that u_1 is correctly decoded is: $(1-\epsilon)(1-\epsilon) = (1-\epsilon)^2$.

For decoding u_2, y_1, y_2, u_1 should be used. Since u_1 is always known, the decoding table of u_2 is:

$$\begin{array}{ccc|c} y_1 & y_2 & u_1 & \widehat{u}_2 \\ \hline u_1 \oplus u_2 & u_2 & u_1 & u_2 \\ ? & u_2 & u_1 & u_2 \\ u_1 \oplus u_2 & ? & u_1 & u_2 \\ ? & ? & u_1 & ? \end{array} \tag{3.9}$$

Those probabilities are summarized below:

$$W(y_1, y_2 u_1 \mid u_2) = \begin{cases} (1-\epsilon)^2 & \text{with} \quad (u_1 \oplus u_2, u_2, u_1) \\ \epsilon(1-\epsilon) & \text{with} \quad (\ ?\ , u_2, u_1) \\ (1-\epsilon)\epsilon & \text{with} \quad (u_1 \oplus u_2,\ ?, u_1) \\ \epsilon^2 & \text{with} \quad (\ ?\ , ?,\ u_1) \end{cases} \tag{3.10}$$

The probability that u_2 is correctly decoded is: $1-\epsilon^2$.

Since the range of ϵ is always $[0,1]$ so that $(1-\epsilon)^2 \leq 1-\epsilon^2$. The smaller the value of ϵ, the more significant the difference between $W_2^{(1)}$ and $W_2^{(2)}$.

Channel polarization gives a way to design a polar code. The basic idea is that according to the different reliability of the splitting sub-channels, the more reliable K channels are used to transmit useful information and the less reliable $N-K$ channels are frozen to a fixed value.

3.2 Encoding

Polar codes transmit K information bits in a block length N code. Polar code encoding will polarize the channel into reliable and unreliable sub-channels. So that K information bits will be transmitted on the most reliable K sub-channels. The remaining $N-K$ channels are unreliable are usually set to a fixed value.

Given $N = 2^n, n \geq 1$ and information bits indices $\mathcal{I} \subseteq \{1, 2, 3, \ldots, N\}$, a polar code is the linear block code with codewords generated by $\mathcal{I}$ that can be obtained by a generator matrix:

$$\mathbf{G}_N = \mathbf{B}_N \cdot \mathbf{F}^{\otimes n},$$

where $\mathbf{B}_N$ is the bit-reversal permutation matrix and $\mathbf{F}^{\otimes n}$ denotes the n-th K Kronecker power of $\mathbf{F}$. Kronecker powers are defined by $A^{\otimes n} \triangleq A \otimes A^{\otimes n-1} = A^{\otimes n-1} \otimes A$. For $n = 1, 2, 3$ that is $N = 2, 4$ and 8, $\mathbf{F}$ and its Kronecker powers are given as:

$$\mathbf{F} = \mathbf{F}^{\otimes 1} = \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}, \quad \mathbf{F}^{\otimes 2} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 1 \end{bmatrix},$$

$$\mathbf{F}^{\otimes 3} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix}.$$

For the Kronecker product, A $\otimes$ B differs from B $\otimes$ A. By using reverse shuffle matrices, the order of rows and columns in A $\otimes$ B can be rearranged into B $\otimes$ A. In particular, let A be a 2×2 matrix and B be an $(N/2) \times (N/2)$ matrix, then

$$\begin{aligned} \mathrm{B} \otimes \mathrm{A} &= S^T_{2,N/2}(\mathrm{A} \otimes \mathrm{B})S_{2,N/2} \\ &= \mathbf{R}_N(\mathrm{A} \otimes \mathrm{B})\mathbf{R}^T_N. \end{aligned} \tag{3.11}$$

From Eqn. 3.11, it is clear that $\mathbf{R}_N = S^T_{2,N/2}$. $\mathbf{R}_N$ denotes the $N \times N$ reverse shuffle permutation matrix defined by:

$$(s_1, s_2, \ldots, s_N)\,\mathbf{R}_N = (s_1, s_3, \ldots, s_{N-1}, s_2, s_4, \ldots, s_N).$$

For N= $2, 4$ and 8, $\mathbf{R}_N$ is shown as:

$$\mathbf{R}_2 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \quad \mathbf{R}_4 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix},$$

$$\mathbf{R}_8 = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}.$$

In the permutation matrix, only one 1 in each column and one 1 in each row, all the other elements are 0. The permutation matrix for bit-reversal can be found recursively by:

$$\mathbf{B}_N = \mathbf{R}_N \cdot \left(\mathbf{I}_2 \otimes \mathbf{B}_{N/2}\right) = \mathbf{R}_N \begin{bmatrix} \mathbf{B}_{N/2} & 0 \\ 0 & \mathbf{B}_{N/2} \end{bmatrix}, \tag{3.12}$$

where $\mathbf{B}_2 = \mathbf{I}_2$ and $\mathbf{R}_N$ is the reverse shuffle, that is $\mathbf{R}_N = \mathbf{B}_N^{\mathrm{t}}$.

For N= $2, 4, 8$, bit-reversal permutations $\mathbf{B}_N$ and reverse shuffle $\mathbf{R}_N$ is shown as:

$$\mathbf{B}_2 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix},$$

$$\mathbf{B}_4 = \mathbf{R}_4 \cdot (\mathbf{I}_2 \otimes \mathbf{B}_2) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix},$$

$$\mathbf{B}_8 = \mathbf{R}_8 \cdot (\mathbf{I}_2 \otimes \mathbf{B}_4) = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}.$$

For the code length N, given an input sequence $\mathbf{u} = (u_1, u_2, u_3, \ldots, u_N)$, the codeword $\mathbf{x}$ can be generated by:

$$\mathbf{x} = \mathbf{u} \cdot \mathbf{G}_N,$$

where $\mathbf{G}_N$ denotes the generator matrix.

- **Encoding Example**

  To design a $\mathcal{P}(8, 4, \mathcal{F})$ polar code, $\mathcal{F}= \{5, 3, 2, 1\}$ and $\mathcal{I} = \mathcal{F}^c = \{8, 7, 6, 4\}$ (unsorted). The information set of the input sequence is $\mathbf{u}_{\mathcal{I}} = (u_8, u_7, u_6, u_4)$. The generator matrix of $\mathcal{P}(8, 4, \mathcal{F})$ is:

$$\mathbf{G}_{(8,4)} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \end{bmatrix} \begin{array}{l} - g8 \\ - g7 \\ - g6 \\ - g4 \end{array}$$

  The original generator matrix of $N = 8$ polar code is:

$$\mathbf{G}_8 = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix}.$$

The codeword $\mathbf{x}$ can be obtained by:

$$\mathbf{x} = u_1^N \cdot \mathbf{G}_N \text{ or } \mathbf{u}_{\mathcal{I}} \cdot \mathbf{G}_{(8,4)}$$

The encoding process can be efficiently realized with $\mathcal{O}(N \log N)$ exclusive or (XOR) complexity [2].

3.2.1 Information Bit Selection

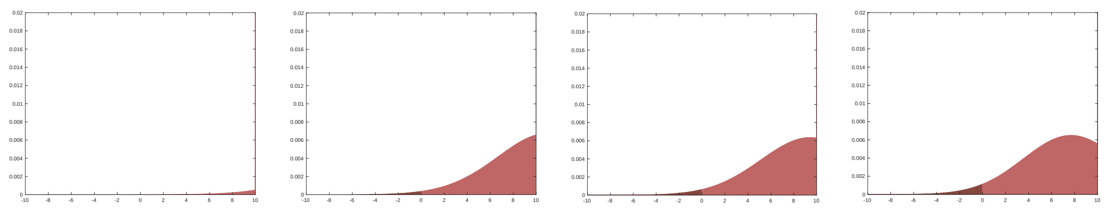
As we mentioned already, to construct polar codes, selecting a set of the most reliable bit positions is necessary. One possibility for the construction is to measure the reliability of a channel by the Bhattacharyya parameter that we introduced in Section 3.1. However, as was expained at Eqn. 3.3, for the local transformation of rate and reliability, equality holds iff W is a BEC. For any other channel, to use the Bhattacharyya parameter measure the reliability of a channel, we have to give an approximation of $Z(W)$ such as:

- For a BSC with error probability p, $Z(W) = 2\sqrt{(1-p)p}$
- For an AWGN with noise variance σ^2, $Z(W) = e^{-\frac{1}{2\sigma^2}}$

By using the approximation, sometimes, it does not give a precise calculation. Design of polar codes using density evolution was first described by Mori and Tanaka [9] in 2009. Instead of using an approximation, density evolution gives probability distribution for each bit position under SC decoding. These distributions can be used to obtain decoder error rates. Then, positions with lower error rates can be selected as information bits.

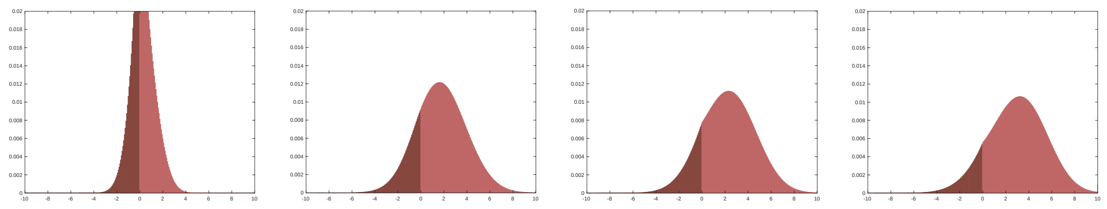
For the general decoder, density evolution is a recursive distribution for the distribution of the messages. Density evolution provides a probability distribution for each bit position; this method can achieve good performance. In this work, since we design short block-length code, it is reasonable to use density evolution.

Assume that an all-zeros coderword is transmitted over a symmetric channel. Let the probability density function of the memoryless channel LLR message be $a_1^{(1)}(x)$.



(a) 8th channel (b) 7th channel (c) 6th channel (d) 4th channel

Figure 3.8: Good channels.



(a) 1st channel (b) 2nd channel (c) 3rd channel (d) 5th channel

Figure 3.9: Bad channels.

The densities may be calculated recursively using the polar code structure by:

$$\begin{aligned} a_{2N}^{(2j)}(x) &= \left(a_N^{(j)} \star a_N^{(j)}\right)(x) \\ a_{2N}^{(2j-1)}(x) &= \left(a_N^{(j)} \boxast a_N^{(j)}\right)(x) \end{aligned} \tag{3.13}$$

for $j = 1, 2, \ldots, N$ where $\star$ is standard convolution for the variable node and $\boxast$ is the specific check node convolution operation. For a block length N polar code, given positions 1 to $j-1$ are correct, the probability of error in position j can be calculated by $a_N^{(j)}(x) = \Pr\left(L_j = x \mid \widehat{u}_1^{j-1} = 0\right)$. $a_N^{(j)}(x)$ is used to make a hard decision in position j. Then, the probability of error in position j can be obtained by:

$$p_j = \int_{-\infty}^{0} a_N^{(j)}(x)dx. \tag{3.14}$$

Using the values of p_j, K positions with smallest values can be selected to be information bits $\mathcal{I}$ and the remaining $N-K$ bits are set to frozen bits.

Fig. 3.8 and 3.9 show an example of good channels and bad channels of code length $N = 8$ with BPSK modulation under AWGN channel with noise variance $\sigma = 0.6$. By observing the shaded part, the error probability of each channel can be also observed.

3.3 Decoding

Polar codes provably achieve the symmetric capacity of binary input discrete memoryless channels with SC decoding. However, SC decoding is a greedy algorithm. For each layer of the code tree, only the locally optimal path is searched for the next layer. At finite code length, due to incomplete channel polarization, there are still some information bits that cannot be decoded correctly. In particular, polar codes of short block length do not give an efficient performance. To address this issue, some improved decoding algorithms have been proposed, such as successive cancellation list (SCL) decoding [10], successive cancellation flip (SCF) decoding [11], successive cancellation stack (SCS) decoding [12], etc. As well, polar codes can be improved by concatenating cyclic redundancy check (CRC) bits. Using ordered statistic decoding (OSD) [13] to decode polar codes is described in [14], which can also improve the performance of polar codes.

3.3.1 Successive Cancellation (SC) Decoding

Arikan described a low-complexity decoding algorithm, SC decoding of polar codes in [2]. From the polar coding principle, the main idea is to select polarization channels. In this work, we use density evolution to obtain the probability distribution for each position. Then, the channel selection is actually based on the error rate for each position. As we mentioned in Sec 3.1.1, for any $n \geq 0, N = 2^n, 1 \leq i \leq N$, the polarization channel transfer probability function was given by:

$$\begin{aligned}
&W_{2N}^{(2i-1)}\left(y_1^{2N}, u_1^{2i-2} \mid u_{2i-1}\right) \\
&= \sum_{u_{2i}} \frac{1}{2} W_N^{(i)}\left(y_1^N, u_{1,o}^{2i-2} \oplus u_{1,e}^{2i-2} \mid u_{2i-1} \oplus u_{2i}\right) \cdot W_N^{(i)}\left(y_{N+1}^{2N}, u_{1,e}^{2i-2} \mid u_{2i}\right)
\end{aligned} \tag{3.15}$$

$$\begin{aligned}
&W_{2N}^{(2i)}\left(y_1^{2N}, u_1^{2i-1} \mid u_{2i}\right) \\
&= \frac{1}{2} W_N^{(i)}\left(y_1^N, u_{1,o}^{2i-2} \oplus u_{1,e}^{2i-2} \mid u_{2i-1} \oplus u_{2i}\right) \cdot W_N^{(i)}\left(y_{N+1}^{2N}, u_{1,e}^{2i-2} \mid u_{2i}\right)
\end{aligned}$$

According to the transfer probability function, each polarization channel is not independent of each other, but has a definite dependency relationship: the polarization channel with a larger channel number depends on all the polarization channels with channels with smaller numbers. In other words, the decoder makes a hard decision for bit u_i using all the previous estimates $\widehat{u}_1, \widehat{u}_2, \ldots, \widehat{u}_{i-1}$ as well as the entire received sequence $\mathbf{y} = (y_1, \ldots, y_N)$. Decoding begins by making an estimate $\widehat{u}_1 \in \{0,1\}$ using the received sequence $\mathbf{y}$. Then the decoder makes an estimate $\widehat{u}_2 \in \{0,1\}$ using $\widehat{u}_1$ and $\mathbf{y}$. Successive cancellation decoder proceeds in this way until it estimates $\widehat{u}_N$ using $\widehat{u}_1, \ldots, \widehat{u}_{N-1}$ and $\mathbf{y}$. Frozen bits are set to a fixed value and do not need to be estimated. Based on this dependency between the polarization channels, the SC decoding algorithm decodes the individual bits assuming that the results of the previous decoding steps are correct.

Given y_1^N and estimates $\widehat{u}_1^{i-1}$ of u_1^{i-1}, the SC decoding algorithm attempts to estimate u_i. This can be implemented by computing the following log-likelihood ratios (LLRs):

$$L_N^{(i)}\left(y_1^N, \widehat{u}_1^{i-1}\right) = \log \frac{W_N^{(i)}\left(y_1^N, \widehat{u}_1^{i-1} \mid u_i = 0\right)}{W_N^{(i)}\left(y_1^N, \hat{u}_1^{i-1} \mid u_i = 1\right)} \tag{3.16}$$

The LLR can be computed recursively by:

$$\begin{aligned} L_N^{(2j-1)}(y_1^N, \widehat{u}_1^{2j-2}) &= 2\tanh^{-1}\left(\tanh(\frac{\alpha_{N,j}}{2}) \cdot \tanh(\frac{\beta_{N,j}}{2})\right) \\ L_N^{(2j)}(y_1^N, \widehat{u}_1^{2j-1}) &= (-1)^{\widehat{u}_{2j-1}}\alpha_{N,j} + \beta_{N,j} \end{aligned} \tag{3.17}$$

where

$$\begin{aligned} \alpha_{N,j} &= L_{N/2}^{(j)}(y_1^{N/2}, \widehat{u}_{1,\text{even}}^{2j-2} + \widehat{u}_{1,\text{odd}}^{2j-2}), \\ \beta_{N,j} &= L_{N/2}^{(j)}(y_{N/2+1}^{N}, \widehat{u}_{1,\text{even}}^{2j-2}). \end{aligned}$$

Then the hard decision of $\widehat{u}_i$ is equivalent to:

$$\widehat{u}_i\left(y_1^N, \hat{u}_1^{i-1}\right) = \begin{cases} 0, \text{ if } L_N^{(i)}\left(y_1^N, \hat{u}_1^{i-1}\right) \geq 0 \\ 1, \text{ if } L_N^{(i)}\left(y_1^N, \hat{u}_1^{i-1}\right) < 0 \end{cases} \tag{3.18}$$

or

$$\widehat{u}_i\left(y_1^N, \hat{u}_1^{i-1}\right) = \delta\left(L_N^{(i)}\left(y_1^N, \hat{u}_1^{i-1}\right)\right) = \frac{1}{2}(1 - \text{sign}(L_N^{(i)}\left(y_1^N, \hat{u}_1^{i-1}\right))) \tag{3.19}$$

The SC decoding algorithm uses the LLR as the decision criterion. It performs

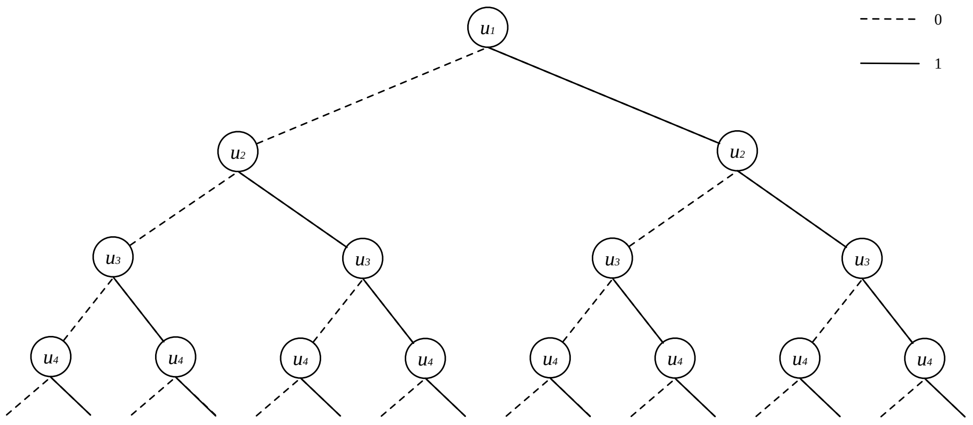


Figure 3.10: SC decoding tree structure.

a hard decision on each bit, then decodes bit-by-bit. As the code length tends to infinity, each information bit is decoded correctly as each split channel approaches complete polarization (its channel capacity is either 0 or 1), which can theoretically make polar codes reach the symmetric channel capacity $I(W)$. In addition, the complexity of the SC decoder is only $O(N \log N)$. However, with finite code lengths, due to incomplete channel polarization, some information bits cannot be decoded correctly.

Consider SC decoding on a tree structure as Fig. 3.10 shows. The SC decoder only selects the optimal branch and use the estimated value to decode the next layer. However, when an decoding error occurs in the $(i-1)$th layer, the error will be propagated to the ith layer. Then SC decoder continues decoding the rest layers by using the error; that is the most erroneous frames are due to one wrong decision. Therefore, polar codes using SC decoding do not perform as efficiently as other codes for finite code lengths.

SC Decoding Example

- Construct a $\mathcal{P}(4, 2, [1, 2])$ binary polar code with input sequence:

$$\mathbf{u} = [0, 0, 1, 0].$$

Note that: we froze the first two bits to 0.

- The codeword is obtained by:

$$\mathbf{x} = \text{mod}_2(\mathbf{u} \cdot \mathbf{G}_4) = [1, 1, 0, 0].$$

- Then $\mathbf{x}$ is transmitted over an AWGN channel with noise variance $\sigma^2 = 0.1$:

$$\mathbf{y} = [1.2119, \quad 1.2150, \quad -0.1108, \quad 0.0914].$$

- From Fig. 3.5, we use $L(\mathbf{u}_1)$ to denote the LLR value at position $\mathbf{u}_1$ and $\widehat{\mathbf{u}}_1$ to denote the estimated information of $\mathbf{u}_1$. The the LLR of $\mathbf{y}$ is:

$$L(\mathbf{y}) = [-7.1190, \quad -7.1500, \quad 6.1080, \quad 4.0860].$$

- Since $\mathcal{F} = [1, 2]$, $[\widehat{\mathbf{u}}_1, \widehat{\mathbf{u}}_2] = [0, 0]$. These values was propagated backward to obtain $[\widehat{\mathbf{v}}_1, \widehat{\mathbf{v}}_3] = [0, 0]$.

- By Eqn. 3.17,

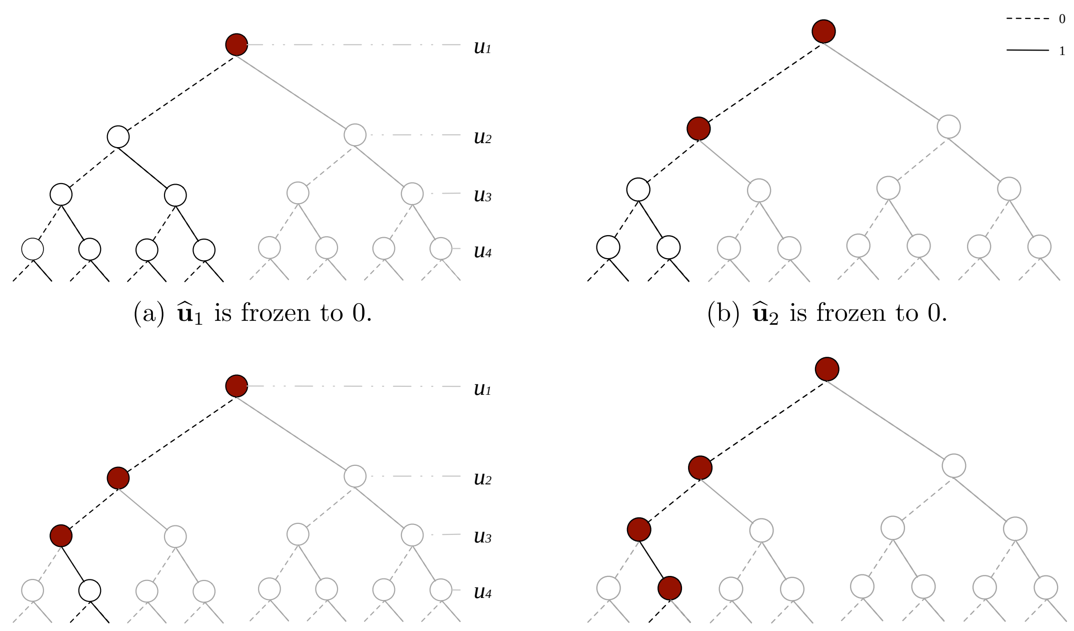
$$\begin{aligned} L(v_1) &= 2\tanh^{-1}\left(\tanh(\frac{L(y_1)}{2}) \cdot \tanh(\frac{L(y_2)}{2})\right) = 6.4412 \\ L(v_2) &= (-1)^{\widehat{v}_1} L(y_1) + L(y_2) = -14.2690 \\ L(v_3) &= 2\tanh^{-1}\left(\tanh(\frac{L(y_3)}{2}) \cdot \tanh(\frac{L(y_4)}{2})\right) = 3.9617 \\ L(v_4) &= (-1)^{\widehat{v}_3} L(y_3) + L(y_4) = 10.1940 \end{aligned} \tag{3.20}$$

- Similarly, $L(u_3)$ can be obtained by $L(v_2)$ and $L(v_4)$ which gives $L(u_3) = -10.1772$. By the hard decision shows in Eqn. 3.18, $\widehat{\mathbf{u}}_3 = 1$.

- Using $\widehat{\mathbf{u}}_3$, $L(v_2)$ and $L(v_4)$, $L(u_4) = 24.4630$, so that $\widehat{\mathbf{u}}_4 = 0$.

- We obtained $\widehat{\mathbf{u}} = [0, 0, 1, 0]$ that is $\widehat{\mathbf{u}} = \mathbf{u}$, BER=0 and WER=0.

The decoding processing on a tree structure can be shown as Fig. 3.11 that explains SC decoder decodes from a root node. It use the hard decision value of last layer to decode bit-by-bit.



(a) $\widehat{\mathbf{u}}_1$ is frozen to 0.

(b) $\widehat{\mathbf{u}}_2$ is frozen to 0.

(c) Since $L(u_3) = -10.1772$, $\widehat{\mathbf{u}}_3 = 1$.

(d) Since $L(u_4) = 24.4630$, $\widehat{\mathbf{u}}_4 = 0$.

Figure 3.11: Tree Structure of SC decoding example.

3.3.2 Successive Cancellation List (SCL) Decoding

SCL decoding is an improved decoding algorithm. To address the shortcomings of the SC decoding algorithm, an immediate improvement is to increase the number of candidate paths allowed to be retained after each layer of path search. SC decoding allows only the best path to be selected for the next layer; for SCL, it will enable the selection of L paths for the next layer, where $L \geq 1$. Like the SC decoding algorithm, SCL decoding still starts from the root node of the code tree and proceeds to the leaf node layer by layer. The difference is that after each layer of expansion, SCL decoding considers both possible values $u_i = 0$ and $u_i = 1$ that is as many successor paths as possible are retained. To avoid growing the number of paths exponentially at each layer, L upper bounds the number of the most reliable paths are preserved.

The path metric (PM) based on LLRs is used to measure the reliability of a path. Let $\hat{u}_i[l]$ denote the estimation of u_i in the l-th path, where $l \in \{1, 2, \ldots, L\}$ and $i \in \{1, 2, \ldots, N\}$. PM at $\hat{u}_i[l]$ can be approximated by:

$$PM_l^{(i)} = \begin{cases} PM_l^{(i-1)} + \left|L_N^{(i)}[l]\right| & \text{if } \hat{u}_i[l] \neq \hat{u}_i[l] = \delta\left(L_N^{(i)}[l]\right) \\ PM_l^{(i-1)} & \text{otherwise} \end{cases} \tag{3.21}$$

where $PM_l^{(1)} = 0$ and $\delta(x) = \frac{1}{2}(1 - \operatorname{sign}(x))$ as we mentioned in Eqn. 3.19. Then the hard decision of $\widehat{u_i}$ is equivalent to:

$$\hat{u}_i = \delta\left(L_N^{(i)}\left(y_1^N, \hat{u}_1^{i-1}\right)\right) \tag{3.22}$$

After completing the path expansion in one layer, the L entries with the smallest PM values are selected and saved in a list, waiting for the expansion in the next layer. As Eqn. 3.21 shows, the path of the less likely bit value is penalized by $\left|L_N^{(i)}[l]\right|$ of that bit. The L paths with smallest PMs are chosen from $2L$ paths at the ith step and stored in ascending order from $PM_1^{(i)}$ to $PM_L^{(i)}$. After decoding the N-th bit, the path with the smallest path metric $PM_1^{(N)}$ is selected as the estimated codeword. Therefore, the algorithm is called successive cancellation list decoding, and the parameter L is called the list-size. When $L = 1$, the SCL decoding degenerates to the SC decoding algorithm; when $L \geq 2K$, the SCL decoding is equivalent to the maximum likelihood decoding. SCL decoding with list size L complexity scales as $O(LN \log N)$.

The term $\mathcal{L}^{(i)}$ is used to refer to the set of candidate paths corresponding to the ith level of code tree $\mathcal{T}$ in the SCL decoder. $d_1^i = (d_1, d_2, \ldots, d_i) \in \mathcal{T}$ is defined as decoding path where $d_i \in \{0, 1\}, i \in \mathcal{I}$ consists of i branches from level 1 to level i. SCL decoding step can be explained as:

- **Step 1 Initialization:** The candidate path list is initialized to an empty path, and the corresponding PM is set to 0 that is $\mathcal{L}^{(1)} = \{\varnothing\}, PM^{(1)} = 0$.

- **Step 2 Extension:** For each node in the list, generate 2 sequences of length i corresponding to the estimation of $\widehat{u_i} = 0$ or 1 that can be explained as:

$$\mathcal{L}^{(i)} = \left\{\left(d_1^{i-1}, d_i\right) \mid d_1^{i-1} \in \mathcal{L}^{(i-1)}, d_i \in \{0, 1\}\right\} \tag{3.23}$$

  for each $d_1^i \in \mathcal{L}^{(i)}$ update the PM.

- **Step 3 Comparison:** After step 2, if the number of paths in the list does not reach L paths, skip this step. Otherwise, save the first L paths with smallest PM values and delete the remaining paths.

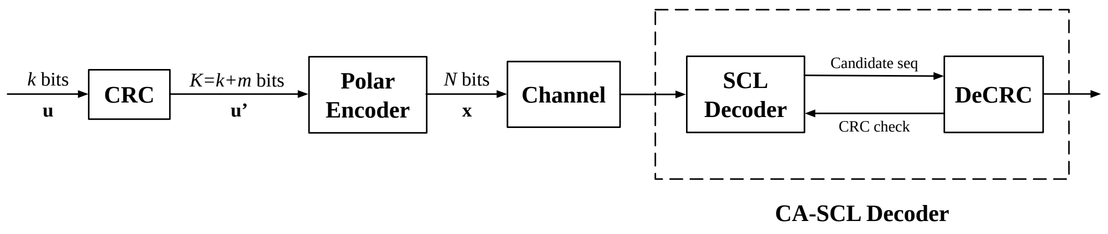


Figure 3.12: Polar coding and CRC-aided decoding schemes.

- **Step 4 Selection:** Repeat step 2 and 3 until the Nth level of the code tree. Sort the candidate paths by PM values, select the $1st$ sequence as the output.

In addition, when the SCL decoding fails to decode, that is: $\widehat{\mathbf{u}} \neq \mathbf{u}$ or $\widehat{\mathbf{x}} \neq \mathbf{x}$. The correct path might still be in the list but not the most likely path. [15] proposed a combination of cyclic redundancy check aided successive cancellation (CA-SCL) decoder to further improve the performance of polar codes. The main idea is to add an m-bit cyclic redundancy check (CRC) code as an additive code to the information bits. It can help the decoder to determine if error exist and find the correct path among the L paths.

As shown in Fig. 3.12, assume $\mathbf{u} = (u_1, \ldots, u_k)$ is the information bits to be transmitted. Append m-bit CRC to $\mathbf{u}$ hence the data fed to the encoder is $\mathbf{u}' = (u_1, \ldots, u_k, p_1, \ldots, p_m)$, and $\mathbf{p} = (p_1, \ldots, p_m)$ denotes the generated CRC bits. Encoding $\mathbf{u}'$ as information bits of a polar code and transmit over a channel. The SCL decoder will not output an $\widehat{\mathbf{u}}$ as the estimation, but outputs L candidate sequences $\widehat{\mathbf{u}}_1, \ldots, \widehat{\mathbf{u}}_L$ into CRC detector. The CRC detector checks these sequences and gives check results determined the codeword.

For CA-SCL decoding, the only difference from SCL decoding is the $4th$ step:

- **Step 4 CRC check and Selection:** Repeat step 2 and 3 until the Nth level of the code tree. Sort the candidate paths by PM values and perform CRC check sequentially. The $1st$ path that passes CRC check is the estimated sequence output by the decoder. If no path passes CRC check, the $1st$ path is used as the decoder output estimation sequence.

However, by adding CRC bits, it increases the polar code rate R from k/N to $(k+m)/N$ that means at low SNR the performance of CA-SCL might not better than SCL. In this thesis, $P(N, k+m)$ denotes a polar code of length N with k information bits concatenated with m-bit CRC.

3.3.3 Ordered Statistic Decoding (OSD)

In [13], ordered statistics decoding (OSD) is introduced as a general method for decoding a linear binary block code. It can be considered as an simplified of the maximum likelihood (ML) decoder for linear block codes. OSD is a type of most reliable independent position (MRIP) processing decoding algorithm. For a code which has K information bits, it performs bit-flips over at most K bits. Due to the high complexity, it is efficient only for short block length codes. In this section, we mainly talk about OSD for binary polar codes that has been described in [14].

Let u_1^K be an information sequence of $\mathcal{P}(N, K, \mathcal{F})$ code. After encoding, transmit the codeword x_1^N over the AWGN channel. The code can be represented by $\mathbf{G}_{(N,K)}$ that is an $K \times N$ generator matrix. For $i = 1, \ldots, N$, the LLRs of each bit L_i^{osd} based on the channel output sequence $\mathbf{y}$ are given by:

$$L_i^{\text{osd}} = \log \frac{W(y_i \mid 0)}{W(y_i \mid 1)} = \frac{1 - 2y_i}{2\sigma^2} \tag{3.24}$$

The OSD algorithm requires to find the position of the MRIP from the columns of $\mathbf{G}$ due to absolute values of LLR in L^{osd} and a reprocessing stage. Here, position with great absolute value of LLR means more reliable.

To find the MRIP, first, reorder $\mathbf{y}$ according to the reliability in descending order that can be representated by:

$$\tilde{\mathbf{y}} = \lambda \cdot \mathbf{y} \tag{3.25}$$

where λ is a permutation function corresponding to the reordering.

Apply the same ordering on the columns of $\mathbf{G}$, i.e., $\mathbf{G}' = \lambda \cdot \mathbf{G}$ in order to obtain the first K positions of linearly independent columns by Gaussian elimination. Then by performing elementary row operations on $\mathbf{G}'$, we obtained the systematic form of $\mathbf{G}'$ that can be represented:

$$\tilde{\mathbf{G}} = [\mathbf{I}_K \mid \mathbf{A}] \tag{3.26}$$

where $\mathbf{I}_K$ is the $K \times K$ identity matrix, and $\mathbf{A}$ is a $K \times (N-K)$ matrix. The first K columns of $\tilde{\mathbf{G}}$ stand for the most reliable independent basis columns of $\mathbf{G}$ correspond to the indices in L^{osd}.

Perform permutation λ of $\mathbf{L}^{\text{osd}}$ that $\tilde{\mathbf{L}}^{\text{osd}} = \lambda \cdot \mathbf{L}^{\text{osd}}$. For the first K values that is $1 \leq i \leq K$, $\tilde{L}_i^{\text{osd}}$ can be used to obtain an initial estimation $\hat{c}_1^K$ by using hard decisions:

$$\hat{c}_i = \begin{cases} 0, & \tilde{L}_i^{\text{osd}} \geq 0 \\ 1, & \text{else} \end{cases} \tag{3.27}$$

Then, a hard decision codeword can be obtained by $\hat{a}_1^N = \hat{c}_1^K G_1$. The estimated codeword can be evaluated by $\hat{x}_1^N = \lambda^{-1} \cdot \hat{a}_1^N$.

To achieve better performance, after we obtained $\widehat{\mathbf{x}}$ and $\widehat{\mathbf{u}}$, defined an l-order OSD that allows at most l reprocessing stage where $0 \leq l \leq K$. For each $0 \leq i \leq l$, flip all possible combinations of i bits in $\hat{\mathbf{u}}$. The term $\mathcal{E}$ refers to a set of error patterns. The cardinality of error patterns $|\mathcal{E}|$ is $\sum_{i=0}^{l} \left(\begin{array}{c} K \\ i \end{array} \right)$. For each error pattern $\mathbf{e}$, re-encode $\hat{\mathbf{u}} \oplus \mathbf{e}$ using $\tilde{\mathbf{G}}$, and calculate the Euclidean distance between the BPSK representation of the resulting codeword $\mathbf{m}$ and the permuted channel output vector $\tilde{\mathbf{y}}$ by:

$$d\left(m_1^N, \tilde{y}_1^N\right) = \sum_{i=1}^{N} \left(1 + \tilde{y}_i^2\right) - 2\sum_{i=1}^{N} m_i\, \tilde{y}_i \tag{3.28}$$

where for a given $\mathbf{y}$, $\sum_{i=1}^{N} (1 + \tilde{y}_i^2)$ is a constant. After calculating the Euclidean distances for all possible codewords, select the permuted codeword $\mathbf{m}$ with minimum Euclidean distance from $\tilde{\mathbf{y}}$, and inversely permute $\mathbf{m}$ by $\widehat{\mathbf{x}} = \lambda^{-1} \cdot \mathbf{m}$. Finally, the OSD($l$) gives $\widehat{\mathbf{x}}$ as an estimation of the transmitted codeword. Order-l OSD decoding complexity is proportional to $\sum_{i=0}^{l} \binom{K}{i}$. For OSD using different order, the performances are clearly different.

OSD Decoding Example

Given a example for $\mathcal{P}(128, 7, \mathcal{F})$ code transmit over an AMGN channel with noise variance σ^2 under OSD with different order. (AMGN channel is introduced in Sec.5.1.1.) The performances shows in Fig.3.13. The information set we use is $\mathcal{I} = [128, 127, 126, 124, 120, 112, 96]$.

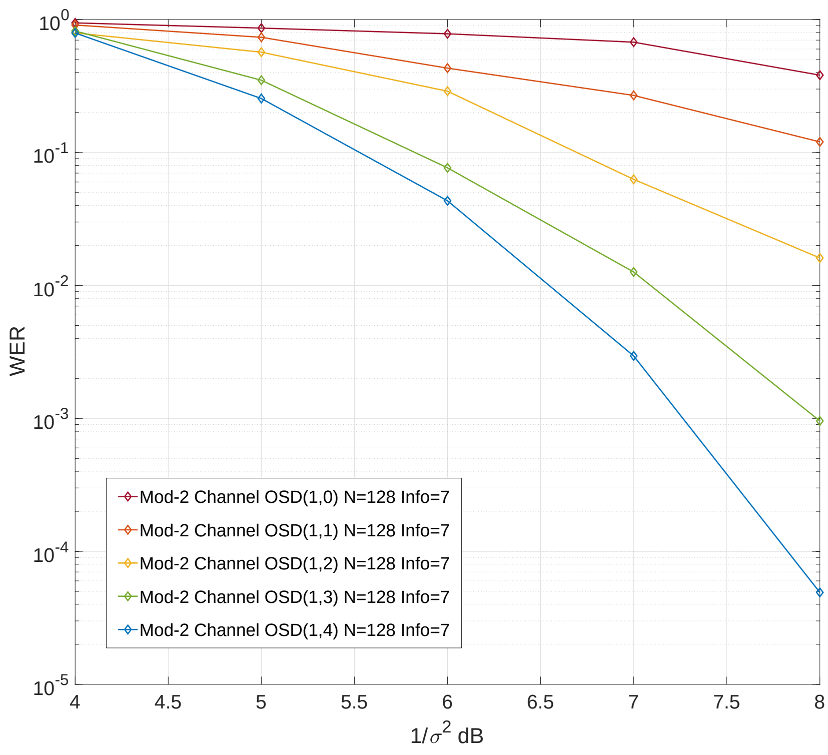


Figure 3.13: On a AMGN channel with noise σ^2, $\mathcal{P}(128, 7, \mathcal{F})$ under different order of OSD.

Chapter 4

Construction D Lattices

4.1 Lattices

In wireless communications, lattices can implement shaping technique efficiently to obtain about 1.53 dB shaping gain on the AWGN channel and can be used for physical layer network coding scheme compute and forward.

An n-dimensional lattice Λ is a discrete additive subgroup of $\mathbb{R}^n$ and can be represented by $\Lambda \subset \mathbb{R}^n$. In an n-dimensional space, a lattice point $\mathbf{x}$ is a set of points given by integral, linear combination of up to n linearly independent basis vectors, and can be represented as:

$$\mathbf{x} = \mathbf{g}_1 b_1 + \mathbf{g}_2 b_2 + \cdots + \mathbf{g}_n b_n \tag{4.1}$$

where $\mathbf{g}_i$ are linearly independent basis vectors and b_i are integers.

Any lattice Λ in $\mathbb{R}^n$ is spanned by some $n \times n$ generator matrix $\mathbf{G}_\Lambda$ such that:

$$\Lambda = \{\mathbf{x} = \mathbf{G}_\Lambda \mathbf{b} : \mathbf{b} \in \mathbb{Z}^n\} \tag{4.2}$$

where the generator matrix $\mathbf{G}_\Lambda$ is:

$$\mathbf{G}_\Lambda = \begin{bmatrix} | & | & & | \\ \mathbf{g}_1 & \mathbf{g}_2 & \cdots & \mathbf{g}_n \\ | & | & & | \end{bmatrix} \tag{4.3}$$

From the definition, it is clear that lattices are codes based on the real field $\mathbb{R}^n$ but not binary. It makes lattices closer to the real physical channel.

Lattices can be formed by different constructions [6] [7] such as Construction A, Construction B and Construction D. While Construction A makes lattices from one code but has limitation that binary codes are not practical at high block length. Construction B is a special case of Construction D. The subject of this thesis is to form polar code lattices using Construction D. Construction D forms lattices from nested binary codes using a multilevel construction. Furthermore, Construction D allows decoding lattices using binary code decoders that significantly reduce the decoding complexity.

4.2 Construction D

Fig. 4.1 [16] shows the multi-level of Construction D lattices which are formed by nested binary codes.

4.2.1 Nested Binary Code

For $a \geq 1$, $C_0 \subseteq C_1 \subseteq \cdots \subseteq C_a = \mathbb{F}_2^n$ are nested binary code if $\mathbf{g}_1, \mathbf{g}_2, \ldots, \mathbf{g}_{k_i}$ span C_i, where $\mathbf{g}_1, \mathbf{g}_2, \ldots, \mathbf{g}_n$ is a basis for $\mathbb{F}_2^n$ that can be written as an $n \times n$ matrix:

$$\widetilde{\mathbf{G}} = \begin{bmatrix} | & | & & | & & | & & | \\ \mathbf{g}_1 & \mathbf{g}_2 & \cdots & \mathbf{g}_{k_0} & \cdots & \mathbf{g}_{k_1} & \cdots & \mathbf{g}_n \\ | & | & & | & & | & & | \end{bmatrix} \tag{4.4}$$

Let $\tilde{\mathbf{G}}_0, \cdots, \tilde{\mathbf{G}}_{a-1}, \tilde{\mathbf{G}}$ be generator matrices of binary sub-code $C_0, C_1, \cdots, C_a$. Table. 4.1 gives several generator matrices that shows $C_0 \subseteq C_1 \subseteq \cdots \subseteq C_a$.

4.2.2 Construction D Lattice Encoding and Decoding

$\widetilde{\mathbf{G}}$ can be used to construct the generator matrix of Construction D as a specific basis for $\mathbb{F}_2^n$ with $k_0, k_1, \ldots, k_{a-1}$. The Construction D generator matrix $\mathbf{G}_{\Lambda_D}$ is

Table 4.1: Generator matrices for nested binary code.

Code C_i	Generator matrix $\tilde{\mathbf{G}}_i$
C_0	$\tilde{\mathbf{G}}_0 = [\mathbf{g}_1, \mathbf{g}_2, \ldots, \mathbf{g}_{k_0}]$
C_1	$\tilde{\mathbf{G}}_1 = [\mathbf{g}_1, \mathbf{g}_2, \ldots, \mathbf{g}_{k_0}, \ldots, \mathbf{g}_{k_1}]$
$\cdots$	$\cdots$
C_a	$\tilde{\mathbf{G}}_a = [\mathbf{g}_1, \mathbf{g}_2, \ldots, \mathbf{g}_{k_0}, \ldots, \mathbf{g}_{k_1}, \ldots, \mathbf{g}_n]$

given by:

$$\mathbf{G}_{\Lambda_D} = \widetilde{\mathbf{G}} \cdot \mathbf{D}^{-1}, \tag{4.5}$$

where $\mathbf{D}$ is a diagonal matrix with diagonal entries d_{ii} :

$$\mathrm{d}_{ii} = 2^{-k} \text{ for } r_{k-1} \leq i \leq r_k \tag{4.6}$$

with $k = 0, 1, \ldots, a$.

From Fig. 4.1, basically, Construction D forms a lattice from two or more codes that $a \geq 2$. The Construction D lattice Λ_D consists of all vectors of the form can be represented by:

$$\mathbf{x} = \sum_{i=0}^{a-1} 2^i \widetilde{\mathbf{G}}_i \cdot \mathbf{u}_i + 2^a \widetilde{\mathbf{G}} \cdot \mathbf{z} \tag{4.7}$$

where $\mathbf{z} \in \mathbb{Z}^n$ and $\mathbf{u}_i = (u_{1,i}, u_{2,i\cdots}, u_{k,i})^t, i \in \{0, 1, ..., a-1\}$ and $u_{j,i} \in \{0, 1\}$ are information bits. A lattice point $\mathbf{x}$ can be also represented as:

$$\mathbf{x} = \mathbf{x}_0 + 2\mathbf{x}_1 + \cdots + 2^{a-1}\mathbf{x}_{a-1} + 2^a\mathbf{z}. \tag{4.8}$$

As we mentioned before, for lattices, operations are over the real field but not the binary field. For Construction D lattices Λ_D, the volume $V(\Lambda_D) = |\det(\mathbf{G}_{\Lambda_D})|$ is defined:

$$V(\Lambda_D) = 2^{aN - N\sum_{i=0}^{a-1} R_i}. \tag{4.9}$$

$\mathbf{x} \in \Lambda_D$ is transmitted over an AWGN channel with $\mathbf{w} \sim \mathcal{N}(0, \sigma^2)$ and the received sequence is $\mathbf{y}$ refers to

$$\mathbf{y} = \mathbf{x} + \mathbf{w}.$$

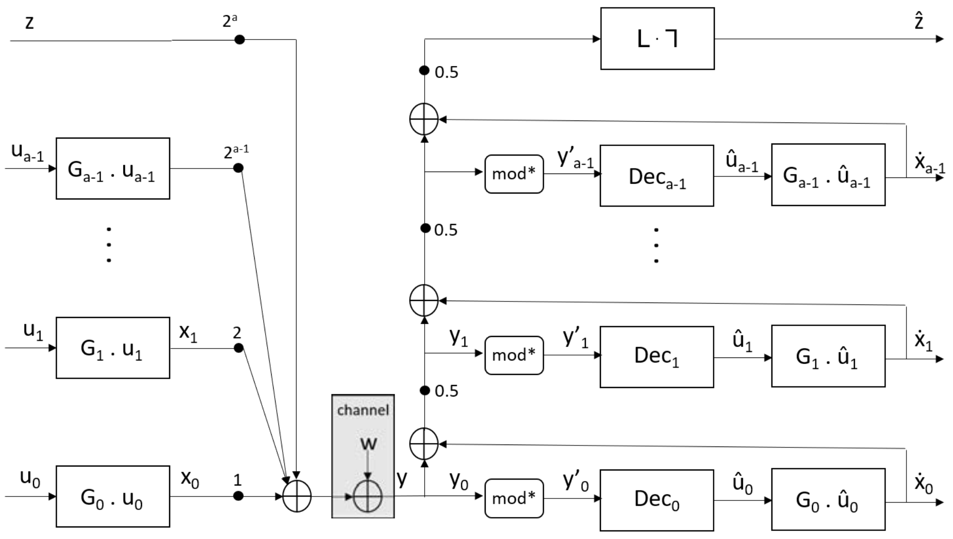


Figure 4.1: Multilevel construction of Construction D.

By using the unconstrained power channel, the lattice is constrained by the volume of the Voronoi region. The volume-to-noise ratio (VNR) is

$$\text{VNR} = \frac{V(\Lambda_D)^{2/N}}{2\pi e \sigma^2}. \tag{4.10}$$

so that VNR is the distance to the Poltyrev limit.

Before decoding the component code, perform a modulo operation to preserve distances to $(0, 1)$. The modulo operation can be represented by:

$$\text{mod}^* (y_i) = |\text{mod}_2 (y_i + 1) - 1| \tag{4.11}$$

where mod_2 refers to a modulo-2 function.

After performing the modulo operation, the result y'_i is the input to binary code decoder Dec_i and then, obtain the estimated information sequence $\widehat{\mathbf{u}}_i$. By using binary code decoder, the decoding complexity of lattice can be reduced. Re-encode $\widehat{\mathbf{u}}_i$ to a codeword $\widehat{\mathbf{x}}_i$. Again, either the encoding or the re-encoding is over the real field. Then propagate $\widehat{\mathbf{x}}_i$ to next level and obtain y_i by $y_{i+1} = \frac{y_i - \hat{x}_i}{2}$. The estimated lattice point can be represented by:

$$\hat{\mathbf{x}} = \hat{\mathbf{x}}_1 + 2\hat{\mathbf{x}}_2 + \cdots + 2^{a-1}\hat{\mathbf{x}}_{a-1} + 2^a\hat{\mathbf{z}}. \tag{4.12}$$

From the decoding processing we introduced above, Construction D start to decode C_0, then by using the estimation of last level, then continue decoding layer-by-layer. This processing can be interpreted as successive cancellation decoding but not successive cancellation decoding for polar code we introduced before.

Construction D forms lattices from nested binary codes and allows decoding using binary code decoder. However, how to choose the code rate for each component code is a challenge we have to solve.

4.2.3 Design of Construction D Lattices

Construction D lattices are formed by nested binary codes $C_0, C_1, \cdots, C_a$. For a $C_i\,(n, k_i)$ code, k_i is the dimension $dim(C_i)$ and $r_i = n - k_i$ refers to the number of rows in parity check matrix. Code rate can be represented by $R_i = k_i/n$. It also has minimum distance d_i. Then, the code nesting gives all of the code rate, the dimension, the number of rows in parity check matrix and the minimum distance are also nested, that is:

$$\begin{aligned} &R_0 \leq R_1 \leq \cdots \leq R_a \\ &k_0 \leq k_1 \leq \cdots \leq k_a \\ &r_0 \leq r_1 \leq \cdots \leq r_a \text{ and} \\ &d_0 \leq d_1 \leq \cdots \leq d_a \end{aligned} \tag{4.13}$$

From the properties above, Construction D lattices can be designed by following various rules such as:

- Balanced distances rule

  Balanced distances rule uses the minimum distance d in the design and have systematic approach for moderate dimension. It has been used to design the Barnes-Wall lattice Λ_{16}, BCH code lattices [17], etc. However, it is hard to find the minimum distance for long block length codes. Designing polar code lattices using minimum distance is a topic for future research.

- Capacity design rule

  Capacity design rule allows selecting the component codes that achieve the capacity of additive mod-2 Gaussian noise (AMGN) channel on each layer of Construction D. It is reasonable to use capacity design rule in high dimension lattices [5], but in this work we consider to design lattices of moderate dimension.

- Probability of error design rule

  Probability of error design rule uses the probability of error on the multilevel structure in the design. In this work, we use equal error probability rule which is actually use the probability of error in the design. On each level, the error probability is set to be equal so that the each level contributes equally to the union bound of the lattice error probability. Using density evolution can avoid doing a lot of simulations. It will be introduced more detail in Sec. 5.1.1

Chapter 5

Polar Code Lattices

In this chapter, we introduce proposed methods in designing polar code lattices. Then we show how we design the polar code lattices and give a design example to introduce design methods in more detail. Finally, simulation results are shown in Sec. 5.4.

5.1 Design Methods

5.1.1 Equal Error Probability Rule

The equal error probability rule was given by Wachsmann et al. [8] and can be used to design multilevel codes. Due to the multilevel structure of Construction D as Fig. 4.1 shows, the estimate in the ith level is used to decode the codeword in $i+1$th level, so that level $i+1$ is decoded successfully only if level i gives the correct estimation. Under the assumption that all levels are decoded successfully, each level can be treated as coding over independent channels. Fig. 5.1 shows the construction with the equivalent encoder, channel and decoder for multilevel decoding of Construction D. On each equivalent layer, each channel can be seen as AMGN channel with noise $\mathbf{w}_0 \sim \mathcal{N}(0, \sigma^2)$, $\mathbf{w}_1 \sim \mathcal{N}(0, s_1)$, $\mathbf{w}_2 \sim \mathcal{N}(0, s_2), \cdots$, $\mathbf{w}_i \sim \mathcal{N}(0, s_i)$ for $i = 1, 2, \cdots$. It is clear that $s_i = \sigma^2/4^i$. Then, $P_{\mathrm{e}}(C_i, s_i)$ denotes the decoding error probability on each level.

Nested binary codes $C_0, C_1, \cdots, C_{a-1}$ form a Construction D lattice, the error

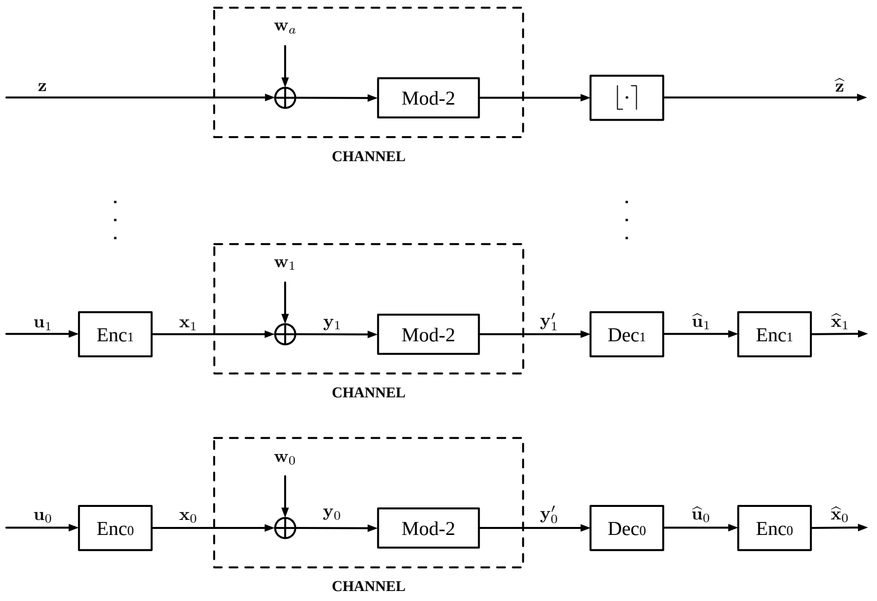


Figure 5.1: Equivalent layer of Construction D.

probability of decoding lattice is defined as:

$$P_{\mathrm{e}}(\sigma^2).$$

There is a relationship between the decoding error probability on each level $P_{\mathrm{e}}(C_i, s_i)$ and the decoding error probability for lattice $P_{\mathrm{e}}(\sigma^2)$. By the union bound:

$$P_{\mathrm{e}}(\sigma^2) \leq P_{\mathrm{e}}(C_0, \sigma^2) + P_{\mathrm{e}}(C_1, \frac{\sigma^2}{4}) + \cdots + P_{\mathrm{e}}(C_a, \frac{\sigma^2}{4^a}). \tag{5.1}$$

Under the equal error probability rule, the codes C_i are selected such that $P_{\mathrm{e}}\left(C_i, \frac{\sigma^2}{4^i}\right)$ are equal.

5.1.2 ρ Function for binary polar codes

In this section, we define a target error rate P_{trgt} that can be used to construct a function ρ. Given a target error rate P_{trgt}, consider a binary polar code $\mathcal{P}(N, K, \mathcal{F})$ transmitted over an AMGN channel with noise variance σ^2. To reach the P_{trgt}, as the code rate R increases, the SNR is also needed to increases. Equivalently, the value of σ^2 required to achieve P_{trgt} will decrease as K increases.

We propose a function ρ to express this trade-off. Given the σ^2 and P_{trgt}, let $\rho\,(\sigma^2, P_{\mathrm{trgt}})$ denote the greatest code rate such that the decoder word-error rate

$P_e(C, \sigma^2)$ is not greater than a target error rate P_{trgt}. When N becomes small, $P_e(C, \sigma^2)$ may be significantly smaller than P_{trgt} because K is an integer. Evidently, the function $\rho(\sigma^2, P_{\text{trgt}})$ depends on the decoding algorithm, the number of CRC bits, and the method to select the frozen bits.

For polar codes with successive cancellation decoding, the function $\rho(\sigma^2, P_{\text{trgt}})$ may be found efficiently by density evolution. Since the AMGN channel is symmetric, analysis of the all-zeros codeword by density evolution is sufficient. Recall that p_j is the probability of error in position j under the assumption of positions 1 to $j-1$ are correct, shown in Equ. 3.14, and $\mathcal{I}$ refers to the set of information bits. Then, under density evolution, the probability of word error for a polar code C can be calculated as:

$$P_e\left(C, \sigma^2\right) = 1 - \prod_{j \in \mathcal{I}} (1 - p_j)\,. \tag{5.2}$$

For a fixed channel, $\rho(\sigma^2, P_{\text{trgt}})$ can be set equal to the code rate R, so that under density evolution the decoder error rate $P_e(C, \sigma^2)$ is as high as possible, while satisfying $P_e(C, \sigma^2) \leq P_{\text{trgt}}$.

Using density evolution, it is feasible to obtain the $\rho(\sigma^2, P_{\text{trgt}})$ function under SC decoding, but not for other decoders. For other decoders, such as in [16], they found function ρ under successive cancellation list decoding by Monte Carlo simulations. For a given number of information bits K, it requires a large number of simulations to find the noise variances σ^2 which produce decoder error rates both above and below P_{trgt}.

For different code lengths under different decoding methods, the function $\rho(\sigma^2, P_{\text{trgt}})$ is shown in Fig. 5.2. Under successive cancellation decoding with $N = 128, \ldots, 262144$, ρ is shown for a target error rate of $P_{\text{trgt}} = \frac{1}{3} \cdot 10^{-4}$. For reference [16], under SCL decoding with list size 8 and 10 CRC bits and $N = 128$, ρ is shown for $P_{\text{trgt}} = 10^{-4}$ and the capacity is also given [5, Fig. 6] [8, Fig. 7]. As can bee seen from the figure, as the block length N increases, the ρ curve is approaches the S-shape characteristic of the capacity curve.

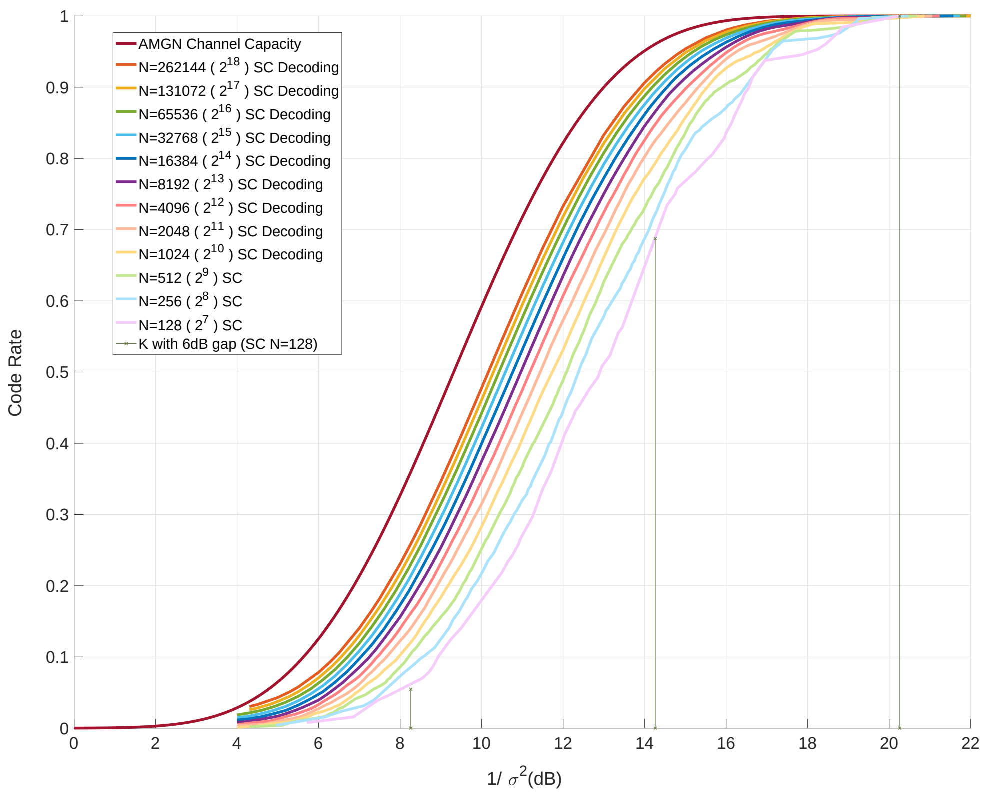


Figure 5.2: On a AMGN channel with noise σ^2, the function $\rho(\sigma^2, P_{\text{trgt}})$ gives a code rate to achieve SC decoder $P_{\text{trgt}} = \frac{1}{3} \cdot 10^{-4}$.

5.2 Design of Polar Code Lattices

We design Construction D polar code lattices using two or more nested binary polar code $C_0, C_1, \cdots, C_{(a-1)}$. For binary polar code C_i with information bit indices $\mathcal{I}_i$ on each level. To satisfy the nesting condition, it has:

$$\begin{aligned} &C_0 \subseteq C_1 \subseteq \cdots \subseteq C_{a-1} \quad \text{and} \\ &\mathcal{I}_0 \subseteq \mathcal{I}_1 \subseteq \cdots \subseteq \mathcal{I}_{a-1}. \end{aligned} \tag{5.3}$$

The structure of information sets are naturally nested to comply with requirements of Construction D.

However, to design Construction D polar code lattices, the challenge is to choose the information bits K_i (or code rates R_i) for component codes C_i such that the polar code lattice has the lowest possible VNR for the word error rate P_e. Under the equal error probability rule, each component code $C_0, C_1, \ldots, C_{a-1}$ should have equal error probabilities:

$$P_e\left(C_0, \sigma^2\right) = P_e\left(C_1, \sigma_1^2\right) = \cdots = P_e\left(C_{a-1}, \sigma_{a-1}^2\right), \quad \sigma_i^2 = \sigma^2/4^i$$

when decoding on the independent equivalent channel shows in Fig. 5.1. Combining the ρ function and equal error probability rule we proposed in Sec. 5.1, the error probability for the component codes $P_e\left(C_i, \sigma_i^2\right)$ is set to a target error rate P_{trgt}. The function $\rho\left(\sigma^2, P_{\text{trgt}}\right)$ gives the greatest code rate under $P_e(C, \sigma^2) \leq P_{\text{trgt}}$. Moreover, each layer sees an AMGN channel as described in Sec. 5.1.1.

Under a designed lattice error rate of P_e, the ρ function can be used to design a Construction D polar code lattice of dimension N by using at least two binary polar codes of block length N. Following the union bound in Equ. 5.1, the designed lattice error rate of P_e needs to satisfy the conditions of $P_e = (a+1) \cdot P_{\text{trgt}}$ for an a-level lattice where P_{trgt} is the target error rate for component codes. In addition, under the equal error probability rule, we allow $P_{\text{trgt}} \approx P_e\left(C_i, \sigma_i^2\right)$, for $i = 0, 1, \ldots, a$.

The ath level of Construction D refers to the integer level which means uncoded. In this level, $K_a = N$ and $R_a = 1$. Let $\sigma_a^2 = \sigma^2/4^a$ be the noise variance of level a that the decoder error rate achieves P_{trgt}. It has:

$$\rho\left(\sigma_a^2, P_{\text{trgt}}\right) = \frac{K_a}{N} = 1. \tag{5.4}$$

In addition, ρ function allow us obtain the σ_a^2 from the equation above by:

$$\sigma_a^2 = \rho^{-1}\left(N, P_{\text{trgt}}\right). \tag{5.5}$$

[18, eqn. (5)] shows that in level a, the probability of error $P_{\mathrm{e}}(C_a, \sigma_a^2)$ can be computed explicitly. Under the equal error probability rule, we have assumed that $P_{\text{trgt}} \approx P_e\left(C_a, \sigma_a^2\right)$. Then, by the inverse of the function, σ_a^2 can be obtained by:

$$\sigma_a^2 = \frac{1}{8 \cdot \left(\operatorname{erfc}^{-1}\left(1 - \sqrt[N]{1 - P_{\text{trgt}}}\right)\right)^2}, \tag{5.6}$$

where erfc^{-1} is the inverse of the complementary error function.

With the fixed σ_a^2, use the function ρ to find the rates of component codes R_i for

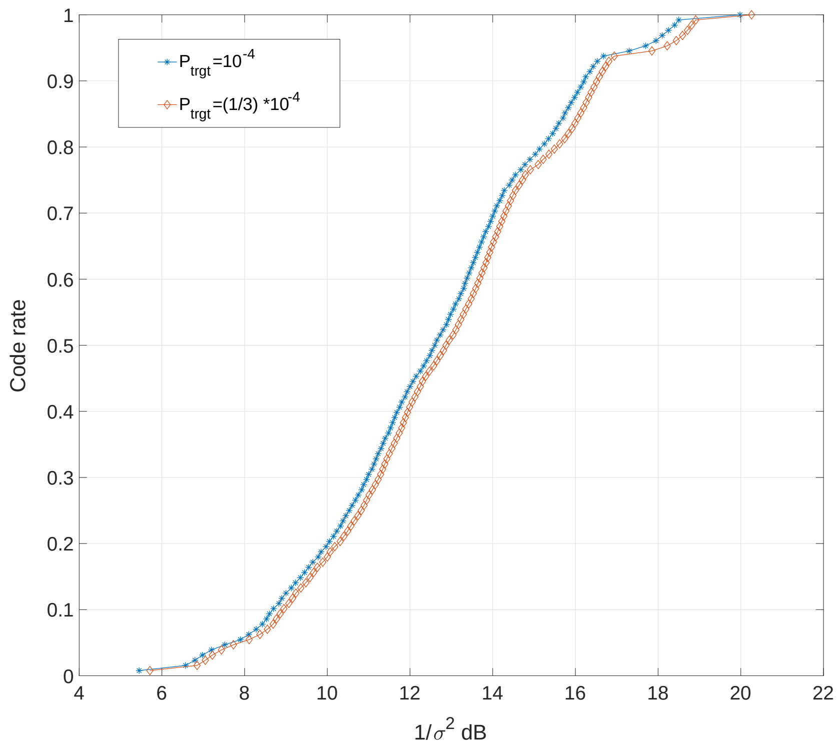


Figure 5.3: The function $\rho(\sigma^2, P_{\text{trgt}})$ gives a code rate to achieve $P_{\text{trgt}} = 10^{-4}$ and $P_{\text{trgt}} = \frac{1}{3}10^{-4}$ over an AMGN channel with noise variance σ^2.

$i = 0, \ldots, a-1$:

$$R_i = \rho\left(4^{a-i}\sigma_a^2, P_{\text{trgt}}\right). \tag{5.7}$$

From the equation above, it is clear that for different values of P_{trgt} R_i is also different. For $N = 128$ polar codes, given $P_{\text{trgt}} = 10^{-4}$ and $P_{\text{trgt}} = \frac{1}{3}10^{-4}$, the different ρ curves are shown in Fig. 5.3.

σ_i^2 refers to the noise variance of the equivalent AMGN channel in level i which is shown in Fig. 5.1. By the definition of Eqn. 2.3, here, we represent $\frac{1}{\sigma_i^2}$ dB by SNR$_i$ dB in order to make the function clear. In the design of polar code lattices, as Fig. 5.2 shows. We use the 6 dB gap of SNR dB between two level which corresponds to the factor of 4, that is the 6 dB gap satisfies $\sigma_i^2 = \sigma^2/4^i$ for $i = 0, \ldots, a$. For $j \geq i$ and $i, j \in \{0, 1, \ldots, a\}$, it has:

$$\text{SNR}_i \text{ dB} = \text{SNR}_j \text{ dB} - (j-i)\cdot 6 \tag{5.8}$$

5.3 Design Example

In this section, we propose a design example of Construction D polar code lattice with $N = 128$. Design a polar code lattice with $a = 2$, $N = 128$ and a designed lattice error rate $P_\mathrm{e} = 10^{-4}$. Since $a = 2$, under equal error probability rule, $P_\mathrm{trgt} = \frac{1}{3} \cdot 10^{-4}$. By Equ. 5.6, we obtain the $\sigma_2^2 = 0.0094258$, which is SNR_2 dB = 20.26 dB. Continuing the design procedure, by Fig. 5.8 using the 6dB gap, it means:

$$\begin{aligned} \mathrm{SNR}_0 \text{ dB} &= \mathrm{SNR}_2 \text{ dB} - 12 \text{ dB} \\ \mathrm{SNR}_1 \text{ dB} &= \mathrm{SNR}_2 \text{ dB} - 6 \text{ dB}. \end{aligned} \tag{5.9}$$

Since SNR_2 dB = 20.26 dB, evidently, SNR_0 dB = 8.26 dB and SNR_1 dB = 14.26 dB. Not only for the SC decoder, but these values are also feasible for any other decoders to design $a = 2$, $N = 128$ polar code lattice under $P_\mathrm{trgt} = \frac{1}{3} \cdot 10^{-4}$.

5.3.1 Polar Code Lattices Under SC Decoding

Under SC decoding, from Fig. 5.2, we obtain the design of $K_0 = 7$ at SNR_0 dB = 8.26 dB and $K_1 = 88$ at SNR_1 dB = 14.26 dB by density evolution efficiently. Density evolution allows calculating the error probability for each position. A position with a small value of error probability means more reliable. We sort the positions in increasing order of error probability from left to right and summarize in Table. 5.1 and choose the first 7 and 88 positions at 8.26 dB and 14.26 dB that marked to red. Note that: the information set satisfy $\mathcal{I}_0 \subseteq \mathcal{I}_1$, thus the nested structure needed by Construction D is met. In addition, $\mathcal{I}_0$ is bolded in $\mathcal{I}_1$.

In addition, Table. 5.2 shows polar code lattice designs for various dimensions N, based on density evolution result in Fig. 5.2 under the defined lattice error rate $P_\mathrm{e} = 10^{-4}$.

5.3.2 Polar Code Lattices Under SCL

We consider the design of polar code lattices using CA-SCL decoder. Since SCL decoding has good performance-complexity trade-off at high SNR, CRC can help to improve the performance. For moderate dimension, selecting the number of CRC bits is particularly important to the performance of the SCL decoder. [19] proposed

Table 5.1: RELIABILITY FOR $N = 128$ POLAR CODE UNDER DIFFERENT VNR.

	128	127	126	124	120	112	96	125	123	122	119	64	118	111	116	110
	108	95	94	104	121	92	63	117	62	88	115	109	60	114	107	80
	56	93	106	103	91	48	102	90	61	7	87	100	32	59	113	86
At 8.26 dB	79	58	84	55	105	78	54	76	47	101	52	89	46	72	99	31
	3	44	19	85	35	30	98	57	11	40	83	67	77	28	53	75
	5	51	45	24	15	71	97	29	43	13	82	81	23	39	73	27
	49	21	41	9	25	69	37	65	33	17	1	16	74	50	6	70
	42	14	38	26	22	10	66	34	18	2	68	36	12	20	4	8
	128	**124**	**120**	**127**	**126**	**112**	**96**	64	119	118	116	108	125	123	122	111
	110	92	104	95	94	88	60	80	63	62	56	48	32	121	117	115
	114	109	107	106	103	102	100	93	91	90	87	86	84	79	78	61
At 14.26 dB	59	58	76	55	54	52	72	47	46	44	40	31	30	28	24	16
	113	105	101	99	98	89	85	83	82	77	75	74	57	53	71	51
	50	70	45	43	42	68	39	29	38	27	26	36	23	22	15	20
	14	97	81	73	12	49	69	41	67	8	37	66	25	35	21	34
	19	13	18	11	10	7	65	6	33	4	17	9	5	3	2	1

Table 5.2: Polar code lattice designs under SC decoding and $P_e = 10^{-4}$.

	$n = 64$	$n = 128$	$n = 256$	$n = 512$	$n = 1024$
k_0	1	7	24	68	178
k_1	40	88	192	410	866
k_2	64	128	256	512	1024
SNR_2 dB	20.03 dB	20.26 dB	20.47 dB	20.68 dB	20.87 dB

choosing the number of CRC bits that balance the trade-off between reliability and code rate.

Compare with the SC decoding, density evolution is not practical under SCL decoding due to list decoding. Accordingly, SCL decoding requires the use of Monte Carlo simulations which is beyond the scope of this thesis. Over the equivalent AMGN channel, Monte Carlo simulation results give the number of information bits K_0 and K_1 that achieve SNR of 8.26 dB and 14.26 dB, respectively.

[16] obtains the $\rho(\sigma^2, 10^{-4})$ function by Monte Carlo simulation under SCL decoder with 10 CRC bits and decoder list size 8. The curve is shown in Fig. 5.4. The designed lattice error probability $P_e = 3 \cdot 10^{-4}$. By Equ. 5.6, SNR_2 dB corresponds to 19.98 dB. From Fig. 5.4, we obtain the designs of $k_0 = 7$ at 7.85 dB and $k_1 = 95$ at 13.82 dB.

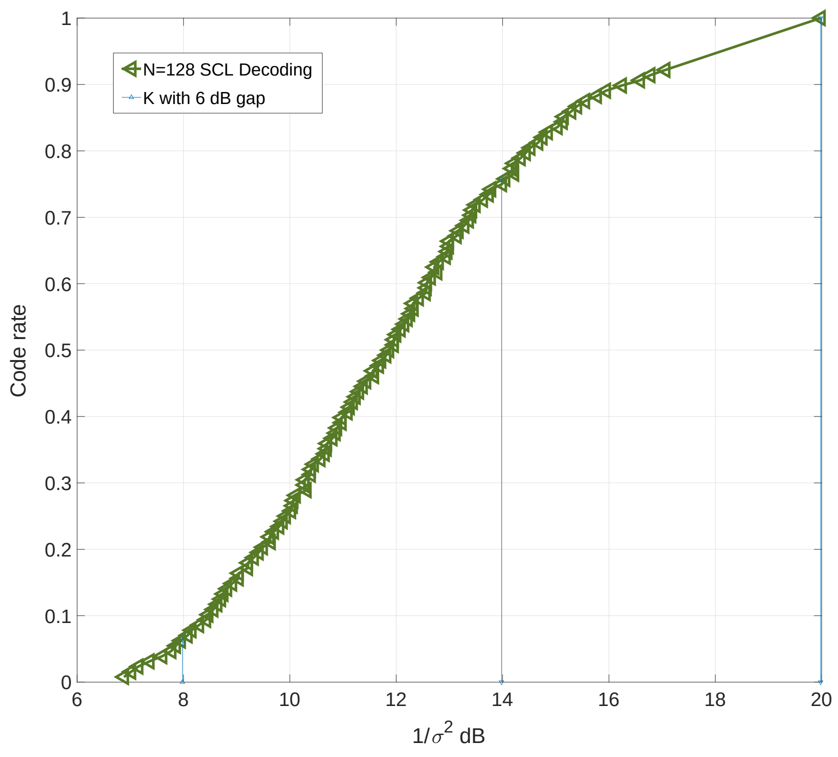


Figure 5.4: On a AMGN channel with noise σ^2, the function $\rho(\sigma^2, P_{\text{trgt}})$ gives a code rate to achieve SCL(CRC-10, Listsize-8) decoder $P_{\text{trgt}} = 10^{-4}$.

5.3.3 Polar Code Lattices Under OSD

Under OSD, we continue selecting the set of information bits $\mathcal{I}_i$ using Table. 5.1. To obtain the number of information bis K_0 and K_1 on each level, similarly, we use Monte Carlo simulation at the designed lattice word error rate WER of $P_{\text{e}} = 10^{-4}$. Due to the complexity of OSD, K_0 is a small value so that we can use a high order 4 to decode. For K_1, to obtain a better performance with low complexity, the order is set to 3. We perform simulations for different lengths of information bits on each level. In the first level, the simulation result is shown in Fig. 5.5. To achieve $P_{\text{trgt}} = \frac{1}{3} \cdot 10^{-4}$ at 8.26 dB, $K_0 = 7$ are selected for the first level. Similarly, for the second level, due to the simulation result in Fig.5.6, $K_1 = 97$ achieves $P_{\text{trgt}} = \frac{1}{3} \cdot 10^{-4}$ at 14.26 dB

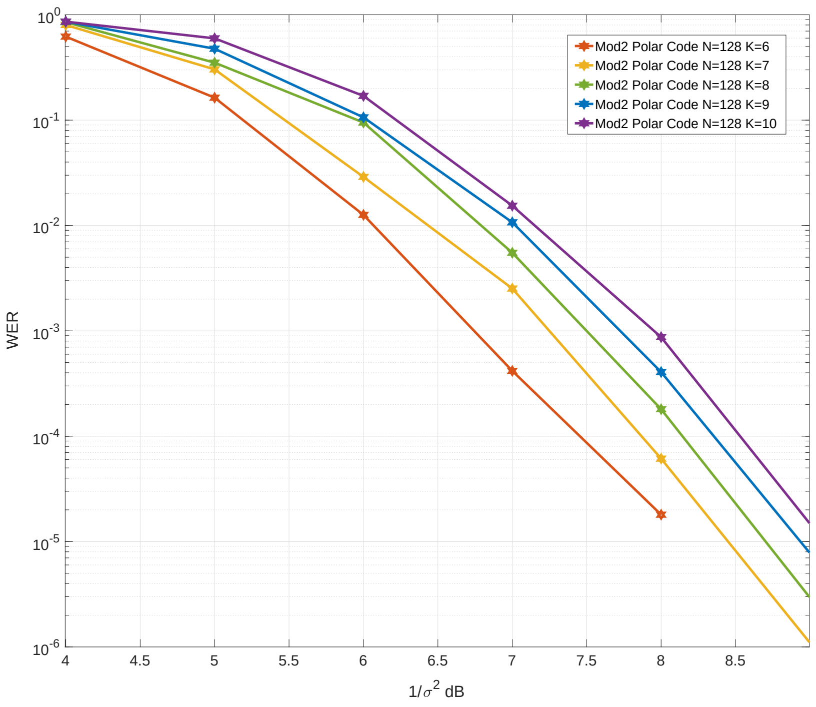


Figure 5.5: On a AMGN channel with noise σ^2, $\mathcal{P}(128, K_0, \mathcal{F})$ polar code using OSD(4) with different K_0 to achieve $P_{\text{trgt}} = \frac{1}{3} \cdot 10^{-4}$.

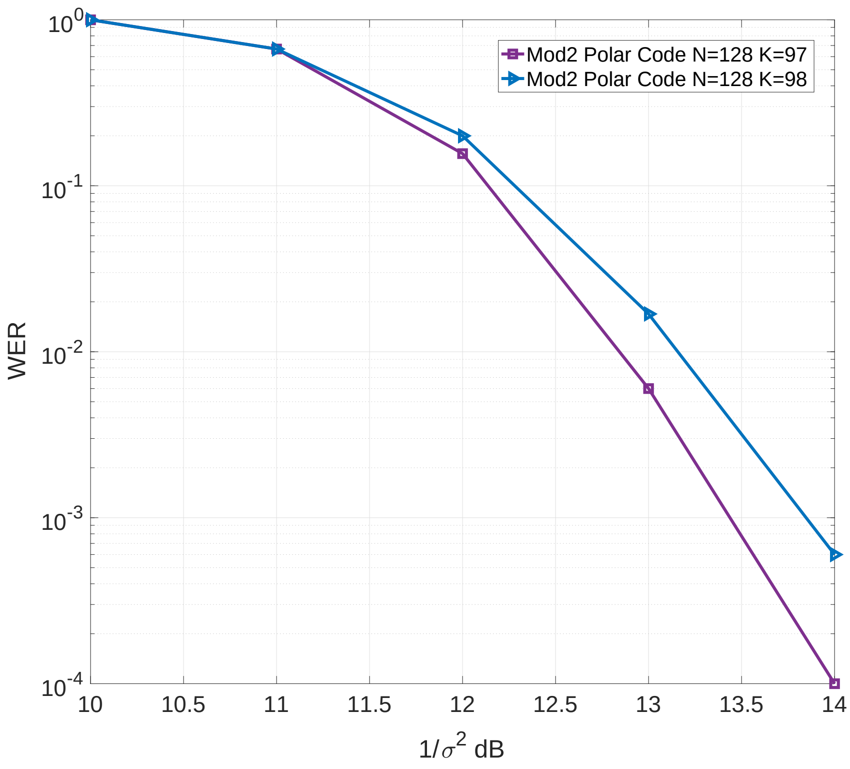


Figure 5.6: On a AMGN channel with noise σ^2, $\mathcal{P}(128, K_1, \mathcal{F})$ polar code using OSD(3) with different K_1 to achieve $P_{\text{trgt}} = \frac{1}{3} \cdot 10^{-4}$.

5.4 Evaluation by Simulation

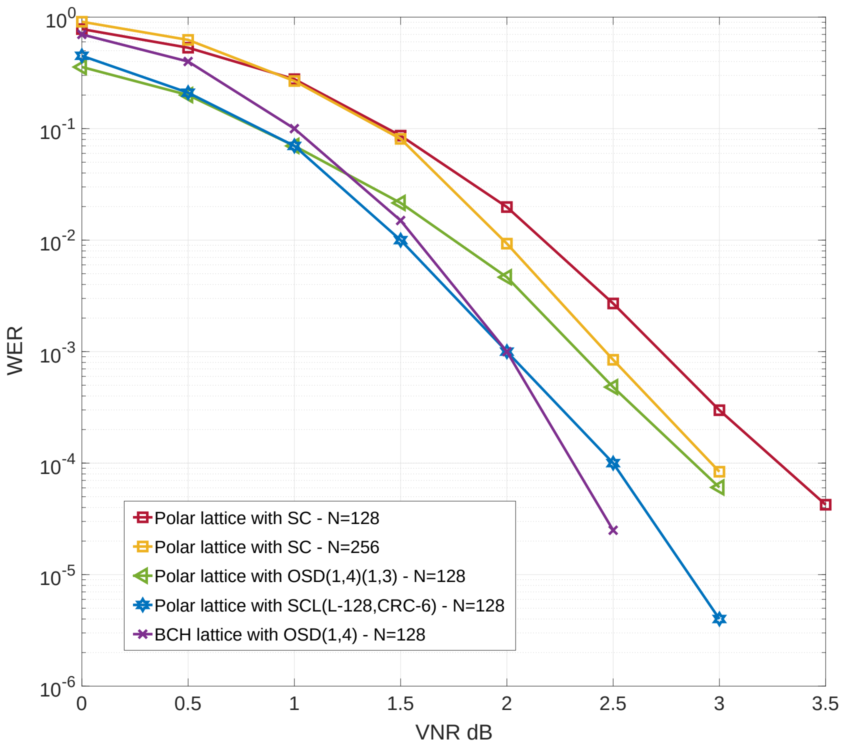


Figure 5.7: WER comparison between $N = 128$ BCH code lattice with OSD decoding and $N = 128, 256$ polar code lattices with SC decoding, OSD and SCL decoding.

We evaluated these polar code lattices with their respective decoders by simulation, WER is shown in Fig. 5.7. In summarize:

- Under SC decoding, at WER of $P_e = 10^{-4}$, dimension $n = 128$ polar code lattice with $K_0 = 7$ and $K_1 = 88$ achieves a VNR of 3.25 dB.

  The $n = 256$ polar code lattice with $K_0 = 24$ and $K_1 = 192$ achieves a VNR of 3.0 dB under similar conditions.

- Under SCL decoding, polar code lattice with $K_0 = 7$, $K_1 = 95$ achieves a VNR of 2.5 dB with 6 CRC bits and list size $L = 128$ at WER of $P_e = 10^{-4}$.

- Under OSD, polar code lattice with $K_0 = 7$ and $K_1 = 97$ using order OSD(4) and OSD(3) achieves a VNR of about 2.75 dB.

- For reference [17], the WER of the $(128, 120, 4)$, $(128, 78, 16)$ BCH code lattice with order $(1, 4)$ OSD decoding achieves a VNR of 2.3 dB, at a WER $P_e =$

10^{-4}.

Table 5.3: Performance Comparison of Dimension $N = 128$ lattices.

Code	Decoder	VNR at 10^{-4}	Time complexity	Remark	(K_0, K_1)
Polar lattice	SC	3.25 dB	$O(N \log N)$	-	$(7, 88)$
Polar lattice	SCL	2.50 dB	$O(LN \log N)$	CRC-6 L=128	$(7, 95)$
Polar lattice	OSD	2.75 dB	Proportional to $\sum_{i=0}^{l} \binom{K}{i}$	l=(4,3)	$(7, 97)$
BCH lattice	OSD	2.30 dB		l=(4,1)	$(78, 120)$

Chapter 6

Conclusions and future work

6.1 Conclusions

In this work, we construct polar code lattices of moderate dimension using Construction D by nested binary codes. Each component code is a binary polar code such that the lattice can be decoded by binary code decoder. We considered SC decoding, SCL decoding and OSD on each level to decode lattices. We give $N = 128$, $a = 2$ polar code lattices as design examples.

Under error probability rule, function ρ expresses the greatest rate which achieves a target word error rate of lattice P_{e}. To find the function ρ, under SC decoding, density evolution can be used efficiently. However, under SCL decoding and OSD, density evolution is not feasible, we use Monte Carlo simulation to obtain the function ρ.

Under SC decoding with complexity $O(\mathrm{N}\log N)$, polar code lattice comes within 1 dB of the BCH code lattice. Under SCL decoding with L=128 and CRC-6, polar code lattice comes within 0.2 dB of the BCH code lattice. SCL decoding with list size L, complexity scales as $O(\mathrm{LN}\log N)$. OSD decoding of BCH lattices has significantly higher complexity. The complexity of order-l OSD is proportional to $\sum_{i=0}^{l}\binom{K}{i}$.

For polar code lattice under OSD, there is still 0.45 dB gap between BCH code lattice. Increasing the value of the order from 3 to 4 for decoding K_1 may be a slight performance improvement, but it also dramatically increases decoding complexity. Thus, we decide to use OSD(3) to decode on the second level.

6.2 Future Work

In current work, we propose design of polar code lattices, better performances are still expected. For the OSD, one possible way to improve the performance is to increase the order of OSD in the first layer. Since the length of information bits is not too long, it is reasonable to consider using order-5 or higher order. Or achieve a better performance with low complexity on each layer by using improved OSD.

We also consider to design Construction D polar code lattice by balanced distance rule. Polar code with some special distances form Reed-Muller codes [20] so that we can also use good decoder for Reed-Muller to decode lattices.

Another idea is to increase the dimension of lattice that may also improve the performance.

Bibliography

[1] C. E. Shannon, "A mathematical theory of communication," *The Bell System Technical Journal*, vol. 27, no. 3, pp. 379–423, 1948.

[2] E. Arikan, "Channel polarization: A method for constructing capacity-achieving codes for symmetric binary-input memoryless channels," *IEEE Transactions on Information Theory*, vol. 55, no. 7, pp. 3051–3073, 2009.

[3] B. Nazer and M. Gastpar, "Compute-and-forward: Harnessing interference through structured codes," *IEEE Transactions on Information Theory*, vol. 57, no. 10, pp. 6463–6486, 2011.

[4] J. Harshan, A. Sakzad, and E. Viterbo, "Integer-forcing linear receivers: A design criterion for full-diversity stbcs," in *2017 IEEE Wireless Communications and Networking Conference (WCNC)*, 2017, pp. 1–6.

[5] L. Liu, Y. Yan, C. Ling, and X. Wu, "Construction of capacity-achieving lattice codes: Polar lattices," *IEEE Transactions on Communications*, vol. 67, no. 2, pp. 915–928, 2019.

[6] J. H. Conway and N. J. A. Sloane, *Sphere packings, lattices and groups.* Springer Science & Business Media, 2013, vol. 290.

[7] E. S. Barnes and N. J. A. Sloane, "New lattice packings of spheres," *cjm*, vol. XXXV, no. 1, pp. 117–130, 1983.

[8] U. Wachsmann, R. F. Fischer, and J. B. Huber, "Multilevel codes: theoretical concepts and practical design rules," *it*, vol. 45, no. 5, pp. 1361–1391, Jul. 1999.

[9] R. Mori and T. Tanaka, “Performance of polar codes with the construction using density evolution,” *IEEE Communications Letters*, vol. 13, no. 7, pp. 519–521, 2009.

[10] I. Tal and A. Vardy, “List decoding of polar codes,” in *2011 IEEE International Symposium on Information Theory Proceedings*, 2011, pp. 1–5.

[11] O. Afisiadis, A. Balatsoukas-Stimming, and A. Burg, “A low-complexity improved successive cancellation decoder for polar codes,” in *2014 48th Asilomar Conference on Signals, Systems and Computers*, 2014, pp. 2116–2120.

[12] K. Niu and K. Chen, “Stack decoding of polar codes,” *Electronics Letters*, vol. 48, pp. 695–697, 2012.

[13] M. Fossorier and S. Lin, “Soft decision decoding of linear block codes based on ordered statistics,” in *Proceedings of 1994 IEEE International Symposium on Information Theory*, 1994, pp. 395–.

[14] D. Wu, Y. Li, X. Guo, and Y. Sun, “Ordered statistic decoding for short polar codes,” *IEEE Communications Letters*, vol. 20, no. 6, pp. 1064–1067, 2016.

[15] K. Niu and K. Chen, “Crc-aided decoding of polar codes,” *IEEE Communications Letters*, vol. 16, no. 10, pp. 1668–1671, 2012.

[16] O. R. Ludwiniananda, N. Liu, K. Anwar, and B. M. Kurkoski, “Design of polar code lattices of finite dimension,” in *2021 IEEE International Symposium on Information Theory (ISIT)*, 2021, pp. 1011–1016.

[17] T. Matsumine, B. M. Kurkoski, and H. Ochiai, “Construction D lattice decoding and its application to BCH code lattices,” in *2018 IEEE Global Communications Conference (GLOBECOM)*, 2018, pp. 1–6.

[18] P. R. Branco da Silva and D. Silva, “Multilevel LDPC lattices with efficient encoding and decoding and a generalization of construction D’,” *it*, vol. 65, no. 5, pp. 3246–3260, 2019.

[19] T. Murata and H. Ochiai, “On design of CRC codes for polar codes with successive cancellation list decoding,” in *isit*. Aachen, Germany: IEEE, Jun. 2017, pp. 1868–1872.

[20] D. E. Muller, “Application of boolean algebra to switching circuit design and to error detection,” *Transactions of the I.R.E. Professional Group on Electronic Computers*, vol. EC-3, no. 3, pp. 6–12, 1954.