UNIVERSITY OF CALIFORNIA, SAN DIEGO

New Frontiers in Polar Coding:
Large Kernels, Convolutional Decoding, and Deletion Channels

A dissertation submitted in partial satisfaction of the
requirements for the degree
Doctor of Philosophy

in

Electrical Engineering (Communication Theory and Systems)

by

Arman Fazeli Chaghooshi

Committee in charge:

Professor Alexander Vardy, Chair
Professor Young-Han Kim
Professor Daniel S. Rogalski
Professor Nambirajan Seshadri
Professor Paul H. Siegel

2018

The dissertation of Arman Fazeli Chaghooshi is approved, and it is acceptable in quality and form for publication on microfilm and electronically:

Chair

University of California, San Diego

2018

EPIGRAPH

The heavenly breeze comes to this estate,
I sit with the wine and a lovely mate.

Why can't the beggar play the king's role?
The sky is the dome, the earth is my state.

The green grass feels like Paradise;
Why would I trade this for the garden gate?

With bricks of wine build towers of love,
Being bricks of clay is our final fate.

Seek no kindness of those full of hate,
People of the mosque with the church debate.

Don't badmouth me, don't blacken my name;
Only God can, my story narrate.

Neither Hafiz's corpse, nor his life negate,
With all his misdeeds, heavens for him wait.

Khwaja Shams-ud-Din Muhammad Hafez-e Shirazi, Ghazal 79
Translated to English by Shahriar Shahriari

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGEMENTS

There are several individuals who made it possible for me to complete my doctoral research. First, I would like to express my sincere gratitude to my academic advisor, Alexander Vardy, who expertly mentored me through six years of graduate school and who taught me the correct way of thinking, conducting research, and presenting my ideas. I still remember the beginning of our journey together when he always had research problems waiting for me if I needed a new one and was always open to spend multiple hours of brainstorming with me if I needed help. He always encouraged me to also find the research problems of my own, which ultimately gave birth to some of the research topics presented here. He is also my role model and has taught me the lessons of life. This dissertation would have not been possible if it wasn't for his continued support for which I will always be indebted.

My appreciation also extends to my favorite teacher, Paul H. Siegel, who was the first person to introduce me into the field of coding theory. I found my deep desire to study coding theory from the first session of Paul's class, which grew more and more every time I attended his lectures. His lecture notes are my bible of coding theory. I am also thankful to my other committee members, Young-Han Kim, Daniel S. Rogalski, and Nambirajan Seshadri for their invaluable recommendations on my research and sharing their vision of future with me.

I was lucky to participate in multiple collaborative projects and to have many amazing co-authors over the past few years. I am thankful to Eitan Yaakobi and Sreechakra Goparaju in particular, who not only taught me a lot but were also great friends. They never shied away from helping me to understand and solve the problems and were always very considerate of me when the research was going slower than usual. I am a better researcher because of them.

I am also thankful to Tara Javidi, who not only was a great teacher to me but also believed in me more than I did myself. While the turn of events split our research paths from

each other, I am always thankful to her for helping me get accepted to the PhD program in Department of Electrical and Computer Engineering at UC San Diego and for offering me the first-year fellowship along with it. Not everyone has the opportunity to search for their passion from the beginning of their graduate studies without worrying about financial hardships.

Being away from my mom and dad for these many years was hard. But, what made it unbearable was to watch them grow old away from their only child. It was my mom who showed me the meaning of unconditional love. She was always there for me without me asking for her and without me knowing that I need her. My dad also taught me to work hard for what I believe in. He is, and will always be my hero in life. It is simply impossible to express my level of gratitude to them over a few sentences. But, I can say that they are my true stars and my only dream is to get reunited with them in near future.

A wise person once said that a good teacher can inspire hope, ignite the imagination, and instill a love of learning. I ask what about good friends? And the answer is that they instill a desire for living. I believe that there is only one happiness in this life, which is to love and be loved. Elina gave me that. She is my best friend and my partner in crime. I am thankful to her for continuously and patiently supporting me in every phase of the work on this dissertation. She has been my inspiration and motivation to move my career forward and I am a better person because of her.

Chapter 2 contains materials as it appear in [4], A. Fazeli and A. Vardy, “On the scaling exponent of binary polarization kernels," *Proceedings of IEEE 52nd Allerton Conference on Communication, Control, and Computing*, Sep. 2014, pp. 797-804, and materials that appear in [5], S. Buzaglo, A. Fazeli, P. H. Siegel, V. Taranalli, and A. Vardy, “On efficient decoding of polar codes with large kernels," *Proceedings of IEEE Wireless Communications and Networking Conference Workshops*, Mar. 2017, pp. 1-6. It is also, in part, a reprint of [6], A. Fazeli, S. H. Hassani, M. Mondelli, and A. Vardy, “Binary linear codes with optimal scaling: polar codes with large kernels," submitted to *IEEE Transactions on Information Theory*, available online at arXiv:1711.01339, and [7], Permuted successive cancellation decoding for polar codes," *Proceedings of IEEE International Symposium on Information Theory*, Jun. 2017, pp. 2618-2622. The dissertation author was the primary investigator and author of these papers.

Chapter 3 contains materials as it appears in [8], A. Fazeli, K. Tian, and A. Vardy, “Viterbi-Aided Successive-Cancellation Decoding of Polar Codes," *Proceedings of IEEE Global Communications Conference*, Dec. 17, pp. 1-6. This chapter, in part, contains materials from the paper in preparation, A. Fazeli, K. Tian, and A. Vardy, “Convolutional decoding of polar codes", to be submitted to *IEEE Transactions on Information Theory*. The dissertation author was the primary investigator and author of these papers.

Chapter 4, in part, contains materials from the paper K. Tian, A. Fazeli, and A. Vardy, “Polar coding for channels with deletion," that is submitted to *IEEE Transactions on Information Theory*. The dissertation author was the primary investigator and author of this paper.

VITA

2012	Bachelor of Science in Electrical Engineering (Telecommunications), Sharif University of Technology, Iran
2016	Master of Science in Electrical and Computer Engineering (Communication Theory and Systems), University of California, San Diego
2018	Doctor of Philosophy in Electrical and Computer Engineering (Communication Theory and Systems), University of California, San Diego

PUBLICATIONS

S. Goparaju, **A. Fazeli**, and A. Vardy, “Minimum Storage Regenerating Codes For All Parameters," *IEEE Transactions on Information Theory*, vol. 63, no. 10, pp. 6318-6328, October 2017.

A. Fazeli, A. Vardy, and E. Yaakobi, “The Generalized Sphere Packing Bound," *IEEE Transactions on Information Theory*, vol. 61, no. 5, pp. 2313-2334, March 2015 .

A. Fazeli, S. Lovett, and A. Vardy, “Nontrivial t-designs over finite fields exist for all t," *Journal of Combinatorial Theory, Series A*, vol. 127, pp. 149-160, September 2014.

A. Fazeli, A. Vardy, and E. Yaakobi, “Codes for Distributed PIR with Optimal Storage Overhead,” in preparation for submission to *IEEE Transactions on Information Theory*.

A. Fazeli, K. Tian, and A. Vardy, “Convolutional List Decoding of Polar Codes," in preparation for submission to *IEEE Transactions on Information Theory*.

K. Tian, **A. Fazeli**, and A. Vardy, “Polar Codes for Deletion Channel," submitted to *IEEE Transactions on Information Theory*.

A. Fazeli, S. H. Hassani, M. Mondelli, and A. Vardy, “Binary Linear Codes with Optimal Scaling: Polar Codes with Large Kernels," submitted to *IEEE Transactions on Information Theory*.

K. Tian, **A. Fazeli**, and A. Vardy, “Polar Coding for Deletion Channels: Theory and Implementation," to appear in *Proceedings of IEEE International Symposium on Information Theory*, June 2018.

A. Fazeli, K. Tian, and A. Vardy, “Viterbi-Aided Successive-Cancellation Decoding of Polar Codes," *Proceedings of IEEE Global Communications Conference*, pp. 1-6. December 2017.

K. Tian, **A. Fazeli**, A. Vardy, and R. Liu, “Polar Codes for Channels with Deletions," *Proceedings of IEEE Allerton Conference on Communication, Control and Computing*, pp. 572-579, October 2017.

S. Buzaglo, **A. Fazeli**, P. H. Siegel, V. Taranalli, and A. Vardy, “Permuted successive cancellation decoding for polar codes,” *Proceedings of IEEE International Symposium on Information Theory*, pp. 2618-2622, June 2017.

S. Buzaglo, **A. Fazeli**, P. H. Siegel, V. Taranalli, and A. Vardy, “On Efficient Decoding of Polar Codes with Large Kernels,” *Proceedings of IEEE Wireless Communications and Networking Conference Workshops*, pp. 1-6, March 2017.

A. Fazeli, S. Goparaju, and A. Vardy, “Minimum Storage Regenerating Codes For All Parameters," *Proceedings of IEEE International Symposium on Information Theory*, pp. 76-80, July 2016.

A. Fazeli, A. Vardy, and E. Yaakobi, “Codes for Distributed PIR with Optimal Storage Overhead,” *Proceedings of IEEE International Symposium on Information Theory*, pp. 2852-2856, June 2015.

A. Fazeli and A. Vardy, “On the scaling exponent of binary polarization kernels," *Proceedings of IEEE 52nd Allerton Conference on Communication, Control, and Computing*, pp. 797-804, September 2014.

A. Fazeli, A. Vardy, and E. Yaakobi, “The Generalized Sphere Packing Bound: Basic Principles,” *Proceedings of IEEE International Symposium on Information Theory*, pp. 1256-1260, June 2014.

A. Fazeli, A. Vardy, and E. Yaakobi, “The Generalized Sphere Packing Bound: Applications,” *Proceedings of IEEE International Symposium on Information Theory*, pp. 1261-1265, June 2014.

Z. Shakeri, **A. Fazeli**, M. Mirmohseni, and M. R. Aref, “Degrees of Freedom in a Three-User Cognitive Interference Channel." *Proceedings of Iran Workshop in Communication and Information Theory*, pp. 1-6, May 2013.

FIELDS OF STUDY

Major Field: Electrical Engineering

Studies in Communication Theory and Systems

Advisor: Alexander Vardy

ABSTRACT OF THE DISSERTATION

New Frontiers in Polar Coding:
Large Kernels, Convolutional Decoding, and Deletion Channels

by

Arman Fazeli Chaghooshi

Doctor of Philosophy in Electrical Engineering (Communication Theory and Systems)

University of California, San Diego, 2018

Professor Alexander Vardy, Chair

The discovery of channel polarization and polar codes is universally recognized as an historic breakthrough in coding theory. Polar codes provably achieve the capacity of *any* memoryless symmetric channel, with low encoding and decoding complexity. Moreover, for short block lengths, polar codes under specific decoding algorithms are currently the *best known coding scheme* for binary-input Gaussian channels [9]. Due to this and other considerations, 3GPP has recently decided to incorporate polar codes in the 5G wireless communications standard. Soon enough, a remarkably short time after their invention, we will be all using polar codes whenever we make a phone call or access the Internet on a mobile device.

Our goal in this dissertation is to explore new frontiers in polar coding, thereby fundamentally advancing the current state-of-the-art in the field. Parts of the results are immediately relevant for successful deployment of polar codes in wireless systems, whereas other parts will focus on key theoretical problems in polar coding that have a longer time-horizon.

We begin by studying the effect of the polarization kernels in the asymptotic behavior of polar codes. We show that replacing the conventional 2×2 kernel in the construction of polar codes with that of a larger size can reduce the gap to the capacity if the larger kernel is carefully selected. A heuristic algorithm is proposed that helps to find such kernels. Furthermore, we prove that a near-optimal scaling behavior is achievable if one is allowed to increase the kernel size as needed. We also study the computational complexity of decoding algorithms for polar codes with large kernels, which are viewed as their main implementation obstacle.

Moving on to the decoding algorithms, we carefully analyze the performance of the successive cancellation decoder with access to the abstract concept of Arıkan's genie. The CRC-aided successive-cancellation list decoding, the primary decoding method of polar codes, is commonly viewed as an implementation of the Arıkan's genie. However, it comes short at completely simulating the genie since the auxiliary information (CRC) comes to the help only at the end of the decoding process. We overcome this problem by introducing the convolutional decoding algorithm of polar codes that is based on a high-rate convolutional pre-coder and utilizes Viterbi Algorithm to mimic the genie all the way through the SC decoding process.

Lastly, we look into channels with deletions. A key assumption in the traditional polar coding is to transmit coded symbols over independent instances of the communication channel. Channels with memory and in particular, deletion channels, do not follow this rule. We introduce a modified polar coding scheme for these channels that depend on much less computational power for decoding than the existing solutions. We also extend the polarization theorems to provide theoretical guarantee and to prove the correctness of our algorithms.

Chapter 1

Introduction

1.1 Background on Error-Correcting Codes

C. Shannon [1] defined the fundamental problem of communication as that of *producing at one point either exactly or approximately a message selected at another point.* As depicted in Figure 1.1, the mathematical model of a communication generally consists of:

1. An *information source*, Alice, that produces a finite or infinite sequence of messages to be communicated to the receiving party.
2. A *transmitter*, which performs encoding on the message to produce a suitable (resistance to noise) signal for transmission over the underlying communication channel.
3. A *channel*, that is the medium used to transmit signal from transmitter to receiver and usually suffers from one or multiple sources of distortion (noise).
4. A *receiver*, that is designed to invert the action of the encoder and hence reconstruct the original message.
5. A *destination*, Bob, which is the party for whom the message was intended.

An encoding module in a coding scheme generates symbols $x_1, x_2, \cdots$ to be transmitted over the channel. Upon passing the channel distortion, a possibly different set of symbols

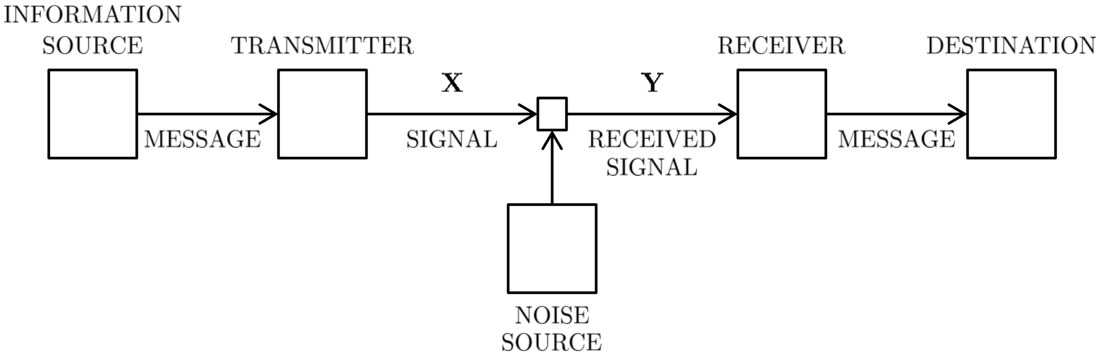


Figure 1.1: Schematic diagram of a general communication system [1].

$y_1, y_2, \cdots$ are received at the receiver, which are then fed to the decoding module. The decoding module is responsible for estimating the original sequence with least amount of mistakes.

Let X and Y be the random variables that represent the input and output of the underlying communication channel with their conditional distribution $P_{Y|X}(y|x)$ to be inherited from the channel. The channel capacity is defined as

$$C \triangleq \sup_{p_X(x)} I(X;Y) \tag{1.1}$$

is proven to be the highest information rate that can be achieved with diminishing error probabilities and is measured in units of information per unit of time. Here, $I(X;Y)$ denotes the mutual information between X and Y. The capacity of channel W is a parameter of the channel and can be calculated based on its conditional probabilities. For example, the capacity of an additive white Gaussian noise (AWGN) channel with B Hz bandwidth and signal-to-noise ratio ${}^{S}/_{N}$ is given by

$$C = B\log_2\left(1 + \frac{S}{N}\right), \tag{1.2}$$

which shows that the communication channel has a higher capacity for larger SNRs.

The encoding function $\mathcal{E} : \{m_1, m_2, \cdots, m_M\} \to \mathcal{X}^n$ is defined as a mapping from the set of M messages provided by Alice to elements in $\mathcal{X}^n$, where $\mathcal{X}$ denotes the input alphabet of the channel. The decoding function $\mathcal{D} : \mathcal{Y}_1^n \to \{m_1, m_2, \cdots, m_M\}$ is similarly defined as a mapping from the received symbols back to the set of all possible messages. Here, $\mathcal{Y}$ denotes channel's output alphabet. The rate and error probability of this coding scheme are defined as

$$R \triangleq \frac{\log_{|\mathcal{X}|}(|M|)}{n}, \qquad \text{and} \qquad P_e \triangleq \mathbb{E}_i\left[\text{Pr.}\Big(\mathcal{D}\big(\mathcal{E}(m_i)\big) \neq m_i\Big)\right]. \tag{1.3}$$

Theorem 1. *(Channel coding) Given a noisy channel W with capacity $C(W)$ and an information transmission rate $R < C(W)$, there exists a family of codes $\{\mathcal{C}_i\}_{i=1}^{\infty}$ with*

$$\textit{rate}(\mathcal{C}_i) = r_i, \qquad \textit{code-length}(\mathcal{C}_i) = n_i, \qquad \textit{and} \qquad P_e(\mathcal{C}_i) = P_i \quad \forall i, \tag{1.4}$$

where $r_i \leqslant R$ $(\forall i)$, and for any given $\epsilon > 0$ there exists a κ_ϵ such that $P_i \leqslant \epsilon$ $(\forall i \geqslant \kappa_\epsilon)$. Moreover, the channel capacity $C(W)$ is the smallest number with such property (tight bound).

While the Shannon proved the existence of capacity achieving codes, an explicit construction of such codes remained unsolved for almost half a decade. Indeed, E. Arıkan was the first to explicitly construct a family of capacity achieving codes in his nominal paper [10], which are today known as polar codes. Prior to the invention of polar codes, different families of codes were usually compared with each other based on their practicality (encoding/decoding complexity) and their finite-length error performances. Some of the most notable code constructions include *algebraic codes* such as Reed-Solomon (RS) or Reed-Muller (RM) codes, *convolutional codes*, *Turbo codes*, *low-density parity-check* (LDPC) codes, or the more recently discovered *spatially-coupled* codes. Figure 1.2 taken from [2] illustrates the spectral efficiencies achieved by some these coded communication schemes.

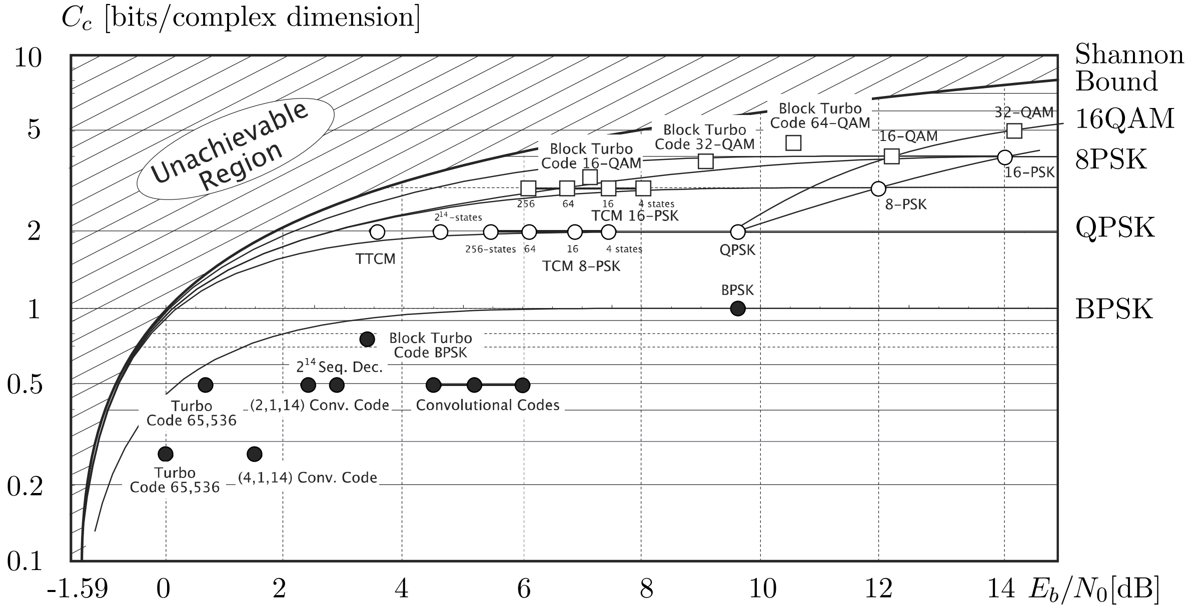


Figure 1.2: Theoretical spectral and power efficiency limits for various signal constellations and spectral efficiencies achieved by multiple coded communication schemes as appears in [2].

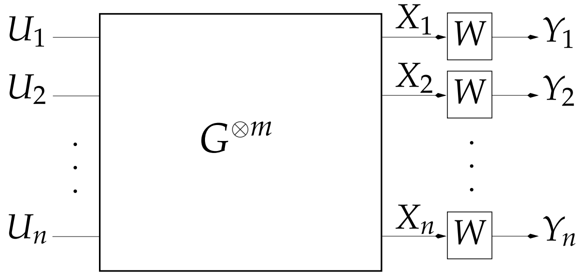
1.2 Polar Coding: Overview

The invention of polar codes by Arıkan [10] is undoubtedly one of the most original and profound developments in coding theory to date. Polar codes achieve the capacity of any memoryless symmetric channel, with low encoding and decoding complexity. Nevertheless, when polar coding was first discovered, it was widely regarded as being of mostly theoretical interest, since major obstacles prevented the utilization of polar codes in practice. However, only seven years later, at its November 2016 meeting, 3GPP has voted to adopt polar codes in the 5G wireless standard.

Like many fundamental discoveries, polar codes are rooted in a simple and beautiful basic idea. Polarization is induced via a simple linear transformation consisting of many Kronecker products of a small binary matrix G, called the *polarization kernel*, with itself. Following Arıkan [10], we take

$$G \triangleq \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}, G^{\otimes 2} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 1 \end{bmatrix}, \quad \cdots \quad, \ G^{\otimes m} \overset{\text{def}}{=} m\text{-th Kronecker power of } G \tag{1.5}$$

Let $W\colon \{0,1\} \to \mathcal{Y}$ be a binary memoryless symmetric (BMS) channel, characterized in terms of its transition probabilities $W(y|x)$, for all $y \in \mathcal{Y}$ and $x \in \{0,1\}$. Further let $\mathbf{U} = (U_1, U_2, \ldots, U_n)$ be a block of $n = 2^m$ bits chosen uniformly at random from $\{0,1\}^n$. We encode $\mathbf{U}$ as $\mathbf{X} = \mathbf{U}G^{\otimes m}$ and transmit $\mathbf{X}$ through n independent copies of W, as shown by



(1.6)

This, in a nutshell, is the *polarization transformation* of Arıkan [10]. To understand what polarization means in this context, let us consider a number of channels associated with the transformation in (2.3). First, there is the channel $W^n\colon \{0,1\}^n \to \mathcal{Y}^n$ that corresponds to n independent uses of W. Arıkan [10] also introduces the channel $W_n\colon \{0,1\}^n \to \mathcal{Y}^n$ with transition probabilities given by $W_n(\boldsymbol{y}|\boldsymbol{u}) = W^n\big(\boldsymbol{y}\,\big|\,\boldsymbol{u}G^{\otimes m}\big)$. Finally, and most importantly, Arıkan [10] further defines, for each $i = 1, 2, \ldots, n$, the channel $W_i : \{0,1\} \to \mathcal{Y}^n \times \{0,1\}^{i-1}$ that is "seen" by the bit U_i, as follows:

$$W_i\big(\boldsymbol{y},\boldsymbol{v}|u_i\big) \;\triangleq\; \frac{1}{2^{n-1}} \sum_{\overline{\boldsymbol{u}}\in\{0,1\}^{n-i}} W_n\Big(\boldsymbol{y}\,\big|\,(\boldsymbol{v},u_i,\overline{\boldsymbol{u}})\Big) \;=\; \frac{1}{2^{n-1}} \sum_{\overline{\boldsymbol{u}}\in\{0,1\}^{n-i}} W^n\Big(\boldsymbol{y}\,\big|\,(\boldsymbol{v},u_i,\overline{\boldsymbol{u}})G^{\otimes m}\Big) \quad (1.7)$$

where $(\cdot,\cdot)$ stands for vector concatenation. It is easy to show that $W_i\big(\boldsymbol{y},\boldsymbol{v}|u_i\big)$ is indeed the probability of the event that $(Y_1, Y_2, \ldots, Y_n) = \boldsymbol{y}$ and $(U_1, U_2, \ldots, U_{i-1}) = \boldsymbol{v}$ given the event $U_i = u_i$.

The key observation of [10] is that, as n grows, the n *bit-channels* W_i defined in (2.4) start polarizing: they approach either a noiseless channel or a useless channel. Formally, given a BMS channel W, its *capacity* $I(W)$ and *Bhattacharyya parameter* $Z(W)$ are given by

$$I(W) \;\triangleq\; \frac{1}{2}\sum_{y\in\mathcal{Y}}\sum_{x\in\{0,1\}} W(y|x)\log_2\frac{W(y|x)}{\frac{1}{2}W(y|0)+\frac{1}{2}W(y|1)}\;; \qquad Z(W) \overset{\text{def}}{=} \sum_{y\in\mathcal{Y}}\sqrt{W(y|0)W(y|1)}.$$

Given a constant $\delta \in (0,1)$, let us say that a bit-channel W_i is δ-*good* if $I(W_i) \geqslant 1-\delta$ and δ-*bad* if $I(W_i) \leqslant \delta$. Then the polarization theorem of Arıkan [10, Theorem 1] can be stated as follows.

Theorem 2. *For every $\delta \in (0,1)$, almost all bit-channels become either δ-good or δ-bad as $n \to \infty$. In fact, as $n \to \infty$, the fraction of δ-good bit-channels approaches the capacity $I(W)$ of the underlying channel W, while the fraction of δ-bad ones approaches $1 - I(W)$.*

Theorem 2 naturally leads to the construction of capacity-achieving *polar codes*. Specifically, an (n,k) polar code is constructed by selecting a set $\mathcal{A}$ of k good bit-channels to carry the information bits, while the input to all the other bit-channels is frozen to zeros. The code parameters k and δ are usually selected according to the target rate and/or the desired probability of error.

Note that the i-th bit-channel W_i is precisely the channel seen by the input bit U_i under a hypothetical decoder that works as follows: it attempts to estimate U_i, having observed the channel output $\boldsymbol{y}$ *and* the first $i-1$ bits of the input $(u_1, u_2, \ldots, u_{i-1}) = \boldsymbol{v}$. A beautiful idea of Arıkan [10] was to convert this hypothetical decoder into a real one by substituting the first $i-1$ decisions $(\widehat{u}_1, \widehat{u}_2, \ldots, \widehat{u}_{i-1})$ in place of the hypothetical observations $\boldsymbol{v}$. Thus, the *successive cancellation decoder* of [10] proceeds as follows: sequentially, for $i = 1, 2, \ldots, n$, set $\widehat{u}_i = 0$ if $i \in \mathcal{A}^c$ and

$$\widehat{u}_i = \begin{cases} 0 & \text{if } i \in \mathcal{A} \text{ and } W_i\big(\boldsymbol{y}, \widehat{u}_1, \widehat{u}_2, \ldots, \widehat{u}_{i-1} | 0\big) \geqslant W_i\big(\boldsymbol{y}, \widehat{u}_1, \widehat{u}_2, \ldots, \widehat{u}_{i-1} | 1\big) \\ 1 & \text{if } i \in \mathcal{A} \text{ and } W_i\big(\boldsymbol{y}, \widehat{u}_1, \widehat{u}_2, \ldots, \widehat{u}_{i-1} | 0\big) < W_i\big(\boldsymbol{y}, \widehat{u}_1, \widehat{u}_2, \ldots, \widehat{u}_{i-1} | 1\big) \end{cases} \tag{1.8}$$

Arıkan [10] furthermore shows how the recursive (FFT-like) structure of polar codes, originating from (1.5), can be used to accomplish the successive-cancellation decoding in (1.8) with complexity $O(n \log n)$ and latency $O(n)$. Today, many extremely efficient realizations of this decoder in custom VLSI are known [11–17].

Since its discovery by Arıkan [10] in 2009, polar coding has been applied to a wide range of fundamental problems in information theory. These include multiple-access channels [18–20], broadcast channels [21, 22], wiretap channels [23–26], source coding [27–29], and write-once memories [30]. There have been great many works on code construction, encoding, and decoding [31–39], system design and hardware implementation [11, 40–42], and finite-length analysis [3, 43–48]. Moreover, the channel polarization paradigm has been both generalized and extended well beyond the domain of information theory [49–51].

As this dissertation is being prepared, the fifth generation of wireless networks (5G) is being developed and standardized. Herein, we briefly review the latest developments concerning polar codes in this ongoing 5G standardization process. Note that the 5G standard has three major connectivity modes: enhanced mobile broadband (eMBB), machine-type communications (MTC), and ultra-reliable low-latency communications (URLLC). At its November 2016 meeting, 3GPP decided to adopt polar codes as the coding scheme for *eMBB control channels*, in both downlink and uplink. They will replace convolutional codes that have been used to code the control channels in 3G and 4G. Later, 3GPP has adopted polar codes for the *eMBB physical broadcast channels* (PBCH). Moreover, polar codes are currently among the candidates for coding the *main data channels* in the MTC and the URLLC connectivity modes. Indeed, a number of leading companies in the wireless sector view polar coding as one of the key new technologies introduced in the 5G standard.

Note that PBCH and control channels are used to initiate the communication between a base station and the user. Thus an error in decoding either PBCH or the control channel will render the entire communication useless. Consequently, the coding scheme for these channels must be *highly reliable*. Furthermore, a delay in decoding these channels will usually result in the loss of some data packets; consequently, the *decoder latency* must be extremely low. Finally,the requirements on the user side place stringent *hardware and power constraints* on the decoder. Discussions on the precise specification of polar-coding architectures for PBCH and control channels are still ongoing. Nevertheless, there is a general consensus to use polar codes *under list decoding* concatenated with an *outer cyclic redundancy check* (CRC) code, as originally was proposed in [39]. It is now almost certain that CRC-aided successive-cancellation list (SCL) decoders for polar codes will be part of 5G chipset modems.

Even upon the completion of the 5G standardization process, the particular encoding/decoding architectures to be deployed at the mobile devices will remain open. Several important challenges will have to be resolved in order to pave the way towards efficient system-on-chip polar codecs that meet the required performance targets.

1.3 Dissertation Contributions

The goal behind this research was to further advance the current state-of-the-art in polar coding. In this dissertation, we will investigate some of the challenges that must be overcome for a more successful implementation of polar codes in the communication systems. We will also introduce new directions in polar coding coming from the theoretical perspective, which lie beyond the horizon of todays wireless technology.

We begin by covering some preliminary materials about conventional polar coding. An overview of the channel polarization theorem is provided, which builds the foundation of polar coding. This is followed by a summary of the construction algorithms for polar codes. We also touch upon the successive cancellation decoding algorithm of polar codes, which becomes more relevant towards the end of the dissertation. Readers can find a detailed discussion on scaling properties of polar codes in this section in which we also provide our motivation behind the first topic of this dissertation.

As discussed earlier in (1.5), the conventional construction of polar codes is based on Kronecker powers of a 2×2 binary matrix, which is usually referred to by the Arıkan's polarization kernel. However, Korada *et. al.* [29] showed that the polarization theorems hold for almost any $\ell \times \ell$ binary kernel, as long as it is non-singular and cannot be transformed into an upper triangular matrix under column permutations. Moreover, a clever selection of such kernel can improve the finite-length scaling properties of polar codes. In particular, they constructed a 16×16 kernel with an *error exponent* greater than that of the Arıkan's kernel.

In Chapter 2, we continue this line of thinking by first looking for kernels larger than Arıkan's with smaller *scaling exponents*. The scaling exponent, denoted by μ, captures the speed of the polarization, or in general the speed of which the gap to the capacity decays as the code-length increases. Random linear codes are proven to achieve the optimal scaling exponent of $\mu = 2$. However, the scaling exponent of the conventional polar codes is only bounded as $3.579 \leqslant \mu_2 \leqslant 4.714$ for arbitrary channels and explicitly calculated as $\mu_2 = 3.626$ for the binary erasure channels. We found the first example of such kernels at $\ell = 8$ through brute force simulations with $\mu_8 = 3.577$ for when the underlying communication channel is limited to binary erasure channels. This is rather disappointing because of the following:

- The difference between μ_8 and μ_2 is negligible particularly for short block-lengths.
- It is computationally impossible to extend the brute force search beyond $\ell = 8$.
- And, replacing the 2×2 kernels with larger ones is not free. Indeed, it comes with a drastic increase in the decoding complexity.

In order to address these issues, we have first developed a heuristic algorithm to construct large kernels with good scaling exponents. This construction is based on the relation between the scaling exponent of a kernel and its *polarization behavior* under erasure channels. The polarization behavior of an $\ell \times \ell$ kernels characterizes the evolution of Bhattacharyya parameter through one step of polarization, which can be explicitly formulated in the case of erasure channels. We construct two larger kernels for $\ell = 16$ and $\ell = 32$ with scaling exponents $\mu_{16} = 3.356$ and $\mu_{32} = 3.122$. We also investigate a conjecture that was originally given in [45] and stated that one can find $\ell \times \ell$ kernels with scaling exponent arbitrary close to $\mu_{\text{opt}} = 2$ if ℓ is large enough. Indeed, we prove this conjecture and show that the scaling exponents of almost-all binary $\ell \times \ell$ polarization kernels are bounded by $\mu_\ell \leqslant 2 + \epsilon$ if

$$\frac{\log \ell}{\log \log \ell} \geqslant c_0 \epsilon, \tag{1.9}$$

where c_0 is a universal constant. However, this only holds for when the underlying communication channel is an erasure channel. We leave the extensions of this theorem for future.

The increased computational complexity of polar codes constructed from larger kernels are commonly viewed as their main practicality challenge. The computational complexity relates to both space and time complexities of the decoder, which determine the chip size and its throughput eventually. While decoding traditional polar codes under successive cancellation algorithm has computational complexity of only $O(n \log n)$, this value scales by a factor of 2^ℓ for when Arıkan's kernel is replaced with an arbitrary $\ell \times \ell$ one. We study this effect at the end of this chapter. We propose a low-complexity implementation of the successive cancellation decoder for a class of large kernels that we refer to by the permuted kernels. This class includes all kernels that can be generated form the Kronecker powers of Arıkan's 2×2 kernel through row permutations. In fact, our heuristic algorithm for $\ell = 8$ generates one such kernel. Furthermore, we generalize this method to cover a wider class of kernels whose structures share certain similarities with the Arıkan's kernels and its powers.

In Chapter 3, we revisit the successive cancellation (SC) decoding algorithm of polar codes. The task of the polar SC decoder is to sequentially estimate the values of the uncoded bits $u_1, u_2, \cdots, u_n$. Some of these bits correspond to the more noisy bit-channels and hence their values are frozen and known to the decoder. The rest of the information bits are to be decided according to the received symbols and their corresponding likelihoods. However, the nature of this decoding method cannot recover from a single mistake while estimating the uncoded bit since there is no coming back. To capture the magnitude of this problem, we recall the definition of the Arıkan's genie, which comes to the rescue for a limited number of times when SC decoder makes mistakes. This can be viewed as a *fictional* side information available to the decoder, which we can artificially provide to the decoder for simulation purposes. Although the Arıkan's genie is an abstract concept, but both numerical simulations and mathematical derivations prove that significant performance improvements are in place if one were able to simulate the genie.

The *CRC-aided list decoding* algorithm of polar codes [39] is the first successful attempt at simulating the Arıkan's genie. It is based on first precoding the information bits with a high-rate *cyclic redundancy check* (CRC) and then pursue not only the most likely path in the successive cancellation decoder but to generate a list of L likely candidates. Upon arriving at those L candidates, decoder cross matches them with the CRC check to eliminate all incorrect candidates. This acts as if the CRC is some side information, which allows the decoder to catch and correct some of its mistakes at the end. As shown in multiple references, CRC-aided SCL decoding of polar codes drastically improves their performance. However, achieving the desired performance targets in wireless communication systems requires decent sized lists. In turn, increased list-size negatively affects the latency, throughput, and power consumption of the resulting decoder.

One may ask if the performance of CRC-aided SCL decoding with a list of size L can be achieved without following L decoding paths? We believe that it can. We propose a new method that is based on using a *convolutional outer code*, in lieu of a CRC code, and try to correct decision errors in the successive-cancellation decoding process *locally*, on the fly. Indeed, we run the Viterbi algorithm on the output of the SC decoder. Upon detecting a decision error on bit u_i, a genie-like feedback is activated to correct the error and reset the SC decoder back to time i. In a nutshell, we propose to implement Arıkan's genie via the Viterbi algorithm with feedback to the SC decoder. Notably, this entails exploring *only one path* in the decision tree, apart from an occasional reset of about a dozen bits back.

Time synchronization is fundamental in all wireless standards, especially those based upon time-division duplex (TDD) methods. For instance, current 4G-TDD systems [52] require accuracy of at least $1.5\mu s$. While these accuracy requirements are already tight, they will be even more stringent in future wireless systems. Therefore, dealing with synchronization errors will become inevitable. Synchronization errors occur when the mismatch of clocks in the transmitter and the receiver exceeds what the underlying protocol can tolerate. Such errors result in the *insertion and/or deletion of bits* in the transport block. Channels corrupted by insertions/deletions have memory; hence techniques developed for memoryless channels do not apply [53]. While polarization theory has been extended to *some* channels with memory [51, 54, 55], unfortunately these results do not apply to the deletion channel. Indeed, even the capacity of the deletion channel is not fully known [56, 57]. Moreover, there is a glaring lack of good coding schemes for correcting insertions and deletions. Somewhat embarrassingly, there is still no satisfactory solution even to the simple problem of correcting *only two* deletion errors [53, 56]. In this part of our research, we ask: Can polarization theory along with the power methods of polar coding be extended to channels with deletions?

In Chapter 4, we propose a new coding scheme for correcting a limited number of deletion errors, whose complexity scales only *polynomially*, rather than exponentially, with the number of deletions d. In fact, the decoding complexity with this approach is $O(d^2 n \log n)$. The proposed algorithm extends the successive-cancellation decoding idea to channels with deletions and cleverly exploits the beautiful recursive structure of polar codes. Let us consider the well-known FFT-like polar graph composed of $m = \log_2(n)$ decoding layers, that underlies SC decoding. It is precisely the structure of this graph that makes it possible to reduce the decoding complexity from exponential to polynomial in d. Rather than *guessing-and-checking* all the $\binom{n}{d}$ possible deletion patterns, each node in the graph propagates its uncertainty about the deletion pattern to the next decoding layer. Magically, with high probability, the correct deletion pattern becomes visible when the last polar bit-channel is decoded.

The secret to this magic is, indeed, the structure of the polar graph. When processing each node in this graph, we only need to know a subset of the received bits. Moreover, this subset always forms a consecutive interval in the channel output. Therefore, in order to compute the output from any node in decoding graph, all we need to know is the number of deletions *before and after the corresponding interval*. This means that instead of $\binom{n}{d}$ possibilities, there are at most $(d+1)(d+2)/2$ different scenarios to consider at each node. We prove that, as far as successive-cancellation decoding is concerned, these $O(d^2)$ scenarios are a *sufficient statistic* for the actual deletion pattern.

We also investigate channel-polarization paradigm for the proposed decoding algorithm. In particular, we characterize the resulting bit-channels and analyze their evolution as the block-length grows. Our decoding algorithm, in effect, re-defines polar bit-channels, so that (2.4) no longer applies. In fact, instead of n bit-channels at each decoding layer, we now have $O(d^2 n)$ of them. We prove that these bit-channel polarize for when d is a constant and conjecture that same result holds for all values of d that grow sublinearly with n, $d = o(n)$.

Each chapter is followed by a short survey of a few related open problems in the field of polar coding, some of which were originated through this research. For some of the research problems we propose, we can see a clear path towards their successful implementation in wireless communication systems. On the other hand, for many others, we feel that substantial technical challenges lie in front of us. It is our intent that this work should open avenues for future research and provide useful techniques that will be of value to other investigators.

Chapter 2

Polar Codes with Large Kernels

2.1 Preliminary Topics

2.1.1 Background and Context

Shannon's coding theorem implies that for every binary-input memoryless symmetric (BMS) channel W, there is a capacity $I(W)$ such that the following holds: for all $\varepsilon > 0$ and $P_e > 0$, there exists a binary code of rate at least $I(W) - \varepsilon$ which enables communication over W with probability of error at most P_{e}. Ever since the publication of Shannon's famous paper [1], the holy grail of coding theory was to find explicit codes that achieve Shannon capacity with polynomial-time complexity of construction and decoding. Today, several such families of codes are known, and the principal remaining challenge is to characterize *how fast we can approach capacity* as a function of the code block length n. Specifically, we now have explicit binary codes (which can be constructed and decoded in polynomial time) of length n and rate R, such that the gap to capacity $\epsilon = I(W) - R$ required to achieve any fixed error probability $P_{\mathrm{e}} > 0$ vanishes as a function of n. The fundamental theoretical problem is to characterize how fast this happens. Equivalently, we can fix $\epsilon = I(W) - R$ and ask how large does the block length n need to be as a function of ϵ. That is, we are interested in the *scaling between the block length and the gap to capacity*, under the constraint of polynomial-time construction and decoding.

It is known that the optimal scaling is of the form $n = O(1/\varepsilon^{\mu})$, where μ is referred to as the *scaling exponent*. It is furthermore known that the best possible scaling exponent is $\mu = 2$, and it is achieved by random linear codes — although, of course, random codes do not admit efficient decoding. In this chapter, we present the first family of binary codes that attains both optimal scaling and quasi-linear complexity on the the binary erasure channel (BEC). Specifically, for any fixed $\delta > 0$, we exhibit codes that ensure reliable communication on the BEC at rates within $\varepsilon > 0$ of the Shannon capacity, with block length $n = O(1/\varepsilon^{2+\delta})$, construction complexity $\Theta(n)$, and encoding/decoding complexity $\Theta(n \log n)$.

To establish this result, we use polar codes, invented by Arıkan [10] in 2009. However, while Arıkan's polar codes are based upon a specific 2×2 kernel, we use $\ell \times \ell$ binary polarization kernels, where ℓ is a sufficiently large constant. The main technical challenge is to prove that this construction works. To this end, we choose the polarization kernel uniformly at random from the set of all $\ell \times \ell$ nonsingular binary matrices, and show that with probability at least $1 - O(1/\ell)$, the resulting scaling exponent is at most $2 + \delta$. Since ℓ is a constant that depends only on δ, the choice of a polarization kernel can be, in principle, de-randomized with complexity which is independent of the block length (and depends only on ℓ).

In the following, we provide a brief summary of what is known about the scaling exponent of major families of linear codes.

A sequence of papers starting with [58, 59] in 1960s and culminating with [60, 61] shows that for any discrete memoryless channel W and *any* code of length n and rate R that achieves error-probability P_{e} on W, we have

$$I(W) - R \geqslant \frac{\text{const}(P_{\text{e}}, W)}{\sqrt{n}} - O\left(\frac{\log n}{n}\right), \tag{2.1}$$

where the constant (which is given explicitly in [61]) depends on W and P_{e}, but not on n. This immediately implies that if $n = O\left(1/\epsilon^{\mu}\right)$, where $\varepsilon = I(W) - R$ is the gap to capacity, then $\mu \geqslant 2$. We further note that expressions similar to (2.1) were derived from the perspective of threshold phenomena in [62] and from the perspective of statistical physics in [63]. The fact that $\mu \geqslant 2$ also follows from a heuristic argument. For simplicity, consider the special case of transmission over the BEC with erasure probability p. As $n \to \infty$, the number of erasures tend to the normal distribution with mean np and standard deviation $\sqrt{np(1-p)}$. Thus, channel randomness yields a variation in the fraction of erasures of order $1/\sqrt{n}$. This implies that, in order to achieve a fixed error probability, the gap to capacity ϵ has to scale at least as $1/\sqrt{n}$.

It is well known [60, 61] that the lower bound $\mu = 2$ is achieved by random linear codes. For the special case of transmission over the BEC, the proof of this fact reduces to computing the rank of a certain random matrix. Indeed, the generator matrix of a random linear code of length n and rate R is a matrix with Rn rows and n columns whose entries are i.i.d. uniform in $\{0, 1\}$. The effect of transmission over the BEC with erasure probability p is equivalent to removing each column of this generator matrix independently with probability p. The probability of error (under maximum-likelihood decoding) is equal to the probability that such residual matrix is not full-rank. This probability is easy to compute, and the desired scaling result immediately follows.

Unfortunately, random linear codes cannot be decoded efficiently. On general BMS channels, this task is NP-hard [64]. On the BEC, decoding a general binary linear code takes time $O(n^\omega)$, where ω is the exponent of matrix multiplication. This leads to the following natural question: what is the lowest possible scaling exponent for binary codes that can be constructed, encoded, and decoded efficiently? For the BEC, we take *efficiently* to mean linear or quasi-linear complexity. Here is a brief survey of the current state of knowledge on this question.

Forney's concatenated codes [65] are a classical example of a capacity-achieving family of codes. However, their construction and decoding complexity are exponential in the inverse gap to capacity $1/\epsilon$ (see [44] for more details), so they are definitely not efficient. Let us also point out that we can define a scaling exponent also for codes that do not achieve capacity by substituting the channel capacity with the specific threshold of the code. In this context, for a large class of ensembles of LDPC codes and channel models, the scaling exponent is also $\mu = 2$ [66]. However, the threshold of such LDPC ensembles does not converge to capacity.

In recent years, three new families of achieve capacity-achieving codes have been discovered; let us review what is known regarding their scaling exponents.

Polar codes: Achieve the capacity of any BMS channel under a successive-cancellation decoding algorithm [10] that runs in time $O(n \log n)$. It was shown in [44] that the block length, construction complexity, and decoding complexity are all bounded by a polynomial in $1/\epsilon$, which implies that the scaling exponent μ is finite. Later, a sequence of papers [3, 43, 46, 47] provided rigorous upper and lower bounds on μ. The specific value of μ depends on the channel W. It is known that $\mu = 3.63$ on the BEC. The best-known bounds valid for any BMS channel W are given by $3.579 \leqslant \mu \leqslant 4.714$.

Spatially-coupled LDPC codes: Achieve the capacity of any BMS channel under a belief-propagation decoding algorithm [67] that runs in linear time. A simple heuristic argument yields that the scaling exponent of these codes is roughly 3 (see [68, Section VI-D]). However, a rigorous proof of this statement remains elusive and appears to be technically challenging.

Reed-Muller codes: Achieve capacity of the BEC under maximum-likelihood decoding [69, 70] that runs in time $O(n^{\omega})$. While it has been observed empirically that the performance of Reed-Muller codes on the BEC is close to that of random codes [71], no bounds on the scaling exponent of these codes are known.

2.1.2 Finite-Length Scaling of Polar Codes

The performance of polar codes has been analyzed in several regimes. In the *error exponent* regime, the rate $R < I(W)$ is fixed, and it is studied how the error probability P_e scales as a function of the block length n. This approach is represented as the vertical/blue cut in Figure 2.1. In [72] it is proved that the error probability under SC decoding behaves roughly as $2^{-\sqrt{n}}$. An even more refined scaling is proved in [73].

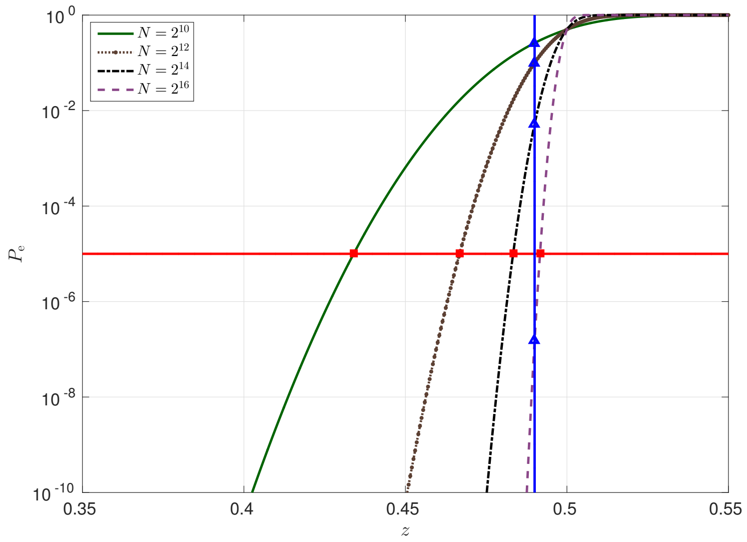


Figure 2.1: Performance of a family of codes with rate $R = 0.5$ as appears in [3].

In the *error floor* regime, the code is fixed, i.e., the rate R and the block length n are fixed, and it is studied how the error probability P_{e} scales as a function of the channel parameter. This approach corresponds to taking into account one of the four curves in Figure 2.1. Each curve corresponds to a code of an assigned block length n; on the x-axis it is represented the parameter z of the transmission channel; and on the y-axis the error probability P_{e}. The error exponent regime captures the behavior of the blue vertical cuts of fixed channel parameter z (or, equivalently, of fixed gap to capacity $I(W) - R$). The error floor regime captures the behavior of a single curve of fixed block length n. The scaling exponent regime captures the behavior of the red horizontal cuts of fixed error probability P_{e}. The figure is courtesy of [3]. In [74] it is proved that the stopping distance of polar codes scales as $\sqrt{n}$, which implies good error floor performance under BP decoding. The authors of [74] also provide simulation results that show no sign of error floor for transmission over the BEC and over the binary-input AWGN channel. This conjecture is settled in [3], where it is showed that polar codes do not exhibit error floors for the transmission over any BMS channel.

In this chapter, our main focus is on the *scaling exponent* regime, where the error probability $P_{\rm e}$ is fixed, and it is studied how the gap to capacity $I(W) - R$ scales as a function of the block length n. This approach is represented as the horizontal/red cut in Figure 2.1. As mentioned earlier, if n is $O\left(1/(I(W) - R)^{\mu}\right)$, then we say that the family of codes has *scaling exponent* μ. For polar codes, the value of μ depends on the particular channel taken into account. In [47], it is presented a heuristic method for computing the scaling exponent for the transmission over the BEC under SC decoding; this method yields $\mu \approx 3.627$. In [44], it is shown that the block length, construction, encoding and decoding complexity are all bounded by a polynomial in the inverse of the gap to capacity for the transmission over any BMS channel. This implies that there exists a finite scaling exponent μ. Rigorous bounds on μ are provided in [3, 43, 46]. In [46], it is proved that $3.579 \leqslant \mu \leqslant 6$, and it is conjectured that the lower bound can be increased up to 3.627, i.e., up to the value heuristically computed for the BEC. In [43], the upper bound is refined to 5.702. The current best upper bounds on the scaling exponent are provided in [3]: for any BMS channel, $\mu \leqslant 4.714$; and for the special case of the BEC, $\mu \leqslant 3.639$, which approaches the value obtained heuristically in [47]. As a side note, let us point out that the heuristic method of [47] is based on a “Scaling Assumption” that requires the existence of a particular limit. The results of [3, 43, 46], as well as the result presented in this paper, do not rely on such an assumption.

In [3], it is also proved that, by allowing a less favorable scaling between the gap to capacity and the block length (i.e., a larger scaling exponent), the error probability goes to 0 sub-exponentially fast in n. This intermediate regime is referred to as *moderate deviations* regime. Here, neither the rate nor the error probability are fixed, and it is studied how the gap to capacity $I(W) - R$ and the error probability $P_{\rm e}$ jointly scale as functions of the block length n (see [3, Theorem 3]).

In a nutshell, the scaling exponent of Arıkan’s polar codes is around 4 (its exact value depends on the transmission channel and it can be bounded as $3.579 \leqslant \mu \leqslant 4.714$). On the contrary, random codes achieve the optimal scaling exponent of 2. This means that, in order to obtain the same gap to capacity, the block length of polar codes needs to be roughly the square of the block length of random codes. Hence, one natural question is how to improve the scaling exponent of polar codes.

One possible approach consists in acting on the decoding algorithm. In particular, the successive cancellation list decoder proposed in [39] empirically provides a significant performance improvement. However, in [48], it is proved a negative result for list decoders: the introduction of any finite list cannot improve the scaling exponent under MAP decoding for the transmission over any BMS channel. Furthermore, for the special case of the BEC, it is also proved that the scaling exponent under SC decoding does not change even if one is given a finite number of helps from a genie.

Another approach is to consider the polarization of kernels larger than the original 2×2 matrix. Indeed, such kernels have the potential to improve the scaling behavior of polar codes. For the error exponent, in [29] it is proved that, as ℓ goes large, the error probability scales roughly as 2^{-n}. For the scaling exponent, in [4] it is observed that μ can be reduced when $\ell \geqslant 8$. In the recent paper [75], it is shown that, for the transmission over the erasure channel, the optimal scaling exponent $\mu = 2$ is approached by using a large kernel and a large alphabet. Furthermore, in [45], the author gives evidence supporting the conjecture that, in order to obtain $\mu = 2$, it suffices to consider a large kernel over a binary alphabet. Here, we finally settle such a conjecture: we show that the scaling exponent $\mu(\ell)$ obtained from the polarization of an $\ell \times \ell$ kernel tends to 2, as ℓ goes large. We furthermore characterize precisely how large ℓ needs to be as a function of the gap between $\mu(\ell)$ and 2. The resulting binary codes maintain the beautiful recursive structure of conventional polar codes, and thereby achieve construction complexity $\Theta(n)$ and encoding/decoding complexity $\Theta(n \log n)$. This implies that block length, construction, encoding, and decoding complexity are all linear or quasi-linear in $1/\varepsilon^2$, which meets the information-theoretic lower bound.

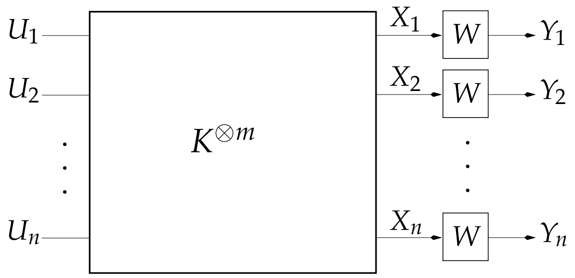
2.1.3 Polarization Theory for Large Kernels

We recall again that polarization nduced via a simple linear transformation consisting of many Kronecker products of a binary matrix K, called the *polarization kernel*, with itself. The conventional polar codes introduced by Arıkan in [10] correspond to the choice

$$K = \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}. \tag{2.2}$$

In general, we can construct polar codes for any kernel K that is an $\ell \times \ell$ non-singular binary matrix, which cannot be transformed into an upper triangular matrix under any column permutations [29].

Let $W\colon \{0,1\} \to \mathcal{Y}$ be a BMS channel, characterized in terms of its transition probabilities $W(y|x)$, for all $y \in \mathcal{Y}$ and $x \in \{0,1\}$. Further, let $\boldsymbol{U} = (U_1, U_2, \ldots, U_n)$ be a block of $n = \ell^m$ bits chosen uniformly at random from $\{0,1\}^n$. We encode $\boldsymbol{U}$ as $\boldsymbol{X} = \boldsymbol{U} K^{\otimes m}$ and transmit $\boldsymbol{X}$ through n independent copies of W, as shown below:



(2.3)

To understand what polarization means in this context, we consider a number of channels associated with this transformation (see also Chapter 5 of [54] and Chapter 2.4 of [45]). Let $W^n\colon \{0,1\}^n \to \mathcal{Y}^n$ be the channel that corresponds to n independent uses of W; let $W_n\colon \{0,1\}^n \to \mathcal{Y}^n$ be the channel with transition probabilities given by

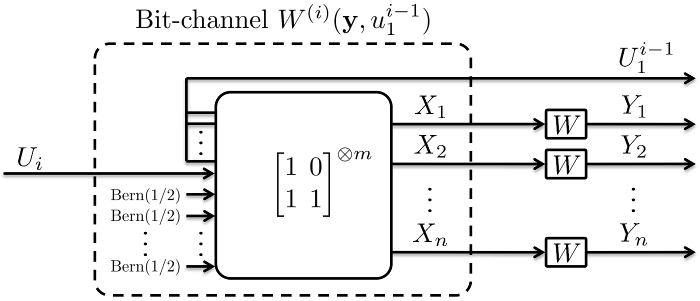

Bit-channel $W^{(i)}(\mathbf{y}, u_1^{i-1})$
U_1^{i-1}
X_1
Y_1
X_2
Y_2
U_i
$\begin{bmatrix}1 & 0\\ 1 & 1\end{bmatrix}^{\otimes m}$
Bern(1/2)
Bern(1/2)
Bern(1/2)
X_n
Y_n
W
W
W

Figure 2.2: Graphical representation of the i-th polar bit-channel.

$W_n(\boldsymbol{y}|\boldsymbol{u}) = W^n\big(\boldsymbol{y}\,\big|\,\boldsymbol{u}K^{\otimes m}\big)$; and, for $i \in [n]$, let $W^{(i)} : \{0,1\} \to \mathcal{Y}^n \times \{0,1\}^{i-1}$ be the channel that is "seen" by the bit U_i, defined as

$$W^{(i)}\big(\boldsymbol{y},\boldsymbol{v}|u_i\big) \overset{\text{def}}{=} \frac{1}{2^{n-1}} \sum_{\overline{\boldsymbol{u}} \in \{0,1\}^{n-i}} W_n\Big(\boldsymbol{y}\,\big|\,(\boldsymbol{v}, u_i, \overline{\boldsymbol{u}})\Big) \;=\; \frac{1}{2^{n-1}} \sum_{\overline{\boldsymbol{u}} \in \{0,1\}^{n-i}} W^n\Big(\boldsymbol{y}\,\big|\,(\boldsymbol{v}, u_i, \overline{\boldsymbol{u}})K^{\otimes m}\Big), \tag{2.4}$$

where $(\cdot,\cdot)$ stands for vector concatenation. It is easy to show that $W^{(i)}\big(\boldsymbol{y},\boldsymbol{v}|u_i\big)$ is indeed the probability of the event that $(Y_1, Y_2, \ldots, Y_n) = \boldsymbol{y}$ and $(U_1, U_2, \ldots, U_{i-1}) = \boldsymbol{v}$ given the event $U_i = u_i$. Consequently, if one considers a "hypothetical decoder" that attempts to estimate the i-th input bit U_i in (2.4) having observed the channel output $\boldsymbol{Y}$ and the first $i-1$ bits of the input $\boldsymbol{U}$, then $W^{(i)}$ is the effective channel seen by such decoder. We refer to $W^{(i)}$ as the *i-th bit-channel* for $i = 1, 2, \ldots, n$ (W_i is called a "split channel" in [10]). A graphical illustration of the i-th bit-channel is given in Figure 2.2.

The key observation of [10] is that, as n grows, the n *bit-channels* W_i defined in (2.4) start *polarizing*: they approach either a *noiseless channel* or a *useless channel*. Formally, given a BMS channel W, its *capacity* $I(W)$ and *Bhattacharyya parameter* $Z(W)$ are given by

$$\begin{aligned} I(W) &\overset{\text{def}}{=} \frac{1}{2} \sum_{y \in \mathcal{Y}} \sum_{x \in \{0,1\}} W(y|x) \log_2 \frac{W(y|x)}{\frac{1}{2}W(y|0) + \frac{1}{2}W(y|1)}\,, \\ Z(W) &\overset{\text{def}}{=} \sum_{y \in \mathcal{Y}} \sqrt{W(y|0)W(y|1)}. \end{aligned} \tag{2.5}$$

Given $\delta \in (0,1)$, let us say that a bit-channel $W^{(i)}$ is *δ-bad* if $Z(W^{(i)}) \geqslant 1-\delta$ and *δ-good* if $Z(W^{(i)}) \leqslant \delta$. Then the polarization theorem of Arıkan [10, Theorem 1] can be informally stated as follows.

Theorem 3. *[Polarization Theorem] For every $\delta \in (0,1)$, almost all bit-channels become either δ-good or δ-bad as $n \to \infty$. In fact, as $n \to \infty$, the fraction of δ-good bit-channels approaches the capacity $I(W)$ of the underlying channel W, while the fraction of δ-bad bit-channels approaches $1 - I(W)$.*

This theorem naturally leads to the construction of capacity-achieving *polar codes*, as long as δ is $o(1/n)$. Specifically, an (n,k) polar code is constructed by selecting a set $\mathcal{A}$ of k good bit-channels to carry the information bits, while the input to all the other bit-channels is frozen to zeros. In practice, the code parameters k and δ are usually selected according to the target rate of the code and/or the desired probability of error.

The error probability under SC decoding is upper bounded by the sum of the smallest k Bhattacharyya parameters of the synthetic channels. Let us also mention that the polarization theorem stated above was originally proved in [10] for $\delta = n^{-5/4}$ (which suffices to give capacity-achieving codes) and later improved to $\delta \sim 2^{-\sqrt{n}}$ in [72].

Let us now focus on the *binary erasure channel*, where the erasure probability z is equal to the Bhattacharyya parameter $Z(W)$ defined in (2.5). It is easy to see that when the underlying channel W is a BEC(z), then, for all i, the i-th bit-channel $W^{(i)}$ is a BEC($z_m(i)$), where $z_m(i)$ is a polynomial of degree at most n in z (see also Section 3.4 of [45]). The proof of the polarization theorem follows by studying the evolution of the Bhattacharyya parameters $z_m(i)$, as m grows. For a fixed kernel K, these $n = \ell^m$ Bhattacharyya parameters $z_m(i)$ can be viewed as the values of the random variable Z_m induced by the uniform distribution on the ℓ^m bit-channels. More formally, the recursive construction of $K^{\otimes m}$ allows $\{Z_m\}_{m\in\mathbb{N}}$ to form the supermartingale:

$$Z_{m+1} = f_{B_m,K}(Z_m), \quad \text{for } B_m \sim \textsf{Uniform}[\ell], \tag{2.6}$$

with the initial condition $Z_0 = z$ and where, for $i \in [\ell]$, $f_{i,K}(z)$ denotes the erasure probability of the i-th bit-channel after one step of polarization. We shall refer to the set $\{f_{i,K}(z) : i \in [\ell]\}$ as the *polarization behavior of* K. We will show in the next section that $f_{i,K}(z)$ is a polynomial of degree at most ℓ in z. For the special case of the kernel (2.2), we have that

$$\begin{aligned} f_{0,K}(z) &= 2z - z^2, \\ f_{1,K}(z) &= z^2. \end{aligned} \tag{2.7}$$

One can view (2.6) as a stochastic process on an infinite binary tree, where in each step we take one of the ℓ available branches with uniform probability. The polarization theorem is then reduced to the martingale convergence theorem for supermartingales, which in this case implies that

$$\lim_{m\to\infty} Z_m(1 - Z_m) = 0. \tag{2.8}$$

This shows that the erasure probability of the bit-channels polarizes to 0 or 1 as $m \to \infty$. Furthermore, by applying the chain rule of mutual information, one can show that this polar transform preserves capacity. Hence, the fraction of bit-channels that polarizes to 0 approaches $I(W)$. The speed with which this polarization phenomenon takes place is the determining factor in the decay rate of the gap to capacity as a function of the block length ℓ^m. We elaborate on this in the next subsection.

2.1.4 Binary Erasure Channels and Polarization Behavior

Let us recall again that, when the transmission channel is a BEC(z), each polar bit-channel channels is also a BEC whose erasure probability is a polynomial in z. In this section, we give more insight into this fact by first describing the successive cancellation decoding method for BECs, and then establishing a connection between the decodability of the i^{th} bit-channel and the column spaces of some sub-matrices of K.

Let $K \in \mathbb{F}_2^{\ell\times\ell}$ be a non-singular binary kernel. Assume that the underlying channel is a BEC(z) and let $\boldsymbol{e}$ denote the erasure pattern, which is a length-ℓ vector in $\{0,\triangle\}^\ell$ with the property that $e_i = \triangle$ if the i-th symbol is erased and $e_i = 0$ otherwise. Let us also define $wt(\boldsymbol{e})$ to denote the number of erasures in $\boldsymbol{e}$. Given that each symbol gets erased independently with probability z, the probability of observing a fixed erasure pattern such as $\boldsymbol{e}$ is given by

$$\mathbb{P}(\text{observing deletion pattern } \boldsymbol{e}) = z^{wt(\boldsymbol{e})}(1-z)^{\ell-wt(\boldsymbol{e})}. \tag{2.9}$$

Definition . *Assume that the underlying communication takes place over BEC(z). Further assume that the input-output relation at the encoder is given by $\boldsymbol{x} = \boldsymbol{u}K$, where K is a given polarization kernel. We define the erasure pattern $\boldsymbol{e}$ to be an (i,K)-uncorrectable erasure pattern if it makes u_i undecidable.*

The erasure probability at the i-th bit-channel can now be formulated as

$$f_{i,K}(z) = \sum_{\boldsymbol{e}\,:\,\boldsymbol{e}\text{ is }(i,K)-\text{uncorrectable}} z^{wt(\boldsymbol{e})}(1-z)^{\ell-wt(\boldsymbol{e})}, \tag{2.10}$$

which is equivalent to

$$f_{i,K}(z) = \sum_{s=0}^{\ell} z^s(1-z)^{\ell-s}\big(\#\text{ of erasure patterns with } s \text{ erasures that make } u_i \text{ undecidable}\big). \tag{2.11}$$

Table 2.1: The list of $(2, K_3)$-uncorrectable erasure patterns.

erasure pattern $\boldsymbol{e}$	000	$00\triangle$	$0\triangle0$	$0\triangle\triangle$	$\triangle00$	$\triangle0\triangle$	$\triangle\triangle0$	$\triangle\triangle\triangle$
erasure weight	0	1	1	2	1	2	2	3
decodable	y	y	y	n	y	y	n	n

Definition. *The polarization behavior of an $\ell \times \ell$ kernel K is defined as the set of ℓ polynomials*

$$\{f_{1,K}(z), f_{2,K}(z), \cdots, f_{\ell,K}(z)\} \tag{2.12}$$

that define the erasure probabilities of polar bit-channels after one step of polarization.

Example 1. Assume that the polarization kernel is given by

$$K_3 = \begin{bmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ 1 & 0 & 1 \end{bmatrix}.$$

We can formulate the relation between uncoded bits $\boldsymbol{u}$ and the received symbols $\boldsymbol{y}$ as $\boldsymbol{y} = \boldsymbol{u}K_3 + \boldsymbol{e}$, where $0 + \triangle = \triangle$ and $1 + \triangle = \triangle$. In successive cancellation decoding of u_2, we assume that the value of u_1 is known. We proceed by canceling out the effect of u_1 as

$$\boldsymbol{y} - u_1\begin{bmatrix}1 & 0 & 0\end{bmatrix} = (u_2, u_3)\begin{bmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \end{bmatrix} + \mathbf{e} = \big(\underbrace{u_2 + u_3 + e_1}_{c_1}, \underbrace{u_2 + e_2}_{c_2}, \underbrace{u_3 + e_3}_{c_3}\big). \tag{2.13}$$

It is now clear that the only combinations that can possibly help to recover u_2 are $c_1 + c_3$ and c_2. Therefore, u_2 is decodable if $e_2 = 0$ or both $e_1, e_3 = 0$. This is also demonstrated in Table 2.1. The erasure probability of $W^{(2)}$ is hence given by

$$f_{2,K_3}(z) = 1 - \big((1 - z) + z(1 - z)^2\big). \tag{2.14}$$

Next, we look at the general case by first fixing an erasure pattern with s erasures. For convenience, let the last s coordinates denote its erasure locations. Hence, the received symbols are given as

$$\mathbf{x}_1^{\ell-s} = \mathbf{u}K_{[:,1:\ell-s]} \ , \tag{2.15}$$

where $K_{[:,1:\ell-s]}$ denotes the sub-matrix of K that is formed by removing its last s columns. Furthermore, note that in the successive cancellation decoding of u_i, we assume that $\mathbf{u}_1^{i-1}$ is known. We can now re-write (2.15) as

$$\mathbf{x}_1^{\ell-s} = \mathbf{u}K_{[:,1:\ell-s]} = \left[\mathbf{u}_1^{i-1}\middle|\mathbf{u}_i^{\ell}\right] K_{[:,1:\ell-s]} = \left[\mathbf{u}_1^{i-1}\middle|\mathbf{0}_{\ell-i+1}\right] K_{[:,1:\ell-s]} + \underbrace{\left[\mathbf{0}_{i-1}\middle|\mathbf{u}_i^{\ell}\right] K_{[:,1:\ell-s]}}_{=\mathbf{u}_i^{\ell}K_{[i:\ell,1:\ell-s]}} , \tag{2.16}$$

where $K_{[i:\ell,1:\ell-s]}$ denotes the sub-matrix of K that is formed by removing its first $i-1$ rows as well as its last s columns. It is now clear that, in order to decode u_i, one needs to express the column vector $(1, 0, \cdots, 0)^t$ as a linear combination of columns in $K_{[i:\ell,1:\ell-s]}$. In other words,

$$u_i = \text{ decodable } \iff (1, \underbrace{0, 0, \ldots, 0}_{\ell-i})^t \in \text{ column space of } K_{[i:\ell,1:\ell-s]}, \tag{2.17}$$

which is also equivalent to the following condition

$$\exists\psi_1^{i-1} \in \mathbb{F}_2^{i-1} : \qquad (\psi_1, \cdots, \psi_{i-1}, 1, \underbrace{0, 0, \cdots, 0}_{\ell-i})^t \in \text{ column space of } K_{[:,1:\ell-s]}. \tag{2.18}$$

Let e_j denote the j-th element of the canonical basis and define the linear subspace $E_j \subset \mathbb{F}_2^{\ell}$ as

$$E_j \triangleq \text{span}\langle e_1, e_2, \cdots, e_j\rangle.$$

Therefore, the decodability condition can be further simplified as

$$u_i = \text{ decodable } \iff (E_i \setminus E_{i-1}) \cap (\text{column space of } K_{[:,1:\ell-s]}) \neq \emptyset. \tag{2.19}$$

2.2 Construction of Large Polarization Kernels

2.2.1 Derivation Methods for Polarization Behavior

Earlier in this chapter, we provided an example of of how to compute the polarization behavior of a given $\ell \times \ell$ kernel K. However, it is easy to verify that the naive approach requires us to cross check all possible 2^ℓ deletions patterns with all 2^ℓ column combinations of the kernel to explicitly the (i, K)-uncorrectable erasure patterns. Therefore the computational complexity of this method is asymptotically given by $O(2^{2\ell})$.

In this subsection, we provide an alternative method which reduces the asymptotic computational complexity of formulating the polarization behavior to $O(\ell^3 2^\ell)$. We also prove that this problem is equivalent to finding the minimum distance of an arbitrary linear code, which is proven to be NP-hard.

We begin by explaining set of operations that preserve the polarization behavior of a kernel. Recall again that a non-singular binary $\ell \times \ell$ matrix is a polarizing kernel conditioned that it cannot be transformed into an upper-triangular matrix under any column permutations. We refer readers to [29] for the proof. It is clear that the Arıkan's kernel is the only binary polarization kernel of size $\ell = 2$. However, this number increases as ℓ becomes larger.

Lemma 2.1. *The number of binary polarization kernels of size ℓ is given by*

$$2^{\frac{\ell(\ell-1)}{2}}\left(\prod_{i=1}^{\ell}(2^i - 1) - \ell!\right). \tag{2.20}$$

Proof. The number of non-singular binary $\ell \times \ell$ matrices in $\mathbb{F}_2^{\ell\times\ell}$ is given by

$$\prod_{i=0}^{\ell-1} 2^\ell - 2^i = 2^{\frac{\ell(\ell-1)}{2}} \prod_{i=1}^{\ell}(2^i - 1). \tag{2.21}$$

However, there are $2^{\frac{\ell(\ell-1)}{2}}$ non-singular upper-triangular matrices in $\mathbb{F}_2^{\ell\times\ell}$. These matrices along with their permuted version (column-wise) are pairwise distinct. Subtracting this number from (2.21) completed the proof. □

As we will show later in this section, it is possible to estimate the scaling exponent of polar codes constructed from kernel K for the binary erasure channels based on its polarization behavior. However, to find *fast* polarizing kernels, we not only need to go though extensive calculations for the polarization behavior but to also repeat the same process for all polarization kernels. On the other hand, the number of such kernels given in (2.20) is an indicator of how practically impossible would this task be.

In the following, we show that the polarization behavior of a given kernel K remains unchanged under certain row operation and under any column permutations. This will reduce the search field to only non-singular lower-triangular matrices in $\mathbb{F}_2^{\ell\times\ell}$ whose size is given by $2^{\frac{\ell(\ell-1)}{2}}$. The same operations allow us to reduce the computational complexity of deriving the polarization behavior to $O(\ell^3 2^\ell)$.

Let us denote the linear combinations of columns in K by $K\boldsymbol{v}$, where $\boldsymbol{v} \in \mathbb{F}_2^\ell$. Further let i denote the location of the last non-zero element in $K\boldsymbol{v}$. As (2.18) suggests, the combination $\boldsymbol{v}$ can help decode u_i if non of symbols that correspond to $\operatorname{supp}(\boldsymbol{v})$ are erased.

Definition. *Let K be a given binary $\ell \times \ell$ polarization kernel. The nested chain of* kernel codes *with respect to K denoted by $\{\mathbf{0}\} = \mathcal{C}_0 \subset \mathcal{C}_1 \subset \cdots \subset \mathcal{C}_\ell = \{0,1\}^\ell$ are defined as*

$$\mathcal{C}_i := \{\boldsymbol{v} | \boldsymbol{v} \in \{0,1\}^\ell, K_{[i+1:\ell]}\boldsymbol{v} = \mathbf{0}\} \qquad \forall i:\ 0 \leqslant i \leqslant \ell. \tag{2.22}$$

Lemma 2.2. *Assume K be a given $\ell \times \ell$ binary polarization kernel, whose kernel codes are defined according to (2.22). Let $\boldsymbol{e}$ be a given erasure pattern. Then, $\boldsymbol{e}$ is an (i,K)-uncorrectable erasure patten if and only if* $\operatorname{supp}(\boldsymbol{e})$ *does not cover any codewords in $\mathcal{C}_i \setminus \mathcal{C}_{i-1}$.*

Proof. Follows immediately from (2.17), (2.18), and (2.19). □

Lemma 2.3. *Assume a same setup as in Lemma 2.2 is in place. Then, $\boldsymbol{e}$ is an (i,K)-uncorrectable erasure patten if and only if* $\operatorname{supp}(\boldsymbol{e})$ *covers a codeword in $\mathcal{C}_{i-1}^\perp \setminus \mathcal{C}_i^\perp$.*

Proof. Let $\boldsymbol{e}$ be an (i, K)-uncorrectable erasure pattern. Therefore, there exists at least two different information vectors $\boldsymbol{u}_1$ and $\boldsymbol{u}_2$ that generate the same bits in the nonerased locations. In other words, $\mathrm{supp}(\boldsymbol{e})$ covers the codeword $(\boldsymbol{u}_1 + \boldsymbol{u}_2)K$, which is a codeword in $\mathcal{C}_{i-1}^{\perp} \setminus \mathcal{C}_i^{\perp}$. The reverse follows similarly. □

Lemma 2.4. *Let K and K^* be two polarization kernels such that K^* is constructed from K by adding its j-th row to its i-th row, where $1 \leqslant i < j \leqslant \ell$. Then, K and K^* have the same polarization behavior.*

Proof. Let $\{\mathcal{C}_i\}_{i=0}^{\ell}$ and $\{\mathcal{C}_i^*\}_{i=0}^{\ell}$ denote the kernel codes corresponding to K and K^* respectively. It suffices to show that for all t, $\mathcal{C}_t = \mathcal{C}_t^*$. Note that $K_{[t+1:\ell]}$ is nothing but a parity check matrix for $\mathcal{C}_t$. Now, consider the following three cases:

- If $j \leqslant t$, then $K_{[t+1:\ell]} = K^*_{[t+1:\ell]}$. Therefore, $\mathcal{C}_t = \mathcal{C}_t^*$.
- If $i \leqslant t < j$, then we again have $K_{[t+1:\ell]} = K^*_{[t+1:\ell]}$ and hence $\mathcal{C}_t = \mathcal{C}_t^*$.
- If $t < i$, then $K_{[t+1:\ell]}$ can be transformed to $K^*_{[t+1:\ell]}$ using simple linear row operations. So, for any $\boldsymbol{v} \in \mathbb{F}_2^{\ell}$ we have $K_{[t+1:\ell]}\boldsymbol{v} = 0$ if and only if $K^*_{[t+1:\ell]}\boldsymbol{v} = 0$, which translates to $\mathcal{C}_t = \mathcal{C}_t^*$.

□

Lemma 2.5. *Let K and K_π be two polarization kernels such that K_π is constructed from K by applying a permutation on its columns. Then, K and K_π have the same polarization behavior.*

Proof. Applying a permutation π on columns of K is equivalent to applying the same permutation on both the erasure patterns $\boldsymbol{e}$ and the column combination vectors $\boldsymbol{v}$. Therefore, each (i, K)-uncorrectable erasure pattern $\boldsymbol{e}$ maps to an (i, K_π)-uncorrectable erasure pattern $\pi(\boldsymbol{e})$ with the same weight, which in turn preserves the overall erasure probability of the i-th bit-channel defined in (2.10). □

It is now clear that by using the column permutations and the one-directional row operations we can always transform a non-singular kernel K to a lower triangular kernel K' with the same polarization behavior. Therefore, the computation is only necessary for non-singular lower triangular kernels whose total number is $2^{\ell(\ell-1)/2}$. We need to mention that not all of these kernels have different polarization behaviors. In fact the actual number of polarization behaviors is much less than $2^{\ell(\ell-1)/2}$ and it is still an unsolved problem to find a relation between those kernels with same polarization behaviors.

Let us rewrite the definition in (2.11) as

$$p_i(z) = \sum_{w=0}^{\ell} E_{i,w} z^w (1-z)^{\ell-w}, \tag{2.23}$$

where $E_{i,w}$ is the number of erasure patterns of weight w that kill the bit channel W_i. It is easy to see that

$$0 \leqslant E_{i,w} \leqslant \binom{\ell}{w} \quad \forall w.$$

However, $E_{i,w} = \binom{\ell}{w}$ means that all erasure patterns with weight w make channel $W^{(i)}$ uncorrectable and there should be no codeword with Hamming weight $\ell - w$ in the coset $\mathcal{C}_{i+1} \setminus \mathcal{C}_i$ otherwise the channel would have survived from the erasure pattern $\mathbf{e}$ with weight w, which has erasures in all the coordinates with value 0 in the codeword. Let us define

$$d_i \triangleq wt_{\min}(\mathcal{C}_i \setminus \mathcal{C}_{i-1}) \qquad \text{for all} \quad 1 \leqslant i \leqslant \ell. \tag{2.24}$$

$$d_i^* \triangleq wt_{\min}(\mathcal{C}_{i-1}^{\perp} \setminus \mathcal{C}_i^{\perp}) \qquad \text{for all} \quad 1 \leqslant i \leqslant \ell. \tag{2.25}$$

Theorem 4. *Let d_i be defined according to (2.24). We have*

$$E_{i,w} < \binom{\ell}{w} \qquad \textit{for } 0 \leqslant w \leqslant \ell - d_i,$$

$$E_{i,w} = \binom{\ell}{w} \qquad \textit{for } w > \ell - d_i;$$

Or in other words $d_i = \ell - \max\{w | E_{i,w} < \binom{\ell}{w}\}$.

Proof. Assume that $E_{i,w} = \binom{\ell}{w}$ for some w.Then every single erasure pattern with weight w kills the channel. Since these erasure patterns are also subset of erasure patterns with larger weights, so all erasure patterns with larger weights kill the channel W_i as well and $E_{i,w'} = \binom{\ell}{w'}$ for all $w' \geqslant w$. Suppose that $\ell - d'_i$ is the largest number such that $E_{i,\ell-d'_i} < \binom{\ell}{\ell-d'_i}$. Let us also denote the codeword with minimum Hamming weight in $\mathcal{C}_{i+1} \setminus \mathcal{C}_i$ by $\mathbf{v_i}$. The corresponding linear combination $K_{[i:\ell-1]}\mathbf{v_i}$ survives if there are erasures in all the columns that are not selected by $\mathbf{v_i}$. Hence

$$\ell - d_i \leqslant \ell - d'_i. \tag{2.26}$$

Also note that if $E_{i,\ell-d'_i} < \binom{\ell}{\ell-d'_i}$ then there should be an erasure pattern with $\ell - d'_i$ erasures that does not kill the channel. So there should be a linear combination of columns from the remaining d'_i columns that can recover u_i. Hence there is a codeword $\mathbf{v}$ in $\mathcal{C}_{i+1} \setminus \mathcal{C}_i$ which has no intersection with the erased coordinates, and its support should be included in the set of non-erased coordinates, and

$$d_i \leqslant d'_i. \tag{2.27}$$

(2.26) and (2.27) complete the proof together. □

Theorem 5. *Let d^*_i be defined according to (2.25). We similarly have*

$$\begin{aligned} E_{i,w} &> 0 && \textit{for } d^*_i \leqslant w, \\ E_{i,w} &= 0 && \textit{for } w < d^*_i; \end{aligned}$$

Or in other words $d^*_i = \min\{w | E_{i,w} > 0\}$.

Proof. Proof follows similar to that of Theorem 4 by noting that any erasure pattern $\boldsymbol{e}$ with $wt(\boldsymbol{e}) < d^*_i$ is correctable. □

These theorems establish a relation between finding the minimum Hamming weight in $\mathcal{C}_{i+1} \setminus \mathcal{C}_i$ and the polarization behavior of the kernel K. Indeed, computing the polarization behavior is in general a harder problem that finding the corresponding minimum weights. We can derive the kernel K from any chain of the nested codes and hence $\mathcal{C}_{i+1}$ can be an arbitrary linear code in $\mathbb{F}_2^\ell$ with dimension $i+1$. Let us construct $\mathcal{C}_{i+1}$ from a given linear code $\mathcal{C} \subset \mathbb{F}_2^{\ell-1}$ with adding an extra bit in the first coordinate (code extension). Also assume that $\mathcal{C}_i$ is obtained from $\mathcal{C}_{i+1}$ by shortening in the first coordinate. Now it is clear that the coset $\mathcal{C}_{i+1} \setminus \mathcal{C}_i$ is the set of all codewords in $\mathcal{C}$ but with an extra 1 in the beginning, and hence

$$\min\{wt(\mathbf{v}) | \mathbf{v} \in \mathcal{C}_{i+1} \setminus \mathcal{C}_i\} = 1 + d_{\min}(\mathcal{C}).$$

Finally, we recall the results from [76] where it is shown that finding the minimum distance of a linear code in general is NP-hard. This in turn shows that the computational complexity of the polarization behavior is also NP-hard in general.

For a given $\ell \times \ell$ kernel K, there are 2^ℓ erasure patterns and 2^ℓ linear combinations. As mentioned earlier, the straightforward way to find the polarization behavior is to cross check each erasure pattern $\boldsymbol{e}$ with all linear combinations in $\mathcal{C}_{i+1} \setminus \mathcal{C}_i$ to see if there is the erasure pattern $\boldsymbol{e}$ is (i, K)-correctable or not. However the complexity of running this algorithm is clearly $O(2^{2\ell})$, which soon becomes impractical as ℓ grows. In the following, we propose an alternative solution that exploits the linear construction of cosets $\mathcal{C}_i \setminus \mathcal{C}_{i-1}$ and reduces the overall complexity to $O(\ell^3 2^\ell)$.

Let us fix the erasure pattern $\boldsymbol{e}$. Define

$$\mathcal{C}_i / \operatorname{supp}(\boldsymbol{e})^c = \{\boldsymbol{v} | \operatorname{supp}(\boldsymbol{v}) \subset \operatorname{supp}(\boldsymbol{e})^c\} \quad \forall i, \tag{2.28}$$

where $\operatorname{supp}(\boldsymbol{e})^c$ denotes the complement vector for $\operatorname{supp}(\boldsymbol{e})$. It is then clear that $\boldsymbol{e}$ is an (i, K)-uncorrectable erasure pattern if and only if $\mathcal{C}_i / \operatorname{supp}(\boldsymbol{e})^c = \mathcal{C}_{i-1} / \operatorname{supp}(\boldsymbol{e})^c$. So, in order to determine if $\boldsymbol{e}$ is an uncorrectable erasure pattern or not, it suffices to determine if $\mathcal{C}_i/\operatorname{supp}(\boldsymbol{e})^c = \mathcal{C}_{i-1}/\operatorname{supp}(\boldsymbol{e})^c$ holds.

First we point out the generating matrices of both $\mathcal{C}_i$ and $\mathcal{C}_{i-1}$ can be derived from their parity check matrices in polynomial complexity. Note that $\mathcal{C}_i/\operatorname{supp}(\boldsymbol{e})^c$ is nothing but $\mathcal{C}_i$ shortened to locations in $\operatorname{supp}(bfe)^c$, which is a linear code whose generating matrix can be derived from the generating of $\mathcal{C}_i$ in polynomial computational complexity of at most $O(\ell^3)$. The same goes for $\mathcal{C}_{i-1}/\operatorname{supp}(\boldsymbol{e})^c$. Let us denote the generating and parity check matrices of these codes respectively by $G_{i,\boldsymbol{e}}$, $H_{i,\boldsymbol{e}}$, $G_{i-1,\boldsymbol{e}}$, and $H_{i-1,\boldsymbol{e}}$. Then

$$\boldsymbol{e} = (i, K)\text{-uncorrectable if and only if} \quad G_{i,\boldsymbol{e}} H_{i-1,\boldsymbol{e}} = 0, \tag{2.29}$$

which is a computation with complexity at most $O(\ell^3)$. This process should be repeated for each erasure pattern $\boldsymbol{e}$ individually. Hence, the overall computational complexity of calculating the polarization behavior for the $\ell \times \ell$ kernel K reduces to $O(\ell^3 2^\ell)$.

The proposed algorithm becomes handy when the computation of the polarization behavior for a specific large kernel is required. Based on this approach, we derive the polarization behavior of a constructed 16×16 kernel with scaling exponent $\mu_{16} = 3.356$ in the next subsection.

2.2.2 Heuristic Construction Algorithm for Large Kernels

When studying the polar bit-channels in finite lengths, they cannot be considered as fully noiseless or fully noisy. We instead define thresholds ϵ and $1 - \epsilon'$ so that bit-channels with Bhattacharyya parameter $Z(W) \leqslant \epsilon$ and $Z(W) \geqslant 1 - \epsilon'$ are considered noiseless and useless respectively. Let us also refer to the rest of them by unpolarized bit-channels. The *polarization speed* in fact captures the speed of which the ratio of unpolarized channels goes to zero as the block-length increases and can be formulated as the following:

Assumption 6 (Scaling Assumption)**.** *Given an $\ell \times \ell$ kernel K and a binary discrete memoryless channel W, there exists a $\mu(W,K) \in (0,\infty)$ such that for any $0 < \epsilon < 1-\epsilon' < 1$*

$$\lim_{n\to\infty} \frac{\textit{number of unpolarized channels}}{\textit{number of total channels}} n^{\frac{m}{\mu(W,K)}}$$

exists in $(0,\infty)$.

In [46], authors proposed a heuristic method to calculate the scaling exponent of polar codes for binary erasure channels based on Arikan's 2×2 polar transformation. They also presented an analytical approach to derive a sequence of both upper and lower bounds. Both the heuristic and the analytical approaches depend solely on the polarization behavior of Arıkan's kernel. In the following, we review their both techniques when applied to a larger polarization kernel and estimate the scaling exponent of a few larger kernels.

From this point forward, we always assume that underlying channel W is the BEC(z) where $z \in [0,1]$. Let us fix an $\ell \times \ell$ kernel K whose polarization behavior is given by polynomials $\{p_i(z)\}_{i=1}^{\ell}$. We recall the formulation for evolution of the Bhattacharyya parameter from (2.6) as a random process Z_n such that

$$\begin{aligned} &Z_0 = z, \\ &Z_n = p_i(Z_{n-1}) \quad \text{w.p. } \frac{1}{\ell} \quad \text{for all } 1 \leqslant i \leqslant \ell. \end{aligned} \tag{2.30}$$

Here, Z_n captures the average erasure probability of bit-channels after n levels of polarization. Let us also fix the polarization thresholds $0 < a < b < 1$ and define

$$f_n(z,a,b) = \mathbb{P}(Z_n \in [a,b]), \tag{2.31}$$

which determines the ratio of the unpolarized channels after n levels of polarization. Note that by combining (2.31) and (2.30) we have

$$f_0(z,a,b) = \mathbb{1}_{\{z\in[a,b]\}}, \qquad f_{n+1}(z,a,b) = \frac{\sum_{i=0}^{\ell-1} f_n(p_i(z),a,b)}{\ell}. \tag{2.32}$$

It is easy to show that Assumption 6 is equivalent to the following: There exists $\mu \in (0, \infty)$ such that, for any $z, a, b \in (0, 1)$ such that $a < b$, the limit

$$f(z, a, b) \triangleq \lim_{n \to \infty} \ell^{\frac{n}{\mu}} f_n(z, a, b) \tag{2.33}$$

exists in $(0, \infty)$. Furthermore,

$$\ell^{-\frac{1}{\mu}} f(z, a, b) = \frac{\sum_{i=0}^{\ell-1} f(p_i(z), a, b)}{\ell}. \tag{2.34}$$

It is also possible to show that the value of μ is independent of the selection for $f_0(z, a, b)$ as long as

$$f_0(0, a, b) = f_0(1, a, b) = 0 \quad \text{and} \quad \max_z f_0(z, a, b) = 1. \tag{2.35}$$

This allows us to numerically find the value of μ by initializing $f_0(z, a, b)$ as

$$f_0(z, a, b) = 4z(1 - z) \tag{2.36}$$

and recursively calculating $f_{n+1}(z, a, b)$ according to

$$f_{n+1}(z, a, b) = \hat{f}_{n+1}(z, a, b) / \hat{f}_{n+1}(\frac{1}{2}, a, b), \tag{2.37}$$

where

$$\hat{f}_{n+1}(z, a, b) = \sum_{i=0}^{\ell} f_n(p_i(z), a, b). \tag{2.38}$$

Given that the scaling assumption is true, we have $\mu = \lim_{n \to \infty} \hat{f}_n(\frac{1}{2}, a, b)$.

We combined the method for derivation of the polarization behavior from previous section and the method for estimating the scaling exponent of a kernel based on it polarization behavior to search for a *faster polarizing* kernel among all lower-triangular $\ell \times \ell$ polarization kernels with $\ell \leqslant 7$. To increase the precision of our estimated, we chose the iteration stop condition of $||f_{n+1}(z, a, b) - f_n(z, a, b)||_\infty \leqslant 10^{10}$. The results are tabulated in Table 2.2.

Table 2.2: Scaling exponent of small kernels

ℓ	2	3	4	5	6	7
# of μ's	1	2	10	107	943	14346
$\mu_{\max}$	3.626	8.910	15.231	22.311	29.894	38.030
$\mu_{\min}$	3.626	4.938	3.626	4.235	4.122	3.978

The first interpretation from Table 2.2 is that there are no kernels of size $\ell \leqslant 7$ with scaling exponents less than that of Arıkan's. Furthermore, the total number of total different scaling exponents is still smaller than the number of lower-triangular polarization kernels, *e.g.* $14346 < 2^{\frac{6\times5}{2}} = 32768$, which shows that the two kernel operations introduced in the previous section are probably not the only ones that preserve the polarization behavior; Or, there are multiple different polarization behaviors that yield in a same scaling exponent.

Continuing the brute-force search becomes almost impossible after $\ell = 8$. However, we were able to find a 8×8 kernel, herein denoted by K_8, with scaling exponent $\mu_8 = 3.577$. This is the first example of a binary polarization kernel with scaling exponent less than Arıkan's. Note than all Kronecker powers of K_2 also have the same scaling exponent as K_2. The slightly faster polarizing kernel, K_8, is presented in the following:

$$K_8 = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix}.$$

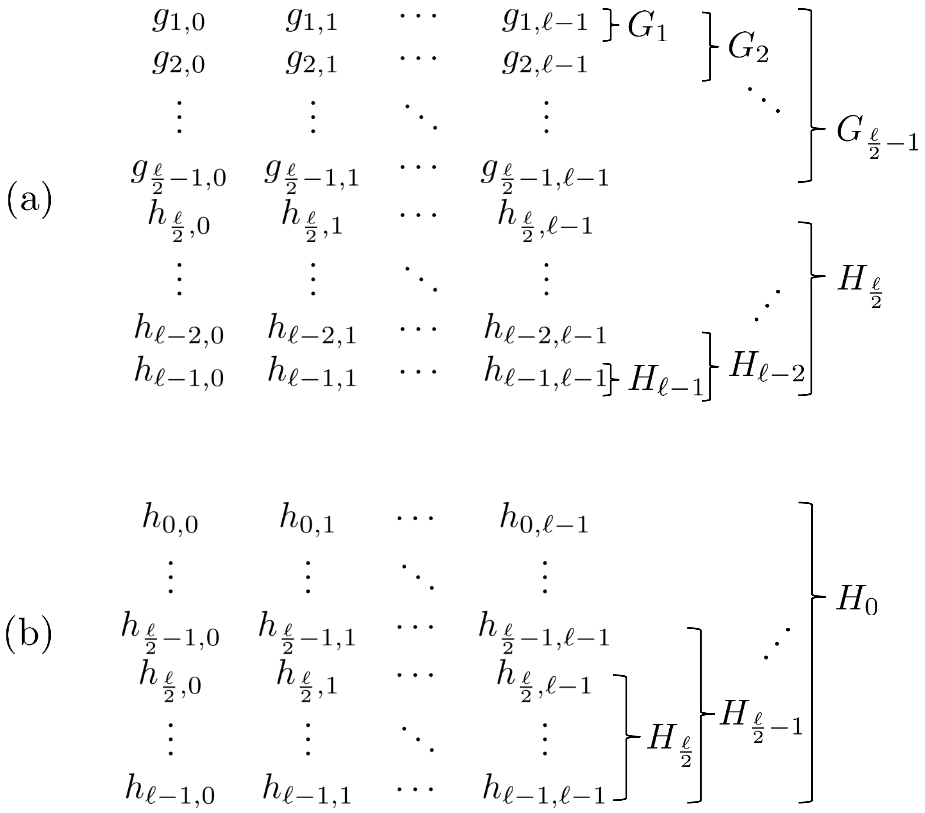
Table 2.3: Polynomial coefficients $E_{i,w}$ in polarization behavior of K_8 and $K_2^{\otimes 3}$.

	K_8									$K_2^{\otimes 3}$								
$i \setminus \ell$	0	1	2	3	4	5	6	7	8	0	1	2	3	4	5	6	7	8
1	0	8	28	56	70	56	28	8	1	0	8	28	56	70	56	28	8	1
2	0	0	16	48	68	56	28	8	1	0	0	16	48	68	56	28	8	1
3	0	0	8	40	66	56	28	8	1	0	0	8	40	66	56	28	8	1
4	0	0	4	24	62	56	8	8	1	0	0	0	0	16	32	24	8	1
5	0	0	0	0	8	32	24	8	1	0	0	4	24	54	56	28	8	1
6	0	0	0	0	4	16	20	8	1	0	0	0	0	4	16	20	8	1
7	0	0	0	0	2	8	12	8	1	0	0	0	0	2	8	12	8	1
8	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1

The polynomial coefficients in polarization behavior of K_8 are tabulated in Table 2.3. The same is presented for $K_2^{\otimes 3}$ for comparison purposes. We observe that the two polarization behaviors almost match except for the middle two bit-channels. Moreover, the polynomials for K_8 are fully sorted, while it is not the case for $K_2^{\otimes 3}$. Let us denote the polarization behaviors of K_8 and $K_2^{\otimes 3}$ by $\{p_i(z)\}_{i=1}^{8}$ and $\{p_i^{'}(z)\}_{i=1}^{8}$ respectively. The third and the probably the most important observation is that

$$\forall i \neq 4, 5: \;\; p_i(z) = p_i^{'}(z) \quad \text{while} \quad p_4(z) \geqslant p_5^{'}(z) \geqslant p_4^{'}(z) \geqslant p_5(z). \tag{2.39}$$

This in a nutshell shows that the middle two bit-channels of K_8 are a bit more polarizing than the two middle bit-channels associated with $K_2^{\otimes 3}$. This leads us to conjecture that a kernel whose polarization behavior is more polarized, should have a smaller scaling exponent. Although we are providing a technical formulation for what *more polarizing* means, we settle on this approach and design a heuristic algorithm to construct large with small scaling exponents.



(a)
$g_{1,0}$ $g_{1,1}$ $\cdots$ $g_{1,\ell-1}$
G_1
$g_{2,0}$ $g_{2,1}$ $\cdots$ $g_{2,\ell-1}$
G_2
$g_{\frac{\ell}{2}-1,0}$ $g_{\frac{\ell}{2}-1,1}$ $\cdots$ $g_{\frac{\ell}{2}-1,\ell-1}$
$G_{\frac{\ell}{2}-1}$
$h_{\frac{\ell}{2},0}$ $h_{\frac{\ell}{2},1}$ $\cdots$ $h_{\frac{\ell}{2},\ell-1}$
$H_{\frac{\ell}{2}}$
$h_{\ell-2,0}$ $h_{\ell-2,1}$ $\cdots$ $h_{\ell-2,\ell-1}$
$h_{\ell-1,0}$ $h_{\ell-1,1}$ $\cdots$ $h_{\ell-1,\ell-1}$
$H_{\ell-1}$
$H_{\ell-2}$
(b)
$h_{0,0}$ $h_{0,1}$ $\cdots$ $h_{0,\ell-1}$
H_0
$h_{\frac{\ell}{2}-1,0}$ $h_{\frac{\ell}{2}-1,1}$ $\cdots$ $h_{\frac{\ell}{2}-1,\ell-1}$
$h_{\frac{\ell}{2},0}$ $h_{\frac{\ell}{2},1}$ $\cdots$ $h_{\frac{\ell}{2},\ell-1}$
$H_{\frac{\ell}{2}}$
$H_{\frac{\ell}{2}-1}$
$h_{\ell-1,0}$ $h_{\ell-1,1}$ $\cdots$ $h_{\ell-1,\ell-1}$

Figure 2.3: An overview of the heuristic kernel construction algorithm.

The idea behind our heuristic approach is to design the kernel with an even size ℓ where the Bhattacharyya parameters of the top $\ell/2$ bit-channels polarize to 1 and the Bhattacharyya parameters of the bottom bit-channels polarize to 0. We construct the kernel in two steps. First, we follow a greedy recursive construction from both top and bottom to simultaneously derive the parity check matrices of the nested kernel codes $\mathcal{C}_{\ell/2} \subset \mathcal{C}_{\ell/2+1} \subset \cdots \subset \mathcal{C}_{\ell}$ and the generating matrices of the other half the kernels codes $\mathcal{C}_0 \subset \mathcal{C}_1 \subset \cdots \subset \mathcal{C}_{\ell/2-1}$. Next, we transform the nested chain of codes to an $\ell \times \ell$ kernel K. Figure 2.3 shows a graphical overview of the heuristic algorithm. Figure 2.3.a corresponds to the first step, where we recursively construct the nested chain of kernel codes. Figure 2.3.b shows the second step, where we continue the construction of the kernel by deriving the parity check matrices of the top half.

In the following, we describe the algorithm details and then we apply it to construct some larger kernels.

We begin by pointing out that for small values of z, the dominant term in 2.23 is the non-zero term $E_{i,w}z^{w}(1-z)^{\ell-w}$ with smallest w. A faster polarization to 0 requires this w to be as large as possible so that the dominant term decays faster. The largest choice would be $w=\ell$, which can be achieved by setting the last row of the kernel as $\mathbf{1}$ (all-1 vector.) On the other hand, when z is close to 1, we prefer to maximize the coefficients $E_{i,w}$ for terms with large w's. As discussed earlier in Section 2.2.1, we can pursue this maximization by maximizing the minimum Hamming weight in coset $\mathcal{C}_1 \setminus \mathcal{C}_0$. So we put the vector $\mathbf{1}$ in $\mathcal{C}_1$ and we get the maximum value of $\mathrm{wt}_{\min}(\mathcal{C}_1 \setminus \mathcal{C}_0)=\ell$.

Let us denote the parity check matrix and the generator matrix of the code $\mathcal{C}_i$ by H_i and G_i respectively. Further let $\boxplus$ denote the sumset operation, which is also known as the Minkowski sum. The greedy construction algorithm is given as follows.

Heuristic Algorithm:

- Step 1. Construct the nested chain of kernel codes $\{\mathbf{0}\} \subset \mathcal{C}_0 \subset \cdots \subset \mathcal{C}_\ell = \{0,1\}^\ell$ from both ends by following the following rules for $i=1,\cdots,\ell/2$:

  1. Extend G_i from G_{i-1} by picking a vector g_i that maximizes

  $$d_i = wt_{\min}(\mathcal{C}_i \setminus \mathcal{C}_{i-1}) = d_{\min}\big(\{g_i\} \boxplus \mathcal{C}_{i-1}\big), \tag{2.40}$$

  while preserving the $\mathcal{C}_i \perp \mathcal{C}_{\ell-i+1}$.

  2. Extend $H_{\ell-i}$ from $H_{\ell-i+1}$ by picking a vector $h_{\ell-i}$ that maximizes

  $$d^{*}_{\ell-i} = wt_{\min}(\mathcal{C}^{\perp}_{\ell-i} \setminus \mathcal{C}^{\perp}_{\ell-i+1}) = d_{\min}\big(\{h_{\ell-i}\} \boxplus \mathcal{C}^{\perp}_{\ell-i+1}\big), \tag{2.41}$$

  while preserving the $\mathcal{C}_i \perp \mathcal{C}_{\ell-i}$.

- Step 2. For $i=\ell/2, \ell/2-1, \cdots, 1$, construct the parity check matrix of $\mathcal{C}_i$ by extending H_{i+1} with h_i. The desired kernel is then given by $K_\ell = [h_1^t|h_2^t|\cdots|h_\ell^t]^t$.

The conditions in (2.40) and (2.41) are the direct results of Theorems 4 and 5 from the previous section. To get insight about the orthogonality condition, $\mathcal{C}_i \perp \mathcal{C}_{\ell-i}$, we point out that for any $i < \ell/2$, the parity check matrix of $\mathcal{C}_i$ includes $H_{\frac{\ell}{2}}$. Hence, all of the codewords in $\mathcal{C}_i$ are orthogonal to the rows in $H_{\frac{\ell}{2}}$. So, g_i must also be orthogonal to all the rows in the bottom half of the kernel. The same argument goes for $h_{\ell-i}$.

Example 2. Now we apply the proposed method to design an 8×8 kernel. Let us first set

$$G_1 = H_7 = [1\ 1\ 1\ 1\ 1\ 1\ 1\ 1].$$

Next, we look for h_6 that has a maximal distance to $\mathbf{1}$. We can set

$$h_6 = g_2 = 00001111,$$

to achieve the optimal distance of 4. We can maintain the same distance for the next steps by

$$h_5 = g_3 = 11001100$$

$$h_4 = g_4 = 10101010.$$

It is interesting to observe that $\mathcal{C}_4$ is nothing but the well-known self-dual $(8, 4, 4)$-Hamming code, which is also known as the Reed-Muller code RM$(1, 4)$. The constructed kernel, $K_{\text{h},8}$, is presented in the following. We can derive K_8 from $K_{\text{h},8}$ with simple row operations and column permutations defined in the previous section.

$$K_{\text{h},8} = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix}.$$

Example 3. We follow a similar method to construct a 16×16 kernel. As expected, the parity check matrix of the RM$(1, 5)$ shows up after five steps of the recursion both as a parity check matrix for the $\mathcal{C}_{11}$ and a generator matrix of the code $\mathcal{C}_5$. This is shown in the following:

$$\begin{array}{rcccccccccccccccc}
g_1= & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
g_2= & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
g_3= & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\
g_4= & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 \\
g_5= & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \\
h_{11}= & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \\
h_{12}= & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 \\
h_{13}= & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\
h_{14}= & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
h_{15}= & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1
\end{array}$$

Unfortunately, we cannot continue the construction by simply picking up rows from the Reed-Muller codes. It is also computationally impractical to look at all vectors of length 16 and cross check them with the other half of the kernel to find the distance maximizer. To our fortunate luck, the quadratic bent functions come to the rescue, which are defined as the vectors with largest distance from RM codes (see [77] page 429.) Without proof, we state that there are 28 quadratic bent functions all with distance 6 from RM$(1, 5)$, which are basically the quadratic combinations of v_1, v_2, v_3, and v_4 defined below

$$v_1 = 0101010101010101 \qquad v_2 = 0011001100110011$$

$$v_3 = 0000111100001111 \qquad v_4 = 0000000011111111$$

which greatly reduces the search radius for the remaining vectors.

Table 2.4: Heuristic construction of the kernel K_{16} based on bent functions.

Step 1		Step 2	
$h_{15} = \mathbf{1}$	$g_1 = \mathbf{1}$	$h_{15} = \mathbf{1}$	$h_0 = v_1v_2v_3v_4$
$h_{14} = v_4$	$g_2 = v_4$	$h_{14} = v_4$	$h_1 = v_1v_2v_3$
$h_{13} = v_3$	$g_3 = v_3$	$h_{13} = v_3$	$h_2 = v_1v_2v_4$
$h_{12} = v_2$	$g_4 = v_2$	$h_{12} = v_2$	$h_3 = v_1v_3v_4$
$h_{11} = v_1$	$g_5 = v_1$	$h_{11} = v_1$	$h_4 = v_2v_3v_4$
$h_{10} = v_1v_3 + v_3v_4 + v_2v_4$	$g_6 = v_1v_3 + v_2v_3 + v_2v_4$	$h_{10} = v_1v_3 + v_3v_4 + v_2v_4$	$h_5 = v_2v_4$
$h_9 = v_1v_4 + v_2v_4 + v_2v_3$	$g_7 = v_1v_2 + v_2v_4 + v_3v_4$	$h_9 = v_1v_4 + v_2v_4 + v_2v_3$	$h_6 = v_3v_4$
$h_8 = v_2v_3$	$g_8 = v_3v_4$	$h_8 = v_2v_3$	$h_7 = v_1v_2 + v_3v_4$

The completed construction is tabulated in Table 2.4. The polarization behavior of the kernel K_{16} and polarization behavior are given in the following. The scaling exponent of K_{16} for BEC is computed as $\mu_{16} = 3.356$, which is no longer negligible in finite lengths.

$$K_{16} = \begin{bmatrix}
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 1 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 1 \\
0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \\
0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 \\
0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1
\end{bmatrix}.$$

Table 2.5: Polarization behavior of K_{16}.

$i\backslash w$	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
0	0	16	120	560	1820	4368	8008	11440	12870	11440	8008	4368	1820	560	120	16	1
1	0	0	64	448	1680	4256	7952	11424	12868	11440	8008	4368	1820	560	120	16	1
2	0	0	32	352	1544	4144	7896	11408	12866	11440	8008	4368	1820	560	120	16	1
3	0	0	16	208	1284	3920	7784	11376	12862	11440	8008	4368	1820	560	120	16	1
4	0	0	8	112	812	3472	7560	11312	12854	11440	8008	4368	1820	560	120	16	1
5	0	0	0	0	80	960	4752	9520	12150	11280	7992	4368	1820	560	120	16	1
6	0	0	0	0	40	480	2616	7760	11430	11120	7976	4368	1820	560	120	16	1
7	0	0	0	0	8	96	624	2608	6732	8688	7200	4224	1808	560	120	16	1
8	0	0	0	0	12	144	808	2752	6138	8832	7384	4272	1812	560	120	16	1
9	0	0	0	0	0	0	32	320	1440	3680	5392	3888	1780	560	120	16	1
10	0	0	0	0	0	0	16	160	720	1920	3256	3408	1740	560	120	16	1
11	0	0	0	0	0	0	0	0	16	128	448	896	1008	448	112	16	1
12	0	0	0	0	0	0	0	0	8	64	224	448	536	352	104	16	1
13	0	0	0	0	0	0	0	0	4	32	112	224	276	208	88	16	1
14	0	0	0	0	0	0	0	0	2	16	56	112	140	112	56	16	1
15	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

We conclude this section by pointing out that the same heuristic algorithm can be used to construct larger kernels, *e.g.* $\ell = 32$. However, the derivation of the polarization behavior itself becomes computationally challenging as ℓ grows. While this is proven to be an NP-hard problem, there have been attempts at reducing its computational complexity, see for example [78]. Furthermore, there are other methods known that are capable of estimating the scaling exponent of polar codes with large kernels other than what we used here, see for example [79]. While the heuristic construction presented in this section produces good kernels for small values of ℓ, it remains open to find an algorithm that provably produces kernels whose scaling exponents tend to the optimal value of $\mu = 2$ as ℓ grows.

2.3 Optimal Scaling of Polar Codes with Large Kernels

2.3.1 Main Theorem

Our main result provides the first family of binary codes for transmission over the BEC that achieves optimal scaling between the gap to capacity ϵ and the block length n, and that can be constructed, encoded, and decoded in quasi-linear time. In other words, the block length, construction, encoding, and decoding complexity are all bounded by a polynomial in $1/\epsilon$ and, moreover, the degree of this polynomial approaches the information-theoretic lower bound $\mu \geqslant 2$. Somewhat informally (cf. Theorem 8), this result can be stated as follows.

Theorem 7. *Consider transmission over a binary erasure channel, W with capacity $I(W)$. Fix $P_{\mathrm{e}} \in (0,1)$ and arbitrary $\delta > 0$. Then, for all $R < I(W)$, there exists a sequence of binary linear codes of rate R that guarantee error probability at most P_{e} on the channel W, and whose block length n satisfies*

$$n \leqslant \frac{\beta}{\big(I(W) - R\big)^{\mu}} \qquad \text{with} \quad \mu \leqslant 2 + \delta, \tag{2.42}$$

where $\beta = \big(1 + 2\,P_{\mathrm{e}}^{-0.01}\big)^3$ is a universal constant. Moreover, the codes in this sequence have construction complexity $\Theta(n)$ and encoding/decoding complexity $\Theta(n \log n)$.

A couple of remarks are of order. First, in the definition of the constant β, the term P_{e} is raised to the power of -0.01. We point out that we could have similarly chosen any negative constant as the exponent of P_{e}. Second, the error probability is upper-bounded by a fixed constant P_{e}. However, a somewhat stronger claim is possible. It can be shown that Theorem 7 still holds if the error probability is required to decay *polynomially fast* with the block length n.

The proof consists of three main steps. In the following, we describe the main ideas behind each of them.

Step 1: Characterization of the Bhattacharyya process. As mentioned in Section 2.1.2, when $m \to \infty$, almost all the bit-channels polarize, i.e., the process Z_m almost surely takes its value inside the set $\{0, 1\}$. In order to study the finite-length behavior of polar codes, we need to understand how fast the process Z_m polarizes. In other words, given a (small) number $\epsilon > 0$, how fast does the quantity $\mathbb{P}\{Z_m \in [\epsilon, 1-\epsilon]\}$ vanish with m? To answer this question, we first relate the decay speed of Z_m with some simpler quantity that can be directly computed from the kernel matrix K.

Recall that the Bhattacharyya process corresponding to the channel BEC(z) and the matrix K has the closed form recursive expression given by (2.6). In order to bound the value of $\mathbb{P}\{Z_m \in [\epsilon, 1-\epsilon]\}$, we look at the behavior of the process $g_\alpha(Z_m) = (Z_m(1-Z_m))^\alpha$ for $\alpha > 0$. By Markov inequality, we have

$$\mathbb{P}\{Z_m \in [\epsilon, 1-\epsilon]\} \leqslant \left(\frac{\epsilon}{2}\right)^{-\alpha} \mathbb{E}[g_\alpha(Z_m)]. \tag{2.43}$$

Furthermore, in order to bound $\mathbb{E}[g_\alpha(Z_m)]$, we can write:

$$\begin{aligned} g_\alpha(Z_m) &= (f_{B_m,K}(Z_{m-1})(1 - f_{B_m,K}(Z_{m-1})))^\alpha \\ &= (Z_{m-1}(1-Z_{m-1}))^\alpha \left(\frac{f_{B_m,K}(Z_m)(1 - f_{B_m,K}(Z_m))}{Z_{m-1}(1-Z_{m-1})}\right)^\alpha \\ &= g_\alpha(Z_{m-1}) \left(\frac{f_{B_m,K}(Z_m)(1 - f_{B_m,K}(Z_m))}{Z_{m-1}(1-Z_{m-1})}\right)^\alpha . \end{aligned} \tag{2.44}$$

Hence, after some simple calculations, we conclude that

$$\mathbb{E}[g_\alpha(Z_m)] \leqslant (\lambda^*_{\alpha,K})^m, \tag{2.45}$$

where

$$\lambda^*_{\alpha,K} = \sup_{z \in (0,1)} \frac{1}{\ell} \frac{\sum_{i=1}^{\ell} (f_{i,K}(z)(1-f_{i,K}(z)))^\alpha}{(z(1-z))^\alpha}. \tag{2.46}$$

Step 2: Sharpness of the one-step erasure probabilities. We show that

$$\lambda^*_{\alpha,K} \leqslant \ell^{-1/2+5\alpha}, \tag{2.47}$$

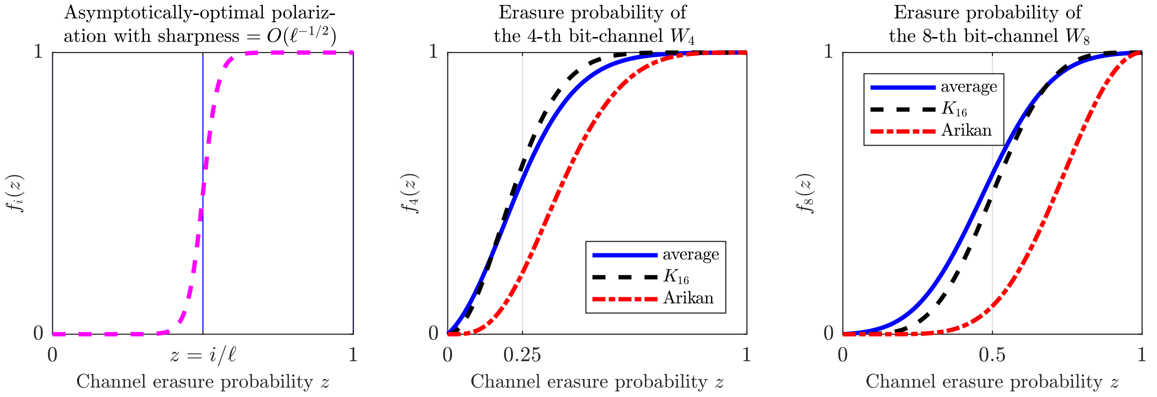


Figure 2.4: The erasure probabilities of polar bit-channels for K_{16}.

with probability $1 - O(1/\ell)$ over the random choice of K. To do so, we prove that, as ℓ grows, the functions $f_{i,K}$ will behave as step functions for most of the non-singular kernels. Note that, for any i and for any K, $f_{i,K}$ is an increasing polynomial with $f_{i,K}(0) = 0$ and $f_{i,K}(1) = 1$. As ℓ increases, we prove that, with probability $1 - O(1/\ell)$ over the choice of K, $f_{i,K}(z)$ has a sharp threshold around the point $z = i/\ell$. More precisely,

$$\begin{aligned} f_{i,K}(z) &\leqslant \ell^{-(2+\log \ell)}, &\quad \text{for } z \leqslant \frac{i}{\ell} - 5\ell^{-1/2}\log \ell, \\ f_{i,K}(z) &\geqslant 1 - \ell^{-(2+\log \ell)}, &\quad \text{for } \geqslant \frac{i}{\ell} + 5\ell^{-1/2}\log \ell, \end{aligned} \tag{2.48}$$

see also Figure 2.4. The figure on the left shows that $f_{i,K}(z)$ exhibits a sharp transition of order roughly $O(\ell^{-1/2})$, when the kernel K is random. The two figures on the right compare different choices of the kernel K: the red curve corresponds to Arıkan's kernel; the black curve to the kernel K_{16} from the previous section; and the blue curve is obtained by taking the average of the functions $f_{i,K}(z)$ for a random kernel. Let us now go back to (2.46) and use this "sharpness" property of $f_{i,K}$ in order to upper bound $\lambda^*_{\alpha,K}$. In the right-hand-side of (2.46), let us only evaluate the term inside the supremum for $z = 1/2$. By using the *sharpness* property, it is not hard to see that this term will be of order

$$\ell^{-1/2}\log \ell + \ell^{-\alpha(2+\log \ell)} \leqslant \ell^{-1/2+5\alpha}, \tag{2.49}$$

for sufficiently large ℓ. With some more effort, we can show a bound which is valid for any $z \in (0,1)$ (and not only for $z = 1/2$).

Step 3: Finite-length scaling law. We derive a finite-length scaling law for polar codes by using the results of the previous steps. From (2.43), (2.45), and the upper bound on $\lambda^*_{\alpha,K}$, we conclude that

$$\mathbb{P}\{Z_m \in [\epsilon, 1-\epsilon]\} = O\big(\epsilon^{-\alpha}(\ell^{-1/2+5\alpha})^m\big). \tag{2.50}$$

Denote the desired error probability by P_e and let $\epsilon = P_\mathrm{e}\ell^{-m}$. Then,

$$\mathbb{P}\{Z_m \in [P_\mathrm{e}\ell^{-m}, 1 - P_\mathrm{e}\ell^{-m}]\} = O(\ell^{m/(2+\delta)}), \tag{2.51}$$

where δ can be made arbitrarily small by choosing a small enough α. As the blocklength n is equal to ℓ^m, (2.51) implies that the gap to capacity is of order $n^{1/(2+\delta)}$. By bounding also $\mathbb{P}\{Z_m \geqslant 1 - P_\mathrm{e}\ell^{-m}\}$, the desired scaling result follows.

2.3.2 Proof of the Main Theorem

As mentioned in the preliminary topics the main result of this chapter is to provide a family of binary codes that achieves optimal scaling between gap to capacity and block length, as well as quasi-linear complexity of construction, encoding and decoding. This is done by showing that binary polar codes obtained from large kernels possess those properties.

Theorem 8. [Binary Polar Codes with Optimal Scaling and Quasi-Linear Complexity] *Consider the transmission over a BEC W with capacity $I(W)$. Let $K \in \mathbb{F}_2^{\ell\times\ell}$ be a kernel that is selected uniformly at random among all $\ell \times \ell$ non-singular binary matrices. Fix $P_\mathrm{e} \in (0,1)$ and let $\mathcal{C}_\ell(n, R, P_\mathrm{e})$ be the code obtained by polarizing K with block length $n = \ell^m$ for some $m \in \mathbb{N}$ and rate $R < I(W)$ such that the error probability under successive cancellation* *decoding is at most P_{e}. Fix a small constant $\delta > 0$. Then, there exists $\ell_0(\delta)$ such that for any $\ell > \ell_0(\delta)$, with high probability over the choice of K, there is a code $\mathcal{C}_\ell(n, R, P_{\mathrm{e}})$ that satisfies*

$$n \leqslant \frac{\beta}{(I(W) - R)^\mu}, \qquad \textit{with} \quad \mu \leqslant 2 + \delta, \tag{2.52}$$

where β is a constant given by $(1 + 2\,P_{\mathrm{e}}^{-0.01})^3$. This code has construction complexity $\Theta(n)$, and encoding/decoding complexity $\Theta(n \log n)$.

It also possible to show that ℓ needs to be of order $\exp{(1/\delta^{1.01})}$, and additional details about this fact are provided at the end of the section. The theorem above follows from the following result that characterizes the behavior of the polarization process.

Theorem 9. [Optimal Scaling of Polarization Process] *Let $K \in \mathbb{F}_2^{\ell \times \ell}$ be a kernel that is selected uniformly at random among all $\ell \times \ell$ non-singular binary matrices. Let Z_m be the random process defined in* (2.6) *with initial condition $Z_0 = z$. Fix $P_{\mathrm{e}} \in (0, 1)$ and a small constant $\delta > 0$. Then, there exists $\ell_0(\delta)$ such that, with high probability over the choice of K, for any $\ell > \ell_0(\delta)$ and for any $m \geqslant 1$*

$$\mathbb{P}\{Z_m \leqslant P_{\mathrm{e}} \ell^{-m}\} \geqslant 1 - z - c_0 \ell^{-\frac{m}{\mu}}, \tag{2.53}$$

with

$$\mu(K) \leqslant 2 + \delta, \tag{2.54}$$

and where c_0 is a constant given by $1 + 2\,P_{\mathrm{e}}^{-0.01}$.

For the sake of clarity, note that in (2.53) the kernel K is fixed and the probability space is defined with respect to the random process Z_m, while (2.54) holds with high probability over the choice of the kernel K. We are now ready to present the proof of Theorem 8.

Proof of Theorem 8. Consider the transmission over the BEC(z) of a polar code with block length $n = \ell^m$ and rate R obtained by polarizing the $\ell \times \ell$ kernel K, where $\ell > \ell_0(\delta)$. By Theorem 9, there is at least a fraction $1 - z - c_0\ell^{-\frac{m}{\mu(K)}}$ of the synthetic channels whose erasure probability is at most $P_{\rm e}\ell^{-m}$, where $c_0 = 1 + 2\,P_{\rm e}^{-0.01}$ and, with high probability over the choice of K, $\mu(K) \leqslant 2 + \delta$. Then, if we take

$$R = 1 - z - (1 + 2\,P_{\rm e}^{-0.01})\ell^{-\frac{m}{\mu(K)}}, \tag{2.55}$$

a simple union bound yields that the error probability under successive cancellation decoding is at most $P_{\rm e}$. As $I(W) = 1 - z$, by re-arranging (2.55), formula (2.52) immediately follows with $\beta = (1 + 2\,P_{\rm e}^{-0.01})^{2+\delta}$. Without loss of generality, we can assume that $\delta < 1$, hence we can take β as prescribed by the statement of the theorem. The claim on the construction complexity follows from the fact that the erasure probabilities of the synthetic channels can be computed exactly according to the recursion (2.6). The claim on the encoding/decoding complexity follows from [29, 54]. □

The rest of the section is devoted to prove Theorem 9. The basic idea consists in bounding the number of un-polarized synthetic channel. To do so, let us define the polarization measure function $g_\alpha(z)$ as

$$g_\alpha(z) \triangleq \big(z(1-z)\big)^\alpha, \tag{2.56}$$

where $\alpha \in (0, 1)$ is a fixed parameter. The first step is to show that an upper bound on $\mathbb{E}[g_\alpha(Z_m)]$ yields an lower bound on $\mathbb{P}\{Z_m \leqslant P_{\rm e}\ell^{-m}\}$. This is done in Lemma 2.6, whose statement and proof immediately follow.

Lemma 2.6. *Let $K \in \mathbb{F}_2^{\ell\times\ell}$ be an $\ell \times \ell$ non-singular binary kernel such that none of its column permutations is upper triangular. Let Z_m be the random process defined in* (2.6) *with initial* *condition* $Z_0 = z$. *Fix* $\alpha \in (0,1)$ *and define* $g_\alpha(z)$ *as in* (2.56). *Fix* ρ, $P_{\mathrm{e}} \in (0,1)$ *and assume that, for any* $m \geqslant 1$,

$$\mathbb{E}[g_\alpha(Z_m)] \leqslant \ell^{-m\rho}. \tag{2.57}$$

Then, for any $m \geqslant 1$,

$$\mathbb{P}\{Z_m \leqslant P_{\mathrm{e}}\ell^{-m}\} \geqslant 1 - z - c_1\ell^{-m(\rho-\alpha)}, \tag{2.58}$$

where $c_1 = 2P_{\mathrm{e}}^{-\alpha} + P_{\mathrm{e}}$.

Proof. First of all, we upper bound $\mathbb{P}\{Z_m \in [P_{\mathrm{e}}\ell^{-m}, 1 - P_{\mathrm{e}}\ell^{-m}]\}$ as follows:

$$\begin{aligned}\mathbb{P}\left\{Z_m \in \left[P_{\mathrm{e}}\ell^{-m}, 1 - P_{\mathrm{e}}\ell^{-m}\right]\right\} &\overset{\text{(a)}}{=} \mathbb{P}\left\{g_\alpha(Z_m) \geqslant g_\alpha(P_{\mathrm{e}}\ell^{-m})\right\} \\ &\overset{\text{(b)}}{\leqslant} \frac{\mathbb{E}[g_\alpha(Z_m)]}{g_\alpha(P_{\mathrm{e}}\ell^{-m})} \\ &\overset{\text{(c)}}{\leqslant} \frac{\ell^{-m\rho}}{g_\alpha(P_{\mathrm{e}}\ell^{-m})} \\ &\overset{\text{(d)}}{\leqslant} 2\,P_{\mathrm{e}}^{-\alpha}\,\ell^{-m(\rho-\alpha)},\end{aligned} \tag{2.59}$$

where the equality (a) uses the concavity of the function $g_\alpha(\cdot)$; the inequality (b) follows from Markov inequality; the inequality (c) uses the hypothesis $\mathbb{E}[g_\alpha(Z_m)] \leqslant \ell^{-m\rho}$; and the inequality (d) uses that $1 - P_{\mathrm{e}}\ell^{-m} \geqslant 1/2$ for any $m \geqslant 1$.

Let us define

$$\begin{aligned}A &= \mathbb{P}\left\{Z_m \in \left[0, P_{\mathrm{e}}\ell^{-m}\right)\right\}, \\ B &= \mathbb{P}\left\{Z_m \in \left[P_{\mathrm{e}}\ell^{-m}, 1 - P_{\mathrm{e}}\ell^{-m}\right]\right\}, \\ C &= \mathbb{P}\left\{Z_m \in \left(1 - P_{\mathrm{e}}\ell^{-m}, 1\right]\right\},\end{aligned} \tag{2.60}$$

and let A', B', and C' be the fraction of synthetic channels in A, B, and C, respectively, that will have a vanishing erasure probability as $n \to \infty$. More formally,

$$\begin{aligned} A' &= \liminf_{m' \to \infty} \mathbb{P}\left\{ Z_m \in \left[0, P_{\mathrm{e}} \ell^{-m}\right), Z_{m+m'} \leqslant \ell^{-m'} \right\}, \\ B' &= \liminf_{m' \to \infty} \mathbb{P}\left\{ Z_m \in \left[P_{\mathrm{e}} \ell^{-m}, 1 - P_{\mathrm{e}} \ell^{-m}\right], Z_{m+m'} \leqslant \ell^{-m'} \right\}, \\ C' &= \liminf_{m' \to \infty} \mathbb{P}\left\{ Z_m \in \left(1 - P_{\mathrm{e}} \ell^{-m}, 1\right], Z_{m+m'} \leqslant \ell^{-m'} \right\}. \end{aligned} \tag{2.61}$$

Recall that any $\ell \times \ell$ non-singular binary matrix none of whose column permutations is upper triangular polarizes symmetric channels [29]. Hence, as K satisfies this condition by hypothesis, we immediately have that

$$A' + B' + C' = \liminf_{m' \to \infty} \mathbb{P}\left\{ Z_{m+m'} \leqslant \ell^{-m'} \right\} = 1 - z. \tag{2.62}$$

In addition, from (2.59), we have that

$$B' \leqslant B \leqslant 2 P_{\mathrm{e}}^{-\alpha} \ell^{-m(\rho - \alpha)}. \tag{2.63}$$

In order to upper bound C', we proceed as follows:

$$\begin{aligned} C' &= \liminf_{m' \to \infty} \mathbb{P}\left\{ Z_{m+m'} \leqslant \ell^{-m'} \mid Z_m \in \left(1 - P_{\mathrm{e}} \ell^{-m}, 1\right] \right\} \cdot \mathbb{P}\left\{ Z_m \in \left(1 - P_{\mathrm{e}} \ell^{-m}, 1\right] \right\} \\ &\leqslant \liminf_{m' \to \infty} \mathbb{P}\left\{ Z_{m+m'} \leqslant \ell^{-m'} \mid Z_m \in \left(1 - P_{\mathrm{e}} \ell^{-m}, 1\right] \right\}. \end{aligned} \tag{2.64}$$

By using again that the kernel K is polarizing, we obtain that the last term equals the capacity of a BEC with erasure probability at least $1 - P_{\mathrm{e}} \ell^{-m}$. Consequently,

$$C' \leqslant P_{\mathrm{e}} \ell^{-m}. \tag{2.65}$$

As a result, we conclude that

$$\begin{aligned} \mathbb{P}\left\{ Z_m \in \left[0, P_{\mathrm{e}} \ell^{-m}\right) \right\} = A \geqslant A' \overset{\text{(a)}}{=} 1 - z - B' - C' &\overset{\text{(b)}}{\geqslant} 1 - z - 2 P_{\mathrm{e}}^{-\alpha} \ell^{-m(\rho-\alpha)} - P_{\mathrm{e}} \ell^{-m}, \\ &\overset{\text{(c)}}{\geqslant} 1 - z - \left(2 P_{\mathrm{e}}^{-\alpha} + P_{\mathrm{e}}\right) \ell^{-m(\rho-\alpha)}, \end{aligned}$$

where the equality (a) uses (2.62); the inequality (b) uses (2.63) and (2.65); and the inequality (c) uses that α and $\rho \in (0, 1)$. This chain of inequalities implies the desired result. □

The second step consists in giving an upper bound on $\mathbb{E}[g_\alpha(Z_m)]$ of the form $(\lambda^*_{\alpha,K})^m$, where $\lambda^*_{\alpha,K}$ depends on the particular kernel K. This is done in Lemma 2.7, whose statement and proof immediately follow.

Lemma 2.7. *Let $K \in \mathbb{F}_2^{\ell\times\ell}$ be an $\ell \times \ell$ binary kernel. Let Z_m be the random process defined in* (2.6) *with initial condition $Z_0^K = z$. Fix $\alpha \in (0,1)$ and define $g_\alpha(z)$ as in* (2.56)*. For $z \in (0,1)$, define $\lambda_{\alpha,K}(z)$ as*

$$\lambda_{\alpha,K}(z) \triangleq \frac{\frac{1}{\ell}\sum_{i=1}^{\ell} g_\alpha(f_{i,K}(z))}{g_\alpha(z)}, \tag{2.66}$$

and let $\lambda^*_{\alpha,K}$ be its supremum, i.e.,

$$\lambda^*_{\alpha,K} \triangleq \sup_{z\in(0,1)} \lambda_{\alpha,K}(z). \tag{2.67}$$

Then, for any $m \geqslant 0$, we have that

$$\mathbb{E}[g_\alpha(Z_m)] \leqslant (\lambda^*_{\alpha,K})^m g_\alpha(z). \tag{2.68}$$

Proof. We prove the claim by induction. The base step $m = 0$ follows immediately from the fact that $Z_0 = z$. To prove the inductive step, first we write

$$\mathbb{E}\big[g_\alpha(Z_{m+1})\big] = \mathbb{E}\big[\mathbb{E}[g_\alpha(f_{B_m,K}(Z_m)) \mid Z_m]\big],$$

where the first expectation on the RHS is with respect to Z_m and the second expectation is with respect to B_m. Then, we have that

$$\begin{aligned}\mathbb{E}\big[\mathbb{E}[g_\alpha(f_{B_m,K}(Z_m)) \mid Z_m]\big] &= \mathbb{E}\left[g_\alpha(Z_m)\frac{\frac{1}{\ell}\sum_{i=1}^{\ell} g_\alpha(f_{i,K}(Z_m))}{g_\alpha(Z_m)}\right]\\ &\leqslant \mathbb{E}\big[g_\alpha(Z_m)\big] \underbrace{\sup_{z\in\{0,1\}} \frac{\frac{1}{\ell}\sum_{i=1}^{\ell} g_\alpha(f_{i,K}(z))}{g_\alpha(z)}}_{\lambda^*_{\alpha,K}},\end{aligned}$$

which concludes the proof. □

The third and final step is to prove that $\lambda^*_{\alpha,K}$ concentrates around $1/\sqrt{\ell}$, when K is selected uniformly at random among all $\ell \times \ell$ non-singular binary matrices. This is done in Theorem 10 that is stated below and whose proof is presented in the next Section.

Theorem 10. *Let $K \in \mathbb{F}_2^{\ell\times\ell}$ be a kernel that is selected uniformly at random among all $\ell \times \ell$ non-singular binary matrices. Fix $\alpha \in (0, 1/16)$ and define $\lambda^*_{\alpha,K}$ as in* (2.67)*. Then, there exists $\ell_1(\alpha)$ such that, for any $\ell > \ell_1(\alpha)$,*

$$\mathbb{P}\{\log_\ell(\lambda^*_{\alpha,K}) \leqslant -\frac{1}{2} + 5\alpha\} \geqslant 1 - \frac{2}{\ell}, \tag{2.69}$$

where the probability space is over the choice of the kernel K.

At this point, we are ready to put everything together and give the proof of Theorem 9.

Proof of Theorem 9. Pick ℓ sufficiently large and define

$$\alpha = \min\left(\frac{\delta}{12(2+\delta)}, \frac{1}{100}\right). \tag{2.70}$$

Then, by Theorem 10, we have that, with high probability over the choice of the kernel K,

$$\lambda^*_{\alpha,K} \leqslant \ell^{-(1/2-5\alpha)}. \tag{2.71}$$

Consequently, as $g_\alpha(z) \leqslant 1$ for any $z \in (0, 1)$, by Lemma 2.7 we have that

$$\mathbb{E}[g_\alpha(Z_m)] \leqslant \ell^{-m(1/2-5\alpha)}. \tag{2.72}$$

Note that, with high probability, the kernel K is such that none of its column permutations is upper triangular. Then, we can apply Lemma 2.6 and we deduce that

$$\mathbb{P}\{Z_m \leqslant P_\mathrm{e}\ell^{-m}\} \geqslant 1 - z - c_1\ell^{-m(1/2-6\alpha)}, \tag{2.73}$$

where $c_1 = 2P_\mathrm{e}^{-\alpha} + P_\mathrm{e}$. Note that, as $\alpha \leqslant 1/100$ and $P_\mathrm{e} \leqslant 1$, we have that $c_1 \leqslant 1 + 2P_\mathrm{e}^{-0.01}$. By plugging in (2.73) the choice of α given by (2.70), the thesis immediately follows. □

2.4 Concentration of Scaling Exponent for Large Kernels

2.4.1 Concentration Theorem

Let us recall that the goal is to show that for most non-singular binary kernels $K \in \mathbb{F}_2^{\ell\times\ell}$ and for $\ell > \ell_1(\alpha)$,

$$\lambda_{\alpha,K}(z) \leqslant \ell^{-\frac{1}{2}+5\alpha} \quad \forall z \in (0,1), \tag{2.74}$$

where $\alpha \in (0,1)$ is fixed and $\ell_1(\alpha)$ is a large integer that depends only on α.

Our strategy is to split the interval $(0,1)$ into the three sub-intervals $(0,1/\ell^2)$, $[1/\ell^2, 1-1/\ell^2]$, and $(1-1/\ell^2, 1)$. Then, we will show that (2.74) holds for each of these sub-intervals. In fact, as we will see, polarization is much faster at the tail intervals. Theorem 11 captures this approach.

Theorem 11. [Concentration of $\lambda^*_{\alpha,K}$] *Let $K \in \mathbb{F}_2^{\ell\times\ell}$ be a kernel that is selected uniformly at random among all $\ell\times\ell$ non-singular binary matrices. Fix $\alpha \in (0,1/16)$ and define $\lambda_{\alpha,K}(z)$ as in* (2.66). *Let $\ell_1(\alpha)$ be the smallest integer such that*

$$\frac{\log \ell_1(\alpha)}{\log\log \ell_1(\alpha)} \geqslant \frac{1}{\alpha}. \tag{2.75}$$

Then, for all $\ell \geqslant \ell_1(\alpha)$, the following results hold.

1. *Near optimal polarization in the middle:*

$$\mathbb{P}\left\{\lambda_{\alpha,K}(z) < \ell^{-\frac{1}{2}+5\alpha},\ \forall z \in \left[\frac{1}{\ell^2}, 1-\frac{1}{\ell^2}\right]\right\} > 1-\frac{1}{\ell}, \tag{2.76}$$

2. *Faster polarization in the tails:*

$$\mathbb{P}\left\{\lambda_{\alpha,K}(z) < \ell^{-\frac{1}{2}},\ \forall z \in \left(0, \frac{1}{\ell^2}\right) \cup \left(1-\frac{1}{\ell^2}, 1\right)\right\} > 1-\frac{1}{\ell}, \tag{2.77}$$

where the probability spaces are over the choice of the kernel K.

The proof of Theorem 10 immediately follows from the result above.

Proof of Theorem 10. By applying the union bound on (2.76) and (2.77), we obtain that

$$\mathbb{P}\left\{\lambda_{\alpha,K}(z) < \ell^{-\frac{1}{2}+5\alpha},\ \ \forall z \in (0,1)\right\} > 1 - \frac{2}{\ell}, \tag{2.78}$$

which yields the desired result. □

In the following, we introduce the average polarization behavior and provide some auxiliary lemmas. Eventually, we provide the proof of Theorem 11.

2.4.2 Average Polarization Behavior

Here, we use the condition (2.19) and we give an explicit formula for the probability that u_i is decodable, i.e., for $\mathbb{P}\{(E_i \setminus E_{i-1}) \cap (\text{column space of } K_{[:,1:\ell-s]}) \neq \emptyset\}$, while K is selected uniformly at random.

For $i \in [\ell]$, define the *average erasure probability* $F_i(z)$ as

$$F_i(z) \triangleq \mathbb{E}_K[f_{i,K}(z)] = \frac{\sum_K f_{i,K}(z)}{\tau_\ell}, \tag{2.79}$$

where τ_ℓ denotes the number of non-singular $\ell \times \ell$ binary matrices, i.e.,

$$\tau_\ell = \prod_{j=0}^{\ell-1}(2^\ell - 2^j). \tag{2.80}$$

Then, it is easy to verify that $F_i(z)$ represents the erasure probability of the i-th bit-channel given that *(i)* the kernel is selected uniformly at random among the $\ell \times \ell$ nonsingular binary matrices, and *(ii)* the transmission channel is a BEC(z).

Next, we analyze the asymptotic behavior of $F_i(z)$ and show that, as ℓ becomes large, $F_i(z)$ becomes close to a step function with jump at $z \sim {}^{i}\!/\!_{\ell}$. We then prove some concentration results to show that, with high probability over the choice of the kernel K, $f_{i,K}(z)$ is also close to a sharp step function centered around $z \sim {}^{i}\!/\!_{\ell}$.

We first recall that $F_i(z)$ is the probability of observing an erasure at the i-th bit-channel, where there are two sources of randomness: *(i)* the selection of the kernel, and *(ii)* the number and location of the erased received symbols. Let the random variable S denote the number of erased symbols at the receiver. As z is the erasure probability of the transmission channel, we have that

$$\mathbb{P}\{S = s\} = \binom{\ell}{s} z^s (1-z)^{\ell - s}. \tag{2.81}$$

Since we also average over all $\ell \times \ell$ non-singular kernels, the location of these s erasures does not affect the average erasure probability of the bit-channels. Hence, without loss of generality, we can assume that the erasures happened at the last coordinates. Let $\mathcal{R}_{\ell-s} \subset \mathbb{F}_2^\ell$ denote the linear span of the first $\ell - s$ columns of the kernel. Since the kernel is selected uniformly at random, it is easy to see that $\mathcal{R}_{\ell-s}$ is also chosen uniformly at random from all subspaces of dimension $\ell - s$ in $\mathbb{F}_2^\ell$. Recalling the decodability condition in (2.19), we have that

$$\mathbb{P}\{u_i = \text{erasure} | S = s\} = \mathbb{P}\{\mathcal{R}_{\ell-s} \cap (E_i \setminus E_{i-1}) = \emptyset\}, \tag{2.82}$$

where $\mathcal{R}_{\ell-s}$ is a subspace of dimension $\ell - s$ in $\mathbb{F}_2^\ell$ that is chosen uniformly at random. The probability space on the LHS is defined with respect to *(i)* the location of the s erasures, and *(ii)* the selection of the random kernel K, while the probability space on the RHS is defined with respect to just the selection of random subspace $\mathcal{R}_{\ell-s}$. Now we can rewrite $F_i(z)$ as

$$F_i(z) = \sum_{s=0}^{\ell} \mathbb{P}\{S = s\}\mathbb{P}\{u_i = \text{erasure}|S = s\} = \sum_{s=0}^{\ell} \binom{\ell}{s} z^s (1-z)^{\ell-s} p_{i|s}, \tag{2.83}$$

where we define the *average conditional erasure probability* $p_{i|s}$ as

$$p_{i|s} \triangleq \mathbb{P}\{\mathcal{R}_{\ell-s} \cap (E_i \setminus E_{i-1}) = \emptyset\}. \tag{2.84}$$

The following lemma provides a closed-form expression for $p_{i|s}$.

Lemma 2.8 (Closed-Form for Average Conditional Erasure Probability)**.** *Let $p_{i|s}$ be the average conditional erasure probability defined in* (2.84)*. Then, for any i and s,*

$$p_{i|s} = \begin{bmatrix} \ell \\ \ell - s \end{bmatrix}^{-1} \sum_{t=\max\{i-s,0\}}^{\min\{\ell-s,i-1\}} \begin{bmatrix} i-1 \\ t \end{bmatrix} \prod_{j=0}^{\ell-s-t-1} \frac{2^{\ell} - 2^{i+j}}{2^{\ell-s} - 2^{t+j}}, \tag{2.85}$$

where $\begin{bmatrix} a \\ b \end{bmatrix}$ is the binary Gaussian binomial coefficient.

Proof. Let $\Delta_{\ell-s}$ be the total number of subspaces in $\mathbb{F}_2^{\ell}$ with dimension $\ell - s$. Then,

$$\Delta_{\ell-s} = \begin{bmatrix} \ell \\ \ell - s \end{bmatrix} \triangleq \prod_{j=0}^{\ell-s-1} \frac{2^{\ell} - 2^{j}}{2^{\ell-s} - 2^{j}}. \tag{2.86}$$

Define $\Gamma_t^{\ell-s,i}$ as the number of subspaces A of dimension $\ell - s$ in $\mathbb{F}_2^{\ell}$ such that $A \cap (E_i \setminus E_{i-1}) = \emptyset$ and $\dim(A \cap E_{i-1}) = t$. Equivalently, $\Gamma_t^{\ell-s,i}$ represents the number of subspaces A of dimension $\ell - s$ in $\mathbb{F}_2^{\ell}$ such that $\dim(A \cap E_{i-1}) = \dim(A \cap E_i) = t$. Consequently, the integer t in the definition of $\Gamma_t^{\ell-s,i}$ satisfies the following conditions:

$$\max\{i - s, 0\} \leqslant t \leqslant \min\{\ell - s, i - 1\}. \tag{2.87}$$

A simple basis counting argument (see [80]) yields that

$$\Gamma_t^{\ell-s,i} = \underbrace{\begin{bmatrix} i-1 \\ t \end{bmatrix}}_{\substack{\text{number of subspaces} \\ \text{in } E_{i-1} \text{ with } \dim = t}} \times \underbrace{\prod_{j=0}^{\ell-s-t-1} \frac{2^{\ell} - 2^{i+j}}{2^{\ell-s} - 2^{t+j}}}_{\substack{\text{normalized number of basis extensions} \\ \text{from } \dim = t \text{ to } \dim = \ell - s}}. \tag{2.88}$$

Then, the desired conditional erasure probability can be written as

$$p_{i|s} = \frac{\sum_{t=\max\{i-s,0\}}^{\min\{\ell-s,i-1\}} \Gamma_t^{\ell-s,i}}{\Delta_{\ell-s}}. \tag{2.89}$$

The thesis immediately follows from (2.88) and (2.89). □

Now, we use this closed-form expression to provide lower and upper bounds on the average conditional erasure probability $p_{i|s}$ and on the average erasure probability $F_i(z)$.

Lemma 2.9 (Lower Bound for Average Conditional Erasure Probability)**.** *Let $p_{i|s}$ be the average conditional erasure probability defined in* (2.84)*. Then, for any i and s,*

$$p_{i|s} \geqslant 1 - 2^{-(s-i)}. \tag{2.90}$$

Proof. If $i \geqslant s$, then the proof is trivial. Hence, let us assume that $i < s$. We drop all but the first term from (2.89) to write

$$p_{i|s} = \frac{\sum\limits_{t=0}^{\min\{\ell-s,i-1\}} \Gamma_t^{\ell-s,i}}{\Delta_{\ell-s}} \geqslant \frac{\Gamma_0^{\ell-s,i}}{\Delta_{\ell-s}} = \frac{\prod\limits_{j=0}^{\ell-s-1} \dfrac{2^\ell - 2^{i+j}}{2^{\ell-s} - 2^j}}{\prod\limits_{j=0}^{\ell-s-1} \dfrac{2^\ell - 2^j}{2^{\ell-s} - 2^j}} = \prod_{j=0}^{\ell-s-1} \frac{2^\ell - 2^{i+j}}{2^\ell - 2^j}. \tag{2.91}$$

The remainder of the proof is derived by simple algebra as follows

$$\begin{aligned}\prod_{j=0}^{\ell-s-1} \frac{2^\ell - 2^{i+j}}{2^\ell - 2^j} > \prod_{j=0}^{\ell-s-1} \frac{2^\ell - 2^{i+j}}{2^\ell} &= \prod_{j=0}^{\ell-s-1} \left(1 - 2^{-(\ell-i)+j}\right) \\ &\geqslant 1 - \sum_{j=0}^{\ell-s-1} 2^{-(\ell-i)+j} > 1 - 2^{-(s-i)}.\end{aligned} \tag{2.92}$$

□

Lemma 2.10 (Lower Bound for Average Erasure Probability)**.** *Let $F_i(z)$ be the average erasure probability of the i-th bit-channel as defined in (2.79). Fix $\beta, \delta \in \mathbb{R}^+$ and assume that*

$$z > \frac{i}{\ell} + \frac{\lceil \delta \log \ell \rceil}{\ell} + \left(\frac{\beta \ln \ell}{2\ell}\right)^{1/2}, \tag{2.93}$$

where $\log$ *and* $\ln$ *denote the logarithm in base* 2 *and* e*, respectively. Then, we have that*

$$F_i(z) > (1 - \ell^{-\beta})(1 - \ell^{-\delta}). \tag{2.94}$$

Proof. We begin by dropping the first $i + \lceil \delta \log \ell \rceil$ terms in (2.83) and applying the lower bound from (2.90):

$$\begin{aligned} F_i(z) = \sum_{s=0}^{\ell} \binom{\ell}{s} z^s (1-z)^{\ell-s} p_{i|s} &> \sum_{s=i+\lceil \delta \log \ell \rceil}^{\ell} \binom{\ell}{s} z^s (1-z)^{\ell-s} (1 - 2^{-(s-i)}) \\ &\geqslant (1 - \ell^{-\delta}) \sum_{s=i+\delta \lceil \log \ell \rceil}^{\ell} \binom{\ell}{s} z^s (1-z)^{\ell-s}. \end{aligned} \tag{2.95}$$

Now, we point out that the sum on the RHS of (2.95) is the tail probability of a binomial distribution with ℓ trials and a success rate of z. More formally, let $X \sim B(\ell, z)$. Then, from (2.95) we immediately obtain that

$$F_i(z) > (1 - \ell^{-\delta}) \mathbb{P}\{X \geqslant i + \lceil \delta \log \ell \rceil\}. \tag{2.96}$$

Furthermore,

$$\begin{aligned} \mathbb{P}\{X \geqslant i + \lceil \delta \log \ell \rceil\} &= 1 - \mathbb{P}\{X < i + \lceil \delta \log \ell \rceil\} \\ &\overset{\text{(a)}}{\geqslant} 1 - \exp\left(-2 \frac{\big(z\ell - (i + \lceil \delta \log \ell \rceil)\big)^2}{\ell} \right) \\ &\overset{\text{(b)}}{\geqslant} 1 - \ell^{-\beta}, \end{aligned} \tag{2.97}$$

where in (a) we use Hoeffding's inequality and in (b) we use (2.93). By combining (2.95) with (2.97), the claim readily follows. □

First, we use the closed-form expression in order find a lower bound on the average conditional erasure probability and on the average erasure probability.

Lemma 2.11 (Upper Bound for Average Conditional Erasure Probability)**.** *Let $p_{i|s}$ be the average conditional erasure probability defined in* (2.84)*. Then, for any i and s,*

$$p_{i|s} \leqslant 2 \left(\frac{2}{3} \right)^{i-s-1}. \tag{2.98}$$

Proof. If $s \geqslant i-1$, then the proof is trivial. Hence, let us assume that $s < i-1$. We start by proving that the term with $t = i-s$ is the dominant one in the expression (2.89) for $p_{i|s}$. For all $t > i-s$, we have that

$$\frac{\Gamma_t^{\ell-s,i}}{\Gamma_{t-1}^{\ell-s,i}} = \frac{\begin{bmatrix} i-1 \\ t \end{bmatrix}}{\begin{bmatrix} i-1 \\ t-1 \end{bmatrix}} \times \Bigg(\prod_{j=0}^{\ell-s-t-1} \frac{2^\ell - 2^{i+j}}{2^{\ell-s} - 2^{t+j}} \Bigg) \Big/ \Bigg(\prod_{j=0}^{\ell-s-t} \frac{2^\ell - 2^{i+j}}{2^{\ell-s} - 2^{t+j-1}} \Bigg),$$

which using simple algebra can be simplified as

$$\frac{(2^{i-1} - 2^{t-1})(2^{\ell-s} - 2^{t-1})}{2^{t-1}(2^t - 1)(2^\ell - 2^{i+\ell-s-t})} \leqslant \frac{1}{2^{t-1}} \times \frac{2^{i-1} \times 2^{\ell-s}}{2^{t-1} \times 2^{\ell-1}} = \frac{2^{i-s-t+1}}{2^{t-1}} \leqslant 2^{-t+1} \leqslant \frac{1}{2}.$$

Therefore, we have that, for any $t > i-s$,

$$\Gamma_t^{\ell-s,i} \leqslant 2^{-\big(t-(i-s)\big)}\, \Gamma_{i-s}^{\ell-s,i},$$

which implies that

$$p_{s|i} \leqslant \frac{\Gamma_{i-s}^{\ell-s,i}}{\Delta_{\ell-s}} \big(1 + 2^{-1} + 2^{-2} + \cdots\big) \leqslant \frac{2\Gamma_{i-s}^{\ell-s,i}}{\Delta_{\ell-s}}. \tag{2.99}$$

In a similar fashion, we fix ℓ and i, and we show the exponential decay of the dominant term in $p_{i|s}$, denoted by $\zeta_s \triangleq \Gamma_{i-s}^{\ell-s,i}/\Delta_{\ell-s}$, as s decreases. We again use simple algebra to obtain

$$\begin{aligned}
\frac{\zeta_s}{\zeta_{s+1}} &= \frac{\Delta_{\ell-s-1}}{\Delta_{\ell-s}} \times \frac{\begin{bmatrix} i-1 \\ i-s \end{bmatrix}}{\begin{bmatrix} i-1 \\ i-s-1 \end{bmatrix}} \times \Bigg(\prod_{j=0}^{\ell-i-1} \frac{2^\ell - 2^{i+j}}{2^{\ell-s} - 2^{i-s+j}} \Bigg) \Big/ \Bigg(\prod_{j=0}^{\ell-i-1} \frac{2^\ell - 2^{i+j}}{2^{\ell-s-1} - 2^{i-s-1+j}} \Bigg) \\
&= \frac{(2^{i-1} - 2^{i-s-1})(2^{\ell-s} - 1)}{(2^{i-s} - 1)(2^\ell - 2^{\ell-s-1})} = \left(\frac{2^s - 1}{2^{s+1} - 1} \right) \frac{1 - 2^{-(\ell-s)}}{1 - 2^{-(i-s)}} \\
&\leqslant \frac{1}{2} \times \frac{1}{1 - 2^{-(i-s)}} \leqslant \frac{1}{2} \times \frac{1}{1 - 1/4} = \frac{2}{3}.
\end{aligned} \tag{2.100}$$

As a result, we conclude that, for any $s < i - 1$,

$$p_{i|s} \overset{(2.99)}{\leqslant} \frac{2\Gamma_{i-s}^{\ell-s,i}}{\Delta_{\ell-s}} \overset{(2.100)}{\leqslant} 2\zeta_{i-1}\left(\frac{2}{3}\right)^{i-s-1} \leqslant 2\left(\frac{2}{3}\right)^{i-s-1}, \tag{2.101}$$

which implies the desired result. □

Lemma 2.12 (Upper Bound for Average Erasure Probability). *Let $F_i(z)$ be the average erasure probability of the i-th bit-channel as defined in (2.79). Fix $\beta, \delta \in \mathbb{R}^+$ and assume that*

$$z < \frac{i}{\ell} - \frac{g(\delta)}{\ell} - \left(\frac{\beta \ln \ell}{2\ell}\right)^{1/2}, \tag{2.102}$$

where $\log$ *and* $\ln$ *denote the logarithm in base* 2 *and* e*, respectively, and*

$$g(\delta) = \frac{\delta \log \ell + \log 6}{\log 3 - 1} = O(\delta \log \ell). \tag{2.103}$$

Then, we have that

$$F_i(z) < \ell^{-\beta} + \ell^{-\delta}. \tag{2.104}$$

Proof. Let us recall the formulation of $F_i(z)$ from (2.83) and split the summation into two parts, where a trivial upper bound is applied to each part: we drop $\binom{\ell}{s} z^s (1-z)^{\ell-s}$ for all terms in the summation with $s \leqslant i - g(\delta) - 1$, and we drop $p_{i|s}$ from the remaining terms that correspond to $s \geqslant i - g(\delta)$. More formally, we have

$$\begin{aligned} F_i(z) &= \sum_{s=0}^{i-g(\delta)-1} \binom{\ell}{s} z^s (1-z)^{\ell-s} p_{i|s} + \sum_{s=i-g(\delta)}^{\ell} \binom{\ell}{s} z^s (1-z)^{\ell-s} p_{s|i} \\ &< \sum_{s=0}^{i-g(\delta)-1} p_{s|i} + \sum_{s=i-g(\delta)}^{\ell} \binom{\ell}{s} z^s (1-z)^{\ell-s}. \end{aligned} \tag{2.105}$$

We apply the upper bound in (2.98) to the first summation, and obtain that

$$\sum_{s=0}^{i-g(\delta)-1} p_{s|i} \leqslant \sum_{s=0}^{i-g(\delta)-1} 2\left(\frac{2}{3}\right)^{i-s-1} \leqslant \sum_{s=g(\delta)}^{\infty} 2\left(\frac{2}{3}\right)^{s} = 6\left(\frac{2}{3}\right)^{g(\delta)} = \ell^{-\delta}. \tag{2.106}$$

The second summation is again upper bounded by applying Hoeffding's inequality on the tail probability of the binomial distribution $X \sim B(\ell, z)$ with ℓ trials and a success rate of z as

$$\sum_{s=i-g(\delta)}^{\ell} \binom{\ell}{s} z^s (1-z)^{\ell-s} = \mathbb{P}\{X \geqslant i - g(\delta)\} \leqslant \mathbb{P}\left\{X \geqslant z\ell + \left(\frac{\beta \ell \ln \ell}{2}\right)^{1/2}\right\} \leqslant \ell^{-\beta}. \tag{2.107}$$

□

2.4.3 Proof of the Concentration Theorem

At this point, we have gathered all the required tools to prove Theorem 11. Our proof consists of two steps. First, we show that the polarization behavior of a random non-singular $\ell \times \ell$ kernel is given, with high probability, by the function $F_i(z)$ analyzed in the previous subsection. Then, we explain how to relate this fact to an upper bound on $\lambda_{\alpha,K}(z)$.

As the theorem itself suggests, we split the proof into two parts: the first part takes care of the middle interval and proves (2.76), while the second one takes care of the tail intervals and proves (2.77).

Proof of (2.76). First, we combine the results from Lemma 2.10 and Lemma 2.12 to show that $F_i(z)$ roughly behaves as a step function. In fact, we have that

$$\begin{cases} F_i(z) > (1-\ell^{-\beta})(1-\ell^{-\delta}), & \text{if } z > \frac{i}{\ell} + \frac{\lceil \delta \log \ell \rceil}{\ell} + \left(\frac{\beta \ln \ell}{2\ell}\right)^{1/2} \\ F_i(z) < \ell^{-\beta} + \ell^{-\delta}, & \text{if } z < \frac{i}{\ell} - \frac{\lfloor \frac{\delta \log \ell + \log 6}{\log 3 - 1} \rfloor}{\ell} - \left(\frac{\beta \ln \ell}{2\ell}\right)^{1/2} \end{cases}. \tag{2.108}$$

Our strategy is to show that, with high probability over the choice of the kernel K, $f_{i,K}(z)$ is sharp for a fixed value of i. Then, we will use a union bound to show that $f_{i,K}(z)$ is sharp for all $i \in [\ell]$. To do so, we set $\beta = \delta = 4.5 + \log \ell$. Furthermore, we can assume that $\ell \geqslant 32$, as

(2.75) holds and $\alpha < 1/16$. It is easy to verify that

$$\begin{cases} F_i(z) > 1 - 2\ell^{-4.5-\log \ell} > 1 - (2\ell^{4+\log \ell})^{-1}, & \text{if } z \geqslant \frac{i}{\ell} + c\ell^{-1/2}\log \ell \\ \\ F_i(z) < 2\ell^{-4.5-\log \ell} < (2\ell^{4+\log \ell})^{-1}, & \text{if } z \leqslant \frac{i}{\ell} - c\ell^{-1/2}\log \ell \end{cases}, \tag{2.109}$$

where

$$c = \frac{(4.5 + \log \ell) + (\log 6)(\log \ell)^{-1}}{\log 3 - 1}\ell^{-1/2} + \left(\frac{4.5(\log \ell)^{-1} + 1}{2 \log e}\right)^{1/2} \leqslant 5, \qquad \forall \ell \geqslant 32. \tag{2.110}$$

Note that there are infinitely many values of z for which we need $f_{i,K}(z)$ to behave similar to (2.109). Hence, a simple union bound would not give us the proof. Fortunately, for all $i \in [\ell]$, $f_{i,K}(z)$ and consequently $F_i(z)$ are increasing functions of z. Hence, it suffices to consider only two points in $(0, 1)$ for each i, one slightly larger than $z = {}^{i}\!/\!_{\ell}$ and one slightly smaller.

Define

$$a_i \triangleq \frac{i}{\ell} + c\ell^{-1/2}\log \ell. \tag{2.111}$$

From (2.109), we have that

$$\mathbb{E}\big[1 - f_{i,K}(a_i)\big] = 1 - F_i(a_i) < (2\ell^{4+\log \ell})^{-1}, \tag{2.112}$$

where the expectation is taken over all non-singular $\ell \times \ell$ kernels. From Markov inequality, we deduce that

$$\mathbb{P}\big\{f_{i,K}(a_i) \leqslant 1 - \frac{1}{\ell^{2+\log \ell}}\big\} = \mathbb{P}\big\{1 - f_{i,K}(a_i) \geqslant \frac{1}{\ell^{2+\log \ell}}\big\} \leqslant \frac{\mathbb{E}_K\big[1 - f_{i,K}(a_i)\big]}{1/\ell^{2+\log \ell}} \leqslant \frac{1}{2\ell^2}. \tag{2.113}$$

Define

$$\mathcal{A}_i \triangleq \big\{K \in \mathbb{F}_2^{\ell\times\ell} \big| K \text{ is non-singular and } f_{i,K}(a_i) \geqslant 1 - \frac{1}{\ell^{2+\log \ell}}\big\}. \tag{2.114}$$

Therefore, (2.113) can be re-written as

$$\mathbb{P}\{K \in \mathcal{A}_i\} \geqslant 1 - \frac{1}{2\ell^2}. \tag{2.115}$$

Similarly, set

$$b_i \triangleq \frac{i}{\ell} - c\ell^{-1/2}\log\ell, \tag{2.116}$$

and define

$$\mathcal{B}_i \triangleq \big\{K \in \mathbb{F}_2^{\ell\times\ell}\big|K \text{ is non-singular and } f_{i,K}(b_i) \geqslant \frac{1}{\ell^{2+\log\ell}}\big\}. \tag{2.117}$$

A very similar use of Markov inequality shows that

$$\mathbb{P}\{K \in \mathcal{B}_i\} \geqslant 1 - \frac{1}{2\ell^2}. \tag{2.118}$$

Then, define

$$\mathcal{D} = \big(\cap_{j=1}^{\ell}\mathcal{A}_j\big) \cap \big(\cap_{j=1}^{\ell}\mathcal{B}_j\big). \tag{2.119}$$

By union bound, we obtain that

$$\mathbb{P}\{K \in \mathcal{D}\} \geqslant 1 - \sum_{i=1}^{\ell}\mathbb{P}\{K \notin \mathcal{A}_i\} - \sum_{i=1}^{\ell}\mathbb{P}\{K \notin \mathcal{B}_i\} \geqslant 1 - \frac{2\ell}{2\ell^2} = 1 - \frac{1}{\ell}. \tag{2.120}$$

Assume that $K \in \mathcal{D}$ throughout the remainder of proof. This implies that, for $i \in [\ell]$,

$$\begin{cases} f_{i,K}(z) > 1 - \frac{1}{\ell^{2+\log\ell}}, & \text{for } z = \frac{i}{\ell} + c\ell^{-1/2}\log\ell \\ f_{i,K}(z) < \frac{1}{\ell^{2+\log\ell}}, & \text{for } z = \frac{i}{\ell} - c\ell^{-1/2}\log\ell \end{cases}. \tag{2.121}$$

As $f_{i,K}(z)$ is an increasing function of z, (2.121) is equivalent to

$$\begin{cases} f_{i,K}(z) > 1 - \frac{1}{\ell^{2+\log\ell}}, & \text{for } z \geqslant \frac{i}{\ell} + c\ell^{-1/2}\log\ell \\ f_{i,K}(z) < \frac{1}{\ell^{2+\log\ell}}, & \text{for } z \leqslant \frac{i}{\ell} - c\ell^{-1/2}\log\ell \end{cases}. \tag{2.122}$$

Given these concentration results, we can proceed to the second step of the proof. Let us define

$$\begin{aligned} T_0(z,\ell) &\triangleq z\ell - c\ell^{1/2}\log\ell, \\ T_1(z,\ell) &\triangleq z\ell + c\ell^{1/2}\log\ell. \end{aligned} \tag{2.123}$$

Note that, for any $z \in (1/\ell^2, 1-1/\ell^2)$, the number of indices i such that $f_{i,K}(z)$ does not satisfy (2.122) is upper bounded by

$$T_1(z,\ell) - T_0(z,\ell) = 2c\ell^{-1/2}\log\ell, \tag{2.124}$$

as

$$\frac{i}{\ell} - c\ell^{-1/2}\log\ell < z < \frac{i}{\ell} + c\ell^{-1/2}\log\ell \Longleftrightarrow z\ell - c\ell^{1/2}\log\ell < i < z\ell + c\ell^{1/2}\log\ell. \tag{2.125}$$

We can re-write $\lambda_{\alpha,K}(z)$ which was defined earlier in (2.66) as

$$\lambda_{\alpha,K}(z) = \frac{\frac{1}{\ell}\sum\limits_{i\in(T_0(z,\ell),T_1(z,\ell))} g_\alpha(f_{i,K}(z))}{g_\alpha(z)} + \frac{\frac{1}{\ell}\sum\limits_{i\notin(T_0(z,\ell),T_1(z,\ell))} g_\alpha(f_{i,K}(z))}{g_\alpha(z)} \tag{2.126}$$

By using (2.122), we have that, for any $i \notin \big(T_0(z,\ell), T_1(z,\ell)\big)$,

$$g_\alpha(f_{i,K}(z)) \leqslant g_\alpha\left(\frac{1}{\ell^{2+\log\ell}}\right) < \left(\ell^{-2-\log\ell}\right)^\alpha. \tag{2.127}$$

By combining (2.127) with the trivial upper bound of $g_\alpha(f_{i,K}(z)) \leqslant 1$ for the left summation, we obtain that

$$\lambda_{\alpha,K}(z) \leqslant \frac{\frac{1}{\ell}2c\ell^{1/2}\log\ell}{g_\alpha(z)} + \frac{\left(\ell^{-2-\log\ell}\right)^\alpha}{g_\alpha(z)} \tag{2.128}$$

Furthermore, note that, for any $z \in (1/\ell^2, 1-1/\ell^2)$,

$$g_\alpha(z) \geqslant \left(\ell^{-2}(1-\ell^{-2})\right)^\alpha. \tag{2.129}$$

By combining (2.128) and (2.129), we have that

$$\lambda_{\alpha,K}(z) \leqslant \frac{1}{(1-\ell^{-2})^{\alpha}}\left(2c\ell^{-1/2+2\alpha}\log \ell + \ell^{-\alpha \log \ell}\right). \tag{2.130}$$

As (2.75) holds with $\alpha \leqslant 1/16$, ℓ is large enough so that

$$\begin{aligned} (1-\ell^{-2})^{\alpha} &\leqslant 2, \\ \log \ell &\leqslant \ell^{\alpha}, \\ -\alpha \log \ell &\leqslant -{}^{1}\!/\!{}_{2} + 3\alpha, \\ 4c+2 &\leqslant \ell^{2\alpha}. \end{aligned} \tag{2.131}$$

By applying the inequalities in (2.131) to (2.130), we finally obtain that

$$\begin{aligned} \lambda_{\alpha,K}(z) &\leqslant 4c\ell^{-1/2+2\alpha}\log \ell + 2\ell^{-\alpha\log \ell} \leqslant 4c\ell^{-1/2+3\alpha} + \ 2\ell^{-1/2+3\alpha} \\ &= (4c+2)\ell^{-\frac{1}{2}+3\alpha} \leqslant \ell^{-\frac{1}{2}+5\alpha}, \end{aligned} \tag{2.132}$$

which concludes the proof. □

Proof of (2.77). The proof of the tail intervals also follows from analyzing the average erasure probabilities. We present the proof mainly for the lower tail, where $z \in (0, 1/\ell^2)$. Similar arguments yield the proof for the upper tail.

We begin by recalling the previously derived upper bound on the average conditional erasure probability in (2.98):

$$p_{i|s} = \mathbb{P}\{\mathcal{R}_{\ell-s} \cap (E_i \setminus E_{i-1}) = \emptyset\} \leqslant 3\left(\frac{2}{3}\right)^{i-s}, \tag{2.133}$$

where the probability space is defined with respect to the selection of a random subspace $\mathcal{R}_{\ell-s} \subset \mathbb{F}_2^{\ell}$ of dimension $= \ell - s$. Once again, let us point out that the above mentioned probability is equal to $\mathbb{P}\{\mathcal{R}'_{\ell-s} \cap (E_i \setminus E_{i-1}) = \emptyset\}$, where $\mathcal{R}'_{\ell-s}$ is the linear span of some *randomly chosen* $\ell - s$ columns of a *random* kernel $K \in \mathbb{F}_2^{\ell\times\ell}$.

Let us define the *conditional erasure probabilities* of a fixed kernel K by

$$q_{i|s}(K) \triangleq \mathbb{P}\{\mathcal{R}_{\ell-s} \cap (E_i \setminus E_{i-1}) = \emptyset | K\} = \mathbb{P}\{\mathcal{R}_{\ell-s}(K) \cap (E_i \setminus E_{i-1}) = \emptyset\}, \tag{2.134}$$

where $\mathcal{R}_{\ell-s}(K)$ is the linear span of $\ell - s$ columns in K that are selected uniformly at random. Note that in (2.134) the kernel K is fixed and the probability is with respect to the selection of the columns of the kernel (i.e., with respect to the location of the s channel erasures). Hence, it is clear that

$$\mathbb{E}_K[q_{i|s}(K)] = p_{i|s} \leqslant 3\left(\frac{2}{3}\right)^{i-s}. \tag{2.135}$$

Similar to the proof for the middle interval, we provide some concentration results about $q_{i|s}(K)$, when K is selected uniformly at random among the non-singular $\ell \times \ell$ matrices. Let us first fix the value of i, and s. Then, by Markov inequality, we have

$$\mathbb{P}\left\{q_{i|s}(K) \geqslant 6\ell^2(\ell+1)\left(\frac{2}{3}\right)^{i-s}\right\} \leqslant \frac{1}{2\ell^2(\ell+1)}, \tag{2.136}$$

where the probability is defined with respect to the selection of the kernel. By union bound, for any $i \in \{1, \dots, \ell\}$ and $s \in \{0, \dots, \ell\}$, we deduce that

$$q_{i|s}(K) \leqslant 6\ell^2(\ell+1)\left(\frac{2}{3}\right)^{i-s}, \tag{2.137}$$

with probability of at least $1 - 1/2\ell$.

Pick a kernel K such that (2.137) holds. Furthermore, as K is non-singular, $q_{i|0}(K) = 0$. Hence, an upper bound on $f_{i,K}(z)$ is given by

$$\begin{aligned} f_{i,K}(z) &= \sum_{s=1}^{\ell} q_{i|s}(K)\binom{\ell}{s} z^s(1-z)^{\ell-s} \\ &\leqslant \sum_{s=1}^{\ell} 6\ell^2(\ell+1)\left(\frac{2}{3}\right)^{i-s}\binom{\ell}{s} z^s(1-z)^{\ell-s} \\ &= 6\ell^2(\ell+1)\left(\frac{2}{3}\right)^{i}\sum_{s=1}^{\ell}\binom{\ell}{s}\left(\frac{3z}{2}\right)^{s}(1-z)^{\ell-s}. \end{aligned} \tag{2.138}$$

Note that

$$\sum_{s=1}^{\ell}\binom{\ell}{s}\left(\frac{3z}{2}\right)^{s}(1-z)^{\ell-s}=\Big(1+\frac{z}{2}\Big)^{\ell}-(1-z)^{\ell}<(1+z)^{\ell}-(1-z)^{\ell}$$
$$\leqslant 2\ell z(1+z)^{\ell-1}, \tag{2.139}$$

where the last inequality in (2.139) comes from the fact that $\forall x\in(0,1)$, there exists a $x_0\in(1-x,1+x)$ such that

$$(1+x)^{\ell}-(1-x)^{\ell}=\big((1+x)-(1-x)\big)\left[\frac{\partial(1+x)^{\ell}}{\partial x}\bigg|_{x=x_0}\right]$$
$$=2\ell x(1+x_0)^{\ell-1}\leqslant 2\ell x(1+x)^{\ell-1}. \tag{2.140}$$

Next, we point out that, for any $z<\ell^{-2}$ and any $\ell\geqslant 2$, we have

$$(1+z)^{\ell-1}\leqslant\left(1+\frac{1}{\ell^2}\right)^{\ell-1}\leqslant\left(1+\frac{1}{\ell^2}\right)^{\ell^2}<\exp(1)<3. \tag{2.141}$$

Now, we replace (2.139) and (2.141) in (2.138) to obtain that

$$f_{i,K}(z)<36\ell^3(\ell+1)\left(\frac{2}{3}\right)^{i}z\leqslant 38\ell^4\left(\frac{2}{3}\right)^{i}z, \tag{2.142}$$

where the last inequality holds for $\ell\geqslant 18$.

Finally, we use (2.142) to derive the following upper bound on $\lambda_{\alpha,K}(z)$ for any $z\in(0,1/\ell^2)$:

$$\lambda_{\alpha,K}(z)=\frac{\frac{1}{\ell}\sum_{i=1}^{\ell}g_\alpha(f_{i,K}(z))}{g_\alpha(z)}=\frac{1}{\ell}\sum_{i}^{\ell}\frac{\Big(f_{i,K}(z)\big(1-f_{i,K}(z)\big)\Big)^{\alpha}}{\big(z(1-z)\big)^{\alpha}}$$
$$<\frac{1}{\ell}\sum_{i=1}^{\ell}\Big(f_{i,K}(z)\Big)^{\alpha}z^{-\alpha}(1-z)^{-\alpha}<\ell^{4\alpha-1}\left(38^{\alpha}\sum_{i=1}^{\ell}\left(\frac{2}{3}\right)^{i\alpha}\right)(1-z)^{-\alpha}. \tag{2.143}$$

As $\alpha\leqslant 1/16$,

$$\left(\frac{38}{1-z}\right)^{\alpha}<\left(\frac{38}{1-(1/18)^2}\right)^{1/16}<\frac{3}{2}. \tag{2.144}$$

Furthermore,

$$\sum_{i=1}^{\ell}\left(\frac{2}{3}\right)^{i\alpha}<\sum_{i=1}^{\infty}\left(\frac{2}{3}\right)^{i\alpha}=\sum_{j=0}^{\infty}\sum_{k=1}^{\lceil 1/\alpha\rceil}\left(\frac{2}{3}\right)^{(j\lceil 1/\alpha\rceil+k)\alpha}<\sum_{j=0}^{\infty}\sum_{k=1}^{\lceil 1/\alpha\rceil}\left(\frac{2}{3}\right)^{(j\lceil 1/\alpha\rceil)\alpha}$$
$$=\lceil\frac{1}{\alpha}\rceil\sum_{j=0}^{\infty}\left(\frac{2}{3}\right)^{(j\lceil 1/\alpha\rceil)\alpha}\leqslant\left(1+\frac{1}{\alpha}\right)\sum_{j=0}^{\infty}\left(\frac{2}{3}\right)^{j}=3\left(1+\frac{1}{\alpha}\right)<\frac{4}{\alpha}. \quad (2.145)$$

Moreover, from (2.75) we obtain that

$$6\alpha^{-1}\leqslant\ell^{1/4}. \quad (2.146)$$

By combining (2.143), (2.144), (2.145) and (2.146) and by using again that $\alpha\leqslant 1/16$, we conclude that, for any $z\in(0,1/\ell^2)$,

$$\lambda_{\alpha,K}(z)<\ell^{4\alpha-1}\times\frac{3}{2}\times\frac{4}{\alpha}\leqslant\ell^{4\alpha-3/4}\leqslant\ell^{-1/2}, \quad (2.147)$$

which yields the desired bound on the lower tail.

The proof for the upper tail follows very similar arguments. First, we define

$$h_{i,K}(z)\triangleq 1-f_{i,K}(z),$$
$$r_{i|s}(K)\triangleq 1-q_{i|s}(K). \quad (2.148)$$

Next, we use the upper bound on the average conditional erasure probability from (2.90) to provide an upper bound on $\mathbb{E}\big[r_{i|s}(K)\big]$ that is very similar to (2.135), i.e.,

$$\mathbb{E}_K[r_{i|s}(K)]\leqslant 2^{-(s-i)}. \quad (2.149)$$

By following steps similar to (2.136)-(2.147) and by using that $\alpha\leqslant 1/16$ and $4\alpha^{-1}\leqslant\ell^{1/4}$, we show that, for any $z\in(1-1/\ell^2,1)$,

$$\lambda_{\alpha,K}(z)\leqslant\ell^{-1/2}, \quad (2.150)$$

with probability at least $1 - 1/(2\ell)$ over the choice of the kernel. By combining (2.147) and (2.150) and using one last union bound, we conclude that

$$\mathbb{P}\left\{\lambda_{\alpha,K}(z) < \ell^{-\frac{1}{2}},\ \ \forall z \in \left(0, \frac{1}{\ell^2}\right) \cup \left(1 - \frac{1}{\ell^2}, 1\right)\right\} > 1 - \frac{1}{\ell}. \tag{2.151}$$

□

In the end, we point out that the increase in kernel size, while improving the error performance, worsens the overall decoding complexity of the code. The recursive implementation of the successive cancellation decoding for polar codes is based on a scheduling problem on the butterfly-like graph of polar codes, where each node represents a polarization kernel. These kernels perform the successive cancellation decoding within themselves and then communicate with each other on a specific schedule that reveals the uncoded information bits sequentially and efficiently.

The asymptotic overall decoding complexity for the conventional polar codes is given by the $O(n \log n)$, where n denotes the code-length. However, the internal SC calculations within the kernels becomes more complicated when the conventional kernel is replaced with larger $\ell\times\ell$ ones, which effectively changes the asymptotic decoding complexity to $O(2^\ell n \log n)$. In the next section, we propose a method that exploits the structure of the kernel to reduce the decoding complexity for certain polarization kernels.

2.5 Decoding Algorithms for Polar Codes with Large Kernels

2.5.1 Permuted Arıkan Kernels

Introduced by Arikan [10], polar codes are the first codes that were proved to achieve the symmetric capacity of a binary-input discrete memoryless channel (B-DMC) $\mathcal{W}$. Polar codes can be viewed as part of a much larger family of codes that are generated according to the 2×2 matrix

$$F_2 \overset{\text{def}}{=} \begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix}.$$

These length $n = 2^m$ codes are cosets of a linear subspace that is spanned by some k rows of $F_2^{\otimes m}$, the m-th Kronecker power of F_2. In this section we suggest a different approach towards polar codes with high performance and efficient SC decoding. In our approach we consider a special type of kernels called *permuted kernels*. These kernels are formed by permuting the rows of $F_2^{\otimes \ell}$. One example of a permuted kernel is the kernel K_8 from Section 2.2.2. On the other hand, the kernel K_{16} also defined in 2.2.2 is not a permuted kernel. While a successive cancellation decoder for a polar code with the kernel F_2 and dimension k decides on the n input bits $u_0 u_1 \ldots u_{n-1}$ ($n - k$ of them are known to the decoder) one after the other according to the sequential order from 0 to $n - 1$, a SC decoder for polar codes with permuted kernels decides on the input bits according to a permuted order of their indices. Therefore, we call the SC decoder for a polar code with a permuted kernel a *permuted successive cancellation* (PSC) decoder.

For simplicity, we only describe our PSC decoding algorithm for length-ℓ polar codes in this section, which is an efficient implementation of the SC decoder, in terms of both time and space complexity. We also propose two new 16×16 permuted kernels and show simulation results for their performance.

Let us begin by introducing the notation and definitions used throughout the remainder of this chapter and review the basic concepts for polar codes. We modify our indexing of vectors from $1, \cdots, \ell$ to $0, 1, \cdots, \ell - 1$ to preserve consistency with most of the papers that study the implementation of polar codes.

For a positive integer n, denote by $[n]$ the set of n integers $\{0, 1, \ldots, n-1\}$. For a positive integer ℓ and for all $r \in [\ell]$, denote by $[n]_r$ the set of all elements in $[n]$ that are equal to r modulo ℓ, where ℓ should be clear from the context. A binary vector of length n is denoted by $u_0^{n-1} = u_0 u_1 \ldots u_{n-1}$. For $\mathcal{A} \subseteq [n]$, denote by $u_{\mathcal{A}}$ the subvector of u_0^{n-1} that is specified according to indices from $\mathcal{A}$. In particular, if ℓ divides n and $r \in [\ell]$ then $u_{[n]_r} = u_r u_{\ell+r} \ldots u_{n-\ell+r}$. All operations on vectors and matrices in this paper are carried out over the field $GF(2)$. The componentwise addition modulo-2 of two binary vectors u_0^{n-1} and v_0^{n-1} is denoted by $u_0^{n-1} \oplus v_0^{n-1}$. For an $\ell \times \ell$ matrix K, denote by $K^{\otimes m}$ the mth Kronecker power of K.

Similar to before, we define $\mathcal{W} : \mathcal{X} \to \mathcal{Y}$ to be a generic B-DMC with input alphabet $\mathcal{X} = \{0, 1\}$, output alphabet $\mathcal{Y}$, and *transition probabilities* $\mathcal{W}(y|x)$, where for all $x \in \mathcal{X}$ and $y \in \mathcal{Y}$, $\mathcal{W}(y|x)$ is the conditional probability that the channel output is y given that the transmitted input is x. For a positive integer n, denote by $\mathcal{W}^n : \mathcal{X}^n \to \mathcal{Y}^n$ the channel that corresponds to transmission over n independent copies of $\mathcal{W}$. Hence, for every $x_0^{n-1} \in \mathcal{X}^n$ and $y_0^{n-1} \in \mathcal{Y}^n$, the transition probability $\mathcal{W}^n(y_0^{n-1}|x_0^{n-1})$ is given by

$$\mathcal{W}^n(y_0^{n-1}|x_0^{n-1}) \overset{\text{def}}{=} \prod_{i=0}^{n-1} \mathcal{W}(y_i|x_i).$$

Let K denote an $\ell \times \ell$ polarization kernel. Let $n = \ell^m$ and let R_n be the permutation matrix for which $u_0^{n-1} R_n = u_{[n]_0} u_{[n]_1} \ldots u_{[n]_{\ell-1}}$, for all $u_0^{n-1} \in \mathcal{X}^n$. For an $\ell \times \ell$ kernel K, define the matrix $G_{m,K}$ recursively by $G_{1,K} \overset{\text{def}}{=} K$ and

$$G_{m,K} \overset{\text{def}}{=} (I_{n/\ell} \otimes K) R_n (I_\ell \otimes G_{m-1,K}). \tag{2.152}$$

For a set $\mathcal{A} \subset [n]$ of size k, and a vector f_0^{n-k-1}, let $\mathcal{C}$ be the code that encodes a length n input vector u_0^{n-1}, for which $u_{[n]\setminus\mathcal{A}} = f_0^{n-k-1}$, to the codeword $x_0^{n-1} = u_0^{n-1}G_{m,K}$. If $i \in [n] \setminus \mathcal{A}$, then u_i is called a *frozen bit*. For all $i \in [n]$, the *i-th bit-channel* with respect to $G_{m,K}$, $\mathcal{W}_{m,K}^{(i)} : \mathcal{X} \to \mathcal{Y}^n \times \mathcal{X}^i$, is defined by the transition probabilities

$$\mathcal{W}_{m,K}^{(i)}(y_0^{n-1}, u_0^{i-1}|u_i) \overset{\text{def}}{=} \sum_{u_{i+1}^{n-1} \in \mathcal{X}^{n-i-1}} \frac{1}{2^{n-1}} \mathcal{W}^n(y_0^{n-1}|u_0^{n-1}G_{m,K}). \tag{2.153}$$

A polar code $\mathcal{C}$ of length m and with kernel K is defined by setting $\mathcal{A}$ to be the set of k indices corresponding to the bit-channels with the lowest Bhattacharyya parameters. Note that by the definition of polar codes, the values of the frozen bits are also required. If the channel is symmetric, then the frozen bits are all taken to be zero. For asymmetric channels, an assignment of the frozen bits that guarantees a vanishing probability of error is known to exist, however no practical method that finds such an assignment is known.

A *successive cancellation* (SC) decoder for $\mathcal{C}$ outputs a *decision vector* $\hat{u}_0^{n-1}$ in n steps, where at the ith step the decoder decides on the value of $\hat{u}_i$ according to the following rule. If $i \in [n] \setminus \mathcal{A}$ then $\hat{u}_i$ is set to the value of the frozen bit u_i. Otherwise, the decoder calculates the pair of transition probabilities

$$\mathcal{W}_{m,K}^{(i)}(y_0^{n-1}, \hat{u}_0^{i-1}|u_i = 0),\ \mathcal{W}_{m,K}^{(i)}(y_0^{n-1}, \hat{u}_0^{i-1}|u_i = 1). \tag{2.154}$$

and sets $\hat{u}_\varphi$ to the more likely value according to these probabilities. Notice that, in general, the complexity of the SC decoder may be exponential in n, since the calculation of the transition probabilities requires a summation of 2^{n-i-1} terms. However, we show that specific structured kernels allow a simpler formulation.

The recursive structure of the matrix $G_{m,K}$ induces recursive formulas for the bit-channels as follows.

Lemma 2.13. *Let $K = (K_{r,s})$ be an $\ell \times \ell$ kernel. For all $i \in [n]$, if $i = \varphi\ell + j$ for some $\varphi \in [n/\ell]$ and $j \in [\ell]$ then*

$$\begin{aligned} \mathcal{W}_{m,K}^{(i)}(y_0^{n-1}, u_0^{i-1}|u_i) = \frac{1}{2^{\ell-1}} \sum_{u_{\varphi\ell+j+1}^{(\varphi+1)\ell-1}} \prod_{s=0}^{\ell-1} \\ \mathcal{W}_{m-1,K}^{(\varphi)}(y_{sn/\ell}^{(s+1)n/\ell-1}, T^{(s)}(u_{[\varphi\ell]})| \oplus_{r=0}^{\ell-1} K_{r,s} \cdot u_{\varphi\ell+r}), \end{aligned} \tag{2.155}$$

where $T^{(s)}(u_{[\varphi\ell]}) \overset{\text{def}}{=} \oplus_{r=0}^{\ell-1} K_{r,s} \cdot u_{[\varphi\ell]_r}$.

Notice that $u_{[\varphi\ell]_r}$ is a vector of length φ and $K_{r,s} \in GF(2)$, hence $T^{(s)}(u_{[\varphi\ell]})$ is a vector of length φ as required by the definition of the φth bit channel. The term $\oplus_{r=0}^{\ell-1} K_{r,s} \cdot u_{\varphi\ell+r}$ is simply the inner product of $u_{\varphi\ell}^{(\varphi+1)\ell-1}$ with the sth column of K.

From Lemma 2.13, it follows that a SC decoding algorithm at the kernel level, i.e., for a length-ℓ polar code with kernel K, that has time complexity t and space complexity s can be extended to a SC decoding algorithm for a length-n polar code with kernel K that has time complexity $O(tn \log n/(\ell \log \ell))$ and space complexity $O(sn/(\ell - 1))$. In particular, there exists an implementation for the SC decoder that runs in $O(2^\ell n \log n/(\ell \log \ell))$. In practice, this time complexity may be too large even for relatively small values of ℓ. For this reason we propose to use a special type of kernel called a *permuted kernel* that can simultaneously reduce the time complexity of the SC decoder and achieve better scaling exponents.

A *permutation* ρ of length ℓ is a bijection $\rho : [\ell] \to [\ell]$. For a permutation ρ, the permutation matrix corresponding to ρ is denoted by M_ρ and defined by

$$(M_\rho)_{r,s} \overset{\text{def}}{=} \begin{cases} 1, & \text{if } s = \rho(r) \\ 0, & \text{otherwise,} \end{cases} \tag{2.156}$$

i.e., $u_0^{\ell-1} M_\rho = v_0^{\ell-1}$, where $v_{\rho(r)} = u_r$, for all $r \in [\ell]$. A *permuted kernel* with respect to ρ is defined by $K_\rho \overset{\text{def}}{=} M_\rho G_L$, where $G_L \overset{\text{def}}{=} G_{L,F_2}$ and $\ell = 2^L$. For ease of notation, for all $i \in [n]$ we denote the i-th bit-channel with respect to G_{m,K_ρ} by $\mathcal{W}_{m,\rho}^{(i)}$ and denote $\mathcal{W}_{m,F_2}^{(i)}$ by $\mathcal{W}_m^{(i)}$.

Let $\mathcal{C}_\rho$ be the polar code with kernel K_ρ and length $n = \ell$. For $i \in [\ell]$, let $D_i = \{\rho(0), \rho(1), \ldots, \rho(i-1), \rho(i)\}$ and let $d_i = \max\{D_i\}$.

Lemma 2.14. *For all $i \in [\ell]$, the i-th bit-channel with respect to K_ρ is equal to*

$$\mathcal{W}_{L,\rho}^{(i)}(y_0^{\ell-1}, u_0^{i-1}|u_i) = \sum_{v_{[d_i+1]\setminus D_i}} \mathcal{W}_L^{(d_i)}(y_0^{\ell-1}, v_0^{d_i-1}|v_{d_i}), \tag{2.157}$$

where $v_0^{\ell-1} = u_0^{\ell-1}M_\rho$, i.e., for all $r \in [\ell]$, $v_{\rho(r)} = u_r$.

Lemma 2.14 provides a connection between bit channels with respect to K_ρ and bit channels with respect to F_2, which will be useful for our SC decoding algorithm for $\mathcal{C}_\rho$, presented in the next section.

2.5.2 Permuted Successive Cancellation Decoding

In this section we formalize our SC decoder for a length-$\ell = 2^L$ code $\mathcal{C}_\rho$ defined by a permuted kernel K_ρ. As mentioned above, SC decoding for $\mathcal{C}_\rho$ is equivalent to SC decoding of a length-ℓ polar code with kernel F_2 that decides on the input bits in a permuted order according to ρ. For this reason, we call our SC decoding scheme *permuted successive cancellation* (PSC) decoding. We present an implementation of PSC decoding for any permutation ρ, which requires significantly less computational power and memory compared to the conventional SC decoder at the kernel level. Our proposed algorithm admits better time complexity only when ρ is not the identity permutation. For the identity permutation the algorithm coincides with the conventional SC decoder. Analysis of time and space complexity is also presented.

The PSC decoding algorithm is similar to the list SC decoding from [39] in the sense that it computes many pairs of transition probabilities for some bit channels. Therefore, we will adapt some of the notation and terminology from [39]. In particular, for all $0 \leqslant \lambda \leqslant L$

define $\Lambda \overset{\text{def}}{=} 2^\lambda$. For $0 \leqslant \varphi < \Lambda/2$ we have

$$\begin{aligned} &\mathcal{W}_\lambda^{(2\varphi)}(z_0^{\Lambda-1}, b_0^{2\varphi-1}|b_\varphi) = \\ &\qquad \sum_{b_{2\varphi+1}} \frac{1}{2}\mathcal{W}_{\lambda-1}^{(\varphi)}(z_0^{\Lambda/2-1}, b_{[2\varphi]_0} \oplus b_{[2\varphi]_1}|b_{2\varphi} \oplus b_{2\varphi+1}) \cdot \mathcal{W}_{\lambda-1}^{(\varphi)}(z_{\Lambda/2}^{\Lambda-1}, b_{[2\varphi]_1}|b_{2\varphi+1}), \end{aligned} \tag{2.158}$$

and

$$\begin{aligned} &\mathcal{W}_\lambda^{(2\varphi+1)}(z_0^{\Lambda-1}, b_0^{2\varphi}|b_{2\varphi+1}) = \\ &\qquad \frac{1}{2}\mathcal{W}_{\lambda-1}^{(\varphi)}(z_0^{\Lambda/2-1}, b_{[2\varphi]_0} \oplus b_{[2\varphi]_1}|b_{2\varphi} \oplus b_{2\varphi+1}) \cdot \mathcal{W}_{\lambda-1}^{(\varphi)}(z_{\Lambda/2}^{\Lambda-1}, b_{[2\varphi]_1}|b_{2\varphi+1}). \end{aligned} \tag{2.159}$$

Notice that, for $r \in \{0,1\}$, $[2\varphi]_r$ is the set of all elements in $[2\varphi]$ that are equal to r modulo-2. The index φ is called a *phase* and λ is called a *layer*. Thus, the pair of transition probabilities in phase i and layer λ is determined by two pairs of transition probabilities in phase $\varphi = \lfloor i/2 \rfloor$ and layer $\lambda - 1$; one corresponds to the output $(z_0^{\Lambda/2-1}, b_{[2\varphi]_0} \oplus b_{[2\varphi]_1})$ and the other to the output $(z_{\Lambda/2}^{\Lambda-1}, b_{[2\varphi]_1})$. From the recursive formulas above we obtain a binary tree of pairs of transition probabilities where the root of this tree is $\mathcal{W}_L^{(i)}(y_0^{\ell-1}, v_0^{i-1}|v_i)$, for some $i \in [n]$. Each pair of transition probabilities in this tree is associated with a *branch number* $0 \leqslant \beta < 2^{L-\lambda}$. For $\lambda = L$ the branch number of $\mathcal{W}_\lambda^{(i)}(y_0^{\ell-1}, v_0^{i-1}|v_i)$ (the root of the tree) is 0. If the branch number of $\mathcal{W}_\lambda^{(i)}(z_0^{\Lambda-1}, b_0^i|b_i)$ is β then $\mathcal{W}_{\lambda-1}^{(\varphi)}(z_0^{\Lambda/2-1}, b_{[2\varphi]_0} \oplus b_{[2\varphi]_1}|b_{2\varphi} \oplus b_{2\varphi+1})$ and $\mathcal{W}_{\lambda-1}^{(\varphi)}(z_{\Lambda/2}^{\Lambda-1}, b_{[2\varphi]_1}|b_{2\varphi+1})$, $\varphi = \lfloor i/2 \rfloor$, have branch numbers 2β and $2\beta+1$, respectively. With this terminology, we can refer to each pair of transition probabilities by a triple $(\varphi, \lambda, \beta)$ and denote

$$P_{\lambda,\beta}[b_0^\varphi] \overset{\text{def}}{=} \mathcal{W}_\lambda^{(\varphi)}(z_0^{\Lambda-1}, b_0^{\varphi-1}|b_\varphi). \tag{2.160}$$

We assign the same triple $(\varphi, \lambda, \beta)$ to the output and input of each pair of transition probabilities in the tree and denote $B_{\lambda,\beta}[\varphi] \overset{\text{def}}{=} b_\varphi$.

Remark 2.1. *We use the notations $(z_0^{\Lambda-1}, b_0^{\varphi-1})$ and b_φ for the output and input of the bit channel $W_\lambda^{(\varphi)}$ of branch number β, whereas the notations $(y_0^{\ell-1}, v_0^{\varphi-1})$ and v_φ are used only* *for* $W_L^{(\varphi)}$*. The values of* $(z_0^{\Lambda-1}, b_0^{\varphi-1})$ *and* b_φ *are determined recursively from* $(y_0^{\ell-1}, v_0^{\varphi-1})$ *and* v_φ.

For $i \in [\ell]$, let $\hat{u}_0^{i-1}$ be the bits decoded so far by the PSC decoding algorithm. Recall that in the i-th step, the SC decoder calculates the pair of transition probabilities defined in (2.154). Denote these transition probabilities by

$$Q_i[b] \overset{\text{def}}{=} \mathcal{W}_{1,\rho}^{(i)}(y_0^{\ell-1}, \hat{u}_0^{i-1} | u_i = b). \tag{2.161}$$

Also, recall that $D_i = \{\rho(0), \rho(1), \ldots, \rho(i)\}$ and d_i is the maximum over D_i. The *decoding window* in the ith step is denoted by $DW_i = [d_i + 1] \setminus D_i$. By Lemma 2.14,

$$Q_i[b] = \sum_{v_s:\ s \in DW_i} P_{L,0}[v_0^i], \tag{2.162}$$

where $v_{\rho(i)} = u_i = b$ and for all $r \in [i]$, $v_{\rho(r)} = \hat{u}_r$. If $j \in DW_i$ then v_j was not determined yet and is therefore called an *unknown bit*. As with every SC decoder for polar codes, for every $i \in [\ell]$, the PSC decoding algorithm processes the ith step of the decoding by two stages:

1) Recursive transition probabilities computations.

2) Recursive decision making.

Next, we will describe these two stages.

1) Recursive transition probabilities computations:

In the beginning of the algorithm, where $i = 0$, $d_0 = \rho(0)$, and $DW_0 = [d_0 + 1]$, the algorithm recursively computes $P_{L,0}[v_0^{d_0}]$, for every choice of $v_0^{d_0-1}$. Thus, for every layer λ and any branch number β it calculates and stores the pair of transition probabilities that are required for the calculation of $P_{L,0}[v_0^{d_0}]$. Figure 2.5 illustrates the execution of this stage for $i = 0$ and a length-four permutation $\rho = (2, 0, 3, 1)$. For every $0 < i \leqslant \ell$, if $d_i = d_{i-1}$,

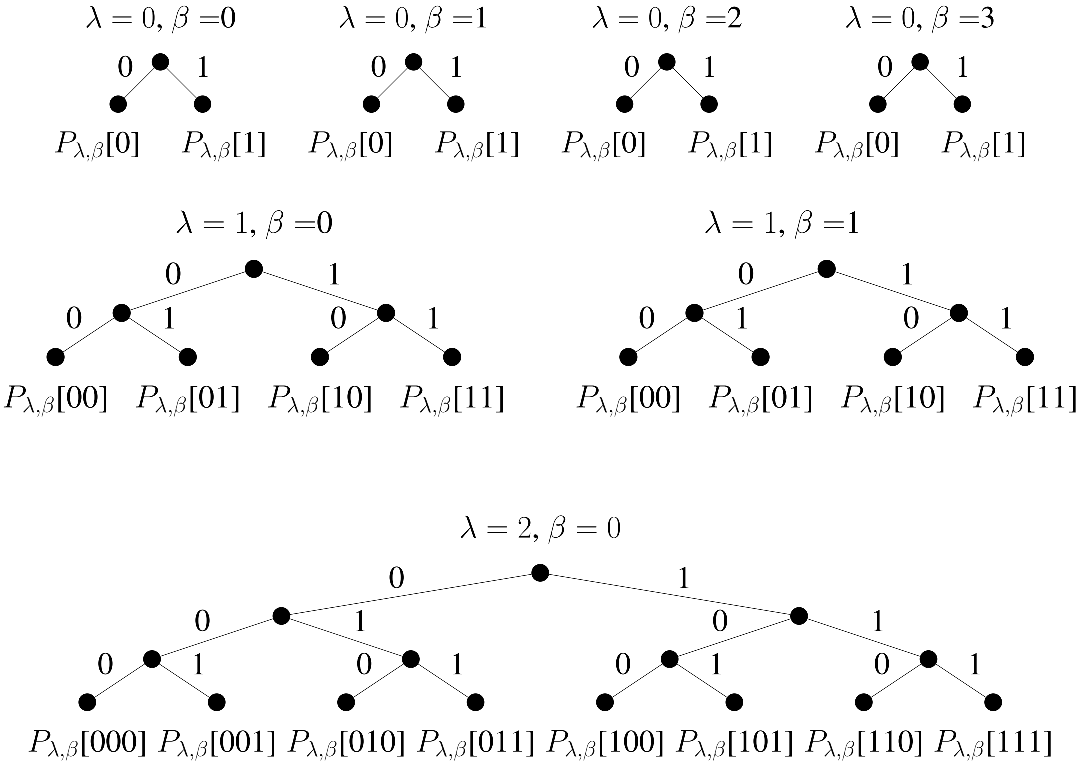


Figure 2.5: Transition probabilities computations for each of the layers at step $i = 0$ and for $\rho = (2, 1, 3, 0)$.

then $P_{L,0}[v_0^{d_i}]$ was already computed, for every choice of v_{DW_i}, and the algorithm does not need to compute anything new. Otherwise, it must recursively calculate and store $P_{L,0}[v_0^{d_i}]$, for every choice of v_{DW_i}. The recursive calculation of $P_{\lambda,\beta}[b_0^j]$ is carried out through equations (2.158) and (2.159) (depending on the parity of j) using $P_{\lambda-1,2\beta}[b_{[2\varphi+1]_0} \oplus b_{[2\varphi+1]_1}]$ and $P_{\lambda-1,2\beta+1}[b_{[2\varphi+1]_1}]$, where $\varphi = \lfloor j/2 \rfloor$. Notice, that if the algorithm needs to calculate $P_{\lambda,\beta}[b_0^j]$ it will never make use of $P_{\lambda,\beta}[b_0^r]$, for $r < j$ and it can remove any such transition probability from the memory. Thus the algorithm stores $\sum_{\lambda=0}^{L} 2^{L-\lambda} = 2^{L+1} - 1 = 2\ell - 1$ vectors of transition probabilities pairs with various lengths.

2) Recursive decision making:

For every $0 \leqslant i < \ell$, after computing all the relevant pairs of transition probabilities in the previous stage, the algorithm computes $Q_i[b]$ according to (2.162) and decides on the value of $\hat{u}_i = v_{\rho(i)}$ according to the SC decoding decision rule, i.e. $\hat{u}_i = u_i$ if u_i is a frozen bit and otherwise it is set to the more likely value based on $Q_i[b]$, $b \in \{0, 1\}$. Once the value of $v_{\rho(i)}$ is determined, the algorithm recursively updates the values of the outputs/inputs for any other layers and branch numbers if these values are available. The recursive update of the inputs is carried out as follows. If $B_{\lambda,\beta}[j]$ was updated and j is odd then $B_{\lambda+1,2\beta+1}[\varphi] = B_{\lambda,\beta}[j]$, $\varphi = \lfloor j/2 \rfloor$, and the transition probabilities in layer $\lambda+1$ and branch number $2\beta+1$, associated with $B_{\lambda+1,2\beta+1}[\varphi] \neq B_{\lambda,\beta}[j]$ are removed. If both $B_{\lambda,\beta}[2\varphi]$ and $B_{\lambda,\beta}[2\varphi+1]$ are available then $B_{\lambda+1,2\beta}[\varphi] = B_{\lambda,\beta}[2\varphi] \oplus B_{\lambda,\beta}[2\varphi+1]$ and the transition probabilities in layer $\lambda+1$ and branch number 2β associated with $B_{\lambda+1,2\beta}[\varphi] \neq B_{\lambda,\beta}[2\varphi] \oplus B_{\lambda,\beta}[2\varphi + 1]$ are removed.

An execution of this stage for $i = 0$ and the permutation $\rho = (2, 1, 3, 0)$ will result in only removing transition probabilities in layer 2. If, for example, the algorithm decision on the value of $v_{\rho(0)}$ was $v_{\rho(0)} = 0$, then the probabilities that are stored in layer 2 are $P_{2,0}[000]$, $P_{2,0}[010]$, $P_{2,0}[100]$, and $P_{2,0}[110]$. Repeating the recursive transition probabilities computation stage for $i = 1$ does not require any new transition probabilities computation since $d_1 = d_0 = 2$. Assuming the decision for $v_{\rho(1)}$ is 0 as well, at the decision-making stage the algorithm removes from layer 0 the transition probabilities associated with $v_{\rho(1)} = 1$. It then computes $B_{1,1}[0] = 0$ and removes the transition probabilities associated with $B_{1,1}[0] = 1$ from layer 1 and branch number 1. Since $\rho(1)$ is odd and $\rho(1) - 1$ was not yet calculated, it cannot make a decision for layer 1 and branch number 0. Figure 2.6 shows the result of propagating the decision $v_{\rho(1)} = 0$ to the layers and branch numbers that are affected by this decision.

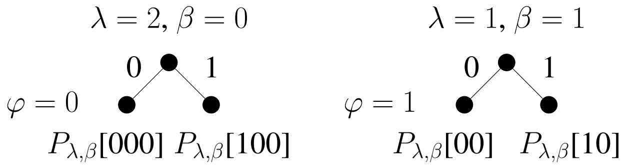


Figure 2.6: The remaining transition probabilities in the affected layers and branch numbers at the end of step $i = 1$, after propagating the decisions $v_2 = 0$ and $v_1 = 0$, where $\rho = (2, 1, 3, 0)$.

Next, we discuss the time and space complexity of our algorithm and compare the performance and decoding complexity of some polar codes with permuted kernels.

Both space complexity and time complexity of the algorithm are highly dependent on the permutation and therefore we express them as $O(s_\rho n)$ and $O(t_\rho n \log n)$, respectively, where s_ρ and t_ρ are constants that depend only on the permutation ρ. To compute t_ρ, we count the total number of pairs of transition probabilities that were calculated by the algorithm using equations (2.158) and (2.159), and divide this number by ℓ. Similarly, to compute s_ρ, we find the maximum number of transition probabilities pairs that were simultaneously stored in the memory and divide this number by ℓ.

Remark 2.2. *In the computation of the time complexity we ignore the computation of $Q_i[b]$, $i \in [\ell]$, by equation (2.162), given the transition probabilities pairs $P_{L,0}[v_0^i]$, since this requires at most $2t_\rho\ell$ operations. Similarly, for the computation of the space complexity we ignored the extra space used by the algorithm to store the phases of the unknown bits for each layer and branch number, as well as values of some known bits that are still being used. The space for this extra information is $o(s_\rho\ell)$.*

Unfortunately, there is no simple formula to compute s_ρ and t_ρ. Yet, we will show how to derive these quantities by considering a more complex example of a length-8 permutation $\rho = (1, 4, 0, 2, 7, 3, 6, 5)$. At step 0 the algorithm computes $P_{3,0}[v_0 v_1]$. To this end it needs to compute the 16 pair of transition probabilities $P_{\lambda,\beta}[b_0]$, for every $0 \leqslant \lambda < 3$ and $0 \leqslant \beta < 2^{3-\lambda}$. At step 1 it needs to compute $P_{3,0}[v_0^4]$ when v_1 is known, i.e., 2^3 new pairs of transition probabilities. To this end it must compute $P_{2,0}[b_0^2]$ and $P_{2,1}[b_0^2]$, where $B_{2,1}[0] = v_1$. Thus, it must compute $4 + 2 = 6$ new pairs of transition probabilities at layer 2. At layer 1 it needs to compute $P_{1,\beta}[b_0^1]$, for every $0 \leqslant \beta < 4$, i.e., $2 \cdot 4 = 8$ new pairs. Overall it computes 22 pairs at this step. The algorithm will only compute new probabilities at step 4, where it computes 16 new pairs of transition probabilities. Overall it computes 54 pairs of transition probabilities and the decoding computation complexity of a length-n polar code with kernel K_ρ is $O(t_\rho n \log n)$, where $t_\rho = 2.25$. The maximum number of transition probabilities pairs that were stored in the memory at the same time is 30 and hence the space complexity of the algorithm in this example is $O(s_\rho n)$, where $s_\rho = 4.286$.

Figure 2.7 depicts the performance comparison of three polar codes of length $n = 2^8$ over binary erasure channels. All codes are optimized for the channel BEC(0.2) and rate $R = \frac{3}{5}$. The ML bound on performance of the F_2 polar code is also given, based upon methods in [39]. Two of these codes are constructed via K_σ, where

$$\sigma = (0, 1, 2, 3, 4, 6, 8, 10, 5, 9, 7, 11, 12, 13, 14, 15) \tag{2.163}$$

and K_π, where

$$\pi = (0, 1, 2, 4, 8, 3, 5, 6, 9, 10, 12, 7, 11, 13, 14, 15). \tag{2.164}$$

The other polar code is the conventional Arikan's polar code, constructed from the 2×2 kernel F_2. We chose these permuted kernels since they have relatively low scaling exponents, $\mu(K_\sigma) = 3.541$ and $\mu(K_\pi) = 3.479$. The time complexity of the PSC decoding algorithm for length-n polar codes with kernels K_σ and K_π is $O(t_\sigma n \log n)$ and $O(t_\pi n \log n)$, respectively, where $t_\rho = 1.907$ and $t_\pi = 2.407$. The space complexity for these codes is $O(s_\sigma n)$ and $O(s_\pi n)$, respectively, where $s_\sigma = 9.143$ and $s_\pi = 11.429$. It is observed that the polar code

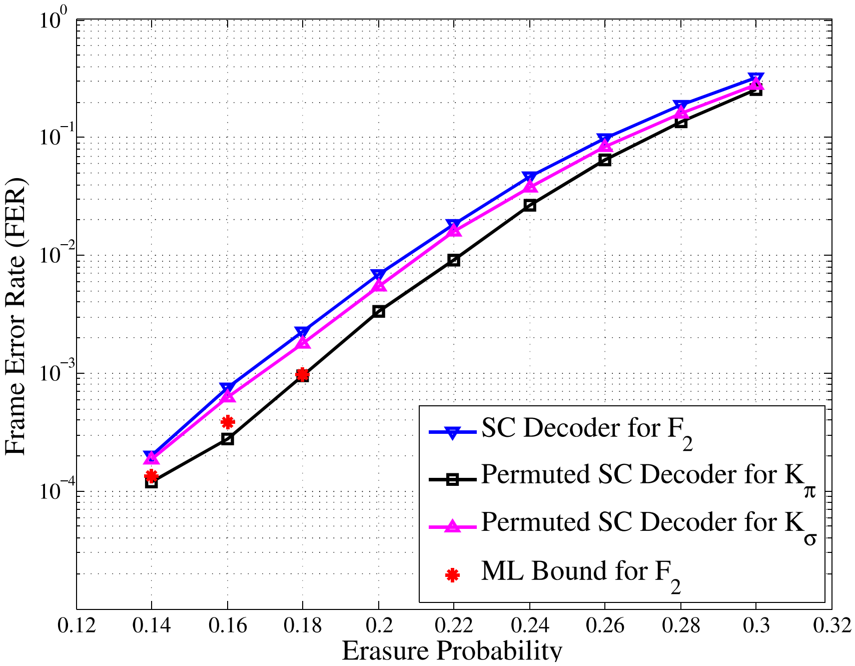


Figure 2.7: FER comparison of three polar codes over binary erasure channels at length $n = 2^8$ constructed with F_2, K_σ, and K_π.

constructed from K_π and decoded with PSC outperforms both the conventional polar code and the code constructed by K_σ. The latter also outperforms the conventional polar code. The results are in agreement with the convention that the smaller the scaling exponent, the better the polar code performance. Notice that for some erasure channels the frame-error-rate of the polar code constructed from K_π is lower than the ML bound. Since the actual performance of the ML decoder can only be worse than the ML bound, the polar code constructed from K_π outperforms Arikan's polar code even when the latter is decoded by the ML decoder. It is to be noted that the conventional construction algorithms such as [38] cannot be applied directly for the kernels with $\ell \geqslant 4$ due to the exponential increase in the number of bit channel outputs. Here, we utilized a Monte-Carlo construction algorithm, which may be improved by using a larger number of iterations.

This chapter contains materials as it appears in [4, 5]:

- A. Fazeli and A. Vardy, “On the scaling exponent of binary polarization kernels," *Proceedings of IEEE 52nd Allerton Conference on Communication, Control, and Computing*, Sep. 2014, pp. 797-804, and
- S. Buzaglo, A. Fazeli, P. H. Siegel, V. Taranalli, and A. Vardy, “On efficient decoding of polar codes with large kernels," *Proceedings of IEEE Wireless Communications and Networking Conference Workshops*, Mar. 2017, pp. 1-6.

It is also, in part, a reprint of [6, 7]:

- A. Fazeli, S. H. Hassani, M. Mondelli, and A. Vardy, “Binary linear codes with optimal scaling: polar codes with large kernels," submitted to *IEEE Transactions on Information Theory*, available online at arXiv:1711.01339, and
- S. Buzaglo, A. Fazeli, P. H. Siegel, V. Taranalli, and A. Vardy, “Permuted successive cancellation decoding for polar codes," *Proceedings of IEEE International Symposium on Information Theory*, Jun. 2017, pp. 2618-2622.

The dissertation author was the primary investigator and author of these papers.

Chapter 3

Convolutional Polar Codes

3.1 Genie-aided Decoding of Polar Codes

Polar codes provably achieve the capacity of memoryless symmetric channels with low encoding and decoding complexity. Nonetheless, for short and moderate block-lengths, polar codes fail to deliver competitive performance under successive cancellation decoding. Consequently, much effort has been devoted to improving the performance of polar codes, either through enhanced decoding algorithms or by modifying the code structure, or both. CRC-aided list decoding of polar codes is the most successful approach along this line of research. However, list decoding requires following L decoding paths, which leads to a significant increase in decoding complexity if L is large.

The encoding structure of the $[n,k]$-polar code consists of an $n \times n$ nonsingular generator matrix G with $n = 2^m$, which as input takes a length-n binary vector $\boldsymbol{u}$ and outputs $\boldsymbol{x} = \boldsymbol{u}G$. This is formulated by

$$\boldsymbol{x} = \boldsymbol{u}G, \text{ where } G = B \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}^{\otimes m}. \tag{3.1}$$

Here, $\otimes$ denotes the Kronecker product, B is the $n \times n$ bit-reversal permutation matrix, $\boldsymbol{x}$ is the length-n polar codeword, and $\boldsymbol{u}$ is a length-n binary vector that includes k information bits and $n-k$ predetermined frozen values. The coordinates on $\boldsymbol{u}$ are divided into two subsets: k indices that carry the information bits, and $n-k$ indices that are frozen to some predetermined values (conventionally zero). The selection of these bits is also predetermined and optimized based on the underlying communication channel.

Given the channel observation vector $\boldsymbol{y}$, the successive cancellation decoder estimates $\hat{u}_0, \hat{u}_1, \cdots, \hat{u}_{n-1}$ one-by-one by first efficiently calculating a pair of probabilities:

$$P(\hat{u}_i = 0|\boldsymbol{y}, \boldsymbol{u}_0^{i-1}) \text{ and } P(\hat{u}_i = 1|\boldsymbol{y}, \boldsymbol{u}_0^{i-1}) \tag{3.2}$$

at each step; and then making a decision on $\hat{u}_i \in \{0, 1\}$. The priority is with a genie-like decision when u_i has a fixed frozen value. Otherwise, the most likely case is selected. The location of frozen bits depends on noise level of the corresponding bit-channels, which itself depends on the communication channel. Ideally, the $n - k$ noisiest bit-channels are frozen, leaving the k less noisy ones for the information bits. There are multiple algorithms in the literature capable of tracking these noise levels very efficiently even for large values of n. Constructions based on Gaussian approximation [81] and channel degradation [38] are among the most common methods. We refer readers to [10] for the detailed definitions along with some discussions on the theory of polar codes and their capacity achieving properties. It is also to be noted that while we do not gain anything from a fixed initialization of the frozen values in the case of the symmetric channels, allocating some dynamic values such as parities of the information bits may increase the minimum distance of the code, and hence improve the error rate [82].

Successive cancellation decoding is an iterative algorithm that utilizes the butterfly structure of polar codes to efficiently calculate the probability pair in (3.2) for $i = 0, 1, \cdots, n-1$ respectively. Upon calculation of the probability pairs for each $i \in \{0, 1, \cdots, n-1\}$, the decision on $\hat{u}_i$ is prioritized by first looking up the available frozen values, and then the freshly calculated probabilities. Algorithm 1 provides a high-level description of the SC decoding algorithm.

Let p_i denote the probability of making an incorrect decision for u_i. Then it is easy to show that the frame error rate (FER) is formulated as

$$P_e = 1 - \prod_{i \in \mathcal{I}} (1 - p_i), \tag{3.3}$$

where $\mathcal{I} \subset \{0, 1, \cdots, n-1\}$ denotes the subset of indices corresponding to the information bits [see [83] for the proof.] A trivial upper bound, commonly known as the *union bound* can

Algorithm 1 Summary of the SC decoding algorithm

```
Input: received vector $\boldsymbol{y}$
Output: a decoded vector $\hat{\boldsymbol{u}}$
for $i = 0, 1, \ldots, n-1$ do
    calculate $P(u_i = 0|\boldsymbol{y}, u_0^{i-1})$ and $P(u_i = 1|\boldsymbol{y}, u_0^{i-1})$
    if $u_i$ is frozen then
        set $\hat{u}_i$ to the frozen value of $u_i$
    else
        if $P(u_i = 0|\boldsymbol{y}, u_0^{i-1}) > P(u_i = 1|\boldsymbol{y}, u_0^{i-1})$ then
            set $\hat{u}_i \leftarrow 0$
        else
            set $\hat{u}_i \leftarrow 1$
return the length-$k$ information sub-vector of $\hat{\boldsymbol{u}}$
```

be deducted from (3.3) as

$$P_e \leqslant \sum_{i \in \mathcal{I}} p_i. \tag{3.4}$$

An important observation is that while almost all bit-channels W_i $(i \in \mathcal{I})$ are almost noiseless, but $p_i's$ range in decibels is significantly large. For example, if one was able to replace to replace all p_i's for $i \in \mathcal{I}$ with some kind of average such as

$$p_{\text{ave}} \triangleq \frac{1}{|\mathcal{I}|} \sum_{i \in \mathcal{I}} p_i \qquad \text{or} \qquad p_{\text{ave}} \triangleq \big(\prod_{i \in \mathcal{I}} p_i \big)^{\frac{1}{|\mathcal{I}|}}, \tag{3.5}$$

then overall frame error rate would become much smaller. In other words,

$$P_{e,\text{avg}} = 1 - (1 - p_{\text{avg}})^{|\mathcal{I}|} \ll P_e = 1 - \prod_{i \in \mathcal{I}} (1 - p_i) \tag{3.6}$$

The proof of the above argument is based on a simple induction on $|\mathcal{I}|$ and the facts that

$$(1-p_1)(1-p_2) \geqslant (1-\frac{p_1+p_2}{2})^2, \tag{3.7}$$

$$(1-p_1)(1-p_2) \geqslant (1-\sqrt{p_1p_2})^2. \tag{3.8}$$

This motivates us to search for a scheme that helps to balance out the noise levels of these bit-channels. We propose two such algorithms in the next two sections. But before getting there, we need to discuss a secondary motivation behind this work.

Soon after discovery of polar codes, it was pointed out that the performance of SC decoder can be further improved if it is equipped with some side information that can help with correcting the first few mistakes it may make during the process. Such mechanism is usually cited as the *Arıkan's genie* due to its discovery by E. Arıkan. We explain this over the following example.

Example 4. Let us assume that the SC decoder is equipped with some side information that helps correcting its first mistake (if there is any) during the decoding process. The probability of successful decoding is then given by

$$P_{\text{success}} = \underbrace{\prod_{i\in\mathcal{I}}(1-p_i)}_{\text{no genie needed}} + \underbrace{\sum_{i\in\mathcal{I}} p_i \prod_{j\in\mathcal{I}, j\neq i} (1-p_j)}_{\text{1-genie needed}}. \tag{3.9}$$

The FER in this case is then expressed as

$$P_e^{(1)} = 1 - \big(\prod_{i\in\mathcal{I}}(1-p_i)\big)\big(1+\sum_{i\in\mathcal{I}}\frac{p_i}{1-p_i}\big). \tag{3.10}$$

The proof follows by dividing the event of successful decoding into two disjoint events: successful decoding without using the genie; and successful decoding by using the genie once. The latter itself can be divided into $|\mathcal{I}|$ disjoint events based on the the index of the bit-channel W_i for which the genie was used. The probability of successful decoding condition on using

Algorithm 2 A high-level description of SC decoding equipped with γ-limited Arikan's genie

Input: received vector $\boldsymbol{y}$

Output: a decoded vector $\hat{\boldsymbol{u}}$

```
set $g \leftarrow \gamma$
for $i = 0, 1, \ldots, n-1$ do
    calculate $P(u_i = 0|\boldsymbol{y}, u_0^{i-1})$ and $P(u_i = 1|\boldsymbol{y}, u_0^{i-1})$
    if $u_i$ is frozen then
        set $\hat{u}_i$ to the frozen value of $u_i$
    else
        if $P(u_i = 0|\boldsymbol{y}, u_0^{i-1}) > P(u_i = 1|\boldsymbol{y}, u_0^{i-1})$ then
            set $\hat{u}_i \leftarrow 0$
        else
            set $\hat{u}_i \leftarrow 1$
        if $\hat{u}_i \neq u_i$ and $\gamma > 0$ then
            set $\hat{u}_i \leftarrow u_i$
              set $\gamma \leftarrow (\gamma - 1)$
return the length-$k$ information sub-vector of $\hat{\boldsymbol{u}}$
```

the genie on the i-th bit-channel is given by $\prod_{i \in \mathcal{I}}(1 - p_i)$, while the probability of genie being needed for the i-th bit-channel is given by p_i.

By allowing the SC decoder to use the Arikan's genie up to γ times, a similar expression for FER can be deducted, which is given in Theorem 12. We remove the proof to avoid duplicate materials. But, we point out that it follows from an inclusion-exclusion kind of argument similar to that of the Example 4.

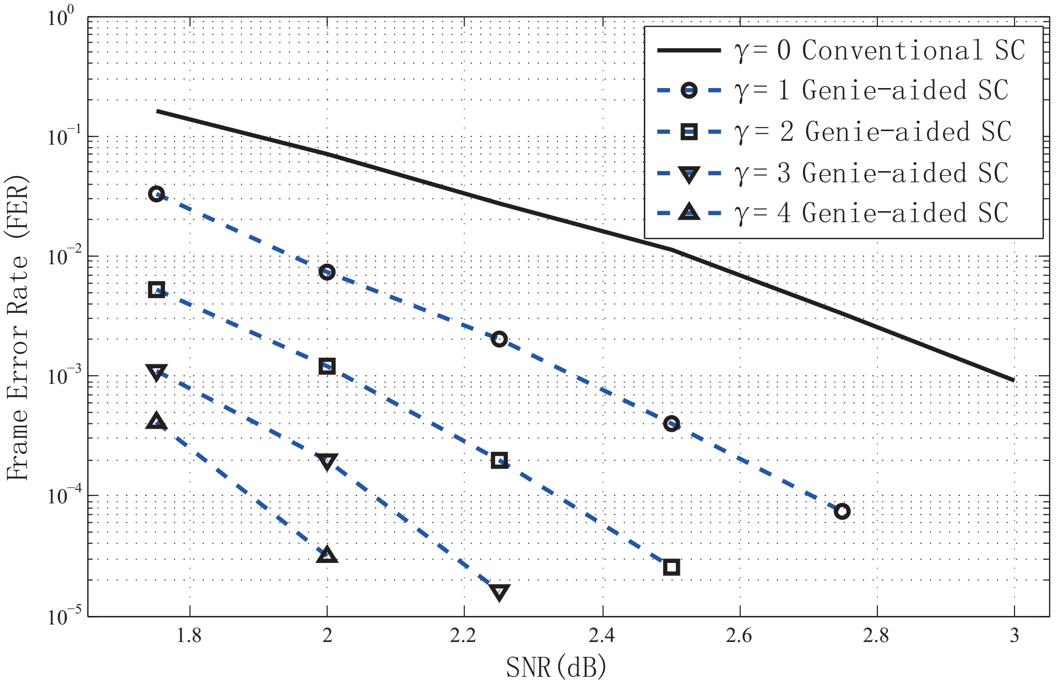


Figure 3.1: Frame error rate (FER) of $[1024, 512]$-polar codes under SC decoding equipped with Arikan's genie.

Theorem 12. *Assume SC decoder is allows to use Arikan's genie provided from some side information for up to γ times. An explicit expression for FER is then given by*

$$P_e^{(\gamma)} = 1 - \left(\prod_{i \in \mathcal{I}} (1 - p_i)\right)\left(1 + \sum_{\substack{\Gamma \subset \mathcal{I} \\ |\Gamma| \leqslant \gamma}} \prod_{i \in \Gamma} \frac{p_i}{1 - p_i}\right). \tag{3.11}$$

A summary of the SC decoding algorithm equipped with Arikan's genie is provided in Algorithm 2. As shown in Figure 3.1, having access to the Arikan's genie, even for a very limited number of times, improves performance of the polar codes drastically. Here, γ is the maximum number of bit corrections provided from the genie. However, we do not always have access to such side information and implementation of the polar genie without side information is completely nontrivial.

The first attempts in simulating the polar genie were presented along with the list decoding algorithm of polar codes in [39]. Polar list decoder is capable of providing a list of L highly likely candidates for $\hat{\boldsymbol{u}}$. It is shown that selecting the most likely candidate from the list brings the error rate down to near optimal value (ML) even when small values of L are taken. However, by precoding the k information bits with a *cyclic redundancy check* (CRC), one can first reject the unverified candidates from the list and then make the ML selection. Hence, the CRC acts like a genie that informs the decoder about invalid codewords. Simulation results provided in [39] showed a drastic performance improvement of list + CRC decoder over conventional polar codes. Despite the existence of more advanced decoding algorithms such as those proposed in [84, 85], which eliminate most of the unnecessary calculations in the list decoder, a true simulation of the Arikan's genie, that is capable of correcting errors on-the-go, does not exist in the literature.

3.2 Viterbi Decoding as an Implementation of the Genie

Despite the impressive empirical results of polar codes under list decoder combined with CRC, their increased decoding complexity makes them impractical for many scenarios. More importantly, the genie (CRC) is not activated until the end of the successive cancellation decoding. Hence, it cannot eliminate incorrect and unnecessary calculations on the go. An imperfect solution to this problem would be to implement multiple shorter and disjoint CRCs, which would provide us a few check points during the decoding process. A similar approach is taken in [86], where authors utilize multiple CRCs in order to terminate some percentage of the unnecessary calculations before reaching the end, and hence improves both space and time complexity of the list decoder. However, not only the genie remains mostly inactive, but also the dependency on the *list* is still in place since the CRC has no local correction capabilities.

In this section, we introduce the first implementation of the polar genie that does not rely on the list decoder. Our construction is based on replacing the CRC with a convolutional code, which provides some local correction capability over its whole length. The SC decoder is then also equipped with the Viterbi Algorithm (VA) [87] to detect incorrectly decoded bits with some short delays. Upon detection of error by the VA module, a genie-like feedback is activated to set back the SC decoder to the corresponding index. The SC decoder then restarts its calculations by utilizing the newly provided values from the genie. This structure allows the successive cancellation polar decoder to verify its output by running it through a Viterbi decoder for the convolutional code. In contrast to conventional CRC-aided list decoding, wherein incorrect decoding paths are rejected only after reaching the last information bit, the Viterbi decoder detects incorrect decisions "on the fly" after a short delay. Simulation results show noticeable improvements by utilizing even simplest convolutional codes. List independent structure of the proposed method also translates into the average computational complexity being very close to that of the SC decoder, particularly for the high SNR regime in which the feedback is rarely activated, which makes it suitable for some practical scenarios.

A different approach has been in studied in [88], where conventional CRCs are replaced with some pseudo-random cross-checks between consecutive blocks of the polar code in order to significantly reduce the decoding latency of list decoder. Our decoder structure has some similarities to the SC flip decoder that was first proposed in [89]. SC flip decoding does not require a list, but it again utilizes the CRC to validate the codeword upon reaching the last bit. When the decoded message is rejected by the CRC, the decoder resets back to a bit-channel that is estimated to be the one with highest probability of mistake. The set-back mechanism is similar to the Viterbi-aided SC decoding algorithm in our paper. However, the method used to estimate the location of error is very inefficient since the CRC does not have any local error detection capabilities. We overcome this problem by replacing the CRC with

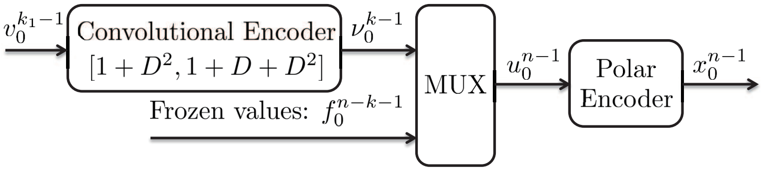


Figure 3.2: Encoding structure of the concatenated polar code from Example 5.

an optimized convolutional code and then use VA to detect errors on the go as if there was a genie. A different concatenation scheme of polar codes with convolutional codes was also considered in [90].

We recall that n denotes the length of the uncoded vector $\boldsymbol{u}$ as defined in (3.1) where its subset of $n-k$ frozen indices are denoted by $\mathcal{F}$. That leaves us with k indices, denoted by $\mathcal{I}$, that traditionally were reserved for only information bits. The improvement in our constructions comes from a convolutional code that we use to precode some k_1 pure information bits into $n_1 = k$ coded ones. These k bits are then seated in the coordinates in $\mathcal{I}$ and sent to a conventional polar encoder as if they were all information bits. It can be viewed as concatenation between a $[n_1, k_1]$ convolutional code and a $[n, k = n_1]$ polar code.

Example 5. Let $G_{\text{conv}} = [1+D^2, 1+D+D^2]$ denote the generator matrix of a rate-$\frac{1}{2}$ convolutional code. The terminated convolutional code of length $n_1 = 2k_1 + 4$ is then generated from a given sequence of k_1 pure information bits, $\boldsymbol{v}_0^{k_1-1}$. Let $\nu_0^{n_1-1}$ denote the generated convolutional codeword. Next, the uncoded vector $\boldsymbol{u}$ is generated by multiplexing these $k = n_1$ bits with $n-k$ frozen bits (all-zero) while preserving the appearance order of ν_0^{k-1} in $\boldsymbol{u}$. The length-n vector $\boldsymbol{u}$ is then multiplied by G defined in (3.1) to form the length-n polar codeword $\boldsymbol{x}$. Figure 3.2 provides the schematic of the encoder. The overall rate is given by

$$R = \frac{k_1}{n} \sim \frac{k}{2n}. \tag{3.12}$$

Convolutional codes provide a natural local error correction capability, which can be utilized as a genie-like aid provided for successive cancellation decoder. The traceback depth in convolutional codes determines the required delay to validate a bit from the received sequence by Viterbi Algorithm. It is known that the successive cancellation of polar codes suffers from the error propagation phenomena, *i.e.* when it makes the first mistake during the decoding process, it is bound to make a large number of additional mistakes on the average. This property translates to a poor bit error rate (BER) for polar codes in general. Hence, it is desired to utilize a convolutional code with *low* traceback depth in order to increase the chance of correcting an error before future incorrect bits appear.

The traceback depth is estimated to have a linear relation with the constraint length of convolutional codes. However, as we point out in the following, precoding with convolutional codes with *large* constraint lengths has its own merits:

- The free distance, denoted by d_{free}, measures the error correction capability of convolutional codes; and, the common constructions of convolutional codes with large d_{free} usually require a large constraint length.
- We cannot tolerate a large rate loss only to add a second layer protection for the information bits, since a simple reduced-rate polar code may show a better performance in fair comparisons. On the other hand, common constructions of convolutional codes with high rates also require larger constraint lengths [91].

We observe that the construction of the optimal convoutional code is nontrivial; indeed, it is not even clear if there exists a choice that improves the overall performance.

Lastly, we point out that not all of k bits in $\mathcal{I}$ require extra protection. In fact, most of the indices in $\mathcal{I}$ correspond to almost-noiseless bit channels. Accordingly, one can modify the structure in Figure 3.2 to protect only a small portion of the information bits with con-

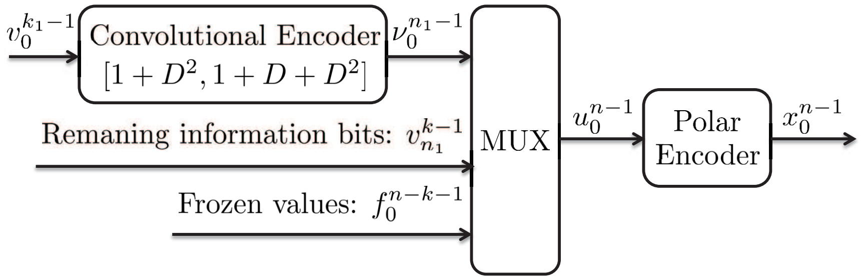


Figure 3.3: Improved encoder structure with shorter convolutional codes.

volutional codes, and hence drop the $k = n_1$ requirement. This modification allows us to use convolutional codes with low traceback depths and simulate the polar genie more efficiently. Figure 3.3 depicts the improved encoding scheme. Less noisy bit channels are left unprotected. The new overall rate is given by

$$R = \frac{k_1 + n_1 - k}{n}. \tag{3.13}$$

With less concern about the rate loss from convolutional code, we take the simple rate-$\frac{1}{2}$ convolutional code from Example 5 and search for the optimal length of underlying convolutional code, n_1, to provide a second layer error protection for the n_1 noisiest bits. FER at each code length is derived according to methods discussed in the next section. Figure 3.4 shows the search oriented simulation results for $[8192, 4096]$-concatenated polar codes. In this figure, k_1 denotes the length of information bit sequence fed to the convolutional encoder with generator matrix G defined in Example 5. The length of the terminated convolutional codeword is given by $n_1 = 2k_1 + 4$. The simulation results for this setup show that the optimal codelength is given by $n_1^{\text{opt}} = 108$.

Next, we propose the Viterbi-aided SC decoding algorithm that utilizes the concatenated convolutional code to simulate Arıkan's genie on-the-go during the decoding process.

Recall that $\mathcal{I} \cup \mathcal{F} = \{0, \cdots, n-1\}$ in which $\mathcal{I}$ denotes the location of the unfrozen indices in $\boldsymbol{u}$. Let $\mathcal{I}_{\text{conv}} = \{\sigma_0, \sigma_1, \cdots, \sigma_{n_1-1}\} \subset \mathcal{I}$ correspond to the indices in which the

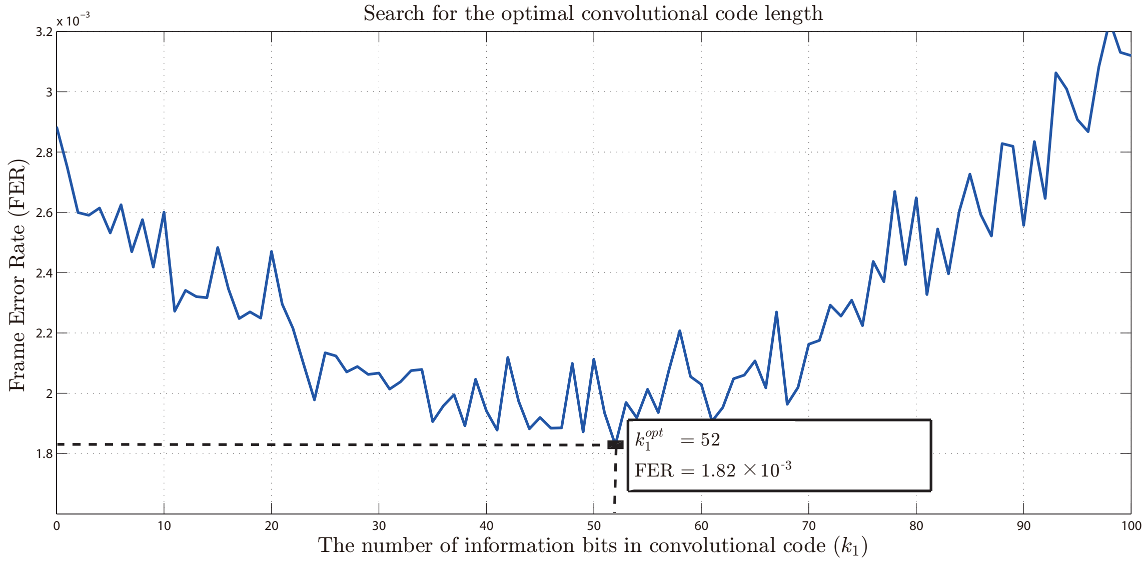


Figure 3.4: Frame error rate of $[8192, 4096]$ polar codes concatenated with convolutional codes of different lengths.

symbols of length-n_1 convolutional code are located. For simplicity we assume, $\sigma_0 < \sigma_1 < \cdots < \sigma_{n_1-1}$. Viterbi-aided SC decoding of polar codes works as follows. Given the received vector $\boldsymbol{y}$ from the channel, a SC decoding block starts by estimating $\hat{u}_0, \hat{u}_1, \cdots$ one by one. After estimating $\hat{u}_i$, two different cases may appear:

- $i \in (\mathcal{I} \cup \mathcal{F}) \setminus \mathcal{I}_{\text{conv}}$: decoder continues the process normally as it would have done in the absence of the convolutional code.
- $i \in \mathcal{I}_{\text{conv}}$ or equivalently $i = \sigma_j$ for some j : the freshly estimated value of $\hat{u}_{\sigma_j}$ is fed to the Viterbi decoder for further validations.

An overview of the concatenation between SC decoder and the VA module is presented in Figure 3.5, where output from SC decoder is split into two sequences: unprotected less-noisy bits, and n_1 bits that form the convolutional codeword for the second step verification. The Viterbi decoder takes as input the estimated values of $\hat{u}_{\sigma_j}$ from the SC decoding module's

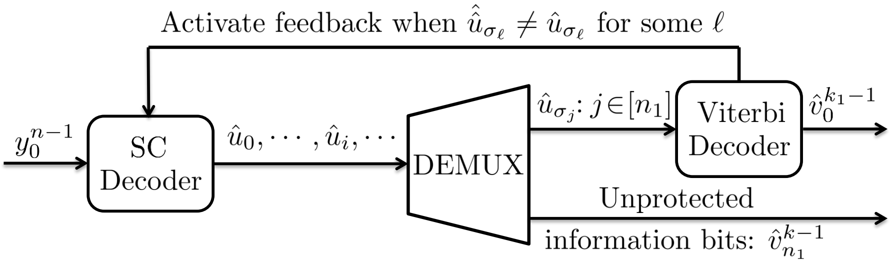

Activate feedback when $\hat{\hat{u}}_{\sigma_\ell} \neq \hat{u}_{\sigma_\ell}$ for some ℓ
y_0^{n-1}
SC Decoder
$\hat{u}_0, \cdots, \hat{u}_i, \cdots$
DEMUX
$\hat{u}_{\sigma_j}$: $j \in [n_1]$
Viterbi Decoder
$\hat{v}_0^{k_1-1}$
Unprotected information bits: $\hat{v}_{n_1}^{k-1}$

Figure 3.5: An overview of the Viterbi-aided SC decoder.

output. If discovers any disparities on $\hat{u}_{\sigma_j}$, or those provided earlier from the SC decoder, *i.e.* $\hat{u}_{\sigma_\ell},\ \ell \leqslant j$, a feedback arm gets activated. Upon activation of the feedback, the correct value of $\hat{u}_{\sigma_\ell}$, denoted by $\hat{\hat{u}}_{\sigma_\ell}$, is sent back to the SC decoding block. The SC decoder then resets its index back to σ_ℓ, and restarts the calculations from there by replacing $\hat{u}_{\sigma_\ell}$ with the new value provided from Viterbi decoder. However, there are two main noticeable challenges about this approach that slightly complicate the decoder structure.

First, we point out again that the Viterbi Algorithm has some delay in validating the value of $\hat{u}_{\sigma_j}$. This is due to the fact that VA is only capable of decoding a symbol, once all of the existing trellis paths agree on that index. Note that the input symbol for the Viterbi algorithm is formed of 2 bits. Let us denote input and output sequences of the Viterbi decoder by $\hat{u}_{\sigma_0}\hat{u}_{\sigma_1}, \hat{u}_{\sigma_2}\hat{u}_{\sigma_3}, \cdots$ and $\hat{\hat{u}}_{\sigma_0}\hat{\hat{u}}_{\sigma_1}, \hat{\hat{u}}_{\sigma_2}\hat{\hat{u}}_{\sigma_3}, \cdots$ respectively. As depicted in Figure 3.6, there is some delay between the last received input symbol and the most recent verified one, which is statistically bounded by traceback depth of the convolutional code [in this case $\tau_b \approx 10$]. The sample trellis depicts the delay in symbol verification. A symbol is verified when all of the trellis paths agree on it. The feedback is activated when a symbol is verified to be incorrectly estimated by SC.

The second complication with this feedback mechanism is that VA module is only capable of finding some disparities. However, if the paths on trellis agree on some symbol other

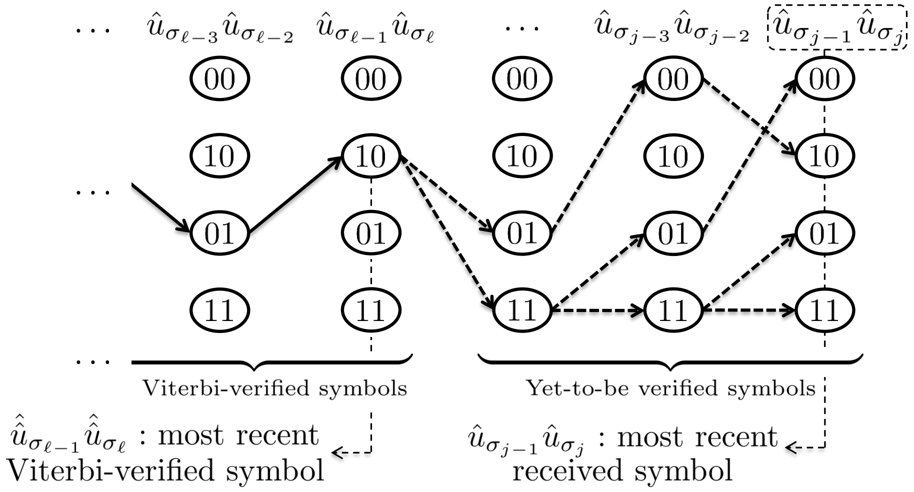
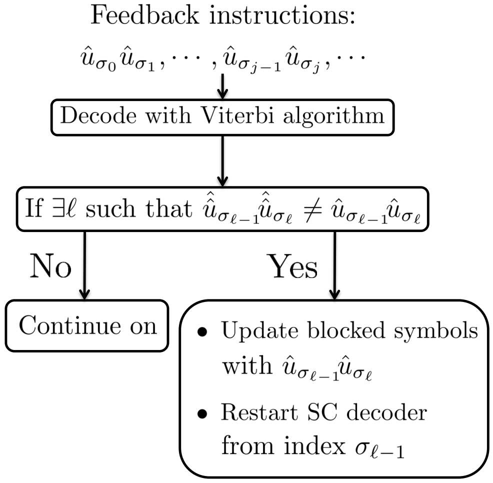

$\cdots$ $\hat{u}_{\sigma_{\ell-3}}\hat{u}_{\sigma_{\ell-2}}$ $\hat{u}_{\sigma_{\ell-1}}\hat{u}_{\sigma_\ell}$ $\cdots$ $\hat{u}_{\sigma_{j-3}}\hat{u}_{\sigma_{j-2}}$ $\hat{u}_{\sigma_{j-1}}\hat{u}_{\sigma_j}$
00
10
01
11
Viterbi-verified symbols
Yet-to-be verified symbols
$\hat{\hat{u}}_{\sigma_{\ell-1}}\hat{\hat{u}}_{\sigma_\ell}$: most recent Viterbi-verified symbol
$\hat{u}_{\sigma_{j-1}}\hat{u}_{\sigma_j}$: most recent received symbol

Figure 3.6: A Snapshot of the Viterbi-aided SC decoding algorithm.

than the one provides from SC, all one can deduce is that the SC decoder made a mistake on that symbol and has to repeat calculations from that index to discover the correct value itself. Let us formulate this by assuming that the mismatch between input and output symbols is discovered at index $\sigma_{\ell-1}\sigma_\ell$, *i.e.* $\hat{\hat{u}}_{\sigma_{\ell-1}}\hat{\hat{u}}_{\sigma_\ell} \neq \hat{u}_{\sigma_{\ell-1}}\hat{u}_{\sigma_\ell}$. Note that we are not allowed to immediately replace $\hat{u}_{\sigma_{\ell-1}}\hat{u}_{\sigma_\ell}$ with $\hat{\hat{u}}_{\sigma_{\ell-1}}\hat{\hat{u}}_{\sigma_\ell}$ when a mismatch occurs since the latter is a function of the input sequence and has to be re-calculated according to the new input symbols. So, the feedback mechanism adds the incorrect symbol $\hat{u}_{\sigma_{\ell-1}}\hat{u}_{\sigma_\ell}$ to the list of blocked symbols for indices $\{\sigma_{\ell-1}, \sigma_\ell\}$. Then, the SC decoder restarts the decoding process at $\sigma_{\ell-1}$ by selecting the next most likely unblocked symbol. To determine the next most likely symbol, we first flip value of the less reliable bit-channel. If that comes back unverified as well, we proceed by flipping the value of the more reliable bit-channel; and, if they both return unverified, the SC decoder flips both of them. Detailed instructions are tabulated in Figure 3.7. Here, an error consists of a symbol that gets corrected by the Viterbi algorithm. Blocked symbols are stored in the memory. The calculations are always restarted from bit-channel $u_{\sigma_{\ell-1}}$. Then, SC decoder keeps or flips the natural decisions based on instructions in the table.


Feedback instructions:
$\hat{u}_{\sigma_0}\hat{u}_{\sigma_1},\cdots,\hat{u}_{\sigma_{j-1}}\hat{u}_{\sigma_j},\cdots$
Decode with Viterbi algorithm
If $\exists\ell$ such that $\hat{\hat{u}}_{\sigma_{\ell-1}}\hat{\hat{u}}_{\sigma_\ell} \neq \hat{u}_{\sigma_{\ell-1}}\hat{u}_{\sigma_\ell}$
No
Yes
Continue on
Update blocked symbols with $\hat{u}_{\sigma_{\ell-1}}\hat{u}_{\sigma_\ell}$
Restart SC decoder from index $\sigma_{\ell-1}$

SC decoder instruction (with blocked symbols):

More noisy index	Blocked symbols $\hat{u}_{\sigma_{\ell-1}}\hat{u}_{\sigma_\ell}$			SC instructions	
	1	2	3	$\hat{u}_{\sigma_{\ell-1}}$	$\hat{u}_{\sigma_\ell}$
$\sigma_{\ell-1}$	$\alpha\beta$			flip	derive
$\sigma_{\ell-1}$	$\alpha\beta$	$\sim\alpha\delta$		keep	flip
$\sigma_{\ell-1}$	$\alpha\beta$	$\sim\alpha\delta$	$\alpha\sim\beta$	flip	flip
σ_ℓ	$\alpha\beta$			keep	flip
σ_ℓ	$\alpha\beta$	$\alpha\sim\beta$		flip	derive
σ_ℓ	$\alpha\beta$	$\alpha\sim\beta$	$\sim\alpha\delta$	flip	flip

$\sim 0 = 1, \sim 1 = 0$

Figure 3.7: Instructions for reseting the SC decoder back to the discovered error location.

Example 6. Suppose that the SC decoder is reset to index $\sigma_{\ell-1}$ and is provided with an ordered set of blocked symbols $\{\alpha\beta, \sim\alpha\delta\}$ from the Viterbi decoder, where $\sim\alpha$ denotes the flipped value of α. Further assume that $\sigma_{\ell-1}$ corresponds to the less reliable bit-channel between the two. It is observed that both values of $\alpha, \sim\alpha$ got rejected from the Viterbi decoder. Hence, the chances are that the decoding mistake was made on the more reliable bit-channel in the first place. SC decoder then proceeds by keeping $\hat{u}_{\sigma_{\ell-1}} = \alpha$ and flipping the decision on the next bit, *i.e.* $\hat{u}_{\sigma_\ell} =\sim\beta$. It is recommended to track these instructions over Figure 3.7 as well.

Before proceeding to the numerical results, we point out that the Viterbi algorithm also accepts soft information (symbol likelihoods) as input. In the other hand, the SC decoder is also capable of calculating the likelihoods for the bits in its output sequence. One may wonder if we can improve the current decoding scheme by feeding the calculated soft information from the SC decoder to the Viterbi decoder instead of the hard decisions. However, despite its simplicity, new instructions for the SC decoder with blocked symbols are completely nontrivial and require further investigations.

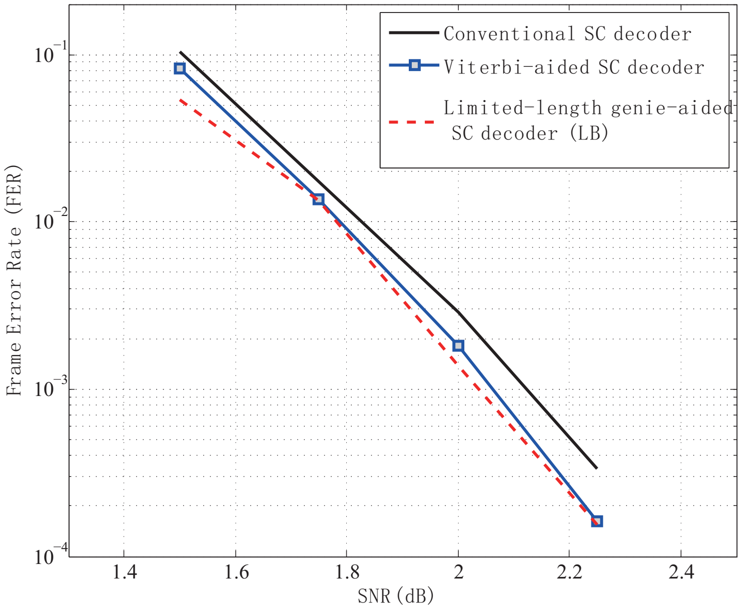


Figure 3.8: Performance comparison between conventional polar codes under SC decoder and concatenated polar codes under Viterbi-aided SC decoder.

We assume the communication channel to be B-AWGN with SNR ranging from 1.5 dB to 2.25 dB. The optimal length of the convolutional code for each SNR point is constructed by the simulation-based search method introduced earlier. The parameters of the concatenated polar codes are given by $[n, k_1] = [8192, 4096]$, with length of the concatenated convolutional code varying between $100 \leqslant n_1 \leqslant 128$. Figure 3.8 depicts the comparison between the newly proposed codes under Viterbi-aided SC decoding, and the conventional $[8192, 4096]$-polar codes under SC decoding. A noticeable improvement is observed particularity at high SNR regime. The lower bound (LB) curve corresponds to a genie-aided SC decoder, which is enabled on all those bit-channels whose coordinates belong to the convolutional codeword. In other words, it prevents SC decoder from making decoding mistakes on those n_1 bit-channels in $\mathcal{I}_{\text{conv}}$ entirely. Surprisingly, the Viterbi-aided SC decoder fits very close to the lower bound, which highlights the efficiency of Arıkan's genie simulation via this method.

Table 3.1: Percentage of iterations required during Viterbi-aided SC decoding at SNR = 2dB.

# of iterations	0	1	$2-5$	$6-15$	$16+$
Probability $(\%)$	99.5	0.35	0.06	0.05	0.04

Despite the performance being very close to the lower bound while keeping the average decoding complexity almost unchanged, polar list decoder outperforms the currently proposed Viterbi-aided SC decoder. However, it is possible to extend this method by allocating a second convolutional code with rate higher than $\frac{1}{2}$ to a subset of those $k-n_1$ unprotected bit-channels. We require a secondary simulation-based search to find the optimal convolutional precoder for the next batch of information bits similar to the methods proposed in this section It is then projected to reach the next lower bound on FER, which corresponds to the genie being enabled on locations that belong to both of the convolutional codewords.

To conclude, we provide the complexity analysis by first mentioning that the decoding complexity of the Viterbi algorithm is an asymptotic linear function of n_1 and hence can be ignored when compared to the SC decoder. More importantly, the Viterbi decoding block never activates the feedback if the message is correctly estimated by SC itself. A quick look at the performance of the SC decoder in Figure 3.8 indicates that this event (successful decoding) for instance happens with very high probability [Pr. > 0.99] at SNR = 2dB. Furthermore, a single iteration fixes the error in most of the cases, in which SC made a mistake. We refer the reader to Table 3.1 for the distribution of required iterations at SNR = 2dB. The average number of set-backs called on SC decoder is then given by $\sim 2.13\%$, which shows an extremely low increase in average decoding complexity.

This chapter contains materials as it appears in [8]:

- A. Fazeli, K. Tian, and A. Vardy, “Viterbi-Aided Successive-Cancellation Decoding of Polar Codes," *Proceedings of IEEE Global Communications Conference*, Dec. 17, pp. 1-6.

It also, in part, contains materials from the paper in preparation:

- A. Fazeli, K. Tian, and A. Vardy, “Convolutional decoding of polar codes", to be submitted to *IEEE Transactions on Information Theory*.

The dissertation author was the primary investigator and author of these papers.

Chapter 4

Polar Codes for Deletion Channels

4.1 Deletion Channels: Overview

Imperfect sampling devices can cause synchronization problems in a communication system, which can cause loss of a few received symbols or sometimes observing a few unwanted ones among the received ones. These types of error are usually referred to by insertion or deletion errors. Channels corrupted by insertions or deletions have memory, hence the polar coding techniques developed for memoryless channels can not be performed straightforwardly. See [53, 56] for detailed surveys of the synchronization errors.

Levenshtein [92] was first to propose algebraic error-correcting codes based on Varshamov-Tenengolts (VT) codes [93] that was capable of correcting one asymmetric error. In general, the performances of the d-deletion-correcting codes are measured in their asymptotic redundancy. For example, Levenshtein proved that VT codes are asymptotically optimal single synchronization-correcting codes. He also derived the lower bound $\Theta(d \log n)$ on the asymptotic redundancy of codes that can correct d deletions with zero error probability.

Various coding schemes with zero-error decoders have been proposed to correct the deletion errors. Some notable algebraic codes that generalize Levenshtein's scheme for arbitrary d's include [94], [95], and [96]. Gallager [97] was first to utilize convolutional codes for correcting the synchronization errors. Brink *et al.* [98] proposed an improved convolutional encoder which can produce a subset of Levenshtein single-synchronization-correcting codes periodically by pruning branch. A decoding scheme based on parallel Viterbi algorithms is proposed in [99], which later on was improved in [100] by adopting the Levenshtein distance as metric of convolutional decoder. There are also multiple concatenation-based coding schemes. Notable such constructions include concatenation of LDPC codes with repetition codes in [101], concatenation of Reed-Solomon codes with repetition codes in [102], concatenation of LDPC codes with Levenshtein single-synchronization-correcting codes in [103], and non-zero rate codes for channels with $d = pn$ deletions [104]. Extension of polar codes to channels with memory also drew a lot of community attention. We refer readers to two recent works [55, 105] which establish strong polarization theorems for processes with memory.

In this chapter, we propose a novel polar coding scheme for the d-deletion channel, where d could be a fixed value or a sub-linear function of the code-length, *i.e.* $d = o(n)$. First implementation of a modified low-complexity SC decoder for deletion channels was initially presented in [106]. Here, we present new theoretical results that provide a mathematical guarantee for the performance of our SC decoder. Our scheme is based on a probabilistic decoding algorithm (instead of the zero-error decoders), which requires only $O(d^2 n \log n)$ computational complexity in comparison to the naive implementation with $O(n^{d+1} \log n)$ computational complexity in [107]. It provably achieves the symmetric information rate of any binary discrete memoryless channel with d deletions, while there is no restriction on the error/deletion patterns. Furthermore, we show how the same setup applies to binary noisy channels with additional deletions.

Let us begin by first explaining our channel model. Here, n denotes the block-length and d denotes the number of deletions. In general, d could be a constant, a function of the block-length such as cn, or a random number generated from a binomial distribution. Furthermore, the location of the deleted symbols could be selected uniformly at random, or in some cases according to another probability distribution over all $\binom{n}{d}$ possible scenarios. We denote such probability distributions by $\mathcal{D}_{n,d}$ if required. We follow the conventional definition of the binary deletion channel that appeared in [92], where the transmitted symbols are denotes by $X_1^n \in \{0,1\}^n$ while $Y_1^n \in \mathcal{Y}^n$ denotes the received symbols prior to the deletion effect. The final $n - d$ received symbols are also given by $\tilde{Y}_1^{n-d} \in \mathcal{Y}^{n-d}$. Here, d randomly chosen symbols among x_i's go through the deletion transformation: $x_i \to \wedge$ ($\wedge$ denotes the empty word.)

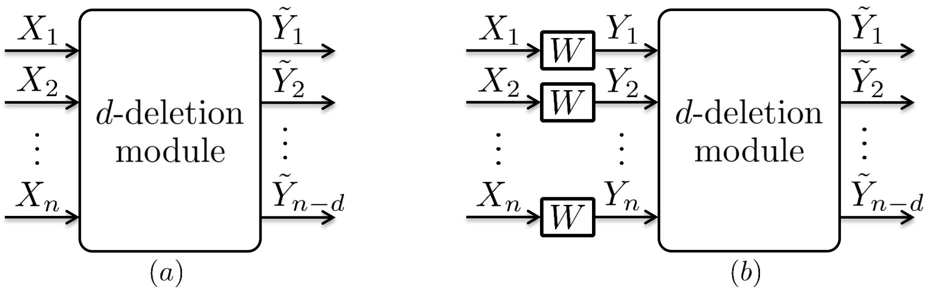


Figure 4.1: Channel models in presence of deletions.

While our focus in this paper is on noiseless deletion channels, we point out that all of the methods used here are applicable to general noisy channels with deletion. The noisy d-deletion channel can be considered as the cascade between B-DMC and d-deletion channel. There is no commonly known channel model for the noisy d-deletion channel. However, since the synchronization problems often happen at the receiver and after the noise effect of the wireless channel, we place the DMC prior to the d deletion channel. Here, we only look at the less-complex case of fixed number of deletions. However, we explain the necessary steps one would want to take to generalize these results to other scenarios. We point out that the same framework, in both theory and implementation, applies to noisy deletion channels as well. Hence, we define everything with respect to the more general case for which we are not aware of any known practical coding schemes that achieve capacity at presence of even fixed number of deletions. Figure 4.1 captures both noisy and noiseless deletion channels as defined above. Figure 4.1.a is the textbook definition of d-deletion channel, while Figure 4.1.b illustrates the noisy d-deletion channel that is defined as a concatenation between n i.i.d. copies of a B-DMC and the d-deletion channel.

Assume the underlying transmission to take place over a binary-input discrete memoryless channel (*B-DMC*) $W : \{0,1\} \to \mathcal{Y}$ with input alphabet $\{0,1\}$, output alphabet $\mathcal{Y}$, and transition probabilities $W(y|x)$. In our simulation, we mainly assume W to be a *binary* *symmetric channel* (BSC). We use W^n to denote the channel corresponding to n uses of W prior to the deletions, which translates to

$$W^n : \{0,1\}^n \to \mathcal{Y}^n \qquad \text{with} \qquad W^n(y_1^n|x_1^n) = \prod_{i=1}^{n} W(y_i|x_i). \tag{4.1}$$

Similarly, we use $\tilde{W}^{n,d}(\tilde{y}_1^{n-d}|x_1^n)$ to denote the corresponding channel after the d deletions take effect. In this notation, we assume that the d deletions are selected uniformly at random from the n possible locations. Alternatively, we denote the channel by $\tilde{W}^{n,\mathcal{D}}(\tilde{y}_1^{n-d}|x_1^n)$ if the deleted symbols are selected according to a probability distribution such as $\mathcal{D}_{n,d}$ or simply $\mathcal{D}$ for when the parameters are clear in the context.

For any given B-DMC such as W, we will deal with its symmetric capacity

$$I(W) \triangleq \sum_{y\in\mathcal{Y}} \sum_{x\in\{0,1\}} \frac{1}{2} W(y|x) \log \frac{W(y|x)}{\frac{1}{2}W(y|0) + \frac{1}{2}W(y|1)} \tag{4.2}$$

and its Bhattacharyya parameter that is an upper bound on the error probability under ML decoder and is defined by

$$Z(W) \triangleq \sum_{y\in\mathcal{Y}} \sqrt{W(y|0)W(y|1)}. \tag{4.3}$$

Let us now review basic concepts of polar codes and recite the definition of polar bit-channels in presence of deletions. The encoder in our setup will be an (n, k) polar encoder, where $n = 2^m$ corresponds to m levels of polarization and k denotes the number of information bits in the polar code that leaves us with $n - k$ frozen bits. The relation between x_i's and u_i's is given by

$$\mathbf{x} = \mathbf{u}G_n, \qquad G_n = B_m \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}^{\otimes m} \tag{4.4}$$

where the generating matrix, G_n, is the $m-$th Kronecker power of the Arıkan's 2×2 kernel (recall that $n = 2^m$) that is multiplied by the length-n bit-reversal matrix B_m and $\mathbf{u}$ denotes

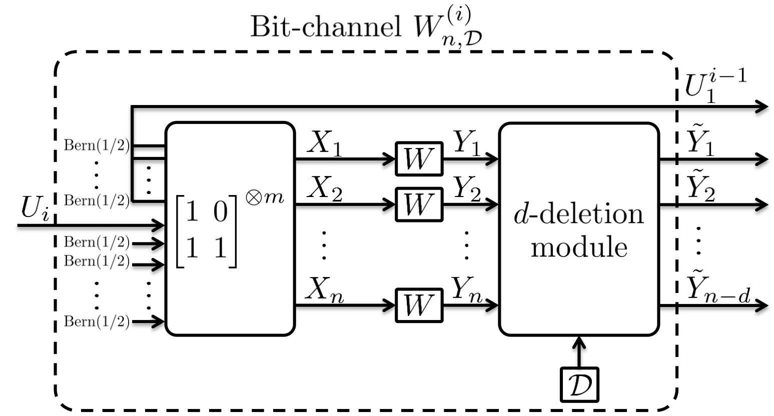


Figure 4.2: The i-th polar bit-channel with deletions induced from distribution $\mathcal{D}$.

the uncoded information vector that includes k information bits and $n-k$ frozen bits. Note that the linear mapping $u_1^n \to x_1^n$ is one to one. Hence, we can define a new channel $\tilde{W}_{n,\mathcal{D}}$ for simplicity with the transition probabilities given by

$$\tilde{W}_{n,\mathcal{D}}(\tilde{y}_1^{n-d}|u_1^n) \triangleq \tilde{W}^{n,\mathcal{D}}(\tilde{y}_1^{n-d}|u_1^n G_n). \tag{4.5}$$

The *successive cancellation decoding* (SC) of polar codes is based on decoding u_i for $i = 1, 2, \cdots, n$ sequentially while assuming the values of the previous u_i's. In other words, to decode u_i, we assume that the previous bits, $u_1, \cdots, u_{i-1}$, are all known (or correctly decoded) and hence available to the decoder similar to the channel observation vector after deletions $\tilde{\mathbf{y}}$. We also assume that the future bits are distributed uniformly at random, *i.e.* $U_j \sim$ Bern($\frac{1}{2}$) for $j = i+1, \cdots, n$. Let

$$W_{n,\mathcal{D}}^{(i)}(\tilde{y}_1^{n-d}, u_1^{i-1}|u_i) \triangleq \sum_{u_{i+1}^n \in \{0,1\}^{n-i}} \frac{1}{2^{n-1}} \tilde{W}_{n,\mathcal{D}}(\tilde{y}_1^{n-d}|u_1^n) \tag{4.6}$$

denote the $i-$th polar bit-channel in presence of $d-$deletions generated from distribution $\mathcal{D}$. Figure 4.2 depicts this definition. Here, the i-th polar bit-channel, $W_{n,\mathcal{D}}^{(i)}$, has input u_i, output $(\tilde{y}_1^{n-d}, u_1^{i-1})$, and deletion pattern distribution $\mathcal{D}$. The polar SC decoder works by calculating the likelihood parameters

$$h_{i,\mathcal{D}}(\tilde{y}_1^{n-d}, \hat{u}_1^{i-1}) \triangleq \frac{W_{n,\mathcal{D}}^{(i)}(\tilde{y}_1^{n-d}, \hat{u}_1^{i-1}|u_i = 0)}{W_{n,\mathcal{D}}^{(i)}(\tilde{y}_1^{n-d}, \hat{u}_1^{i-1}|u_i = 1)} \tag{4.7}$$

sequentially for $i = 1, 2, \cdots, n$, while making hard decisions on each u_i according to the following rule:

$$\hat{u}_i \triangleq \begin{cases} u_i & \text{if } u_i \text{ is a frozen bit} \\ 0 & \text{if } h_{i,\mathcal{D}}(\tilde{y}_1^{n-d}, \hat{u}_1^{i-1}) \geqslant 1 \\ 1 & \text{otherwise} \end{cases} . \tag{4.8}$$

In the next section, we explain how the conventional implementation of the polar SC decoder can be modified so that we can still calculate the likelihood parameters in (4.7) in presence of additional deletions with low computational complexity. We also postpone the discussion about location and values of the frozen bits to the last section.

4.2 Decoding Algorithms for Deletion Channels

In this section, we look into different decoding algorithms of polar codes in presence of deletions. While we assume that the reader is familiar with basic concepts of the polar SC decoder, we provide a quick overview of the recursive likelihood calculations, the decoding graph, and its computational complexity before going into the details of our modified decoding algorithm. [10] is a great reference to review these topics.

We begin by discussing two naive extensions of the SC decoding algorithm for channels with deletion. There are $\binom{n}{d}$ different deletion patterns. By fixing any such pattern, we can replace the removed symbols with erasure and generate a length$-n$ vector $\hat{y}_1^n \in (\mathcal{Y} \cup \{e\})^n$. It is possible to then simply feed the generated vector to the conventional polar SC decoder and estimate the uncoded information vector that corresponds to this specific deletion pattern.

Let us denote any such estimation by $\hat{u}_1^n$. The next step of the decoding algorithm would be to select the most likely deletion pattern based on the estimated $\hat{u}_1^n$'s. This can be realized in one of the two following ways:

- *Decoder A.* As first introduced in [107], one can precode the information bits with some *cyclic redundancy check* (CRC) which can be used to detect the correct estimated pattern. This is somehow similar to the application of CRC's in list decoding of polar codes [39], where CRC helps decoder to pick the correct codeword without comparing their likelihoods. While this approach usually results in a better performance than SC decoding alone, its computational complexity is of order $O(n^{d+1}\log n)$ for fixed values of d. In additional to its impracticality, this scheme also suffers from a rate loss caused by the added CRC. Note that for the CRC to be able to select one and only one of the $\binom{n}{d}$ candidates with high probability, it has to have at least $O(d\log n)$ redundancy bits.

- *Decoder B.* A secondary approach, in absence of CRCs, aims at finding the most-likely deletion pattern. To do so, the decoder needs to select one of the $\binom{n}{d}$ candidates for $\hat{u}_1^n$. However, it is not possible to compare the likelihoods of these estimated candidates by running the SC decoder for each deletion pattern separately since the likelihoods are conditioned on their corresponding deletion pattern. There is a way, although inefficient, to fix this problem. Note that polar SC decoder is not only efficient in estimating the information vector, but is also very efficient in calculating the likelihood of any desired information vector given the channel observations. Let us denote such conditional probabilities by $P(\mathbf{u}|\hat{\mathbf{y}})$. Here $\hat{\mathbf{y}}$ is constructed from $\tilde{\mathbf{y}}$ by inserting back d erasures that correspond to deleted locations. Let τ denote the number of different estimated $\hat{\mathbf{u}}$'s from $\binom{n}{d}$ deletion patterns. Since some deletion patterns may result in a same $\hat{\mathbf{u}}$, we have $\tau \leqslant \binom{n}{d}$. Now, one can run the SC decoder for $\tau\binom{n}{d}$ many times to calculate $P(\mathbf{u}|\hat{\mathbf{y}})$ for each such setup. The most-likely information vector is then revealed by comparing the following summations:

$$P(\mathbf{u}) = \mathbb{E}_{\hat{\mathbf{y}}} P(\mathbf{u}|\hat{\mathbf{y}}). \tag{4.9}$$

The decoding complexity in this case is given by $O(n^{2d+1} \log n)$, which is even worse than Decoder A.

Both decoders provide great recovery from the deletions. However, the immense increase in the decoding complexity forces us to design an alternative decoding algorithm. We also point out that neither of the two decoding method in above are correct implementations of the SC decoder in presence of deletions. The correct implementation of the SC decoder requires the decoding algorithm to sequentially perform ML decoding on bit-channels $W_{n,\mathcal{D}}^{(i)}$ for $i = 1, 2, \cdots, n$. A naive way to do so would be to setup $\binom{n}{d}$ *parallel* SC decoders blocks each with a different deletion pattern to start with. In other words, each SC decoder picks a specific deletion pattern and reconstructs the vector of received symbols denoted by $\hat{y}_1^n$ in which the deleted symbols are treated as erasures. Then, we start from $i = 1$ and go until $i = n$, where in each step all decoders are clocked simultaneously to calculate the conventional channel probabilities

$$W_n^{(i)}(\hat{y}_1^n, \hat{u}_1^{i-1}|u_i = 0) \quad \text{and} \quad W_n^{(i)}(\hat{y}_1^n, \hat{u}_1^{i-1}|u_i = 1), \tag{4.10}$$

and then a hard decision is made on u_i similar to (4.8) that is based on the value of

$$h_{i,\mathcal{D}}(\tilde{y}_1^{n-d}, \hat{u}_1^{i-1}) = \frac{\sum W_n^{(i)}(\hat{y}_1^n, \hat{u}_1^{i-1}|u_i = 0)}{\sum W_n^{(i)}(\hat{y}_1^n, \hat{u}_1^{i-1}|u_i = 1)}. \tag{4.11}$$

The summations are taken over all $\binom{n}{d}$ deletion patterns with the assumption that they are all equally likely. A slight modification is required if $\mathcal{D}$ is a nonuniform distribution, which we leave to the reader.

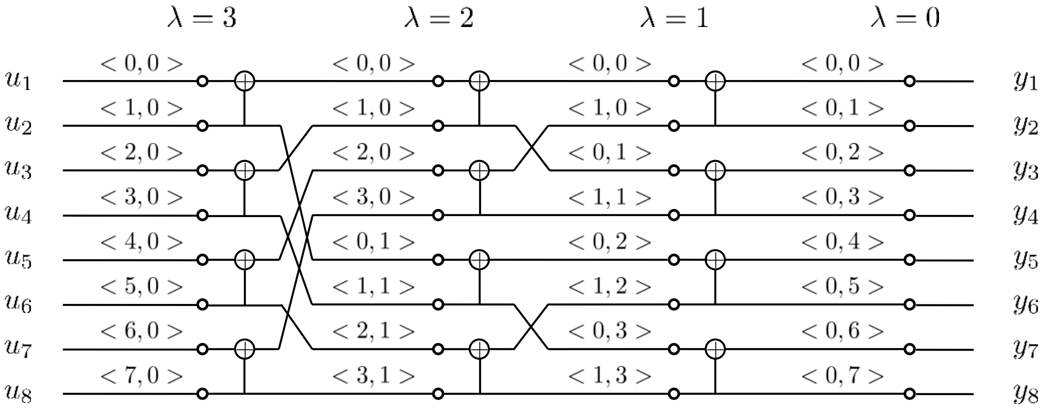


Figure 4.3: The polar encoding/decoding graph for $n = 8$.

It is now clear why we no longer depend on CRCs to make the final decision. However, the decoding complexity is still $O(n^{d+1} \log n)$. In the following, we propose an alternative implementation of the SC decoder, which only requires $O(d^2 n \log n)$ in computation complexity. Let us proceed by first reviewing some details about the successive cancellation decoding algorithm for polar codes.

The construction of the polar generating matrix (4.4) based allows its encoding circuit to be realized over a FFT-like graph that is sometimes referred to as the Tanner graph of polar codes, or more commonly as just the polar graph. The butterfly structure of this graph allows the encoding algorithm to be realized with $O(n \log n)$ computational complexity. Moreover, the very same graph can be used in decoder to successively estimate u_i's again with $O(n \log n)$ computational complexity. Figure 4.3 depicts the graph of a length-8 polar code. Here, all nodes are labeled with $\langle \varphi, \beta \rangle_\lambda$, where φ, β, and λ denote the phase, branch number, and the layer respectively. The bit reversal permutation is absorbed in between the layers. The polar graph has $m + 1$ layers that correspond to m steps of the polarization. Each layer also has n nodes in it. These nodes are labeled with $\langle \varphi, \beta \rangle$, where φ denotes their phase number and β denotes their branch number. We sometimes drop λ if its value is clear from the context. This notation is consistent with [39]. We refer the readers to this paper for further discussions about the motivation behind this type of labeling and how it helps with the efficient implementation of the polar codes in general. For nodes within layer λ we have

$$0 \leqslant \varphi < 2^{\lambda} \qquad \text{and} \qquad 0 \leqslant \beta < 2^{m-\lambda}. \tag{4.12}$$

There are also two data structures that help with the decoding algorithm. The first one stores a binary value $\in \{0, 1\}$ for each node in the graph that corresponds to their hard values and is denoted by B. The other one, denoted by P stores the probabilities (or the likelihoods) of each node being equal to 0 or 1 given the received vector $\mathbf{y}$. The structure of the polar graph allows one to efficiently calculate these values in a recursive fashion. We do not cover more details about the conventional successive cancellation decoder since we practically redefine them all for channels with deletion in the following.

The idea behind our modified SC decoding algorithm is to not fixing a deletion pattern from the beginning but to limiting our deletion pattern gradually during the recursive process of the SC decoder. Let us briefly explain the process in the first step of the SC decoding recursion, which also helps with understanding the pseudo codes that follow after.

The first step in the conventional SC decoding algorithm attempts at decoding the first information bit, u_1. To do so, it recalls itself asking for the evaluation of two intermediate bit-channels in layer $\lambda = m - 1$ that are labeled with $\langle \varphi, \beta \rangle = \langle 0, 0 \rangle$ and $\langle 0, 1 \rangle$. See nodes v_1 and v_2 on the polar graph depicted in Figure 4.4. These two bit-channels are independent and are looking at two disjoint sub-vectors of length-$\frac{n}{2}$ from $\hat{y}_i$'s as the output. However in presence of deletions, it is unclear which of the $\tilde{y}_i$ symbols should be mapped to the top half, and which belong to the bottom half. Note that, we do not need to distinguish between all different $\binom{n}{d}$ deletion patterns at this stage. Instead, we simply decide on the number of deletions that belong to the each half and postpone further calculations to the future steps. Let d_1 denote the number of deleted symbols from the first half of the y_i's. It is clear that

$$0 \leqslant d_1 \leqslant d. \tag{4.13}$$

So, we only require to make $d+1$ copies of nodes v_1 and v_2 at this stage, where each copy corresponds to a different subset of received symbols that belong to its output span. By doing so, we partition all mappings of the form $\tilde{\mathbf{y}} \to \hat{\mathbf{y}}$ into $d+1$ subsets. We chunk down these subsets in the next steps of the decoding algorithm.

Let us for example assume d is even and then look at the most-likely scenario, where $\tilde{y}_1^{(n-d)/2}$ is mapped to the top half and $\tilde{y}_{(n-d)/2+1}^{n-d}$ is mapped to the bottom half of the $\hat{y}_i$'s. Also, assume that the decoding pattern is generated according to the uniform distribution. Let $\mathcal{U}_d$ denote the uniform distribution of d deletions. The two corresponding intermediate bit-channels with inputs v_1, and v_2 are respectively defined as

$$W^{(1)}_{n/2,\mathcal{U}_{d/2}}(\tilde{y}_1^{(n\,-\,d)/2}|v_1), \qquad \text{and} \qquad W^{(2)}_{n/2,\mathcal{U}_{d/2}}(\tilde{y}_{(n\,-\,d)/2+1}^{n-d}|v_2). \tag{4.14}$$

The relation between the first two original bit-channel $W^{(1)}_{n,\mathcal{U}_d}$, $W^{(2)}_{n,\mathcal{U}_d}$, and these $d+1$ intermediate bit-channels in general is given by

$$\begin{aligned}
&W^{(1)}_{n,\mathcal{U}_d}(\tilde{y}_1^{n-d}|u_1)\\
&= \sum_{t=0}^{d} \Pr.\left[\tilde{y}_1^{n/2-t} \to \hat{y}_1^{n/2}\right]\left\{\frac{1}{2}\sum_{u_2} W^{(1)}_{n/2,\mathcal{U}_t}(\tilde{y}_1^{n/2-t}|u_1+u_2)W^{(2)}_{n/2,\mathcal{U}_{d-t}}(\tilde{y}_{n/2-t+1}^{n-d}|u_2)\right\} \qquad (4.15)\\
&= \frac{1}{\binom{n}{d}}\sum_{t=0}^{d}\left\{\binom{n/2}{t}\binom{n/2}{d-t} \times \frac{1}{2}\sum_{u_2} W^{(1)}_{n/2,\mathcal{U}_t}(\tilde{y}_1^{n/2-t}|u_1+u_2)W^{(2)}_{n/2,\mathcal{U}_{d-t}}(\tilde{y}_{n/2-t+1}^{n-d}|u_2)\right\},
\end{aligned}$$

and

$$\begin{aligned}
&W^{(2)}_{n,\mathcal{U}_d}(\tilde{y}_1^{n-d}, u_1|u_2)\\
&= \sum_{t=0}^{d} \Pr.\left[\tilde{y}_1^{n/2-t} \to \hat{y}_1^{n/2}\right]\left\{\frac{1}{2}W^{(1)}_{n/2,\mathcal{U}_t}(\tilde{y}_1^{n/2-t}|u_1+u_2)W^{(2)}_{n/2,\mathcal{U}_{d-t}}(\tilde{y}_{n/2-t+1}^{n-d}|u_2)\right\} \qquad (4.16)\\
&= \frac{1}{\binom{n}{d}}\sum_{t=0}^{d}\left\{\binom{n/2}{t}\binom{n/2}{d-t} \times \frac{1}{2}W^{(1)}_{n/2,\mathcal{U}_t}(\tilde{y}_1^{n/2-t}|u_1+u_2)W^{(2)}_{n/2,\mathcal{U}_{d-t}}(\tilde{y}_{n/2-t+1}^{n-d}|u_2)\right\},
\end{aligned}$$

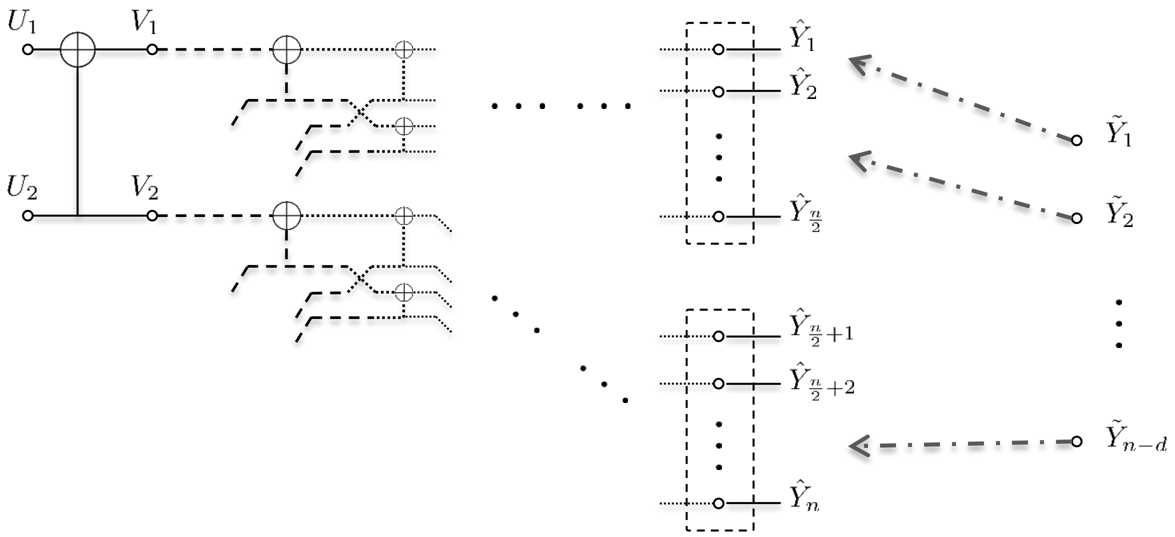


Figure 4.4: An instance of the SC recursion in the modified decoder.

where, Pr.$\left[\tilde{y}_1^{n/2-t} \to \hat{y}_1^{n/2}\right]$ denotes the probability that t out of d deletions appear in the first half of y_i's. Note that these $d+1$ scenarios are not equally likely. Hence, we combine their corresponding channel probabilities in a weighted fashion. Also, from this point on, we assume that the distribution of deleted symbols is always uniform and reduce the notation of bit-channels to $W_n^{(i)}$ unless otherwise is stated.

We recall that in the conventional successive cancellation decoding, the subsets of received symbols that correspond to the intermediate bit-channels always form a continuous sub-vector of y_1^n. Indeed, the output span of a node labeled by $\langle \varphi, \beta \rangle_\lambda$ is given by

$$\{y_{\beta 2^\lambda+1}, y_{\beta 2^\lambda+2}, \cdots, y_{(\beta+1)2^\lambda}\}, \tag{4.17}$$

which not surprisingly is independent of φ. Now, consider the scenario when d symbols are deleted and we are to map the remaining $\tilde{y}_1^{n-d}$ received symbols back to these n positions. The node $\langle \varphi, \beta \rangle_\lambda$ is only concerned about the part of the mapping that corresponds to its output; And, to determine that, it only requires to know the number deletions that occurred on or before $y_{\beta 2^\lambda}$ and the number of deleted symbols among $\{y_{\beta 2^\lambda+1}, y_{\beta 2^\lambda+2}, \cdots, y_{(\beta+1)2^\lambda}\}$. Let

d_0 and d_1 denote these numbers respectively. The desired order-preserving mapping can be expressed as

$$\hat{y}_{\beta 2^\lambda+1}, \hat{y}_{\beta 2^\lambda+2}, \cdots, \hat{y}_{(\beta+1)2^\lambda} \longleftarrow \tilde{y}_{\beta 2^\lambda - d_0+1}, \tilde{y}_{\beta 2^\lambda - d_0+2}, \cdots, \tilde{y}_{(\beta+1)2^\lambda - d_0 - d_1}. \tag{4.18}$$

Let us for simplicity define

$$\tilde{y}_1^{n-d} \setminus \langle d_0, d_1, \beta, \lambda \rangle \triangleq \tilde{y}_{\beta 2^\lambda - d_0+1}, \tilde{y}_{\beta 2^\lambda - d_0+2}, \cdots, \tilde{y}_{(\beta+1)2^\lambda - d_0 - d_1}, \tag{4.19}$$

which allows us to rewrite the layer-m bit-channels in (4.6) as $W_n^{(i)}(y_1^{n-d} \setminus \langle 0, d, 0, m \rangle, u_1^{i-1} | u_i)$.

Theorem 13.*Assume* $\Lambda = 2^\lambda$ *and* $0 < 2\psi+1 < \Lambda$*. We can rephrase the recursive construction of the layer-*λ *polar bit-channels based on SC cancellation decoder in presence of deletions as the following [see (22), (23) in [10] or (4), (5) in [39] for the original formulation]:*

$$\begin{aligned}
&\overbrace{W_{\Lambda}^{(2\psi+1)}(\tilde{y}_1^{n-d} \setminus \langle d_0, d_1, \beta, \lambda \rangle, u_1^{2\psi} | u_{2\psi+1})}^{\textit{branch } \beta} = \frac{1}{\binom{\Lambda}{d_1}} \sum_{t=0}^{d_1} \Bigg\{ \binom{\Lambda/2}{t} \binom{\Lambda/2}{d_1 - t} \times \\
&\frac{1}{2} \sum_{u_{2\psi+2}} \underbrace{W_{\Lambda/2}^{(\psi+1)}(\tilde{y}_1^{n-d} \setminus \langle d_0, t, 2\beta, \lambda-1 \rangle, u_{1,\textit{even}}^{2\psi} \oplus u_{1,\textit{odd}}^{2\psi} | u_{2\psi+1} + u_{2\psi+2})}_{\textit{branch } 2\beta} \\
&\qquad\qquad \cdot \underbrace{W_{\Lambda/2}^{(\psi+1)}(\tilde{y}_1^{n-d} \setminus \langle d_0 + t, d_1 - t, 2\beta+1, \lambda-1 \rangle, u_{1,\textit{even}}^{2\psi} | u_{2\psi+2})}_{\textit{branch } 2\beta+1} \Bigg\},
\end{aligned} \tag{4.20}$$

and

$$\begin{aligned}
&\overbrace{W_{\Lambda}^{(2\psi+2)}(\tilde{y}_1^{n-d} \setminus \langle d_0, d_1, \beta, \lambda \rangle, u_1^{2\psi+1} | u_{2\psi+2})}^{\textit{branch } \beta} = \frac{1}{\binom{\Lambda}{d_1}} \sum_{t=0}^{d_1} \Bigg\{ \binom{\Lambda/2}{t} \binom{\Lambda/2}{d_1 - t} \times \\
&\frac{1}{2} \underbrace{W_{\Lambda/2}^{(\psi+1)}(\tilde{y}_1^{n-d} \setminus \langle d_0, t, 2\beta, \lambda-1 \rangle, u_{1,\textit{even}}^{2\psi} \oplus u_{1,\textit{odd}}^{2\psi} | u_{2\psi+1} + u_{2\psi+2})}_{\textit{branch } 2\beta} \\
&\qquad\qquad \cdot \underbrace{W_{\Lambda/2}^{(\psi+1)}(\tilde{y}_1^{n-d} \setminus \langle d_0 + t, d_1 - t, 2\beta+1, \lambda-1 \rangle, u_{1,\textit{even}}^{2\psi} | u_{2\psi+2})}_{\textit{branch } 2\beta+1} \Bigg\}.
\end{aligned} \tag{4.21}$$

Proof. We only discuss the first formulation since the other one follows from a similar argument. The output span of the corresponding bit-channel, that is the subsequence of $\tilde{y}_1^{n-d}$, is given by

$$\tilde{y}_{\beta\Lambda-d_0+1}, \tilde{y}_{\beta\Lambda-d_0+2}, \cdots, \tilde{y}_{(\beta+1)\Lambda-d_0-d_1}.$$

Now, we have to cut this sequence in two, where the first tail will form the output span of a bit-channel whose phase and branch numbers are given by $\langle\psi+1, 2\beta\rangle_{\lambda-1}$ and the other will form the output span of another bit-channel with phase and branch numbers given by $\langle\psi+1, 2\beta+1\rangle_{\lambda-1}$. To do so, we have to determine how many of the d_1 deleted symbols belonged to the top half and how many belonged to the other one. The length of the output span prior to deletions is given by $\Lambda = 2^\lambda$. Furthermore, all $\binom{\Lambda}{d_1}$ deletion patterns are equally likely. So, the conditional probability of observing t deletions in the first half is given by

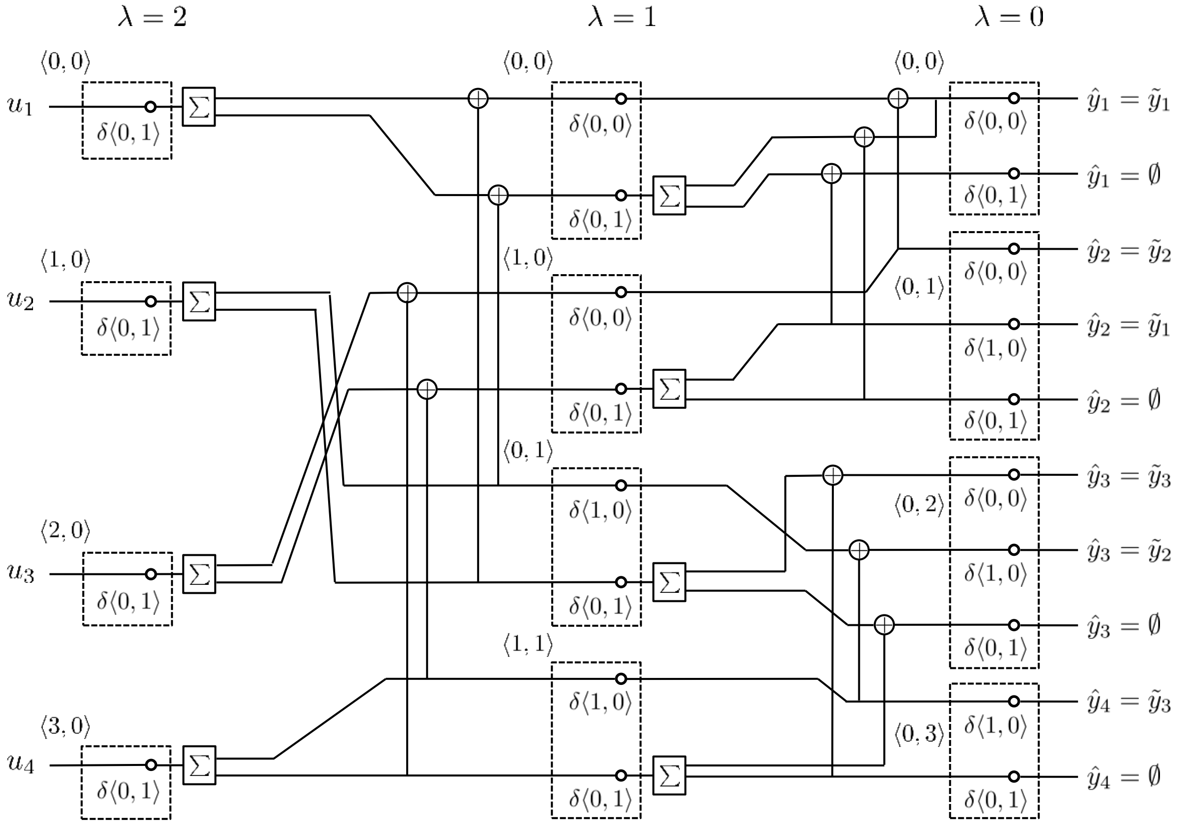
$$\begin{aligned}\text{Pr.}[t \text{ symbols deleted among } y_{\beta\Lambda+1}^{\beta 2^\lambda+2^{\lambda-1}} \mid\ d_1 \text{ symbols deleted among } y_{\beta\Lambda+1}^{(\beta+1)\Lambda}]\\ = \frac{\binom{\Lambda/2}{t}\binom{\Lambda/2}{d_1-t}}{\binom{\Lambda}{d_1}}.\end{aligned} \tag{4.22}$$

Note that a simple double-counting argument yields in

$$\sum_{t=0}^{d_1}\binom{\Lambda/2}{t}\binom{\Lambda/2}{d_1-t} = \binom{\Lambda}{t}, \tag{4.23}$$

which helps with understanding of the weighted averages in (4.20), (4.21) while not necessary for the proof. The rest of the proof follows similar to the proof of the original formulation that appeared in [10]. □

It is now clear that the original nodes in the polar graph should be replaced with multiple copies for the decoding purposes, where each replacement addresses a different set of mappings from the the received symbols to its corresponding output span. In particular, any original node such as $\langle\varphi, \beta\rangle_\lambda$ should be replaced with a group of $\tau_d(\beta, \lambda)$ new nodes, where



$\lambda = 2$
$\lambda = 1$
$\lambda = 0$
$\langle 0, 0\rangle$
u_1
$\delta\langle 0, 1\rangle$
$\langle 1, 0\rangle$
u_2
$\delta\langle 0, 1\rangle$
$\langle 2, 0\rangle$
u_3
$\delta\langle 0, 1\rangle$
$\langle 3, 0\rangle$
u_4
$\delta\langle 0, 1\rangle$
$\langle 0, 0\rangle$
$\delta\langle 0, 0\rangle$
$\delta\langle 0, 1\rangle$
$\langle 1, 0\rangle$
$\delta\langle 0, 0\rangle$
$\delta\langle 0, 1\rangle$
$\langle 0, 1\rangle$
$\delta\langle 1, 0\rangle$
$\delta\langle 0, 1\rangle$
$\langle 1, 1\rangle$
$\delta\langle 1, 0\rangle$
$\delta\langle 0, 1\rangle$
$\langle 0, 0\rangle$
$\delta\langle 0, 0\rangle$
$\delta\langle 0, 1\rangle$
$\langle 0, 1\rangle$
$\delta\langle 0, 0\rangle$
$\delta\langle 1, 0\rangle$
$\delta\langle 0, 1\rangle$
$\langle 0, 2\rangle$
$\delta\langle 0, 0\rangle$
$\delta\langle 1, 0\rangle$
$\delta\langle 0, 1\rangle$
$\langle 0, 3\rangle$
$\delta\langle 1, 0\rangle$
$\delta\langle 0, 1\rangle$
Σ
$\hat{y}_1 = \tilde{y}_1$
$\hat{y}_1 = \emptyset$
$\hat{y}_2 = \tilde{y}_2$
$\hat{y}_2 = \tilde{y}_1$
$\hat{y}_2 = \emptyset$
$\hat{y}_3 = \tilde{y}_3$
$\hat{y}_3 = \tilde{y}_2$
$\hat{y}_3 = \emptyset$
$\hat{y}_4 = \tilde{y}_3$
$\hat{y}_4 = \emptyset$

Figure 4.5: The modified decoding graph of a length-4 polar code with $d = 1$ deleted symbol.

$\tau_d(\beta, \lambda)$ denotes the number of integer solutions to

$$\begin{cases} d - (n - (\beta + 1)2^\lambda) \leqslant d_0 + d_1 \leqslant d \\ \quad 0 \quad \leqslant \quad d_0 \quad \leqslant \beta 2^\lambda \\ \quad 0 \quad \leqslant \quad d_1 \quad \leqslant 2^\lambda \end{cases} . \tag{4.24}$$

Figure 4.5 depicts the modified decoding graph for a polar code of length 4 in presence of 1 deletion error. A node that was originally labeled with $\langle \varphi, \beta \rangle$ is now replaced with a group of nodes each labeled with $\delta\langle d_0, d_1 \rangle$ internally, where d_0 and d_1 denote the number of deletions prior and inside the output span of the original $\langle \varphi, \beta \rangle_\lambda$ node. The $\sum$ symbol corresponds to the calculation of weighted averages based on the conditional probabilities in (4.22). The number of new nodes within these decoding groups can be upper bounded by

$$\tau_d(\beta, \lambda) \leqslant \binom{d+2}{2}, \tag{4.25}$$

which clearly is not a tight bound for many choices of β and λ. However, it gives us the following result on the new overall computational complexity that is based on the extended decoding graph.

Corollary 4.1. *The computational complexity of the modified polar SC decoding algorithm for noisy channels with* d *deleted symbols is asymptotically upper bounded by* $O(d^2 n \log n)$*, where* $n = 2^m$ *denotes the length of the polar code. The number of deletions can be an arbitrary function of* n*.*

Next, we go over the details of modified SC decoder. We provide some high-level pseudo-codes that are consistent with [39] in notation. Parts of these pseudo-codes are exact replicas of the space inefficient implementation of SC decoder in [39], which are reintroduced here for the sake of completeness.

For each layer λ, we require two data structures, namely B_λ and P_λ, where B_λ stores the hard values of the nodes in the extended decoding graph and P_λ stores the corresponding probability pairs (or likelihoods). Nodes in the graph are labeled with $\langle \varphi, \beta \rangle_\lambda | \langle d_0, d_1 \rangle$. Here, φ and β denotes the phase and branch number of the node. $\langle d_0, d_1 \rangle$ also uniquely represents the available symbols in the output span of this node, which can also be viewed as the state of this bit-channel. We may also drop λ if its value is clear within the context.

The probability array data structure P_λ is used as follows. Let a layer $0 \leqslant \lambda \leqslant m$, phase $0 \leqslant \varphi < \Lambda = 2^\lambda$, the branch number $0 \leqslant \beta < 2^{m-\lambda}$, and the output state $\langle d_0, d_1 \rangle$ be given, where d_0 and d_1 satisfy the conditions in (4.24). Denote the output corresponding to branch β of $W_\Lambda^{(\varphi+1)}$ with state $\langle d_0, d_1 \rangle$ as $(\hat{y}_1^{\Lambda-d_1}, u_1^\varphi)$. Then, upon ending the SC decoding algorithm, we will have for both values of $b = 0, 1$ that

$$P_\lambda[\langle\varphi, \beta\rangle|\langle d_0, d_1\rangle][b] = W_\Lambda^{(\varphi+1)}(\hat{y}_1^{\Lambda-d_1}, u_1^\varphi|b). \tag{4.26}$$

To introduce the bit array data structure B_λ, we first point out that a same hard decision applies to all nodes with the same φ and β. Similar to before, let layer $0 \leqslant \lambda \leqslant m$, phase $0 \leqslant \varphi < \Lambda = 2^\lambda$, and the branch number $0 \leqslant \beta < 2^{m-\lambda}$ be given. We also denote the input corresponding to branch β of all bit-channels $W_\Lambda^{(\varphi+1)}$ (regardless of their output state) by $\hat{u}(\lambda, \varphi, \beta)$. Then, this data structure is used to ultimately store

$$B_\lambda[\langle\varphi, \beta\rangle] = \hat{u}(\lambda, \varphi, \beta). \tag{4.27}$$

Algorithm 3 illustrates the high-level description of the modified SC decoder for polar codes when d out of n transmitted symbols are deleted uniformly at random. The main difference between Algorithm 3 and its original version is in the initialization of the probability array. Given any $0 \leqslant \beta < n$, there are $\min\{d, \beta\} + 2$ different sets of mappings from the received symbols to the output of the node $\langle 0, \beta\rangle_0$. $d_1 = 1$ determines one set of such mapping, in which the output is nothing but the erasure (deleted symbol). The other $\min\{d, \beta\} + 1$ output spans also correspond to the case where $d_1 = 0$ but d_0 varies from 0 to its maximum value. This initializations are described in lines (1)-(5) of the algorithm. We recommend the reader to follow these initializations in the example provided in Figure 4.5.

The structure of the decoding graph allows us to fill out our data structures recursively and efficiently. This is realized in the main loop of the Algorithm 3, where functions *recursivelyCalcP* and *recursivelyUpdateB* are called. A high level description of these functions is given in Algorithms 4 and 5. Note that Algorithm 5 is the same for channels with or without deletion, while Algorithm 4 differs from its original form in [39] in the calculations for the update rule. The combinatorial coefficients used in (4.20) and (4.21) can be calculated in advance to avoid duplicate calculations.

Theorem 14. *Algorithm 3, 4, and 5 are a valid implementation of the polar SC decoder in for channel with deletion defined in (4.5) and (4.6).*

Proof. Proof follows similar to that of Lemma 2 in [39]. □

Algorithm 3 A high-level description of SC decoding with d random deletions

```
Input: received vector $\tilde{y}_1^{n-d}$
Output: a decoded codeword $\hat{\mathbf{c}}$
for $\beta = 0, 1, \ldots, n-1$ do                                    // Initialization
    for $d_0 = 0, 1, \ldots, \min\{d, \beta\}$ do
        $P_0[\langle 0, \beta\rangle | \langle d_0, 0\rangle][0] \leftarrow W(\tilde{y}_{\beta+1-d_0}|0)$,    $P_0[\langle 0, \beta\rangle | \langle d_0, 0\rangle][1] \leftarrow W(\tilde{y}_{\beta+1-d_0}|1)$
    for $d_0 = 0, 1, \ldots, \min\{d-1, \beta\}$ do
        $P_0[\langle 0, \beta\rangle | \langle d_0, 1\rangle][0] \leftarrow 1/2$,    $P_0[\langle 0, \beta\rangle | \langle d_0, 1\rangle][1] \leftarrow 1/2$
for $\varphi = 0, 1, \ldots, n-1$ do                                   // Main loop
    recursivelyCalcP($m, \varphi$)
    if $u_{\varphi+1}$ is frozen then
        set $B_m[\langle \varphi, 0\rangle]$ to the frozen value of $u_{\varphi+1}$
    else
        if $P_m[\langle \varphi, 0\rangle | \langle 0, d\rangle][0] > P_m[\langle \varphi, 0\rangle | \langle 0, d\rangle][1]$ then
            set $B_m[\langle \varphi, 0\rangle] \leftarrow 0$
        else
            set $B_m[\langle \varphi, 0\rangle] \leftarrow 1$
    if $\varphi \mod 2 = 1$ then
        recursivelyUpdateB($m, \varphi$)
return the decoded codeword: $\hat{\mathbf{c}} = (B_0[\langle 0, \beta\rangle])_{\beta=0}^{n-1}$
```

Algorithm 4 An implementation of recursivelyCalcP(λ, φ) for channels with deletion

Input: layer λ and phase φ

```
if $\lambda = 0$ then return                                    // Stopping condition
Set $\psi \leftarrow \lfloor \varphi/2 \rfloor$ and $\Lambda \leftarrow 2^{\lambda}$                      // Recurse first, if needed
if $\varphi \mod 2 = 0$ then recursivelyCalcP($\lambda - 1, \psi$)
for $\beta = 0, 1, \ldots, 2^{m-\lambda} - 1$ do                                 // calculation
    for $\forall d_0, d_1$ that satisfy conditions in (4.24) do
        if $\varphi \mod 2 = 0$ then                           // apply Equation (4.20)
            for $u' \in \{0, 1\}$ do
                $P_{\lambda}[\langle \varphi, \beta \rangle | \langle d_0, d_1 \rangle][u'] \leftarrow \frac{1}{2} \sum_{t=0}^{d_1} \sum_{u''} \binom{\Lambda/2}{t} \binom{\Lambda/2}{d_1 - t} \binom{\Lambda}{d_1}^{-1} \cdot$
                    $P_{\lambda-1}[\langle \psi, 2\beta \rangle | \langle d_0, t \rangle][u' \oplus u''] \cdot P_{\lambda-1}[\langle \psi, 2\beta + 1 \rangle | \langle d_0 + t, d_1 - t \rangle][u'']$
        else                                                   // apply Equation (4.21)
            set $u' \leftarrow B_{\lambda}[\varphi - 1, \beta]$  for $u'' \in \{0, 1\}$ do
                $P_{\lambda}[\langle \varphi, \beta \rangle | \langle d_0, d_1 \rangle][u''] \leftarrow \frac{1}{2} \sum_{t=0}^{d_1} \binom{\Lambda/2}{t} \binom{\Lambda/2}{d_1 - t} \binom{\Lambda}{d_1}^{-1} \cdot$
                    $P_{\lambda-1}[\langle \psi, 2\beta \rangle | \langle d_0, t \rangle][u' \oplus u''] \cdot$  $P_{\lambda-1}[\langle \psi, 2\beta + 1 \rangle | \langle d_0 + t, d_1 - t \rangle][u'']$
```

Algorithm 5 An implementation of recursivelyUpdateB(λ, φ) as appears in [39]

Require: φ is odd

```
set $\psi \leftarrow \lfloor \varphi/2 \rfloor$
for $\beta = 0, 1, \ldots, 2^{m-\lambda} - 1$ do
    $B_{\lambda-1}[\langle \psi, 2\beta \rangle] \leftarrow B_{\lambda}[\langle \varphi - 1, \beta \rangle] \oplus B_{\lambda}[\langle \varphi, \beta \rangle]$
    $B_{\lambda-1}[\langle \psi, 2\beta + 1 \rangle] \leftarrow B_{\lambda}[\langle \varphi, \beta \rangle]$
if $\psi \mod 2 = 1$ then
    recursivelyUpdateB($\lambda - 1, \psi$)
```

4.3 Channel Polarization Theorems for Deletion Channels

In this section we discuss the polarization theorems for noisy channels with deletion. We begin by outlining the proof for weak polarization theorem when $d = o(n)$. It is followed by the strong polarization theorem for fixed values of d. Many parts of the proofs, in particular those that are very similar to the existing theorems in literature, are omitted to save in space. We provide detailed references for reader who prefer to track them down.

Theorem 15. [Weak Polarization] *Define the noisy d-deletion channel as depicted in Figure 4.1-b. Assume that $d = o(n)$. Also assume that a rate-1 polar encoder is in place, which related U_i's and X_i's according to (4.4). Here, U_i' are assumed to be i.i.d. random variables with uniform distribution over $\{0, 1\}$. Also let X, Y denote the input and output of a single copy of the middle noisy channel W, where X is uniformly distributed over $\{0, 1\}$. Then for any $0 < \epsilon < 1$, we have*

$$\lim_{n\to\infty} \frac{1}{n}|i : H(U_i|U_1^{i-1}\tilde{Y}_1^{n-d}) > 1 - \epsilon| \geqslant H(X|Y), \tag{4.28}$$

$$\lim_{n\to\infty} \frac{1}{n}|i : H(U_i|U_1^{i-1}\tilde{Y}_1^{n-d}) < \epsilon| = 1 - H(X|Y). \tag{4.29}$$

Proof. Note that (4.28) immediately follows from the channel polarization theorems for B-DMCs since for all i

$$H(U_i|U_1^{i-1}\tilde{Y}_1^{n-d}) > H(U_i|U_1^{i-1}Y_1^{n}). \tag{4.30}$$

The second claim also follows from the mutual information chain rule and the following lemma that shows the total capacity preserving property of the channel when $d = o(n)$ and n is sufficiently large. While the proof is removed due to lack of space, we point out that this is practically a special case of [108, Theorem 5.1].

Lemma 4.1. *Let the noisy d-deletion channel and its related parameters to be defined similar to Theorem 15. The symmetric information rate is then lower bounded by*

$$\frac{I(U_1^n;\tilde{Y}_1^{n-d})}{n} \geqslant 1 - h_2\Big(\frac{n-d}{n}\Big) - \frac{n-d}{n}H(X|Y). \tag{4.31}$$

□

Next, we proceed by stating the strong polarization theorem, and then outlining its proof in three main steps.

Theorem 16.[Strong Polarization] *Assume d is a fixed number. Let Z_m denote a random variable that takes values from $Z\big(W_{n,\mathcal{U}_d}^{(i)}(\tilde{\mathbf{Y}}, U_1^{i-1}|U_i)\big)$ for $i = 1, \cdots, n$ with equal probabilities. Note that $m = \log n$. Also let X, Y denote the input and output of a single copy of the middle noisy channel W, where X is uniformly distributed over $\{0,1\}$. For any $\beta < 1/2$, we have*

$$\lim_{n\to\infty} P(Z_m \leqslant 2^{-2^{\beta m}}) = 1 - H(X|Y). \tag{4.32}$$

Proof. At first, we define auxiliary channels $W_n^{*(i)}$ which differ with our previously defined bit-channels only in their deletion distribution pattern:

$$\begin{aligned} \mathcal{F}_{i,d} &\triangleq \ \text{argmax}_{\mathcal{D}_d}\ Z\big(W_{n,\mathcal{D}_d}^{(i)}(\tilde{Y}_1^{n-d}, U_1^{i-1}|U_i)\big), \\ W_n^{*(i)}(\tilde{Y}_1^{n-d}, U_1^{i-1}|U_i) &\triangleq W_{n,\mathcal{F}_{i,d}}^{(i)}(\tilde{Y}_1^{n-d}, U_1^{i-1}|U_i). \end{aligned} \tag{4.33}$$

Also define Z_m^* to be random variable that takes values from $Z\big(W_n^{*(i)}\big)$ for $i = 1, \cdots, n$ uniformly. It satisfies to prove

$$\lim_{n\to\infty} P(Z_m^* \leqslant 2^{-2^{\beta m}}) = 1 - H(X|Y). \tag{4.34}$$

To do so, we apply the following lemma which is a special case of [109, Lemma 4.2]. The proof is long but exactly the same as appears in [109], and hence removed to avoid duplication.

Lemma 4.2. *Let $B_1, B_2, \cdots$ be an i.i.d. process where B_i is uniformly distributed over $\{1, 2\}$. Also let $Z_0^*, Z_1^*, \cdots$ be a $[0, 1]$-valued random process such that*

$$\forall m > m_0 : \qquad Z_m^* \leqslant \begin{cases} 2Z_{m-1}^*, & \textit{if } B_m = 1 \\ Z_{m-1}^{*2}, & \textit{if } B_m = 2 \end{cases}, \tag{4.35}$$

where m_0 is a fixed non-negative integer. Suppose also Z_m^* converges almost surely to a $\{0, 1\}$-valued random variable Z_∞^* with $P(Z_\infty^* = 0) = \alpha$. Then, for any $\beta < 1/2$, we have

$$\lim_{m \to \infty} P(Z_m^* \leqslant 2^{-2^{\beta m}}) = \alpha. \tag{4.36}$$

Now we recall that

$$Z(W^{(i)}) \leqslant \sqrt{1 - (1 - H(W^{(i)}))^2}, \tag{4.37}$$

where $H(W^{(i)})$ is equal to the entropy of the input of $W^{(i)}$ given its output when we assume uniform distribution on the inputs. Therefore, $H(W^{(i)}) \sim 0$ is equivalent to $Z(W^{(i)}) \sim 0$. Combining this fact with (4.28), and (4.29) results in $P(Z_\infty = 0) = 1 - H(X|Y)$, where Z_∞ is the limit of the random process Z_m. We state, without proof, that the same argument is also applicable to Z_∞^*, which would satisfy the second condition of Lemma 4.2.

Lastly, we have to establish the Bhattacharyya inequalities through recursive structure of the polar graph as required in (4.35). Let us for simplicity look at first instance of this recursion that is depicted in Figure 4.4. Let $\eta = \frac{n}{2}$ and define a new auxiliary channel $W_n^{\Delta(1)}$ that is formed by erasing the d middle output symbols of $\tilde{Y}_1^{n-d}$ from $W_n^{*(1)}$, *i.e.*

$$W_n^{\Delta(1)}(\tilde{y}_1^{\eta-d}, \tilde{y}_{\eta+1}^{n-d}|u_1) \triangleq \sum_{\tilde{y}_{\eta-d+1}^{\eta} \in \mathcal{Y}^d} W_n^{*(1)}(\tilde{y}_1^{n-d}|u_1). \tag{4.38}$$

It is clear that $W_n^{\Delta(1)}$ is statistically degraded with respect to the $W_n^{*(i)}$. Hence,

$$Z(W_n^{*(1)}) \leqslant Z(W_n^{\Delta(1)}). \tag{4.39}$$

Note that in Figure 4.4 we always map $\tilde{y}_1^{\eta-d}$ to the top half of $\hat{y}_1^{\eta}$, while $\tilde{y}_{\eta+1}^{n-d}$ are always mapped to the bottom half of $\hat{y}_{\eta+1}^{n}$. Therefore, $\tilde{y}_1^{\eta-d}$ only belongs to the output of the intermediate bit-channel observed from V_1; and, $\tilde{y}_{\eta+1}^{n-d}$ only appears in the output of the other bit-channel that is observed from V_2. The Bhattacharyya parameter of $W_n^{\Delta(1)}$ can then be formulated as

$$\begin{aligned}
&Z\big(W_n^{\Delta(1)}(\tilde{y}_1^{\eta-d}, \tilde{y}_{\eta+1}^{n-d}|u_1)\big) = \\
&\sum_{\tilde{y}_1^{\eta-d}, \tilde{y}_{\eta+1}^{n-d}} \sqrt{W_n^{\Delta(1)}(\tilde{y}_1^{\eta-d}, \tilde{y}_{\eta+1}^{n-d}|u_1=0)} \times \sqrt{W_n^{\Delta(1)}(\tilde{y}_1^{\eta-d}, \tilde{y}_{\eta+1}^{n-d}|u_1=1)},
\end{aligned} \tag{4.40}$$

which is equal to

$$\begin{aligned}
\frac{1}{2} \sum_{\tilde{y}_1^{\eta-d}, \tilde{y}_{\eta+1}^{n-d}} \Bigg(&\big(W_{\eta,\mathcal{D}_1}^{(1)}(\tilde{y}_1^{\eta-d}|v_1=0) W_{\eta,\mathcal{D}_2}^{(2)}(\tilde{y}_{\eta}^{n-d}|v_2=0) \\
&+ W_{\eta,\mathcal{D}_1}^{(1)}(\tilde{y}_1^{\eta-d}|v_1=1) W_{\eta,\mathcal{D}_2}^{(2)}(\tilde{y}_{\eta}^{n-d}|v_2=1)\big) \\
&\times \big(W_{\eta,\mathcal{D}_1}^{(1)}(\tilde{y}_1^{\eta-d}|v_1=0) W_{\eta,\mathcal{D}_2}^{(2)}(\tilde{y}_{\eta}^{n-d}|v_2=1) \\
&+ W_{\eta,\mathcal{D}_1}^{(1)}(\tilde{y}_1^{\eta-d}|v_1=1) W_{\eta,\mathcal{D}_2}^{(2)}(\tilde{y}_{\eta}^{n-d}|v_2=0)\big)\Bigg)^{1/2},
\end{aligned} \tag{4.41}$$

where $\mathcal{D}_1$ and $\mathcal{D}_2$ are the resulting distributions of length-η from erasing the middle d symbols in $\tilde{y}_1^{n-d}$. Note that these distributions are not necessary uniform even if the initial length-n distribution was uniform, which is the main reason behind definition of the *worst* distribution in (4.33). We replace $\mathcal{D}_1, \mathcal{D}_2$ with the worst distribution as well to arrive at

$$\begin{aligned}
&Z\big(W_n^{\Delta(1)}(\tilde{y}_1^{\eta-d}, \tilde{y}_{\eta+1}^{n-d}|u_1)\big) \\
&\leqslant \frac{1}{2} \sum_{\tilde{y}_1^{\eta-d}, \tilde{y}_{\eta+1}^{n-d}} \Bigg(\big(W_{\eta}^{*(1)}(\tilde{y}_1^{\eta-d}|v_1=0) W_{\eta}^{*(2)}(\tilde{y}_{\eta}^{n-d}|v_2=0) \\
&\qquad + W_{\eta}^{*(1)}(\tilde{y}_1^{\eta-d}|v_1=1) W_{\eta}^{*(2)}(\tilde{y}_{\eta}^{n-d}|v_2=1)\big) \\
&\qquad \times \big(W_{\eta}^{*(1)}(\tilde{y}_1^{\eta-d}|v_1=0) W_{\eta}^{*(2)}(\tilde{y}_{\eta}^{n-d}|v_2=1) \\
&\qquad + W_{\eta}^{*(1)}(\tilde{y}_1^{\eta-d}|v_1=1) W_{\eta}^{*(2)}(\tilde{y}_{\eta}^{n-d}|v_2=0)\big)\Bigg)^{1/2}.
\end{aligned} \tag{4.42}$$

Next, we apply then Jensen's inequality and proceed to

$$\begin{aligned}
&Z\big(W_n^{\Delta(1)}(\tilde{y}_1^{\eta-d}, \tilde{y}_{\eta+1}^{n-d}|u_1)\big) \\
&\leqslant \sum_{\tilde{y}_1^{\eta-d}, \tilde{y}_{\eta+1}^{n-d}} \sqrt{W_\eta^{*(1)}(\tilde{y}_1^{\eta-d}|v_1=0) W_\eta^{*(1)}(\tilde{y}_1^{\eta-d}|v_1=1)} \\
&\qquad\qquad + \sqrt{W_\eta^{*(2)}(\tilde{y}_\eta^{n-d}|v_2=0) W_\eta^{*(2)}(\tilde{y}_\eta^{n-d}|v_2=1)} \\
&= Z\big(W_\eta^{*(1)}(\tilde{y}_1^{\eta-d}|v_1)\big) + Z\big(W_\eta^{*(2)}(\tilde{y}_\eta^{n-d}|v_2)\big) = 2Z_{m-1}^{*(1)}.
\end{aligned} \tag{4.43}$$

The proof of the other inequality in (4.35) that corresponds to $B_m = 2$ also follows the same steps through expanding the recursive formulation of second bit-channel. Hence, it is removed to avoid duplication. □

We finish this section by pointing out that the strong polarization proof can be extended to $d = o(n)$ as well. However, it requires further modifications in theorems and lemmas that we simply recalled from existing literature.

Chapter 4, in part, contains materials from

- K. Tian, A. Fazeli, and A. Vardy, "Polar coding for channels with deletion," submitted to *IEEE Transactions on Information Theory*.

The dissertation author was the primary investigator and author of this paper.

Bibliography

[1] C. E. Shannon, “A mathematical theory of communication,” *ACM SIGMOBILE Mobile Computing and Communications Review*, vol. 5, no. 1, pp. 3–55, 2001.

[2] C. B. Schlegel and L. C. Perez, *Trellis and turbo coding: iterative and graph-based error control coding*. John Wiley & Sons, 2015.

[3] M. Mondelli, S. H. Hassani, and R. L. Urbanke, “Unified scaling of polar codes: Error exponent, scaling exponent, moderate deviations, and error floors,” *IEEE Transactions on Information Theory*, vol. 62, no. 12, pp. 6698–6712, 2016.

[4] A. Fazeli and A. Vardy, “On the scaling exponent of binary polarization kernels,” in *Proceedings of 52nd Annual Allerton Conference on Communication, Control, and Computing*, 2014, pp. 797–804.

[5] S. Buzaglo, A. Fazeli, P. H. Siegel, V. Taranalli, and A. Vardy, “On efficient decoding of polar codes with large kernels,” in *Proceedings of IEEE Wireless Communications and Networking Conference Workshops (WCNCW)*, 2017, pp. 1–6.

[6] A. Fazeli, S. H. Hassani, M. Mondelli, and A. Vardy, “Binary linear codes with optimal scaling and quasi-linear complexity,” *arXiv preprint arXiv:1711.01339*, 2017.

[7] S. Buzaglo, A. Fazeli, P. H. Siegel, V. Taranalli, and A. Vardy, “Permuted successive cancellation decoding for polar codes,” in *In Proceedings of IEEE International Symposium on Information Theory (ISIT)*, 2017, pp. 2618–2622.

[8] A. Fazeli, K. Tian, and A. Vardy, “Viterbi-aided successive-cancellation decoding of polar codes,” in *Proceedings of IEEE Global Communications Conference (GLOBECOM)*, 2017, pp. 1–6.

[9] Y. Polyanskiy, “SPECTRE: Short packet communication toolbox,” *GitHub Reposi-*

tory, available online https://github.com/yp-mit/spectre, 2018.

[10] E. Arıkan, “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.

[11] C. Leroux, A. J. Raymond, G. Sarkis, I. Tal, A. Vardy, and W. J. Gross, “Hardware implementation of successive-cancellation decoders for polar codes,” *Journal of Signal Processing Systems*, vol. 69, no. 3, pp. 305–315, 2012.

[12] G. Sarkis, P. Giard, A. Vardy, C. Thibeault, and W. J. Gross, “Fast polar decoders: Algorithm and implementation,” *IEEE Journal on Selected Areas in Communications*, vol. 32, no. 5, pp. 946–957, 2014.

[13] G. Sarkis, P. Giard, A. Vardy, C. Thibeault, and W. J. Gross, “Fast list decoders for polar codes,” *IEEE Journal on Selected Areas in Communications*, vol. 34, no. 2, pp. 318–328, 2016.

[14] Y. Fan and C.-y. Tsui, “An efficient partial-sum network architecture for semi-parallel polar codes decoder implementation,” *IEEE Transactions on Signal Processing*, vol. 62, no. 12, pp. 3165–3179, 2014.

[15] Y. Fan, C. Xia, J. Chen, C.-Y. Tsui, J. Jin, H. Shen, and B. Li, “A low-latency list successive-cancellation decoding implementation for polar codes,” *IEEE Journal on Selected Areas in Communications*, vol. 34, no. 2, pp. 303–317, 2016.

[16] P. Giard, A. Balatsoukas-Stimming, T. C. Müller, A. Bonetti, C. Thibeault, W. J. Gross, P. Flatresse, and A. Burg, “Polarbear: A 28-nm fd-soi asic for decoding of polar codes,” *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, vol. 7, no. 4, pp. 616–629, 2017.

[17] P. Giard, G. Sarkis, C. Thibeault, and W. J. Gross, “Multi-mode unrolled architectures for polar decoders,” *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 63, no. 9, pp. 1443–1453, 2016.

[18] E. Abbe and E. Telatar, “Polar codes for the m-user multiple access channel,” *IEEE Transactions on Information Theory*, vol. 58, no. 8, pp. 5437–5448, 2012.

[19] H. Mahdavifar, M. El-Khamy, J. Lee, and I. Kang, “Achieving the uniform rate region of general multiple access channels by polar coding,” *IEEE Transactions on Communications*, vol. 64, no. 2, pp. 467–478, 2016.

[20] E. Şaşoğlu, E. Telatar, and E. M. Yeh, "Polar codes for the two-user multiple-access channel," *IEEE Transactions on Information Theory*, vol. 59, no. 10, pp. 6583–6592, 2013.

[21] N. Goela, E. Abbe, and M. Gastpar, "Polar codes for broadcast channels," *IEEE Transactions on Information Theory*, vol. 61, no. 2, pp. 758–782, 2015.

[22] M. Mondelli, S. H. Hassani, I. Sason, and R. L. Urbanke, "Achieving martonâĂŹs region for broadcast channels using polar codes," *IEEE Transactions on Information Theory*, vol. 61, no. 2, pp. 783–800, 2015.

[23] M. Andersson, V. Rathi, R. Thobaben, J. Kliewer, and M. Skoglund, "Nested polar codes for wiretap and relay channels," *IEEE Communications Letters*, vol. 14, no. 8, pp. 752–754, 2010.

[24] H. Mahdavifar and A. Vardy, "Achieving the secrecy capacity of wiretap channels using polar codes," *IEEE Transactions on Information Theory*, vol. 57, no. 10, pp. 6428–6443, 2011.

[25] O. O. Koyluoglu and H. El Gamal, "Polar coding for secure transmission and key agreement," *IEEE Transactions on Information Forensics and Security*, vol. 7, no. 5, pp. 1472–1483, 2012.

[26] E. Şaşoğlu and A. Vardy, "A new polar coding scheme for strong security on wiretap channels," in *Proceedings of IEEE International Symposium on Information Theory Proceedings (ISIT)*, 2013, pp. 1117–1121.

[27] E. Arıkan, "Source polarization," in *Proceedings of IEEE International Symposium on Information Theory Proceedings (ISIT)*, 2010, pp. 899–903.

[28] H. S. Cronie and S. B. Korada, "Lossless source coding with polar codes," in *Information Theory Proceedings (ISIT), 2010 IEEE International Symposium on*. IEEE, 2010, pp. 904–908.

[29] S. B. Korada, E. Sasoglu, and R. Urbanke, "Polar codes: Characterization of exponent, bounds, and constructions," *IEEE Transactions on Information Theory*, vol. 56, no. 12, pp. 6253–6264, 2010.

[30] D. Burshtein and A. Strugatski, "Polar write once memory codes," *IEEE Transactions on Information Theory*, vol. 59, no. 8, pp. 5088–5101, 2013.

[31] A. Alamdar-Yazdi and F. R. Kschischang, “A simplified successive-cancellation decoder for polar codes,” *IEEE communications letters*, vol. 15, no. 12, pp. 1378–1380, 2011.

[32] E. Arıkan, “Systematic polar coding,” *IEEE communications letters*, vol. 15, no. 8, pp. 860–862, 2011.

[33] M. Bakshi, S. Jaggi, and M. Effros, “Concatenated polar codes,” in *Proceedings of IEEE International Symposium on Information Theory Proceedings (ISIT)*, 2010, pp. 918–922.

[34] M. El-Khamy, H. Mahdavifar, G. Feygin, J. Lee, and I. Kang, “Relaxed polar codes,” *IEEE Transactions on Information Theory*, vol. 63, no. 4, pp. 1986–2000, 2017.

[35] S. B. Korada, *Polar codes for channel and source coding*. Ph.D. dissertation, Ecole Polytechnique Fédérale de Lausanne, 2009.

[36] H. Mahdavifar, M. El-Khamy, J. Lee, and I. Kang, “Performance limits and practical decoding of interleaved reed-solomon polar concatenated codes,” *IEEE Transactions on Communications*, vol. 62, no. 5, pp. 1406–1417, 2014.

[37] R. Mori and T. Tanaka, “Performance and construction of polar codes on symmetric binary-input memoryless channels,” in *Proceedings of IEEE International Symposium on Information Theory*, 2009, pp. 1496–1500.

[38] I. Tal and A. Vardy, “How to construct polar codes,” *IEEE Transactions on Information Theory*, vol. 59, no. 10, pp. 6562–6582, 2013.

[39] I. Tal and A. Vardy, “List decoding of polar codes,” *IEEE Transactions on Information Theory*, vol. 61, no. 5, pp. 2213–2226, 2015.

[40] E. Arıkan, “Polar codes: A pipelined implementation,” in *Proceedings of 4th International Symposium on Broadband Communincations (ISBC)*, 2010, pp. 11–14.

[41] C. Leroux, A. J. Raymond, G. Sarkis, and W. J. Gross, “A semi-parallel successive-cancellation decoder for polar codes,” *IEEE Transactions on Signal Processing*, vol. 61, no. 2, pp. 289–299, 2013.

[42] B. Yuan and K. K. Parhi, “Low-latency successive-cancellation list decoders for polar codes with multibit decision,” *IEEE Transactions on Very Large Scale Integration*

(VLSI) Systems, vol. 23, no. 10, pp. 2268–2280, 2015.

[43] D. Goldin and D. Burshtein, “Improved bounds on the finite length scaling of polar codes,” *IEEE Transactions on Information Theory*, vol. 60, no. 11, pp. 6966–6978, 2014.

[44] V. Guruswami and P. Xia, “Polar codes: Speed of polarization and polynomial gap to capacity,” *IEEE Transactions on Information Theory*, vol. 61, no. 1, pp. 3–16, 2015.

[45] S. H. Hassani, “Polarization and spatial coupling: Two techniques to boost performance,” *Ecole Polytechnique Fédérale de Lausanne*, no. 5706, 2013.

[46] S. H. Hassani, K. Alishahi, and R. L. Urbanke, “Finite-length scaling for polar codes,” *IEEE Transactions on Information Theory*, vol. 60, no. 10, pp. 5875–5898, 2014.

[47] S. B. Korada, A. Montanari, E. Telatar, and R. Urbanke, “An empirical scaling law for polar codes,” in *Proceedings of IEEE International Symposium on Information Theory Proceedings (ISIT)*, 2010, pp. 884–888.

[48] M. Mondelli, S. H. Hassani, and R. L. Urbanke, “Scaling exponent of list decoders with applications to polar codes,” *IEEE Transactions on Information Theory*, vol. 61, no. 9, pp. 4838–4851, 2015.

[49] E. Hof, I. Sason, and S. Shamai, “Polar coding for reliable communications over parallel channels,” in *Proceedings of IEEE Information Theory Workshop (ITW)*, 2010, pp. 1–5.

[50] H. Mahdavifar, M. El-Khamy, J. Lee, and I. Kang, “Polar coding for bit-interleaved coded modulation,” *IEEE Transactions on Vehicular Technology*, vol. 65, no. 5, pp. 3115–3127, 2016.

[51] E. Şaşoğlu, “Polarization in the presence of memory,” in *Proceedings of IEEE International Symposium on Information Theory Proceedings (ISIT)*, 2011, pp. 189–193.

[52] D. Bladsjö, M. Hogan, and S. Ruffini, “Synchronization aspects in LTE small cells,” *IEEE Communications Magazine*, vol. 51, no. 9, pp. 70–77, 2013.

[53] H. Mercier, V. K. Bhargava, and V. Tarokh, “A survey of error-correcting codes for channels with symbol synchronization errors,” *IEEE Communications Surveys & Tutorials*, vol. 12, no. 1, 2010.

[54] E. Şaşoğlu *et al.*, "Polarization and polar codes," *Foundations and Trends® in Communications and Information Theory*, vol. 8, no. 4, pp. 259–381, 2012.

[55] E. Şaşoğlu and I. Tal, "Polar coding for processes with memory," in *Proceedings of IEEE International Symposium on Information Theory (ISIT)*, 2016, pp. 225–229.

[56] M. Mitzenmacher *et al.*, "A survey of results for deletion channels and related synchronization channels," *Probability Surveys*, vol. 6, pp. 1–33, 2009.

[57] S. N. Diggavi and M. Grossglauser, "On transmission over deletion channels," in *Proceedings of the Annual Allerton Conference on Communication Control and Computing*, vol. 39, no. 1, 2001, pp. 573–582.

[58] R. Dobrushin, "Mathematical problems in the shannon theory of optimal coding of information," in *Proceedings of 4th Berkeley Symposium on Mathematics, Statistics, and Probability*, vol. 1, 1961, pp. 211–252.

[59] V. Strassen, "Asymptotische abschätzungen in shannon's informationstheorie," *IEEE Transactions on 3rd Prague Conference Information Theory*, pp. 689–723, 1962.

[60] M. Hayashi, "Information spectrum approach to second-order coding rate in channel coding," *IEEE Transactions on Information Theory*, vol. 55, no. 11, pp. 4947–4966, 2009.

[61] Y. Polyanskiy, H. V. Poor, and S. Verdú, "Channel coding rate in the finite blocklength regime," *IEEE Transactions on Information Theory*, vol. 56, no. 5, pp. 2307–2359, 2010.

[62] J.-P. Tillich and G. Zémor, "Discrete isoperimetric inequalities and the probability of a decoding error," *Combinatorics, Probability and Computing*, vol. 9, no. 5, pp. 465–479, 2000.

[63] A. Montanari, "Finite size scaling and metastable states of good codes," in *Proceedings of Allerton Cconference on Communication, Control, and Ccomputing*, vol. 39, no. 1, 2001, pp. 655–661.

[64] E. Berlekamp, R. McEliece, and H. Van Tilborg, "On the inherent intractability of certain coding problems (corresp.)," *IEEE Transactions on Information Theory*, vol. 24, no. 3, pp. 384–386, 1978.

[65] G. D. Forney, *Concatenated codes.* Cambridge, MA: MIT Press, 1965.

[66] A. Amraoui, A. Montanari, T. Richardson, and R. Urbanke, "Finite-length scaling for iteratively decoded ldpc ensembles," *IEEE Transactions on Information Theory*, vol. 55, no. 2, pp. 473–498, 2009.

[67] S. Kudekar, T. Richardson, and R. L. Urbanke, "Spatially coupled ensembles universally achieve capacity under belief propagation," *IEEE Transactions on Information Theory*, vol. 59, no. 12, pp. 7761–7813, 2013.

[68] M. Mondelli, S. H. Hassani, and R. L. Urbanke, "How to achieve the capacity of asymmetric channels," *IEEE Transactions on Information Theory*, vol. 64, no. 5, pp. 3371–3393, 2018.

[69] S. Kudekar, S. Kumar, M. Mondelli, H. D. Pfister, E. Şaşoğlu, and R. Urbanke, "Reed-muller codes achieve capacity on erasure channels," in *Proceedings of the 48th annual ACM Symposium on Theory of Computing*, 2016, pp. 658–669.

[70] S. Kudekar, S. Kumar, M. Mondelli, H. D. Pfister, E. ŞaşoÇğlu, and R. L. Urbanke, "Reed–muller codes achieve capacity on erasure channels," *IEEE Transactions on Information Theory*, vol. 63, no. 7, pp. 4298–4316, 2017.

[71] M. Mondelli, S. H. Hassani, and R. L. Urbanke, "From polar to reed-muller codes: A technique to improve the finite-length performance," *IEEE Transactions on Communications*, vol. 62, no. 9, pp. 3084–3091, 2014.

[72] E. Arıkan and E. Telatar, "On the rate of channel polarization," in *Proceedings of IEEE International Symposium on Information Theory (ISIT)*, 2009, pp. 1493–1495.

[73] S. H. Hassani, R. Mori, T. Tanaka, and R. L. Urbanke, "Rate-dependent analysis of the asymptotic behavior of channel polarization," *IEEE Transactions on Information Theory*, vol. 59, no. 4, pp. 2267–2276, 2013.

[74] A. Eslami and H. Pishro-Nik, "On finite-length performance of polar codes: stopping sets, error floor, and concatenated design," *IEEE Transactions on Communications*, vol. 61, no. 3, pp. 919–929, 2013.

[75] H. D. Pfister and R. Urbanke, "Near-optimal finite-length scaling for polar codes over large alphabets," *arXiv preprint arXiv:1605.01997*, 2016.

[76] A. Vardy, "Algorithmic complexity in coding theory and the minimum distance problem," in *Proceedings of the 29th Annual ACM Symposium on Theory of computing*, 1997, pp. 92–109.

[77] F. J. MacWilliams and N. J. A. Sloane, *The theory of error-correcting codes*. Elsevier, 1977.

[78] V. Miloslavskaya and P. Trifonov, "Design of binary polar codes with arbitrary kernel," in *Proceedings of IEEE Information Theory Workshop (ITW)*, 2012, pp. 119–123.

[79] O. Shental, "Grab a cup of coffee and revisit the scaling exponent of polar codes," *Presentation at IEEE Information Theory and Application Workshop*, 2018.

[80] A. Vardy and Y. Be'ery, "Maximum-likelihood soft decision decoding of bch codes," *IEEE Transactions on Information Theory*, vol. 40, no. 2, pp. 546–554, 1994.

[81] P. Trifonov, "Efficient design and decoding of polar codes," *IEEE Transactions on Communications*, vol. 60, no. 11, pp. 3221–3227, 2012.

[82] P. Trifonov and V. Miloslavskaya, "Polar subcodes," *IEEE Journal on Selected Areas in Communications*, vol. 34, no. 2, pp. 254–266, 2016.

[83] D. Wu, Y. Li, and Y. Sun, "Construction and block error rate analysis of polar codes over awgn channel based on gaussian approximation," *IEEE Communications Letters*, vol. 18, no. 7, pp. 1099–1102, 2014.

[84] K. Niu and K. Chen, "Stack decoding of polar codes," *Electronics letters*, vol. 48, no. 12, pp. 695–697, 2012.

[85] V. Miloslavskaya and P. Trifonov, "Sequential decoding of polar codes," *IEEE Communications Letters*, vol. 18, no. 7, pp. 1127–1130, 2014.

[86] M.-C. Chiu and W.-D. Wu, "Reduced-complexity scl decoding of multi-crc-aided polar codes," *arXiv preprint arXiv:1609.08813*, 2016.

[87] A. Viterbi, "Error bounds for convolutional codes and an asymptotically optimum decoding algorithm," *IEEE Transactions on Information Theory*, vol. 13, no. 2, pp. 260–269, 1967.

[88] P. Trifonov, "Star polar subcodes," in *Proceedings of IEEE Wireless Communications*

and Networking Conference Workshops Proceedings (WCNC), 2017, pp. 1–6.

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

[90] Y. Wang, K. R. Narayanan, and Y.-C. Huang, “Interleaved concatenations of polar codes with bch and convolutional codes,” *IEEE Journal on Selected Areas in Communications*, vol. 34, no. 2, pp. 267–277, 2016.

[91] P. CharnKeitKong, H. Imai, and K. Yamaguchi, “On classes of rate k/(k+ 1) convolutional codes and their decoding techniques,” *IEEE Transactions on Information Theory*, vol. 42, no. 6, pp. 2181–2193, 1996.

[92] V. I. Levenshtein, “Binary codes capable of correcting deletions, insertions, and reversals,” in *Soviet physics doklady*, vol. 10, no. 8, 1966, pp. 707–710.

[93] R. Varshamov and G. M. Tenengolts, “Codes with correct single asymmetric errors,” *Automation and Remote Control*, vol. 26, no. 2, pp. 286–290, 1965.

[94] A. S. Helberg and H. C. Ferreira, “On multiple insertion/deletion correcting codes,” *IEEE Transactions on Information Theory*, vol. 48, no. 1, pp. 305–308, 2002.

[95] T. G. Swart and H. C. Ferreira, “A note on double insertion/deletion correcting codes,” *IEEE Transactions on Information Theory*, vol. 49, no. 1, pp. 269–273, 2003.

[96] K. Saowapa, H. Kaneko, and E. Fujiwara, “Systematic binary deletion/insertion error correcting codes capable of correcting random bit errors,” *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, vol. 83, no. 12, pp. 2699–2705, 2000.

[97] R. G. Gallager, “Sequential decoding for binary channels with noise and synchronization errors,” Massachusetts Institute of Technology: Lincoln Laboratory, Tech. Rep., 1961.

[98] B. Brink, H. Ferreira, and W. Clarke, “Pruned convolutional codes for flexible unequal error protection against insertion/deletion/reversal errors,” in *Proceedings of IEEE International Symposium on Information Theory*, 2000, p. 260.

[99] M. Dos Santos, W. Clarke, H. Ferreira, and T. Swart, “Correction of insertions/deletions

using standard convolutional codes and the viterbi decoding algorithm," in *Proceedings og IEEE Information Theory Workshop (ITW)*, 2003, pp. 187–190.

[100] L. Cheng, H. C. Ferreira, and T. G. Swart, "Bidirectional viterbi decoding using the levenshtein distance metric for deletion channels," in *Proceedings of IEEE Information Theory Workshop (ITW)*, 2006, pp. 254–258.

[101] M. C. Davey and D. J. MacKay, "Reliable communication over channels with insertions, deletions, and substitutions," *IEEE Transactions on Information Theory*, vol. 47, no. 2, pp. 687–698, 2001.

[102] S. K. Hanna and S. El Rouayheb, "Guess & check codes for deletions and synchronization," in *Proceedings of IEEE International Symposium on Information Theory (ISIT)*, 2017, pp. 2693–2697.

[103] J. Chen, M. D. Mitzenmacher, C. Ng, and N. Varnica, "Concatenated codes for deletion channels," in *Proceedings of International Symposium on Information Theory (ISIT)*, 2003, p. 218.

[104] V. Guruswami and R. Li, "Coding against deletions in oblivious and online models," in *Proceedings of the 29th Annual ACM-SIAM Symposium on Discrete Algorithms*, 2018, pp. 625–643.

[105] B. Shuval and I. Tal, "Fast polarization for processes with memory," *arXiv preprint arXiv:1710.02849*, 2017.

[106] K. Tian, A. Fazeli, A. Vardy, and R. Liu, "Polar codes for channels with deletions," in *Proceedings of 55th Annual IEEE Allerton Conference on Communication, Control, and Computing*, 2017, pp. 572–579.

[107] E. K. Thomas, V. Y. Tan, A. Vardy, and M. Motani, "Polar coding for the binary erasure channel with deletions," *IEEE Communications Letters*, vol. 21, no. 4, pp. 710–713, 2017.

[108] S. Diggavi and M. Grossglauser, "On information transmission over a finite buffer channel," *IEEE Transactions on Information Theory*, vol. 52, no. 3, pp. 1226–1237, 2006.

[109] E. ŞaŞoğlu, "Polar coding theorems for discrete systems," *Ecole Polytechnique Fédérale de Lausanne*, 2011.