CONCATENATED POLAR CODES AND JOINT SOURCE-CHANNEL DECODING

A Dissertation

by

YING WANG

Submitted to the Office of Graduate and Professional Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

Chair of Committee,	Krishna R. Narayanan
Committee Members,	Anxiao (Andrew) Jiang
	Joseph Boutros
	Alex Sprintson
Head of Department,	Miroslav M. Begovic

August 2017

Major Subject: Electrical Engineering

ABSTRACT

In this dissertation, we mainly address two issues: 1. improving the finite-length performance of capacity-achieving polar codes; 2. use polar codes to efficiently exploit the source redundancy to improve the reliability of the data storage system.

In the first part of the dissertation, we propose interleaved concatenation schemes of polar codes with outer binary BCH and convolutional codes to improve the finite-length performance of polar codes. For asymptotically long blocklength, we show that our schemes achieve exponential error decay rate which is much larger than the sub-exponential decay rate of standalone polar codes. In practice we show by simulation that our schemes outperform stand-alone polar codes decoded with successive cancellation or belief propagation decoding. The performance of concatenated polar and convolutional codes can be comparable to stand-alone polar codes with list decoding in the high signal to noise ratio regime. In addition to this, we show that the proposed concatenation schemes require lower memory and decoding complexity in comparison to belief propagation and list decoding of polar codes. With the proposed schemes, polar codes are able to strike a good balance between performance, memory and decoding complexity.

The second part of the dissertation is devoted to improving the decoding performance of polar codes where there is leftover redundancy after source compression. We focus on language-based sources, and propose a joint-source channel decoding scheme for polar codes. We show that if the language decoder is modeled as erasure correcting outer block codes, the rate of inner polar codes can be improved while still guaranteeing a vanishing probability of error. The improved rate depends on the frozen bit distribution of polar codes and we provide a formal proof for the convergence of that distribution. Both lower bound and maximum improved rate analysis are provided. To compare with the non-

iterative joint list decoding scheme for polar codes, we study a joint iterative decoding scheme with graph codes. In particular, irregular repeat accumulate codes are exploited because of low encoding/decoding complexity and capacity achieving property for the binary erasure channel. We propose how to design optimal irregular repeat accumulate codes with different models of language decoder. We show that our scheme achieves improved decoding thresholds. A comparison of joint polar decoding and joint irregular repeat accumulate decoding is given.

DEDICATION

To my parents and grandmother

ACKNOWLEDGMENTS

First and foremost, I would like to devote the greatest thanks to my advisor, Prof. Krishna Narayanan for his guidance throughout my PhD study. This dissertation cannot be accomplished without his consistent support. It is his great insight and passion that help me move forward. He is a professor who always comes up with interesting problems and valuable ideas. He is interested in exploring new ideas in different areas and always refreshes himself with new knowledge. He is open minded and enjoys taking challenges. Prof. Narayanan gives me a lot of freedom to work on topics that I am interested in. He is very thoughtful and responsible for his students. I am deeply impressed by his personality. I would say that the past five years are the most precious time in my life.

I would like to thank Prof. Andrew Jiang, Prof. Joseph Boutros and Prof. Alex Sprintson for serving on my committee. Part of the work in this dissertation is done jointly with Prof. Andrew Jiang, who is smart and offers consistent help on my research work. I highly appreciate the patient guidance from him. Prof. Boutros is always willing to offer help when I am encountered with problems. I appreciate the remote Skype meetings with him. I am grateful to Prof. Henry Pfister for his teaching of two channel coding courses and his guidance in the first two years. I would also like to thank Prof. Jiang Hu, Prof. Scott Miller, Prof. Zixiang Xiong, Prof. Jean-Francois Chamberland and Prof. Gwan Choi for helpful discussions on my research work.

Thanks to the internship at Bell Labs in Stuttgart, I had chance to learn the culture and history in Germany. I would like to thank Dr. Laurent Schmalen and Dr. Vahid Aref for their help during my internship. I enjoyed fruitful discussions with them. I would also like to thank Dr. Zvonimir Bandic, Dr. Minghai Qin and Dr. Kiran Gunnam during my internship at HGST, Western Digital.

I am grateful to my colleagues in the Information Sciences and Systems group, Jerry Huang, Avinash Vem, Nagaraj Thenkarai Janakiraman, Pingchung Wang, Santosh Kumar Emmadi, Swandand Kadhe, Narayanan Regaswamy, Santhosh Kumar Vanaparthy, Carlos Mejia, etc. I appreciate Jerry spending a lot time discussing problems with me and giving feedbacks. I enjoyed the time taking classes, discussing problems and attending seminars together with my colleagues.

Thanks also to my friends Aky Chen, Han Deng, Pengcheng Pi, Ziwei Xuan, Qian Hu, Qi Liao, Jun Chen, Minjie Lu, Muxi Yan, Yiling Song, etc. I appreciate the time together with those friends. We share both happiness and sadness. It is their company and support that help me get through difficulties in life in the past five years.

Finally, I would like to thank my parents, brother and my grandmother. I appreciate their enduring and selfless love and support. The happy and peaceful family always inspires me to move forward.

CONTRIBUTORS AND FUNDING SOURCES

Contributors

This work was supported by a dissertation committee consisting of Professor Krishna R. Narayanan, Professor Joseph Boutros and Professor Alex Sprintson of the Department of Electrical & Computer Engineering, and Professor Andrew (Anxiao) Jiang of the Department of Computer Science & Engineering.

All work conducted for the dissertation was completed by the student independently.

Funding Sources

The work in Chapter 3 was supported by Qatar National Research Foundation under Grant NPRP 5-597-2-241. The work in Chapter 4 and 5 was supported by the National Science Foundation under grants IIP-1439722 and ECCS-1611285.

NOMENCLATURE

BCH	Bose, Chaudhuri and Hocquenghem
NVM	Non-volatile Memory
ECC	Error Correcting Code
RS	Reed Solomon
LDPC	Low Density Parity Check
LZW	Lempel-Ziv-Welch
JSCD	Joint Source-Channel Decoding
BEC	Binary Erasure Channel
EXIT	Extrinsic Information Transfer
BDMS	Binary-Discrete Memoryless Channel
BSC	Binary Symmetric Channel
AWGN	Additive White Gaussian Noise
BPSK	Binary Phase-Shift Keying
SC	Successive Cancellation
LLR	Log-Likelihood Ratio
CRC	Cyclic Redundancy Check
BP	Belief Propagation
MAP	Maximum *a Posteriori*
ML	Maximum Likelihood
SNR	Signal to Noise Ratio
RCPC	Rate Compatible Punctured Convolutional

BCJR	Bahl, Cocke, Jelinek and Raviv
SOMID	Soft-Output Multistage Iterative Decoding
GMD	Generalized Minimum Distance
VLC	Variable Length Code
BMS	Binary Memoryless Source
RA	Repeat Accumulate
IRA	Irregular Repeat Accumulate
VN	Variable Node
CN	Check Node
ACC	Accumulator
PEG	Progressive Edge Growth
$\mathbb{R}$	Set of Real Numbers

TABLE OF CONTENTS

Page

LIST OF FIGURES

FIGURE Page

LIST OF TABLES

TABLE Page

1. INTRODUCTION

Polar codes, invented by Arikan [2] in 2008, are a major breakthrough in coding theory in recent years. Polar codes are the first known capacity-achieving codes with explicit construction methods; they have low encoding and decoding complexities; they don't exhibit error floors in bit error rates. Polar codes have been widely applied to problems including channel coding [3, 4], source coding [4, 5, 6], transmission through multiple access channels (MACs) [7, 8, 9], broadcast channels [10, 11] and wiretap channels [12, 13]. Many asymptotic properties of polar codes have been studied, such as the rate of channel polarization [14], their scaling exponents [15, 16, 17] and minimum distance [4], etc. However there are several drawbacks of polar codes. First, the block error probability of polar codes decays with the blocklength N only as $O(2^{N^{0.5-\epsilon}})$. The sub-exponential error decay rate cannot be improved by using advanced decoding strategies such as belief propagation (BP) or list decoding. Second, the finite length performance (for small to moderate block lengths) of polar codes is not as impressive as that of other state of the art coding schemes [3, 18].

In the first part of the dissertation, we aim at improving the finite-length performance of polar codes. We propose interleaved concatenation schemes of polar codes with outer binary codes, including binary Bose, Chaudhuri and Hocquenghem (BCH) and convolutional codes. Our insight of using interleaved concatenation comes from the increased cutoff rate of the channel by the polarization process [19]. The interleaved concatenation is a clever way for sequential decoding to efficiently take advantage of the increased cutoff rate of polar codes. With the interleaved structure, sequential decoding can exploit the improved cutoff rate by using long constraint length convolutional codes, while maintaining a low complexity of communication at rates up to the capacity. We show that for asymp-

totically long blocklength our schemes give exponential error decay rate which is a large improvement over the standalone polar codes. For moderate blocklengths, our simulation results show show substantial performance improvement over existing schemes.

In the second part of the dissertation, we study how to use polar codes to exploit the benefit of source redundancy. Our motivation is from the observation that in the practical data storage system, there is always residual redundancy even after source compression. Joint source-channel decoding (JSCD) is a good way to take advantage of source redundancy. There are works on JSCD with convolutional codes, Turbo codes and LDPC codes. However, the performance of polar codes with JSCD has not been studied in detail. We show that list decoding of polar codes can fully exploit the structure inherent in the source redundancy. For language based source , the redundancy is described by a tree structured dictionary. The tree structure of list decoding of polar codes thus can be efficiently combined with the dictionary in JSCD. It has been shown that the concatenation of polar codes with a few cyclic redundancy check (CRC) bits under list decoding has outstanding performance compared with Turbo and LDPC codes [20]. If polar codes are used as error correction codes for sources with redundancy, the system can be considered as concatenation of polar codes with a set of short length outer codes. The source redundancy has a critical advantage over CRC constraint. It acts as a set of local constraints thus is able to detect the validity of decoded paths before decoding the whole codeword. We propose a joint list decoding scheme for polar codes for language-based sources, and show impressive performance improvement obtained from the scheme. To understand how much gain can be fully achieved by exploiting the source redundancy, we make simplifying assumptions of the source model and analyze the improved rate of polar codes in the asymptotically long blocklength. We show that the improved rate depends on the frozen bit distribution of polar codes, and the frozen bit distribution converges to a limit. We then derive a lower bound of improved rate of polar codes based on the converged frozen bit distribution. Fur-

ther we propose an optimal bit allocation algorithm to maximize the increase in rate of polar codes.

The joint decoding scheme with polar codes is a non-iterative decision feedback scheme. Since iterative decoding is shown to be a powerful and efficient tool for decoding Turbo and LDPC codes with low decoding complexity, it is interesting to study how to fully explore the benefit of iterative decoding when the source redundancy is available at the decoder. We extend our work to study an alternative scheme to explore the natural redundancy – joint iterative source-channel decoding. We study how to design optimal irregular repeat accumulate codes in the joint decoding scheme with improved decoding thresholds. Finally we provide a comparison of joint polar decoding and joint LDPC decoding schemes to see the benefit of each scheme.

1.1 Organization

The rest of this dissertation is organized as follows. In Chapter 2 we include some background on polar codes and their decoding methods, rate of polarization, cutoff rate and sequential decoding. In Chapter 3 we introduce the finite-length problem of polar codes, and propose interleaved concatenation schemes of polar codes with outer binary codes. For convolutional outer codes, we use sequential decoding to take advantage of increased cutoff rate by polarization. We study comprehensively the memory space, decoding complexity and performance compared to other existing schemes such as the concatenated RS and polar codes.

In Chapter 4 we study the problem of how to exploit the natural redundancy in the source to improve the decoding performance in the data storage system. We propose a JSCD scheme based on polar codes to efficiently exploit the natural redundancy. Simplifying assumptions of the source model is made, and we study the improved rate of polar codes by the natural redundancy.

In Chapter 5 we extend the joint decoding scheme proposed in Chapter 4 by considering LDPC codes as channel codes and do joint iterative decoding. In particular, we consider irregular repeat accumulate codes because of their special structure, capacity-achieving property for BEC and ease of implementation, etc. Linear programming and extrinsic information transfer (EXIT) chart methods are used to design codes with natural redundancy. Finally we give a comparison of joint polar decoding and joint LDPC decoding schemes.

1.2 Notation

Throughout the dissertation, we use the following notation. The channels are assumed to be binary-discrete memoryless channels (B-DMCs). We use uppercase letters X, Y to denote random variables and lower case letters to x, y to denote the realizations. We use $W : \mathcal{X} \to \mathcal{Y}$ to denote a binary-discrete memoryless channel with input alphabet $\mathcal{X}$, output alphabet $\mathcal{Y}$, and transition probability $W(y|x)$ where $x \in \mathcal{X}$ and $y \in \mathcal{Y}$. X_1^n is used to represent a vector of sequence $[X_1, \cdots, X_n]$. We denote $I(X;Y)$ as the mutual information between X and Y.

We use BEC(ϵ) to denote the BEC with erasure probability ϵ, and use BSC(α) to represent the binary symmetric channel (BSC) with crossover probability α. When the codewords are transmitted through additive white Gaussian noise (AWGN) channel, we assume binary phase-shift keying (BPSK) modulation is employed if the modulation scheme is not explicitly specified.

2. BACKGROUND

In this chapter we provide a background on polar codes and their asymptotic properties, the cutoff rate of a channel and sequential decoding of convolutional codes, which lay the foundation for the following chapters.

2.1 Channel polarization and polar codes

In his seminal work [2], Arikan introduced the phenomenon of channel polarization by the processes of channel combining and splitting. We illustrate the idea of channel polarization from the basic transform shown in Fig. 2.1. The transform is characterized by the matrix $G_2 = \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}$. Let U_1 and U_2 be two bits which independently take 0 and 1 values with probability $1/2$. Let X_1 and X_2 be two binary transmitted bits. We have $X_1^2 = U_1^2 G_2$, that is, $X_1 = U_1 \oplus U_2$ and $X_2 = U_2$. Here $\oplus$ denotes the modulo two summation. Let Y_1 and Y_2 be two received bits. First in the channel combining process two copies of the channel W are combined to generate a channel W_2:

$$W_2(y_1^2|u_1^2) = W(y_1|u_1 + u_2)W(y_2|u_2).$$

Then the combined channel is split into two different channels resulting in the following transform: $(W, W) \mapsto (W_2^{(1)}, W_2^{(2)})$ where $W_2^{(1)} : \{0,1\} \mapsto \mathcal{Y}^2$ and $W_2^{(2)} : \{0,1\} \mapsto \{0,1\} \times \mathcal{Y}^2$ are defined as

$$W_2^{(1)}(y_1^2|u_1) = \sum_{u_2} \frac{1}{2} W_2(y_1^2|u_1^2) = \sum_{u_2} \frac{1}{2} W(y_1|u_1 \oplus u_2)W(y_2|u_2),$$
$$W_2^{(2)}(y_1^2, u_1|u_2) = \frac{1}{2} W_2(y_1^2|u_1^2) = \frac{1}{2} W(y_1|u_1 \oplus u_2)W(y_2|u_2).$$

$W_2^{(1)}$ can be considered as the channel seen from U_1, considering U_2 as noise. $W_2^{(2)}$ can be seen as the channel seen from U_2, assuming that U_1 is decoded correctly. Let $I(W)$ be the symmetric capacity of a B-DMC W, which is defined as

$$I(W) = \sum_y \sum_x \frac{1}{2} W(y|x) \log \frac{W(y|x)}{\frac{1}{2}\left(W(y|0) + W(y|1)\right)}.$$

It can be seen that

$$\begin{aligned} I(W_2^{(1)}) + I(W_2^{(2)}) &= 2I(W), \\ I(W_2^{(1)}) \leq I(W) &\leq I(W_2^{(2)}) \end{aligned} \tag{2.1}$$

The preserve of the sum capacity follows from the chain rule given as

$$I(U_1^2; Y_1^2) = I(U_1; Y_1^2) + I(U_2; Y_1^2 | U_1).$$

We use the Bhattacharyya parameter $Z(W)$ to measure the reliability of the channel.

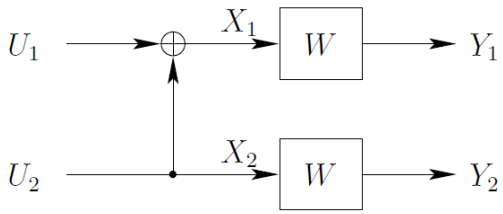


Figure 2.1: Basic transform

$Z(W)$ is defined as

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

It is an upper bound on the error probability of maximum likelihood (ML) decoding. The reliability after transformation has the following properties:

$$\begin{aligned} Z(W_2^{(2)}) &= Z(W)^2, \\ Z(W_2^{(1)}) &\leq 2Z(W) - Z(W)^2, \\ Z(W_2^{(2)}) &\leq Z(W) \leq Z(W_2^{(1)}). \end{aligned}$$

It is easy to see that $Z(W_2^{(1)}) + Z(W_2^{(2)}) \leq 2Z(W)$, which indicates that reliability can only improve under this transformation, whereas from (2.1) it can be seen that the sum capacity is preserved under the transformation.

From above analysis we see that after the basic transform, the channel W is converted into a better channel $W_2^{(2)}$ and a worse channel $W_2^{(1)}$. If the channel combining and splitting process is repeated n times, we get $N = 2^n$ different bit channels $W_N^{(i)}, i \in [1, N]$. Each time the recursive transform can be written as

$$(W_m^{(i)}, W_m^{(i)}) \mapsto (W_{2m}^{(2i)}, W_{2m}^{(2i+1)}).$$

In general, N copies of the channel are combined to create the channel

$$W_N(y_1^N|u_1^N) \triangleq W(y_1^N|u_1^N G_N) = \prod_{i=1}^{N} W(y_i|x_i)$$

due to the memoryless property of the channel W. W_N can be recursively constructed. Fig. 2.2 shows a recursive construction of W_N from two copies of $W_{N/2}$. The channel

splitting process then splits W_N back into a set of N bit channels

$$W_N^{(i)}(y_1^N, u_1^{i-1}|u_i) \triangleq \frac{1}{2^{N-1}} \sum_{u_{i+1}^N} W_N(y_1^N|u_1^N),\ i = 1, \cdots, N. \tag{2.2}$$

Arikan has proved the following polarization theorem:

Theorem 1. [2] For any B-DMC W, almost all bit channels $\{W_N^{(i)}\}$ polarize in the sense that, for any $\delta \in (0, 1)$, the fraction of indices $i \in \{1, \cdots, N\}$ for which $I(W_N^{(i)}) \in (1 - \delta, 1]$ goes to $I(W)$ and the fraction for which $I(W_N^{(i)}) \in [0, \delta)$ goes to $1 - I(W)$.

The idea of polar codes is to transmit information only on those noiseless channels while freezing bits on those noisy channels. Let $\mathcal{F}$ denote the set of frozen bits and let $\mathcal{F}^c$ be its complement set $\{1, \cdots, N\} \setminus \mathcal{F}$. Polar codes are formally defined in the following:

Definition 2. An (N, K) polar code for a B-DMC W is a code with its information set $\mathcal{F}^c$ chosen as a K-element subset of $\{1, \cdots, N\}$ such that $Z(W_N^{(i)}) \leq Z(W_N^{(j)})$ for all $i \in \mathcal{F}^c$ and $j \in \mathcal{F}$.

Construction of polar codes then involves finding a set of best channels $\mathcal{F}^c$ based on the channel reliabilities. The construction methods include the upgrading/degrading quantization method [21] and density evolution [22]. The encoding of polar codes is to use the generator matrix $G_N = R_N G_2^{\otimes n}$, where R_N is an $N \times N$ bit-reversal permutation matrix, and $\otimes$ is the Kronecker product.

2.2 Decoding of polar codes

In this section various decoding schemes of polar codes are reviewed.

2.2.1 Successive cancellation decoding

Let u_1^N and y_1^N be the uncoded bits and received sequence, respectively. The successive cancellation (SC) decoder generates an estimate of bits in $\mathcal{F}^c$. It first computes the log-

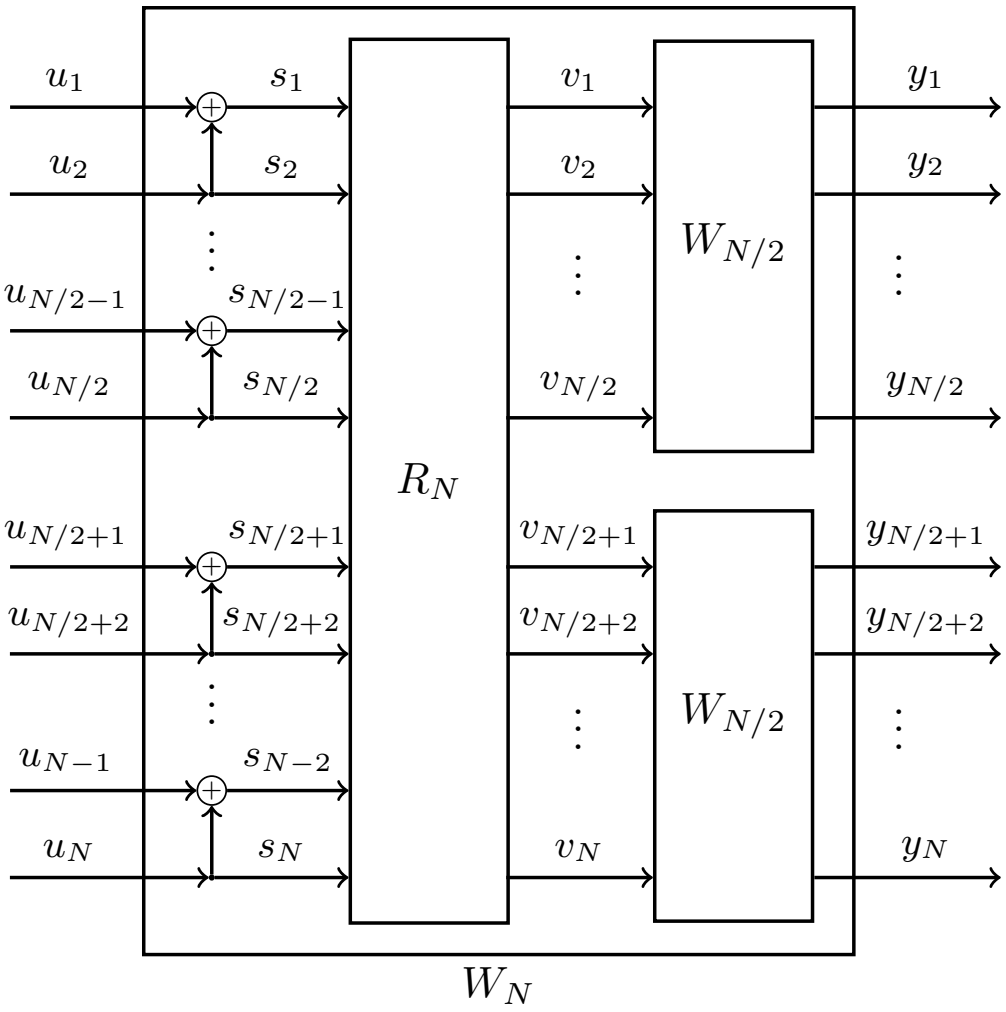


Figure 2.2: Recursive construction of W_N from two copies of $W_{N/2}$. R_N is the bit-reversal permutation.

likelihood ratio (LLR) of each bit channel

$$L_N^{(i)}(y_1^N, \hat{u}_1^{i-1}) = \log \frac{W_N^{(i)}(y_1^N, \hat{u}_1^{i-1}|u_i = 0)}{W_N^{(i)}(y_1^N, \hat{u}_1^{i-1}|u_i = 1)} \tag{2.3}$$

Then it makes decisions as follows: $\hat{u}_i = 0$ if $i \in \mathcal{F}$. If $i \in \mathcal{F}^c$,

$$\hat{u}_i = \begin{cases} 0, \text{ if } L_N^{(i)}(y_1^N, \hat{u}_1^{i-1}) \geq 0 \\ 1, \text{ otherwise.} \end{cases}$$

Arikan has shown that (2.3) admits a recursive structure with decoding complexity $O(N \log N)$. It has been proved that polar codes are capacity achieving under SC decoding.

2.2.2 List decoding

SC decoding suffers from error propagation, that is, if bits in early stages are decoded incorrectly, the later decoded bits will have high probability to be decoded in error. Instead of making hard decision of u_i at each stage, list decoder keeps a list of most probable paths [20]. In each stage, the decoder extends a path by hypothesizing both 0 and 1 for the unfrozen bit and the number of paths doubles. Assume the list size is L. When the number of paths exceeds L, the decoder picks L most probable paths as surviving paths and prunes the rest. After decoding the last bit, the most probable path is picked. Fig. 2.3 gives an illustration of list decoding with list size $L = 4$, with solid paths as surviving paths. The complexity of list decoding is $O(Ln \log n)$, where n is the block length. An extra improvement can be brought by adding CRC bits, which increases the minimum distance of polar codes and helps to select the most probable path in the list. The adaptive list decoder with a large list size can be used to fully exploit the benefit of CRC while largely reducing the decoder complexity [23].

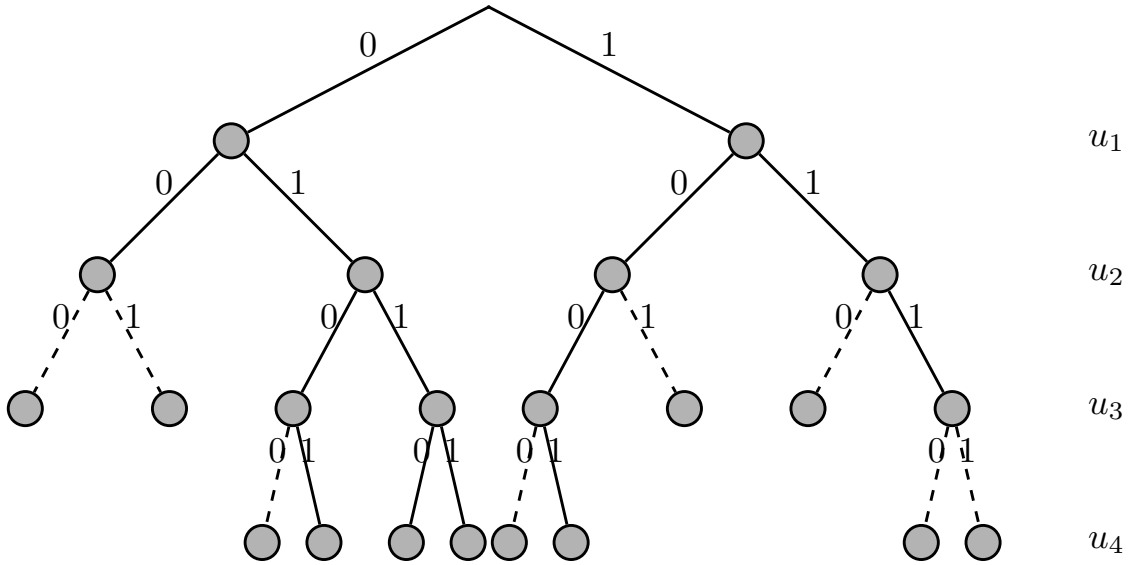


Figure 2.3: Tree structure of list decoding, with list size $L = 4$. The dashed paths are pruned.

2.2.3 Belief propagation decoding

Belief propagation (BP) decoding for polar codes was studied in [4, 18, 24]. For length $N = 2^n$ polar codes, there are $(n + 1)$ layers of bit nodes and n layers of check nodes in the tanner graph. It has been shown that BP decoding has superior performance over SC decoding in general, but the finite-length performance of BP decoding is still not satisfactory. The stopping set and minimum distance structure in the factor graph of polar codes are analyzed in [18]. Recently Reed Muller (RM) codes have been proved to achieve the capacity of BEC [25, 26]. RM codes essentially have a close relationship with polar codes [24, 27]. The main difference in the code construction is the way of picking rows in the generator matrix. Arikan gave a comparison of BP decoding between the two codes in [24]. He demonstrated the performance advantages of polar codes over RM codes under BP decoding. It is shown in [4, 28] that scheduling is important to obtain good perfor-

mance under BP decoding for general DMCs.

2.3 Error exponent of polar codes

Let $P_B(N, R)$ denote the block error probability of a code with blocklegth N and rate R. Arikan has shown the rate of polarization in the following theorem.

Theorem 3. [14] Let $R < I(W)$ and $\beta < \frac{1}{2}$ be fixed. For $N = 2^n$, $n \geq 0$, the best achievable block error probability under successive cancellation decoding satisfies

$$P_B(N, R) = o(2^{-N^\beta}).$$

A refined bound that includes rate as a parameter is derived in [29] and [30]:

$$P_B(N, R) = o(2^{-2^{(n+t\sqrt{n})/2}})$$

for any t satisfying $t < Q^{-1}(R/I(W))$, where $Q(x) = \frac{1}{\sqrt{2\pi}} \int_x^\infty e^{-u^2/2} du$.

A comparison between the error exponent of random codes and that of polar codes exposes some weaknesses of polar codes. The random coding bound on the ensembles of block codes is [31]

$$P_B(N, R) \leq 2^{-NE_r(R)}$$

where $E_r(R)$ is the random coding exponent which is defined as

$$E_r(R) = \max_{\mathbf{q}} \max_{\rho \in [0,1]} -\rho R + E_0(\rho, \mathbf{q}).$$

Here $\mathbf{q}(x)$ is the probability distribution of the input. Gallager has shown that for binary symmetric memoryless channels, $E_r(R)$ is maximized when X is uniformly distributed.

For random block code ensemble, there exists codes that have exponent $E_r(R) = 1$ under maximum *a posteriori* (MAP) decoding [31]. This is much larger than the sub-exponential error decay rate of polar codes.

To determine whether the suboptimal error exponent of polar codes is due to the SC decoder or the weakness in the code itself, we study the minimum distance of polar codes. The minimum distance of a length-N polar code is

$$d_{min} = \min_{i \in \mathcal{F}^c} 2^{wt(i)}$$

where $wt(i)$ is the number of 1's in the binary expansion of i, $i \in [0, N-1]$. An upper bound of d_{min} is given in the following theorem

Theorem 4. [4] For any rate $R > 0$ and any choice of information bits, the minimum distance of a polar code of length $N = 2^n$ is upper bounded by

$$d_{min} \leq 2^{\frac{n}{2} + c\sqrt{n}}$$

for large enough n and a constant c.

It follows that for any rate $R > 0$, $\beta > \frac{1}{2}$ and any B-DMC, the error probability of polar codes under MAP decoding satisfies $P_B^{MAP} > 2^{-N^\beta}$. We can see that SC decoder already achieves the same order of error exponent as MAP decoding. Thus the code itself is weak in terms of the error exponent.

2.4 Cutoff rate and sequential decoding

Sequential decoding is a search algorithm for guessing the correct path through the expanded tree of possible transmitted sequences. Consider a (n_o, k_o, ν) convolutional code with k_o input bits and n_o output bits at each stage with a constraint length of ν. It is

well known that the Viterbi algorithm provides a ML sequence decoder but the decoding complexity grows exponentially with the constraint length. Sequential decoding is a depth-first search on the code tree which provides a computationally efficient alternative to the Viterbi algorithm. Sequential decoding was first introduced by Wozencraft for decoding of convolutional codes in [32]. An improvement in the decoding complexity of sequential decoding algorithm was presented by Fano [33]. Fano's work largely inspired further research on sequential decoding. A practical stack decoding algorithm was proposed by Zigangirov [34] and Jelinek [35].

2.4.1 Cutoff rate

Definition 5. The cutoff rate of a channel W is defined as

$$R_c(W) = \max_{\mathbf{q}(x)} -\log_2 \left(\sum_y \Big(\sum_x \mathbf{q}(x) \sqrt{W(y|x)} \Big)^2 \right) \tag{2.4}$$

where $\mathbf{q}(x)$ is the input probability distribution.

The cutoff rate of a channel is a fundamental quantity associated with the channel and it has implications for ML decoding or minimum distance decoding and sequential decoding of certain classes of codes [36]. It is smaller than or equal to the capacity of the channel. In particular, its significance to sequential decoding of convolutional codes is discussed below.

In sequential decoding, the number of computations required to decode a given number of bits is a random variable, but is more or less independent of ν [37]. Let $\mathbf{q}(x)$ be the input probability distribution. Define $E_0(\rho, \mathbf{q})$ as

$$E_0(\rho, \mathbf{q}) = -\log \sum_y \left(\sum_x \mathbf{q}(x) P(y|x)^{1/(1+\rho)} \right)^{1+\rho},$$

and define Gallager function $E_0(\rho)$ as $E_0(\rho) = \max_{\mathbf{q}} E_0(\rho, \mathbf{q})$. We have $R_c = E_0(1)$. Let T_n be the number of hypotheses made on the nth node of the path. Gallager has shown an upper bound to T_n with sequential decoding in the following theorem.

Theorem 6. [31] Let R be the code rate and let ν be the constraint length of the code. With sequential decoding on a DMC the average number of hypothesis required per decoded subblock $\bar{T}_n$ satisfies

$$\bar{T}_n \leq 4(1 - \exp(-\nu E_0(1) + \nu R))^{-2}$$

for $R \leq E_0(1)$.

From Theorem 6 it can be seen that the average number of hypothesis is bounded when $R < R_c$. The converse result was first shown by Arikan in [38], that is, $\bar{T}_n = \infty$ for $R > R_c$. Jacobs and Berlekamp showed that the distribution of computations follows a Pareto distribution [39]. The cutoff rate is the rate at which the exponent of the Pareto distribution becomes 1 and, hence, the average computational effort becomes unbounded [40]. For all rates smaller than the cutoff rate, while the average computational effort is bounded, the average bit error rate of the ensemble of random convolutional code with sequential decoding can be bounded as [40]

$$P_b \leq K \mathrm{e}^{-n_o \nu \rho R_c} \tag{2.5}$$

where K is a constant independent of ν and R_c, and $0 \leq \rho < 1$.

These results together show that the cutoff rate represents a sharp transition in the average computational effort required to obtain arbitrarily small bit error rates with sequential decoding of convolutional codes, i.e., for all rates below the cutoff rate, very small bit error rates can be obtained at fairly small average computational complexity.

2.5 Increase in cutoff rate due to polarization

It was noticed by Massey that the sum cutoff rate can be increased if a channel is split into correlated subchannels [41]. Arikan showed that the basic transform given in 2.1 with successive cancellation decoder is able to achieve increased sum cutoff rate [19]. For a N-length polar transform, let $R_c(W_N^{(i)})$ denote the cutoff rate of the ith equivalent channel defined in (2.2) and let $\bar{R}_c = \frac{1}{N}\sum_{i=1}^{N} R_c(W_N^{(i)})$ denote the average cutoff rate. As shown by Arikan in [19], one of the remarkable aspects of polarization is that $\bar{R}_c$ monotonically increases with N and in the limit $N \to \infty$, $\bar{R}_c$ approaches the capacity of the channel $I(W)$. Thus, polarization can be thought of as a technique to increase the cutoff rate of the channel without decreasing the capacity.

In Fig. 2.4, we plot the ratio of the average cutoff rate to the channel capacity $(\bar{R}_c/I(W))$ with a polar transform of length $N = 2^n$ for an AWGN channel at $E_b/N_0 = 2$ dB. We can observe that the average cutoff rate increases towards the channel capacity as the length of the transform (code) increases. More importantly, it can be seen that there is a substantial increase in the cutoff rate even for small N. Even for $n = 10$, $\bar{R}_c$ reaches 95% of the channel capacity. This is a fact that we will exploit in the design of concatenated polar codes in Chapter 3.

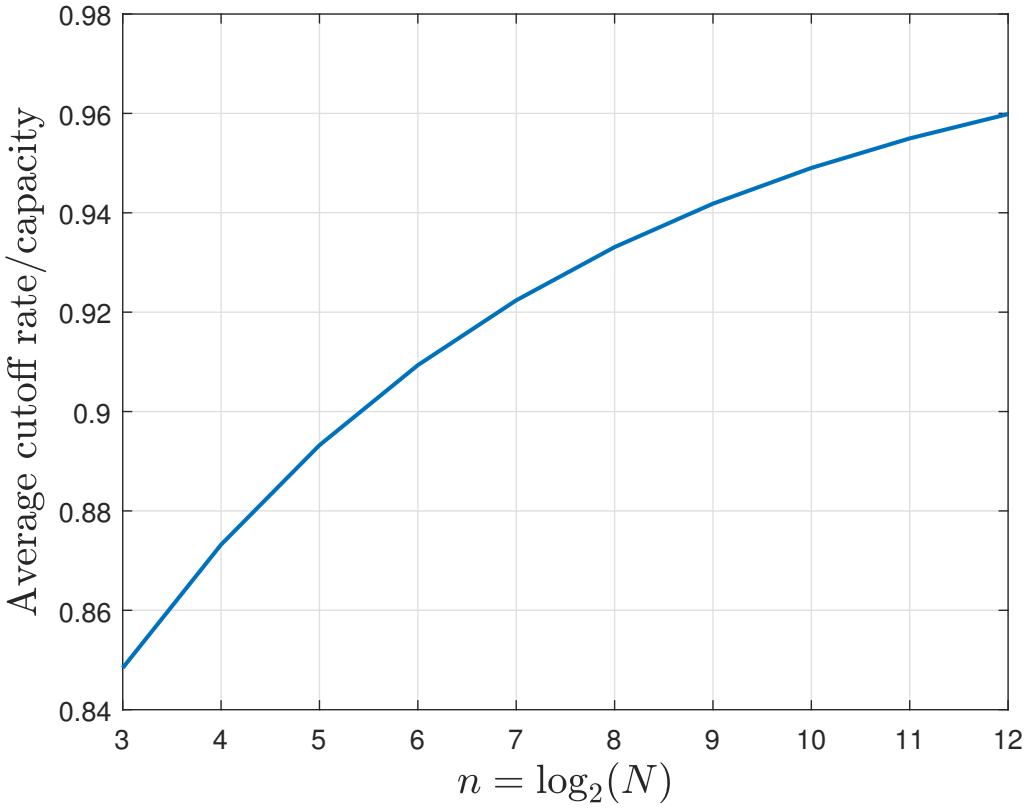


Figure 2.4: The average cutoff rate with a polar transformation of length $N = 2^n$ for a binary-input AWGN channel; $E_b/N_0 = 2$ dB

3. INTERLEAVED CONCATENATIONS OF POLAR CODES WITH OUTER BINARY CODES

3.1 Introduction

Polar codes are known for their ability to achieve the capacity of symmetric discrete memoryless channels with an explicit construction and a computationally-efficient SC decoding algorithm [2]. However, the error performance of polar codes in the finite length regime is not as impressive as that of other state of the art coding schemes [18]. In this chapter we use “frame error rate" to represent the error rate of the whole code, and use “block error rate" to represent the error rate of the inner or outer code. One way to improve the finite length performance of polar codes is to consider more sophisticated decoders than the SC decoder. BP decoding of polar codes has been proposed in [3], but the improvement is not significant over the AWGN channel. SC list decoding can provide substantial improvement over SC decoding [20]. However, both the memory requirement and the decoding complexity increase linearly with the list size; further, the error exponent cannot be improved with any finite list size [42]. Another way to improve the finite length performance is to change the structure of the code, such as through concatenation with other codes or by using non-binary construction schemes. For instance, it has been shown that the concatenation of polar codes with CRC bits can even outperform some LDPC codes under list decoding. The concatenation of polar codes with RS codes proposed in [43] has been shown to increase the frame error rate of the code to be almost exponential in the code length but the field size of RS codes also increases exponentially with the length of the polar code. The concatenation of polar codes with short outer block codes is

proposed in [44] and a modest improvement in performance has been reported. In [45], it is shown that the performance of polar codes can be improved by concatenation with an LDPC code.

In [46] and [47] interleaved concatenation of polar codes with outer RS codes and BCH codes are considered, respectively. In [46], Mahdavifar *et al.* show that the concatenation of polar codes with RS (referred to as RS-polar) codes can increase the error decay rate to be $O(2^{-N^{1-\epsilon}})$. In [47], Trifonov and Semenov view the concatenation schemes as multilevel coding with multistage decoding and apply the design criteria in [48] to the problem considered. They then provide empirical results for designs with outer BCH codes (referred to as BCH-polar codes).

We analyze interleaved concatenations of polar codes with outer binary codes such as BCH and convolutional codes. We show that these concatenated schemes strike a better balance between performance and complexity than existing schemes. The main contributions of this chapter and the insights obtained from this chapter are summarized in the following.

- First, we show that for binary polar codes, binary outer codes are more effective at reducing error propagation since they are matched to the SC decoder that works at the bit level. This indicates that RS-polar codes designed over non-binary fields may be mismatched to the SC decoder. We then provide an analysis of BCH-polar codes which have binary BCH outer codes and show that for these codes, for binary input discrete memoryless channels, the frame error rate decays exponentially with the code length for all code rates $R < C$, where C is the capacity of the channel.

- It is known that the polarization process increases the cutoff rate of the channel [19], while preserving the capacity. Interleaved concatenation is proposed as a way to naturally leverage the increase in the cutoff rate to decrease the complexity of

communication at rates close to the capacity. In particular, it is well known that sequential decoding of long constraint length convolutional codes is a computationally efficient way to communicate at rates lower than the cutoff rate. Based on this insight, we then propose a novel family of concatenation codes called Conv-polar codes which use polar codes as inner codes and binary convolutional codes as outer codes [49]. The frame error rate of Conv-polar codes is shown to decay exponentially with the code length for all rates up to the capacity with SC decoding for the inner polar code and sequential decoding for the outer convolutional codes. Indeed, it has been known for a long time (for e.g. from [50]) that the performance of sequential decoding of convolutional codes can be improved by using an inner code. Our results show that a polar code is indeed an optimal inner code since it increases the cutoff rate all the way up to the capacity of the channel.

- We propose a soft-output multistage iterative decoding method for Conv-polar codes that combines the benefits of soft-output decoding and multistage decoding. This decoding algorithm avoids making hard decisions too early and prevents error propagation. Simulation results show that for the Conv-polar codes, the iterative decoding can provide roughly 0.2 dB gain over SC decoding with only 2 iterations. Some methods to further tradeoff performance for complexity are also discussed.

- Finally, we provide a comprehensive comparison of the performance, memory requirement, and decoding complexity of proposed Conv-polar codes with those of stand alone polar codes. Simulation results show that the proposed approach substantially outperforms stand-alone polar codes under SC or BP decoding and is even comparable to that of list decoding in the high SNR regime. Polar codes with list decoding may require prohibitively large memory size, while Conv-polar codes provide performance gain with much lower memory size. This suggests that using the

proposed soft-output multistage iterative decoding for concatenated polar codes with interleaving/deinterleaving may have a better balance compared to existing methods. Generally speaking, our scheme is advantageous in the relatively high SNR regime and is particularly suitable for applications with stringent memory size constraint. Data storage may be a good application.

3.2 Interleaved concatenation schemes of polar codes with outer binary codes

3.2.1 Interleaved structure and encoding

A schematic of the interleaved concatenated scheme is shown in Fig. 3.1. The figure shows an $m \times n$ array which represents the coded bits of the outer code. The ith column in the array represents the codeword of a (m, k_i) binary block or terminated convolutional code with rate R_{oi}, $i \in [1, n]$. This column is formed by simply encoding k_i information bits into m coded bits. $R_{oi} = 0$ if there's no outer code in the column. The (j, i)-th entry in the array, namely $u_{j,i}$ represents the jth coded bit in the ith code, $i \in [1, n], j \in [1, m]$. Each row in the matrix represents the input bits to a (n, k) polar code and each row is independently encoded by a polar code and the encoded bits form the coded bits of the overall code. All the polar codes are identical and hence, the frozen bits in all the polar codes belong to some $n - k$ columns which are shown in the figure. These columns can also be thought of as codewords of a zero rate column code. The columns in gray and white correspond to the information and frozen bits of the polar codes, respectively. This results in an overall $(mn, \sum_{i=1}^{n} \lfloor mR_{oi} \rfloor)$ linear block code. To further reduce error propagation, random interleavers may be added to the outer codes. We will assume the received vector is also arranged in an $m \times n$ array and we use $y_{j,i}$ to denote the output of the channel corresponding to the transmission of the coded bit $x_{j,i}$.

3.2.2 Multistage successive cancellation decoder

The m polar codes are decoded in parallel and *a posteriori* probabilities are produced for the bits in the first column in Fig. 3.1. Then, the decoding of the first outer code takes place. The decisions from the decoder of the outer code are used in producing the *a posteriori* probabilities for all the bits in the second column and so on. In general, when *a posteriori* probabilities are produced from the polar decoder for the bits in the ith column, the decisions from the outer decoder for the bits in the first $i-1$ columns are available. Decoding of the outer code ensures that these decisions are highly reliable, thereby reducing the error propagation substantially.

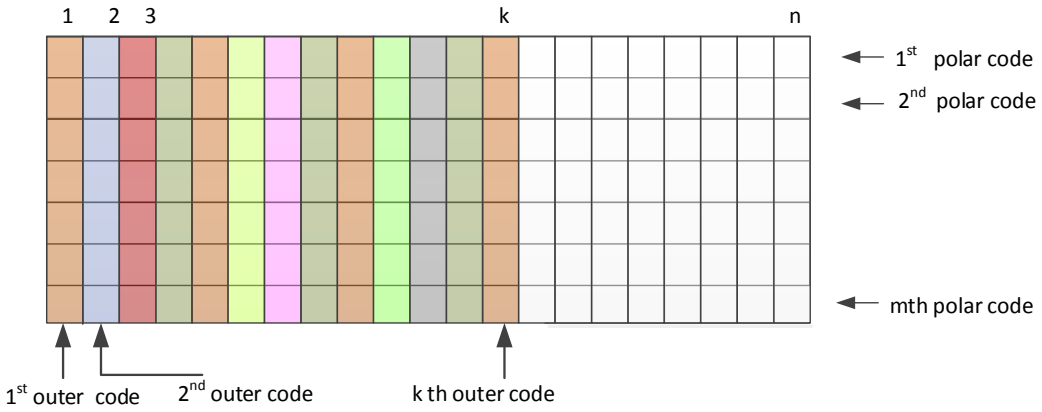


Figure 3.1: Concatenation of polar codes with binary codes

3.2.3 Rate optimization scheme

In this section, we consider the issue of how to choose the rates R_{oi} for each column based on the equal block error rate design rule. Let $u_{j,1}^{j,i}$ and $y_{j,1}^{j,i}$ denote the uncoded and

received bits in the jth row indexed from 1 to i. Since the m polar codes are completely independent of each other, the equivalent channel between the jth bit in the ith column, $u_{j,i}$ and the received vector $\underline{y}$ depends only on $y_{j,1}^{j,n}$ and hence, let $W_n^{(i)}(y_{j,1}^{j,n}, u_{j,1}^{j,i-1}|u_{j,i})$ denote this equivalent channel. Since all the polar codes used are identical, this equivalent channel is independent of j and hence we can drop the dependence of j and with an abuse of notation write this as $W_n^{(i)}(y_1^n, u_1^{i-1}|u_i)$. This is simply the ith bit channel in the polar code.

The block error rate P_{Bi} of each outer code depends on the specific codes used, the outer code rate R_{oi} and the equivalent channel $W_n^{(i)}(y_1^n, u_1^{i-1}|u_i)$. Here, we will assume that the outer codes are chosen from a family of codes such as BCH or convolutional codes. If we use $P_B(N, R)$ to represent the frame error rate of the overall concatenated code with rate R, then $P_B(N, R)$ can be computed as

$$P_B(N, R) = 1 - \prod_{i=1}^{n}(1 - P_{Bi})$$

Let P^* denote the target block error rate of the outer codes. If we set equal target block error rates for all the outer codes, to maximize the overall rate, R_{oi} is chosen to be the maximum value satisfying the condition that $P_{Bi} \leq P^*$. Since there are k outer codes, $P_B(N, R)$ is bounded by

$$P_B(N, R) \leq 1 - (1 - P^*)^k$$

The overall rate R of the concatenated code is

$$R = \frac{\sum_{i=1}^{n} \lfloor R_{oi} \cdot m \rfloor}{mn}$$

To design codes with a fixed overall rate, the rates of inner and outer codes should be optimized to achieve the minimum frame error rate. The rate optimization algorithm is

shown in [51].

3.3 BCH-polar codes

In this section, we consider binary BCH codes as the outer codes. BCH codes are a large class of binary codes within which we can find a t-error correcting code of length m for $0 \leq t \leq \lfloor \frac{m-1}{2} \rfloor$. We first consider Bounded Distance (BD) decoding of BCH codes.

3.3.1 BD decoding over a general channel

Assume that the ith outer code is t_i-error correcting code of rate R_{oi}. p_i is the error probability of the ith bit channel of polar codes. Then P_{Bi} is given by

$$P_{Bi} = \sum_{j=t_i+1}^{m} \binom{m}{j} p_i^j \cdot (1-p_i)^{(m-j)}, i = 1, \cdots, n.$$

It is well known that the rate of a t_i-error correcting BCH code of length $m = 2^l - 1$ can be lower bounded as follows: [52]

$$R_{oi} \geq 1 - \frac{t_i l}{m}$$

3.3.1.1 Error probability analysis

The following theorem shows that the frame error rate of BCH-polar codes when decoded with an outer bounded distance decoder decays exponentially with the code length.

Theorem 7. For any discrete memoryless channel with capacity C, and for rate $R = C(1-\delta)$, for any $0 < \delta < 1$, there exists a sequence of BCH-polar codes of rate R indexed by the block length N such that

$$\lim_{N\to\infty} P_B(N, R) \leq N^{\epsilon} 2^{-N^{1-\epsilon}}, \text{ for any } 0 < \epsilon < 1$$

Proof. This proof follows the steps from [46]. First, for any $0 < \epsilon < 1$ we let the inner polar code length $n = N^{\epsilon}$ and the length of the outer code $m = N^{1-\epsilon}$. Assume that the inner polar code rate is $R_{in} < C$. Then, $n \to \infty$ as $N \to \infty$, the block error rate P_{BI} of the polar code satisfies $P_{BI} \leq 2^{-n^{0.5-\epsilon'}}$ for any $\epsilon' > 0$ as $N \to \infty$ [14]. Let k be the number of information bits of the polar code. As the error rate of each bit channel satisfies $p_i \leq P_{BI}$ for all $i = 1, \cdots, n$, we have $p_i \leq 2^{-n^{0.5-\epsilon'}}$. If bounded distance decoding is used for BCH codes, we have $t_i \geq \frac{(1-R_{oi})m}{\log_2(m+1)}$ from (3.3.1). Hence the block error rate of the i-th outer code is

$$\begin{aligned} P_{Bi} &= \sum_{j=t_i+1}^{m} \binom{m}{j} p_i^j (1-p_i)^{m-j} \\ &\leq \binom{m}{t_i+1} p_i^{t_i+1} \\ &\leq \binom{m}{t_i+1} 2^{-n^{0.5-\epsilon'} \frac{(1-R_{oi})m}{\log_2(m+1)}} \end{aligned}$$

The term $\binom{m}{t_i+1}$ can be bounded by Stirling's approximation

$$\binom{m}{t_i+1} \leq 2^{mH(\frac{t_i+1}{m}) - \frac{1}{2}\log_2 m + O(1)}$$

P_{Bi} is further bounded as

$$\begin{aligned} P_{Bi} &\leq 2^{-m[n^{0.5-\epsilon'} \frac{1-R_{oi}}{\log_2(m+1)} + H(\frac{1-R_{oi}}{\log_2(m+1)} + \frac{1}{m})]} \\ &= 2^{-N^{1-\epsilon}\left[N^{\epsilon(0.5-\epsilon')} \frac{1-R_{oi}}{\log_2(N^{1-\epsilon}+1)} + H\left(\frac{1-R_{oi}}{\log_2(N^{1-\epsilon}+1)} + \frac{1}{N^{1-\epsilon}}\right)\right]} \end{aligned} \tag{3.1}$$

For any R_{oi} satisfying

$$N^{\epsilon(0.5-\epsilon')} \frac{1-R_{oi}}{\log_2(N^{1-\epsilon}+1)} + H\left(\frac{1-R_{oi}}{\log_2(N^{1-\epsilon}+1)} + \frac{1}{N^{1-\epsilon}}\right) \geq 1,$$

from (3.1) we get an upper bound on P_{Bi} given by

$$P_{Bi} \leq 2^{-N^{1-\epsilon}}$$

The frame error rate $P_B(N, R)$ is bounded as

$$P_B(N, R) \leq kP_{Bi} \leq n2^{-N^{1-\epsilon}} = N^{\epsilon}2^{-N^{1-\epsilon}}$$

which indicates $P_B(N, R) \to 0$ as $N \to \infty$. The overall rate of the code R' is

$$\begin{aligned} R' &= \frac{1}{N}\sum_{i=1}^{k}\lfloor mR_{oi}\rfloor = \frac{1}{N}\lfloor nR_{in}\rfloor \cdot \lfloor mR_{oi}\rfloor \\ &= R_{in}R_{oi} + \mathrm{o}(1) \end{aligned}$$

To make the overall rate $R' = R$, we choose R_{in} to be

$$R_{in} = \frac{C(1-\delta) + \mathrm{o}(1)}{R_{oi}}$$

Thus we have designed the code with overall rate R with the frame error rate upper bounded by $N^{\epsilon}2^{-N^{1-\epsilon}}$ when N is large enough. □

3.3.2 Tradeoff between performance and complexity for the BEC

In the previous section, we considered the use of a bounded distance decoder for the outer code and the performance of the concatenated scheme can be improved by considering ML decoding of the outer codes. Even though the complexity of ML decoding of arbitrary outer codes can be exponential in m, for the binary erasure channel, ML decoding is equivalent to a matrix inversion whose complexity is only $O(m^3)$.

For a length-N polar code, the decoding complexity is $O(N \log N)$ and the block error

rate is bounded by $2^{-N^{0.5-\epsilon}}$. If we set the inner code length $n = N^{\epsilon}$ and the outer code length $m = N^{1-\epsilon}$ for any $0 < \epsilon < 1$, from Theorem 1 we know that the concatenated BCH-polar code is able to boost the frame error rate to be $P_B(N, R) \leq N^{\epsilon} 2^{-N^{1-\epsilon}}$, which is a substantial improvement over the stand-alone polar code. The decoding complexity T for the BCH-polar code is $O(mn \log n + nm^3)$, where $O(mn \log n)$ is the complexity of decoding the inner polar codes, and $O(nm^3)$ is the complexity of decoding the outer BCH codes. T can be further derived as

$$\begin{aligned} T &= O(N \log N^{\epsilon} + N^{\epsilon} N^{3(1-\epsilon)}) \\ &= O(\epsilon N \log N + N^{3-2\epsilon}) \end{aligned}$$

The decoding complexity decreases while the frame error rate increases as ϵ increases. Thus we can have a balance between the performance and complexity by choosing ϵ appropriately.

3.4 Proposed Conv-Polar Codes

In this section, we consider the case when the outer codes in the interleaved concatenated scheme are convolutional codes. Assume R_{ci} is the cutoff rate of the i-th bit channel. The ith convolutional code is of rate $R_{oi} < R_{ci}$ and constraint length ν. We refer to these codes as Conv-polar codes. The decoder is a multistage SC decoder which uses m parallel polar successive cancellation decoders and sequential decoders or a Viterbi decoders for decoding the outer convolutional codes.

3.4.1 Error probability analysis of Conv-polar codes

We will first construct a sequence of codes for which $n = N^{\epsilon}, m = N^{1-\epsilon}$ and $\nu = \alpha m$ for an arbitrarily small constant α. For these parameters, we will first show that Conv-polar codes can be used to obtain a frame error rate that decays exponentially with N with

both multistage Viterbi decoding and multistage sequential decoding.

Theorem 8. Let N be the code length and C be the capacity of the channel. There exists a Conv-polar code with constraint length $\nu = \alpha m$, where α is a constant and m is the length of the convolutional code, such that for any discrete memoryless channel and for any overall rate $R < C$, the frame error rate of the code $P_B(N, R)$ is upper bounded by $KN\mathrm{e}^{-k'\rho\alpha N^{1-\epsilon}}$ for some constants K and k' and for any $0 < \epsilon \leq 1$ and large enough N under either Viterbi decoding or sequential decoding. Here, $0 \leq \rho \leq 1$ if Viterbi decoding is used and $0 \leq \rho < 1$ if sequential decoding is adopted.

Proof. The average bit error rate of the ensemble of (n_o, k_o, ν) convolutional code be bounded as follows [53]

$$P_b \leq K\mathrm{e}^{-n_o\nu\rho R_c} \tag{3.2}$$

where K is a constant, $0 \leq \rho \leq 1$ and R_c is the cutoff rate. (3.2) holds if Viterbi decoding is used; if sequential decoding is used, we set the code rate $R_o < R_c$, then (3.2) holds with $0 \leq \rho < 1$. The block error rate of the ensemble is $P_B \leq mP_b$. For each equivalent bit channel, the cutoff rate R_{ci} can be computed based on (2.4). Thus, there always exists at least one (n_{oi}, k_{oi}, ν) convolutional code with the bit error rate for the i-th channel $P_{bi} \leq K\mathrm{e}^{-n_{oi}\nu\rho R_{ci}}$. Then $P_B(N, R)$ can be bounded as

$$\begin{aligned} P_B(N, R) &\leq \sum_{i=1}^{n} P_{Bi} \leq \sum_{i=1}^{n} Km\mathrm{e}^{-n_{oi}\nu\rho R_{ci}} \\ &\leq Kmn\mathrm{e}^{-k'\rho\nu} \end{aligned}$$

where $k' = \min_i\{n_{oi}R_{ci}\}$, P_{Bi} is the block error rate of the ith outer code and n is the inner code length. If we pick $n = N^{\epsilon}$, $m = N^{1-\epsilon}$, $P_B(N, R)$ can be further bounded as

$$P_B(N, R) \leq KN\mathrm{e}^{-k'\rho\alpha N^{1-\epsilon}}$$

As n increases, $\frac{1}{n}\sum_{i=1}^{n} R_{ci} \to C$. For sequential decoding, we choose $R_{oi} = R_{ci} - \Delta$ and we note that there is a rate loss associated with terminating the convolutional codes. The rate achievable is then $\frac{1}{n}\sum_{i=1}^{n} R_{ci}(1 - \frac{\alpha m}{m}) - \Delta$. By choosing α and Δ to be arbitrarily small any rate $C - \delta$ can be obtained. □

Remark 9. Notice that in the above theorem, we have proved that an exponential error decay rate can be achieved for the proposed Conv-polar code under either Viterbi decoding or sequential decoding, when $\nu = \alpha m$. However, our primary interest of this theorem is in the one with sequential decoding as Viterbi decoding with a large constraint length (linear in m) is typically forbidden in practice due to its large complexity. We have shown in this theorem that the proposed Conv-polar codes efficiently exploit the increased cutoff rate due to channel polarization and hence allow sequential decoding to have bounded complexity while enjoying an exponential error decay rate. On the other hand, when Viterbi decoding is used, one can trade off the error rate for decoding complexity by choosing the constraint length to grow with the length at different rates. The following theorem makes this precise.

Theorem 10. Consider a sequence of Conv-polar codes with inner polar code-length n equal to N^{ϵ} for any $0 < \epsilon < 1$. Assume (n_{oi}, k_{oi}, ν) is the i-th outer code and R_{ci} is the cutoff rate of the i-th channel. Let the constraint length $\nu = \alpha \ln m$, where m is the outer code length, $\alpha > \frac{1}{(1-\epsilon)h}$ and $h = \min_i\{n_{oi}R_{ci}\}$. The frame error rate of the code $P_B(R, N)$ is upper bounded by $KN^{\epsilon-(1-\epsilon)(h\alpha-1)}$ where K is a constant. Meanwhile, the decoding complexity is proportional to $N^{\epsilon+(1-\epsilon)\alpha \ln 2}$ if the Viterbi decoder is adopted.

Proof. With Viterbi decoding we can set $\rho = 1$ in (3.2). By following the proof of Theorem 2 we can bound $P_B(R, N)$ by

$$P_B(R, N) \leq Kmn\mathrm{e}^{-h\nu}$$

where $h = \min_i\{n_{oi}R_{ci}\}$. By replacing ν by $\alpha \ln m$, and picking $n = N^{\epsilon}$, $m = N^{1-\epsilon}$, we can easily prove the upper bound of $P_B(R, N)$. The complexity of Viterbi decoding for each code is proportional to 2^{ν}, which is further derived as $2^{\nu} = 2^{\alpha \ln m} = m^{\alpha \ln 2}$. Thus the overall decoding complexity of outer codes is proportional to $nm^{\alpha \ln 2} = N^{\epsilon+(1-\epsilon)\alpha \ln 2}$. $\square$

3.4.2 Convolutional codes with multistage Viterbi decoding

When the constraint lengths of the convolutional codes are not too large, optimal Viterbi decoding can be used to decode the codes. For finite lengths, the multistage Viterbi decoder with moderate constraint length codes may outperform sequential decoding with long constraint length convolutional codes and we now consider the design of convolutional codes for finite lengths.

3.4.2.1 Design of Conv-polar codes for finite lengths

The bit error rate P_b at the output of the Viterbi decoder for a convolutional code in an AWGN channel is bounded by the following:

$$P_b \leq \sum_{d=d_{free}}^{\infty} a_d Q\left(\sqrt{\frac{2dE_s}{N_0}}\right) \tag{3.3}$$

where d_{free} is the free distance and a_d is the weight distribution of the code. E_s/N_0 is the signal to noise ratio. $Q(x)$ is defined as $Q(x) = \int_x^{\infty} \frac{1}{\sqrt{2\pi}} e^{-t^2/2} dt$. If the convolutional code is terminated to an (m, k) block code, the block error rate of the ith outer code P_{Bi} is

$$P_{Bi} \leq kP_{bi}$$

where P_{bi} is the bit error rate of the i-th outer code. Since we're able to estimate the error rate of outer codes from the weight distribution of the codes, the rates of inner and outer codes can be optimized based on the equal block error rate design rule. Convolutional

codes with a wide range of rates are provided in [54] and [55].

If the convolutional code is terminated, additional ν bits should be appended to drive the encoder to the zero state. This will result in rate loss. The actual rate then becomes $\frac{k}{m}(1-\frac{\nu}{k})$. There are several ways to avoid the rate loss. Tail biting is a good way with a slight degradation in the performance [56]. When decoding the outer codes, we use soft input for Viterbi decoding, which greatly outperforms hard input [57]. The SC decoding of inner polar codes naturally gives out the LLRs of all the bits, and we can take them as soft input for Viterbi decoding.

3.4.2.2 Methods to decrease the decoding complexity

As the constraint length or the rate of the convolutional code increases, Viterbi decoding would be too complex to be used in practice. There are various ways to decrease the decoding complexity. For the case when the constraint length is not large but the rate is high, the dual algorithm proposed in [54] can be applied. The decoding complexity is roughly the same as decoding the low rate dual code. Another way to alleviate the high complexity problem is to obtain a high-rate code from puncturing a low-rate mother code [55]. A general (n_o, k_o, ν) convolutional code with rate k_o/n_o and constraint length ν, requires $(2^{k_o} \cdot 2^{\nu})/k_o$ operations in terms of addition, comparison and selection per decoded bit [58]. However, with puncturing the branch for each stage remains to be 2, thus the complexity of decoding the punctured code remains to be the same as that of mother codes which is $2 \cdot 2^{\nu}$ per coded bit [58]. Also puncturing provides flexibility in designing a wide range of rates. From a more practical point of view, we propose to use the Rate Compatible Punctured Convolutional (RCPC) codes as outer codes. RCPC codes are obtained by puncturing the same low-rate mother code. A main advantage is that they can use the same encoder and decoder as the mother code [58] which simplifies hardware implementation. Since the mother code is a low-rate code the decoding complexity is rather small.

3.5 Soft-output multistage iterative decoding of Conv-polar codes

In the multistage successive cancellation decoder considered so far, the outer decoder uses a hard output algorithm and it is natural to consider an extension where soft information is passed between the outer decoder and the inner polar decoder. Here we consider a decoder that adopts the same scheduling as the multistage SC decoder, but at each stage a soft output decoder is used for the outer code. In addition, the procedure is iterative and a few iterations are performed between the inner and outer decoders. We call it the soft-output multistage iterative decoder. For outer convolutional codes, we use Bahl, Cocke, Jelinek and Raviv (BCJR) algorithm to obtain soft outputs. For inner polar codes, we use soft SC decoding proposed in [28]. This algorithm pushes the hard decision to the end of the decoding process and hence prevents premature hard decision. The reason that we use the soft decoding in [28] with SC scheduling for decoding polar codes instead of simply applying BP decoding on the whole graph is that the proposed algorithm will give us faster convergence compared to BP decoding. Moreover, the SC scheduling admits a recursive structure and therefore can be exploited to reuse the memory in hardware implementation.

The proposed soft-output multistage iterative decoding (SOMID) algorithm is outlined in Algorithm 1. We use $\underline{x}_j$ and $\underline{y}_j$ to represent the coded and received vector corresponding to the j-th row (jth polar code) in the interleaved structure, $j \leq m$. R_{oi} denotes the outer code rate corresponding to i-th column. The factor graph of a length-n polar code consists of $l+1$ layers where $l = \log n$. We use the same notation as in [28]. Each layer λ has 2^λ groups and each group includes $2^{l-\lambda}$ nodes. Each node in the graph can be indexed by (λ, i, ω), meaning that the node is the ω-th node in the i-th group of the λ-th layer. $0 \leq i < 2^\lambda$, and $0 \leq \omega < 2^{l-\lambda}$. Let B_j and L_j denote the memory space required to store LLRs passing forward and backward in the tanner graph of the jth polar code, and let $L_{j,\lambda}(i,\omega)$ and $B_{j,\lambda}(i,\omega)$ denote the LLRs passed in λ-th layer of the jth polar code,. For

the outer codes, we use L_{conv}^{ap} and L_{conv}^{ext} to represent the a-priori and extrinsic LLRs of the codeword bits. n_{iter} is the maximum number of iterations and σ is the standard deviation of the Gaussian noise. The functions updateB and updateL are to update the LLRs passed forward and backward with soft SC decoding for polar codes. The function convBCJRdec represents BCJR decoding of outer convolutional codes.

Algorithm 1 Soft-output multistage iterative decoder of the Conv-polar codes

Input: $\{\underline{y}_j\}_{j=1}^m, \{R_{oi}\}_{i=1}^n, \sigma, n_{iter}$
Output: $\{\underline{\hat{x}}_j\}_{j=1}^m$

```
for t = 1 → n_iter do
    {L_{j,0}(0,i)}_{i=1}^n ← 2y_{j,i}/σ^2 for all j = 1, ··· , m
    B_{j,l}(i,0)_{i∈F} ← ∞ for all j = 1, ··· , m
    for i = 1 → n do
        L_j ← updateL(n, i, L_j, B_j) for all j = 1, ··· , m
        if R_oi > 0 and R_oi < 1 then
            L_conv^ap ← {L_{j,l}(i,0)}_{j=1}^m
            L_conv^ext ← convBCJRdec(m, L_conv^ap, R_oi)
            {L_{j,l}(i,0)}_{j=1}^m ← L_conv^ext
        end if
        if i is odd then
            B_j ← updateB(n, i, L_j, B_j) for all j = 1, ··· , m
        end if
        if t = n_iter then
            if i ∈ F then
                x̂_{j,i} ← 0 for all j = 1, ··· , m
            else
                if L_{j,0}(0,i) + B_{j,0}(0,i) ≥ 0 then
                    x̂_{j,i} ← 0 for all j = 1, ··· , m
                else x̂_{j,i} ← 1 for all j = 1, ··· , m
                end if
            end if
        end if
    end for
end for
```

3.6 Memory space and decoding complexity analysis of Conv-polar codes

In this section, we analyze and compare the performance, required memory size, and decoding complexity of various designs including polar codes with SC decoding, polar codes with BP decoding, polar codes with list decoding, and Conv-polar codes with both multistage SC decoding and sof-output multistage iterative decoding.

3.6.1 Memory space analysis

We use B and L to denote the memory space required to store the LLRs passed forward and backward in the Tanner graph of polar codes. For a length-N polar code, BP decoding requires $N(\log N+1)$ units for both L and B and hence the total memory is $2N(\log N+1)$. SC decoding requires $2N-1$ units for L and $2(2N-1)$ units for B, since we can reuse space with the recursive structure [20]. Therefore, the total memory for SC decoding is $3(2N-1)$. Soft-output SC decoding in [28] requires $2N-1$ units for L and $4N-2+\frac{N\log N}{2}$ units for B. The total memory for soft-output SC decoding is $6N-3+\frac{N\log N}{2}$. This reduction (for large N) is a consequence of the space reuse in the decoding. For list decoding, the memory space required for L and B are $l(2N-1)$ and $2l(2N-1)$ with careful reuse of memory [20], respectively, where l is the decoding list size. Another $3l(\log N+1)+2l$ memory space is needed for path mapping [20, Alg. 8]. Thus the total memory space for list decoding is $l(6N+3\log N+2)$. Table 3.1 shows the memory space for L and B for polar codes with different decoding methods.

Now, let us discuss the memory space required for the Conv-polar codes. Since m polar codes are decoded in parallel, the memory for the polar codes is $3m(2n-1)$ for SC decoding and it is $m(6n-3+\frac{n\log n}{2})$ for soft-output SC decoding. The outer convolutional codes are decoded in order, which enables us to reuse the memory. The memory for outer codes is the maximum of the memory required for all the outer codes. If Viterbi algorithm is used, the memory space required is $2k'\cdot 2^{\nu}$ [57] where k' is the length of information

bits and ν is the constraint length. This can be upper bounded by $2m \cdot 2^{\nu}$ since $k' \leq m$. If BCJR algorithm is used, the memory space required is $3k' \cdot 2^{\nu}$, which is upper-bounded by $3m \cdot 2^{\nu}$. Then the total memory required for the Conv-polar codes with multistage SC decoding is upper bounded by $3m(2n-1) + 2m \cdot 2^{\nu}$. The total memory required for the Conv-polar codes with soft-output multistage iterative decoding is upper-bounded by $m(6n - 3 + \frac{n \log n}{2}) + 3m \cdot 2^{\nu}$. Further memory reduction is possible by windowed decoding of convolutional codes. The memory required is about 5ν. We do not pursue it in this chapter.

Table 3.2 summarizes the above analysis. Let the memories of Conv-polar and stand-alone polar codes be $M_{Conv-polar}$ and M_{polar}, respectively. In Fig. 3.2, we show the ratio $\alpha = \frac{M_{Conv-polar}}{M_{polar}}$ of the memory space required by Conv-polar codes to that by stand-alone polar codes with BP decoding. The outer code length m is set to 64 or 128. The constraint length ν of outer codes is 8. We observe that for $m = 128$, when multistage SC decoding is used, the memory space required for Conv-polar codes is about 50% less than that of BP decoding when when $N = 2^{13}$, and is 70% less when $N = 2^{14}$. When soft-output multistage iterative decoding is used, the memory space required for Conv-polar codes is about 25% less than that of BP decoding when $N = 2^{13}$, and is 50% less when $N = 2^{14}$. In Fig. 3.3, we show the ratio α of the memory space required for Conv-polar codes to that for stand-alone polar codes with list decoding. The list size l is 2 for polar codes. m is set to 64 and 128. One can see that when $N = 8192$ and $m = 128$ the memory of Conv-polar codes with multistage SC decoding is about 1.1 times that of polar codes, and the memory of Conv-polar codes with SOMID is 1.7 times that of polar codes with list decoding. When N exceeds 2^{15}, the memory of Conv-polar codes with multistage SC decoding is about half that of polar codes, and the memory of Conv-polar codes with SOMID is close to that of polar codes with list decoding. Notice that the memory of polar codes increases linearly with the list size. When the list size is greater than 2, the memory

of polar codes would be substantially larger than that of Conv-polar codes.

Table 3.1: Memory required for different decoders of polar codes

	L	B	Total memory for L and B
SC	$2N-1$	$2(2N-1)$	$3(2N-1)$
BP	$N(\log N+1)$	$N(\log N+1)$	$2N(\log N+1)$
List	$l(2N-1)$	$2l(2N-1)$	$3l(2N-1)$
soft-output SC	$2N-1$	$4N-2+\frac{N\log N}{2}$	$6N-3+\frac{N\log N}{2}$

Table 3.2: Overall memory space required

	Overall memory space
Polar SC decoding	$3(2N-1)$
Polar BP decoding	$2N(\log N+1)$
Polar soft-output SC decoding	$6N-3+\frac{N\log N}{2}$
Polar List decoding	$l(6N+3\log N+2)$
Conv-polar multistage SC decoding	$3m(2n-1)+2m\cdot 2^{\nu}$
Conv-polar SOMID	$m(6n-3+\frac{n\log n}{2})+3m\cdot 2^{\nu}$

3.6.2 Decoding complexity analysis

It is well-known that the time complexity of polar codes with SC decoding is $O(N\log N)$. The complexity for BP decoding is $O(n_{iter}N\log N)$ [3], where n_{iter} is the number of iterations. Typically, in order to have a reasonable performance, polar codes with BP decoding requires a large number of iterations. For list decoding, the decoding complexity is $O(lN\log N)$ [20]. For RS-polar codes in [46], the decoding method is generalized minimum distance (GMD) decoding. The candidate in the list of codewords which is closest to the received word is picked. This method is denoted as GMD-ML decoding. The complexity of RS-polar codes is $O(N\log n+N\log^2 m\log\log m)$ [46].

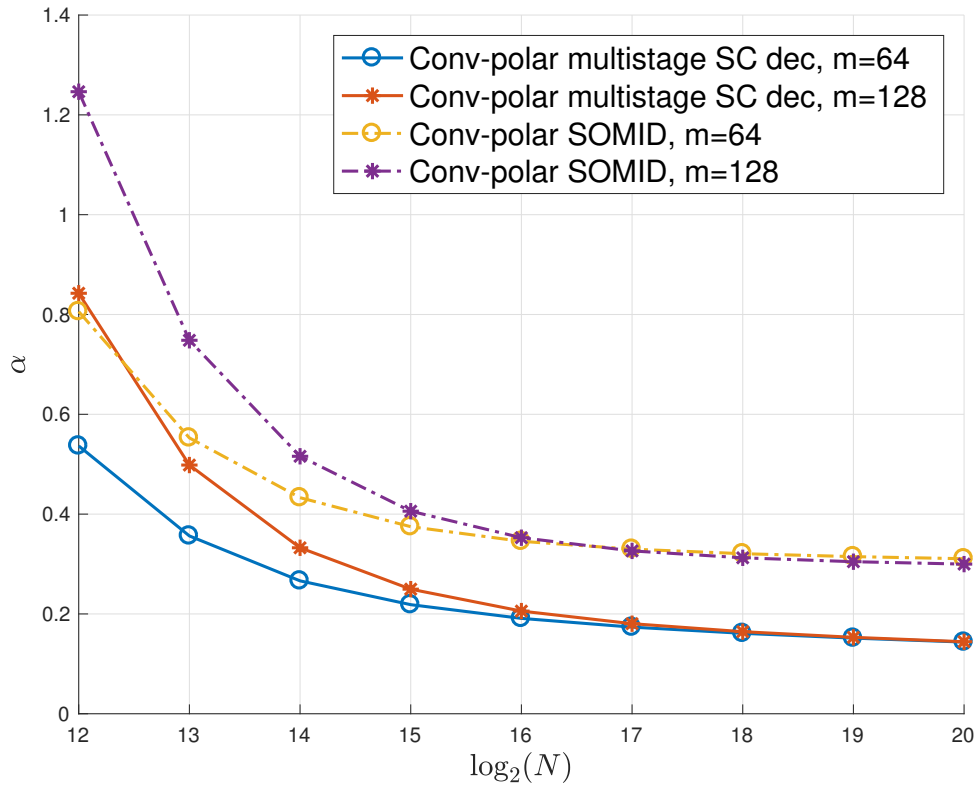


Figure 3.2: The ratio α of memory space of Conv-polar codes to that of stand-alone polar codes with BP decoding

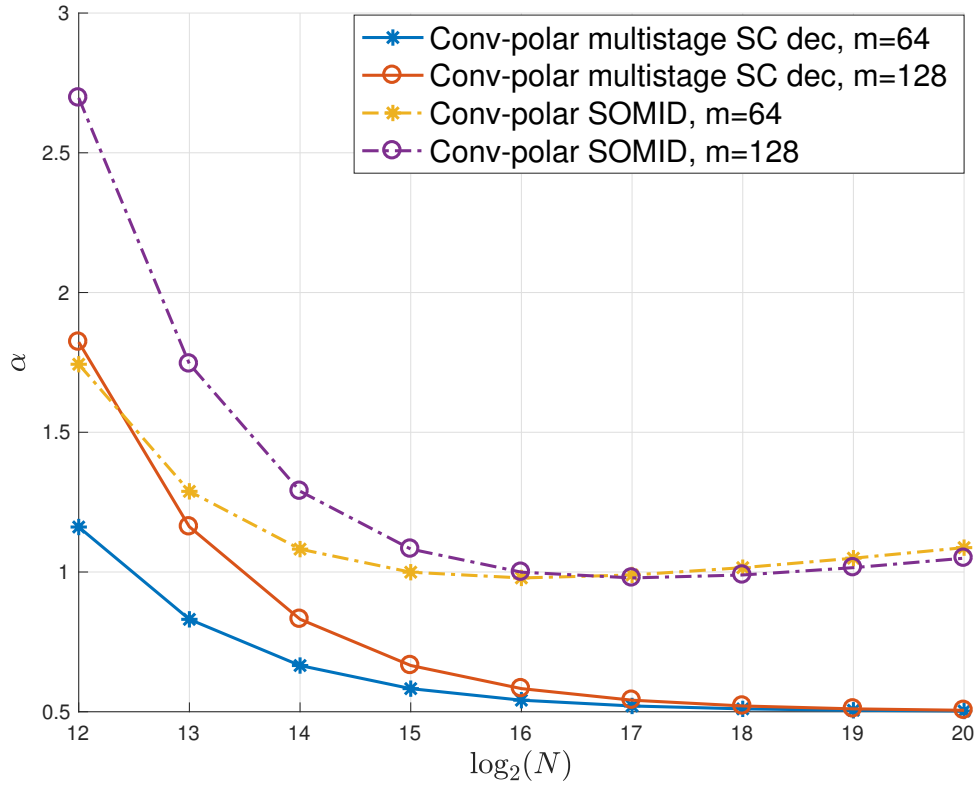


Figure 3.3: The ratio α of memory space of Conv-polar codes to that of stand-alone polar codes with list decoding ($l = 2$)

We now analyze the decoding complexity of Conv-polar codes. For the inner polar codes, the decoding complexity is $O(mn \log n)$ per iteration. The complexity of using Viterbi algorithm to decode the outer (n_o, k_o, ν) convolutional codes is $O(\sum_{i=1}^{n} k_i 2^{k_o} 2^{\nu}) = O(K 2^{k_o} 2^{\nu})$, where k_i is the number of information bits in the ith column and K is the total information length of the concatenated code. The complexity of BCJR decoding is roughly twice as much of Viterbi decoding. Notice that as mentioned above, we benefit from using punctured codes and get high rate codes by puncturing a low-rate mother code; therefore, we can always set $k_o = 1$ by considering rate $1/n_o$ mother codes. This would not affect the asymptotic result but would have non-negligible impact for finite length codes. The overall complexity is $O(mn \log n + K 2^{k_o} 2^{\nu}))$ for multistage SC decoding, and is $O(n'_{iter}(mn \log n + 2K 2^{k_o} 2^{\nu}))$ for soft-output multistage iterative decoding, where n'_{iter} is the number of iterations. It is worth mentioning that unlike BP decoding which typically requires a large number of iterations, the simulation results shown in the following section suggest that a few iterations suffice to provide improvement. Also, when N or l is large, the decoding complexity of Conv-polar codes can be much smaller than stand-alone polar codes with list decoding. Table 3.3 summarizes the above analysis.

Table 3.3: Decoding complexity of different schemes

	Decoding complexity
Polar SC decoding	$O(N \log N)$
Polar BP decoding	$O(n_{iter} N \log N)$
Polar List decoding	$O(lN \log N)$
RS-polar GMD-ML decoding [46]	$O(N \log n + N \log^2 m \log \log m)$
Conv-polar multistage SC decoding	$O(N \log n + K \cdot 2^{k_o} \cdot 2^{\nu})$
Conv-polar SOMID	$O(n'_{iter}(N \log n + 2K \cdot 2^{k_o} \cdot 2^{\nu}))$

3.7 Simulation results

3.7.1 Performance of BCH-polar codes

We design BCH-polar codes for the BEC, with BD decoding and ML decoding for the BCH outer codes. The overall rate of the code is 0.4. The code length N is 1024, $n = 8$ and $m = 128$. The rates of the outer codes are optimized for BEC(0.4). We use the Extended BCH codes and the weight distributions for the BCH codes are obtained from [59]. The outer code rates are $0, 0, 0, 71/128, 8/128, 99/128, 113/128, 120/128$. Fig. 3.4 plots the frame error rate of BCH-polar codes and stand-alone polar codes with both SC and BP decoding over the BEC as a function of the erasure probability. The frame error rate for RS-polar codes with the same code length and rate is also plotted. The extended outer RS codes are designed over F_{2^4}. $n = 16$ and $m = 64$. RS-polar codes are optimized using the methods in [46]. It can be seen that BCH-polar codes outperform RS-polar codes with BD decoding. While BCH-polar codes with BD decoding are inferior to the stand-alone polar codes, BCH-polar codes with ML decoding significantly outperform polar codes with SC or BP decoding. As discussed earlier, the length of the BCH codes is only $1/n$-th the overall length and hence the ML decoding complexity is substantially smaller than ML decoding of a single BCH code of the overall length.

3.7.2 Performance of Conv-polar codes

In this section, we show simulation results to compare the finite-length performance of Conv-polar codes over the AWGN channel with BPSK modulation. Multistage iterative decoding are considered. Punctured codes are used to reduce the decoding complexity. We use terminated convolutional codes of rate ranging from 1/8 to 7/8. The convolutional codes are chosen from [60] and [61]. For the very good equivalent bit channels, we use high-rate BCH codes with 1 or 2 error correction ability or single parity check codes in those channels, which provides improvement in the high SNR regime.

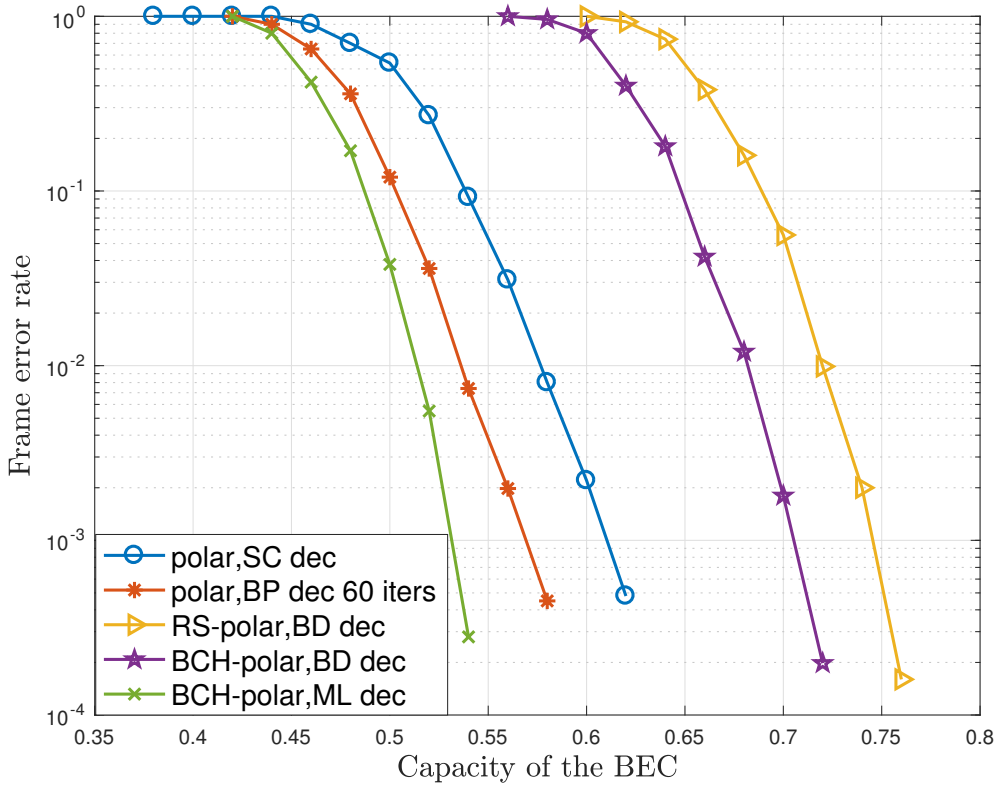


Figure 3.4: FER performance of codes with $N = 1024$ and rate 0.4 over BEC

In Fig. 3.5 we compare the frame error rate performance of the Conv-polar codes with multistage SC decoding and soft-output multistage iterative decoding, stand-alone polar codes with SC, BP and list decoding, polar-CRC codes with list decoding. The performance of concatenated codes with $\nu = 13$ and sequential decoding is also shown. The overall block length is $N = 2048$ and overall rate is $1/2$. The target E_b/N_0 that we use to construct codes is 2 dB. For the concatenated Conv-polar codes, we split N into $n = 8$ and $m = 256$. The iteration number is set to 60 in BP decoding and the list size of list decoding is 32. For polar-CRC codes, the number of CRC bits is 16 and the list size is 32. One can see from this figure that the Conv-polar codes outperform stand-alone polar codes and their performance can be better than that of list decoding at high SNR region. Soft-ouput multistage iterative decoding provides a gain of about 0.2 dB with 2 iterations. Further increasing the iteration number doesn't provide much gain. When convolutional codes with larger constraint lengths are used, the performance of Conv-polar codes with sequential decoding would be much better than that of list decoding at high SNR region. In all cases however, list decoding of polar codes with CRC provides better performance.

To compare Conv-polar codes with RS-polar codes, we construct Conv-polar codes with code length $N = 8192$ and rate $1/3$. $n = 64$ and $m = 128$. Fig. 3.6 shows the frame error rate of Conv-polar codes along with other codes. The performance of RS-polar codes shown in the figure is from Fig. 5 of [46]. It can be observed that while RS-polar codes do not outperform standalone polar codes, Conv-polar codes significantly outperform polar codes. There's 0.2 dB gain for the second iteration. Fig. 3.7 shows the frame error rate of Conv-polar codes with length $N = 16384$ and rate $1/2$. $n = 128$ and $m = 128$. We choose the best RS-polar code among different lengths of inner and outer codes in Fig. 6 of [46] as a comparison. Again the result shows the superiority of Conv-polar codes over RS-polar codes.

From the above we can conclude that the proposed Conv-polar codes outperform SC

and BP decoding of stand-alone polar codes, while the memory space required is much less than that of polar codes with BP decoding. List decoding provides performance gain at the cost of much larger memory size, as seen from Fig. 3.3. RS-polar codes achieve exponential error decay rate as Conv-polar codes but their performance is inferior to Conv-polar codes. The performance of Conv-polar codes can be improved by increasing the constraint length, while sequential decoding is able to maintain low decoding complexity. Thus we can conclude that Conv-polar codes strike a better balance than the existing schemes.

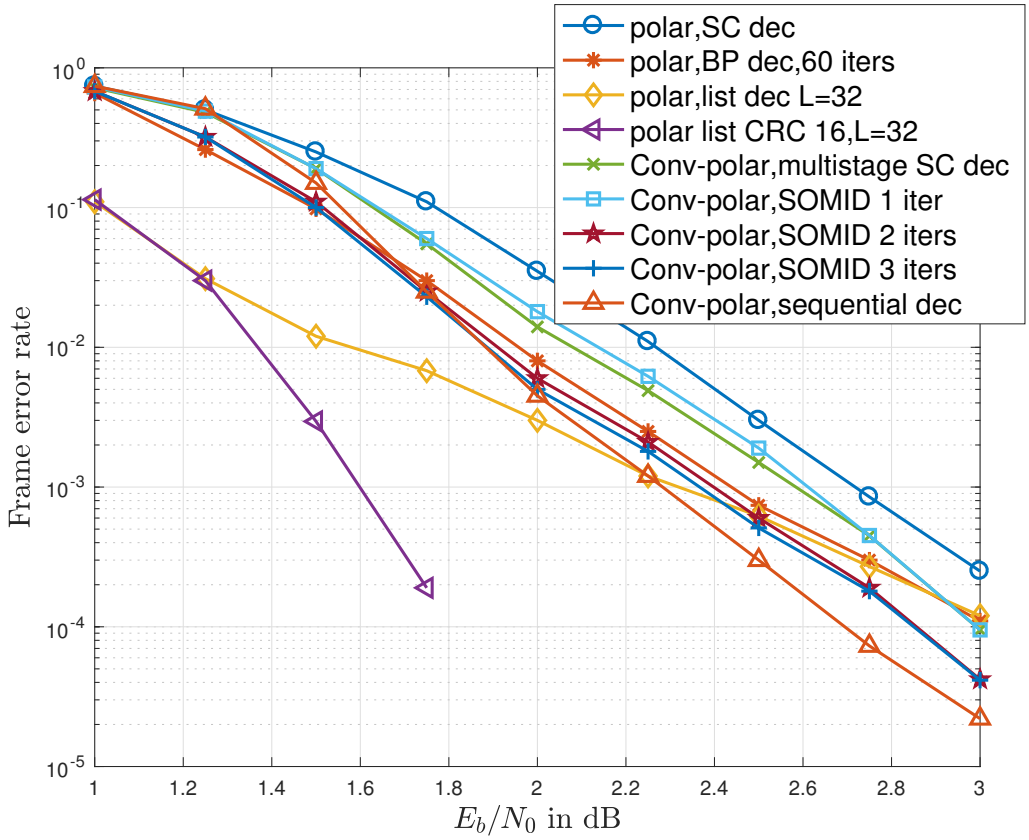


Figure 3.5: FER performance of codes with $N = 2048$ and rate 1/2 codes over AWGN channel.

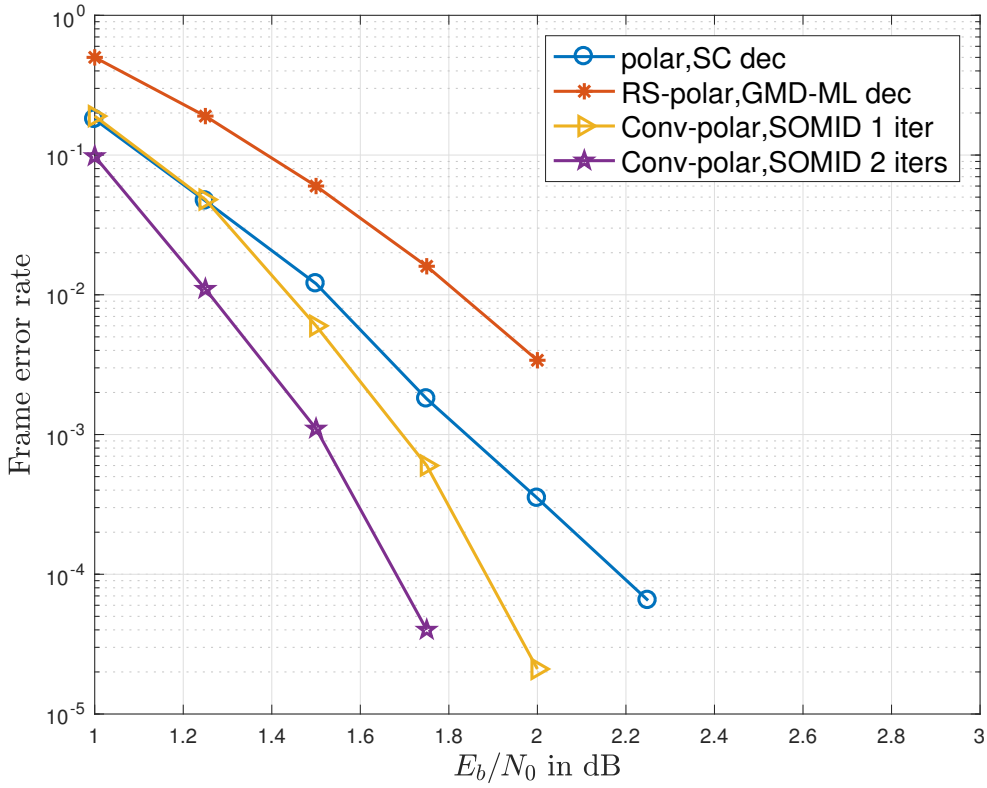


Figure 3.6: FER performance of codes with $N = 8192$ and rate 1/3 over AWGN channel.

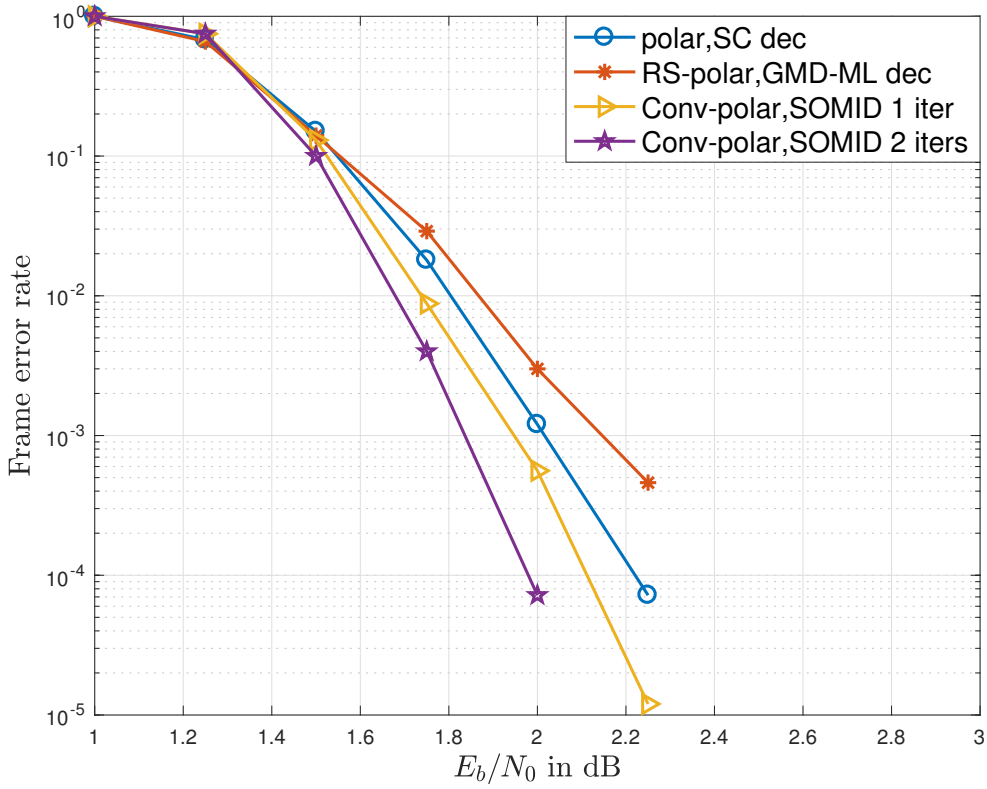


Figure 3.7: FER performance of codes with $N = 16384$ and rate 1/2 over AWGN channel.

3.8 Conclusion

Interleaved concatenation schemes of polar codes with binary codes have been considered. We have reviewed BCH-polar codes and have shown that the frame error rate can decay exponentially with the code length for all rates up to capacity. We have also shown that BCH-polar codes outperform RS-polar codes for the binary erasure channel. We have then proposed a new class of concatenation schemes, namely the Conv-polar codes. We have shown that such schemes also have an exponentially decaying error rate with a low-complexity sequential decoder for all rates up to capacity. Furthermore, we have proposed a soft-output multistage iterative decoder for the concatenated Conv-polar codes to further improve the performance. From a more practical point of view, we have proposed using RCPC codes as outer codes which enable the same encoder and decoder for outer codes and are more hardware friendly. We have also investigated polar codes with various decoders in many practical aspects including error-correcting capability, required memory space, and decoding complexity. Analysis and simulation results have shown that the Conv-polar codes with the proposed decoders outperform stand-alone polar codes with SC and BP decoding , while having much smaller required memory size than BP decoding. Conv-polar codes also outperform RS-polar codes. Further performance improvement can be obtained by the use of outer codes with much larger constraint length and sequential decoding. This has suggested that the Conv-polar codes with the proposed decoding may strike a better balance than existing methods.

4. JOINT SOURCE-CHANNEL DECODING OF POLAR CODES FOR LANGUAGE BASED SOURCES

4.1 Introduction

In this chapter, we study how to improve the performance of polar codes using the *natural redundancy* in data. By natural redundancy, we refer to the inherent redundancy in data (e.g., features in languages, images and videos) that is not artificially added for error correction. We focus on compressed languages here. Current works have shown that after compression of texts (at a compression ratio higher than practical systems), lots of natural redundancy still exists [62]. Shannon has estimated the English entropy to be 11.82 bits/word or 1.34 bits/character [63]. With Huffman coding for characters, the source after compression has on average 37 bits/word or 4.59 bits/character. With Lempel-Ziv-Welch (LZW) coding with a dictionary of 2^{20} patterns (much larger than LZW dictionaries in many practical systems), the average length per character after source compression is 2.94 bits [62]. We can observe that it is difficult for the standard source compression schemes to approach the Shannon estimation. There are also significant amount of redundancy brought by various backup devices or cloud storage. By exploiting the natural redundancy, the performance of error correction codes can be substantially improved [64, 1, 65, 66, 67, 68, 69, 70]. For example, after English texts are compressed by a LZW code and transmitted over a BEC, a decoding algorithm using only natural redundancy can reduce the noise in the compressed texts by over 85% for channel erasure rates from 5% to 30%.

To better understand how natural redundancy is efficient in improving the decoding performance, we show an example of decoding a piece of English text. Assume the text

is encoded by Huffman code into a length-n binary sequence and stored. As time goes errors occur in the stored data. Assume that the number of bit errors is at most t. If we use exhaustive search to find all possible solutions, the number of all solutions is $\sum_{i=0}^{t} \binom{n}{i}$. However valid solutions are very sparse. By valid solution we mean that the solution satisfies two properties: (1) the bit string can be mapped successfully to a character string by Huffman code; (2) all words in the character string appear enough number of times in Wikipedia. Table 4.1 shows an example of decoding a corrupted version of text “fundamental problem". We see that there are only 6 valid solutions, and it is easy to pick the correct one from valid solutions.

Table 4.1: An example of how the natural redundancy helps with error correction

Number of bits	Number of errors	Number of all solutions	Number of valid solutions	Valid solutions
92	2	3547	6	(1) fundamental proc ira (2) fundamental proc-m (3) fundamental problem (4) fundamental pro w ra (5) fundamental pr ch ra (6) fundamental t9ch ra

This work is an extension of JSCD and denoising, which have been studied extensively. However it has distinctive features. First, it does not consider joint optimization between source compression and error correction. It assumes that the codes for source compression are given and fixed. The reason is that typically the encoder needs to apply source and channel encoding to data following a standard and the encoded data are sent to different decoders. However, each decoder has its own freedom to develop advanced algorithms to decode data. Second, our work explores new features in big data. One example is the co-location relationship in languages shown in [62]. The natural redundancy acts as global

constraints.

Shannon's theorem [71] shows that separate optimization of source and channel codes suffices for communicating sources over a large class of channels. However, such a separation-based scheme is often subject to impractical computational complexity and unlimited delay. It is well known that JSCD can outperform separation-based schemes in the presence of complexity and delay constraints. Several works have considered JSCD schemes that exploit the leftover redundancy from the source encoder. In [72], JSCD using a soft-output Viterbi algorithm is considered. In [73], a trellis based decoder is used as a source decoder in an iterative decoding scheme. Joint decoding of Huffman and Turbo codes is proposed in [74]. In [75], joint decoding of variable length codes (VLCs) and convolutional/Turbo codes is analyzed. Applications of turbo codes to image/video transmission are shown in [76] and [77]. Joint decoding using LDPC codes for VLCs and images are illustrated in[78] and [79], respectively. However, few works have considered JSCD specifically for language-based sources. In [1], LDPC codes are combined with a language decoder and a message passing algorithm is designed to exploit the natural redundancy.

In this chapter, we propose a joint source-channel decoding scheme where the source is source encoded by Huffman codes and channel encoded by polar codes and stored. The proposed scheme decodes polar codes jointly with a language decoder using list decoding, based on a word dictionary. We assume the dictionary is only available and used on the decoder side, which is a reasonable assumption when the decoder has larger storage space and stronger calculation power, e.g., uplink channels where the source is compressed at a mobile device and uploaded to a data center. The insights of exploring joint list decoding of polar codes in the JSCD scheme include the following:

1. CRC-aided list decoding of polar codes has outstanding performance for short to

medium length codes [20]. The language decoder has a similar function to that of a CRC. The language decoder uses the word dictionary to select most probable paths, where the word dictionary can be viewed as local constraints on the decoded subsequences. The language decoder has a critical advantage over global CRC, that is, it can detect the validity of partially decoded paths before decoding the whole codeword. In this way, incorrect paths can be pruned at early stages, resulting in a larger probability that the correct path survives in the list. Based on this, we expect the joint list decoding scheme for polar codes to have superior performance over others.

2. Both polar list decoder and language decoder work over trees and that they can be combined in a computationally efficient way. This provides an efficient framework for exploiting the benefit of source redundancy;

3. The sequential nature of polar decoding makes polar codes more naturally suited for exploiting source redundancy than LDPC type codes. In general, SC decoding of polar codes suffers from error propagation. Wrongly decoded bits in early stages may severely degrade the decoding performance of the whole sequence. To alleviate this, we can reorder the words before encoding and put more reliable sub-sequences in front to suppress error propagation. By cleverly arranging the order of information bits based on the reliability/recover ability, we are able to substantially improve the overall performance.

We show from simulation results that the proposed decoder provides substantial improvement in performance over the CRC-aided list decoding of standalone polar codes, while the decoding complexity is kept in the same order. The improvement in the finite length performance inspires us to explore how much the rate of channel codes can be improved asymptotically by source redundancy. We first give a general model of decoding

with side information, and show that the source redundancy can help improve the error exponent and rate of channel codes. A theoretical model is studied for natural redundancy in compressed languages. That is, we model the source redundancy as a set of t erasure correcting block codes concatenated to the information bits. The block code is a simple and effective way to model the erasure correcting ability of words in the dictionary. Each block code corresponds to a word or a longer text. We show that the improvement in the rate of polar codes depends on the distribution of frozen bits within a codeword. We formally prove that the distribution of frozen bits converges to a limit distribution. Given the limit distribution, we first obtain lower bounds on the improved rate. Then an optimal information-bit allocation algorithm is proposed, and we analyze the convergence of maximum improvement in the rate of polar codes with the proposed algorithm.

4.2 System model and joint source-channel decoding

Fig. 4.1 illustrates the framework of the proposed coding scheme. We consider text in English, and the extension to other languages is straightforward. In our framework, the text is first compressed by Huffman codes and then encoded by polar codes. On the decoder side, the received sequence is jointly decoded by the polar code and a language decoder. The language decoder consists of Huffman decoding and dictionary tracing. It checks the validity of the decoded sequence by recognizing words in the dictionary. A detailed description of the proposed JSCD scheme is given below.

The maximum *a posteriori* decoder aims to find $\max_{u_0^{n-1}} P(u_0^{n-1}|y_0^{n-1})$. To avoid exponential complexity in n, we use list decoding to maximize $P(u_0^i|y_0^{n-1}), i \in [0, n-1]$ progressively by breadth-first search of a path in the decoding tree, where for each length-$(i+1)$ path, a constant number, often denoted by L, of most probable paths are kept to

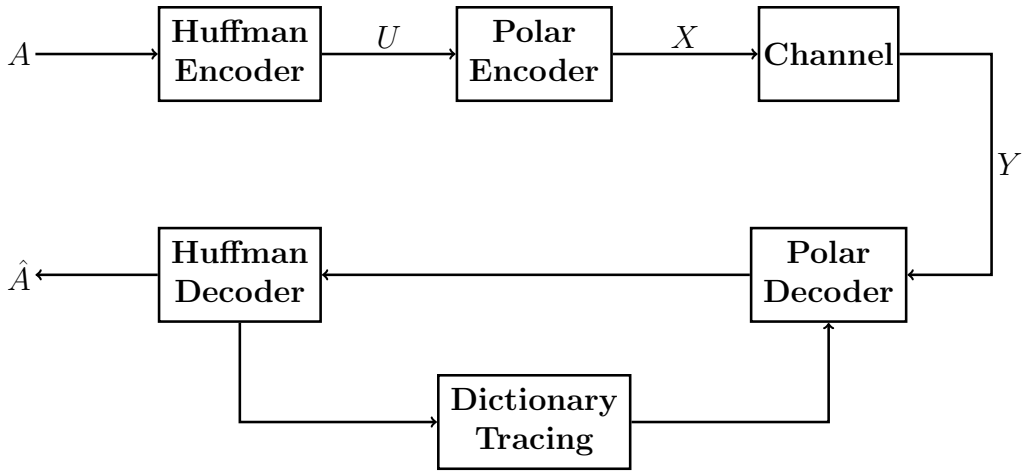


Figure 4.1: A system model for joint source-channel decoding

search for length-$(i+2)$ paths. Since

$$P(u_0^i|y_0^{n-1}) = \frac{P(u_0^i, y_0^{n-1})}{P(y_0^{n-1})} \propto P(y_0^{n-1}|u_0^i)P(u_0^i),$$

by source-channel separation theorem, a stand-alone polar decoder calculates the first term $P(y_0^{n-1}|u_0^i) \propto P(y_0^{n-1}, u_0^{i-1}|u_i)$ by a recursive structure, assuming u_0^i are independently and identically distributed (i.i.d.) Bernoulli(0.5) random variables, and thus the second term can be obliterated since $P(u_0^i) = 2^{-i-1}, \forall u_0^i \in \{0,1\}^{i+1}$. However, in the language-based JSCD framework, u_0^i are no longer i.i.d., one obvious consequence of which is that u_0^i is feasible only if the decoded text, translated from u_0^i by Huffman decoder, consists words in the dictionary. Therefore, $P(u_0^i)$ contributes critically to the path metric $P(u_0^i|y_0^{n-1})$, and in particular, if $P(u_0^i) = 0$, this path should be pruned despite the metric $P(y_0^{n-1}|u_0^i)$ obtained from the channel. This pruning technique enables early detection of decoding errors and is critical in keeping the correct path in the list. Algorithm 2 shows a high-level description of JSCD.

Algorithm 2 A high-level description of JSCD

Input: y_0^{n-1}, L
Output: u_0^{n-1}

Initialize: $i \leftarrow 0$; $l_{\text{act}} \leftarrow 1$;
while $i < n$ **do**
  if $i \in F$ **then**
    $u_i \leftarrow 0$ for each active path;
  else
    $k \leftarrow 1$;
    for each active path $l_j, j \in [l_{\text{act}}]$ **do**
      for $u_i = 0, 1$ **do**
        Compute $P(y_0^{n-1}, u_0^{i-1}|u_i)$;
        Update $P(u_0^i)$;
        $M_k \leftarrow P(y_0^{n-1}, u_0^{i-1}|u_i)P(u_0^i)$;
        $k \leftarrow k + 1$;
      end for
    end for
    $\rho \leftarrow \min(2l_{\text{act}}, L)$;
    Keep most probable ρ paths according to $M_1^{2l_{\text{act}}}$;
    $l_{\text{act}} \leftarrow \rho$;
  end if
  $i \leftarrow i + 1$;
end while
Select the most probable path and output u_0^{n-1}.

4.3 Practical implementation of JSCD

In this section we show a practical implementation of JSCD scheme. Let $\mathcal{A}$ be the alphabet of symbols in text (e.g., $\{a, b, \ldots, z\}$ for lowercase English letters, $\{0, \ldots, 127\}$ for symbols in ASCII table). Let $\mathcal{D}$ be the set of words in the dictionary. We assume a first order approximation of English words where all words are independent. The performance improvement of considering higher order Markov models of words [80] in languages is diminishing since redundancy within a Huffman-encoded word are much larger than redundancy across a sequence of words.

Proposition 11. The prior probability of source $P(u_0^i)$ can be efficiently computed from dictionary as follows:

$$P(u_0^i) = \prod_{m=1}^{j-1} P(w_m) P(l_1^k r) = \prod_{m=1}^{j-1} P(w_m) \sum_{w} P(w), \tag{4.1}$$

where w_1^{j-1} are $j-1$ uniquely decoded words in $\mathcal{D}$, l_1^k are k uniquely Huffman-decoded symbols in $\mathcal{A}$ and r is the remaining bit sequence. In the summation, $w \in \mathcal{D}$ satisfies that in binary Huffman-coded representation, the first k symbols equals l_1^k and r is a prefix of the remaining bit sequences.

Remark 12. The calculation of $P(u_0^i)$ should also take into account the probability of spaces (or punctuations) between words. We append a space mark to all words.

Now we focus on the efficient calculation of (4.1). Two trees are used to facilitate the calculation, one is a tree for Huffman coding and the other is a prefix tree (i.e., a trie) for tracing a partially decoded word in the dictionary.

4.3.1 Trie representation of the dictionary

A trie is an ordered tree data structure that is used to store a dynamic set or associative array where the keys are usually strings [81]. In our implementation, each node in the trie is instantiated as an object of a class named `DictNode`. As shown in Table 4.2, it has 4 data members, a symbol `c` (e.g., English letter), a variable `count` representing the frequency of the presence of this prefix, an indicator `is_a_word` indicating if the path from root to this node is a whole word, and a vector of pointers `child[]` pointing to their children. Fig. 4.2 is an illustrative example of the dictionary represented by a trie. In an established trie, if the pointer that points to the end of a word (or a partial word) w is known, then the calculation of $P(w)$ can be accomplished in $O(1)$ by dividing the count of the end node of the path associated with w by the count of the root node.

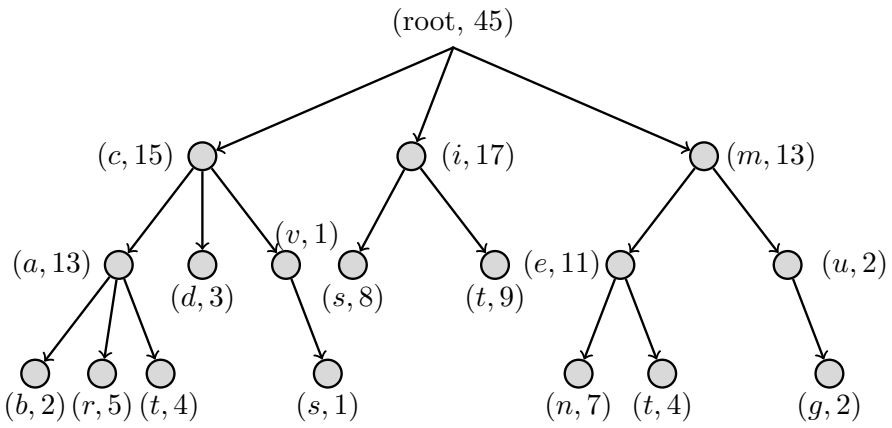
Table 4.2: `DictNode` members

member	type
c	char
count	int
is_a_word	bool
child[]	DictNode*

Table 4.3: `HuffNode` members

member	type
p	double
leftChild	huffNode*
rightChild	huffNode*
symSet	char*

In order to establish the trie from extracted text (e.g., from books, websites, etc.), an algorithm with an inductive process can be used. That is, suppose we have a trie $\mathcal{T}$ that represents the first i words of the extracted text, for the $(i+1)$st word $w = (l_1 \dots l_k)$ (assuming it contains k symbols), a pointer `p_dict` is created to point to the root and the first symbol l_1 in the word is compared with the children of the root in $\mathcal{T}$. If l_1 exists as the symbol of a depth-1 node m_1, then `p_dict` moves to m_1 and l_2 is compared with the children of m_1. The same operation continues until some l_j does not exist in the children



word	count
cab	2
car	5
cat	4
cd	3
cvs	1
is	8
it	9
men	7
met	4
mug	2

Figure 4.2: An illustrative example of a trie to represent the dictionary

set of the node m_{j-1} corresponding to path l_1^{j-1}. Then a new child with symbol l_j is added to m_{j-1} and the rest of the word l_{j+1}^k is added accordingly. During the scan of $(i+1)$st word, the counts for each node `p_dict` visits are increased by 1. Algorithm 3 shows the details of the algorithm.

Since searching for a symbol as a child of a node in $\mathcal{T}$ can be accomplished in $O(1)$ using a Hash table (e.g., `unordered_map` STL container in C++), the time complexity of establishing the trie would be $O(N_{\text{length}} N_{\text{word}})$, where N_{length} is the average length of a word and N_{word} is the number of words extracted from some resource.

4.3.2 Tree representation of Huffman codes

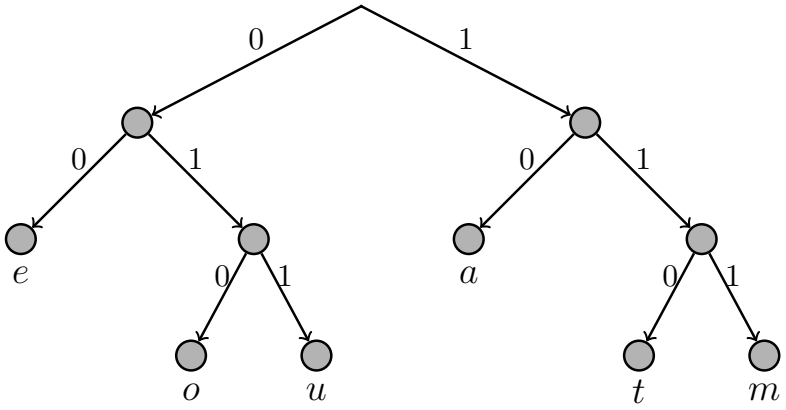
The Huffman codes for source coding are for 1-grams, namely characters, or more specifically, letters and space mark. In principle, we can also build a Huffman code for n-grams. The Huffman codes are represented as a binary tree. Each node in the tree is instantiated as an object of a class `HuffNode` whose members are shown in Table 4.3. In a typical Huffman tree realization, a node m consists of three members: the probability `p` of the associated symbol and two pointers to their left and right children (`leftChild`

Algorithm 3 Establish a trie for the dictionary from text

Input: a sequence of words $(w_1 w_2 \dots w_N)$, each word is represented as a string
Output: a trie $\mathcal{T}$

Initialize: Create a root node of $\mathcal{T}$ as an object of `DictNode`;
for $k = 1$ to N **do**
    Let `p_dict` point to the root of $\mathcal{T}$;
    for $i = 1$ to the length of w_k **do**
        if `*p_dict` has no child or $w_k[i]$ is not in the children set of `*p_dict` **then**
            Create a new node as an object of `DictNode` with `c` $\leftarrow w_k[i]$, `count` $\leftarrow$ `1` and `is_a_word` $\leftarrow$ `False`;
            Insert the new node as a child of `*p_dict`;
            Move `p_dict` to the new node;
            if $i ==$ the length of w_k **then**
                `p_dict->is_a_word` $\leftarrow$ `True`;
            end if
        else
            Find j, s.t. $w_k[i] ==$`p_dict->child[j]->c`;
            `p_dict->count++`;
            `p_dict` $\leftarrow$ `p_dict->child[j]`;
        end if
    end for
end for

and `rightChild`). In addition, we implement a fourth data member `symSet`, that is, a set of symbols that are descendants of m. This extra data member helps in simplifying the calculation of (4.1) in the following manner. Note that in (4.1), $P(l_1^k r)$, the probability of a partial word is required. Assume l_1^k is a path that ends in a node n_k in the trie-represented dictionary $\mathcal{T}$ and r is a path that ends in a node n_r in the Huffman tree $\mathcal{H}$. Then $P(l_1^k r)$ can be calculated by summing up the counts (or probability) of the subset of children of $n_k \in \mathcal{T}$, such that the symbols associated with this subset are all descendants of $n_r \in \mathcal{H}$. By associating all descendants of n_r as a data member to the node itself, the complexity of calculating $P(l_1^k r)$ is linear in the number of descendants of n_r, which is typically a small number and decreases exponentially in the depth of n_r. Fig. 4.3 shows an illustrative example of a Huffman tree.



symbol	code
e	00
o	010
u	011
a	10
t	110
m	111

Figure 4.3: An illustrative example of a Huffman tree.

4.3.3 Calculation of $P(u_0^i)$ with $\mathcal{T}$ and $\mathcal{H}$

Next, we present an algorithm to calculate $P(u_0^i)$ progressively according to (4.1). In each of $\mathcal{T}$ and $\mathcal{H}$, two pointers, denoted by `p_dict` and `p_huff`, are used respectively

to locate the current decoding stages $i \in [0, n-1]$. They are initiated to point to the root of $\mathcal{T}$ and $\mathcal{H}$, respectively. A simple description of the algorithm is as follows. Let u_0^{i-1} be represented as $(w_1^{j-1} l_1^k r)$ and suppose each term in (4.1) is known up to index $i-1$. Suppose `p_dict` and `p_huff` point to two nodes in $\mathcal{T}$ and $\mathcal{H}$. To update $P(u_0^i)$, first, `p_huff` moves to its left or right child according to u_i. Let $\mathcal{S}$ denote all descendant symbols of $\star$`p_huff`. Replace $P(l_1^k r)$ by the summation of probabilities associated with a set of children, denoted by $\mathcal{C}$, of $\star$`p_dict` such that $\forall a \in \mathcal{C}$, the symbols associated with a belongs to $\mathcal{S}$; If $\star$`p_huff` is a leaf, then `p_dict` moves to its child according to the symbol $\star$`p_huff` associates and `p_huff` is reset to point to the root of $\mathcal{H}$. If the symbol that $\star$`p_huff` associates with does not exist in the children of $\star$`p_dict`, that means $P(u_0^i)$ should be set to 0 and this path has a decoding error and thus be pruned. If furthermore $\star$`p_dict` is an end node of a word in $\mathcal{T}$, replace $P(l_1^k r)$ by $P(w_j)$ and `p_dict` is reset to point to the root of $\mathcal{T}$. Let the multiplication of probabilities in (4.1) be denoted by P_{wd}, i.e., $P_{wd} = \prod_{m=1}^{j-1} P(w_m)$, where P_{wd} can be updated recursively. A detailed description is presented in Algorithm 4, 5 and 6.

Theorem 13. The overall decoding complexity of JSCD in Algorithm 2 and 4 is $O(Ln(\log n))$.

Proof. Note that the only difference of Algorithm 2 from the list decoder of stand-alone polar codes presented in [20] is the introduction of $P(u_0^i)$ in 10th line. The complexity of Algorithm 4 involves operations of the two pointers, both of which are local operations in the dictionary tree and Huffman tree. It takes $O(1)$ operations to extract the descendants of $\star$`p_huff` and it takes at most $O(N_{\text{child}})$ to sum up their probabilities, where N_{child} is the number of children of a node in $\mathcal{T}$. Therefore, updating $P(u_0^i)$ is constant in n. Thus, complexity of Algorithm 2 is $O(Ln(\log n + C))$, where C is a constant. In other words, the proposed JSCD algorithm has the same order of complexity as list decoders. □

Algorithm 4 Update $P(u_0^i)$

Input: u_i, $\mathcal{T}$, $\mathcal{H}$, `p_dict`, `p_huff`, P_{wd}
Output: `p_dict`, `p_huff`, $P(u_0^i)$, P_{wd}

1: $\mathcal{S} \leftarrow$`TraceHuffmanTree(`$\mathcal{H}$`,p_huff,`u_i`);`
2: $\mathcal{C} \leftarrow$`TraceDict(`$\mathcal{T}$`,p_dict,`$\mathcal{S}$`);`
3: $P(l_1^k r) \leftarrow \sum_{w \in \mathcal{C}} P(w)$;
4: $P(u_0^i) \leftarrow P_{wd} \cdot P(l_1^k r)$;
5: **if** `p_huff` points to a leaf in $\mathcal{H}$ **then**
6: Move `p_dict` to its child according to `p_huff` ;
7: Move `p_huff` to the root of $\mathcal{H}$;
8: **if** `p_dict` points to a leaf in $\mathcal{T}$ **then**
9: $P(w_j) \leftarrow P(l_1^k r)$;
10: $P_{wd} \leftarrow P_{wd} \cdot P(w_j)$;
11: **end if**
12: **end if**

Algorithm 5 `TraceHuffmanTree(`$\mathcal{H}$`,p_huff,`u_i`)`

Input: u_i, $\mathcal{H}$, `p_huff`
Output: $\mathcal{S}$

1: **if** $u_i == 0$ **then**
2: Move `p_huff` to its left child;
3: **else**
4: Move `p_huff` to its right child;
5: **end if**
6: $\mathcal{S} \leftarrow$ `p_huff->symSet`;

Algorithm 6 `TraceDict(`$\mathcal{T}$`,p_dict,`$\mathcal{S}$`)`

Input: $\mathcal{T}$, `p_dict`, $\mathcal{S}$
Output: $\mathcal{C}$

1: $\mathcal{C} \leftarrow \emptyset$;
2: **for** each symbol $s \in \mathcal{S}$ **do**
3: **if** s is found in the children set of $\star$`p_dict` **then**
4: s is added to $\mathcal{C}$;
5: **end if**
6: **end for**

4.3.4 List-size adaptive JSCD

To improve the efficiency of JSCD, we implement the list-size adaptive list decoders as in [23]. A few CRC bits are added for error detection. The adaptive list decoders start with $L = 1$ and computes an estimate u_0^{n-1}. If u_0^{n-1} satisfies the CRC, then u_0^{n-1} are output as the decoded bits, otherwise, the list size doubles and the list decoding is repeated. This process continues until u_0^{n-1} satisfies the CRC for some L_{success} or the list size reaches a threshold L_{max}.

4.4 Simulation results

We present some numerical results showing the superiority of JSCD over the stand-alone polar list decoder.

4.4.1 Dictionary

The dictionary is built from about 10 million extracted words in Wikipedia pages. According to a word frequecy analysis in [82], the top 3000 most frequent words take 81% of the probability. In the dictionary tree implemented in this chapter, there are $N_s = 180133$ nodes of type `DictNode`.

4.4.2 Polar codes and channel parameters

In our simulation, the length of polar codes is fixed to $n = 8192$ and the code rate is 0.923. Two typical B-DMCs are assumed, namely, AWGN channels and BSCs. The polar code used for AWGN channels is constructed by density evolution in [22] at $\frac{E_b}{N_0} = 4$ dB. The polar code used for BSCs is similarly constructed for a BSC with cross-over probability 0.002, which is the same as the channel parameter for LDPC designs in [1].

4.4.3 Results

Fig. 4.4 gives a comparison of different decoders for AWGN channels. It can be seen that at block error rate below 10^{-3}, more than 0.6 dB gain over stand-alone CRC-aided list

decoders with $L = 1024$ can be realized by the list-size adaptive joint list decoders. It is observed in our simulation that $L = 1024$ would be large enough such that further increase of the list size will not contribute much to the performance. The decoding complexity of the list-size adaptive joint list decoding is much lower than that of joint decoding with fixed list size. Table 4.4 shows that the average list size L_{success} decreases dramatically with the increase of SNRs. We see that at $\frac{E_b}{N_0} = 4$ dB, $L_{\text{success}} = 2.24$ for all $L_{\max} = 128$ and 1024.

Fig. 4.5 gives a comparison of four decoders for BSCs. The results consistently show the superiority of JSCD over CRC-aided list decoding. Fig. 4.6 shows a comparison of joint decoding using LDPC codes [1] and our schemes. The polar code has length $n = 4096$ and rate 0.936, which is smaller in length and the same in rate as in [1]. We should note that dictionaries for two schemes are built separately. The dictionary in [1] is incomplete, i.e., not all words in sources are in the dictionary, while our dictionary is complete. Thus the comparison is not entirely fair.

Table 4.4: Average list size of JSCD

E_b/N_0 (dB)	3	3.25	3.5	3.75	4
$L_{\max} = 32$	30.89	25.94	13.68	5.46	2.22
$L_{\max} = 128$	113.09	59.27	21.84	5.66	2.24
$L_{\max} = 1024$	547.66	177.57	34.12	6.08	2.24

4.5 Discussion on language statistics

In this section, some properties of language-based sources are discussed to explain the significant gains achieved by JSCD.

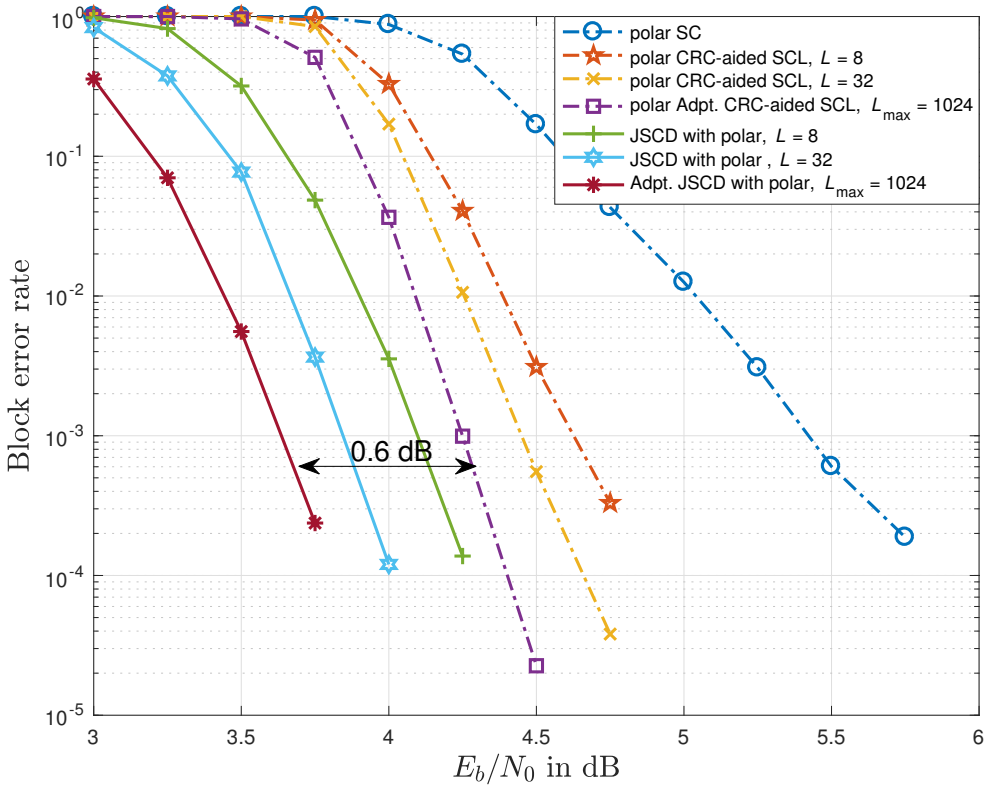


Figure 4.4: Block error rate of different decoding schemes over AWGN channels: a) SC decoding; b) CRC-aided list decoding ($L = 8, 32$); c) List-size adaptive CRC-aided list decoding ($L_{\max} = 1024$); d) JSCD ($L = 8, 32$); e) List-size adaptive JSCD ($L_{\max} = 1024$). All codes have length $n = 8192$ and $k = 7561$.

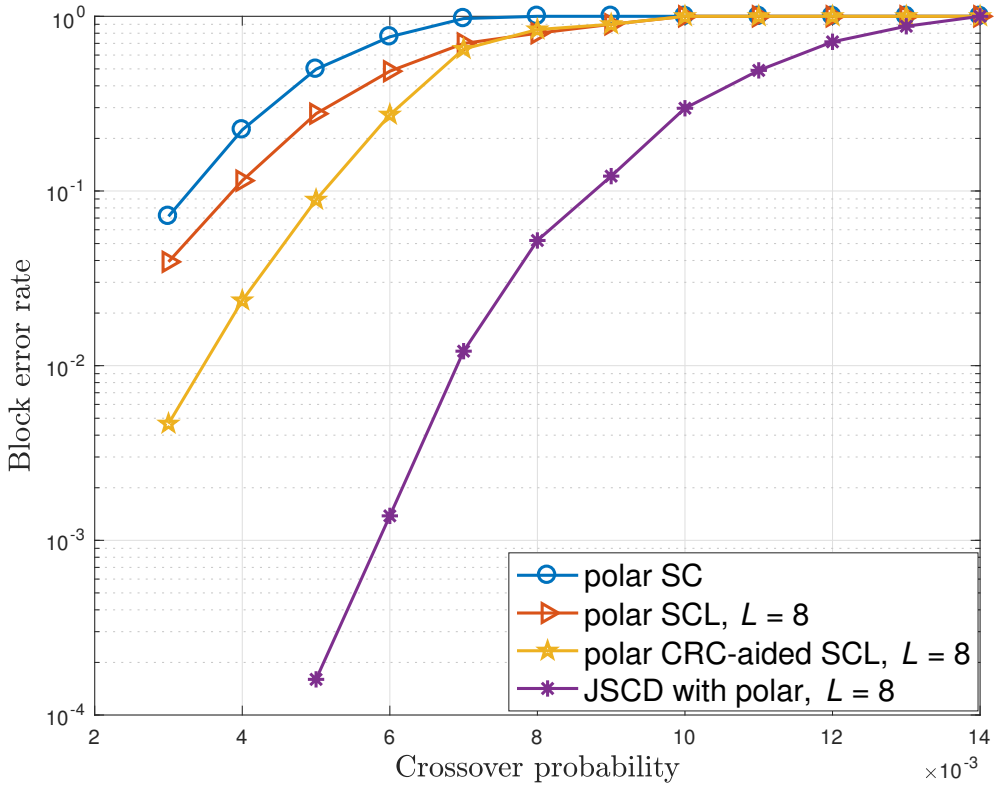


Figure 4.5: Block error rate of different decoding schemes over BSCs

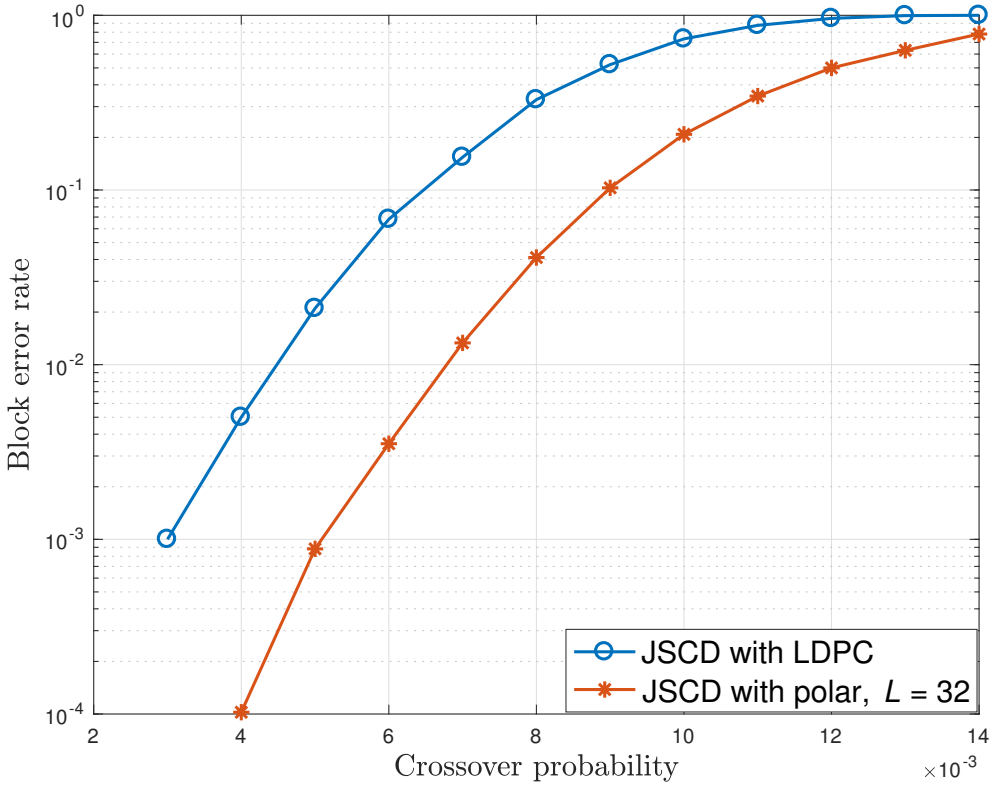


Figure 4.6: Comparison of JSCD with LDPC codes [1] over BSCs

4.5.1 Redundancy of Huffman-encoded text

The language has diverse features including semantics, grammar and syntax, *etc*. From the fundamental coding theorem, the average number of bits to represent a word is lower bounded by the entropy of words. Shannon estimated the word entropy of printed English to be 11.82 bits per word [63]. Practically, we have collected a large number of words from extracted text and computed the entropy of words by $H(X) = -\sum_i p_i \log_2 p_i$, where $H(X)$ is the entropy of the source X and p_i is the probability of the ith unique word, assuming that the words are independent. In our extracted text, the resultant entropy of words is estimated to be 10.41 bits per words. However, the average number of bits for a Huffman-encoded word is approximately 37 bits per words, which is much larger than both estimates, showing great redundancy remaining in the compressed text. The major reason for such redundancy is that Huffman codebook is generated by the distribution of English letters instead of words, where strong correlation between letters exists. Some other factors to cause the difference in the length of Huffman-encoded word and the entropy of words include integer-length constraint of Huffman codes and mismatch between the source model and the actual text transmitted.

4.5.2 Sparsity of words

Let M_n denote the number of Huffman-encoded binary sequences of length n that correspond to a word in the dictionary. We call such a sequence a valid binary sequence. Let P_n be defined as $P_n = \frac{M_n}{2^n}$, i.e., P_n is the probability that a uniformly and randomly chosen binary sequence of length n corresponds to a valid word. We can write P_n in an exponential form $P_n = 10^{-x_n}$, where x_n represents the growth rate of sparsity of valid binary sequences. Based on statistics of the extracted text, M_n and x_n are shown in Fig. 4.7 and Fig. 4.8, respectively. Fig. 4.7 illustrates that the length of Huffman-encoded binary sequence for more than 97% of words is concentrated between 15 and 70. Fig. 4.8 shows

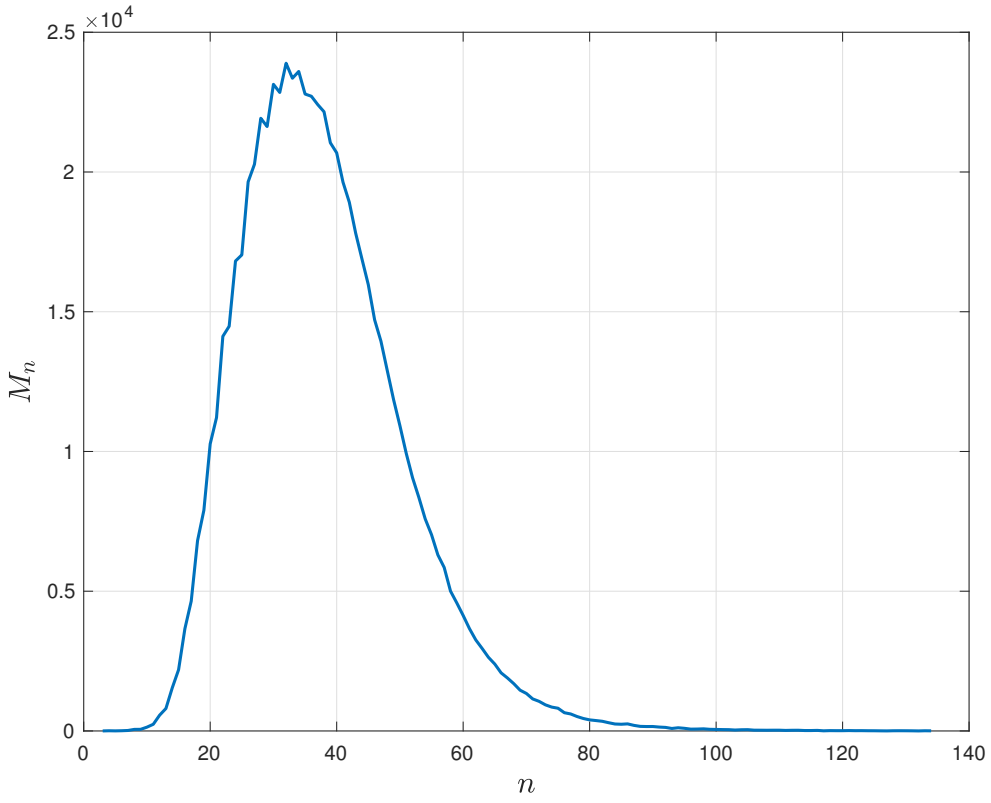


Figure 4.7: The number of words with length-n Huffman-encoded binary sequences

that x_n increases almost linearly in n for $n > 15$, thus P_n decreases exponentially in n. Therefore, if n is large, P_n is very small, meaning valid binary sequences are sparse. The sparsity of valid words indicates that once the decoded binary sequence corresponds to a valid word, there is a high probability that the decoded sequence is correct. In fact, the Hamming distance and the probability of words largely affect the error probability. The sparsity of valid words is a necessary condition for a large Hamming distance. Thus we see that the sparsity of words implies high efficiency of the dictionary in pruning paths.

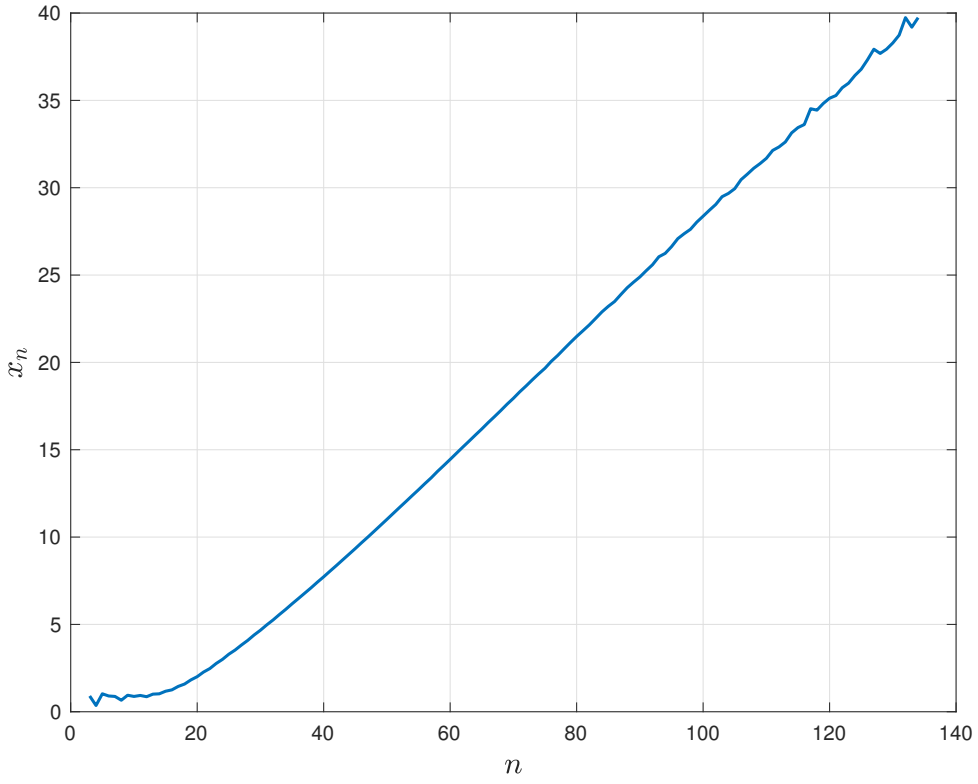


Figure 4.8: The sparsity exponent x_n, where $P_n = 10^{-x_n}$ is the probability that a uniformly and randomly chosen binary sequence of length n corresponds to a valid word.

4.6 Theoretical analysis of improved rate of polar codes

Simulation results in Section 4.4 show significant gains of the JSCD scheme in practice, which lay ground for theoretical analysis of the benefits of the scheme. Particularly, we give a theoretical analysis of improved rate of polar codes at high level in this section. First a general model of decoding is given with side information. We show that the source redundancy can help improve the error exponent and rate of channel codes. Then we analyze the rate improvement of polar codes.

4.6.1 General decoding model with source redundancy

In a system where the source is compressed without removing all the redundancy, the leftover redundancy can act as the side information to help improve the decoding performance, while no change is made to the encoding structure. In this chapter, the source redundancy is modeled as side information as shown in Fig. 4.9. There are two parallel channels, one transmitting the normal codewords, and the other transmitting the side information. The source $\mathbf{w}$ is first source encoded into $\mathbf{u}$ and channel encoded into $\mathbf{x}$, and then is transmitted through a channel to the decoder. The decoder receives $\mathbf{y}$ and has side information $\mathbf{v}$ available. $\mathbf{v}$ is correlated with the source $\mathbf{w}$ and the correlation depends on the redundancy left in the source.

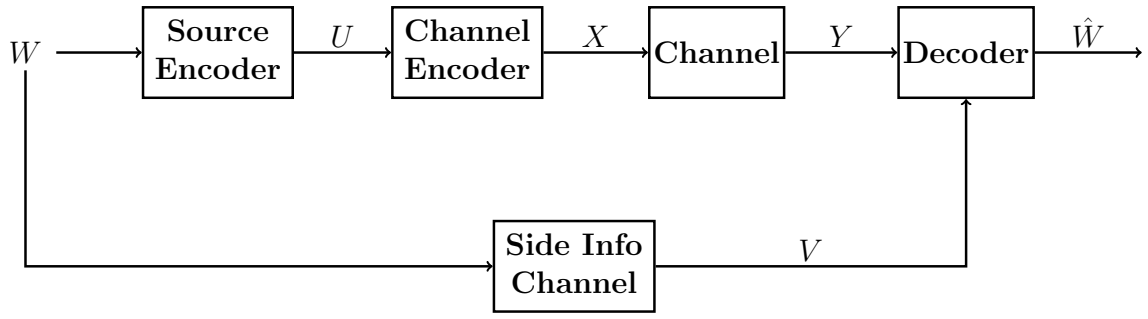


Figure 4.9: A decoding model with side information.

Lemma 14. Assume the source is encoded by source and channel codes and transmitted through a DMC with transition probability $P(y|x)$. Each codeword is chosen i.i.d from distribution $\mathbf{q}$. Denote n as the length of the code. Assume the side information $\mathbf{v}$ is available at the decoder with prior probability $p(\mathbf{v})$. The source $\mathbf{w}$ is transmitted through a channel with transition probability $p(\mathbf{v}|\mathbf{w})$. The average error probability of the ensemble is bounded by

$$\bar{P}_e \leq 2^{-nE_0(\rho,\mathbf{q})+E_s(\rho)}, \rho \in (0,1],$$

where $E_0(\rho,\mathbf{q})$ and $E_s(\rho)$ are defined as

$$E_0(\rho,\mathbf{q}) = -\log \sum_j \left(\sum_k q(k)P(j|k)^{1/(1+\rho)} \right)^{1+\rho},$$
$$E_s(\rho) = \log \left(\sum_{\mathbf{v}} p(\mathbf{v}) \left(\sum_{\mathbf{w}} p(\mathbf{w}|\mathbf{v})^{1/(1+\rho)} \right)^{1+\rho} \right).$$

Proof. Let $\bar{P}_{e,\mathbf{v}}$ denote the average error probability given side information $\mathbf{v}$. From Exercise 5.16 of [31] we know that if the prior probability of the source is available at the decoder, with MAP decoding the average error probability of the ensemble given $\mathbf{v}$ is bounded by

$$\bar{P}_{e,\mathbf{v}} \leq \left(\sum_{\mathbf{w}} p(\mathbf{w}|\mathbf{v})^{1/(1+\rho)} \right)^{1+\rho} 2^{-nE_0(\rho,\mathbf{q})}. \tag{4.2}$$

The average error probability is derived as

$$\bar{P}_e = \sum_{\mathbf{v}} p(\mathbf{v})\bar{P}_{e,\mathbf{v}} \leq 2^{-nE_0(\rho,\mathbf{q})} \sum_{\mathbf{v}} p(\mathbf{v}) \left(\sum_{\mathbf{w}} p(\mathbf{w}|\mathbf{v})^{1/(1+\rho)} \right)^{1+\rho}.$$

Then $E_s(\rho)$ can be derived from above. $\square$

Example 15. Assume that the source $\mathbf{w}$ is a binary memoryless source (BMS) with length l, and the channel code has length n. Each symbol in the source is i.i.d. Assume that the channel to transmit the side information $\mathbf{v}$ is a BEC with erasure probability β. Then, $E_s(\rho)$ can be computed as

$$\begin{aligned} E_s(\rho) &= l \log \sum_v p(v) \left(\sum_w p(w|v)^{1/(1+\rho)} \right)^{1+\rho} \\ &= l \log \left((1-\beta) + \beta 2^{\rho} \right) \end{aligned}$$

Without loss of generality assume $R = \frac{l}{n}$, we have

$$E_s(\rho) = Rn \log \left((1-\beta) + \beta 2^{\rho} \right). \tag{4.3}$$

Assume the channel is BEC(ϵ) and the channel for side information is BEC(β). If we define $E_0(\rho)$ as $E_0(\rho) = \max_{\mathbf{q}} E_0(\rho, \mathbf{q})$, for BEC($\epsilon$) $E_0(\rho)$ is derived as

$$E_0(\rho) = -\log \left((1-\epsilon) 2^{-\rho} + \epsilon \right),$$

and $E_s(\rho)$ is given in (4.3). Let $E_r(R) = E_0(\rho) - E_s(\rho)/n$ denote the random exponent. We plot $E_r(R)$ in Fig. 4.10. It can be observed that the random exponent with $\beta \in [0, 1)$ is larger than that without side information. Fig. 4.11 shows the maximum rate for which $E_r(R) > 0$ as a function of β. We see that the maximum rate is also improved by the side information.

The preceding discussion shows that for the random code ensemble, the rate can be improved by side information available at decoder. In the following sections, we consider a practical joint decoding scheme with polar codes. We study a particular model for the

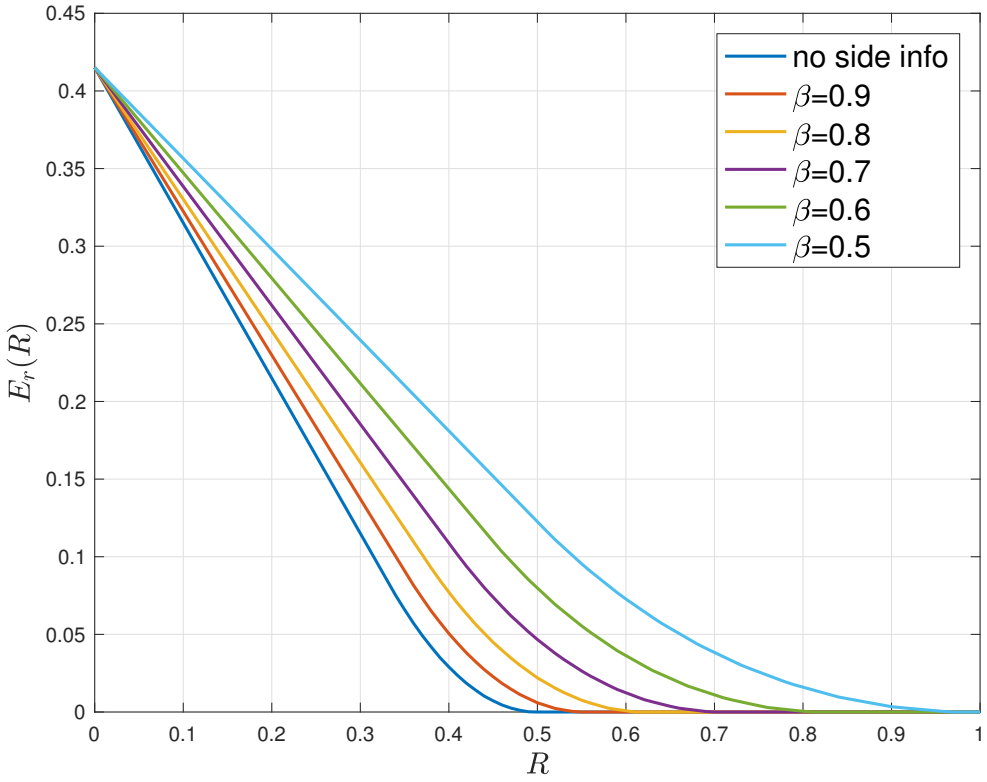


Figure 4.10: Random error exponent for BEC with side information.

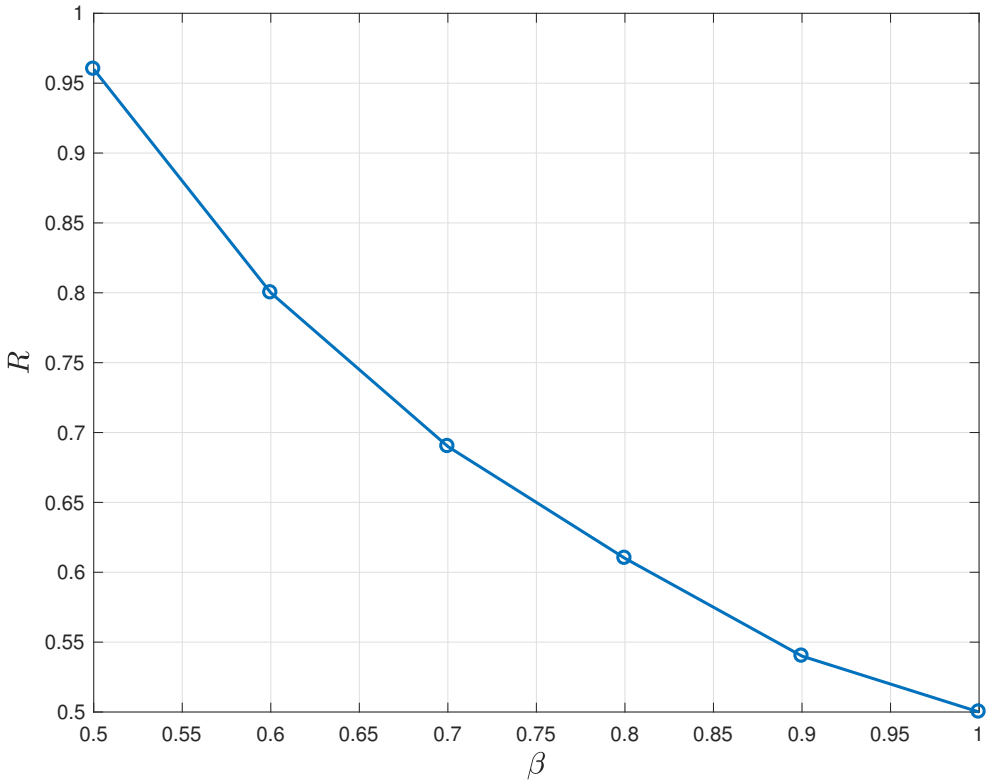


Figure 4.11: Maximum rate with side information for different β.

side information and analyze the improved rate of polar codes for that model.

4.6.2 A simplified model for natural redundancy in languages

Let $[n_0, t]$ denote a block code of length n_0 that can correct t erasures. The natural redundancy in the source is modeled such that each block of n_0 consecutive output bits from the side information channel corresponds to a word in the dictionary or a longer text and is assumed to be a codeword of a $[n_0, t]$ outer code. Let $B = (b_0, b_1, b_2 \cdots)$ be the bits in a compressed text. We assume that B is a sequence of codewords of $[n_0, t]$ codes.

Let $\mathcal{C}$ be an (n, k) polar code. Let $U = (u_0, u_1, \cdots, u_{n-1})$ be the input bits and let $X = (x_0, x_1, \cdots, x_{n-1})$ be a codeword of polar codes . The encoder normally encodes U – k of which are information bits, and $n - k$ of which are frozen bits – into X. Let $\mathcal{F} \subset \{0, 1, \cdots, n-1\}$ denote the indices of the frozen bits. Here, we consider a scheme where the natural redundancy in the information bits is used to improve the code rate by *unfreezing some of the frozen bits*, namely, some frozen bits are also used as information bits. Let $\mathcal{E} \subset \mathcal{F}$ denote the indices of the bits we unfreeze. We use the bits $\{u_i \mid i \in \{0, 1, \cdots, n-1\} - (\mathcal{F} - \mathcal{E})\}$ to store the bits in B, and use the SC decoding for error correction. For the SC decoder, if the first $n_0 - t$ bits of the n_0 bits are decoded correctly, the last t bits can be filled in by the outer erasure code, regardless of the decisions on the last t bits made by the SC decoder. Equivalently, the rate of the polar code can be increased by sending those last t bits in frozen bit positions. Since B is a sequence of $[n_0, t]$ codes, to make decoding successful, the requirement is that for every $[n_0, t]$ code, its first $n_0 - t$ bits should be stored in bits (of U) whose indices are not in $\mathcal{F}$. Let $|\mathcal{E}|$ be denoted as the cardinality of the set $\mathcal{E}$. The code-rate improvement is $\Delta R = |\mathcal{E}|/n$. We give a simple example.

Example 16. Consider an $(8, 4)$ polar code and let the outer code be a $[2, 1]$ code. Let u_1, u_2, u_3 and u_5 be the frozen bits. We can unfreeze u_5 by grouping u_5 with u_4 into a

codeword of the $[2, 1]$ outer code that corrects 1 erasure. If u_4 is decoded correctly by the polar code, the outer code can correctly decode u_5. The rate of inner polar code is therefore increased by $\frac{1}{8}$ without increasing the block error rate.

The model is suitable for many compression algorithms, such as Huffman coding and LZW coding with a fixed dictionary of patterns, where every binary codeword in the compressed text represents a sequence of characters in the original text, and the natural redundancy in multiple adjacent codewords (e.g., the sparsity of valid words, phrases or grammatically correct sentences) can be used to correct erasures. Theoretically, the greater n_0 is, the better this model is for languages.

The improvement in code rate depends on the distribution of frozen bits in the polar code. Generally speaking, the further "behind" the frozen bits' positions are in the polar code, the more code-rate improvement can be achieved because more frozen bits can be used in the $[n_0, t]$ codes. The distribution of the frozen bits is in general a function of n. In the next subsection, we show that the distribution converges to a limit distribution.

We should mention that there is another natural way to exploit the redundancy when the source is modeled as codewords of a $[n_0, t]$ code. After the source is compressed into $[n_0, t]$ codes, we can puncture the last t bits of coded bits and encode the remaining bits with capacity-achieving channel codes. This is equivalent to compressing the source further before encoding by the channel code and is in the spirit of separate source and channel coding. Indeed, when the channel is perfectly known at the receiver and for asymptotic lengths, this scheme would be optimal. However, our scheme of keeping the redundancy is more robust to the variations of the channel. First, if the channel is time-varying, the decoder can decide how much redundancy to exploit based on its own channel conditions. When the channel is degraded, for finite lengths, the newly unfrozen bits can still provide some information which can be exploited in the decoder. When the channel is upgraded,

the decoding complexity at the language decoder can be reduced. Second, if the source is transmitted over multiple channels, a universal coding scheme may be required and unfreezing the frozen bits may provide a more robust solution. We do not explore these specific situations in detail in this dissertation but the theoretical results presented in the next section may be useful in analyzing these situations further. A more careful analysis is required to fully understand the utility of the proposed scheme.

4.6.3 Convergence of frozen-bit distribution in polar codes

We characterize the distribution of frozen bits as follows. Given a real number y, let y^- be $y-1$ if y is an integer, and be $\lfloor y \rfloor$ otherwise. $\forall\ x \in [0,1]$, let $\mathcal{F}_n(x)$ be the frozen set (i.e., indices of frozen bits) among the first $(nx)^- + 1$ bits of the polar code. Formally, given an arbitrarily small $\epsilon \in [0,1)$,

$$\mathcal{F}_n(x) \triangleq \{i \in [0,(nx)^-] : I(W_n^{(i)}) \in [0, 1-\epsilon)\}.$$

Let $f_n(x) \triangleq \frac{|\mathcal{F}_n(x)|}{nx} \in [0,1]$, which describes the distribution of frozen bits (or more specifically, the proportion of frozen bits among the first nx bits). We show the convergence of $f_n(x)$ in the following theorem [67].

Theorem 17. For a polar code with length $n = 2^m$ and rate $R = I(W) - \delta$ where δ is an arbitrarily small positive number, the function $f_n(x)$ converges as $n \to \infty$.

Proof. Assume a polar code has length $n = 2^m$ and rate R. Let $\mathcal{B}_n(x)$, $\mathcal{M}_n(x)$ and $\mathcal{A}_n(x)$ be defined as follows: for any arbitrarily small $\epsilon \in [0,1)$,

$$\mathcal{B}_n(x) \triangleq \{i \in [0,(nx)^-] : I(W_n^{(i)}) \in [0, \epsilon]\},$$
$$\mathcal{M}_n(x) \triangleq \{i \in [0,(nx)^-] : I(W_n^{(i)}) \in (\epsilon, 1-\epsilon)\},$$
$$\mathcal{A}_n(x) \triangleq \{i \in [0,(nx)^-] : I(W_n^{(i)}) \in [1-\epsilon, 1]\}.$$

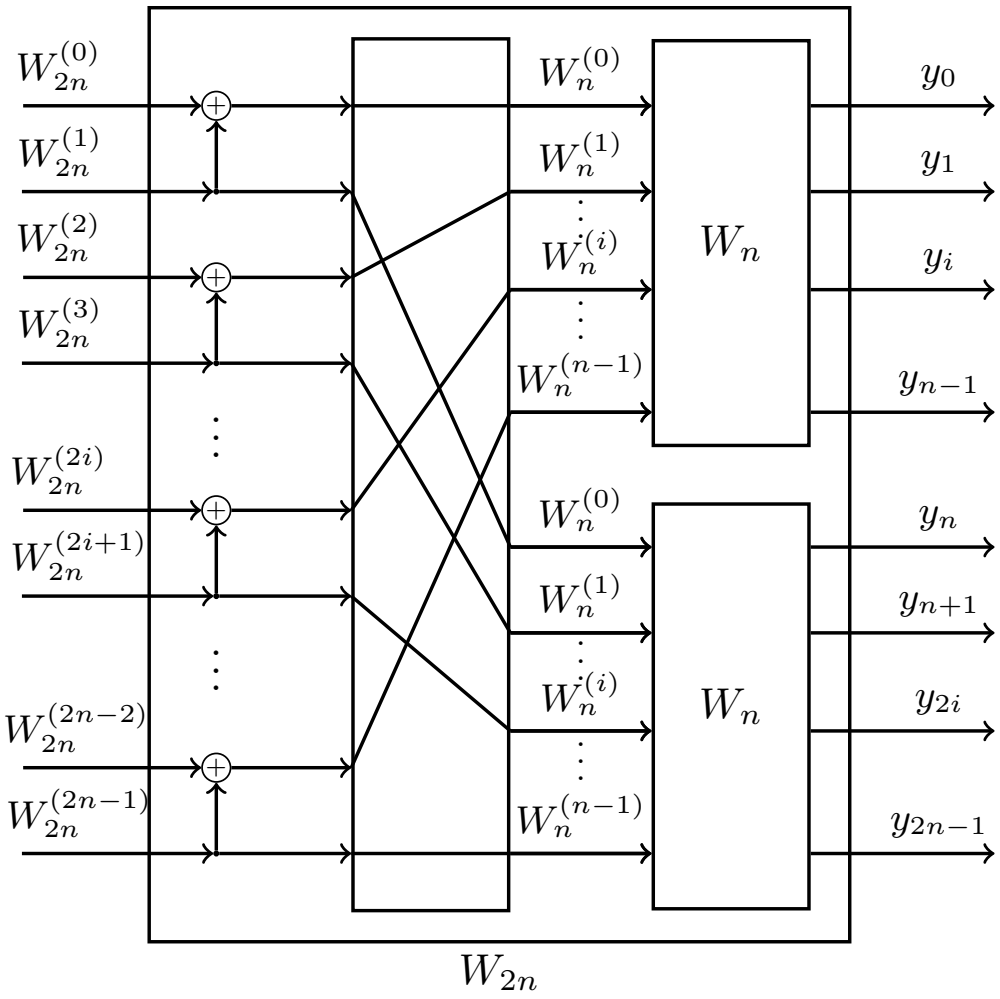


Figure 4.12: One step transformation to get $W_{2n}^{(2i)}$ and $W_{2n}^{(2i+1)}$ from $W_n^{(i)}$

We have $\mathcal{F}_n(x) = \mathcal{B}_n(x) \cup \mathcal{M}_n(x)$. Define $g_n(x)$, $t_n(x)$ and $h_n(x)$ as

$$h_n(x) = \frac{|\mathcal{B}_n(x)|}{nx},\ t_n(x) = \frac{|\mathcal{M}_n(x)|}{nx},\ \text{and}\ g_n(x) = \frac{|\mathcal{A}_n(x)|}{nx}.$$

From the definition of frozen set, we have $f_n(x) = h_n(x) + t_n(x) = 1 - g_n(x)$. As $n \to \infty$, almost all channels polarize in the sense that $g_n(1) \to I(W)$, $h_n(1) \to 1 - I(W)$ and $t_n(1) \to 0$. For polar code of length $n = 2^m$, if we perform one more step of transformation, the length of the code is increased to $2n$. The ith bit channel $W_n^{(i)}$ is

split to $W_{2n}^{(2i)}$ and $W_{2n}^{(2i+1)}$, where $I(W_{2n}^{(2i)}) \leq I(W_n^{(i)}) \leq I(W_{2n}^{(2i+1)})$. Fig. 4.12 shows the process of one step transformation to get $W_{2n}^{(2i)}$ and $W_{2n}^{(2i+1)}$ from $W_n^{(i)}$. Define $\Delta_I = I(W_n^{(i)}) - I(W_{2n}^{(2i)})$. We know that $\Delta_I = I(W_{2n}^{(2i+1)}) - I(W_n^{(i)})$. By computing Δ_I for the two extreme channels (BEC and BSC), upper and lower bounds of Δ_I can be derived. The maximum value of Δ_I is achieved for BEC:

$$\Delta_I^{max} = I(W_n^{(i)}) - I^2(W_n^{(i)}),$$

and the minimum value is achieved for BSC:

$$\Delta_I^{min} = H(2p(1-p)) - H(p)$$

where $I(W_n^{(i)}) = 1 - H(p)$, and $H(p)$ is the entropy of a BSC with crossover probability p. The upper and lower bounds of Δ_I are plotted in Fig. 4.13. We see that if the ith channel has $I(W_n^{(i)}) \in [0, \epsilon]$ or $I(W_n^{(i)}) \in [1-\epsilon, 1]$, after one step of transformation the capacity of $W_{2n}^{(2i)}$ or $W_{2n}^{(2i+1)}$ is very close to that of $W_n^{(i)}$.

Consider the ith bit channel, if $i \in \mathcal{A}_n(x)$, $I(W_n^{(i)})$ is bounded by $1-\epsilon \leq I(W_n^{(i)}) \leq 1$. It is easily seen that $2i+1 \in \mathcal{A}_{2n}(x)$ as $I(W_n^{(i)}) \leq I(W_{2n}^{(2i+1)})$. Since $I(W_{2n}^{(2i)}) \geq I(W_n^{(i)})^2$, we have $I(W_{2n}^{(2i)}) \geq (1-\epsilon)^2$. Since $(1-\epsilon)^2 > \epsilon$ for $\epsilon < \frac{3-\sqrt{5}}{2} = 0.382$, if $\epsilon < 0.382$, it is guaranteed that $I(W_{2n}^{(2i)}) > \epsilon$, which indicates $2i \notin \mathcal{B}_{2n}(x)$. Similarly, if $i \in \mathcal{B}_n(x)$, $I(W_n^{(i)})$ satisfies $0 \leq I(W_n^{(i)}) \leq \epsilon$. Since $I(W_{2n}^{(2i+1)}) \leq 2I(W_n^{(i)}) - I(W_n^{(i)})^2$, we have $I(W_{2n}^{(2i+1)}) \leq 2\epsilon - \epsilon^2$. Since $2\epsilon - \epsilon^2 < 1 - \epsilon$ for $\epsilon < \frac{3-\sqrt{5}}{2} = 0.382$, if $\epsilon < 0.382$, it is guaranteed that $I(W_{2n}^{(2i+1)}) < 1 - \epsilon$, indicating $2i+1 \notin \mathcal{A}_{2n}(x)$. In summary, if $\epsilon < 0.382$, the following constraints are satisfied: if $i \in \mathcal{A}_n(x), 2i \notin \mathcal{B}_{2n}(x)$, and if $i \in \mathcal{B}_n(x), 2i+1 \notin \mathcal{A}_{2n}(x)$. We can conclude that for n large enough the following hold:

1. If $i \in \mathcal{B}_n(x)$, $2i \in \mathcal{B}_{2n}(x)$, and $2i+1$ can be either in $\mathcal{B}_{2n}(x)$ or $\mathcal{M}_{2n}(x)$;

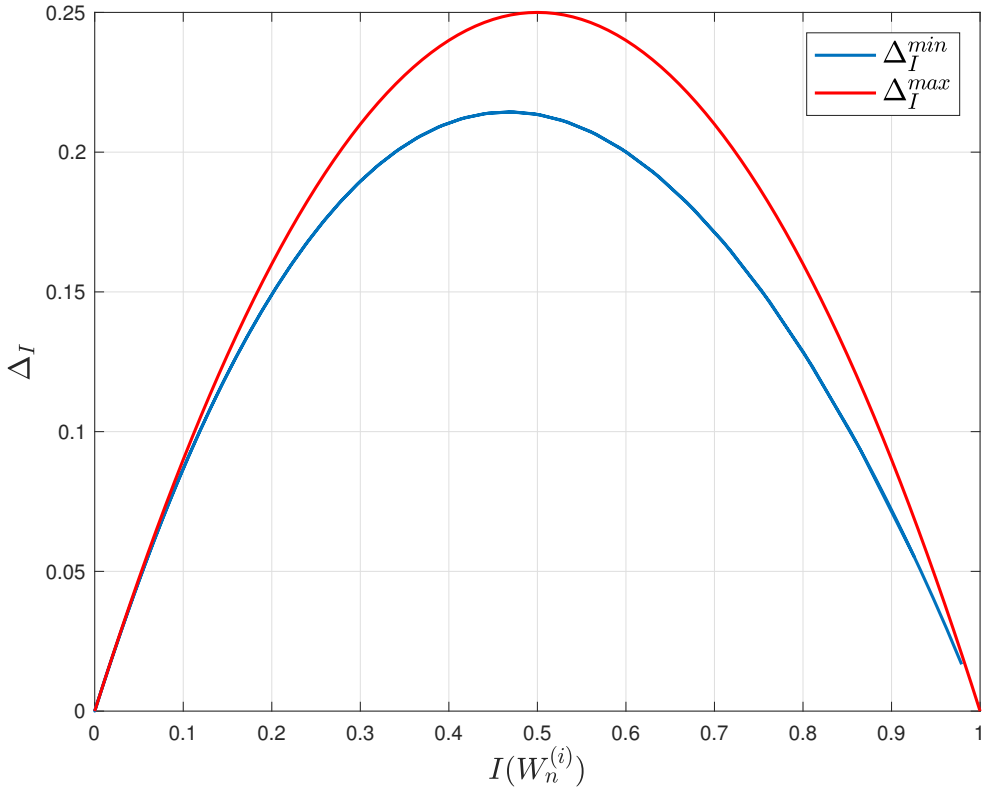


Figure 4.13: Difference between mutual information of $W_n^{(i)}$ and $W_{2n}^{(2i)}$ after one step transformation

2. If $i \in \mathcal{A}_n(x)$, $2i+1 \in \mathcal{A}_{2n}(x)$, and $2i$ can be either in $\mathcal{A}_{2n}(x)$ or $\mathcal{M}_{2n}(x)$;

3. If $i \in \mathcal{M}_n(x)$, $2i$ can be in either $\mathcal{M}_{2n}(x)$ or $\mathcal{B}_{2n}(x)$; $2i+1$ can be in either $\mathcal{M}_{2n}(x)$ or $\mathcal{A}_{2n}(x)$.

From above facts, $|\mathcal{A}_{2n}(x)|$ can be bounded by

$$|\mathcal{A}_{2n}(x)| \leq 2|\mathcal{A}_n(x)| + |\mathcal{M}_n(x)|.$$

Thus $g_{2n}(x)$ is bounded by

$$\begin{aligned} g_{2n}(x) = \frac{|\mathcal{A}_{2n}(x)|}{2nx} &\leq \frac{|\mathcal{A}_n(x)|}{nx} + \frac{|\mathcal{M}_n(x)|}{2nx} \\ &= g_n(x) + \frac{1}{2}t_n(x). \end{aligned}$$

We have a lower bound for $f_{2n}(x)$

$$f_{2n}(x) = 1 - g_{2n}(x) \geq f_n(x) - \frac{1}{2}t_n(x) \tag{4.4}$$

On the other hand, $|\mathcal{B}_{2n}(x)|$ is upper bounded by

$$|\mathcal{B}_{2n}(x)| \leq 2|\mathcal{B}_n(x)| + |\mathcal{M}_n(x)|.$$

Thus $h_{2n}(x)$ is bounded by

$$h_{2n}(x) = \frac{|\mathcal{B}_{2n}(x)|}{2nx} \leq h_n(x) + \frac{1}{2}t_n(x) = f_n(x) - \frac{1}{2}t_n(x),$$

and we obtain an upper bound of $f_{2n}(x)$

$$f_{2n}(x) = h_{2n}(x) + t_{2n}(x) \leq f_n(x) - \frac{1}{2}t_n(x) + t_{2n}(x). \tag{4.5}$$

Combining (4.4) and (4.5), $f_{2n}(x)$ is bounded by

$$f_n(x) - \frac{1}{2}t_n(x) \leq f_{2n}(x) \leq f_n(x) - \frac{1}{2}t_n(x) + t_{2n}(x).$$

By induction, if taking s steps of transformation for any $s > 0$, $f_{2^s n}(x)$ is bounded by

$$\begin{aligned} f_n(x) - \frac{1}{2}\sum_{i=1}^{s} t_{2^{i-1}n}(x) \leq & f_{2^s n}(x) \\ \leq & f_n(x) + \frac{1}{2}\left(\sum_{i=1}^{s} t_{2^{i-1}n}(x)\right) - t_n(x) + t_{2^s n}(x) \end{aligned} \tag{4.6}$$

Thus the difference between $f_{2^s n}(x)$ and $f_n(x)$ is bounded by

$$|f_{2^s n}(x) - f_n(x)| \leq \frac{1}{2}\sum_{i=1}^{s} t_{2^{i-1}n}(x) + t_{2^s n}(x) \tag{4.7}$$

Since almost all bit channels polarize in the limit of blocklength, we have $\lim_{n\to\infty} t_n(x) = 0$. It is shown in [29] that the fraction of intermediate bit channels $t_n(1)$ is bounded by

$$\beta 2^{-bm} \leq t_n(1) \leq \alpha 2^{-am} \tag{4.8}$$

for some $\alpha > 0$ and $\beta > 0$. Here $m = \log_2(n)$. For BEC, the exponents are chosen as $a = 0.2669$ and $b = 0.2786$ [29]. It is easily seen that $t_n(x)$ is upper bounded by

$$t_n(x) = \frac{|\mathcal{M}_n(x)|}{nx} \leq \frac{|\mathcal{M}_n(1)|}{nx} = \frac{1}{x}t_n(1).$$

Hence we have an upper bound for $t_n(x)$

$$t_n(x) \leq \frac{\alpha}{x} n^{-a}. \tag{4.9}$$

$t_{2^s n}(x)$ is upper bounded by

$$t_{2^s n}(x) \leq \frac{\alpha}{x}(2^s n)^{-a} \leq \frac{\alpha}{x} n^{-a}.$$

The term $\sum_{i=1}^{s} t_{2^{i-1}n}(x)$ in (4.6) is bounded by

$$\begin{aligned}\sum_{i=1}^{s} t_{2^{i-1}n}(x) &\leq \sum_{i=1}^{s} \frac{\alpha}{x}(2^{i-1}n)^{-a} \leq \frac{\alpha}{x}\sum_{i=1}^{\infty}(2^{i-1}n)^{-a} \\ &= \frac{\alpha}{x}\frac{1}{1-2^{-a}} n^{-a}.\end{aligned}$$

Then the right hand side of (4.7) is bounded by

$$\frac{1}{2}\sum_{i=1}^{s} t_{2^{i-1}n}(x) + t_{2^s n}(x) \leq \frac{\alpha}{x}\left(\frac{1}{2(1-2^{-a})} + 1\right) n^{-a}. \tag{4.10}$$

Given an arbitrarily small δ, we can find an n' such that

$$\frac{\alpha}{x}\left(\frac{1}{2(1-2^{-a})} + 1\right)(n')^{-a} < \delta \tag{4.11}$$

Thus from (4.7), (4.10) and (4.11) we derive that for any $\delta > 0$, there exists $n' > 0$ such that for all $n > n'$ and $s > 0$,

$$|f_{2^s n}(x) - f_n(x)| < \delta.$$

We have proved that $f_n(x)$ is a Cauchy sequence for any $x \in [0, 1]$. Since every Cauchy

sequence converges in $\mathbb{R}$ [83], the convergence of $f_n(x)$ holds. $\square$

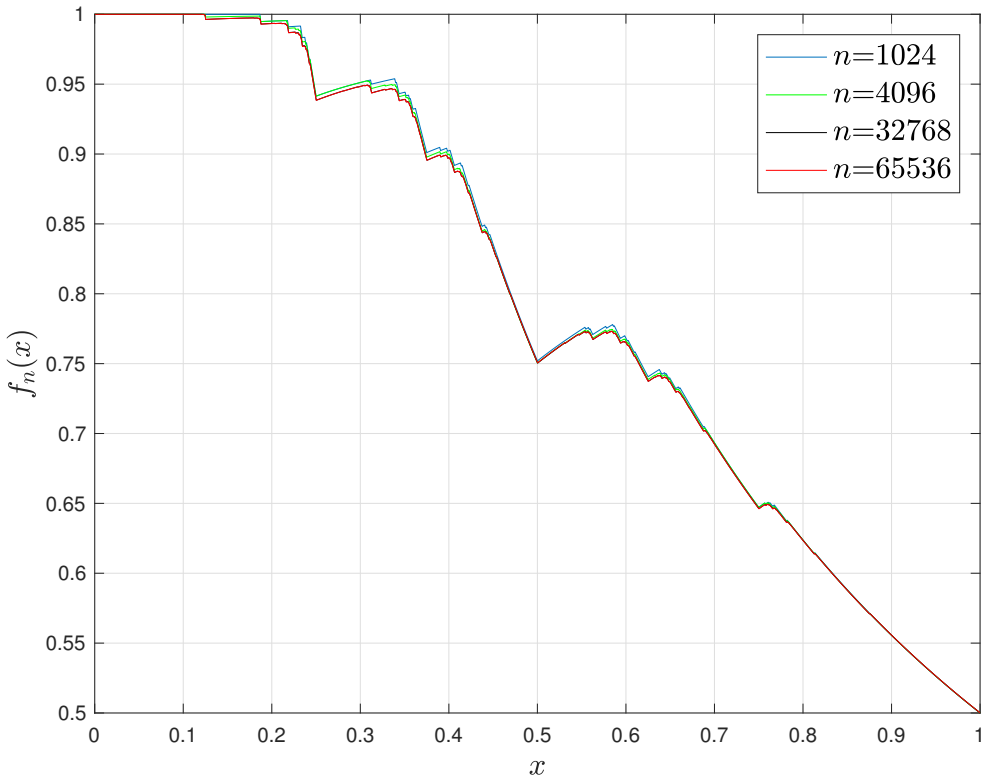


Figure 4.14: Distributions of frozen bits with different code lengths.

We show $f_n(x)$ for rate 1/2 polar codes with different code lengths in Fig. 4.14. It can be seen that the distribution is very close to each other, which validates Theorem 17.

4.6.4 Lower bound of rate improvement

Given the converged distribution of frozen bits in the polar codewords, we show how to compute lower bounds of the improved rate.

4.6.4.1 $[2, 1]$ *outer codes*

We start by considering $[2, 1]$ outer codes. Denote the lower bound of improved rate by ΔR_l. ΔR_l is given in Theorem 18.

Theorem 18. For $[2, 1]$ outer codes, the rate of the polar code can be increased by

$$\begin{aligned}\Delta R_l &= \lim_{n\to\infty} \frac{n(1-R)\left(1-f_n(1-R)\right)}{n} \\ &= (1-R)\left(1-f(1-R)\right)\end{aligned}$$

Proof. Among the first $n(1-R)$ bits, there are $n(1-R)\left(1-f_n(1-R)\right)$ unfrozen bits. This is equal to the number of frozen bits among the last nR bits. Therefore from the polar code perspective, all frozen bits in the last nR bits can be unfrozen and grouped with information bits from the first $n(1-R)$ bits into $[2, 1]$ codes. □

4.6.4.2 $[n_0, t]$ *outer codes*

For general $[n_0, t]$ outer codes, we give the lower bound of improved rate in Theorem 19.

Theorem 19. Let $h(x)$ be an upper bound on $f(x)$. If outer codes can be modeled as length-n_0 codes that correct t erasures, the rate of polar code can be increased by $\Delta R = 1 - R - x_0 h(x_0)$, where $x_0 \in (0, 1)$ satisfies the condition

$$\frac{nx_0(1-h(x_0))}{n_0-t} \geq \frac{n-k-nx_0h(x_0)}{t}. \tag{4.12}$$

Proof. Let $h_n(x)$ upper bounds $f_n(x)$ for each n and assume $h(x) = \lim_{n\to\infty} h_n(x)$. Let $x_0 \in (0, 1)$. Among the first nx_0 bits, there are $nx_0(1 - f_n(x_0)) \geq nx_0(1 - h_n(x_0))$ unfrozen bits. Among the last $n(1 - x_0)$ bits, there are $n - k - nx_0f_n(x_0) \geq n - k - nx_0h_n(x_0)$ frozen bits. If inequality (4.12) holds, we can unfreeze $n - k - nx_0h_n(x_0)$

frozen bits among the last $n(1-x_0)$ bits, and take $nx_0(1-h_n(x_0))$ unfrozen bits among the first nx_0 bits and group them into length-n_0 outer codes. Then the increased rate is $\Delta R_l = \lim_{n\to\infty} \frac{n-k-nx_0h_n(x_0)}{n} = 1 - R - x_0h(x_0)$. □

A simple yet useful choice of $h(x)$ is a piece-wise linear function as follows. Let $\alpha > 0$ and $\beta > \max(1, \alpha)$ be two constant real numbers. Then $h(x)$ can be chosen as

$$h(x) = \begin{cases} 1, \text{ if } 0 < x \leq \dfrac{\beta - 1}{\alpha} \\ -\alpha x + \beta, \text{ if } \dfrac{\beta - 1}{\alpha} < x \leq 1. \end{cases} \tag{4.13}$$

Example 20. We choose $h(x)$ in (4.13) with $\alpha = 0.6$ and $\beta = 1.1$. $h(x)$ is shown as a red curve in Fig. 4.15. If outer codes are $[7, 2]$ codes that correct 2 erasures, $x_0 = 0.59$ satisfies inequality (4.12). Then we can derive $\Delta R_l = 0.06$. If outer codes are $[8, 3]$ codes that correct 3 erasures, we can derive that $\Delta R_l = 0.076$.

Notice that in Theorem 18 and Theorem 19, n bits are partitioned into two segments, where frozen bits in the second segment are then unfrozen to transmit bits that are correctable by unfrozen bits in the first segment. This idea can be generalized by partitioning the n bits into $m + 1 \geq 2$ segments. Let $h(x)$ and $g(x)$ be upper and lower bounds on $f(x)$, respectively. Let $x_1, x_2, \ldots, x_m$ satisfy $\frac{\beta-1}{\alpha} < x_1 < x_2 < \cdots < x_m < 1$. They partition the codewords into $m + 1$ segments.

Theorem 21. If outer codes can be modeled as length-n_0 codes correcting t erasures, we partition the codewords into $m + 1$ segments. The rate of polar code can be increased by

$$\Delta R_l = \lim_{n\to\infty} \frac{\sum_{i=1}^{m} F_{i+1}}{n} = \sum_{i=1}^{m} \left(x_{i+1}g(x_{i+1}) - x_ih(x_i) \right),$$

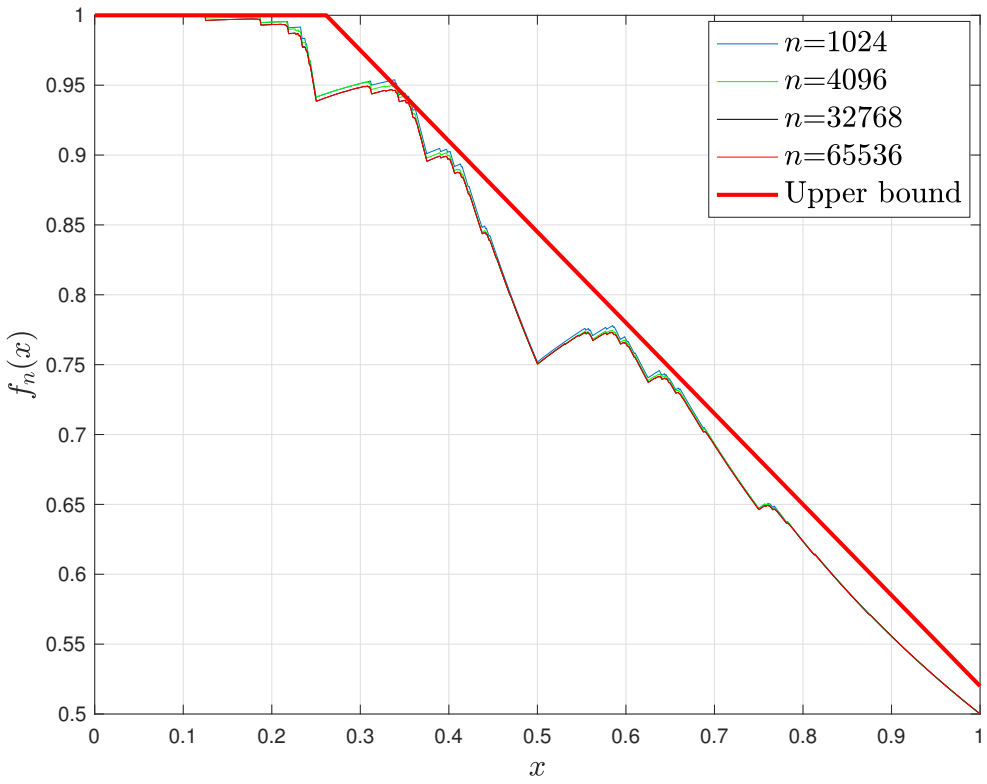


Figure 4.15: An upper bound to the frozen bit distribution

where $x_1, \ldots, x_m$ satisfy $\frac{U_i}{n-t} \geq \frac{F_{i+1}}{t}, \forall i \in [m]$, F_i and U_i are defined as

$$F_i \triangleq nx_i g(x_i) - nx_{i-1} h(x_{i-1}),$$
$$U_i \triangleq nx_i - nx_{i-1} - nx_i h(x_i) + nx_{i-1} g(x_{i-1}),$$

i.e., lower bounds on number of frozen and unfrozen bits in the ith segment.

4.6.5 Maximum rate improvement

Having obtained lower bounds of the improved rate, now we present a new algorithm for mapping the information bits $b_0, b_1, b_2 \cdots$ to the input bits of the polar code's encoder, which maximizes ΔR.

4.6.5.1 Bit-allocation algorithm for optimal rate improvement of polar codes

The algorithm has the following procedures: (1) As initialization, let $\alpha \leftarrow -1$ and $\beta \leftarrow -1$. (2) For $j = 0, 1, 2 \cdots$, if $(j \mod n_0) < n_0 - t$ (which means b_j is one of the first $n_0 - t$ bits of an $[n_0, t]$ code), let $\alpha \leftarrow \min\{i \mid \alpha < i < n, i \notin \mathcal{F}\}$ and then assign b_j to u_α. Otherwise (which means b_j is one of the last t bits of an $[n_0, t]$ code), assign b_j as follows:

1. If $\{i \mid i > \alpha, i > \beta, i \in \mathcal{F}\} \neq \emptyset$, let $\beta \leftarrow \min\{i \mid i > \alpha, i > \beta, i \in \mathcal{F}\}$, and assign b_j to u_β.

2. If $\{i \mid i > \alpha, i > \beta, i \in \mathcal{F}\} = \emptyset$, let $\alpha \leftarrow \min\{i \mid \alpha < i < n, i \notin \mathcal{F}\}$ and then assign b_j to u_α.

The above algorithm ends when there is no more value to assign to α (namely, $\{i \mid \alpha < i < n, i \notin \mathcal{F}\} = \emptyset$). Let $\mathcal{E}$ include all bits denoted by u_β. The algorithm maximizes $|\mathcal{E}|$ and therefore ΔR. It is a greedy algorithm, and has linear time complexity. We present a sketch of proof for its optimality: for every $[n_0, t]$ outer-codeword, call its first $n_0 - t$

bits “pseudo-info bits”, and call its last t bits “pseudo-check bits". Since SC decoder decodes the bits of outer-codewords in the order of their increasing indices, in encoding, it is optimal to allocate the pseudo-info bits to non-frozen bits *codeword by codeword*, without interleaving them. For pseudo-check bits, it is optimal to allocate them to as many frozen bits as possible; and each time we unfreeze a frozen bit to store a pseudo-check bit, it is optimal to choose the feasible frozen bit of the minimum index.

4.6.5.2 Convergence of the improved rate

Theorem 22. Assume a polar code has length n and rate $R = I(W) - \delta$, where δ is an arbitrarily small positive number. Let the polar code be outer coded by a set of $[n_0, t]$ block codes, where n_0 and t scales linearly with n. If the frozen bit distribution $f_n(x)$ converges to some function $f(x)$, the improved rate of polar codes ΔR_n converges as $n \to \infty$.

Sketch of proof. Consider the optimal bit allocation scheme in Section 4.6.5.1. The bits are grouped into $[n_0, t]$ outer codes sequentially, where the first $n_0 - t$ bits are information bits and last t bits are what can be unfrozen. Let nx_i and $ny_i, i \in [1, s]$ be the last information bit and last bit in the ith outer code, respectively. Given the convergence of $f_n(x)$ and linearity of n_0 and t with n, we can show the convergence of x_i and $y_i, i \in [1, s]$, and thus the convergence of s. Since $\Delta R_n = \frac{st+c}{n}$ where $c < t$, the convergence of ΔR_n follows. □

Let ΔR_{max} denote the optimal code-rate improvement achieved by the bit allocation algorithm. We illustrate examples of ΔR_{max} for rate 1/2 polar codes of different lengths in Fig. 4.16, for three different $[n_0, t]$ outer codes: $[2, 1]$, $[7, 2]$ and $[8, 3]$ codes. It can be seen that ΔR_{max} converges as n increases. The convergence of both the lower bound and the maximum improved rate depends on the convergence of $f_n(x)$. A comparison of the lower bound and maximum code rate improvement is shown in Table 4.5, for sufficiently large polar-code length n. We can see that the code-rate improvement by the optimal bit

allocation algorithm exceeds the lower bound to ΔR. Longer outer codes are also simulated to better model the language-based sources. Fig. 4.17 shows ΔR_{max} with $[100, 25]$ and $[100, 40]$ outer codes. We see the convergence of both improved rates.

Table 4.5: A comparison of the lower bound of improved rate and maximum improved rate

	ΔR_{max}	ΔR_l
$[2, 1]$ outer codes	0.181	0.125
$[7, 2]$ outer codes	0.090	0.060
$[8, 3]$ outer codes	0.124	0.076

4.7 Conclusion

Source redundancy is exploited to improve the decoding of polar codes. We propose a joint list decoding scheme for polar codes taking into account the source redundancy using a dictionary. The decoding complexity is of the same order as that of list decoding for stand-alone polar codes. Simulation results show that our scheme significantly outperforms CRC-aided list decoding of polar codes. Theoretically we show the improved rate of polar codes by source redundancy. We first model the general source redundancy as side information, and show the improved coding exponent of channel codes. Then a particular model is given for natural redundancy in languages and we analyze the improved rate of polar codes for that model. We prove the convergence of frozen bit distribution of polar codes, and derive lower bounds of the improved rate. Further we propose an optimal information-bit allocation algorithm to achieve the maximum improved rate. We show the convergence of the improved rate. Both the finite-length performance and improved rate in the asymptotic sense imply the superiority of polar codes in exploring the natural redundancy.

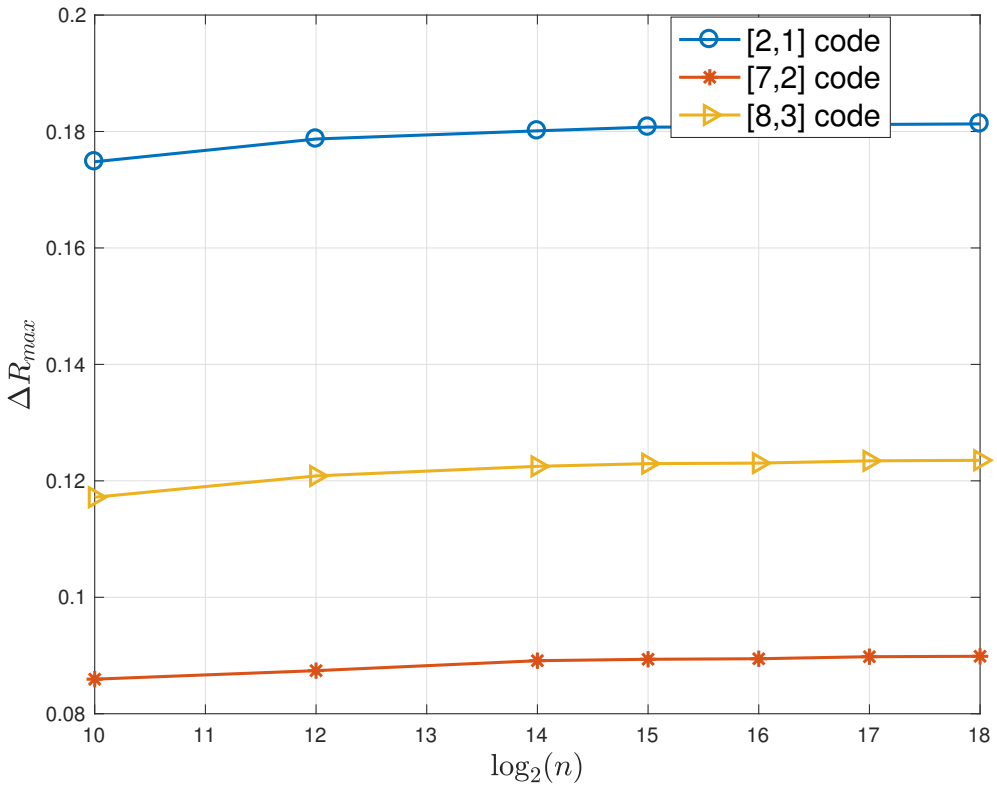


Figure 4.16: Improved rates with different outer $[n_0, t]$ codes. (The $[2, 1]$, $[7, 2]$, $[8, 3]$ outer codes can correct 1, 2 and 3 erasures, respectively.)

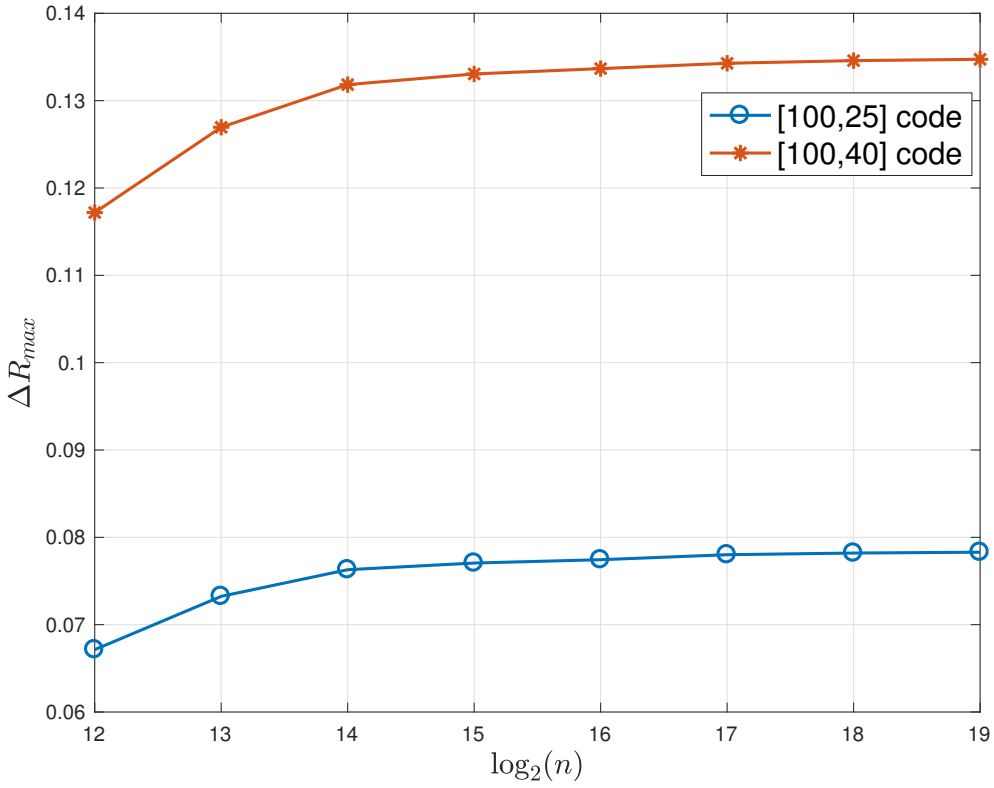


Figure 4.17: Improved rate ΔR_{max} with $[100, 25]$ and $[100, 40]$ outer codes.

5. DESIGN OF IRREGULAR REPEAT ACCUMULATE CODES FOR JOINT DECODING OF SOURCES WITH REDUNDANCY

5.1 Introduction

The work in this chapter is an extension of the work in Chapter 4. In Chapter 4 we have proposed a non-iterative decision feedback scheme for polar codes to exploit the natural redundancy in the source. We have shown substantial performance improvement over the separate decoding scheme [66], and we have also shown the improved rate of polar codes. In this chapter we propose an alternative method – joint iterative decoding with irregular repeat accumulate (IRA) codes–to exploit the natural redundancy.

As a subclass of Turbo and LDPC codes, IRA codes combine many of the favorable attributes of both codes [84]. An important advantage of IRA codes over other LDPC codes is that they are encodable in linear time. They have simple structure but are shown to have good performance. The structure enables flexible adjustment of degree distributions to the desired criterion.

In this chapter we propose a joint iterative decoding scheme for IRA codes, with low encoding and complexity decoding. We propose how to design IRA codes for two models of sources. In the first model, the source redundancy is modeled as a genie telling the correct value with a certain probability. In the second model, the source redundancy is treated as a set of erasure/error correcting block codes. We show that optimized IRA codes can achieve improved decoding thresholds. With fast encoder and low complexity decoder, they are powerful candidates for practical use.

5.2 Joint iterative decoder

The framework of joint iterative decoding with LDPC codes is illustrated in Fig. 5.1. The source is first source encoded by Huffman codes and then channel encoded by LDPC

codes. The encoded sequence is transmitted through a channel. At the decoder side, the source decoder and LDPC decoder work jointly to recover the source. The redundancy in the source is exploited to improve the decoding of LDPC codes. The soft information is exchanged between the two decoders iteratively.

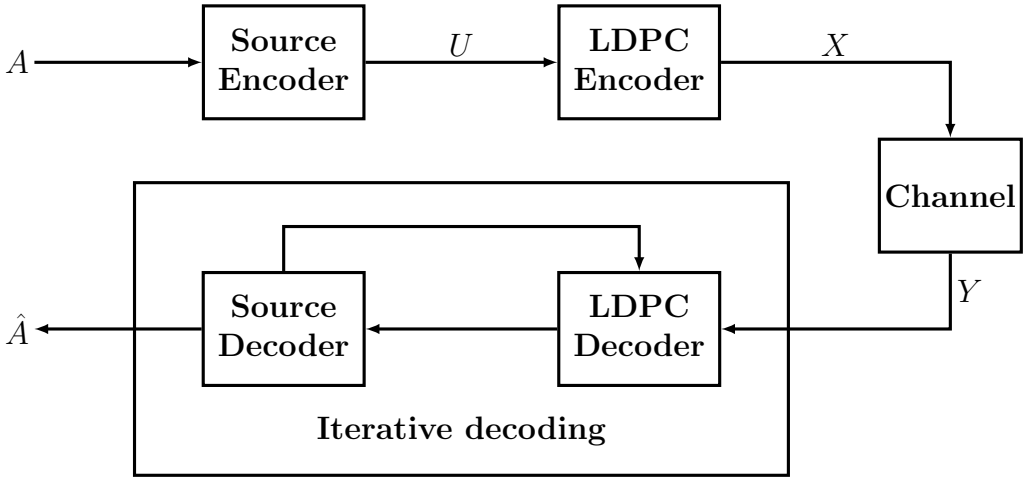


Figure 5.1: Framework of joint iterative decoding with LDPC codes

5.3 Design of IRA codes for BEC

In this section we show how to take advantage of source redundancy in designing IRA codes. The code structure for joint decoding is shown in Fig. 5.2. Assume codewords are transmitted through a BEC. Two models for the language decoder are considered. In the first model, it is assumed that there is a genie that provides the correct value of erased bits with probability q. In the second model, the language decoder acts as a set of $[n_0, t]$ block codes, where n_0 is the length of the code and t is the erasure correction capability. We should point out that $[n_0, t]$ is not a standard notation of an error correction code. We use this notation to put emphasis on the error correction ability of a code with a certain length.

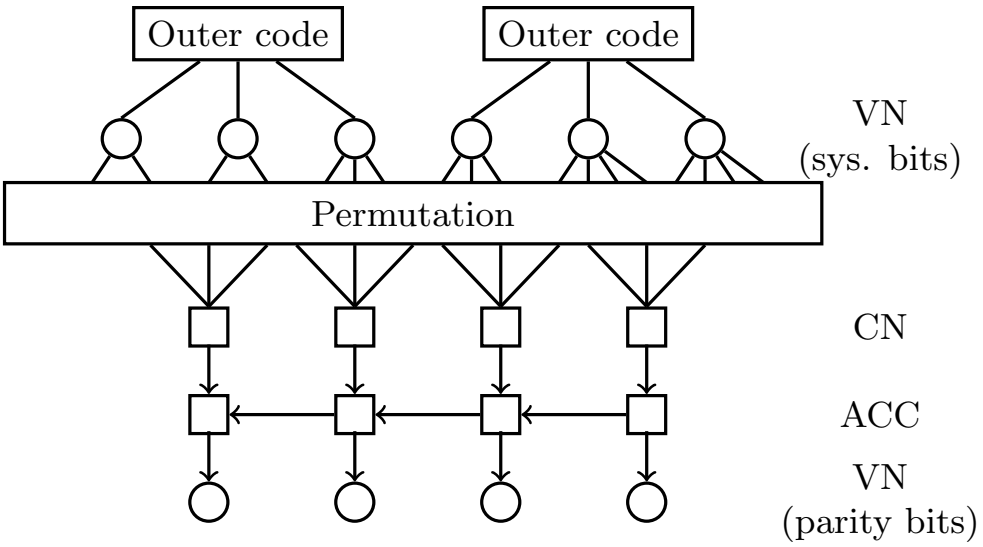


Figure 5.2: Joint decoding structure with IRA codes

IRA codes are composed of repetition codes, parity checks and accumulators [84]. Information variable nodes (VNs) are first encoded by repetition codes and then permuted. The permuted bits are connected to check nodes (CNs) and then encoded by accumulators (ACCs) to generate parity bit nodes. The IRA code ensemble can be specified by two degree profiles,

$$\lambda(x) = \sum_{i=1}^{d_{vm}} \lambda_i x^{i-1},$$
$$\rho(x) = \sum_{i=1}^{d_{cm}} \rho_i x^{i-1},$$

where λ_i and ρ_i are the fraction of edges connected to bit nodes and check nodes with degree i, respectively. d_{vm} and d_{cm} are the maximum degrees of VNs and CNs. We show how to optimize the degree profiles $\{\lambda(x), \rho(x)\}$ to get improved decoding thresholds. Density evolution can be used to keep track of the error probability, assuming the blocklength is infinity and the expanded graph is tree structured.

5.3.1 Genie-aided decoding

Assume that the language decoder acts as a genie that tells each bit the correct value with probability q. We give two methods for designing IRA codes: linear programming and EXIT chart design.

5.3.1.1 Linear programming design

We explore density evolution to find optimal degree distribution of IRA codes for each q. Assume the average degree of bit nodes and parity checks are $\bar{d}_v$ and $\bar{d}_c$, respectively. Let k and m be the number of information bits and parity bits. We have $m = \frac{k\bar{d}_v}{\bar{d}_c}$. For systematic codes, the rate is $R_{sys} = \frac{k}{k+m} = \frac{1}{1+\bar{d}_v/\bar{d}_c}$, while for non-systematic codes, the rate is $R_{nonsys} = \frac{k}{m} = \frac{\bar{d}_c}{\bar{d}_v}$.

In general IRA codes with constant check node degree are shown to be able to achieve the capacity of BEC [84]. To find optimized IRA codes, without loss of generality we assume the degree of check nodes is a constant a, thus $\rho(x) = x^{a-1}$. The rate of systematic codes is

$$R_{sys} = \frac{a}{a + \frac{1}{\sum_{i=2}^{d_{vm}} \lambda_i / i}},$$

and the rate of non-systematic codes is

$$R_{nonsys} = \frac{a}{\frac{1}{\sum_{i=2}^{d_{vm}} \lambda_i / i}}.$$

Next we show how to optimize λ_i given q. Let V_I be an information bit node, V_p be a parity bit node and C be a check node. Let $x_0^{(l)}$, $x_1^{(l)}$, $x_2^{(l)}$, $x_3^{(l)}$ be the erasure probability in the l-th iteration on an edge from V_I to C, C to V_p, V_q to C and C to V_I, respectively.

For systematic IRA codes with genie-aided decoding, the erasure probabilities of dif-

ferent types of messages are:

$$
\begin{aligned}
x_1^{(l)} &= 1 - (1 - x_2^{(l-1)})(1 - x_0^{(l-1)})^a, \\
x_2^{(l)} &= \epsilon x_1^{(l-1)}, \\
x_3^{(l)} &= 1 - (1 - x_2^{(l-1)})^2 (1 - x_0^{(l-1)})^{a-1}, \\
x_0^{(l)} &= (1 - q)\epsilon\lambda(x_3^{(l-1)}),
\end{aligned}
$$

with initial condition: $x_0^{(0)} = \epsilon(1 - q)$. From the above we can derive the fixed point of iterative decoding:

$$
(1 - q)\epsilon\lambda\left(1 - \left(\frac{1 - \epsilon}{1 - \epsilon(1 - x_0)^a}\right)^2 (1 - x_0)^{a-1}\right) = x_0.
$$

If no solution is found in $(0, 1]$ for (5.1), the erasure probability of information bit $x_0^{(l)}$ will converge to 0. Thus the condition for successful BP decoding is

$$
(1 - q)\epsilon\lambda\left(1 - \left(\frac{1 - \epsilon}{1 - \epsilon(1 - x_0)^a}\right)^2 (1 - x_0)^{a-1}\right) < x_0, \forall x_0 \in (0, \epsilon(1 - q)].
$$

For non-systematic IRA codes with genie aided decoding, the erasure probabilities of different messages are:

$$
\begin{aligned}
x_1^{(l)} &= 1 - (1 - x_2^{(l-1)})(1 - x_0^{(l-1)})^a, \\
x_2^{(l)} &= \epsilon x_1^{(l-1)}, \\
x_3^{(l)} &= 1 - (1 - x_2^{(l-1)})^2 (1 - x_0^{(l-1)})^{a-1}, \\
x_0^{(l)} &= (1 - q)\lambda(x_3^{(l-1)})
\end{aligned}
$$

with initial condition: $x_0^{(0)} = 1 - q$. The condition for x_0 to converge to zero is

$$(1-q)\lambda\left(1-\left(\frac{1-\epsilon}{1-\epsilon(1-x_0)^a}\right)^2(1-x_0)^{a-1}\right) < x_0, \forall x_0 \in (0, 1-q].$$

Our goal is to maximize the rate under the constraint that the bit erasure probability of information bits goes to 0 in the limit of large number of iterations. For systematic IRA codes with a fixed check node degree a, the degree optimization can be performed by linear program:

$$\max \sum_{i=2}^{d_{vm}} \lambda_i / i \tag{5.1}$$

$$s.t. \sum_{i=2}^{d_{vm}} \lambda_i = 1, \tag{5.2}$$

$$(1-q)\epsilon\lambda\left(1-\left(\frac{1-\epsilon}{1-\epsilon(1-x)^a}\right)^2(1-x)^{a-1}\right) < x, \forall x \in (0, \epsilon(1-q)]. \tag{5.3}$$

Note that (5.3) can be converted to a set of linear inequalities. For non-systematic IRA codes with genie aided decoding, constraint (5.3) is changed to

$$(1-q)\lambda\left(1-\left(\frac{1-\epsilon}{1-\epsilon(1-x)^a}\right)^2(1-x)^{a-1}\right) < x, \forall x \in (0, 1-q].$$

Fig. 5.3 shows BP decoding thresholds of IRA codes with genie-aided decoding. The LDPC code rate is $1/2$. For both systematic and non-systematic codes, the maximum left degree is set to be 40. The check node degree a is chosen to be 4 or 5. The upper bound in the figure is the threshold corresponding to the overall rate of the code $R_{joint} = (1-q)R$, where R is the channel code rate. This is in accordance with the maximum rate in Fig. 4.11. We observe that non-systematic IRA codes have much better thresholds than systematic codes. When the check node degree $a = 4$, the threshold of non-systematic codes is very

close to the upper bound when $q \geq 0.3$. Codes with $a = 5$ slightly outperform codes with $a = 4$ when $q \geq 0.5$.

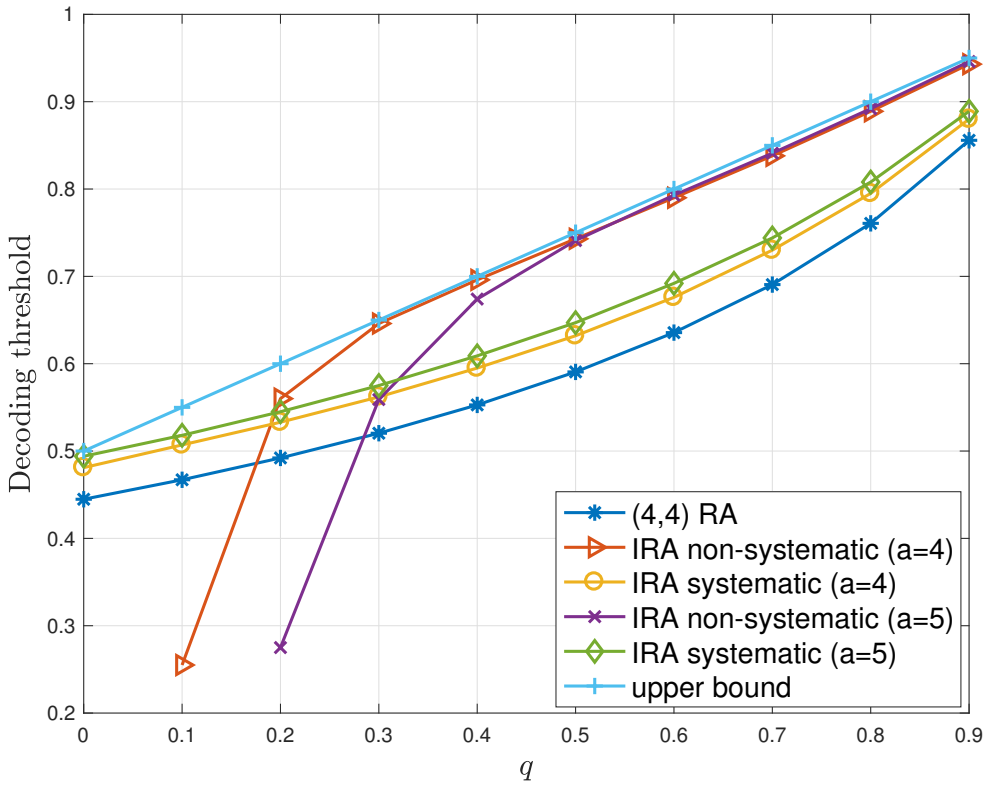


Figure 5.3: Threshold of joint decoding of IRA codes in BEC. IRA codes are optimized in terms of q.

5.3.1.2 EXIT chart design

Linear programming has some limitation on the joint design of codes. If the output erasure probability of outer codes depends on the input erasure probability, the degree optimization cannot be done by linear program. It is a good alternative to density evolution. EXIT chart is a more general technique to design graph codes [85], it doesn't

matter whether the optimization program is linear or nonlinear. As we have shown that non-systematic codes have better decoding thresholds, we mainly consider design for non-systematic codes.

Since non-systematic codes with all check node degrees greater than 1 cannot start converging [84], we introduce a nonzero fraction of degree-1 CNs. For simplicity we design IRA codes with biregular CNs, that is, $\rho(x) = \rho_1 + \rho_{d_{cm}} x^{d_{cm}-1}$. The non-systematic code rate R is

$$R = \bar{d}_c/\bar{d}_v = \frac{\sum_i \lambda_i/i}{\sum_i \rho_i/i}.$$

Let I_A be the mutual information between the bits on the decoder graph edges and the a priori LLR values from the received vector, and let I_E be the mutual information between bits on the edges and the extrinsic LLR values. The degree-i VN transfer curve can be derived as

$$I_{E,VN}(I_{A,VN}, i, q) = 1 - (1 - I_{A,VN})^{i-1}(1 - q).$$

Thus the average mutual information is

$$\bar{I}_{E,VN} = \sum_{i=1}^{d_{vm}} \lambda_i I_{E,VN}(I_{A,VN}, i, q).$$

The CN should take into account of the ACC shown in Fig. 5.2. Assume the channel is BEC(ϵ). The mutual information of the ACC for BEC is:

$$I_{E,ACC}(I_{A,ACC}, \epsilon) = (\frac{1-\epsilon}{1-\epsilon I_{A,ACC}})^2. \tag{5.4}$$

The mutual information of degree-i CN can be computed as

$$I_{E,CN}(I_{A,CN}, I_{E,ACC}, i) = I_{A,CN}^{i-1} \cdot I_{E,ACC}. \tag{5.5}$$

The average mutual information of CN is

$$\bar{I}_{E,CN}(I_{A,ACC}, \epsilon) = \sum_{i=1}^{d_{cm}} \rho_i I_{E,CN}(I_{A,CN}, I_{E,ACC}, i). \tag{5.6}$$

If we plot $I_{E,CN}$ v.s. $I_{A,CN}$ and $I_{A,VN}$ v.s. $I_{E,CN}$ in the same figure, as long as there is an open tunnel between the two curves, the BP decoder will decode successfully. Notice that increasing the fraction of degree-1 CNs lifts the EXIT curve to a wider convergence tunnel. By optimizing the degree of VNs such that there is an open tunnel between the two curves, we can design codes with desired decoding thresholds.

Example 23. Assume the channel is BEC(0.5). The code rate is fixed to $R = 0.5$. Consider that each bit can be decoded by a genie with $q = 0.5$. We set $\rho(x) = x^2$. The optimized degree distribution is $\lambda(x) = 0.100x + 0.025x^2 + 0.200x^3 + 0.675x^{14}$. It can achieve the decoding threshold $\epsilon^* = 0.75$, which is equal to the upper bound $1 - R(1-q)$. Fig. 5.4 shows the EXIT chart of the designed code.

5.3.2 t erasure correcting outer codes

In this subsection we show how to design non-systematic IRA codes when the source is modeled as a set of $[n_0, t]$ outer codes. Since the output erasure probability of outer codes depends on the input erasure probability, the degree optimization cannot be done by linear programming. EXIT chart design method is used instead. The erasure probability

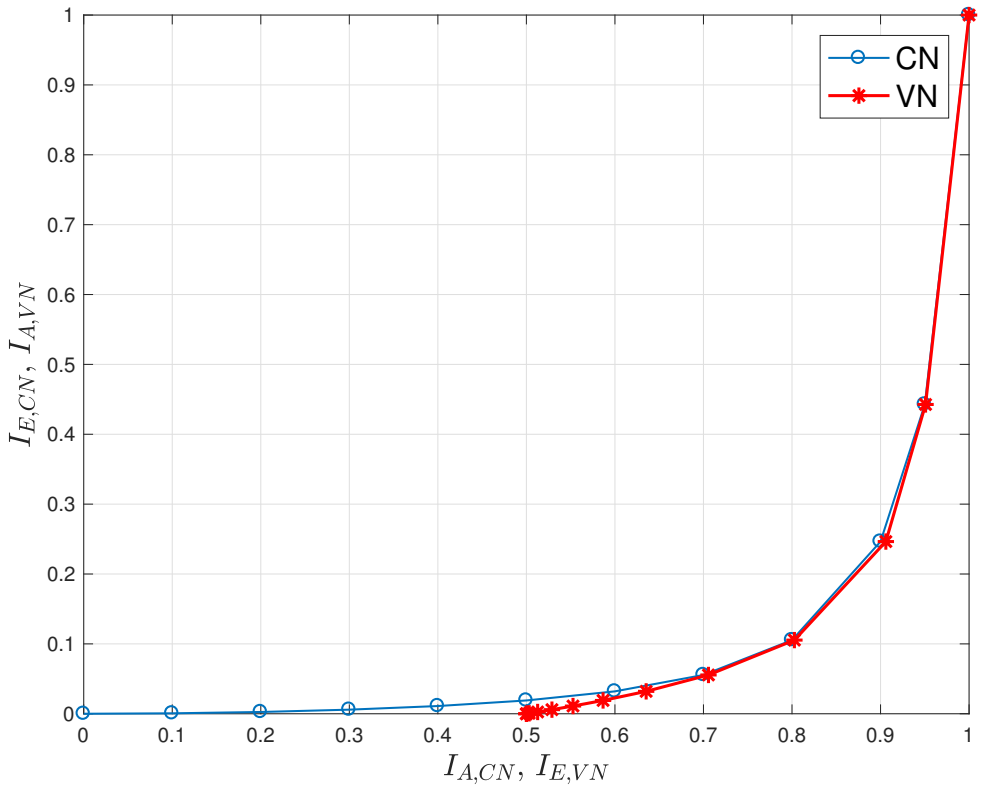


Figure 5.4: EXIT chart for non-systematic IRA code with genie aided decoding, $q = 0.5$. $\epsilon^* = 0.75$.

P_e of an $[n_0, t]$ outer code is

$$P_e(n_0, t, I_A) = 1 - \sum_{i=0}^{t} \binom{n_0}{i} (1 - I_A)^i (I_A)^{n_0 - i}.$$

The mutual information of degree-i VN is derived as

$$I_{E,VN}(I_{A,VN}, i, n_0, t) = 1 - (1 - I_{A,VN})^{i-1} P_e(n_0, t, I_{A,VN}).$$

Thus the average mutual information of VN is

$$\bar{I}_{E,VN} = \sum_{i=1}^{d_{vm}} \lambda_i I_{E,VN}(I_{A,VN}, i, n_0, t). \tag{5.7}$$

The mutual information of CN can be computed from (5.5) and (5.6). IRA codes are designed by optimizing $\lambda(x)$ for desired decoding thresholds.

Example 24. Assume the channel is BEC(0.5), and the code rate is fixed to $R = 0.5$. If the source is modeled as $[2, 1]$ outer codes, we can design codes with decoding threshold $\epsilon^* = 0.68$. Fig. 5.5 shows the EXIT curves of the designed code with decoding threshold $\epsilon^* = 0.68$. The optimized degree distributions are: $\rho(x) = 0.077 + 0.923x^2$, and $\lambda(x) = 0.280x + 0.011x^2 + 0.100x^3 + 0.609x^{29}$.

Example 25. If the source is modeled as $[7, 2]$ outer codes, we can design code with decoding threshold $\epsilon^* = 0.56$. The optimized degree distributions are $\rho(x) = 0.077 + 0.923x^2$, and $\lambda(x) = 0.280x + 0.016x^2 + 0.704x^{14}$.

From above examples we see that the joint decoding threshold of IRA codes ϵ^* exceeds 0.5, the threshold without source redundancy. We conclude that by optimizing the degree distribution of IRA codes, improved decoding thresholds can be achieved with the help of source redundancy.

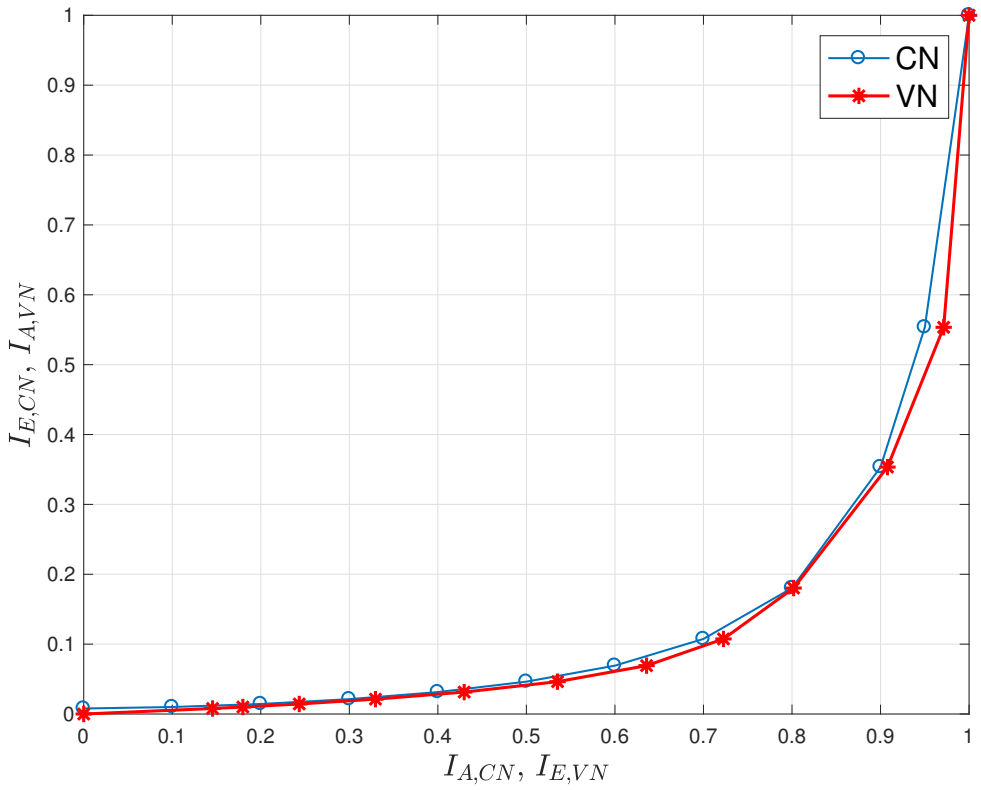


Figure 5.5: EXIT curves for non-systematic IRA code with $[2, 1]$ outer codes. $\epsilon^* = 0.68$.

5.4 Design of IRA codes for AWGN channel

If codewords are transmitted through an AWGN channel, we can assume that the extrinsic messages are conditionally Gaussian and it is enough to keep track of the mean and variance of the LLR messages. In this subsection we show how to design IRA codes for AWGN channel.

5.4.1 Genie-aided decoder

If the language decoder is modeled as a genie-aided decoder with probability q to tell the correct value of each information bit, we have

$$1 - q = Q\left(\sqrt{\frac{2E_b}{N_0}}\right).$$

The variance of the output LLR message from the language decoder can be equivalently considered as [85]

$$\sigma_q^2 = 8\frac{E_b}{N_0}. \tag{5.8}$$

Let $J(\sigma_l)$ be the mutual information between the input X and output Y with LLR variance σ_l^2. $J(\sigma_l)$ is expressed as [85]

$$J(\sigma_l) = I(X;Y) = 1 - \int_{-\infty}^{\infty} \frac{e^{-(\epsilon-\sigma_l^2/2)^2/2\sigma_l^2}}{\sqrt{2\pi\sigma_l^2}} \cdot \log_2\left(1+e^{-\epsilon}\right) d\epsilon.$$

The EXIT function of a degree i information bit node is

$$I_{E,VN} = J\left(\sqrt{(i-1)[J^{-1}(I_{A,VN})]^2 + \sigma_q^2}\right). \tag{5.9}$$

From (5.8) and (5.9) we have $I_{E,VN}$ as a function of $I_{A,VN}$.

$$I_{E,VN} = J\left(\sqrt{(i-1)[J^{-1}(I_{A,VN})]^2 + 4[Q^{-1}(1-q)]^2)}\right).$$

The averaged value of $I_{E,VN}$ can be obtained from (5.7). The EXIT function of a degree a check node is

$$I_{E,CN} = 1 - J\left(\sqrt{(a-1)[J^{-1}(1-I_{A,CN})]^2 + [J^{-1}(1-I_{E,ACC})]^2}\right). \tag{5.10}$$

The EXIT function of the accumulator of an IRA code is derived as

$$I_{E,ACC} = \left(\frac{1-\epsilon}{1-\epsilon I_{A,ACC}}\right)^2. \tag{5.11}$$

where $\epsilon = 1 - C(E_b/N_0)$ and $C(E_b/N_0)$ is the capacity of AWGN channel at SNR E_b/N_0. Further the EXIT function of the input accumulator is

$$I_{A,ACC} = 1 - J\left(\sqrt{a}J^{-1}(1-I_{A,CN})\right). \tag{5.12}$$

From (5.10), (5.11) and (5.12) we can derive $I_{E,CN}$ as a function of $I_{A,CN}$. The averaged value of $I_{E,CN}$ is computed from (5.6).

5.4.2 t error correcting outer codes

If the language decoder is modeled as an $[n_0, t]$ error correcting code that can correct t errors, the variance of the output LLR message from the language decoder σ_q^2 can be computed as follows. First we have

$$J^{-1}(I_{A,VN}) = \sqrt{\frac{8E_b}{N_0}}.$$

Let the error probability of each information bit be denoted as p. It is derived as

$$p = Q\left(\sqrt{\frac{2E_b}{N_0}}\right) = Q\left(\frac{1}{2}J^{-1}(I_{A,VN})\right).$$

The error probability of the output of language decoder is

$$P_e = 1 - \sum_{i=0}^{t} p^i (1-p)^{n_0 - i}.$$

The corresponding E_b/N_0 for P_e is

$$\frac{E_b}{N_0} = \frac{[Q^{-1}(P_e)]^2}{2}.$$

Thus σ_q is derived as

$$\sigma_q = \sqrt{\frac{8E_b}{N_0}} = 2Q^{-1}(P_e). \tag{5.13}$$

The mutual information $I_{E,VN}$ as a function of $I_{A,VN}$ is obtained from (5.9) and (5.13). The mutual information $I_{E,CN}$ as a function of $I_{A,CN}$ is derived from (5.10), (5.11) and (5.12). By optimizing the degree distribution $\lambda(x)$ and ensuring an open tunnel between the two EXIT curves we are able to design IRA codes with best decoding thresholds.

5.5 A comparison of joint LDPC decoding with joint polar decoding

We show in Fig. 5.6 a performance comparison between joint polar and joint LDPC decoding schemes in AWGN channel. Source codes are Huffman codes. Both polar and LDPC codes have length $n = 8192$ and rate $R = 0.923$. In the joint LDPC decoding scheme, RA codes with VN degree 3 and CN degree 36 are constructed with progressive edge growth (PEG) algorithm. From the figure we observe that polar list decoding with list size $L = 8$ outperforms joint LDPC decoding.

Asymptotically we compare the improved rates of the two schemes. The rate of both channel codes is $R = 0.5$. The source is modeled as a set of $[n_0, t]$ outer codes. In Table 5.1 a comparison of improved rates is provided for the two schemes. We see that if source codes are modeled as short length outer codes, the improved rates of two schemes are very close. However, if source codes are modeled as long length outer codes, the improved rate of joint LDPC codes tends to be larger than that of joint polar codes. The reason is that the improved rate of polar codes depends on the frozen bit distribution. For long length outer codes, frozen bits of polar codes cannot be efficiently grouped into outer codes.

Table 5.1: A comparison of improved rates for two joint decoding schemes

	$[2, 1]$ codes	$[7, 2]$ codes	$[8, 3]$ codes
Joint polar	0.18	0.09	0.12
Joint LDPC	0.18	0.08	0.10

From above results, we believe that joint polar decoding has advantages over joint LDPC decoding. First, joint polar decoding scheme provides a systematic way to improve performance, by increasing the list size. Second, it is natural and efficient to combine two tree structures of polar codes and the dictionary, thus polar codes are able to make better use of the soft information provided by the source redundancy. Finally, error propagation is more effectively reduced by polar joint list decoding. With sequential decoding nature, we can even reorder information bits in the word level before encoding to further reduce error propagation.

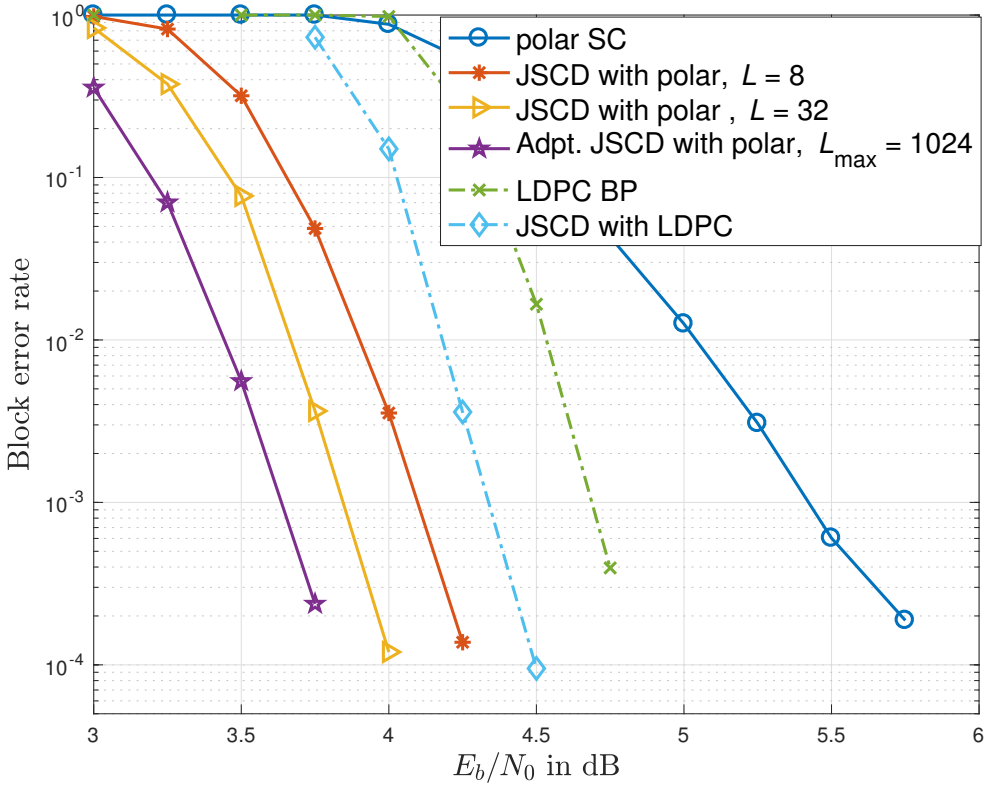


Figure 5.6: Performance comparison of joint polar decoding and joint LDPC decoding schemes.

5.6 Conclusion

In the joint source-channel decoding scheme, source redundancy is explored to improve the decoding thresholds under iterative decoding. We consider IRA codes as channel codes, and propose schemes to design IRA codes for two different source models and show that they achieve improved decoding thresholds.

6. CONCLUSIONS

In this dissertation, we have studied the potential of polar codes in the finite length regime, and have applied polar codes to the joint source-channel decoding problem. The main contributions of this dissertation are summarized in the following.

In Chapter 3, we address the problem of improving the finite-length performance of polar codes. We propose interleaved concatenation schemes of polar codes with binary codes including BCH and convolutional codes. Interleaved concatenation with long constraint length convolutional codes is an effective way to leverage the fact that polarization increases the cutoff rate of the channel. We show that both schemes outperform the conventional decoding schemes, and they both achieve exponential error decay rate. Furthermore, we propose a soft-output multistage iterative decoder for the concatenated Conv-polar codes to further improve the performance. From a more practical point of view, we propose using rate compatible punctured convolutional codes as outer codes which enable the same encoder and decoder for outer codes and are more hardware friendly.

In Chapter 4, we propose a joint source-channel decoding scheme for polar codes for language-based sources. The source redundancy is exploited to improve the performance polar codes. Joint list decoding is employed as a non-iterative decision feedback approach. Simulation results demonstrate that our scheme significantly outperforms list decoding of CRC-aided polar codes. Theoretically, we propose models for the language decoder and show how the rate of polar codes can be improved by exploiting the source redundancy while achieving vanishing error probability. We show that the rate improvement depends on the frozen bit distribution of polar codes. We prove that the distribution converges to a limit distribution. We derive a lower bound of improved rate based on the distribution. We further propose an optimal bit allocation scheme to achieve the maximum improved

rate. Both simulation results and theoretical analysis lead us to conclude that polar code is a good candidate to exploit the source redundancy in the joint decoding scheme.

In Chapter 5, we continue to study the joint-source channel decoding problem but focus on another type of decoding scheme – joint iterative decoding. We show how to design irregular repeat accumulate codes when the source redundancy is available at the decoder. Different models of language decoder are considered. We show that the improved decoding thresholds can be achieved by our design. Non-systematic IRA codes are able to achieve better decoding threshold compared to the systematic ones in the joint decoding scheme. Further we provide a comparison of the joint polar decoding and joint LDPC decoding schemes. The joint polar decoding tends to outperform joint LDPC decoding. It exploits the inherent structure in the source redundancy more efficiently.

REFERENCES

[1] A. Jiang, Y. Li, and J. Bruck, "Enhanced error correction via language processing," in *Proc. NMVW*, 2015.

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

[3] N. Hussami, S. B. Korada, and R. Urbanke, "Performance of polar codes for channel and source coding," in *Proc. IEEE ISIT*, pp. 1488–1492, 2009.

[4] S. B. Korada, *Polar codes for channel and source coding*. PhD thesis, Ecole Polytechnique Fédérale de Lausanne, 2009.

[5] E. Arikan, "Source polarization," in *Proc. IEEE ISIT*, pp. 899–903, 2010.

[6] S. B. Korada and R. L. Urbanke, "Polar codes are optimal for lossy source coding," *IEEE Trans. Inf. Theory*, vol. 56, no. 4, pp. 1751–1768, 2010.

[7] E. Abbe and E. Telatar, "Polar codes for the m-user multiple access channel," *IEEE Trans. Inf. Theory*, vol. 58, no. 8, pp. 5437–5448, 2012.

[8] E. Şaşoğlu, E. Telatar, and E. M. Yeh, "Polar codes for the two-user multiple-access channel," *IEEE Trans. Inf. Theory*, vol. 59, no. 10, pp. 6583–6592, 2013.

[9] H. Mahdavifar, M. El-Khamy, J. Lee, and I. Kang, "Achieving the uniform rate region of general multiple access channels by polar coding," *IEEE Trans. Commun.*, vol. 64, no. 2, pp. 467–478, 2016.

[10] N. Goela, E. Abbe, and M. Gastpar, "Polar codes for broadcast channels," *IEEE Trans. Inf. Theory*, vol. 61, no. 2, pp. 758–782, 2015.

[11] M. Mondelli, S. H. Hassani, I. Sason, and R. L. Urbanke, "Achieving marton's region for broadcast channels using polar codes," *IEEE Trans. Inf. Theory*, vol. 61, no. 2, pp. 783–800, 2015.

[12] H. Mahdavifar and A. Vardy, "Achieving the secrecy capacity of wiretap channels using polar codes," *IEEE Trans. Inf. Theory*, vol. 57, no. 10, pp. 6428–6443, 2011.

[13] E. Şaşoğlu and A. Vardy, "A new polar coding scheme for strong security on wiretap channels," in *Proc. IEEE ISIT*, pp. 1117–1121, 2013.

[14] E. Arıkan and I. Telatar, "On the rate of channel polarization," in *Proc. IEEE ISIT*, pp. 1493–1495, 2009.

[15] S. H. Hassani, K. Alishahi, and R. L. Urbanke, "Finite-length scaling for polar codes," *IEEE Trans. Inf. Theory*, vol. 60, no. 10, pp. 5875–5898, 2014.

[16] D. Goldin and D. Burshtein, "Improved bounds on the finite length scaling of polar codes," *IEEE Trans. Inf. Theory*, vol. 60, no. 11, pp. 6966–6978, 2014.

[17] M. Mondelli, S. H. Hassani, and R. L. Urbanke, "Unified scaling of polar codes: Error exponent, scaling exponent, moderate deviations, and error floors," *IEEE Trans. Inf. Theory*, vol. 62, no. 12, pp. 6698–6712, 2016.

[18] A. Eslami and H. Pishro-Nik, "On finite-length performance of polar codes: stopping sets, error floor, and concatenated design," *IEEE Trans. Commun.*, vol. 61, no. 3, pp. 919–929, 2013.

[19] E. Arıkan, "Channel combining and splitting for cutoff rate improvement," *IEEE Trans. Inf. Theory*, vol. 52, no. 2, pp. 628–639, 2006.

[20] I. Tal and A. Vardy, "List decoding of polar codes," in *Proc. IEEE ISIT*, pp. 1–5, 2011.

[21] I. Tal and A. Vardy, "How to construct polar codes," 2011.

[22] R. Mori and T. Tanaka, "Performance of polar codes with the construction using density evolution," *IEEE Commun. Lett.*, vol. 13, no. 7, pp. 519–521, 2009.

[23] B. Li, H. Shen, and D. Tse, "An adaptive successive cancellation list decoder for polar codes with cyclic redundancy check," *IEEE Commun. Lett.*, vol. 16, no. 12, pp. 2044–2047, 2012.

[24] E. Arikan, "A performance comparison of polar codes and Reed-Muller codes," *IEEE Commun. Lett.*, vol. 12, no. 6, 2008.

[25] S. Kumar and H. D. Pfister, "Reed-Muller codes achieve capacity on erasure channels," *arXiv:1505.05123*, 2015.

[26] S. Kudekar, S. Kumar, M. Mondelli, H. D. Pfister, E. Sasoglu, and R. Urbanke, "Reed-Muller codes achieve capacity on erasure channels," *IEEE Trans. Inf. Theory*, 2017.

[27] E. Arikan, "A survey of Reed-Muller codes from polar coding perspective," in *Proc. IEEE ITW*, pp. 1–5, 2010.

[28] U. U. Fayyaz and J. R. Barry, "Low-complexity soft-output decoding of polar codes," *IEEE J. Sel. Areas Commun.*, vol. 32, no. 5, pp. 958–966, 2014.

[29] S. H. Hassani, K. Alishahi, and R. Urbanke, "On the scaling of polar codes: II. the behavior of un-polarized channels," in *Proc. IEEE ISIT*, pp. 879–883, 2010.

[30] T. Tanaka and R. Mori, "Refined rate of channel polarization," in *Proc. IEEE ISIT*, 2010.

[31] R. G. Gallager, *Information Theory and Reliable Communication*. New York: Wiley, 1968.

[32] J. M. Wozencraft, "Sequential decoding for reliable communication," 1957.

[33] R. Fano, "A heuristic discussion of probabilistic decoding," *IEEE Trans. Inf. Theory*, vol. 9, no. 2, pp. 64–74, 1963.

[34] K. Zigangirov, "Some sequential decoding procedures," *Probl. Peredachi Inf.*, vol. 2, no. 4, pp. 13–25, 1966.

[35] F. Jelinek, "Fast sequential decoding algorithm using a stack," *IBM J. of Res. and Dev.*, vol. 13, no. 6, pp. 675–685, 1969.

[36] R. Blahut, "The missed message of the cutoff rate," in *Proc. ISTC*, pp. 449–451, Sept. 2010.

[37] G. D. Forney Jr. and G. Ungerboeck, "Modulation and coding for linear Gaussian channels," *IEEE Trans. Inf. Theory*, vol. 44, no. 6, pp. 2384–2415, 1998.

[38] E. Arikan, "An upper bound on the cutoff rate of sequential decoding," *IEEE trans. inf. theory*, vol. 34, no. 1, pp. 55–63, 1988.

[39] I. Jacobs and E. Berlekamp, "A lower bound to the distribution of computation for sequential decoding," *IEEE Trans. Inf. Theory*, vol. 13, no. 2, pp. 167–174, 1967.

[40] G. D. Forney Jr., "Convolutional codes III. Sequential decoding," *Inf. Control*, vol. 25, no. 3, pp. 267–297, 1974.

[41] J. Massey, "Capacity, cutoff rate, and coding for a direct-detection optical channel," *IEEE Trans. Commun.*, vol. 29, no. 11, pp. 1615–1621, 1981.

[42] M. Mondelli, S. H. Hassani, and R. Urbanke, "Scaling exponent of list decoders with applications to polar codes," in *Proc. IEEE ITW*, pp. 1–5, 2013.

[43] M. Bakshi, S. Jaggi, and M. Effros, "Concatenated polar codes," in *Proc. IEEE ISIT*, pp. 918–922, 2010.

[44] M. Seidl and J. B. Huber, "Improving successive cancellation decoding of polar codes by usage of inner block codes," in *Proc. IEEE ISTC*, pp. 103–106, 2010.

[45] J. Guo, M. Qin, A. Guillén i Fàbregas, and P. H. Siegel, "Enhanced belief propagation decoding of polar codes through concatenation," in *Proc. IEEE ISIT*, pp. 2987–2991, 2014.

[46] H. Mahdavifar, M. El-Khamy, J. Lee, and I. Kang, "Performance limits and practical decoding of interleaved Reed-Solomon polar concatenated codes," *IEEE Trans. Commun.*, vol. 62, no. 5, pp. 1406–1417, 2014.

[47] P. Trifonov and P. Semenov, "Generalized concatenated codes based on polar codes," in *ISWCS*, pp. 442–446, 2011.

[48] U. Wachsmann, R. F. H. Fischer, and J. Huber, "Multilevel codes: theoretical concepts and practical design rules," *IEEE Trans. Inf. Theory*, vol. 45, pp. 1361–1391, Jul 1999.

[49] Y. Wang, K. R. Narayanan, and Y.-C. Huang, "Interleaved concatenations of polar codes with BCH and convolutional codes," *IEEE J. Sel. Areas Commun.*, vol. 34, no. 2, pp. 267–277, 2015.

[50] F. Jelinek and J. Cocke, "Bootstrap hybrid decoding for symmetrical binary input channels," *Inf. Control*, vol. 18, no. 3, pp. 261–298, 1971.

[51] Y. Wang and K. R. Narayanan, "Concatenations of polar codes with outer BCH codes and convolutional codes," in *Proc. 2014 Allerton Conf. Commun., Control Comput.*, pp. 813–819.

[52] M. Sudan, "Lecture on BCH codes," *Algorithmic Introduction to Coding Theory*, 2001.

[53] G. D. Forney Jr., "Convolutional codes II. Maximum-likelihood decoding," *Inf. Control*, vol. 25, no. 3, pp. 222–266, 1974.

[54] A. Graell i Amat, G. Montorsi, and S. Benedetto, "Design and decoding of optimal high-rate convolutional codes," *IEEE Trans. Inf. Theory*, vol. 50, no. 5, pp. 867–881, 2004.

[55] I. Bocharova and B. Kudryashov, "Rational rate punctured convolutional codes for soft-decision Viterbi decoding," *IEEE Trans. Inf. Theory*, vol. 43, no. 4, pp. 1305–1313, 1997.

[56] H. H. Ma and J. K. Wolf, "On tail biting convolutional codes," *IEEE Trans. Commun.*, vol. 34, pp. 104–111, 1986.

[57] T. K. Moon, *Error Correction Coding: Mathematical Methods and Algorithms*. John Wiley and Son, 2005.

[58] J. Hagenauer, "Rate-compatible punctured convolutional codes (RCPC codes) and their applications," *IEEE Trans. Commun.*, vol. 36, no. 4, pp. 389–400, 1988.

[59] R. H. Morelos-Zaragoza, *The Art of Error Correcting Coding*. John Wiley and Sons, 2006.

[60] D. Haccoun and G. Begin, "High-rate punctured convolutional codes for Viterbi and sequential decoding," *IEEE Trans. Commun.*, vol. 37, no. 11, pp. 1113–1125, 1989.

[61] G. Bégin, D. Haccoun, and C. Paquin, "Further results on high-rate punctured convolutional codes for Viterbi and sequential decoding," *IEEE Trans. Commun.*, vol. 38, no. 11, pp. 1922–1928, 1990.

[62] A. Jiang, P. Upadhyaya, E. Haratsch, and J. Bruck, "Error correction by natural redundancy for long term storage," in *Proc. NVMW*, 2017.

[63] C. E. Shannon, "Prediction and entropy of printed English," *Bell system technical journal*, vol. 30, no. 1, pp. 50–64, 1951.

[64] A. Jiang, Y. Li, and J. Bruck, "Error correction through language processing," in *Proc. IEEE ITW*, pp. 1–5, 2015.

[65] J. Luo, Q. Huang, S. Wang, and Z. Wang, "Error control coding combined with content recognition," in *Proc. 8th International Conference on Wireless Communications and Signal Processing*, pp. 1–5, 2016.

[66] Y. Wang, M. Qin, K. R. Narayanan, A. Jiang, and Z. Bandic, "Joint source-channel decoding of polar codes for language-based sources," in *Proc. IEEE Globecom*, pp. 1–6, 2016.

[67] Y. Wang, K. R. Narayanan, and A. Jiang, "Exploiting source redundancy to improve the rate of polar codes," in *Proc. IEEE ISIT*, 2017.

[68] Y. Wang, A. Jiang, and K. R. Narayanan, "Modeling and analysis of joint decoding of language-based sources with polar codes," in *Proc. NVMW*, 2017.

[69] P. Upadhyaya and A. Jiang, "LDPC decoding with natural redundancy," in *Proc. NVMW*, 2017.

[70] Y. Li, Y. Wang, A. Jiang, and J. Bruck, "Content-assisted file decoding for nonvolatile memories," in *Proc. 46th Asilomar Conference on Signals, Systems and Computers*, pp. 937–941, 2012.

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

[72] J. Hagenauer, "Source-controlled channel decoding," *IEEE Trans. Commun.*, vol. 43, no. 9, pp. 2449–2457, 1995.

[73] R. Bauer and J. Hagenauer, "On variable length codes for iterative source/channel decoding," in *Proc. 2001 Data Compress. Conf.*, pp. 273–282, 2001.

[74] L. Guivarch, J.-C. Carlach, and P. Siohan, "Joint source-channel soft decoding of Huffman codes with Turbo-codes," in *Proc. DCC 2000*, pp. 83–92.

[75] M. Jeanne, J.-C. Carlach, and P. Siohan, "Joint source-channel decoding of variable-length codes for convolutional codes and Turbo codes," *IEEE Trans. Commun.*, vol. 53, no. 1, pp. 10–15, 2005.

[76] Z. Peng, Y.-F. Huang, and D. J. Costello Jr, "Turbo codes for image transmission-a joint channel and source decoding approach," *IEEE J. Sel. Areas Commun.*, vol. 18, no. 6, pp. 868–879, 2000.

[77] A. N. Kim, S. Sesia, T. Ramstad, and G. Caire, "Combined error protection and compression using Turbo codes for error resilient image transmission," in *Proc. Int. Conf. Image Process. (ICIP)*, vol. 3, pp. III–912–15, 2005.

[78] C. Poulliat, D. Declercq, C. Lamy-Bergot, and I. Fijalkow, "Analysis and optimization of irregular LDPC codes for joint source-channel decoding," *IEEE Commun. Lett.*, vol. 9, no. 12, pp. 1064–1066, 2005.

[79] L. Pu, Z. Wu, A. Bilgin, M. W. Marcellin, and B. Vasic, "LDPC-based iterative joint source-channel decoding for JPEG2000," *IEEE Trans. Image Process.*, vol. 16, no. 2, pp. 577–581, 2007.

[80] P. F. Brown, P. V. Desouza, R. L. Mercer, V. J. D. Pietra, and J. C. Lai, "Class-based n-gram models of natural language," *Computational linguistics*, vol. 18, no. 4, pp. 467–479, 1992.

[81] E. Fredkin, "Trie memory," *Communications of the ACM*, vol. 3, no. 9, pp. 490–499, 1960.

[82] "Word frequency data," *http://www.wordfrequency.info/free.asp*.

[83] W. Rudin *et al.*, *Principles of mathematical analysis*, vol. 3. McGraw-Hill New York, 1964.

[84] H. Jin, A. Khandekar, and R. McEliece, "Irregular repeat-accumulate codes," in *Proc. 2nd Int. Symp. Turbo codes and related topics*, pp. 1–8, Citeseer, 2000.

[85] S. Ten Brink, G. Kramer, and A. Ashikhmin, "Design of low-density parity-check codes for modulation and detection," *IEEE Trans. Commun.*, vol. 52, no. 4, pp. 670–678, 2004.