IMPROVED SUCCESSIVE CANCELLATION DECODING OF POLAR CODES

ABDELKAREIM ABULGAASEM ALRTAIMI

SEPTEMBER 2021

IMPROVED SUCCESSIVE CANCELLATION DECODING OF POLAR CODES

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED
SCIENCES OF
ÇANKAYA UNIVERSITY

BY
ABDELKAREIM ABULGAASEM ALRTAIMI

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE
DEGREE OF
DOCTOR OF PHILOSOPHY
IN
THE DEPARTMENT OF
ELECTRONIC AND COMMUNICATION ENGINEERING

SEPTEMBER 2021

Title of the Thesis: Improved Successive Cancellation Decoding of Polar Codes

Submitted by **ABDELKAREIM ABULGAASEM ALRTAIMI**

Approval of the Graduate School of Natural and Applied Sciences, Çankaya University.

Assoc. Prof. Dr. Ziya ESEN
Director

I certify that this thesis satisfies all the requirements as a thesis for the degree of Doctor of Philosophy.

Prof. Dr. Sıtkı Kemal İDER
Head of Department

This is to certify that we have read this thesis and that in our opinion it is fully adequate, in scope and quality, as a thesis for the degree of Doctor of Philosophy.

Assoc. Prof. Dr. Orhan GAZİ
Supervisor

Examination Date: 07.09.2021

Examining Committee Members

Prof. Dr. Yahya Kemal BAYKAL	(Çankaya University)	_____
Assoc. Prof. Dr. Orhan GAZİ	(Çankaya University)	_____
Assist. Prof. Dr. Serap ARPALİ	(Çankaya University)	_____
Assist. Prof. Dr. Özgür ERGÜL	(Gazi University)	_____
Assist. Prof. Dr. Muhsin GÖKÇE	(TED University)	_____

STATEMENT OF NON-PLAGIARISM

I hereby declare that all information in this document has been obtained and presented following academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name	:	Abdelkareim
Last Name		Alrtaimi
Signature	:	
Date	:	07.09.2021

ABSTRACT

Improved Successive Cancellation Decoding of Polar Codes

Alrtaimi, Abdelkareim Abulgaasem

Ph.D., Department of Electronic and Communication Engineering

Supervisor: Assoc. Prof. Dr. Orhan GAZİ

September 2021

In this thesis, we propose improved successive cancellation polar code decoding algorithms. Polar codes are fragile to error propagation. Considering this issue, in our first proposal, we consider $LR = 1$, for which decision is made for the favor of bit 0 in classical successive cancellation algorithm, and propose multi-SC decoders, where we consider more than one decoders working in parallel and these decoders make opposite decisions for $LR = 1$. The proposed technique provides a flexible configuration and leads to the pruning of unnecessary path searching operations, which provide low complexity compared to successive cancelation list decoding algorithm. Multi-Parallel SC decoding shows a significant performance improvement compared with the original SC decoding and its performance is comparable to that of the successive cancellation list decoding algorithm.

In our next proposal, we propose a method for the iterative decoding of polar codes replacing the unreliable received samples with randomly generated samples. In this method, first, a straight decoding operation is performed for the received frame and CRC check is performed, and if it is not satisfied, the received symbols are passed through virtual random channels before they are sent to the polar decoders employing successive cancellation decoding algorithm. When the received symbols are passed through virtual random channels, randomly generated noise is added to the inputs of the VRCs falling into a threshold interval, which contains unreliable information about the transmitted polar code-bit before they are sent to the polar decoder. For the decoded sequence, if the CRC check is not satisfied, a different randomly generated noise sequence is added to the unreliable inputs and the decoding operation is repeated. This

procedure is repeated until a predefined maximum iteration number as long as CRC is not satisfied.

Keywords: Polar codes, multi-parallel SCD, BEC, iterative polar decoding, virtual random channels (VRCs), AWGN, and Rayleigh fading channels.

ÖZ

Geliştirilmiş Ardışık Giderim Algoritması ile Kutup Kodlarının Çözümlenmesi

ALRTAIMI, Abdelkareim Abulgaasem
Doktora, Elektronik ve Haberleşme Mühendisliği
Tez Yöneticisi: Doç. Dr. Orhan GAZİ

Eylül 2021

Bu tezde, geliştirilmiş ardışık iptal (SC) kutupsal kod çözme algoritmaları öneriyoruz. Çözümlenen bitin yanlış değerde seçilmesi diğer bitlerin de çözümlenmesini etkileyecektir. Bu duruma kutup kodlarında hata yayılımı ismi verilmektedir. Bu konuyu göz önünde bulundurarak ilk önerimizde, klasik ardışık giderim algoritmasında bit 0 lehine karar verilen $LR = 1$ durumunu ele alıyoruz ve birden fazla kod çözücünün paralel olarak çalıştığı çoklu SC kod çözücüleri öneriyoruz. Bu kod çözücüler $LR = 1$ için zıt kararlar verir. Önerilen teknik esnek bir konfigürasyon sağlar ve ardışık giderim liste kod çözme algoritmasına kıyasla düşük karmaşıklık sağlayan ve bunu da gereksiz patika arama işlemlerinin elenmesiyle sağlayan bir algoritmadır. Çoklu paralel SC kod çözme, orijinal SC kod çözme ile karşılaştırıldığında önemli bir performans artışı gösterir ve performansı, ardışık iptal liste kod çözme algoritmasının performansına yaklaşmaktadır.

Bir sonraki önerimizde, alınan sinyaldeki güvenilir olmayan örneklerin rastgele üretilen örneklerle değiştirilmesiyle kutupsal kodların yinelemeli bir şekilde çözülmesi için bir yöntem öneriyoruz. Bu yöntemde, önce alınan örnekler ile klasik kod çözme işlemi gerçekleştirilir ve CRC kontrolü yapılır ve bu sağlanmazsa alınan örnekler sanal rastgele kanalından (VRC) geçirilir. Alınan örnekler sanal rasgele kanalından geçirildiğinde, VRC'lerin girişinde içerisinde yeterli bilgi içermeyen örnekler rasgele örneklerle değitirilirler. Daha sonra VRC çıktısı SC çözücüye gönderilir. Çözücünün çıktısı için CRC kontrolü yapılır, ve eğer CRC kontrolü sağlanmazsa alınan örnekler

tekrar VRC'ye gönderilir ve işlemler tekrar edilir. Bu prosedür, CRC karşılanmadığı sürece önceden tanımlanmış bir maksimum yineleme sayısına kadar tekrarlanır.

Anahtar Kelimeler: Kutup kodlar, Çoklu paralel SCD, BEC, yinelemeli polar kod çözme, sanal rastgele kanallar (VRC'ler), AWGN ve Rayleigh sönümleme kanalları.

ACKNOWLEDGEMENTS

I would like to great fully thanks to my supervisor the associate professor Dr. Orhan Gazi for his mentoring, distinctive guidance, serious suggestions and his continual encouragement during the development of this dissertation.

I would like to express my sincere respect to the Government of Libya, represented by the Ministry of Higher Education and Scientific Research, for their financial support throughout my studies.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

TABLES

LIST OF ABBREVIATIONS

SC	Successive Cancellation
BEC	Binary Erasure Channel
AWGN	Additive White Gaussian Noise
SCL	Successive Cancellation List
DMC	Discrete Memoryless Channel
FER	Frame Error Rate
BER	Bit Error Rate
BLER	Block Error Rate
SNR	Signal to Noise Ratio
SCS	Successive Cancellation Stack
LDPC	Low-Density Parity-Check code
B-DMC	Binary- Discrete Memoryless Channel
BCJR	Bahl-Cocke-Jelinek-Raviv
CRC	Cyclic Redundancy Check
LR	Likelihood Ratio
CD	Compact Disk
DVD	Digital Video Disc
ML	Maximum Likelihood
XOR	Exclusive OR
LLR	Log Likelihood Ratio
BP	Belief Propagation
BCH	Bose-Chaudhuri-Hocquenghem
BMS	Binary input Memoryless Symmetric
BSC	Binary Symmetric Channel

CHAPTER 1

1. INTRODUCTION

Channel coding problem is one of the main topics of information theory. Error control codes are used for reliable data transmission in the presence of noise and interference. The recently introduced polar codes [1] are used to correct the transmission errors in communication systems. The polar codes are constructed in a recursive manner. The polar codes can achieve the channel capacity of discrete-input memoryless symmetric channels. The construction of polar codes is based on specific procedure of recursive encoding, and to this end, the transmission channel is synthesized from N virtual channels, where the N represents the length of the code.

1.1 Successive Cancellation (SC) Decoding

Depending on the number of recursions used for the construction of polar codes, the sub-channels get either low reliability or high reliability, the data stream are allocated to the channel with high reliability. Sub-channels may not be fully polarized when the recursions number is low for a finite length of polar codes, and this may cause poor error correction performance.

To tackle this issue, different approaches have been proposed in the literature, where several studies investigated rely on either modified the kernels of the recursive encoding procedure, in respect of increasing the rate of polarization [2], [3], or enhanced versions of the SC decoder [4], [5], [6], in order to increase its ability to deal with incompletely polarized channels.

1.2 Successive Cancellation List (SCL) Decoding.

The SC-List (SCL) decoding procedure was introduced in [4]. Different from the SC algorithm, where only a single candidate code word is searched, SCL saves L's most

reliable candidate code words at every step, and it greatly enhances the performance of the error correction for codeword lengths between short and moderate length. The SCL decoding mechanism, based on the same principle as the maximum likelihood (ML) decoding, performs well at a high signal-to-noise-ratio (SNR). Researchers have investigated the potential utility of applying the SCL decoding, along with cyclic redundancy check (CRC) code and its concatenation to determine the transmitted data [4]. It is shown that CRC concatenated polar codes employing SCL decoding outperforms the existing error-correcting codes including turbo codes as well as low-density parity-check (LDPC) codes.

However, the severe disadvantage of the SCL decoder is that it requires large memory storage with high run time. The latency increases as the list size increases. To alleviate the large computational complexity successive cancellation stack (SCS) decoding is proposed in [6], however, this method requires large memory storage.

1.3 Successive Cancellation Stack (SCS) Decoding

This algorithm provides the error-correcting performance similar to the SCL algorithm with a complexity that varies with channel conditions. The complexity of SCS is the same as SCL at high channel noise, and as the channel noise decreases, the SCS complexity approaches to that of SC.

1.4 Successive Cancellation Bit-Flipping Decoding

The SC-based decoding algorithm, called SC Bit-Flipping, employing iterative decoding, to reduce the computation complexity is introduced in [5]. In the BF decoding algorithm after SC decoding operation, candidate bits, which have low |LR| values, are selected, and in sequel, the bit with the smallest |LR| is flipped and decoding is performed assuming that CRC is not satisfied. In each decoding attempt, only one bit, $w = 1$, is flipped in the decoded code word. After each decoding operation, CRC check is performed, and the

decoding operation with bit flipping is repeated until either CRC is satisfied or maximum iteration number, T_{max}, is reached. This decoding algorithm achieves an error-correcting performance close to that of the SC-List with $L = 2$. Its complexity is $O(N)$, which is the memory complexity of the SC algorithm, and it has an average computational complexity of $O(N \log N)$ at high SNR.

1.5 Improve Versions of Successive Cancellation Bit-Flipping Decoding

Multi-bit flip SC decoding is introduced in [7] where a critical set is constructed as a bit-flip index set and a new metric is utilized to determine the priority of bits to be flipped, aiming to find the true first error faster, to reduce the computational complexity. The maximum number of bit-flip equals the level of the critical set [7].

The bit-flipping methods employed with SC-list (SCL) decoder for polar codes are introduced in [8, 9]. In [8], the characteristics of path splitting states in SCL decoding are analyzed and used to determine the type of path splitting for bit flipping, and a revised critical set (RCS) is defined to indicate the bit to be flipped in case the original CA-SCL decoding fails. Row weight-based successive cancellation list flipping, RWB-SCL-Flipping, algorithm is proposed in [9] where the correlation between error distribution of information bits and their associated row weights in the generator matrix is used. The information bits associated with small row weights in the generator matrix are deduced as error-prone, and they are selected as the flipping bits with a high priority.

In spite of many attempts that have been made with the purpose of improving the polar decoders in terms of performance and complexity [10–18], it is still an open issue to explore an efficient decoding algorithm that can achieve an adequate frame error rate performance, FER, with low complexity, particularly when polar codes with finite length are considered.

1.6 Thesis Contribution

In this thesis work, we introduce new efficient low complexity algorithms to improve the performance polar decoders. We focus on the LR=1 case, and an high performance low complexity improved SC decoding algorithm is proposed.

Besides, we also propose a new technique that is used in AWGN and Rayleigh fading channels by adding random noise to the unreliable soft information, falling into a threshold interval, in the received sequence.

1.6.1 High-Performance Low Latency Parallel Successive Cancelation Decoder Structures

Polar codes are decoded using successive cancellation (SC) algorithm where likelihood ratios (LRs) for data bits are calculated sequentially, and decisions are made using the calculated LRs. During the decoding of an information bit, the decision results for the predecessor bits are used, and a wrongly decided predecessor bit has a negative effect on the accurate calculation of the LR for the information bit being decoded. In the SC algorithm, when LR=1, the information bit is decoded as $\widehat{\boldsymbol{u}}_{\boldsymbol{i}} = \mathbf{0}$, however, such a decision has a 50% of chance of being correct. In this thesis, we propose improved polar decoders utilizing a number of SC decoders. We consider the case of $LR = 1$ and propose polar decoder structures for the more accurate calculation of the LRs of the successor bits.

1.6.2 Improving the Performance of Polar Decoders Using Virtual Random Channels.

In this part, we propose the use of virtual random channels (VRCs) to improve the performance of successive cancellation decoding algorithms used to decode the polar codes. For this purpose, we consider CRC concatenated information sequences before polar encoding operation. We introduce a decoder structure called the Noise-Aided Iterative SC decoder, where the received signal is decoded in an iterative manner using

VRCs. The proposed system shows comparable performance to that of the state-of-the-art CA-SCL, besides it has low complexity for high SNR values.

1.7 Thesis Organization

The thesis is structured as follows: Chapter 2 presents a brief literature review of channel coding. The concept of channel polarization for polar codes, and the calculation of the Bhattacharyya parameters, $Z(W)$, in the channel polarization process for different channels are explained in Chapter 3.

In Chapter 4, we explain the decoding of polar codes using the SC decoding algorithm, which can be illustrated by the factor graph resembling a Butterfly-based structure. The Butterfly-based structure can be explained using the tree structure. In the logarithmic version of the SC algorithm LLRs (log-likelihood ratios) are used to reduce the complexity of the calculations.

In Chapter 5, we introduce M-Parallel SC Decoding algorithm. In SC decoding of polar codes, whenever $LR(\hat{u}_i) = 1$ the decision is made as $\hat{u}_i = 0$. However, in such an approach, we have a 50% chance of making a correct decision. A wrong decision will certainly have negative effects on the determination of the successive bits. Considering this issue, we propose multi-parallel structures employing SC decoding algorithms.

Chapter 6 deal with the use of virtual random channels (VRCs) to improve the performance of successive cancellation decoding algorithms used to decode the polar codes. For this purpose, we consider CRC concatenated information sequences before polar encoding operation. We introduce a decoder structure called the *Noise-Aided Iterative* SC decoder, where the received signal is decoded in an iterative manner using VRCs.

CHAPTER 2

LITERATURE REVIEW

Information theory was born with the publication of Shannon's paper a mathematical theory of communication in 1948. Shannon in his paper showed the criteria for the reliable transmission of the data over noisy channels and stated the need of channel codes for reliable data transmission but he did not propose any channel coding method for the reliable transmission. Shannon in his paper drew the frontiers of the reliable communication [19]. Following the Shannon's paper, a new research area in communication society called channel coding appeared and channel coding became one of the main concepts of communication systems

The philosophy of channel coding lies on the addition of the guard bits to the information bits and using these guard or parity bits for the error correction procedure. Addition of the parity bits to the information bits increases the bit vector length, and this leads to an increase in the Hamming distance among codewords. Increased Hamming distance among codewords decreases the probability of error for the determination of the transmitted bit vector, i.e., codeword, at the receiver side. The channel coding is a crucial operation in any communication applications, such as internet, and cellular phones. Channel encoding is a vital part of the modern communication systems. A typical modern communication system is depicted in Fig 1.1.

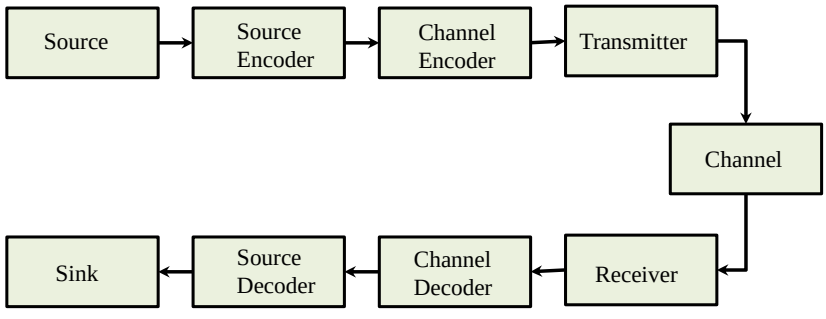

Figure 1.1 The main components of digital communication system.

In communication theory, a communication channel is characterized by the conditional channel probabilities.

Let X and Y denote the input and output alphabet of a communication channel. The channel is described by $W: X \rightarrow Y$ along with its transition probabilities $W(y \mid x)$ such that $x \in X$ is transmitted through the channel, and y is the received signal at the receiver side. The channel input alphabet can be considered as the range set of a random variable, and similarly the channel output alphabet can be considered as the range set of a random variable assuming that we use discrete channel. In case, we use a continuous channel, such as Gaussian channel, we use intervals for input and output alphabets, i.e., we have infinitely many inputs and outputs. $W(y \mid x)$ can be considered as the conditional probability mass or density function, considering both discrete and continuous channels, of two random variables. Shannon in his paper defined the channel capacity, $C(W)$, which indicates the maximum reliable transmission speed per-transmission or per-second, considering the transmission of a single symbol or the transmission of a number of symbols per-second. When the transmission speed R is smaller than the channel capacity, i.e., when $R < C(W)$, then it is possible to achieve error free transmission using the channel codes.

Channel codes can be classified as block and convolutional channel codes. Block channel codes are the subject of linear algebra. Block channel codes are vector subsets, and the aim of channel coding is to deploy linear binary codes that satisfy the properties of linear algebra with a large minimum distance. In case transmitted data is corrupted during the transmission, the codes are used to recover the erroneous bits altered by the noise. Decoding operation at the receiver side can be classified as hard and soft decoding. In the hard decoding operation, each bit decision is made after its decoding operation. On the other hand, in soft decoding operation, we calculate 0 and 1 the probabilities for the bit being decoded, and usually after all the 0 and 1 probabilities for all the bits are calculated, decisions are made for all the bits. Minimum distance, which is the smallest Hamming weight of all codewords of a code, is directly related to the code's error correction capability.

The decoding operation can be considered as selecting the codeword that is closest to the received word (after hard-decision). Therefore, it is desirable to employ codes with large minimum distances so that a maximum number of erroneous bits are corrected.

Hamming introduced the first algebraic codes and he called them Hamming codes [20]. The Hamming codes are single error correction codes. After Hamming codes, there are other various efficient algebraic codes such as as Golay codes, BCH codes [21, 22], Reed-Muller codes [23, 24], and the Reed-Solomon codes [25] are introduced. These codes are use in different applications such as in modems, CDs as well as DVDs. To achieve efficient performance, codes with large clock lengths must be constructed and implemented in practice. The improvement of computer resources made it possible to design codes with large block lengths with complex decoding algorithms.

Elias introduces product codes in [26] where large codes are constructed by combining two or more shorter length codes. Code concatenation was proposed by Forney [27] where data word is encoded using the first code C_1, and the output of C_1 is encoded by C_2. Forney's findings illustrate that by selecting the component codes in an appropriate manner, the error probability can decay almost exponentially with a decoding algorithm.

Elias presented convolutional codes in [28]. Convolutional codes are different from block codes in terms of encoding and decoding. The Viterbi algorithm is used for the decoding of block codes [29]. The BCJR algorithm can also be for the decoding of convolutional codes [30]. Those algorithms run on linear complexity within the desired block length. If convolutional codewords have large lengths, then it is possible to get probability of errors vanishing exponentially. Nevertheless, the run time of the decoding can increase exponentially at such large lengths. Fano's algorithm for the decoding of convolutional codes is introduced in [31].

Low-density parity-check (LDPC) codes are discovered during the '60s [32]. The parity check matrices of these codes have very few non-zero entries. These matrices have a constant number of non-zero entries on each row and column.

Gallager showed that these codes have a non-zero relative distance. He also proposed a low-complexity iterative decoding algorithm. Unfortunately, due to the lack of computational resources at that time, the power of these codes and the decoding algorithms were not realized. MacKay and Neal constructed codes based on sparse matrices [34] and observed that they perform very well using a low complexity belief propagation algorithm. It was later noticed that these codes were a special case of LDPC codes and that the decoding algorithm was similar to the probabilistic decoding suggested by Gallager. Around the same time, Sipser and Spielman [35] constructed expander codes and came up with a simple decoding algorithm that could correct a linear fraction of adversarial errors.

The invention of turbo codes by Berrou, Glavieux, and Thitimajshima [33] was a breakthrough in the practice of coding. Turbo codes achieved rates close to the capacity with a linear complexity-decoding algorithm. The turbo codes are constructed by concatenating two convolutional codes in parallel with a random bit interleaver in between. Turbo codes are decoded using iterative decoding algorithms. BCJR algorithm is adopted for the decoding of turbo codes. Wiberg, Loeliger, and Kotter [36, 37] unified turbo codes and LDPC codes using code graphs, and the turbo decoding algorithm, the belief propagation algorithm of MacKay and Neal, and the probabilistic decoding of Gallager turned out to be different incarnations of the same algorithm. The study leaded to a bridge between sparse graph codes and other fields like machine learning, statistical mechanics, and computer science.

The success of turbo codes and the subsequent rediscovery of LDPC codes aroused the interest in LDPC codes and message passing algorithms. Some of the important contributions to the message-passing algorithms were done in a series of papers by Luby, Mitzenmacher, Shokrollahi, Spielman, and Stemann [38, 39, 40, 41]. In [38, 40], the authors analyzed a suboptimal decoder known as “peeling decoder” for the binary erasure channel (BEC). They constructed codes for the BEC, which achieve capacity using the peeling decoder. Later, in [42], the peeling decoder was formulated on a tree.

In [43], Richardson and Urbanke developed density evolution [42] and a class of message-passing algorithms. This class includes the important belief propagation algorithm. By combining density evolution by belief propagation with optimization techniques, channel codes were constructed achieving the capacity of Gaussian channel to within 0.0045dB [44]. Many techniques over turbo and LDPC codes have been proposed which empirically achieve rates close to capacity for various channels.

In this thesis, we propose low complexity capacity achieving decoding methods for polar codes introduced by Arıkan [1]. Polar codes are the primary class of error-correcting codes that achieve the capacity of communication channels with “low encoding and decoding complexity”, and performance of polar codes can be proven mathematically.

CHAPTER 3

POLAR CODING

3.1 Useful definitions and notations

We will define the notations that will be used throughout this chapter. Let W be the channel on which the transmission takes place; X and Y the channel input and output alphabets, $W(y|x)$, $x\epsilon X, y\epsilon Y$ are the transition probabilities. The operator $\oplus$ is modul-2 addition, $\otimes$ is the Kronecker product defined as:

$$A \otimes B = \begin{bmatrix} A_{11}B & \cdots & A_{1n}B \\ \vdots & \ddots & \vdots \\ A_{m1}B & \cdots & A_{mn}B \end{bmatrix} \tag{3.1}$$

where A, B and, $A \otimes B$ are matrices of sizes $m \times n, q \times g$ and $mq \times ng$. The Kronecker power $A^{\otimes N}$ is defined as:

$$A^{\otimes n} = A \otimes A^{\otimes(n-1)} \tag{3.2}$$

for all $n \geq 1$, with, $A^{\otimes 0} \triangleq 1$.

For vectors, we use the following notations: a_1^N denotes the row vector $(a_1, \ldots, a_N)$ with $N = 2^n$, n being a positive integer; a_i^j: the subvector defined by $(a_i, \ldots, a_j)$ with $1 \leq i \leq j \leq N$; $a_{i,e}^j$: the subvector $(a_k : i \leq k \leq j; k\ even)$ and $a_{i,o}^j$: $(a_k : i \leq k \leq j; k\ odd)$; $A \subset \{1, \ldots, N\}$, a_A: the subvector $(a_k : k \in A)$.

Definitions:

The capacity of a discrete transmission channel is defined as the maximum value of the mutual information between its input and output considering all possible probability distributions. The symmetrical capacity (in bits/s) of a binary discrete memoryless channel, B-DMC, is defined as [1]

$$I(W) \triangleq \sum_{y\epsilon Y} \sum_{x\epsilon X} \frac{1}{2} W(y|x) \log \frac{W(y|x)}{\frac{1}{2}W(y|0) + \frac{1}{2}W(y|1)} \tag{3.3}$$

Moreover, the Bhattacharyya parameter, i.e., channel reliability parameter, is defined as

$$Z(W) \triangleq \sum_{y \epsilon Y} \sqrt{W(y|0) + W(y|1)} \tag{3.4}$$

These two parameters represent the capacity and reliability of the channel and they are the basic expressions of the polar coding. The capacity $I(W)$ is the highest bit rate that can be transmitted reliably through channel W. The $Z(W)$ parameter is an upper bound for the maximum transmission error.

Two common types of symmetrical channels are binary erasure channel, BEC, and binary symmetric channel, BSC. For the output alphabet $Y = \{0,1\}$, for a BSC we have $W(0|0) = W(1|1)$ and $W(1|0) = W(0|1)$. A BEC is a B-DMC such as $W(y|0)W(y|1) = 0$ and $W(y|0) = W(y|1)$. We write W^N to denote the channel corresponding to N uses of W, and $W^N: x^N \rightarrow y^N$ is defined by

$$W^N(y_1^N|x_1^N) = \prod_{i=1}^{N} W(y_i|x_i) \tag{3.5}$$

Properties:

For all B-DMC W, we have [1]:

$$\log_2 \frac{2}{1 + Z(W)} \leq I(W) \leq \sqrt{1 - Z(W)^2} \tag{3.6}$$

$$I(W)^2 + Z(W)^2 \leq 1 \leq I(W) + Z(W) \tag{3.7}$$

The parameters $I(W)$ and $Z(W)$ take their values in the interval [0,1] and further we have

$$I(W) \approx 1 \rightarrow Z(W) \approx 0, \text{perfect channel}$$

$$I(W) \approx 0 \rightarrow Z(W) \approx 1, \text{completely noisy channel}$$

which can be verified using 3.6 and 3.7. The larger $I(W)$ means the better channel and vice versa. That is, the channel with the smallest value of $Z(W)$ is the most reliable one. Starting from the two equivalences, we can say that to know the properties of a B-DMC

channel, it is sufficient to study one of the two parameters. We will use $Z(W)$, which is simpler and easier to handle.

The main idea of polar coding is to build virtual channels from the physical channel W such that a fraction of them tends to be perfect channels and the rest tend to be completely noisy channels. This is called channel polarization.

3.2 Channel Polarization

Polarization forms the basis of the construction of polar codes. It consists of synthesizing N independent copies of a given B-DMC W to construct N other channels $\{W_i^N : 1 \leq i \leq N\}$. The polarization appears in the sense that $I(W_N^{(i)})$ tends towards 0 or 1 [1]. The channel polarization operation is done in two stages: channel combination and channel splitting.

3.2.1 Channel combining

It consists of grouping N copies of a given B-DMC W channel into $W_N$channel. For $n = 1$, $N = 2^n = 2^1$ independent copies of $W_1 = W$ are combined to form the channel $W_2 : X^2 \rightarrow Y^2$. For W_2, we have

$$W_2(y_1, y_2|u_1, u_2) = W(y_1|u_1 \oplus u_2)W(y_2|u_2) \tag{3.8}$$

W_2 channel is depicted in Fig. 3.1.

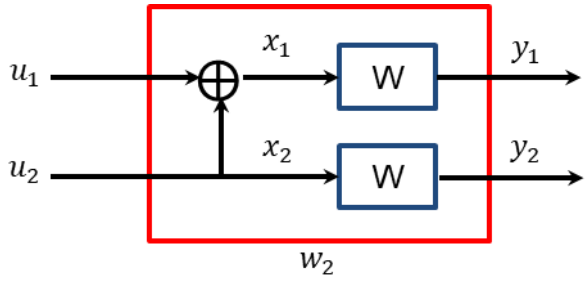

Figure 3.1 The W_2 channel.

From Fig. 3.1, we obtain $x_1 = u_1 \oplus u_2$ and $x_2 = u_2$. The relationship between x_1^2 and u_1^2 can be expressed using

$$x_1^2 = u_1^2 G_2$$

where $G_2 = \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}$.

For $n = 2$, $N = 2^2 = 4$, the channel combining is shown in Fig. 3.1. Two independent copies of the channel W_2 are combined in the same way to construct $W_4: X^4 \to Y^4$ whose conditional probability satisfies

$$W_4(y_1^4|u_1^4) = W_2(y_1^2|u_1 \oplus u_2, u_3 \oplus u_4) W_2(y_3^2|u_2, u_4) \tag{3.9}$$

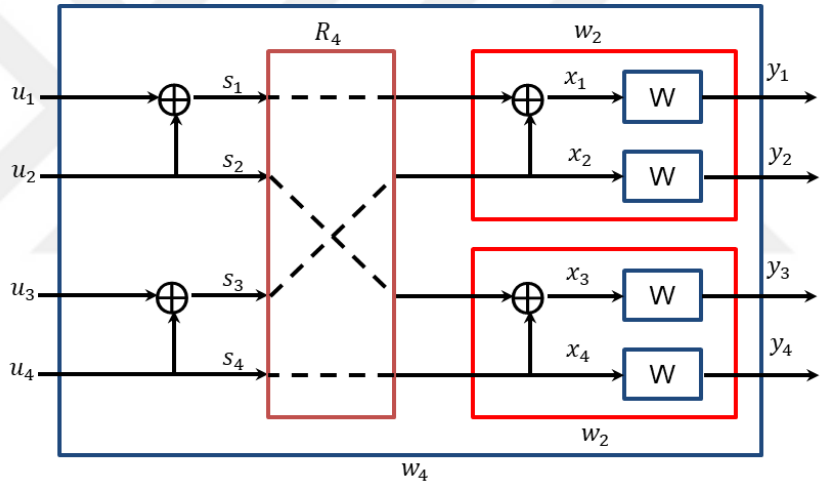

Figure 3.2 Construction of W_4 from two copies of W_2

The permutation operation R_4 transforms the vector $s_1^4 = (s_1, s_2, s_3, s_4)$ to $v_1^4 = (s_1, s_3, s_2, s_4) = (s_{1,o}^4, s_{1,e}^4)$. From Fig. 3.2, we obtain

$$x_1^4 = (x_1, x_2, x_3, x_4) = (u_1 \oplus u_2 \oplus u_3 \oplus u_4, u_3 \oplus u_4, u_2 \oplus u_4, u_4) \tag{3.10}$$

The relationship between u_1^4 and x_1^4 can be written as

$$x_1^4 = u_1^4 G_4$$

where $G_4 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 1 \end{bmatrix}$.

The relationship between the probabilities of transitions W_4 and W^4 is $W_4(y_1^4|u_1^4) = W^4(y_1^4|u_1^4 G_4)$. We have $G_4 = B_4 G_2^{\otimes 2}$, where B_4 is a permutation matrix.

The combined channel W_N for any $n \geq 1, N = 2^n$ is formed using two independent copies of the channel $W_{N/2}$ as indicated in Figure 4 where it is seen that $s_{2i-1} = u_{2i-1} \oplus u_{2i}$ and $s_{2i} = u_{2i}$ for $1 \leq i \leq N/2$. R_N is the as the reverse shuffling operation, and it takes s_1^N and produces $v_1^N = (s_1, s_3, \ldots, s_{n-1}) = \left(s_{1,o}^N, s_{1,e}^N\right)$, which is the input for $W_{N/2}$ as shown in Fig 3.3.

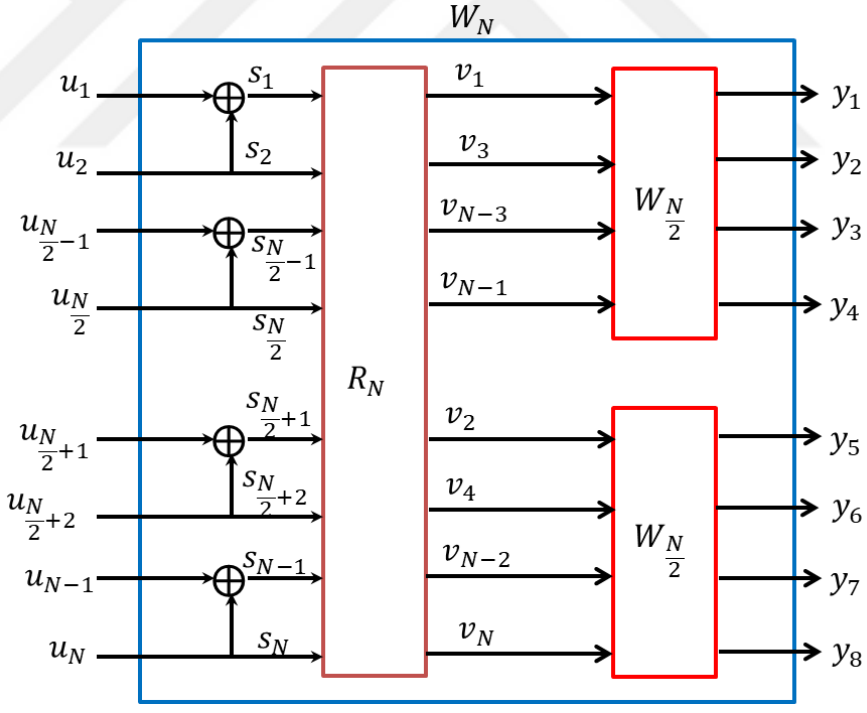

Figure 3.3 Recursive construction of the W_N channel from two $W_{N/2}$.

The code bits are obtained as

$$x_1^N = u_1^N G_N. \tag{3.11}$$

The relationship between the conditional transition probabilities of W_N and W^N can be written as

$$W_N(y_1^N|u_1^N) = W^N(y_1^N|u_1^N G_N). \tag{3.12}$$

The G_N matrix of size N is called the generator matrix. Arikan showed that $G_N = B_N G_2^{\otimes n}$ where B_N is a row permutation matrix called bit-reversal.

3.2.2 Channel splitting

After the channel combination, the next step of channel polarization is to divide the combined channel W_N into $W_N^i: X \to Y^N \times X^{i-1}$ split channels defined by the transition probabilities:

$$W_N^{(i)}\left(y_1^N, u_1^{i-1}\middle|u_i\right) \triangleq \sum_{u_{i+1}^N \in X^{N-i}} \frac{1}{2^{N-i}} W(y_1^N|u_1^N) \tag{3.13}$$

$W_N^{(i)}$ is the channel seen by u_i. For the channel input u_i, channel outputs are

$$\left(y_1^N, u_1^{i-1}\right), 1 \le i \le N.$$

Let's now consider the decoding operation considering the split channels.

First, we decode the bit u_1 using the received symbols $y_1^N = [y_1\ y_2\ \dots\ y_{N-1}\ y_N]$. The channel W_N^1 is defined as [45]:

$$W_N^1: u_1 \to y_1^N.$$

In the second step, the bit u_2 can be decoded using the previously decoded bit u_1 and the received symbols y_1^N. The channel W_N^2 is defined as

$$W_N^2: u_2 \to y_1^N, \hat{u}_1.$$

In general for the decoding of the bit u_i, the channel W_N^i is defined as

$$W_N^i: u_i \to y_1^N, \hat{u}_1, \hat{u}_2, \dots, \hat{u}_{i-1}$$

The channel splitting operation is illustrated in Fig 3.4.

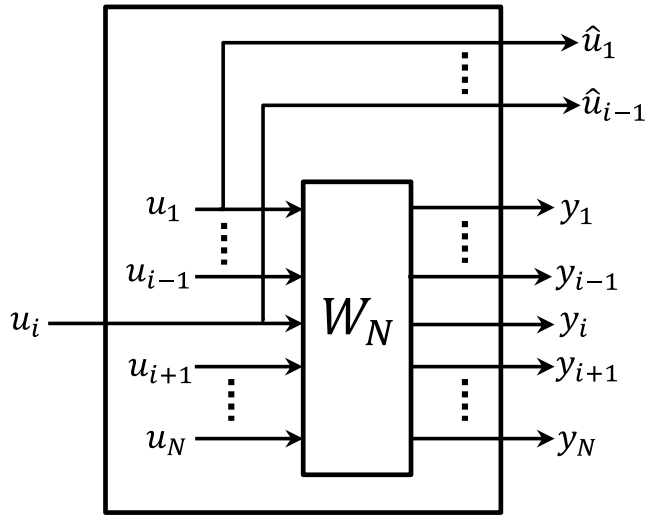

Figure 3.4 Illustration of channel splitting.

3.2.3 The phenomenon of the channel polarization

This polarization phenomenon can be explained with the following theorem [1]:

Channel polarization refers to the fact that it is possible to synthesize N independent copies of a given B-DMC channel and $\delta \in [0,1]$ then the set of the values of N binary-input channels $I\left(W_N^{(i)}\right) \in [1-\delta, 1]$. such that as N becomes large the fraction of indices i for which $I\left(W_N^{(i)}\right)$ is near 1 approaches $I(W)$ and the fraction for which $I\left(W_N^{(i)}\right)$ is near 0 approaches $1 - I(W)$.

In other words the two poles of $I\left(W_N^{(i)}\right)$ are $I(W)$ and $1 - I(W)$. Here is a diagram showing the channel bias effect.

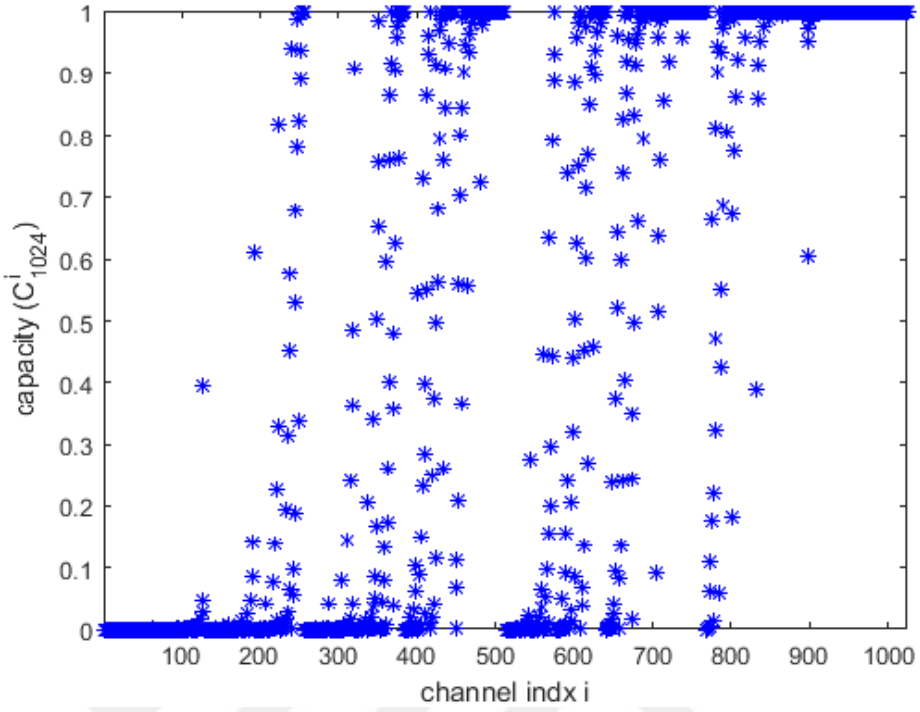

Figure 3.5 Channel polarization for a BEC with erasure probability $\varepsilon = 0.5$.

In Fig. 3.5, it is seen that a set of $I\left(W_N^{(i)}\right)$ $i = 2^0, \ldots, N = 2^{10}$, gets value of 1, and some others get the value of 0, and the rest get value in between 0 and 1.

3.3 The recursive channel transformation

The channel polarization is nothing but a channel transformation from N independent copies of a given B-DMC W. This transformation begins by grouping of two copies of W by a technique based on the mutual information chain rule. For independent channels we have the property

$$I(u_1, u_2; y_1, y_2) = I(u_1; y_1) + I(u_2; y_2) = 2I(W) \tag{3.14}$$

For combined channel, we can write

$$I(u_1, u_2; y_1, y_2) = I(u_1; y_1, y_2) + I(u_2; y_1, y_2, u_1) \tag{3.15}$$

$W^-$and W^+ are the split channels. The expressions on the right side of 3.15, are the mutual information for split channels, i.e., $I(W^-) = I(u_1; y_1, y_2)$ and $I(W^+) = I(u_2; y_1, y_2, u_1)$, and we have

$$I(W^-) + I(W^+) = 2I(W) \tag{3.16}$$

The conditional channel probabilities for split channels can be written as

$$W^-(y_1; y_2|u_1) = \sum_{u_2 \in X} W(y_1|u_1 \oplus u_2) W(y_2|u_2) \tag{3.17}$$

$$W^+(y_1, y_2, u_1|u_2) = \frac{1}{2} W(y_1|u_1 \oplus u_2) W(y_2|u_2) \tag{3.18}$$

Let $W^- = W_2^{(1)}$ and $W^+ = W_2^{(2)}$, the channel splitting operation can be illustrated as $(W, W) \to \left(W_2^{(1)}, W_2^{(2)}\right)$.

The relationship between split channels can be illustrated using [1]:

$$\left(W_N^{(i)}, W_N^{(i)}\right) \to \left(W_{2N}^{(2i-1)}, W_{2N}^{(2i)}\right) \tag{3.19}$$

Conditional split channel probabilities can be calculated using

$$W_{2N}^{(2i-1)}\left(y_1^{2N}, u_1^{2i-2} \middle| u_{2i-1}\right) =$$
$$= \sum_{u_{2i} \in X} \frac{1}{2} W_N^{(i)}\left(y_1^N, u_{1,o}^{2i-2} \oplus u_{1,e}^{2i-2} \middle| u_{2i-1} \oplus u_{2i}\right) W_N^{(i)}\left(y_{N+1}^{2N}, u_{1,e}^{2i-2} \middle| u_{2i}\right) \tag{3.20}$$

and

$$W_{2N}^{(2i)}\left(y_1^{2N}, u_1^{2i-1} \middle| u_{2i}\right) =$$
$$= \frac{1}{2} W_N^{(i)}\left(y_1^N, u_{1,o}^{2i-2} \oplus u_{1,e}^{2i-2} \middle| u_{2i-1} \oplus u_{2i}\right) W_N^{(i)}\left(y_{N+1}^{2N}, u_{1,e}^{2i-2} \middle| u_{2i}\right) \tag{3.21}$$

For the split channel capacities we have the expressions [1]

$$I\left(W_{2N}^{(2i-1)}\right) + I\left(W_{2N}^{(2i)}\right) = 2I\left(W_N^{(i)}\right) \tag{3.22}$$

$$Z\left(W_{2N}^{(2i-1)}\right) + Z\left(W_{2N}^{(2i)}\right) \leq 2Z\left(W_N^{(i)}\right) \tag{3.23}$$

We have the inequalities

$$I\left(W_{2N}^{(2i-1)}\right) \leq I\left(W_N^{(i)}\right) \leq I\left(W_{2N}^{(2i)}\right) \tag{3.24}$$

$$Z\left(W_{2N}^{(2i-1)}\right) \geq Z\left(W_N^{(i)}\right) \geq Z\left(W_{2N}^{(2i)}\right) \tag{3.25}$$

for the split channel capacities and for the Bhattacharyya parameters of the split channels. If BEC is used for the transmission of the code bits, then we have the capacity equalities

$$I\left(W_N^{(2i-1)}\right) = I\left(W_{\frac{N}{2}}^{(i)}\right)^2 \tag{3.26}$$

$$I\left(W_N^{(2i)}\right) = 2I\left(W_{\frac{N}{2}}^{(i)}\right) - I\left(W_{\frac{N}{2}}^{(i)}\right)^2 \tag{3.27}$$

where $I\left(W_1^{(1)}\right) = 1 - \epsilon$ and for the Bhattacharyya parameters of the split channels we have the equalities

$$Z\left(W_N^{(2i-1)}\right) = 2Z\left(W_{\frac{N}{2}}^{(i)}\right) - Z\left(W_{\frac{N}{2}}^{(i)}\right)^2 \tag{3.28}$$

$$Z\left(W_N^{(2i)}\right) = Z(W_{\frac{N}{2}}^{(i)})^2 \tag{3.29}$$

where $\left(W_1^{(1)}\right) = \epsilon$.

In Fig. 3.6, recursive channel transformation using Butterfly structure is depicted. It is seen from Fig. 3.6 that the split channels $W_8^{(i)}: 1 \leq i \leq 8$, are obtained from 8 copies of the B-DMC W channel. The split channel located at a node is obtained from the two channels to the right of the node to which it is connected, for example using $W_2^{(1)}$and $W_2^{(2)}$, we get $W_4^{(1)}$.

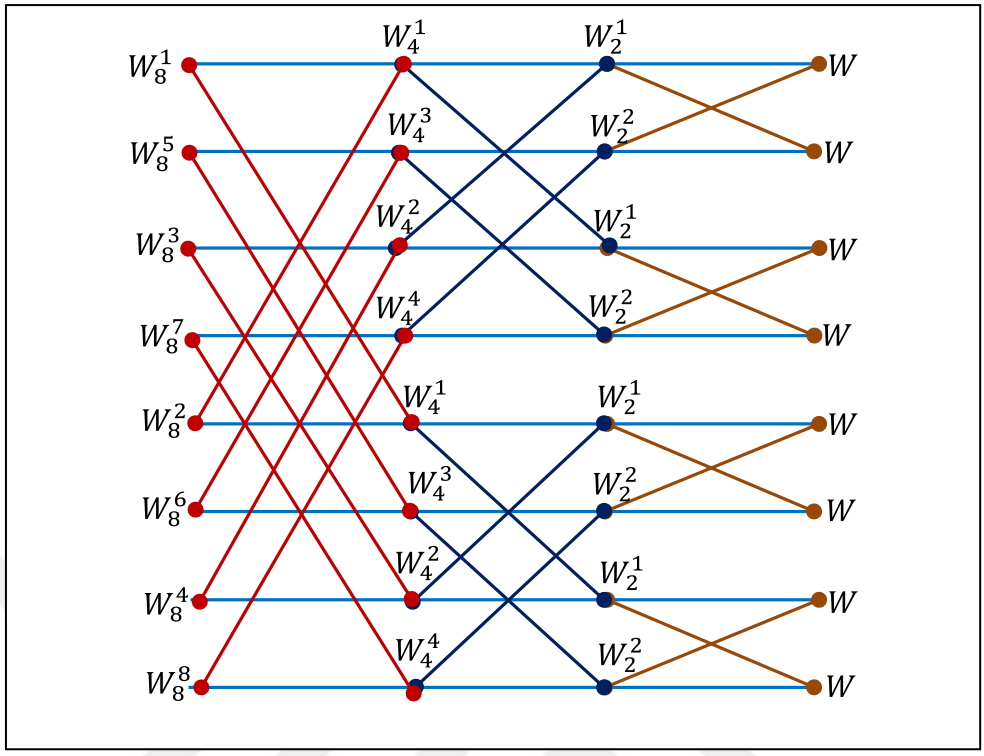

Figure 3.6 Butterfly structure for recursive channel transformation for N = 8.

3.4 Polar coding

Polar coding utilize the channel polarization effect to construct codes that attain the capacity of the channel. The principle of polar coding lies on sending the data through those the most reliable channels, i.e., sending data those channels whose $Z\left(W_8^{(i)}\right)$ are closer to 0.

Polar codes belong to a special class of codes called G_N-coset codes.

Polar coding is achieved using

$$x_1^N = u_1^N G_N \tag{3.30}$$

where $N = 2^n, n \geq 1$, represents the block length and G_N is the generator matrix of the code G_N −coset. Given a subset A such that $A \subset \{1, \ldots, N\}$, 3.30 can be written in the form

$$x_1^N = u_A G_N(A) \oplus u_{A^c} G_N(A^c). \tag{3.31}$$

Where u_A is a subvector, whose dimension is k, of u_1^N, and it is composed of the elements of u_1^N whose indices are in A. $G_N(A)$ is the submatrix of G_N constructed using the rows whose indices belong to A. The vector u_A is called the information vector and u_{A^c} is called frozen bit vector, i.e., it is a type of party bit vector. In general, we choose u_{A^c} as 0_1^{n-k}. The choice of u_{A^c} does not affect the performance of the system [1]. Hence, 3.31 can be simplified as $x_1^N = u_A G_N(A)$. Polar code can be considered as a linear block code with input vector u_A and the generator matrix G_N. The parity check matrix can be formed by the columns of G_N of the polar code whose indices are in A^c [44].
For example, for $(4,2)$ polar code, encoding operation can be detailed as

$$x_1^4 = u_1^4 G_4 \rightarrow x_1^4 = u_A G_4(A) \otimes u_{A^c} G_4(A^c) \rightarrow$$
$$x_1^4 = (u_2, u_4)\begin{bmatrix} 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 1 \end{bmatrix} + (0,0)\begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \end{bmatrix}.$$

For data word $(u_2, u_4) = (1,1)$, the codeword is obtained as $x_1^4 = (0,1,0,1)$.

The polar code is a G_N −coset code with parameter (n, k, A, u_{A^c}) where the elements of A, i.e., the indices for information bits, correspond to the most reliable channels. The indices in A are chosen such that $Z\left(W_N^{(i)}\right) \leq Z\left(W_N^{(j)}\right)$ for all $i \in A$ and $j \in A^c$. Note, that from this selection rule it is clear that polar code construction is specific to the channel for which it is designed.

3.4.1. Bhattacharyya parameter based channel construction

The channel polarization process begins with the calculations of the Bhattacharyya parameter values, $Z(W)$, for the type of channel under concern. The values of the Bhattacharyya parameters are used to estimate the noiseless and the noisy bit channels. After Bhattacharyya parameter calculation, before the polar encoding information, data vector is formed. In the data vector, information bits are placed to those locations with good Bhattacharyya parameter values converging to 0, and frozen bits are placed to locations with bad Bhattacharyya parameter values converging to 1.

The Bhattacharyya parameters can be calculated in a recursive manner as depicted in Fig. 3.7 where it is seen that for the upper braches, we use $2Z - Z^2$ and for the lower branches we employ Z^2 during the recursive calculations. The initial values of Bhattacharyya parameters differ considering the type of the channel and they are determined as shown in Table 1.

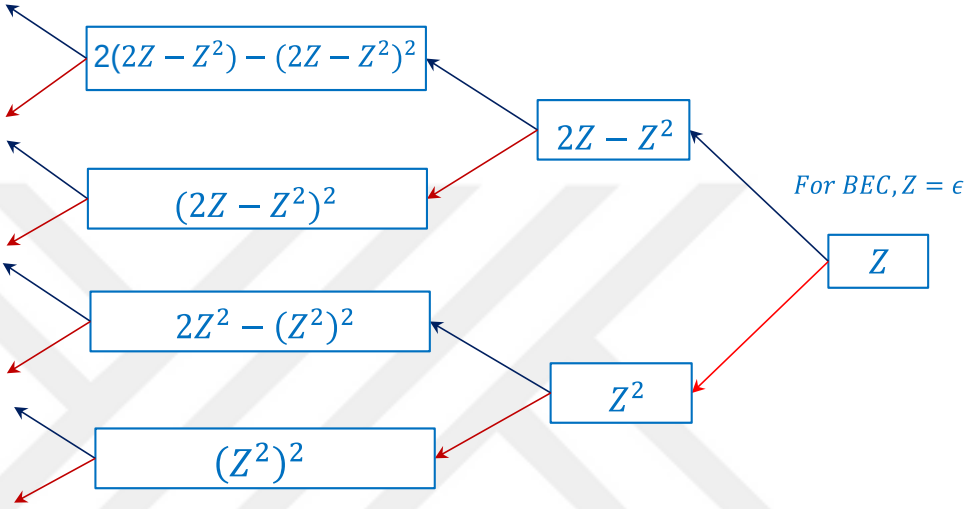

Figure 3.7 Polarized transformation Bhattacharyya parameter Z (W)

Table 1 Initial values of the Bhattacharyya parameters for different channel types.

Channel type	Affected parameter	Initial $Z(W)$	Good channel	Bad channel
BEC	Erasure probability ϵ	ϵ	$2Z(W) - Z^2(W)$	$Z^2(W)$
BSC	Symmetric probability P	$2 \times \sqrt{P(1-P)}$	$2Z(W) - Z^2(W)$	$Z^2(W)$
$AWGN$	SNR	$e^{-R\frac{E_b}{N_o}}$	$2Z(W) - Z^2(W)$	$Z^2(W)$

3.4.1.1 Binary erasure channel (BEC)

As it is illustrated in Fig. 3.8, the initial value of the Bhattacharyya parameter is set to be $Z = \epsilon$. After that, the channel is split into two sub-channels $\left(W_2^{(1)}, W_2^{(2)}\right)$, and for the

split channels the reliability measures are calculated using $Z\left(W_2^{(1)}\right) = 2Z(W) - Z(W)^2$ and $\left(W_2^{(1)}\right) = Z(W)^2$.

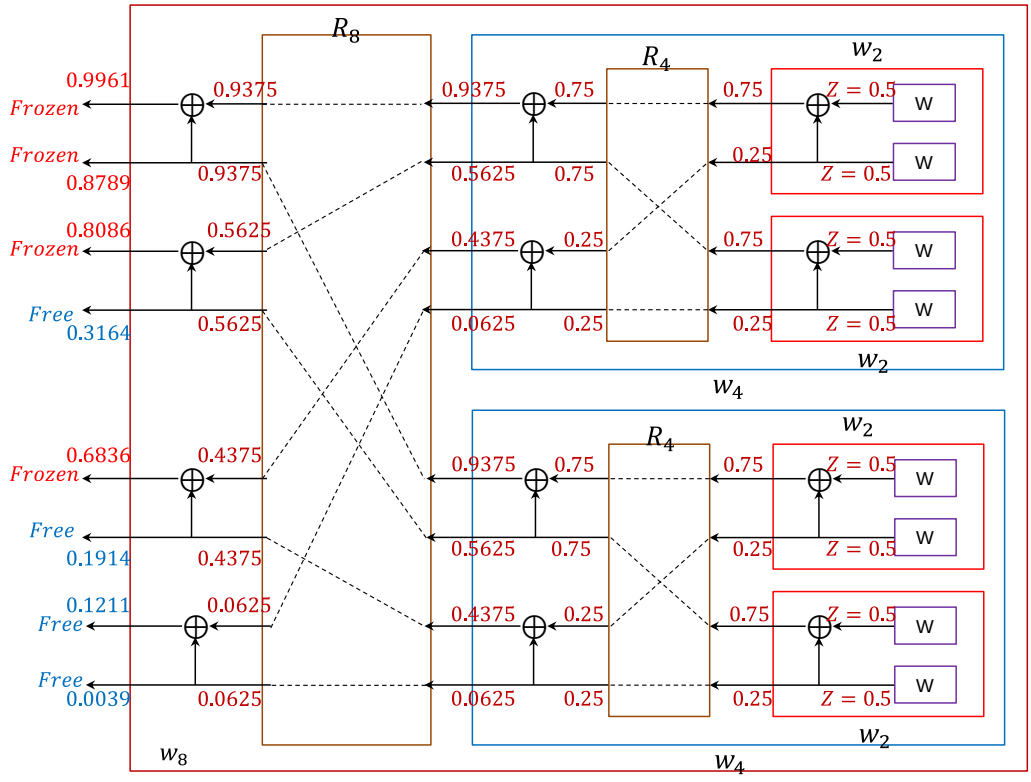

Figure 3.8 Code construction of BEC for $N = 8, K = 4$, and $\epsilon = 0.5$

The calculation of the Bhattacharyya values for BEC for $N = 8, K = 4$, and $\epsilon = 0.5$ is explained in Fig. 3.8 where it is seen that a backward recursive calculation is performed. It is also seen in Fig. 3.8 that there are $3 = \log(8)$ stages where Bhattacharyya values are calculated, and frozen bit indices are chosen for large Z values.

3.4.1.2. Binary symmetric channel (BSC)

The input alphabet of the BSC channel is $x = \{0, 1\}$ and the channel output alphabet is $y = \{0, 1\}$. The channel transition probabilities are $p(y = 0|x = 1) = P$, $p(y = 1|x = 0) = P$ and we have $p(y = 0|x = 0) = 1 - P$, $p(y = 1|x = 1) = 1 - P$. The Bhattacharya parameter for BSC can be calculated as $Z = 2 \times \sqrt{P(1 - P)}$.

3.4.1.3. Additive white Gaussian noise (AWGN) channel

The output of the $AWGN$ is represented by $y = x + n$ where $x \in \{+1, -1\}$ and n is the additive white Gaussian noise with zero mean and variance σ^2. The channel transition probability can be computed using,

$$p(y|x) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left[-\frac{(y-1)^2}{2\sigma^2}\right] \tag{3.32}$$

The Bhattacharya parameter of AWGN can be calculated as $Z = e^{-R\frac{E_b}{N_0}}$ where E_b is the transmitted bit energy and $N_0/2$ is the double sided noise power spectral density.

3.5 Polar Encoding

For polar encoding operation, we need generator matrix of the polar code under concern. The generator matrix of the polar code for frame length N is calculated as [1]

$$G_N = R_N\left(G_2 \otimes G_{\frac{N}{2}}\right) = B_N G_2^{\otimes n} = G_2^{\otimes n} B_N. \tag{3.33}$$

Where B_N is a permutation matrix, and it is recursively calculated as in

$$B_N = R_N\left(I_2 \otimes R_{\frac{N}{2}}\right)\left(I_4 \otimes R_{\frac{N}{4}}\right) \ldots \ldots \left(I_{\frac{N}{2}} \otimes R_2\right) = R_N\left(I_2 \otimes B_{\frac{N}{2}}\right). \tag{3.34}$$

Let $N = 2^n, n \geq 0$, and I_k be a k-dimensional identity matrix for $k \geq 1$. Substituting 3.34 into 3.33, we get

$$G_N = \left(I_{\frac{N}{2}} \otimes F\right) R_N\left(I_2 \otimes G_{\frac{N}{2}}\right), \text{for } N \geq 2 \tag{3.35}$$

where $G_1 = I_1$.

3.35 can also be written as

$$G_N = R_N\left(F \otimes I_{\frac{N}{2}}\right)\left(I_2 \otimes G_{\frac{N}{2}}\right). \tag{3.36}$$

3.35 and 3.36 are equivalent to each other. From formulas 3.35 and 3.36, we notice that $\left(I_{N/2} \otimes F\right)R_N = R_N(F \otimes I_{N/2})$. Hence, we can write formula 3.36 as

$$G_N = R_N\left(F \otimes G_{\frac{N}{2}}\right). \tag{3.37}$$

For $N/2$, we get

$$G_{\frac{N}{2}} = R_{\frac{N}{2}}\left(F \otimes G_{\frac{N}{4}}\right) \tag{3.38}$$

leading to

$$G_N = R_N\left(F \otimes R_{\frac{N}{2}}\left(F \otimes G_{\frac{N}{4}}\right)\right). \tag{3.39}$$

Using the identity $(AC) \otimes (BD) = (A \otimes B)(C \otimes D)$ with $A = I_2, B = R_{N/2}, C = F, D = F \otimes G_{N/4}$ for 3.39 we get

$$G_N = R_N\left(I_2 \otimes R_{\frac{N}{2}}\right)\left(F^{\otimes 2} \otimes G_{\frac{N}{4}}\right). \tag{3.40}$$

Where we have

$$G_{\frac{N}{2}} = B_N F^{\otimes n} \tag{3.41}$$

$$B_N = R_N\left(I_2 \otimes B_{\frac{N}{2}}\right) \tag{3.42}$$

where B_N represents permutation matrix also known as bit-reversal and I_2 is the identity matrix, B_2 is initialized as $B_2 = I_2$. $\otimes$ is the Kronecker product, R_N is the permutation operation which maps the input sequence $\{1, 2, 3, 4, \dots, N\}$ to $\{1,3, \dots, N-1, 2, 4, \dots, N\}$ and $n = \log_2 N$.

Numerical Example:

For $N = 8$, we can calculate the generator matrix as

$$n = \log_2 8 = 3,\ G_8 = B_8 F^{\otimes 3} = B_8 \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}^{\otimes 3},\ B_2 = I_2 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

$$B_4 = R_4(I_2 \otimes B_2) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}\left(\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \otimes \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}\right) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$B_8 = R_8(I_2 \otimes B_4) =$

$$B_8 = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \left(\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \otimes \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \right)$$

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

$$G_8 = B_8 F^{\otimes 3} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}^{\otimes 3} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix}$$

and encoding operation for $N = 8$ can be performed as

$$x_1^N = u_1^N G_N \rightarrow x_1^N = u_1^N \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix}$$

leading to the equation set

$$x_1 = u_1 \oplus u_2 \oplus u_3 \oplus u_4 \oplus u_5 \oplus u_6 \oplus u_7 \oplus u_8$$

$$x_2 = u_5 \oplus u_6 \oplus u_7 \oplus u_8$$

$$x_3 = u_3 \oplus u_4 \oplus u_7 \oplus u_8$$

$$x_4 = u_7 \oplus u_8$$

$$x_5 = u_2 \oplus u_4 \oplus u_6 \oplus u_8$$

$$x_6 = u_6 \oplus u_8$$

$$x_7 = u_4 \oplus u_8$$

$$x_8 = u_8.$$

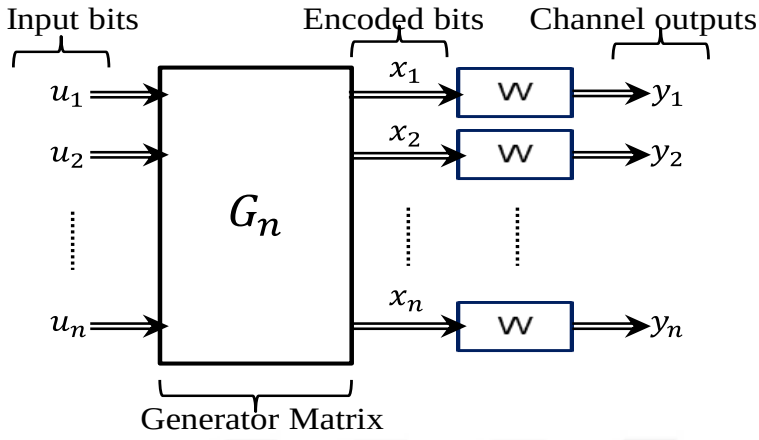

Figure 3.9 Block diagram illustration of polar encoding.

The encoding process can be using factor graphs as in Fig. 3.10 where information bits are used for the indices in $A = \{4, 6, 7, 8\}$ and the frozen bits are used for the indices in $A^c = \{1, 2, 3, 5\}$. Frozen bits are all equal to 0. The graph explains the mapping operation $x_1^8 = u_1^8 G_8 = u_1^8 G_2^{\otimes 3}$.

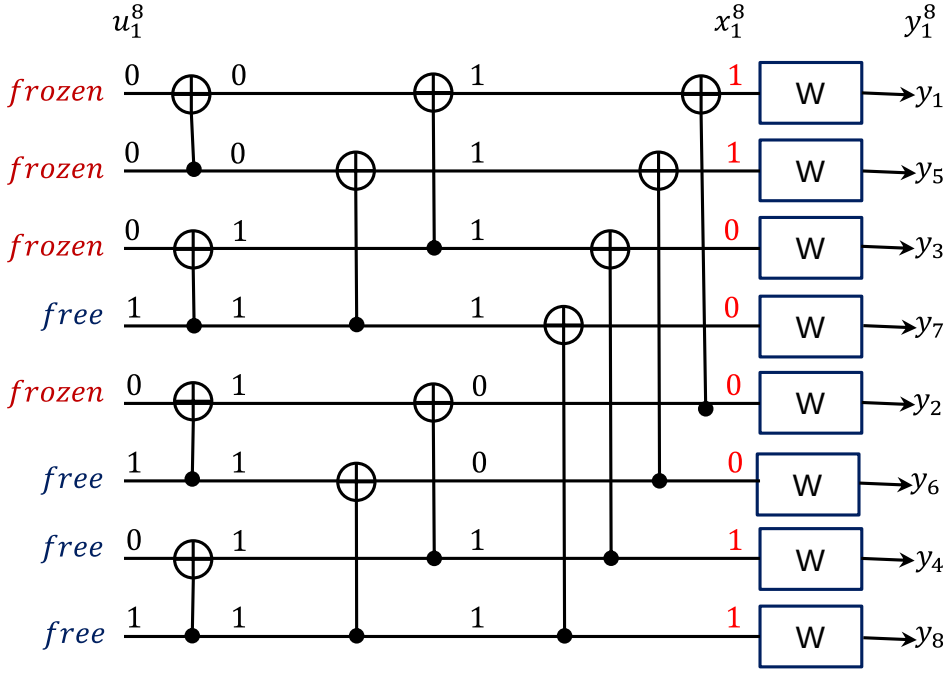

Figure 3.10 Polar encoding illustration for N=8 and K=4 using factor graph.

CHAPTER 4

POLAR DECODING

For the decoding of polar codes, several decoding algorithms are used in the literature. Some of these algorithms as: successive cancellation (SC) [1], linear programming decoding (LP) [45], and SC with list (SCL), i.e., SC List [4]. In this chapter, we explain SC decoding of polar codes in details. Besides, we provide brief information for SCL decoding algorithm.

4.1 SC Decoding of Polar Codes

4.1.1 Fundamental concepts for polar decoding

The kernel units that are repeatedly utilized in encoder and decoder structures of polar codes are depicted in Fig.4.1. The kernel unit can be considered for the smallest length codeword of polar encoder and decoder units.

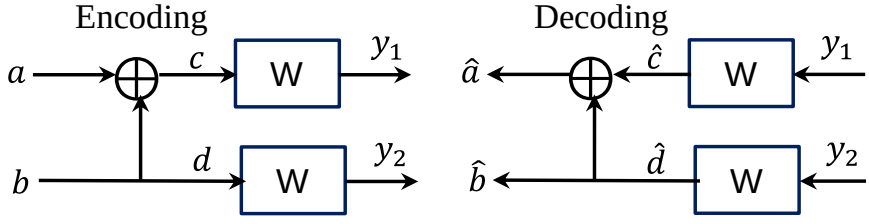

Figure 4.1 Kernel encoding and decoding units of polar codes.

In Figure 12 a, b, c, d and $\hat{a}, \hat{b}, \hat{c}, \hat{d}$ are binary variables. It is clear from Fig.4.1 that

$$\hat{a} = \hat{c} \oplus \hat{d} \qquad \hat{b} = \hat{d} \tag{4.1}$$

From $\hat{a} = \hat{c} \oplus \hat{d}$, it is clear that $\hat{a} = 0$ if $\hat{c} = \hat{d} = 0$ or $\hat{c} = \hat{d} = 1$, which implies that

$$P(\hat{a} = 0) = P(\hat{c} = 0)P(\hat{d} = 0) + P(\hat{c} = 1)P(\hat{d} = 1) \tag{4.2}$$

and in the same manner we have $\hat{a} = 1$ if $\hat{c} = 0$, $\hat{d} = 1$ or $\hat{c} = 1, \hat{d} = 0$ from which we can write that

$$P(\hat{a}=1) = P(\hat{c}=0)P(\hat{d}=1) + P(\hat{c}=1)P(\hat{d}=0) \tag{4.3}$$

Likelihood ratio for estimated the bit $\hat{a}$ can be calculated as:

$$LR(\hat{a}) = \frac{P(\hat{a}=0)}{P(\hat{a}=1)}$$

in which substituting 4.2 and 4.3, we obtain

$$LR(\hat{a}) = \frac{P(\hat{c}=0)P(\hat{d}=0) + P(\hat{c}=1)P(\hat{d}=1)}{P(\hat{c}=0)P(\hat{d}=1) + P(\hat{c}=1)P(\hat{d}=0)}$$

leading to

$$LR(\hat{a}) = \frac{1 + LR(\hat{c})LR(\hat{d})}{LR(\hat{c}) + LR(\hat{d})} \tag{4.4}$$

which is used to estimate the value of $\hat{a}$. After the estimation of $\hat{a}$, the bit $\hat{b}$ can be determined. Assuming that the bit $\hat{a}$ is determined as 0, i.e., $\hat{a} = 0$ then we have $\hat{b} = 0$ if $\hat{c} = 0$, and $\hat{d} = 0$, and $\hat{b} = 1$ if $\hat{c} = 1$ and $\hat{d} = 1$. Then we can write

$$P(\hat{b}=0)_{\hat{a}=0} = P(\hat{c}=0)P(\hat{d}=0) \tag{4.5}$$

and

$$P(\hat{b}=1)_{\hat{a}=0} = P(\hat{c}=1)P(\hat{d}=1) \tag{4.6}$$

Likelihood ratio for the bit $\hat{b}$ can be written as

$$LR(\hat{b})_{\hat{a}=0} = \frac{P(\hat{b}=0)_{\hat{a}=0}}{P(\hat{b}=1)_{\hat{a}=0}} = \frac{P(\hat{c}=0)P(\hat{d}=0)}{P(\hat{c}=1)P(\hat{d}=1)}$$

which can be calculated as

$$LR_{\hat{a}=0}(\hat{b}) = LR(\hat{c})LR(\hat{d}). \tag{4.7}$$

If $\hat{a}$ is decided to be 1, i.e., $\hat{a} = 1$, then $\hat{b} = 0$ if $\hat{c} = 1$ and $\hat{d} = 0$, and $\hat{b} = 1$ if $\hat{c} = 0$ and $\hat{d} = 1$. Then, we can write

$$P(\hat{b}=0)_{\hat{a}=1} = P(\hat{c}=1)P(\hat{d}=0) \tag{4.8}$$

$$P(\hat{b}=1)_{\hat{a}=1} = P(\hat{c}=0)P(\hat{d}=1) \tag{4.9}$$

Likelihood ratio for the bit $\hat{b}$ under constraint $\hat{a} = 1$ can be written as

$$LR(\hat{b})_{\hat{a}=1} = \frac{P(\hat{b}=0)_{\hat{a}=1}}{P(\hat{b}=1)_{\hat{a}=1}} = \frac{P(\hat{c}=1)P(\hat{d}=0)}{P(\hat{c}=0)P(\hat{d}=1)}$$

$$LR(\hat{b})_{\hat{a}=1} = LR(\hat{c})^{-1}LR(\hat{d}). \tag{4.10}$$

The formulas 4.7 and 4.10 can be combined under a single term as

$$LR(\hat{b}) = [LR(\hat{c})]^{1-2\hat{a}} \times LR(\hat{d}). \tag{4.11}$$

For any codeword length, the formulas 4.4 and 4.10 takes the forms

$$L_N^{(2i-1)}(y_1^N, \hat{u}_1^{2i-2}) = \frac{L_{\frac{N}{2}}^{(i)}\left(y_1^{\frac{N}{2}}, \hat{u}_{1,o}^{2i-2} \oplus \hat{u}_{1,e}^{2i-2}\right) L_{\frac{N}{2}}^{(i)}\left(y_{\frac{N}{2}+1}^{N}, \hat{u}_{1,e}^{2i-2}\right) + 1}{L_{\frac{N}{2}}^{(i)}\left(y_1^{\frac{N}{2}}, \hat{u}_{1,o}^{2i-2} \oplus \hat{u}_{1,e}^{2i-2}\right) + L_{\frac{N}{2}}^{(i)}\left(y_{\frac{N}{2}+1}^{N}, \hat{u}_{1,e}^{2i-2}\right)} \tag{4.12}$$

and

$$L_N^{(2i)}(y_1^N, \hat{u}_1^{2i-1}) = \left[L_{\frac{N}{2}}^{(i)}\left(y_1^{\frac{N}{2}}, \hat{u}_{1,o}^{2i-2} \oplus \hat{u}_{1,e}^{2i-2}\right)\right]^{1-\hat{u}_{2i-1}} \times L_{\frac{N}{2}}^{(i)}\left(y_{\frac{N}{2}+1}^{N}, \hat{u}_{1,e}^{2i-2}\right). \tag{4.13}$$

The bit decision considering its likelihood ratio is made according to

$$u_i = \begin{cases} 0 & if \;\; LR(u_i) \geq 1 \\ 1 & otherwise \end{cases} \tag{4.14}$$

4.1.2 Determination of the Level indices and the node bit distribution

The decoding operation using the tree structure can be divided into two parts, the distribution of the previously decoded bits to the nodes, and the node LRs calculation for the current decoding bit [46]. The tree is divided into $\log_2(N) + 1$ levels in the bit distribution stage, N is frame length such that $N = 2^n$. For instance, for $N = 16$, the total number of levels is $\log_2(16) + 1 = 5$ and the indices of the levels are 0,1,2,3,4. The number of nodes at the topmost level is $2^0 = 1$, and the number of nodes in the bottommost level is $2^4 = 16$.

The distribution of the previously decoded bits to the nodes can be defined using formula 4.15, where the active nodes appear at certain levels which are called g-nodes and the

passive nodes appear at certain levels which are called f-nodes. The active levels can be determined using

$$L = \sum_{i} 2^{i} \tag{4.15}$$

where i denotes the indices of the active levels. The labels of g-nodes on the active levels can be 0 or 1. Let us assume as an example that $N = 16$ and the first 11 bits are decoded. For decoding of 12th bit, the previously decoded bits are distributed among the nodes and the number 11 can be written as

$$11 = 2^0 + 2^1 + 2^3.$$

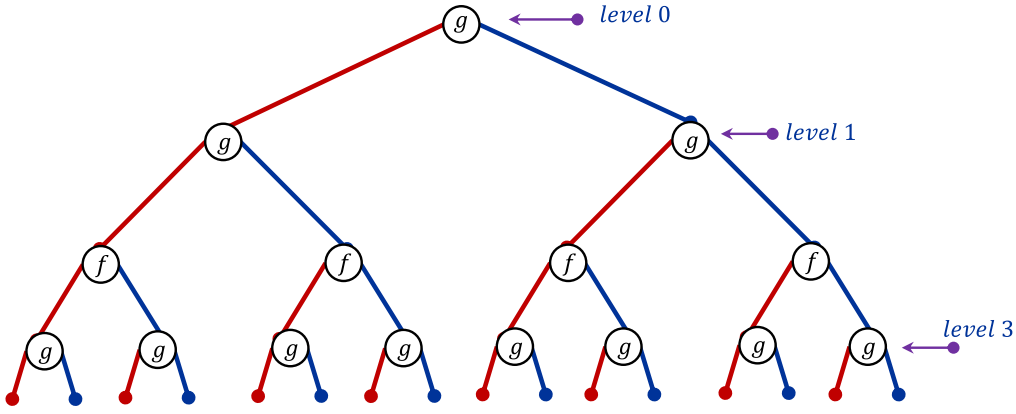

Figure 4.2 The node bits location for the decoding of 12th data bit.

4.1.3 Distribution of the decoded bits to the nodes using generator matrix

The distribution of previously decoded bits to the nodes can be achieved using the generator matrix. Assume that we have the code word length N and the first M bits are decoded $u_1^M = [u_1\ u_2\ u_3\ \dots u_M]$. We consider the decoding the bits $(M+1)^{th}$ bit. M can be written as

$$M = \sum_{i} 2^{i} = M_1 + M_2 + \cdots .. + M_k.$$

Where i denotes the indices of the active levels. The labels of g-nodes on the active levels can be obtained by dividing the decoded bit vector u_1^M into sub-vectors $v_{1,}v_2, v_3 \dots v_k$ containing $M_1, M_2, \dots, M_k$ bits as in

$$\left[\underbrace{u_1\, u_2 \dots}_{v_k} \dots \underbrace{\dots \dots}_{v_2}\ \underbrace{\dots u_M}_{v_1}\right].$$

Since the sub-vectors $v_{1,}v_2, v_3 \dots v_k$ are defined, the node bits at certain levels can be calculated as

$$b_1 = v_1 \times G_{M_1}\ , \qquad b_2 = v_2 \times G_{M_2} \dots\ b_k = v_k \times G_{M_k}.$$

For instance, assume that $N = 16$ and the first 7 bits are decoded. For the decoding of the 8th bit, the previously decoded bits are distributed among the nodes, for this purpose first we write the number 7 as in

$$7 = 2^0 + 2^1 + 2^2 = 1 + 2 + 4.$$

Next, the previously decoded bits vector $u_1^M = [u_1\, u_2\, u_3 \dots u_M]$ are divided into sub-vectors as

$$\left[\underbrace{u_1\, u_2\, u_3\, u_4}_{v_4}\ \underbrace{u_5\, u_6}_{v_2}\ \underbrace{u_7}_{v_1}\right].$$

Multiplying sub-vectors by the corresponding generator matrices, we get the node-bit vectors as

$$b_1 = v_1 \times G_1 = u_7 \times 1 \to u_7$$

$$b_2 = v_2 \times G_2 = [u_5\, u_6] \times \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix} \to b_2 = [u_5 \oplus u_6\ \ u_6]$$

$$b_4 = v_4 \times G_4 = [u_1\, u_2\, u_3\, u_4] \times \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 1 \end{bmatrix} \to$$

$$b_4 = [u_1 \oplus u_2 \oplus u_3 \oplus u_4\ \ u_3 \oplus u_4\ \ u_2 \oplus u_4\ \ u_4].$$

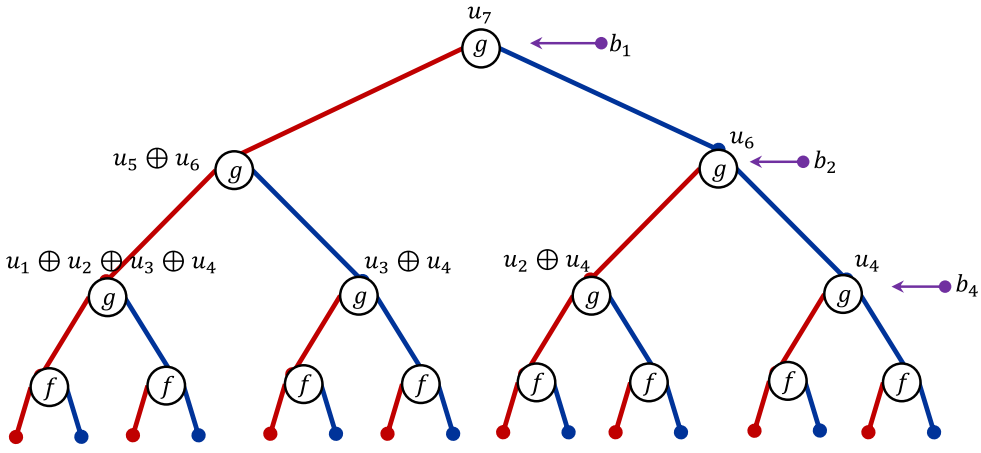

Figure 4.3 The distribution of the decoded bits to the nodes using generator matrix

4.2 Principle of SC decoding

This decoding technique, proposed by Arıkan [1], [47], is known as successive cancellation. In this algorithm, the i^{th} bit u_i is decoded using the previously decoded $i-1$ bits i.e., $\hat{u}_1^{i-1}$, and the received word y_1^N. For the frozen bit positions, no calculation is necessary and $\hat{u}_{A^c} = u_{A^c}$. First, an estimation $\hat{u}_1$ of u_1 is made using y_1^N. Then, $\hat{u}_2$ is decoded from y_1^N and $\hat{u}_1$, and this procedure is repeated till the decoding of last bit. The task of SC decoder, therefore, is to determine an estimate $\hat{u}_1^N$ of u_1^N from the knowledge of A, u_{A^c} and y_1^N. Since the decoder knows the frozen bits, i.e., we have $\hat{u}_{A^c} = u_{A^c}$, its task is to determine an estimate $\hat{u}_A$ of the information bits u_A. The decoder generates its estimate according to

$$\hat{u}_i \triangleq \begin{cases} u_i \;\; if \; i \in A^c \\ 0 \;\; if \; L_N^{(i)}\left(y_1^N, \hat{u}_1^{i-1}\right) \geq 1 \;\; , \;\; 1 \leq i \leq N \\ 1 \;\; otherwise \end{cases} \tag{4.16}$$

where $L_N^{(i)}$ is defined as

$$L_N^{(i)}\left(y_1^N, \hat{u}_1^{i-1}\right) = \frac{W_N^{(i)}\left(y_1^N, \hat{u}_1^{i-1} \middle| 0\right)}{W_N^{(i)}\left(y_1^N, \hat{u}_1^{i-1} \middle| 1\right)} \tag{4.17}$$

Using (3.20) and (3.21) it can be shown that

$$L_N^{(2i-1)}\left(y_1^N, \hat{u}_1^{2i-2}\right) = \frac{L_{\frac{N}{2}}^{(i)}\left(y_1^{\frac{N}{2}}, \hat{u}_{1,o}^{2i-2} \oplus \hat{u}_{1,e}^{2i-2}\right) L_{\frac{N}{2}}^{(i)}\left(y_{\frac{N}{2}+1}^{N}, \hat{u}_{1,e}^{2i-2}\right) + 1}{L_{\frac{N}{2}}^{(i)}\left(y_1^{\frac{N}{2}}, \hat{u}_{1,o}^{2i-2} \oplus \hat{u}_{1,e}^{2i-2}\right) + L_{\frac{N}{2}}^{(i)}\left(y_{\frac{N}{2}+1}^{N}, \hat{u}_{1,e}^{2i-2}\right)} \tag{4.18}$$

and

$$L_N^{(2i)}\left(y_1^N, \hat{u}_1^{2i-1}\right) = \left[L_{\frac{N}{2}}^{(i)}\left(y_1^{\frac{N}{2}}, \hat{u}_{1,o}^{2i-2} \oplus \hat{u}_{1,e}^{2i-2}\right)\right]^{1-\hat{u}_{2i-1}} \times L_{\frac{N}{2}}^{(i)}\left(y_{\frac{N}{2}+1}^{N}, \hat{u}_{1,e}^{2i-2}\right) \tag{4.19}$$

4.2.1 Butterfly structure of SC decoder

Fig.4.4 illustrates SC decoding with $N = 8$. This decoding begins on the right of the graph with the $LR(y_i)$ of the bits calculated from the received word. $LR(y_i)$ are combined in pairs as it progresses to the left of the graph. There are $\log_2(N) = 3$ stages each containing $N = 8$ nodes. The decoded bits are located to the left of the graph. For each node, the two incoming LRs, denoted by L_a and L_b are combined to produce the LR of the node. Two functions for calculating LRs, for f and g nodes, are used. These functions are given in 4.18 and 4.19. The nodes for which the function f are used are labeled by f, i.e., white, and for those the function g are used are labeled by g, i.e., gray. In general, the LRs of the level j are calculated from the LRs of the level $j-1$. Equations 4.18 and 4.19 can be expressed for a single node as

$$f(L_a, L_b) = \frac{1 + L_a . L_b}{L_a + L_b}$$

$$g(L_a, L_b, \hat{u}_{sum}) = {L_a}^{(1-2\hat{u}_{sum})} L_b \rightarrow g(L_a, L_b, \hat{u}_{sum}) = \begin{cases} L_a . L_b & if\ \hat{u}_{sum} = 0 \\ \frac{L_b}{L_a} & if\ \hat{u}_{sum} = 1 \end{cases} \tag{4.20}$$

where $\hat{u}_{sum}$ is the partial binary sum of the previously estimated bits. These are the binary sums at the labels of the nodes. The value of $\hat{u}_{sum}$ determines whether the function g performs a multiplication or a division.

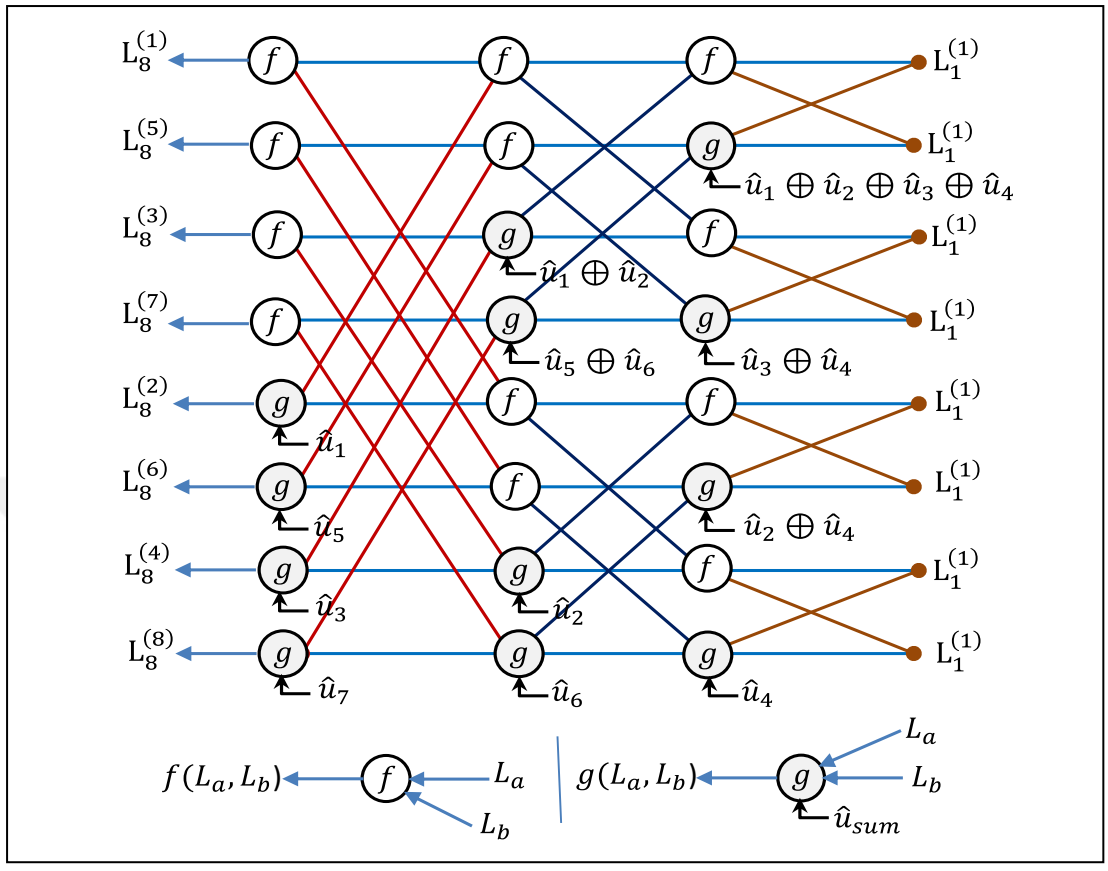

Figure 4.4 Butterfly-based structure of SC decoder for $N = 8$.

Example: For a transmission system employing polar codes, we have codeword length is $N = 8$, code rate $R = 1/3$ and AWGN is used. The information set is $A = \{4,6,8\}$. The values of information bits are all one. The frozen bits u_1, u_2, u_3, u_5, u_7 are chosen as 0. The calculations of the likelihood ratios for the information bits are shown in Fig.4.5.

The likelihood ratio calculation for the fourth bit u_4 is depicted in Fig.4.5.

In Fig.4.5, the channel observations are given as $\{0.4800,\ -2.7848,\ 0.8946,\ -0.7669,\ 0.5275,\ -2.3709,\ 1.1148,\ -0.7829\}$.

Since the frozen set is $\{1, 2, 3, 5\}$, then we know that u_1 is a frozen bit and its value is 0. Similarly u_2 and u_3 are both frozen bits and there is no need to decode them. For the decoding of u_4, the likelihood ratio is calculated from right to left. The likelihood ratio is found as $L_8^{(4)} = 0.0435$, and since $L_8^{(4)}$ is smaller than 1, then u_4 is decided to be 1.

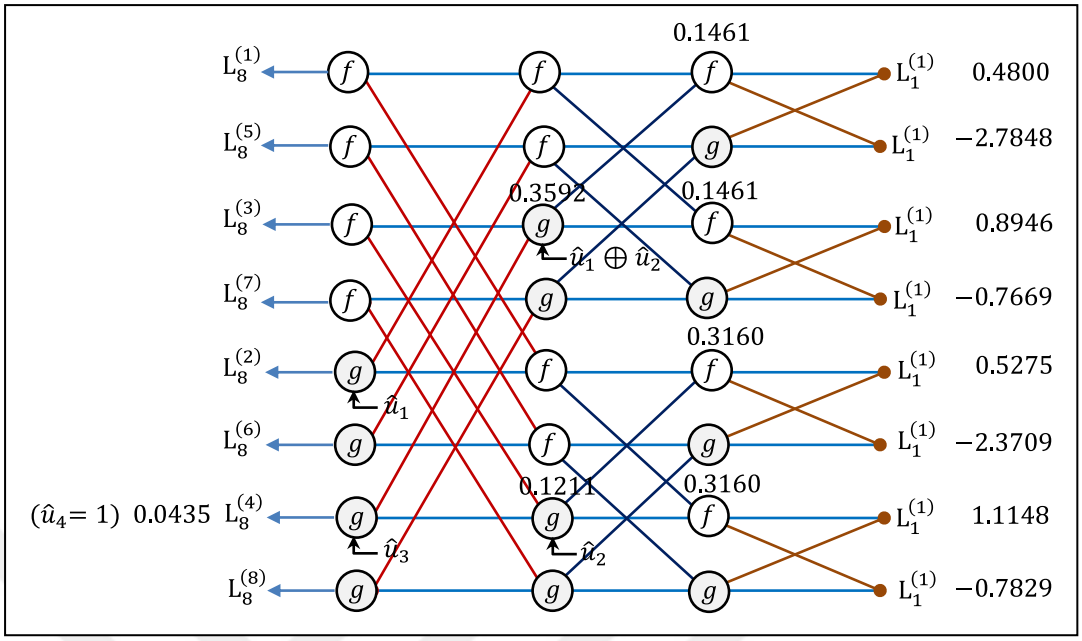

Figure 4.5 Factor graph for the decoding of bit u_4.

The rest of the bits $u_5, \ldots, u_8$ are decoded in a similar manner and the full result is shown in Fig.4.6.

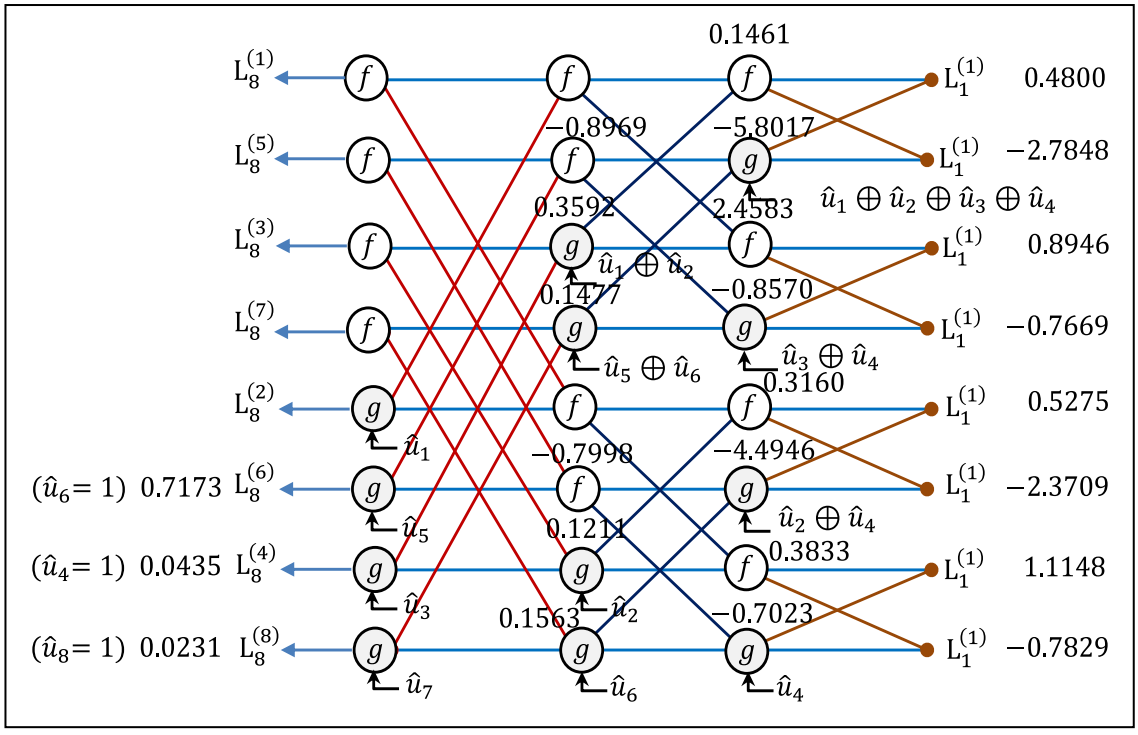

Figure 4.6 Factor graph for the decoding of data bits for $N = 8$.

4.2.2 Tree structure of SC decoder

The tree structure for the decoding of polar codes can be constructed and in this structure the previously decoded bits are distributed to the nodes considering the left child nodes and the right child nodes on the decoding tree. Bit distribution process is performed after the total number of levels and its indices are found, and number of nodes assigned in each level is determined.

The distribution of the previously decoded bits to the nodes is used to determine the active nodes, classified as $g-nodes$, that appear at certain levels and the passive nodes, classified as $f-nodes$.

The decoding process for frame length N consists of two steps. In the first step the previously decoded bits are distributed to the nodes as explained in algorithm-1 [46,48].

Algorithm 1: The distribution of decoded bits to the nodes

Input N frame length and received data bits.
1: **If** i is odd, then node-check-bit = x_i and
2: Left child-node: $L_c = x_{1,o}^{i-1} \oplus x_{1,e}^{i-1}$
3: Right child-node: $R_c = x_{1,e}^{i-1}$
4: **else**
5: Left child-node: $L_c = x_{1,o}^{i} \oplus x_{1,e}^{i}$
6: Right child-node: $R_c = x_{1,e}^{i}$
7: **end**
8: **If** $i = 1$ then
9: Terminate
10: **else**
11: $i = i/2$
12: Go to step-1 and repeat $1-6$ for the left-child and right-child nodes
13: **end**

Let's consider a numerical example for the decoding operation. Assume that the codeword length for polar code is $N = 8$, and assume that the first 5 bits are decoded as

$u_1^5 = [1\ 0\ 1\ 1\ 1]$, and we want to decode the 6^{th}bit.

As it is clear that the decoded bit vector $u_1^5 = [1\ 0\ 1\ 1\ 1]$ contains an odd number of bits. The decoded bit u^5 is assigned to the topmost head node as shown in Fig.4.7.

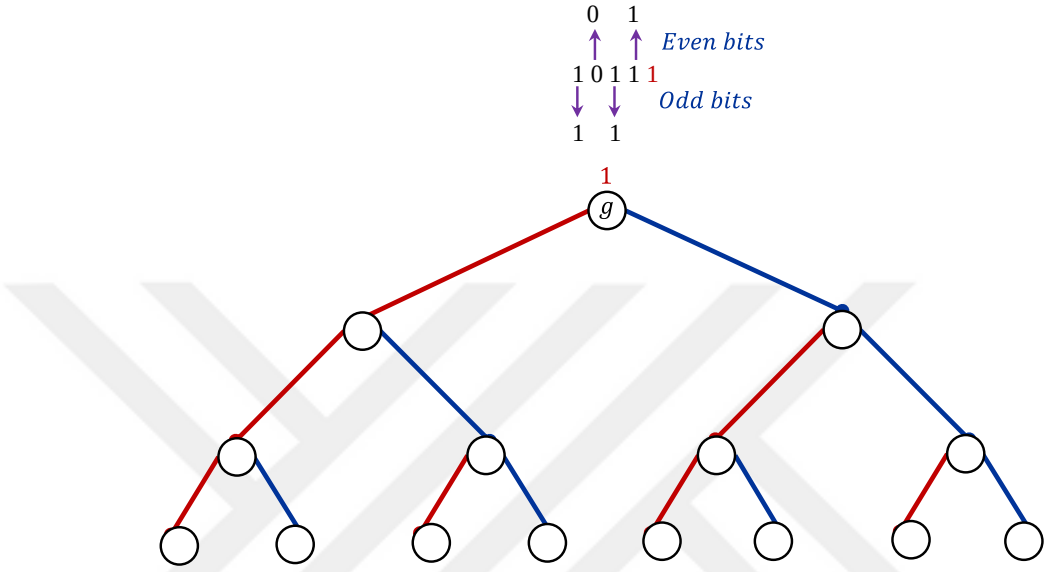

Figure 4.7 The first step of the decoded-bit distribution to the nodes.

The distribution of the all the previously decoded bits to the nodes is illustrated in Fig.4.8.

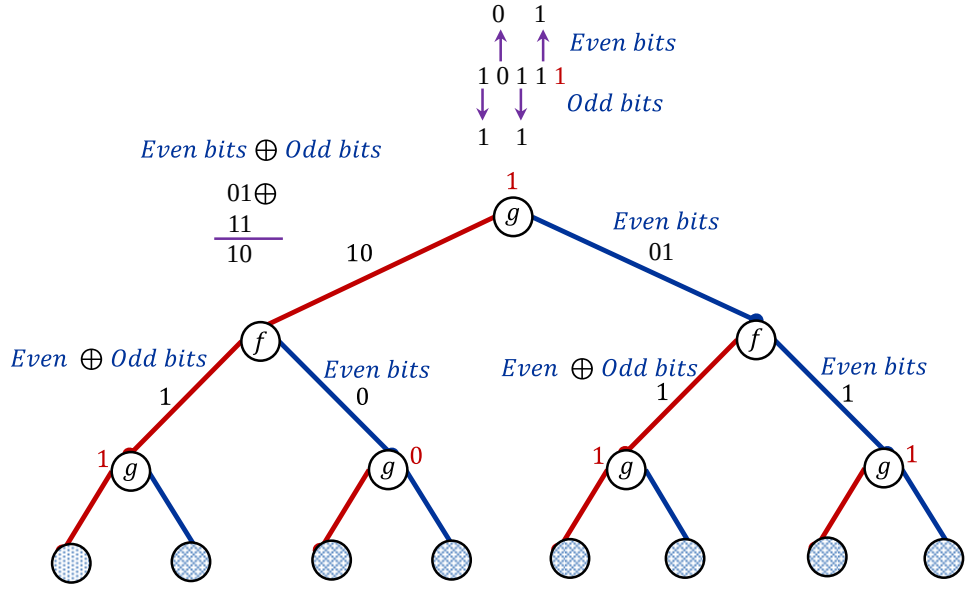

Figure 4.8 Distribution of the decoded bits to the nodes.

The second step involves the calculation of the likelihood values recursively. Since each $LR : L_N^{(i)}$ is associated with the $W_N^{(i)}$ channel, for the computation of $L_N^{(i)}$ a similar path to the one in Fig.4.4 , used for the construction of $W_N^{(i)}$, is followed. To calculate $L_N^{(i)}$ for $W_N^{(i)}$ we use the likelihood ratios of the channels participated in the formation of the channel $W_N^{(i)}$. Each node has a label. If there is a bit assigned to a node, i.e., a g-node, then 4.19 is used to compute the node LR, otherwise 4.18 is used for f-node

4.3 LLR based SC decoder version

The SC decoder presented in the previous section is based on the LR, likelihood ratio, calculations and the functions f and g require multiplications and divisions which are difficult to implement in practice. Leroux and others [49] proposed log-domain SC decoding algorithm where $LLRs$, log-likelihood ratios, are used for the calculations of f and g functions. Log-domain version of 4.20 can be written as

$$f(LL_a, LL_b) = 2\tanh^{-1}\left(\tanh\left(\frac{LL_a}{2}\right).\tanh\left(\frac{LL_b}{2}\right)\right)$$
$$g(LL_a, LL_b, \hat{u}_{sum}) = LL_a.(-1)^{(\hat{u}_{sum})} + LL_b = \begin{cases} LL_a + LL_b & if\ \hat{u}_{sum} = 0 \\ -LL_a + LL_b & if\ \hat{u}_{sum} = 1 \end{cases} \quad (4.21)$$

where $LL_a \triangleq \ln(L_a)$ and $LL_b \triangleq \ln(L_b)$. The g function can easily be implemented by an adder/subtractor conditioned by the $\hat{u}_{sum}$ bit in hardware while the implementation of the f function is still complex. As in LDPC codes, the min-sum approximation [49] can also be exploited to reduce the complexity of f, and we get

$$f(LL_a, LL_b) \triangleq sign(LL_a).sign(LL_b).\ min(|LL_a|, |LL_b|) \quad (4.22)$$

The computational complexity of the SC decoder is $O(N \log N)$ [1].

4.4 Alternative tree representation of the SC decoder

The SC decoding process can be seen represented using a code tree. The decoding starts with the root which has two edges with labels 0 and 1, and the metrics $W_N^{(i)}(y|0)$ and $W_N^{(i)}(y|1)$, respectively. The SC decoding chooses the path with the greater metric and the other is discarded. The chosen edge, in turn, gives two edges with labels 0 and 1 with the metrics $W_N^{(i)}(y, \hat{u}_1|0)$and $W_N^{(i)}(y, \hat{u}_1|1)$, respectively. Usually at each level the $\hat{u}_i$ bit is decoded by comparing the two metrics $W_N^{(i)}\left(y, \hat{u}_1^{i-1}|0\right)$ and $W_N^{(i)}\left(y, \hat{u}_1^{i-1}|1\right)$. If i belongs to a frozen index, then the SC decoder gives $\hat{u}_i = 0$. This procedure continues up to the leaf nodes.

Figure 20, shows an example of the SC decoding procedure for $N = 4$ and $k = 4$.

This tree is made up of 4 levels, where each level represents a decoded bit. The value associated with each node is the LR-based metric for the decoding path from the root node to the current node. The bold red edges in the figure show the SC decoding path.

The number written next to each of the nodes is the metric of the decoding path from the root to that node. The nodes that are extended during the SC decoding procedure are represented by the numbered circles and the corresponding numbers indicate the order of processing. The black circles represent nodes that are visited (whose path metric is calculated) but have not been retained, and the gray circles are those which are not visited during the search process. The path taken by the decoding process is not guaranteed to be the most likely one. As it is seen from the example shown in Fig.4.9 , the bit string 1000 has the largest probability of all paths of length N, but it failed in the competition at the first level. The decoding sequentially evaluates the metrics : $W_N^{(i)}(y|0) = 0.55$ $W_N^{(i)}(y, \hat{u}_1 = 0|0) = 0.3$, $W_N^{(i)}(y, \hat{u}_1^2 = 00|1) = 0.25$ and $W_N^{(i)}(y, \hat{u}_1^3 = 001|1) = 0.2$. So the decoder output becomes $\hat{u}_1^4 = (\hat{u}_1, \hat{u}_2, \hat{u}_3, \hat{u}_4) = (0011)$.

The output of the decoder corresponding to that of the ML decoder is the path of length 4 which has the greatest metric at the lowest level. For this example, 1000 with the path metric $W_N^{(i)}(y, 100|0) = 0.36$ corresponds to the optimal path.

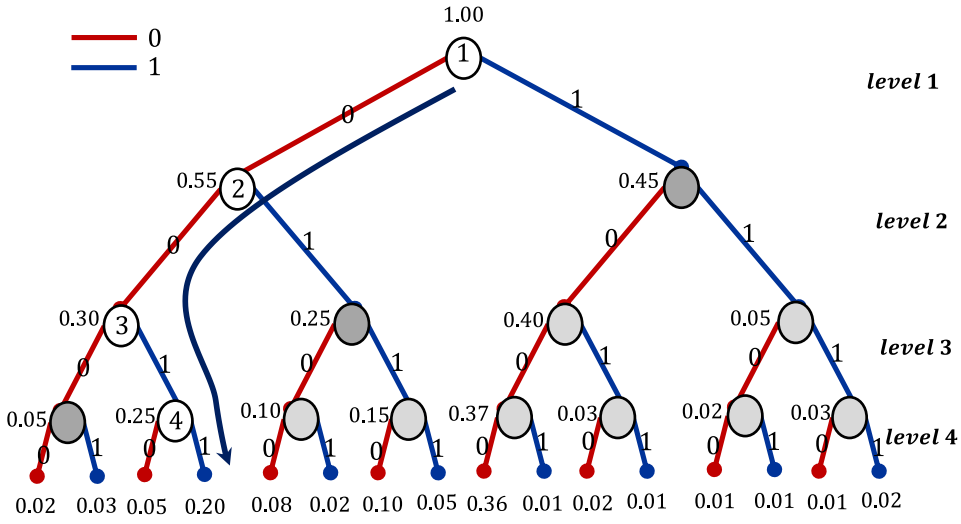

Figure 4.9 SC decoding on a tree for $N = 4$ and $K = 4$.

Decoding fails if $\hat{u}_1^N \neq u_1^N$. The complexity of the decoding algorithm is determined primarily by the calculation of LRs. Using a space-efficient structure [50] to implement the SC decoder, the time and space complexities can be reduced from $O(N \log N)$ to $O(N)$. Although the polar codes asymptotically achieve the capacity of the channel, empirical studies have shown that for finite block lengths, for low and medium lengths, the polar codes with SC decoding show worse performance than that of turbo codes and LDPC codes. To improve the performance of SC decoding algorithm, Tal and Vardy proposed a variant of SC decoding algorithm called the SC List Decoding (SCL).

4.5 Successive Cancellation List (SCL) Decoding

The sequential decoding nature of SC algorithm is a bottle neck for the performance of polar codes. Since if a bit is wrongly decoded, there is no chance of correcting it in the rest of the decoding process.

To improve the performance of SC decoding algorithm, Tal and Vardy [28] proposed the SCL. The SCL algorithm produces a number of bit strings, L known as the list size, each is a candidate for the transmitted data word, and the best one is decided as the output of the decoder. Like the SC, the SCL can be represented using a tree.

4.5.1 The SCL decoder seen in the form of a code tree

Like SC, the SCL decodes the input bits $\hat{u}_i, i = 1, \ldots, N$, successively one-by-one. Unlike the SC where only one path is kept after processing at each level, SCL allows simultaneous exploitation of up to L candidate paths for the next level. For each level, the SCL decoder doubles the number of candidate paths for each $\hat{u}_i$ bit, i.e., $\hat{u}_i = 0$ and $\hat{u}_i = 1$, as shown in Fig.4.10. If $2L$ candidate paths are obtained, then a pruning procedure is used to select the most probable L paths having the largest path metrics. These L paths are stored in a list for processing at the next level. Note that for a frozen bit, the number of candidate paths is not doubled because its value is known. At the end of the decoding process, when the leaf nodes are arrived in, the most probable path having the largest path metric in the list is chosen as the output of the decoder.

In Fig.4.10, SCL decoding operation with $L = 2, N = 4$, and $K = 4$ is illustrated [4]. At level 1, the SCL has two path for $\hat{u}_1 = 0$ and $\hat{u}_1 = 1$. Then, at level 2, the path number is doubled. Since the size of the decoder list is $L = 2$, after calculation of every $2L = 4$ new path metrics, the SCL decoder selects the $L = 2$ paths having the largest path metrics

.

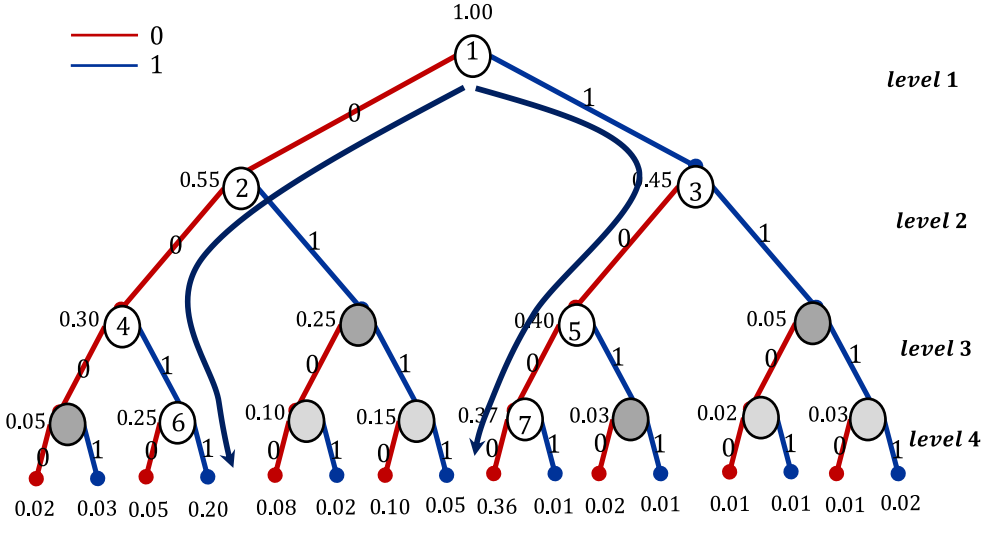

Figure 4.10 SCL decoding with $N = 4$, $K = 4$ and $L = 2$.

This procedure is repeated at each level. At the bottom level, we have $L = 2$ candidate paths, 0011 with a metric of 0.2 and 1000 with a metric of 0.36, and the path with larger metric is the winner.

Based on SCL, another variant of SC called SC stack, SCS, SC stack, is proposed by Chen and others [6]. The authors proposed a stack instead of a list. They showed that SCS gives similar results as that of SCL.

4.5.1.1 Performance of SCL decoding algortihm

A larger list size results in better performance of the SCL decoder. In Fig.4.11 [4], the results of simulations for a block length N = 2048 bits and a rate $R = 1/2$ are shown. List size of $L = 32$ achieves the ML decoder's performance.

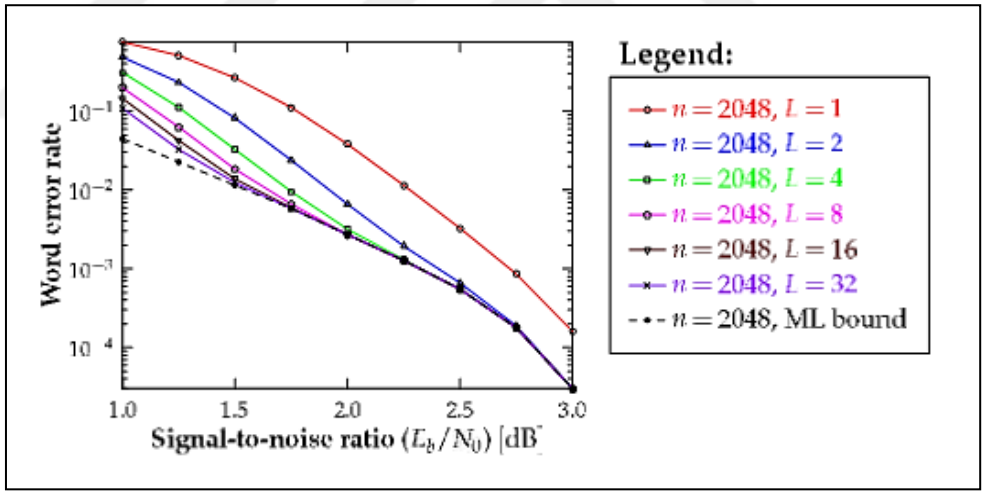

Figure 4.11. List decoding performance for length $N = 2048$ and rate $R =$ ½ [4]

4.5.1.2 The complexity of SCL Decoding

Since L decode paths are maintained simultaneously, and each path has a complexity of $O(N)$, then the complexity of the SCL decoder is $O(LN)$. Since the code tree has N levels, a direct implementation of the SCL decoder would need $O(LN^2)$ calculations. However, a smart choice of data structures and the recursive nature of the calculations inspired the

authors of [4] to develop a technique called “lazy copy”, a copy-on-write mechanism, based on the structure of memory sharing between candidate paths to reduce the complexity. They showed that the SCL decoder can be implemented with a computational complexity of $O(N \log N)$. SCL decoder can also be implemented using LLR based metrics.

4.5.1.3 LLR based SCL decoding

The original SCL algorithm [4] is described using LRs, while most of the processing and storage modules in modern digital transmission systems are LLR based [51]. Balatsoukas and others [52] shown that the SCL decoding algorithm can be formulated exclusively using $LLRs$. They used some useful properties of the LLR based formulation to reduce the complexity of the SCL decoding algorithm. The LLR based SCL decoder can easily be incorporated into existing communication systems, while, the LR based decoder would require additional processing steps to convert the channel's $LLRs$ to LRs. In addition, the LLR based SCL decoding allows a space-efficient and numerically stable implementation, a significant reduction in the size of the previous hardware architecture as well as an increase in the maximum operating frequency [52]. Yuan and Parhi [51] proposed n LLR based SCL algorithm by redefining the updating of path metrics using $LLRs$. The proposed approach can achieve both latency and hardware complexity reduction.

4.5.1.4 SCL decoder concatenated with CRCs (Cyclic Redundancy Check)

Tal and Vardy [4] observed in their simulations that in most cases when the SCL decoder fails, the transmitted codeword was among the L paths of the list, but it was not the most likely one, and therefore it was not chosen as the output of the decoder.

A decoding error occurs because there is another more likely path that is selected as the output of the decoder. Note that in such a situation, the ML decoder would also fail. Tal and Vardy concluded that the performance of polar codes would be further improved significantly with a tool called "genie aided" capable of identifying the transmitted

codeword if it is in the list. This can easily be implemented using the cyclic redundancy check, CRC, precoding [4]. It consists of adding more non-frozen bits to the polar code. The SCL decoder first discards the paths among the L candidates that do not pass the CRC and then chooses the most likely path from the remaining ones. However, there is a tradeoff between the length of the CRC and the performance gain. Indeed, the longer the CRC, the better it can detect incorrect code words, and the more it degrades the performance of the polar code due to the increase in its rate. The choice of CRC length is guided by the size L of the SCL decoder list and the SNR value considered. We give three examples of different CRCs of lengths 4, 8, and 16 whose generator polynomials are

$$\begin{aligned} g(x) &= x^4 + x + 1 && for\ CRC\ 4bits \\ g(x) &= x^8 + x^7 + x^6 + x^4 + x^2 + 1 && for\ CRC\ 8\ bits \\ g(x) &= x^{16} + x^{15} + x^2 + 1 && for\ CRC\ 16\ bits. \end{aligned} \tag{4.23}$$

The empirical results of [4] show that a polar code of length, $N = 2048$ and $R = ½$. decoded by SCL with $L = 32$, assisted by 16-bit CRC, achieves better performance in terms of binary error rate BER than turbo-codes and LDPC codes with frame length $N = 2304$ used in the WiMAX standard with the same rate.

CHAPTER 5

High-Performance Low Latency Parallel Successive Cancelation Decoder Structures for Polar Codes

Polar codes are decoded using successive cancellation (SC) algorithm where likelihood ratios (LRs) for data bits are calculated sequentially, and decisions are made using the calculated LRs. During the decoding of an information bit, the decision results for the predecessor bits are used, and a wrongly decided predecessor bit has a negative effect on the accurate calculation of the LR for the information bit being decoded. In the SC algorithm, when LR=1, the information bit is decoded as $\hat{u}_i = 0$, however, such a decision has a 50% chance of being correct. In this chapter, we propose improved polar decoders utilizing a number of SC decoders. We consider the case of $LR = 1$ and propose polar decoder structures for the more accurate calculation of the LRs of the successor bits.

In this thesis, we propose a new decoding approach for polar codes considering the case of $LR = 1$. The proposed method employs multi-SC decoders constructed in parallel with enhanced decision functions to determine the erroneous bits that degrade the code performance leading to error propagation. The proposed technique provides a flexible configuration and leads to the pruning of unnecessary path searching operations, which reduce the decoding complexity. Multi-Parallel SC decoding shows a significant performance improvement compared to the original SC decoding.

5.1 Notation and SC Decoding

In this thesis, we write $W: X \rightarrow Y$ to denote a generic binary discrete memoryless channel, B-DMC with input alphabet X, output alphabet Y and transition probabilities $W(y|x)$, $x \in X$, $y \in Y$. For a BEC, the input alphabet X is chosen from the binary set $\{0, 1\}$ while Y and the transition probabilities may take arbitrary values.

$y_1^N = (y_1, y_2, \dots, y_N)$ are the observations of the code bits, the codeword $x_1^N = (x_1, x_2, \dots, x_N)$ is obtained via encoding of the information bits $u_1^N = (u_1, u_2, \dots, u_N)$.

At the destination side, using the received word $y_1^N = (y_1, y_2, \dots, y_N)$, information bits are estimated successively using the likelihood ratios of the bits appearing in code structure. In this thesis, binary erasure channel, BEC, is employed for performance evaluation. A bit transmitted through the binary erasure channel is either received correctly with probability $1-\epsilon$ or lost with probability ϵ. It is shown in [1] that the LRs of the information bits can be calculated recursively as

$$\mathrm{L}_{\mathrm{N}}^{(2\mathrm{i}-1)}\left(\mathrm{y}_1^{\mathrm{N}}, \hat{\mathrm{u}}_1^{2\mathrm{i}-2}\right) = \frac{\mathrm{L}_{\frac{\mathrm{N}}{2}}^{(\mathrm{i})}\left(\mathrm{y}_1^{\frac{\mathrm{N}}{2}}, \hat{\mathrm{u}}_{1,\mathrm{o}}^{2\mathrm{i}-2} \oplus \hat{\mathrm{u}}_{1,\mathrm{e}}^{2\mathrm{i}-2}\right)\mathrm{L}_{\frac{\mathrm{N}}{2}}^{(\mathrm{i})}\left(\mathrm{y}_{\frac{\mathrm{N}}{2}+1}^{\mathrm{N}}, \hat{\mathrm{u}}_{1,\mathrm{e}}^{2\mathrm{i}-2}\right)+1}{\mathrm{L}_{\frac{\mathrm{N}}{2}}^{(\mathrm{i})}\left(\mathrm{y}_1^{\frac{\mathrm{N}}{2}}, \hat{\mathrm{u}}_{1,\mathrm{o}}^{2\mathrm{i}-2} \oplus \hat{\mathrm{u}}_{1,\mathrm{e}}^{2\mathrm{i}-2}\right)+\mathrm{L}_{\frac{\mathrm{N}}{2}}^{(\mathrm{i})}\left(\mathrm{y}_{\frac{\mathrm{N}}{2}+1}^{\mathrm{N}}, \hat{\mathrm{u}}_{1,\mathrm{e}}^{2\mathrm{i}-2}\right)} \tag{5.1}$$

$$\mathrm{L}_{\mathrm{N}}^{(2\mathrm{i})}\left(\mathrm{y}_1^{\mathrm{N}}, \hat{\mathrm{u}}_1^{2\mathrm{i}-1}\right) = \left[\mathrm{L}_{\frac{\mathrm{N}}{2}}^{(\mathrm{i})}\left(\mathrm{y}_1^{\frac{\mathrm{N}}{2}}, \hat{\mathrm{u}}_{1,\mathrm{o}}^{2\mathrm{i}-2} \oplus \hat{\mathrm{u}}_{1,\mathrm{e}}^{2\mathrm{i}-2}\right)\right]^{1-\hat{\mathrm{u}}_{2\mathrm{i}-1}} \times \mathrm{L}_{\frac{\mathrm{N}}{2}}^{(\mathrm{i})}\left(\mathrm{y}_{\frac{\mathrm{N}}{2}+1}^{\mathrm{N}}, \hat{\mathrm{u}}_{1,\mathrm{e}}^{2\mathrm{i}-2}\right) \tag{5.2}$$

The decision is made according to

$$\hat{u}_i = \begin{cases} 0 & if \;\; LR(\hat{u}_i) \geq 1 \\ 1 & otherwise \end{cases} \tag{5.3}$$

where $LR(\hat{u}_i)$ is defined as

$$LR(\hat{u}_i) = \frac{Prob(\hat{u}_i = 0|y)}{Prob(\hat{u}_i = 1|y)}.$$

5.2 Proposed M-Parallel SC Decoding Algorithm

In SC decoding of polar codes [1], whenever $LR(\hat{u}_i) = 1$ the decision is made as $\hat{u}_i = 0$ according to 5.3. However, in such an approach, we have a 50% chance of making a correct decision. A wrong decision will certainly have negative effects on the determination of the succeeding bits. Considering this issue, we propose multi- parallel structures for SC decoding operations.

5.2.1 Multi-Parallel SCD (MP-SCD) for M=2

Successive cancelation decoding operation is a sequential decoding operation. The decoding decision for the i^{th} bit, u_i, affects the decoding decisions of the successive bits, i.e., bits $u_j, j > i$. The absolute value of the log-likelihood ratio, i.e., $|LLR|$, is an indicator for the accuracy of the decision. Large $|LLR|$ implies a more robust decision. And a wrong estimation for the information bit u_i will push the values of $|LLRs|$ towards 0 for the successive bits.

To alleviate the information loss and improve the decision robustness, considering the case of $LR(u_i) = 1$, we propose a structure as depicted in Fig.5.1 where we utilize two SC decoders, one of which uses the decision logic

$$PD_1\colon \hat{u}_i = \begin{cases} \mathbf{0}, & \boldsymbol{if\ \ LR(u_i) = 1} \\ 0, & if\ \ LR(u_i) > 1 \\ 1, & otherwise \end{cases} \tag{5.4}$$

whereas the other employs

$$PD_2\colon \hat{u}_i = \begin{cases} \mathbf{1}, & \boldsymbol{if\ \ LR(u_i) = 1} \\ 0, & if\ \ LR(u_i) > 1 \\ 1, & otherwise. \end{cases} \tag{5.5}$$

Once the decoding operation for all information bits is complete, the sum of the absolute log-likelihood ratios, i.e. $LLRs$, is performed according to

$$S_{PD_1} = D1\colon \sum_i |LLR_i| \quad S_{PD_2} = D2\colon \sum_i |LLR_i| \tag{5.6}$$

where LLR_i indicates the log-likelihood ratio for information bit u_i for the corresponding decoders. The winner decoder is chosen considering

$$PD_w = \begin{cases} PD_1 & if\ S_{PD1} \geq S_{PD2} \\ PD_2 & otherwise. \end{cases} \tag{5.7}$$

The estimated bits of the winner decoder are accepted as the result of the decoding operation. The overall logic for the proposed decoding operation is illustrated graphically in Fig.5.1 and Fig.5.4.

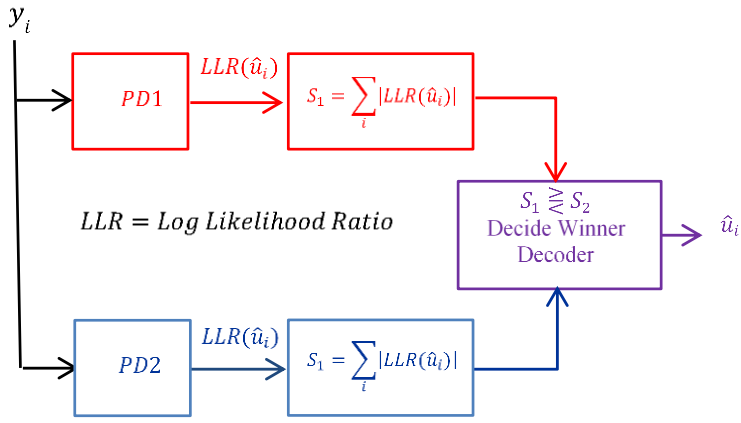

Figure 5.1 Block diagram for $M = 2$ parallel SC decoding operation.

The structure is shown in Fig.5.1 corresponds to the case $M = 2$ i.e., we have two parallel decoders. The sum of the absolute values of the LLRs is calculated for each decoder, and the decoder with a larger summation result is chosen for decision operation. Unlike the SC decoder where only one path is reserved at each level, the multi-parallel algorithm utilizes M different searching paths. Therefore, it is more likely for the M-parallel algorithm to find the desired path than the SC algorithm. In Fig.5.2 and Fig.5.3, two examples for $N = 8$ considering $LR = 1$ cases are depicted.

The example depicted in Fig.5.2 illustrates the worst case. It is seen from the example that the decoded sequences can be 000 or 111, in this case just the first bit can be exactly correct and the number of paths is not greater than $M = 2$. In the example depicted in Figure 25, the 1^{st}, 3^{rd} bits are lost and the 2^{nd} bit is normally detected as1.

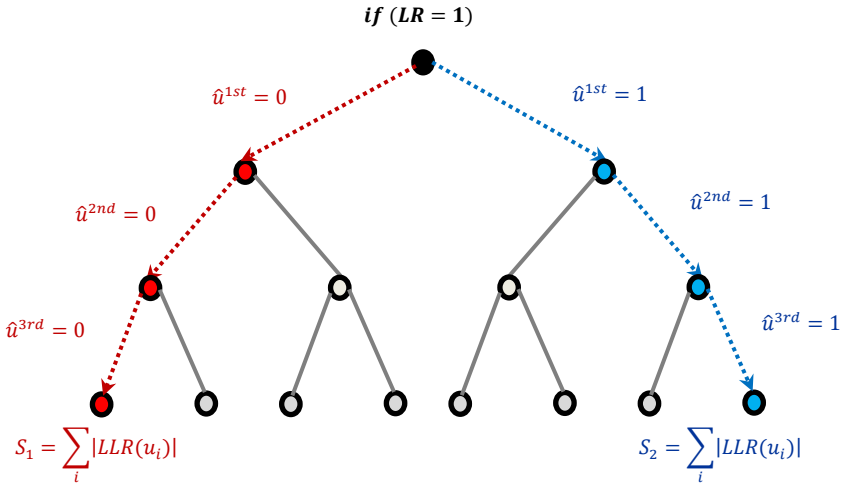

Figure 5.2 Decoding procedure for case 1, where the $1^{st}, 2^{nd}$ *and* 3^{rd} bits are lost

The possible decoded words can be $\{010,111\}$. In the decoded sequence, the first and second bits are true, but the third bit can be true with a probability of 0.5.

In this case, we note that the path searching is changed by the second bit for which it is assumed that $LR < 1$. The path having the highest absolute logarithmic likelihood sum is chosen for the determination of the decoded sequence

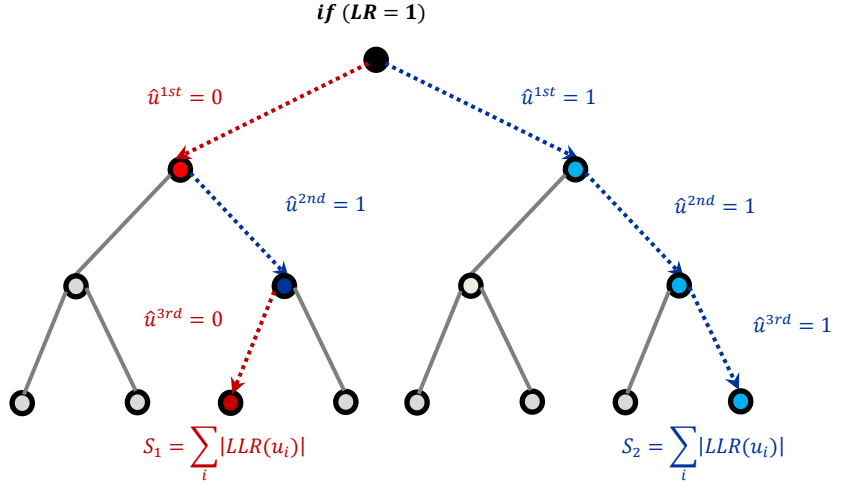

Figure 5.3 Decoding procedure for case 2, where the 1^{st} and 3^{rd} bits are lost but the 2^{st} bit is normally detected as 1.

5.2.2 Multi-Parallel SCD (MP-SCD) for M=4

To improve the decision accuracy, we can increase the number of parallel decoders, which handle $LR(u_i) = 1$ case in different ways. In Fig.5.4, the multi-parallel SCD structure for $M = 4$ is depicted.

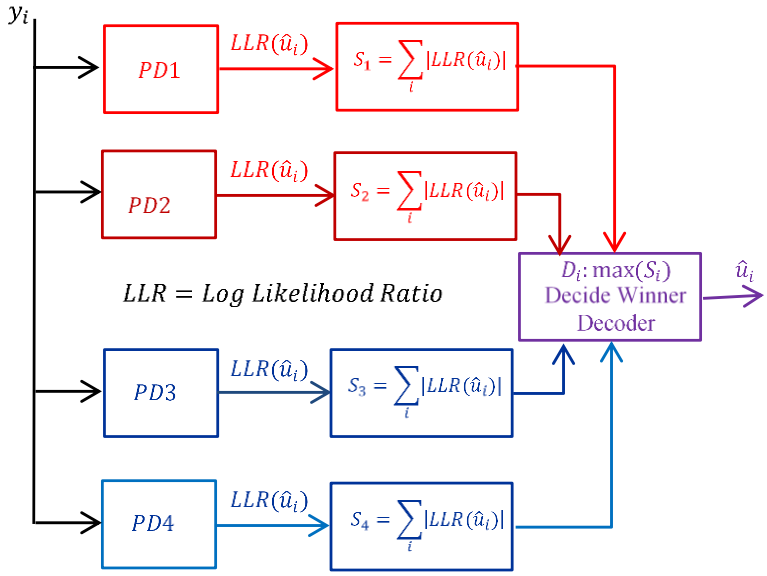

Figure 5.4 Block diagram for $M = 4$ parallel SC decoding operation

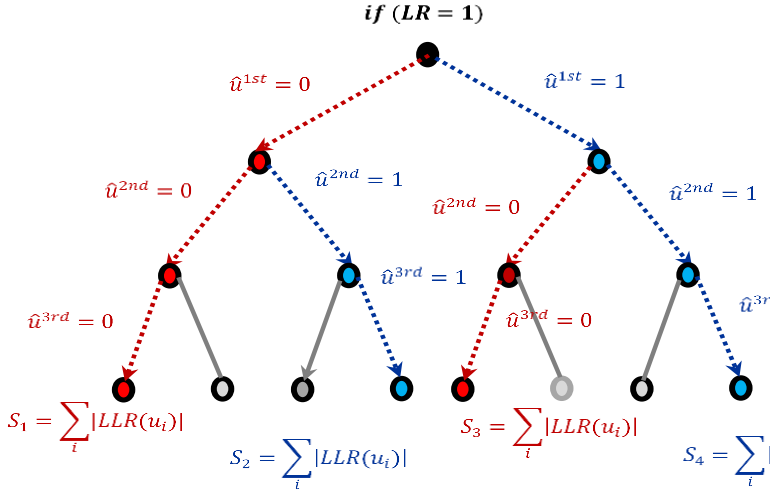

Figure 5.5 Decoding procedure in case 1, where the $1^{st}, 2^{nd}$ and 3^{rd} bits are lost

In Fig.5.4, PD_1 always makes the decision '0' when $LR = 1$, PD_2 makes the decision '0' only once when $LR = 1$, and it makes the decision '1' for the other $LR = 1$ cases, PD_4 always makes the decision '1', PD_3 makes the decision '1' only once when $LR = 1$, and it makes the decision '0' for the other $LR = 1$ cases. Two examples for $M = 4$ are provided in Fig.5.5 and Fig.5.6.

In Fig.5.5 for $M = 4$, we consider the case where the first 3 bits are erased, and in this case, the candidate sequences for the decoder output are {000, 110, 001, 111}. For the example of Fig.5.6 for $M = 4$, we consider the case where the 1^{st} and 3^{rd} bits are lost but the 2^{nd} bit is normally decoded as 1, and in this case, the candidate sequences for the decoder output are {010, 110, 011, 111}. As it is seen from Fig.5.6 the second bit, which acts as a path corrector, alters that search path and the winner path has the largest absolute logarithmic likelihood sum.

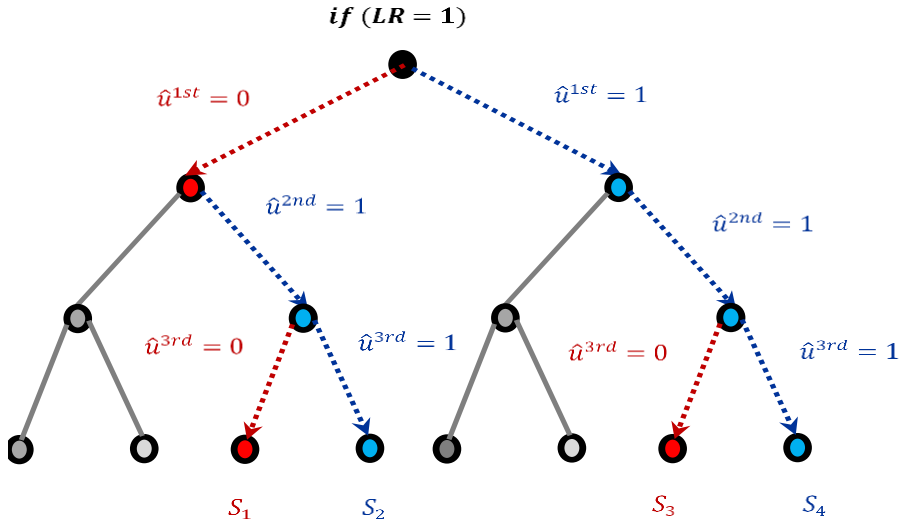

Figure 5.6 Decoding and procedure in case 2, where the 1^{st} and 3^{rd} bits are lost and the 2^{nd} bit is decoded as 1.

5.3 Simulation Results

Computer simulations are performed using frame lengths of 1024 and 128 for binary erasure channels having erasure probabilities 0.5, bit-error-rate (BER), and frame-error-rate (FER) performance curves for moderate frame lengths are obtained as shown in Figs. 5.7, 5.8, 5.9 and 5.10. It is seen from Fig.5.7 that two types of simulation curves are available.

In Figures, the lines with label '*' indicate the simulation results assuming that perfect channel knowledge is available at the receiver side, and if the sequence associating with one of the decoded paths matches the transmitted frame, it is accepted as the decoder result, otherwise, the path having the largest absolute LLR sum is chosen. It is seen from the performance curves that the code performance increases as the number of parallel branches increases, and when the parallel branch number is 4, the performance of the proposed method approaches to the performance for which perfect channel knowledge is available. We obtain significant performance improvement over SCD.

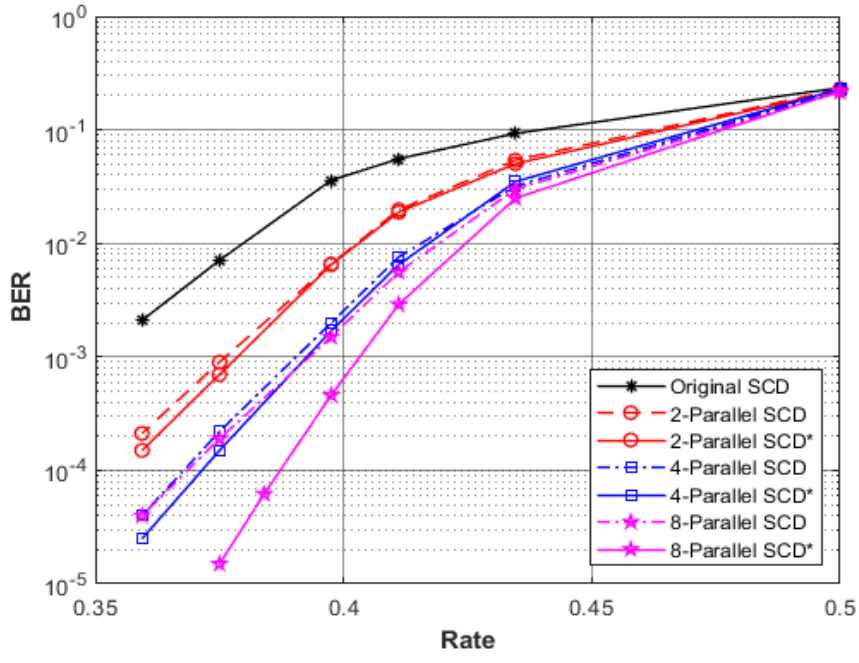

Figure 5.7 BER performance under different M-size for frame length P(1024) over binary erasure channel.

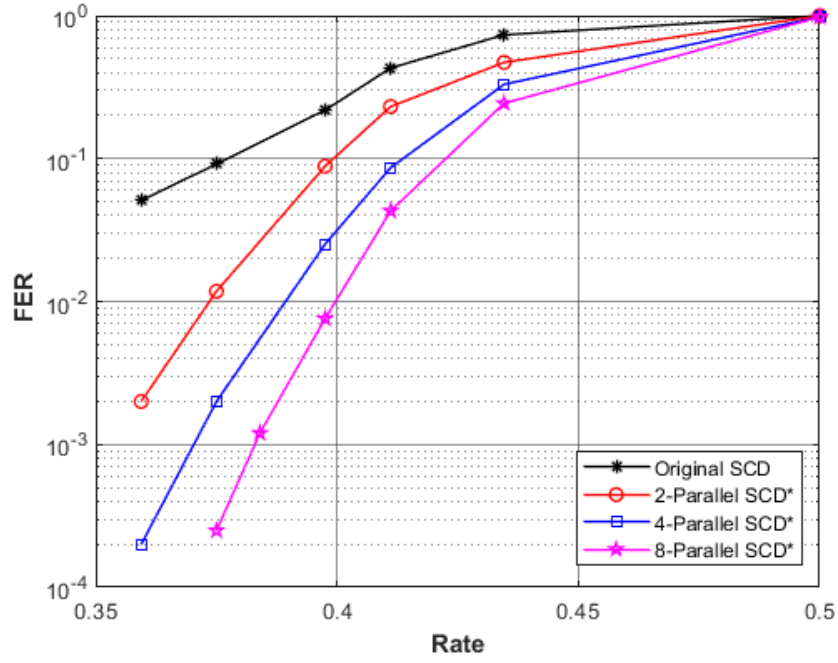

Figure 5.8 FER performance under different M-size for frame length P(1024) over binary erasure channel.

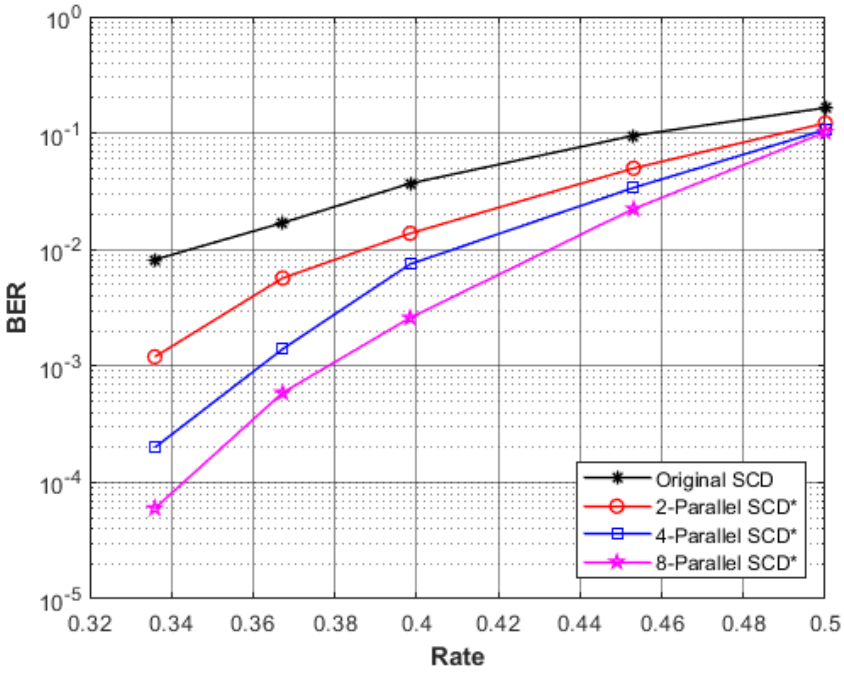

Figure 5.9 BER performance under different M-size for frame length P(128) over binary erasure channel.

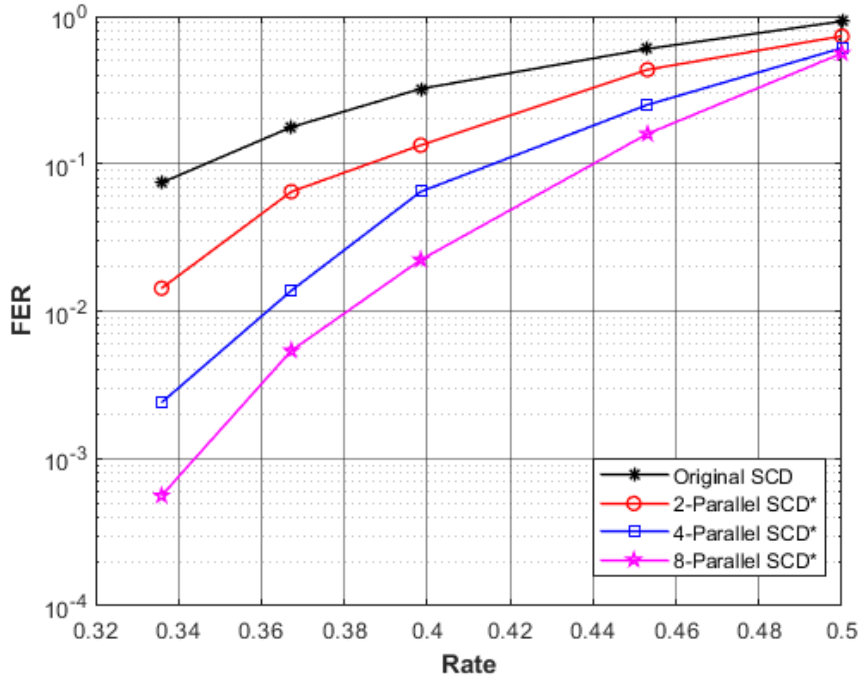

Figure 5.10 FER performance under different M-size for frame length P(128) over binary erasure channel.

5.4 Conclusions

In SC decoding of polar codes, when $LR = 1$ for an information bit, the decision is made in the favor of 0, however, in such a decision an uncertainty occurs, and in this instance polar decoder has a 50% chance of making a correct decision. Considering this circumstance, we proposed novel polar decoding methods, which consider two decisions i.e., '0' and '1', for $LR = 1$ at the node of the decoding tree.

In one approach, path doubling is only done once when LR=1, which corresponds to two parallel decoders, i.e., $M = 2$. Moreover, in the next approach, we considered path doubling twice, i.e., a path doubling is made when $LR = 1$ and another path doubling is made when $LR = 1$ and no more, which corresponds to four parallel decoders, i.e., $M = 4$. Simulation results indicate that the proposed approaches show significant performance improvement over classical SC decoder.

CHAPTER 6

Improving the Performance of Polar Decoders Using Virtual Random Channels.

6.1 Background

Noise is typically thought s an unwanted signal or disturbance that leads to less channel capacity, worse detection performance. Hence, noise is often removed or mitigated by a variety of filters and signal processing algorithms. The concept of stochastic perturbation opens a new perspective to benefit from adding virtually generated noise. Adding white noise to a non-linear system to reach a stable model is studied for the first time in [53]. It is shown that random noise can have a positive effect and plays an important constructive role in many information processing systems and algorithms [54]. Moreover, adding independent noise to the received data can improve the detectability of weak signals [55]. Random noise helps to enhance the detection of weak information signals in nonlinear systems under some type of threshold. Noise enhancement is used in many fields such as dithering in quantization, stochastic optimization techniques such as genetic algorithms or simulated annealing, and learning.

The successive cancellation list, SCL, introduced in [4] shows significant performance improvement compared to SC decoding. In the SCL decoder, both 0 and 1 are considered as estimated bits and two decoding paths are generated at each decoding stage. The cyclic redundancy check, CRC, is used in [4] to select the correct decoding path in the SCL algorithm. However, SCL decoding has a much higher decoding complexity compared to SC.

The SC-based decoding algorithm, called SC Bit-flipping, using the iterative decoding is introduced in [5], in order to reduce the computation complexity. The BF decoding algorithm contains a standard SC decoding and several additional flipping decoding attempts. The selection of the candidate bits is performed by comparing the absolute value of the LR, and the bit with an index of the smallest $|LR|$ is flipped firstly. In each

bit-flipping decoding attempt, the one-bit SCF decoding flips only one bit, i.e., $w = 1$, in the decoded codeword of the initial SC. Then, the result of bit-flipping decoding is verified by the CRC check until the decoded codeword passes the CRC check or maximum iteration number, T_{max}, is reached. This decoding algorithm provides a gain of error-correcting performance matches to that of SC-List with $L = 2$ with $O(N)$ memory complexity of the original SC algorithm and has an average computational complexity of $O(N\ logN)$ at high SNR.

In this chapter, we propose iterative polar decoding as depicted in Fig.6.3, where a straight decoding operation is performed for the received frame and CRC check is performed, and if it is not satisfied, the first iteration is performed using virtual random channels through which received samples are passed before they are sent to the polar decoders employing successive cancellation decoding algorithm. Randomly generated noise is added to the inputs of the VRCs falling into a threshold interval, which contains unreliable information about the transmitted polar code-bit before they are sent to the polar decoder. For the decoded sequence, if the CRC check is not satisfied, a different randomly generated noise sequence is added to the unreliable inputs and the decoding operation is repeated. This procedure is repeated until a predefined maximum iteration number as long as CRC is not satisfied.

The proposed method is used for additive white Gaussian noise, AWGN, and Rayleigh fading channels. We proposed techniques for the determination of threshold intervals.

It is shown via computer simulations that the proposed technique achieves the performance of the state of art CRC-aided SCL polar decoder, CA-SCL, with much smaller complexity at the practical region of interest.

6.2 Notation and SC Decoding

In this chapter, $u_1^N = (u_1, u_2, \dots, u_N)$ is used to represent N-bit information vector. After polar encoding operation, the code-bit vector $x_1^N = (x_1, x_2, \dots, x_N)$ is obtained and it is

transmitted through a communication channel after digital modulation. At the output of the communication channel the vector $y_1^N = (y_1, y_2, \dots, y_N)$ is observed.

At the destination side, using the received word $r_1^N = (r_1, r_2, \dots, r_N)$, information bits are estimated successively using the likelihood ratios, LRs, of the bits appearing in code structure. AWGN and Rayleigh fading channels are employed for performance evaluation. The log-likelihood ratio (LLR) for $W_n^i\left(r_1^N, \hat{u}_1^{i-1} \middle| u_i\right)$ is defined as

$$L_n^i\left(y_1^N, \hat{u}_1^{i-1} \middle| u_i\right) \triangleq \ln \frac{W_n^i\left(r_1^N, \hat{u}_1^{i-1} \middle| u_i = 0\right)}{W_n^i\left(r_1^N, \hat{u}_1^{i-1} \middle| u_i = 1\right)} \tag{6.1}$$

Decisions are made according to

$$\hat{u}_i = \begin{cases} 0 \;\; if \;\; L_n^i\left(r_1^N, \hat{u}_1^{i-1} \middle| u_i\right) \geq 0 \\ 1 \;\; if \;\; L_n^i\left(r_1^N, \hat{u}_1^{i-1} \middle| u_i\right) < 0. \end{cases} \tag{6.2}$$

The node $LLRs$ are calculated from preceding nodes, having LLRs, denoted by L_1 and L_2. The *LLRs* for f and g nodes are calculated from preceding node *LLRs* as

$$f(L_1, L_2) = sign(L_1) sign(L_2) \min(|L_1|, |L_2|), \tag{6.3}$$

$$g(L_1, L_2, u) = (-1)^u L_1 + L_2 \tag{6.4}$$

6.2.1 AWGN Channel

Let $u_i, x_i \in \{0,1\}\ i = 1, \dots, N$ be the information and polar code bits. The output of the discrete additive white Gaussian noise (AWGN) channel can be written as

$$r_i = y_i + n_o$$

where y_i are obtained by digitally modulating polar code bits x_i, and n_o is a random variable with zero mean and variance σ^2. Using the received word $r_1^N = (r_1, r_2, \dots, r_N)$ at the destination side, information bits u_i are estimated successively. For BPSK modulated code-bits, i.e., $y_i = 2 \times x_i - 1$ and $y_i \in \{-1, 1\}$, the conditional probability density functions, i.e., maximum likelihood probabilities, at the receiver side can be calculated as

$$p(r_i|y_i = \pm 1) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left[-\frac{(r_i \mp 1)^2}{2\sigma^2}\right]. \quad (6.5)$$

For BPSK modulation, likelihood ratio (LR) for y_i is defined as

$$LR(y_i) = \frac{p(r_i|y_i = -1)}{p(r_i|y_i = 1)}$$

whose logarithmic form, i.e., log-likelihood ratio (LLR), is calculated as

$$LLR(r_i) = \ln\left(\frac{p(r_i|y_i = -1)}{p(r_i|y_i = 1)}\right) \rightarrow LLR(r_i) = \frac{2r}{\sigma^2}. \quad (6.6)$$

6.2.2 Rayleigh Fading Channel

The output of the discrete Rayleigh fading is expressed as

$$r_i = h_i y_i + n_o$$

where h_i is a random variable with Rayleigh distribution, y_i and n_o are the same signals as defined in the previous sub-section. Maximum likelihood probability for r_i can be calculated using

$$p(r_i|y_i, h_i) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left[-\frac{(r_i - h_i y_i)^2}{2\sigma^2}\right]. \quad (6.7)$$

Assuming that the Rayleigh channel coefficient h_i are perfectly known at the receiver and BPSK modulation is used, log-likelihood ratio, LLR, for r_i can be calculated as

$$LLR(r_i) = \ln\left(\frac{p(r_i|h_i, y_i = -1)}{p(r_i|h_i, y_i = 1)}\right) \rightarrow LLR(r_i) = \frac{2hr}{\sigma^2}. \quad (6.8)$$

6.3 Threshold Determination

For the determination of threshold [56, 57], we consider two approaches. In the first method, a predefined constant threshold is determined for received signal values

following a mathematical analysis. In the second approach, we propose the calculation of the threshold values using received values. In the second approach, threshold values are dynamic and they may change from frame to frame.

6.3.1 Threshold μ_t determination for AWGN channel

We assume that data bits u_i are encoded, and the obtained polar code bits x_i are BPSK modulated resulting in y_i which are transmitted over the AWGN channel. The frame length is N, and r_i are the received symbols. The conditional probability density function $p(r_i|y_i)$ given by

$$p(r_i|y_i) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left[-\frac{(r_i - y_i)^2}{2\sigma^2}\right]. \tag{6.9}$$

The graphs of $p(r_i|y_i = -1)$ and $p(r_i|y_i = 1)$ are depicted in Fig.6.1. We define the absolute difference function $\delta(r)$ between the two conditional probability density functions as

$$\delta(r) = |p(r|y = 1) - p(r|y = -1)|. \tag{6.10}$$

The received symbol is considered as the least reliable symbol if its value r_i approaches the zero point or $\mu_0(\delta = 0)$. Furthermore, the value of r with maximum difference δ_{max} can be taken as a second point μ_m for which the received symbol is considered as most reliable symbol if its value r_i approaches μ_m. The absolute value of the second point $|\mu_m|$ is the same for both $\mp r_i$ values as the two density functions are symmetric around zero point.

The maximum value of $\delta(r)$, i.e., δ_{max}, can be determined taking the derivative of $\delta(r)$ and equating it to zero as in

$$\delta(r) = \frac{1}{\sqrt{2\pi\sigma^2}}\left[e^{\frac{-(r-1)^2}{2\sigma^2}} - e^{\frac{-(r+1)^2}{2\sigma^2}}\right] \rightarrow \tag{6.11}$$

$$\left.\frac{\mathrm{d}\delta(r)}{dr}\right|_{r=\mu_m} = 0 \rightarrow$$

$$\frac{1}{\sqrt{2\pi\sigma^2}}\left[\frac{1-r}{\sigma^2}e^{-(r-1)^2/2\sigma^2} - \frac{-(1+r)}{\sigma^2}e^{-(r+1)^2/2\sigma^2}\right] = 0. \tag{6.12}$$

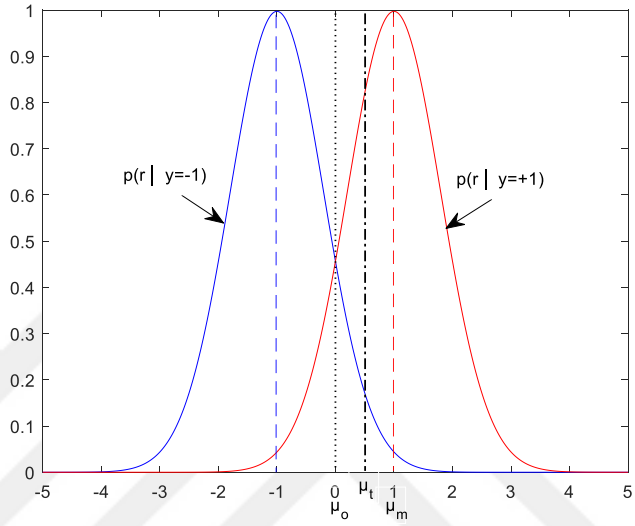

Figure 6.1 Determination of μ_t for AWGN channel.

From 6.12, we obtain

$$(1-\mu_m)e^{\frac{\mu_m}{\sigma^2}} = -(1+\mu_m)e^{\frac{-\mu_m}{\sigma^2}} \tag{6.13}$$

which can be solved numerically by using the Newton Raphson method [57] and for various values of $\sigma^2(0.1 \rightarrow 0.9)$. The value of r at which $\delta(r)$ is maximum is found as $\mu_m \approx 1.04$ which is almost equal to the mean value of $p(r|y=1)$.

6.3.2 Generating Random Noise and Threshold Estimation

The maximum threshold is determined in subsection (6.3.1) which is almost equal to the mean value of $p(r|y=1)$. The threshold level μ_t, can be calculated using

$$\mu_{t(estimated)} = \frac{1}{C_o * N} \times \sum_{i=1}^{N} |r_i| . \tag{6.14}$$

Where the summation part ($\sum_{i=1}^{N}|r_i|/N$) is the arithmetic average of the sum of the absolute values of the received symbols, and this part has a dynamic value, which depends on the SNR and it's approximately equal to 1. The maximum value of the selected threshold μ_t have to be less than μ_m, on the other word, the maximum value has to be in the range of $(-\mu_m < \mu_t < \mu_m)$ where $\mu_m = 1$. By specifying the constant value C_o which is found as the best value as 2, which leads to the optimal threshold value which is approximately $\mu_t \cong 0.5$ as shown in Fig.6.4.

The equation 6.14 takes the form

$$\mu_{t(estimated)} = \frac{1}{2 * N} \times \sum_{i=1}^{N} |r_i| . \tag{6.15}$$

At the output of the VRC, noise is added to the inputs falling into the threshold interval $[-\mu_t \ \ \mu_t]$ and the noise has the Gaussian distribution $\widetilde{N}(0, \sigma^2_{(estimated)})$.

The average received signal power can be calculated as

$$\hat{P}_{s+n_o} = \frac{1}{N}\sum_{i=1}^{N} |r(i)|^2$$

where $\hat{n}_o(i)$ is the estimated noise. Hence, the estimated average noise power can be calculated as

$$\hat{P}_{n_o} = \frac{1}{N}\sum_{i=1}^{N} |\hat{n}_o(i)|^2.$$

Average SNR can be calculated as

$$\widehat{SNR} = \sum_{i=1}^{N} \frac{|r(i)|^2 - |\hat{n}_o(i)|^2}{|\hat{n}_o(i)|^2} = \sum_{i=1}^{N} \frac{|r(i)|^2}{|\hat{n}_o(i)|^2} - 1 . \tag{6.16}$$

From 6.16, the average noise power can be obtained as

$$\hat{P}_{n_o} = \frac{1}{N * \left(\widehat{SNR} + 1\right)} \sum_{i=1}^{N} |r(i)|^2 . \tag{6.17}$$

At the output of the VRC, noise has the Gaussian distribution $\widetilde{N}(0, \sigma^2_{(estimated)})$ is only added to the inputs falling into the threshold interval $[-\mu_t \ \ \mu_t]$, the estimated noise variance has to be less than the channel noise variance $\sigma^2_{(estimated)} < \sigma^2_{(channel)}$, where the additional noise is applied only on the received symbol that falling into the threshold interval. On the other hand, treating the least reliable received symbols will decrease the probability of the transmitted symbol error.

Using the Shannon capacity of AWGN channel for BPSK modulation as $C_{AWGN} = 0.8$, the SNR for this channel capacity can be calculated using

$$C_{AWGN} = \frac{1}{2} log_2(1 + SNR)$$

as 3dB.

Using $\widehat{SNR} = 3dB$, an approximate estimation for noise variance can be written as

$$\sigma^2_{(estimated)} = \frac{1}{3 * N} * \sum_{i=1}^{N} |r_i|^2 . \tag{6.18}$$

In this study, equations 6.15 and 6.18 are used in our simulations.

6.4 Virtual Random Channel

Information bits are polar encoded and transmitted through a continuous channel, such as AWGN or Rayleigh, after digital modulation. At the receiver side, before starting the decoding operation, we consider a virtual random channel (VRC) and pass the received signal through a virtual random channel as illustrated in Fig.6.2. VRC takes the inputs from the continuous channel (AWGN/Rayleigh) and produces its outputs. The operation of a virtual random channel is described in

$$\tilde{r}_i = \begin{cases} r_i + \hat{r}_i, & \text{if } -\mu_t \leq r_i \leq +\mu_t \\ r_i, & otherwise \end{cases} \tag{6.19}$$

where μ_t is the threshold value, r_i are the inputs of the VRC and $\tilde{r}_i$ are the outputs of VRC. $\hat{r}_i$ are the noise samples generated by a normal random variable, i.e., $N(0, \sigma^2)$. Equation 6.19 implies that noise is added to those unreliable inputs falling into a threshold interval.

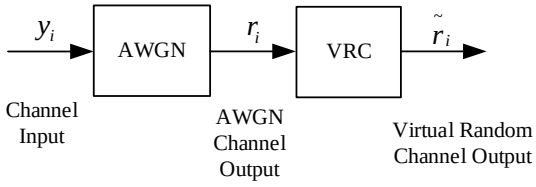

Figure 6.2 AWGN concatenated with VRC.

6.5 The Proposed System

The proposed iterative structure is depicted in Fig.6.3. First, a straight decoding operation is performed for the received frame and a CRC check is performed, and if it is not satisfied, the first iteration is performed. In the next iteration, noise is added to the inputs falling into the threshold interval, and a decoding operation followed by a CRC check is performed. If CRC is not satisfied, iteration is performed. We use a limit, named as maximum iteration number, for the total number of iterations in a decoding stage. If the maximum iteration number is reached, the loop is terminated and the first decoded result before the iteration starts is accepted as the decision of the decoder.

The number of iterations performed for the decoding of each frame is recorded as in Fig.6.5, and the average iteration number considering the total number of frames used for the simulation is calculated. The average iteration number is used to measure the complexity of the iterative structure for comparison to state of art studies.

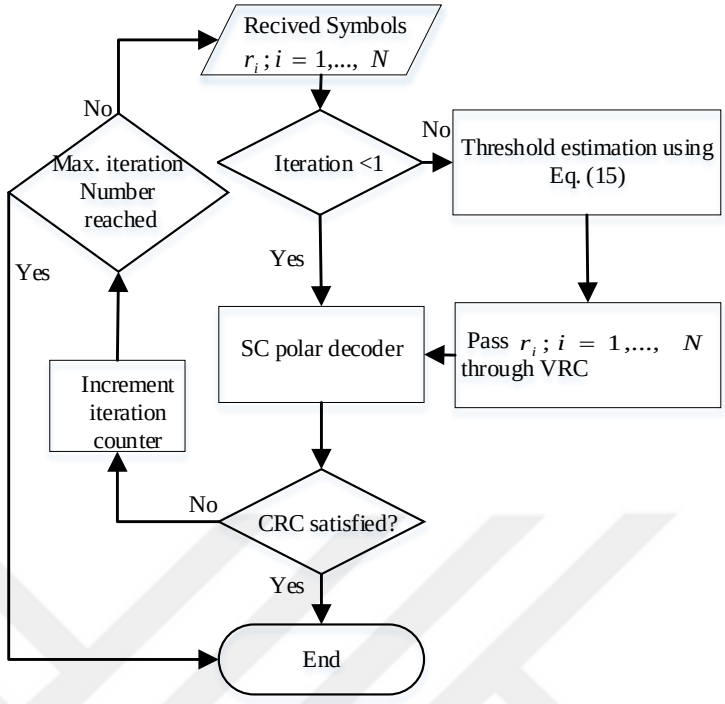

Figure 6.3 Proposed structure.

6.6 Simulation Results

We evaluate the effect of different threshold intervals on the FER performance, for polar code length P(128,64), on an AWGN channel with BPSK modulation at different E_b/N_0 levels. The effect of threshold on the FER performance of proposed algorithm is depicted in Fig. 6.4. Where four threshold intervals are used in simulations using 6.15.

Constant values Co = 4, 3, 2 and 1 which give the threshold intervals frontiers (± 0.25, ± 0.33, ± 0.5, ± 1) are used. One can observe that for any value of E_b/N_0, the performance for the threshold frontiers (± 1) is the worst. It is also seen that the best performance is obtained with threshold interval [-0.5 0.5], with Co = 2, for E_b/N_0= 3dB.

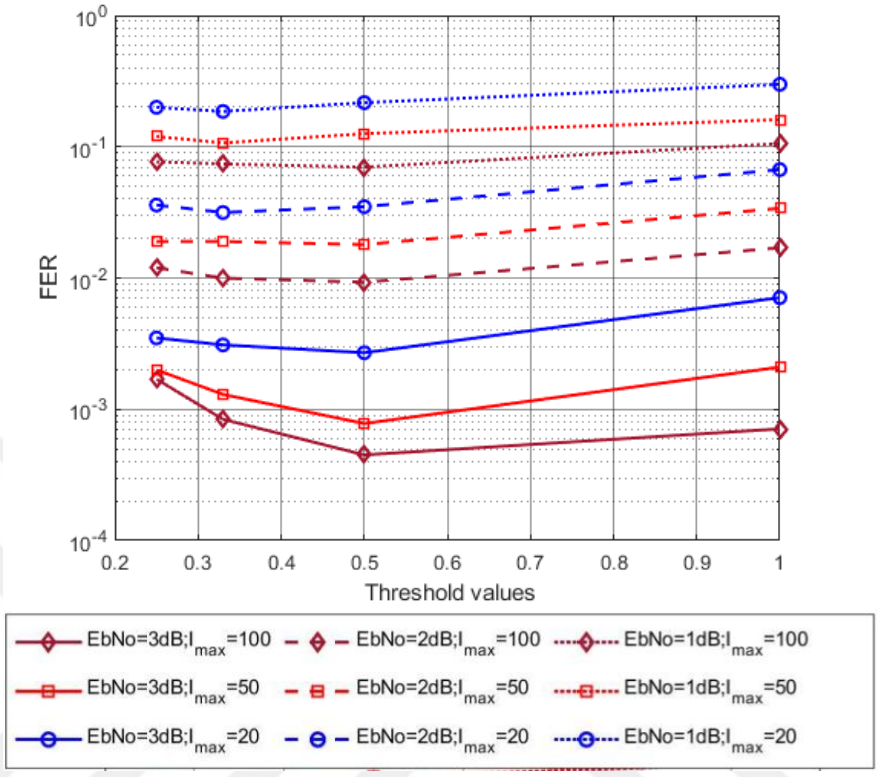

Figure 6.4 The effect of different threshold intervals on the FER performance, for P(128,64), on an AWGN channel

Also, we evaluate the performance of the proposed iterative polar code decoding structure for frame lengths $N = 128$ and 256 for AWGN and Rayleigh channels with code rate $R = 0.5$. For CRC-8, the generator polynomial $g(x) = x^8 + x^7 + x^6 + x^4 + x^2 + 1$ is used.

A set of predefined maximum iteration numbers ($Imax$) is used. We compare our simulation results to that of the state-of-the-art studies CRC-aided successive cancellation list with list size $L = 32$, i.e., compare to CS-SCL32. The average iteration number w.r.t. E_b/N_0 for P(128,64) and P(256,128) over AWGN is depicted in Fig.6.5 where it is seen that the average iteration number decreases as E_b/N_0 increases, and the average complexity of the iterative structure decreases as E_b/N_0 increases.

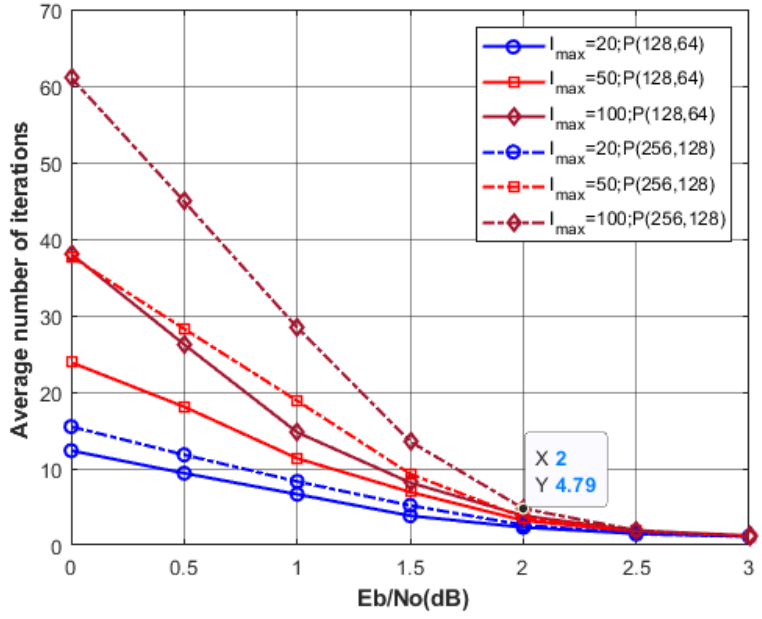

Figure 6.5 Average iteration number versus E_b/N_0 for polar codes P(128,64) and P(256,128) over the AWGN channel using the proposed iterative structure.

In Fig.6.6, the performance of the proposed system is measured for frame length $N = 128$ and P(128, 64). It is seen from Fig.6.6 that as the maximum iteration number increases, the performance of the iterative structure increases as well. And the iterative structure over performs the state of the artwork CA-SCL32 in terms of FER for the maximum iteration number $Imax \geq 50$. If we consider Fig.6.5 and Fig.6.6 together, it is seen that the performance of CA-SCL32 is almost achieved when the largest value of maximum iteration number $Imax = 50$ is used, however, the average iteration number gets its smallest value, i.e., the computational complexity of the iterative structure

decreases. In Fig.6.7, the performance of the proposed structure for frame length $N = 256$, and P(256, 128) is compared to that of the state of the artwork CA-SCL32. It is seen from Fig.6.7 that the performance of the proposed structure increases as the maximum iteration number is increased, the proposed structure over performs the state-of-the-art works CA-SCL32. In Fig.6.6 and Fig.6.7, the performance of the proposed decoding algorithm is also compared with SC flip decoding, for both frame lengths P(128,64) and P(256,128). We observe that the performance of the proposed scheme with $Imax = 20$, has much throughput compared with the one Bit- flipping decoder ($w = 1$) with $T = 42$.

In Fig.6.8 the simulation results for P(128, 64) over Rayleigh fading channels are depicted. It is seen from Fig.6.8 that significant performance is obtained for SC polar decoder for Rayleigh fading channels with the proposed iterative polar decoder.

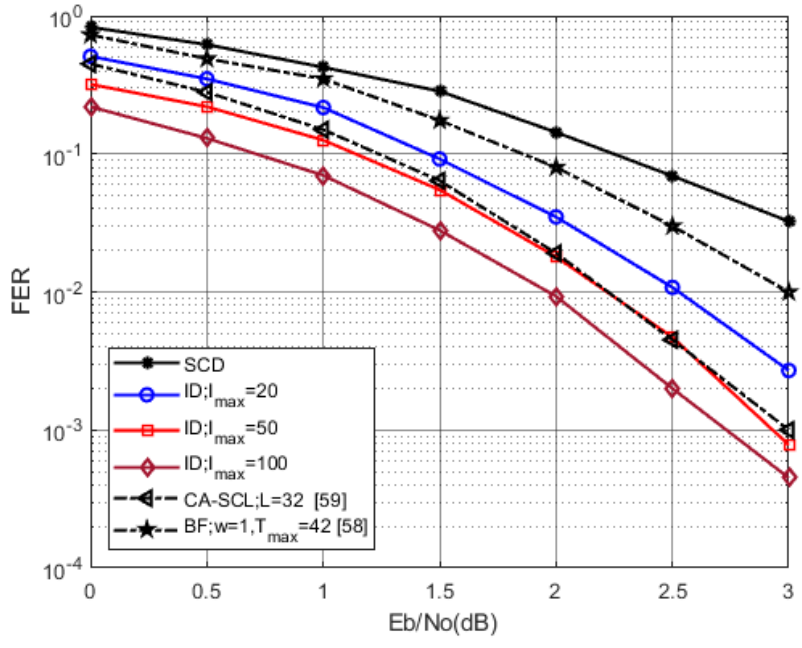

Figure 6.6 Performance improvement for the proposed structure for different maximum iteration numbers for P(128,64) for AWGN channel versus CA-SCL32 performance.

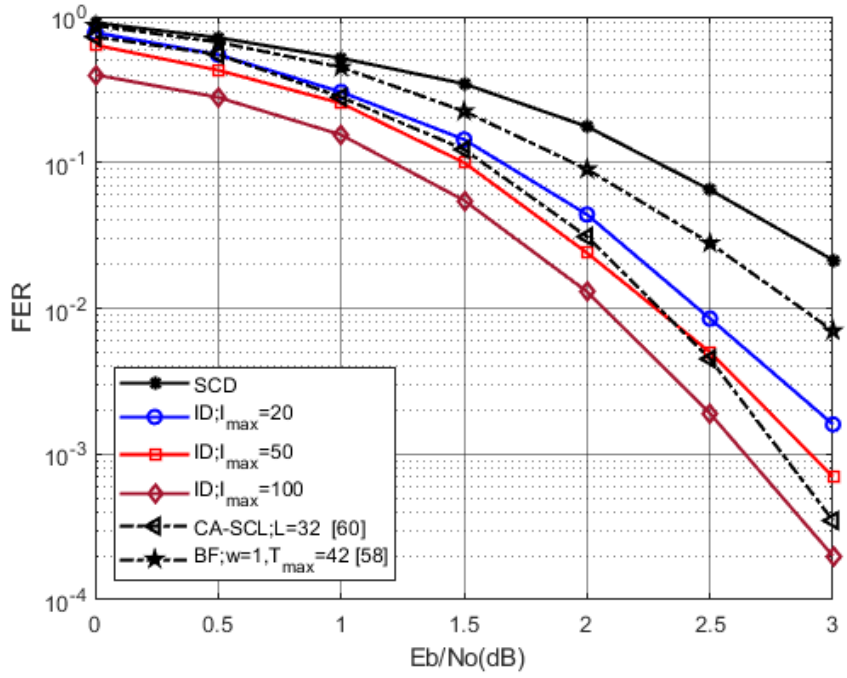

Figure 6.7 Performance improvement for the proposed structure for different maximum iteration numbers for P(256, 128) for AWGN channel versus CA-SCL32 performances.

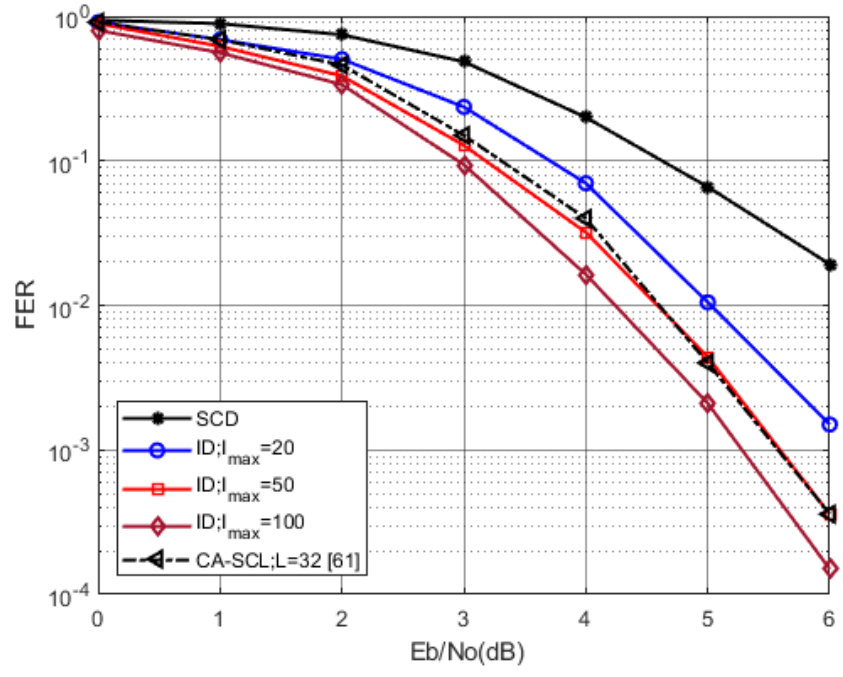

Figure 6.8 Performance improvement for the proposed structure for different maximum iteration numbers for P(128,64) for Rayleigh fading channels.

6.7 Complexity Comparison

For AWGN channels, the SNR region of interest can be accepted as 3dB and beyond. At 3dB, it is seen from Fig.6.5 that the average iteration number is less than 2. This indicates that the complexity of the proposed system can be roughly estimated as 2 times the complexity of the SC polar decoder. On the other hand, for the state-of-the-art works CA-SCL32, the complexity is at least 32 times the complexity of the SC polar decoder. From this discussion, it is seen that the complexity of our proposed structure is less than that of any state-of-the-artwork CA-SCL.

6.8 Conclusion

In this chapter, we introduce the use of virtual random channels for the decoding of polar codes. Virtual random channels add noise to their low reliable inputs. To benefit from the idea of replacing low reliable inputs with high reliable ones, we propose an iterative structure and considered the use of CRC for the termination of the loop structure when successful decoding is detected. It is shown that the proposed structure boosts the performance of the SC polar decoder, and even achieves the performance of state-of-the-art works CA-SCL polar decoders. The complexity of the proposed structure is less than that of the state-of-the-art works CA-SCL. The proposed iterative structure can also be employed to increase the performance of state-of-the-artwork CA-SCL as well.

REFERENCES

[1] **E. Arıkan**, (2009), "*Channel Polarization A Method for Constructing Capacity-Achieving Codes for Symmetric Binary-Input Memoryless Channels*", IEEE Trans. Inf. Theory, 55, (7), pp. 3051–3073.

[2] **N. Presman, O. Shapira, and S. Litsyn**, (2011), *"Polar codes with mixed kernels*", in IEEE International Symposium on Information Theory Proceedings (ISIT). IEEE, pp. 6–10. 29

[3] **V. Miloslavskaya and P. Trifonov**, (2012), *"Design of binary polar codes with arbitrary kernel"*, in Information Theory Workshop (ITW). IEEE, pp. 119–123.

[4] **I. Tal and A. Vardy**, (2015), *"List decoding of polar codes"*, Information Theory, IEEE Transactions on, vol. 61, no. 5, pp. 2213–2226.

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

[6] **K. Niu and K. Chen, "**Stack decoding of polar codes, (2012),*" Electronics Letters*", vol. 48, no. 12, pp. 695–697.

[7] **Z. Zhang, K. Qin, L. Zhang, GT. Chen**, (2018), "*Progressive bit-flipping decoding of polar codes: A critical-set based tree search approach*", IEEE Access, (6), pp.57738-57750.

[8] **Y. Yongrun, P. Zhiwen, L. Nan, Y. Xiaohu**, (2018), "*Successive cancellation list bit-flip decoder for polar codes*", International Conference on Wireless Communications and Signal Processing (WCSP) (10th), pp. 1-6, IEEE.

[9] **H. Sun, J. Gao, L. Li, Z. Ma, P. Fan**, (2021), "*Successive Cancellation List Flipping for Short Polar Codes Based on Row Weights of Generator Matrix*", IEEE Wireless Communications and Networking Conference (WCNC), pp. 1-6.

[10] **E. Furkan, C. Condo, and W.J. Gross**, (2018), "*Improved bit-flipping algorithm for successive cancellation decoding of polar codes*", IEEE Transactions on Communications, 67,(1), pp.61-72.

[11] **C. Carlo, F. Ercan, and W.J. Gross**, (2018), "*Improved successive cancellation flip decoding of polar codes based on error distribution*", IEEE Wireless Communications and Networking Conference Workshops (WCNCW), pp. 19-24.

[12] **E. Furkan, W.J. Gross**, (2020), "*Fast thresholded SC-flip decoding of polar codes*", IEEE International Conference on Communications (ICC), pp. 1-7.

[13] **K. Chen, K. Niu, and J. R. Lin**, (2013), "*Improved successive cancellation decoding of polar codes*", IEEE Trans. Commun., 61, (8), pp. 3100-3107.

[14] **K. Daesung, I. Park**, (2018), "*A Fast Successive Cancellation List Decoder for Polar Codes with an Early Stopping Criterion*", IEEE Transactions on Signal Processing, 66, (18), pp. 4971-4979.

[15] **G. Sarkis, P. Giard, A. Vardy, C. Thibeault, and W. J. Gross**, (2016), "*Fast list Decoders for polar codes*", IEEE J. Sel. Areas Commun., (34), (2), pp. 318–328.

[16] **S. A. Hashemi, C. Condo, and W. J**, (2016), "*Simplified Successive Cancellation List Decoding of Polar Codes*", IEEE International Symposium on Information Theory, pp.815-819.

[17] **B. Li, H. Shen, and D. Tse**, (2012), "*An adaptive successive cancellation list decoder for polar codes with cyclic redundancy check*", IEEE Commun. Lett., (16), 12, pp. 2044-2047.

[18] **G. Sarkis, P. Giard, A. Vardy, C. Thibeault, and W. J. Gross**, (2016), "*Fast list Decoders for polar codes*", IEEE J. Sel. Areas Commun., (34), (2), pp. 318–328.

[19] **C. E. Shannon**, (1948) "*A mathematical theory of communication*", Bell System Tech. J., vol. 27, pp. 379–423, 623–656.

[20] **Hamming, R.W.**, (1950). *"Error detecting and error correcting codes"*, The Bell system technical journal, 29(2), pp.147-160.

[21] **R. C. Bose and D. K. Ray-Chaudhuri**, (Mar.1960), "*On a class of error-correcting binary group codes*", Info. and Control, vol. 3, no. 1, pp. 68–79.

[22] **A. Hocquenghem**, (1959), "*Error correction codes*", vol. 2, pp.147–156.

[23] **I. S. Reed**, (1954) "*A class of multiple-error-correcting codes and the decoding scheme*", IRE Transactions on Inform. Theory, vol. 4, pp. 38–49.

[24] **D. E. Muller**, (1954), "*Application of Boolean algebra to switching circuit design*", IRE Transactions on Electron. Comput., vol. 3, pp. 6–12.

[25] **I. S. Reed and G. Solomon**, (1960), "*Polynomial codes over certain finite fields*", J. SIAM, vol. 8, no. 2, pp. 300–304.

[26] **P. Elias**, (1954), "*Error-free coding*" IEEE Trans. Inform. Theory, vol. 4, pp.29–37.

[27] **G. D. Forney Jr**, (1966), "*Concatenated Codes*", MIT Press.

[28] **P. Elias**, (1955), "*Coding for noisy channels*," in IRE International Convention Record, pp. 37–46.

[29] **A. J. Viterbi**, (1967), "*Error bounds of convolutional codes and an asymptotically optimum decoding algorithm*", IEEE Trans. Inform. Theory, vol. 13, no. 2, pp. 260–269.

[30] **L. Bahl, J. Cocke, F. Jelinek, and J. Raviv**, (1974), "*Optimal decoding of linear codes for minimizing symbol error rate*", IEEE Trans. Inform. Theory, vol. 20, no. 2, pp. 284–287.

[31] **R. M. Fano**, (1963), "*A heuristic discussion of probabilistic decoding*", IEEE Trans. Inform. Theory, vol. 9, no. 2, pp. 64–74.

[32] **R. G. Gallager**, (1963), "*Low-Density Parity-Check Codes*". Cambridge, MA, USA: M.I.T. Press.

[33] **C. Berrou, A. Glavieux, and P. Thitimajshima**, (1993), "*Near Shannon limit error-correcting coding and decoding*", in Proc. of ICC, Geneve, Switzerland, pp. 1064–1070.

[34] **D. J. C. MacKay and R. M. Neal**, (1995), "*Good codes based on very sparse matrices*", in Cryptography and Coding. 5th IMA Conference, ser. Lecture Notes in Computer Science, C. Boyd, Ed. Berlin: Springer, no. 1025, pp. 100–111.

[35] **M. Sipser and D. A. Spielman**, (1996), "*Expander codes*," IEEE Trans. Inform. Theory, vol. 42, no. 6, pp. 1710–1722.

[36] **N. Wiberg, H.-A. Loeliger, and R. K¨otter**, (1995), "*Codes and iterative decoding on general graphs*" European Transactions on Telecommunications, vol. 6, pp. 513–526.

[37] **N. Wiberg**, (1996), "*Codes and decoding on general graphs*," Ph.D. dissertation, Linkoping University, S-581 83, Linkoping, Sweden.

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

[39] **M. Luby, M. Mitzenmacher, A. Shokrollahi, and D. A. Spielman**, (1998), "*Analysis of low density codes and improved designs using irregular graphs*", in Proc. of the 30th Annual ACM Symposium on Theory of Computing, pp. 249–258.

[40] **M.G. Luby, M. Mitzenmacher, M.A. Shokrollahi, D.A. Spielman**, (2001), "*Efficient erasure correcting codes*", IEEE Trans. Inform. Theory, vol. 47, no. 2, pp. 569–584, Bibliography 159

[41] **M.G. Luby, M. Mitzenmacher, M.A. Shokrollahi, D.A. Spielman**, (2001), "*Improved low-density parity-check codes using irregular graphs*", IEEE Trans. Inform. Theory, vol. 47, no. 2, pp. 585–598.

[42] **M. G. Luby, M. Mitzenmacher, and M. A. Shokrollahi**, (1998),"*Analysis of random processes via and-or tree evaluation*", in SODA, pp. 364–373.

[43] **T. Richardson and R. Urbanke**, (2001). "*The capacity of low-density parity-check codes under message-passing decoding*", IEEE Trans. Inform. Theory, vol. 47, no. 2, pp. 599–618.

[44] **S.-Y. Chung, G. D. Forney, Jr., T. Richardson, and R. Urbanke**, (2001) "*On the design of low-density parity-check codes within 0.0045 dB of the Shannon limit*" IEEE Communications Letters, vol. 5, no. 2, pp. 58–60.

[45] **N. Goela, S. B. Korada, and M. Gastpar**, (2010), *"On LP Decoding of Polar Codes "*, submitted to IEEE Trans. Inform. Theory.

[46] **O**. **Gazi,** (2019), "*Polar Codes: A Non-Trivial Approach to Channel Coding*", Springer, ISBN: 978-981-13-0736-2.

[47] **E. Arikan and E. Telatar**, (2009), *"On the rate of channel polarization"*, IEEE International Symposium on Information Theory, pp. 1493\1495.

[48] **O**. **Gazi , A**. **Andi** , (2019), "*Fast Decoding of Polar Codes Using Tree Structure",* IET Communications. doi:10.1049/iet-com.2018.5019.

[49] **C. Leroux, I. Tal, A. Vardy, and W. J. Gross**, (2011), *"Hardware Architectures for Successive Cancellation Decoding of Polar Codes,"* in Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 1665-1668.

[50] **M. P. C. Fossorier, M. Mihaljevic, and H. Imai**, (1999), *"Reduced complexity iterative decoding of low-density parity-check codes based on belief propagation,"* IEEE Trans. on Comm., vol. 47, no. 5, pp. 673 –680.

[51] **B. Yuan and K. K. Parhi**, (2014), *"Successive cancellation list polar decoder using Loglikelihood ratios,"* in Proc. of Asilomar Conf. on Signal, Systems and Computers, pp.548-552.

[52] **A. B. Stimming, M. B. Parizi, and A. Burg**, (2014), *"LLR-based successive cancellation list decoding of polar codes"*, in Proc. of 2014 IEEE Intl. Conf. on Acoustics, Speech and Signal Processing (ICASSP), pp. 3903–3907.

[53] **A. Sutera**, (1980), "*Stochastic perturbation of a pure convective motion*", J. Atmos. Sci., vol. 37, pp. 245–249.

[54] **H. Chen, L. R. Varshney and P. K. Varshney**, (2014), "*Noise-Enhanced Information Systems*", in Proceedings of the IEEE, vol. 102, no. 10, pp. 1607-1621, DOI: 10.1109/JPROC.2014.2341554.

[55] **S. Kay,** (2000), "*Can detectability be improved by adding noise*", IEEE signal processing letters, 7(1), pp8-10.

[56] **C. Senger, V. R. Sidorenko, and V. V. Zyablov**, (2009), '*On Generalized Minimum Distance decoding thresholds for the AWGN channel*', in Proc. XII Symposium Problems of Redundancy in Information and Control Systems, St. Petersburg, Russia, pp. 155–163.

[57] **S. Elsanadily, A. Mahran and O. Elghandour, (**2018), "*Classification-based algorithm for bit-flipping decoding of GLDPC codes over AWGN channels*", IEEE Commun. Lett, (22), 8, pp. 1520-1523.

[58] **O. Afisiadis, A. Balatsoukasstimming, A. Burg**, (2014), "*A low-complexity improved successive cancellation decoder for polar codes*", [C]//2014 48th Asilomar Conference on Signals, Systems and Computers. [S.l.]: IEEE, 2116-2120.

[59] **M. Geiselhart, A. Elkelesh, M. Ebada, and others**, (2020), "*Crc-aided belief propagation list decoding of polar codes*", [C]//2020 IEEE International Symposium on Information Theory (ISIT). [S.l.]: IEEE, pp. 395-400.

[60] **O. Iscan, D. Lentner, W. Xu ,** (2016). "*A Comparison of Channel Coding Schemes for 5G Short Message Transmission*". pp.1-6. 10.1109/GLOCOM.2016.7848804.

[61] **M.C. Liberatori, L. Coppolillo, L.J. Arnone,** (2017), "*List CRC-aided decoding of polar codes over local area networks*", [C]//2017 XVII Workshop on Information Processing and Control (RPIC). [S.l.]: IEEE, pp.1-6.